Dr. Joseph Teguh Santoso, S.Kom, M.Kom

PROYEK CODING dengan PYTHON

YAYASAN PRIMA AGUS TEKNIK

PROYEK CODING dengan PYTHON

Dr. Joseph Teguh Santoso, S.Kom, M.Kom

BIODATA PENULIS

Dr. Joseph Teguh Santoso, S.Kom, M.Kom adalah Rektor dari Universitas Sains & Teknologi Komputer (Universitas STEKOM) Semarang yang memiliki banyak pengalaman praktis dalam bidang *e-commerce*sejak Tahun 2002. Beliau mempunyai 3 (tiga) toko *Official Online Store* di China untuk merek Sepeda Raleigh, dengan omzet tahunan pada Tahun 2019 mencapai lebih dari Rp. 35 Milyar rupiah dan terus meningkat. Dr. Joseph T.S memiliki lisensi tunggal sepeda merek “Raleigh” untuk penjualan *Online* di seluruh China. Di samping itu beliau juga memiliki pabrik sepeda dan sepeda listrik merek “Fengjiu”, yaitu Pabrik Sepeda Listrik yang masih tergolong kecil di China. Pengalaman beliau malang melintang di dunia *online store* di China seperti Alibaba, Tmall, Taobao, JD, Aliexpress sangat membantu mahasiswa untuk memiliki pengalaman teknis dan praktis untuk membuka toko *online* bersama beliau.

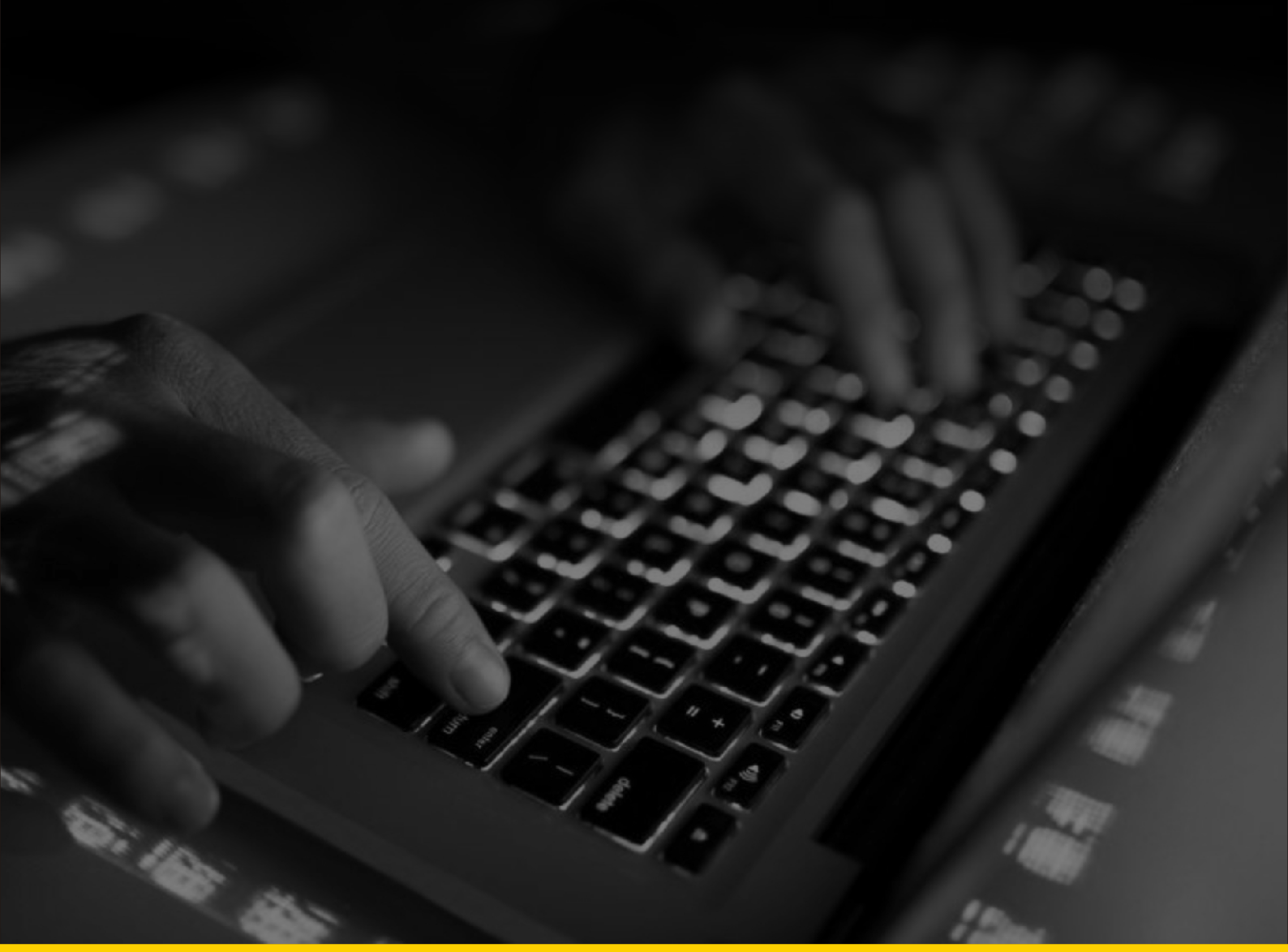
YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-623-5734-31-6 (PDF)
9 786235 734316

PROYEK CODING dengan PYTHON

Dr. Joseph Teguh Santoso, S.Kom, M.Kom



YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Proyek Coding dengan Python

Penulis :
Dr. Joseph Teguh Santoso, S.Kom., M.Kom

ISBN : 978-623-5734-31-6

Editor :
Muhammad Sholikan, M.Kom

Penyunting :
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

Desain Sampul dan Tata Letak :
Irdha Yunianto, S.Ds., M.Kom

Penebit :
Yayasan Prima Agus Teknik Bekerja sama dengan
Universitas Sains & Teknologi Komputer (Universitas STEKOM)

Redaksi :
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Distributor Tunggal :
Universitas STEKOM
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : info@stekom.ac.id

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan karena buku berjudul *"Proyek Coding dengan Python"* dapat terselesaikan dengan baik. Kita hidup di dunia digital, dan komputer adalah bagian dari hampir di semua aspek yang kita lakukan. Komputer adalah mesin besar dan berisik yang sebagian besar diletakkan di meja, tetapi sekarang komputer adalah perangkat kecil yang sunyi yang tersembunyi di dalam ponsel, mobil, TV, dan bahkan jam tangan kita. Manusia menggunakannya untuk bekerja, bermain game, menonton film, berbelanja, dan tetap berhubungan dengan teman dan keluarga.

Komputer saat ini sangat mudah digunakan sehingga siapa pun dapat mengoperasikannya. Tetapi tidak banyak orang yang tahu cara menulis kode yang membuatnya berfungsi. Menjadi seorang pembuat kode memungkinkan Anda untuk melihat di balik kap mesin dan melihat bagaimana komputer benar-benar bekerja. Dengan sedikit latihan, Anda dapat membuat aplikasi Anda sendiri, menulis game Anda sendiri, atau hanya bermain-main dengan program orang lain dan menyesuaikan kreasi cerdas Anda sendiri. Selain menjadi hobi yang membuat ketagihan, *coding* adalah keterampilan yang sangat diminati di seluruh dunia. Pelajari cara membuat kode dan itu akan memberi Anda manfaat yang baik di mana pun hidup Anda mengarah, apakah Anda tertarik pada sains, seni, musik, olahraga, atau bisnis.

Saat ini, ada ratusan bahasa pengkodean yang dapat Anda pelajari, mulai dari bahasa *drag-and-drop* sederhana seperti ScratchTM hingga bahasa pemrograman web seperti JavaScript®. Buku ini didasarkan pada Python®, salah satu bahasa pengkodean yang paling banyak digunakan di dunia. Sama-sama populer di kalangan pelajar dan profesional, Python mudah dipelajari namun kuat dan serbaguna. Ini adalah bahasa yang bagus untuk dipelajari apakah Anda seorang pemula atau naik dari bahasa sederhana seperti Scratch.

Cara terbaik untuk belajar *coding* adalah dengan mendalaminya, dan begitulah cara kerja buku ini. Cukup ikuti langkah-langkah bernomor dan Anda akan membuat aplikasi, game, grafik, dan teka-teki dalam waktu singkat. Jika Anda baru mengenal pemrograman, mulailah dari awal dan selesaikan. Jangan khawatir jika Anda tidak memahami setiap detail.

Semakin banyak proyek yang Anda bangun, semakin baik yang akan Anda dapatkan. Dan jangan khawatir jika program Anda tidak berfungsi saat pertama kali dijalankan. Bahkan para profesional harus men-debug pekerjaan mereka. Setelah Anda selesai membangun setiap proyek, ada tip tentang cara mengubah dan menyesuaikannya. Jangan ragu untuk mencoba peretasan Anda sendiri. Dengan sedikit imajinasi dan keterampilan, tidak ada batasan untuk apa yang dapat dicapai oleh seorang pembuat kode. Akhir kata semoga buku ini bermanfaat bagi para pembaca.

Semarang, Januari 2022
Penulis

Dr. Joseph Teguh Santoso, M. Kom.

DAFTAR ISI

BAB 1
MULAI DENGAN PYTHON

1.1 APA ITU PENGKODEAN?

Pemrogram komputer, atau "pemrogram," adalah orang yang menulis instruksi langkah demi langkah yang dapat membuat komputer melakukan tugas. *Coders* bisa mendapatkan komputer untuk melakukan penambahan, membuat musik, memindahkan robot melintasi ruangan, atau menerbangkan roket ke Mars.

Kotak bodoh

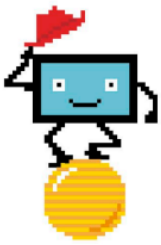
Komputer tidak dapat melakukan apa pun dengan sendirinya—ia hanya duduk di sana seperti kotak bodoh sampai diberi tahu apa yang harus dilakukan. Karena komputer tidak dapat berpikir sendiri dan hanya dapat melakukan apa yang diperintahkan, pembuat kode harus berpikir untuk mereka dan menulis instruksi dengan hati-hati.

1.2 BAHASA PEMROGRAMAN

Untuk memberi tahu komputer apa yang harus dilakukan, Anda perlu mempelajari bahasa pemrograman. Bahasa visual mudah dipelajari oleh pemula, sementara pembuat kode profesional menggunakan bahasa berbasis teks. Buku ini didasarkan pada bahasa populer berbasis teks Python.

Pertunjukan hewan peliharaan

Dengan mempelajari cara membuat kode, Anda akan dapat menulis program Anda sendiri dan membuat komputer melakukan apa yang Anda inginkan. Ini seperti memiliki hewan peliharaan elektronik yang bisa Anda ajarkan untuk melakukan trik!

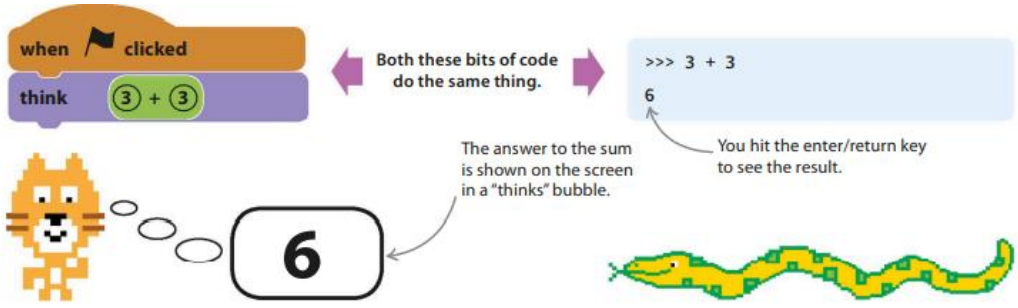


Scratch

Scratch adalah bahasa pemrograman visual. Ini bagus untuk membuat game, animasi, dan cerita interaktif. Anda menulis kode di Scratch dengan menyatukan blok instruksi.

Python

Python adalah bahasa pemrograman berbasis teks. Dalam Python, programmer menulis kode menggunakan kata-kata, singkatan, angka, dan simbol. Instruksi diketik menggunakan keyboard komputer.



Siapa pun dapat membuat kode

Untuk menjadi seorang pembuat kode, Anda hanya perlu mempelajari beberapa aturan dan perintah dasar, dan kemudian Anda dapat mulai menulis program yang sesuai dengan keahlian dan minat Anda. Jika Anda menyukai sains, misalnya, Anda dapat membuat aplikasi yang menggambar grafik dari hasil eksperimen Anda. Atau Anda dapat menggunakan keterampilan seni Anda untuk merancang dunia asing untuk gim video Anda sendiri.

Berpikir logis

Coders perlu berpikir logis dan hati-hati untuk menulis kode yang baik. Jika instruksinya kurang tepat atau urutan langkahnya salah, program tidak akan bekerja dengan benar. Pikirkan setiap langkah dan pastikan hal-hal terjadi dalam urutan yang logis — lagipula, Anda tidak akan mengenakan mantel sebelum sweter Anda, bukan!

Perhatikan detailnya

Jika Anda pandai memecahkan teka-teki perbedaan, Anda mungkin akan menjadi pembuat kode yang hebat. Keterampilan penting dalam pengkodean adalah menemukan kesalahan dalam kode Anda. Kesalahan ini disebut bug, dan bahkan bug kecil dapat menyebabkan masalah besar. Coders bermata elang dapat memilih kesalahan ejaan dan kesalahan dengan logika atau urutan instruksi. Men-debug program bisa jadi rumit, tetapi belajar dari kesalahan Anda adalah cara yang bagus untuk meningkatkan kemampuan pengkodean Anda.

Bug

Bug adalah kesalahan dalam kode yang membuat program berperilaku dengan cara yang tidak terduga. Disebut demikian karena komputer awal terkadang salah ketika serangga terjebak di sirkuit mereka!

Dapatkan pengkodean

Pengkodean mungkin terdengar menakutkan, tetapi mempelajari cara melakukannya itu mudah. Rahasianya adalah langsung masuk. Buku ini dirancang untuk mengajari Anda cara membuat kode dengan membimbing Anda melalui proyek-proyek sederhana. Cukup ikuti langkah-langkah bernomor dan Anda akan membuat game, aplikasi, dan seni digital dalam waktu singkat.

1.3 BERTEMU PYTHON

Python adalah salah satu bahasa pemrograman komputer paling populer di dunia. Ini pertama kali dirilis pada 1990-an dan sekarang digunakan untuk membangun jutaan aplikasi, game, dan situs web.

Mengapa Python?

Python adalah bahasa yang bagus untuk memulai pemrograman komputer. Banyak sekolah dan universitas menggunakannya untuk mengajar coding. Berikut adalah beberapa alasan mengapa Python sangat berguna.

Mudah dibaca dan ditulis

Python adalah bahasa pemrograman komputer berbasis teks. Anda menulis instruksi menggunakan campuran kata-kata bahasa Inggris, karakter tanda baca, simbol, dan angka. Ini membuat kode Python mudah dibaca, ditulis, dan dipahami.

Bekerja di mana-mana

Python bersifat portabel. Ini berarti Anda dapat menulis dan menjalankan kode Python di banyak komputer yang berbeda. Kode Python yang sama akan berfungsi pada PC, Mac, mesin Linux, dan komputer Raspberry Pi. Program berperilaku dengan cara yang sama pada setiap mesin.

Sudah termasuk baterai

Pemrogram mengatakan Python memiliki "baterai termasuk." Ini karena ia dilengkapi dengan semua yang Anda butuhkan untuk segera memulai pengkodean.

Alat praktis

Python dikemas dengan banyak alat yang berguna dan kode terprogram yang dapat Anda gunakan dalam program Anda. Ini disebut Perpustakaan Standar. Menggunakan alat-alat ini membuat Anda lebih mudah dan lebih cepat untuk membangun program Anda sendiri.

Dukungan hebat

Python memiliki dokumentasi yang ditulis dengan baik. Ini memiliki panduan untuk memulai, bagian referensi untuk mencari apa artinya, dan banyak contoh kode.

Python beraksi

Python bukan hanya alat pendidikan. Ini adalah program yang sangat kuat yang digunakan untuk banyak tugas menarik dan mengasyikkan dalam bisnis, kedokteran, sains, dan media. Bahkan dapat digunakan untuk mengontrol lampu dan pemanas di rumah Anda.

Merangkak web

Python banyak digunakan di Internet. Bagian dari mesin pencari Google ditulis dengan Python. Sebagian besar YouTube juga dibuat menggunakan kode Python.

Bisnis serius

Python membantu bank melacak uang di rekening mereka, dan rantai toko besar untuk menetapkan harga barang yang mereka jual.

Keluar dari dunia ini

Insinyur perangkat lunak menggunakan Python untuk membuat alat untuk Pusat Kontrol Misi NASA. Alat-alat ini membantu kru mempersiapkan dan memantau kemajuan setiap misi.

Keajaiban medis

Python dapat digunakan untuk memprogram robot untuk melakukan operasi rumit. Seorang ahli bedah robot yang diprogram Python dapat bekerja lebih cepat daripada manusia, dan lebih akurat dan kecil kemungkinannya untuk membuat kesalahan.

Dalam film

Disney menggunakan Python untuk mengotomatiskan bagian berulang dari proses animasi. Daripada animator melakukan langkah yang sama berulang-ulang, mereka menggunakan program Python untuk mengulangi langkah-langkah tersebut secara

otomatis. Ini menghemat pekerjaan, mempersingkat waktu yang dibutuhkan untuk membuat film.

Penerjemah

Beberapa bahasa pemrograman menggunakan juru bahasa. Interpreter adalah program yang dapat menerjemahkan dari satu bahasa pemrograman ke bahasa lain. Setiap kali Anda menjalankan program Python, interpreter menerjemahkan setiap baris kode Python menjadi kode khusus yang dapat dipahami komputer, yang dikenal sebagai kode mesin.

1.4 MENGINSTAL PYTHON

Semua proyek dalam buku ini menggunakan Python 3, jadi pastikan Anda mengunduh versi yang benar dari situs web. Ikuti petunjuk yang sesuai dengan komputer Anda.

Python di Windows

Sebelum Anda menginstal Python 3 di PC Windows, cari tahu apakah itu menggunakan windows versi 32-bit atau 64-bit. Klik "Mulai", klik kanan "Komputer", dan klik kiri "Properti". Kemudian pilih "Sistem" jika opsi muncul.

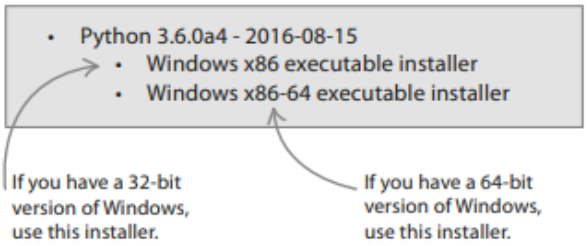
1. Buka situs web Python

   Ketik alamat di bawah ini ke browser web Anda untuk membuka situs web Python. Kemudian klik "Unduhan" untuk membuka halaman unduhan.

   - https://www.python.org/

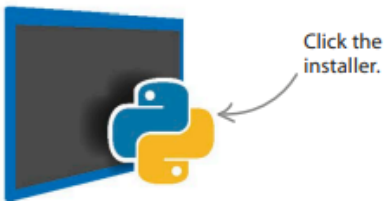
2. Unduh Python

   Klik versi terbaru Python untuk Windows, dimulai dengan angka 3. File penginstal akan diunduh secara otomatis. Dari opsi penginstal yang berbeda, pilih "penginstal yang dapat dieksekusi".



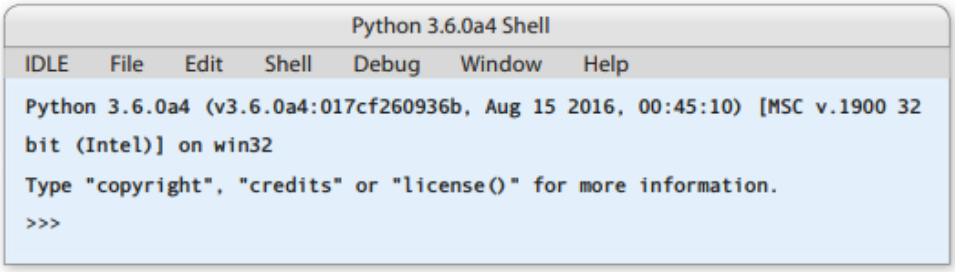
3. Jalankan penginstal

   Klik dua kali file penginstal untuk menginstal Python. Pilih "instal untuk semua pengguna" dan klik "berikutnya" di setiap permintaan, tanpa mengubah pengaturan default.



4. Buka IDLE

   Ketika instalasi selesai, periksa apakah itu berhasil dengan membuka program IDLE. Masuk ke menu “Start”, pilih “All Apps”, lalu pilih “IDLE”. Jendela seperti di bawah ini akan terbuka.



Python di Mac

Sebelum Anda menginstal Python 3 di Mac, periksa sistem operasi yang digunakan komputer. Klik ikon Apple di kiri atas layar dan pilih "Tentang Mac ini" dari menu tarik-turun.

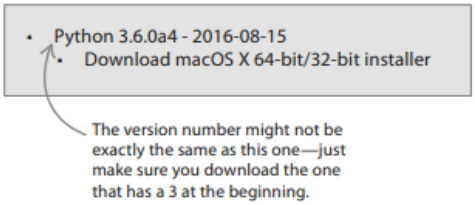
1. Buka situs web Python

   Ketik alamat di bawah ini ke browser web Anda untuk membuka situs web Python. Kemudian klik "Unduhan" untuk membuka halaman unduhan.

   https://www.python.org/

2. Unduh Python

   Dari opsi unduhan, klik versi terbaru Python 3. yang cocok dengan sistem operasi Anda. File Python.pkg akan diunduh ke Mac Anda secara otomatis.

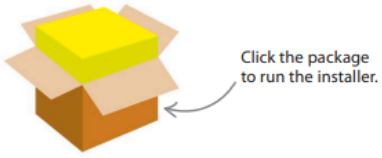


3. Instal Python

   Anda akan menemukan file .pkg di folder "Unduhan". Ikonnya terlihat seperti parsel terbuka. Klik dua kali untuk memulai instalasi. Saat diminta, klik "Lanjutkan" dan kemudian "Instal" untuk menerima pengaturan default.

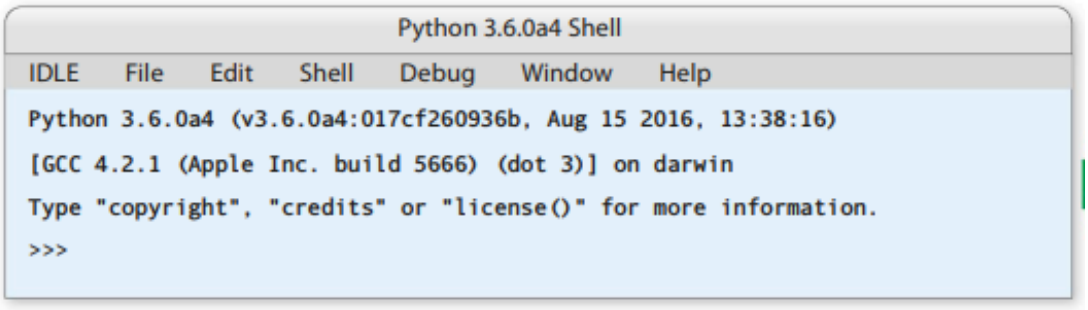
4. Buka IDLE

   Ketika instalasi selesai, periksa apakah itu berhasil dengan membuka program IDLE. Buka folder "Aplikasi", lalu folder "Python". Klik dua kali "IDLE" dan jendela seperti ini akan muncul.



Minta izin

Jangan pernah menginstal Python atau program lain apa pun kecuali Anda memiliki izin untuk melakukannya dari pemilik komputer. Anda mungkin juga perlu meminta pemilik untuk memberikan kata sandi administrasi selama instalasi.



Gambar 1.1 Lisensi Program Python

1.5 MENGGUNAKAN IDLE

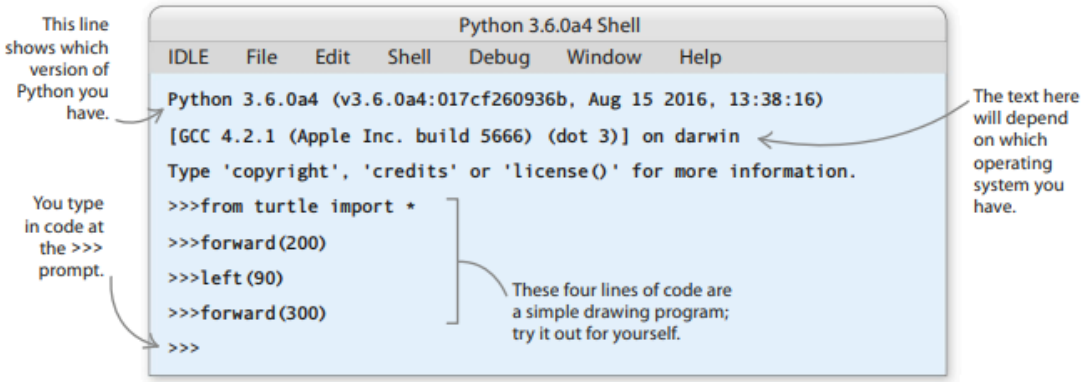
IDLE memiliki dua jendela berbeda tempat Anda dapat bekerja. Jendela editor dapat digunakan untuk menulis dan menyimpan program, sedangkan jendela shell menjalankan instruksi Python dengan segera.

Jendela Cangkang

Saat Anda membuka IDLE, jendela shell akan muncul. Ini adalah tempat terbaik untuk memulai Python karena Anda tidak perlu membuat file baru terlebih dahulu. Cukup ketik kode langsung ke jendela shell.

Bekerja di dalam cangkang

Kode yang Anda ketik dapat langsung dijalankan, dan setiap pesan atau "bug" (kesalahan) akan ditampilkan. Anda dapat menggunakan jendela shell seperti notepad, untuk menguji potongan kode sebelum Anda menambahkannya ke dalam program yang lebih besar.



Gambar 1.2 Keterangan Lisensi Program Python

Berikan shell uji coba

Ketik masing-masing cuplikan kode ini ke dalam jendela shell dan tekan tombol *enter/return* setelah masing-masingnya. Baris pertama menampilkan pesan dan baris

kedua melakukan perhitungan. Bisakah Anda mencari tahu apa yang dilakukan baris ketiga?

```
>>> print('I am 10 years old')

>>> 123 + 456 * 7 / 8

>>> ''.join(reversed('Time to code'))
```

Jendela yang berbeda

Untuk membantu Anda mengetahui di jendela mana Anda harus mengetikkan kode, kami telah memberikan setiap jendela di IDLE warna yang berbeda.

Shell window

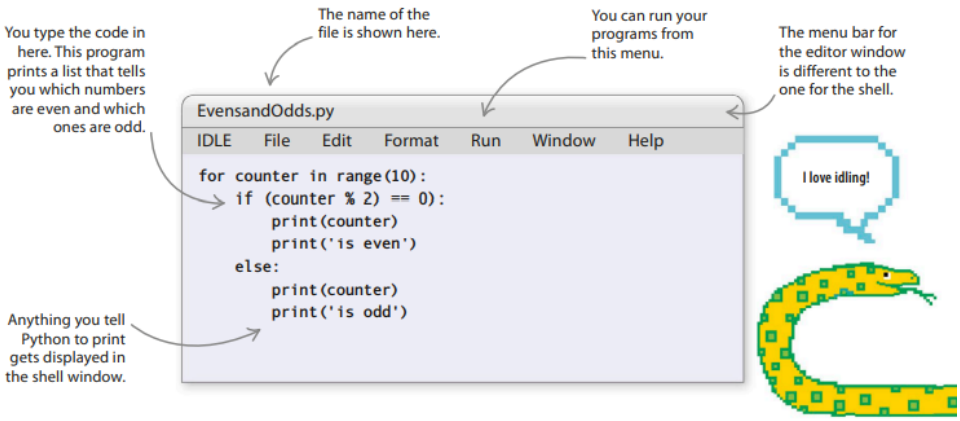
Editor window

Jendela editor

Shell tidak dapat menyimpan kode Anda, jadi ketika Anda menutup jendela Shell, kode yang Anda ketik akan hilang selamanya. Itu sebabnya Anda harus menggunakan jendela editor IDLE saat Anda mengerjakan sebuah proyek. Jendela ini memungkinkan Anda menyimpan kode Anda. Ini juga memiliki alat bawaan untuk membantu Anda menulis program dan untuk memecahkan masalah kesalahan apa pun.

Jendela editor

Untuk membuka jendela editor di IDLE, klik menu File di bagian atas dan pilih File Baru. Jendela editor kosong kemudian akan muncul. Anda akan menggunakan jendela editor untuk menulis dan menjalankan program untuk proyek dalam buku ini.



Gambar 1.3 Jendela Editor

Warna dalam kode

IDLE secara otomatis mewarnai teks untuk menyorot bagian kode yang berbeda. Warna membuatnya lebih mudah untuk memahami kode, dan berguna saat Anda mencoba menemukan kesalahan.

Perintah bawaan
Perintah Python, seperti "cetak", ditampilkan dalam warna ungu.

Simbol dan nama
Sebagian besar teks kode berwarna hitam.

Keluaran
Setiap teks yang dihasilkan saat program berjalan berwarna biru.

Kesalahan
Python menggunakan warna merah untuk memperingatkan Anda tentang kesalahan dalam kode Anda.

Kata Kunci
Kata-kata tertentu, seperti "jika" dan "lain", adalah kata-kata khusus yang digunakan Python. Mereka disebut kata kunci dan ditampilkan dalam warna oranye.

Teks dalam tanda kutip
Teks dalam tanda kutip berwarna hijau. Tanda kurung hijau di sekitar teks menunjukkan bahwa Anda kehilangan tanda kutip.

BAB 2
LANGKAH PERTAMA

2.1 PROGRAM PERTAMA ANDA

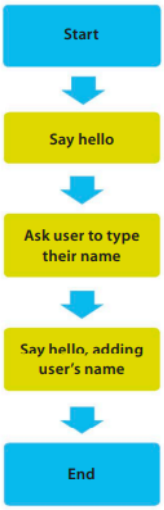
Sekarang Anda telah menginstal Python dan IDLE, saatnya untuk menulis program pertama Anda dengan Python. Ikuti langkah-langkah ini untuk membuat program sederhana yang menyapa pengguna dengan pesan ceria.

Bagaimana itu bekerja

Program pertama kali menampilkan pesan "Halo, Dunia!" lalu menanyakan namamu. Setelah Anda mengetikkan nama Anda, itu menyapa lagi, tapi kali ini menyertakan nama Anda di salam. Program menggunakan sesuatu yang disebut variabel untuk mengingat nama Anda. Sebuah variabel digunakan dalam pengkodean untuk menyimpan informasi.

Diagram alir Halo Dunia

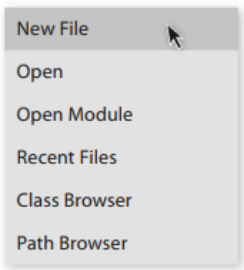
Pemrogram menggunakan diagram yang disebut diagram alur untuk merencanakan program mereka dan untuk menunjukkan cara kerjanya. Setiap langkah ditampilkan dalam kotak, dengan panah mengarah ke langkah berikutnya. Terkadang langkah-langkahnya adalah pertanyaan dan memiliki lebih dari satu panah yang mengarah ke depan, tergantung pada jawaban pertanyaannya.



Gambar 2.1 Diagram Alur / *Flowchart "Hello World"*

1. **Luncurkan IDLE**

   Sebuah jendela shell muncul ketika Anda memulai IDLE. Abaikan dan klik File di menu IDLE. Pilih File Baru untuk membuat jendela editor kosong tempat Anda dapat menulis program.



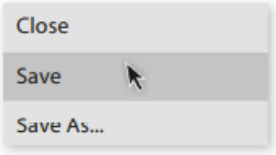
2. **Ketik baris pertama**

Di jendela editor, ketik baris teks ini. Kata "cetak" adalah instruksi Python yang memberitahu komputer untuk menampilkan sesuatu di layar, seperti kata-kata "Halo, Dunia!"

```
print('Hello, World!')
```

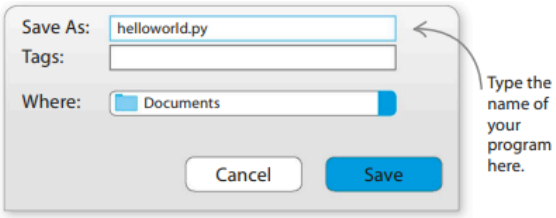
3. **Simpan file Anda**

Sebelum Anda dapat menjalankan kode, Anda harus menyimpannya. Buka menu File dan pilih Simpan.



4. **Simpan file**

Sebuah kotak pop-up akan muncul. Ketikkan nama untuk program Anda, seperti “helloworld.py”, dan klik Simpan.



5. **Periksa berfungsi**

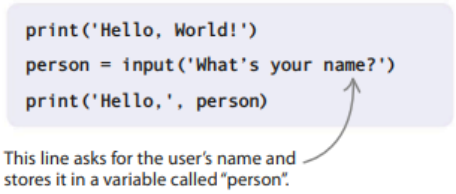
Sekarang jalankan baris pertama program untuk melihat apakah itu berfungsi. Buka menu Run dan pilih Run Module. Anda akan melihat pesan "Halo, Dunia!" di jendela cangkang.

6. **Perbaiki kesalahan**

Jika kode tidak berfungsi, tetap tenang! Setiap programmer membuat kesalahan, dan menemukan "bug" ini sangat penting jika Anda ingin menjadi ahli dalam pengkodean. Kembali dan periksa kode Anda untuk kesalahan pengetikan. Apakah Anda termasuk tanda kurung? Apakah Anda mengeja kata "cetak" dengan benar? Perbaiki kesalahan, lalu coba jalankan kode lagi.

7. **Tambahkan lebih banyak baris**

Kembali ke jendela editor dan tambahkan dua baris lagi ke skrip Anda. Sekarang garis tengah meminta nama Anda dan kemudian menyimpannya dalam sebuah variabel. Baris terakhir menggunakan nama Anda untuk mencetak salam baru. Anda dapat mengubahnya menjadi sapaan yang berbeda jika Anda mau—sesopan atau sekasar yang Anda suka!



8. **Tugas akhir**

   Jalankan kode lagi untuk memeriksanya. Saat Anda mengetikkan nama Anda dan menekan tombol enter/return, shell akan menampilkan pesan yang dipersonalisasi. Selamat telah menyelesaikan program Python pertama Anda! Anda telah mengambil langkah pertama untuk menjadi programmer yang handal.

2.2 VARIABEL

Jika Anda ingin menulis kode yang berguna, Anda harus dapat menyimpan dan memberi label pada bagian informasi. Itulah yang dilakukan variabel. Variabel sangat bagus untuk segala macam hal— mulai dari melacak skor Anda dalam game hingga melakukan perhitungan dan menyimpan daftar item.

Cara membuat variabel

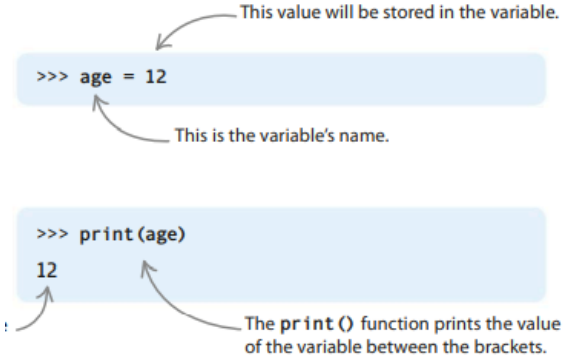
Sebuah variabel membutuhkan nama. Pikirkan nama yang akan mengingatkan Anda apa yang ada di dalam variabel. Kemudian putuskan apa yang ingin Anda simpan dalam variabel. Ini adalah nilai variabel. Ketik nama, diikuti dengan tanda sama dengan, diikuti dengan nilainya. Kami menyebutnya "menetapkan nilai" ke variabel.

1. **Tetapkan nilai**

   Di jendela shell, ketikkan baris kode ini untuk membuat variabel usia dan berikan nilai padanya. Gunakan usia Anda sendiri jika Anda mau.

2. **Cetak nilainya**

   Sekarang ketikkan baris kode yang ditunjukkan di sebelah kanan ke dalam jendela shell. Tekan tombol enter/return untuk melihat apa yang terjadi.



Kotak penyimpanan

Variabel seperti kotak dengan label nama. Anda dapat menyimpan data di dalam kotak dan kemudian menggunakan nama untuk menemukan data lagi ketika Anda perlu menggunakannya.

Penamaan variabel

Memilih nama yang baik untuk variabel Anda akan membuat program Anda lebih mudah dipahami. Misalnya, variabel yang melacak kehidupan pemain dalam game bisa disebut live_remaining, bukan hanya nyawa atau lr. Nama variabel dapat berisi huruf, angka, dan garis bawah, tetapi harus dimulai dengan huruf. Ikuti aturan yang ditampilkan di sini dan Anda tidak akan salah.

Anjuran dan larangan

- Mulai nama variabel dengan huruf.
- Setiap huruf atau angka dapat digunakan dalam nama.
- Simbol seperti -, /, #, atau @ tidak diperbolehkan.
- Spasi tidak dapat digunakan.
- Garis bawah (_) dapat digunakan sebagai pengganti spasi.
- Huruf besar (kapital) dan huruf kecil berbeda. Python akan memperlakukan "Skor" dan "skor" sebagai dua variabel yang berbeda.
- Hindari kata-kata yang digunakan Python sebagai perintah, seperti "cetak".

Bilangan bulat dan float

Dalam pengkodean, bilangan bulat disebut “bilangan bulat”, sedangkan bilangan dengan titik desimal di dalamnya dikenal sebagai “mengambang”. Program biasanya menghitung sesuatu menggunakan bilangan bulat. Pelampung lebih sering digunakan untuk pengukuran.

Menggunakan angka

Variabel dapat digunakan untuk menyimpan angka dan melakukan penjumlahan. Anda dapat menggunakannya dengan simbol untuk melakukan perhitungan, seperti yang Anda lakukan dalam matematika. Beberapa dari simbol ini akan familiar, tetapi hati-hati dengan simbol yang berarti “kalikan” dan “bagi”—simbol ini sedikit berbeda dari yang Anda gunakan di kelas.

1. **Sebuah perhitungan sederhana**

   Ketik kode ini di jendela shell. Ini menggunakan angka yang disimpan dalam dua variabel, bernama x dan y, untuk melakukan perkalian sederhana. Tekan tombol enter/return untuk mendapatkan jawabannya.

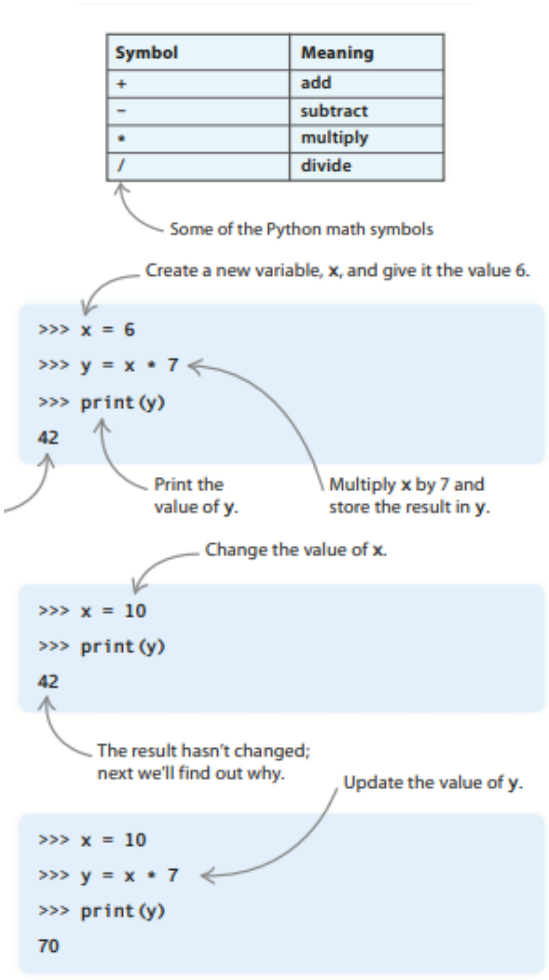
2. **Ubah nilai**

   Untuk mengubah nilai variabel, Anda cukup menetapkan nilai baru padanya. Dalam kode Anda, ubah nilai x menjadi 10 dan jalankan kalkulasi lagi. Apa yang Anda harapkan hasilnya?

3. **Perbarui nilainya**

   Nilai y perlu diperbarui untuk mendapatkan hasil yang benar. Ketik baris ini. Sekarang kode memberikan nilai baru ke y setelah x diubah. Jika Anda memperbarui nilai satu

variabel dalam program Anda sendiri, selalu periksa untuk melihat apakah Anda perlu memperbarui yang lain.



Gambar 2.3 Type-type Variabel

String

Coders menggunakan kata "string" untuk setiap data yang terdiri dari urutan huruf atau karakter lain. Kata dan kalimat disimpan sebagai string. Hampir semua program menggunakan string di beberapa titik. Setiap karakter yang dapat Anda ketik di *keyboard* Anda, dan bahkan yang tidak dapat Anda ketik, dapat disimpan dalam sebuah string.

1. **String dalam variabel**

   String dapat dimasukkan ke dalam variabel. Ketik kode ini ke jendela shell. Ini memberikan string 'Ally Alien' ke nama variabel dan kemudian menampilkannya. String harus selalu memiliki tanda kutip di awal dan akhir.

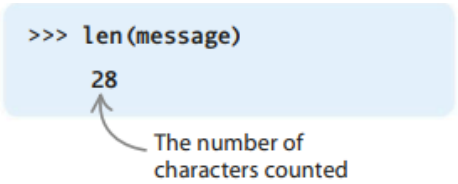
2. **Menggabungkan string**

   Variabel menjadi sangat berguna ketika Anda menggabungkannya untuk membuat variabel baru. Jika Anda menambahkan dua string bersama-sama, Anda dapat menyimpan kombinasi dalam variabel baru. Cobalah ini.

Gambar 2.4 Variabel String

Len()

Anda dapat menggunakan trik praktis, len(), untuk menghitung jumlah karakter dalam string (termasuk spasi). Perintah len() adalah contoh dari apa yang oleh pembuat kode disebut sebagai fungsi. (Anda akan menggunakan banyak fungsi dalam buku ini.) Untuk mengetahui berapa banyak karakter yang ada di 'Welcome to Earth, Ally Alien', ketikkan baris di bawah ini ke dalam shell setelah Anda membuat string, lalu tekan enter/return.



Daftar / list

Bila Anda ingin menyimpan banyak data, atau mungkin urutan datanya penting, Anda mungkin perlu menggunakan daftar. Sebuah daftar dapat menampung banyak item bersama-sama dan menjaganya agar tetap teratur. Python memberi setiap item nomor yang menunjukkan posisinya dalam daftar. Anda dapat mengubah item dalam daftar kapan saja.

1. **Beberapa variabel**

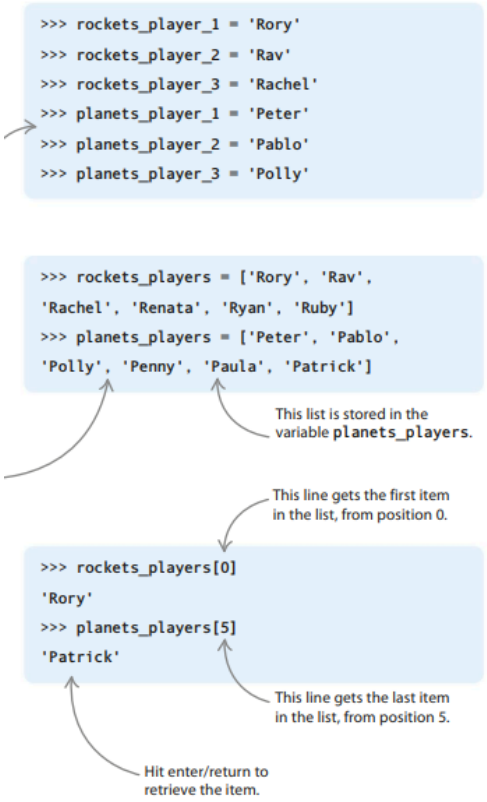
   Bayangkan Anda sedang menulis game multipemain dan ingin menyimpan nama pemain di setiap tim. Anda dapat membuat variabel untuk setiap pemain, yang mungkin terlihat seperti ini...

2. **Masukkan daftar ke dalam variabel**

   ...tapi bagaimana jika ada enam pemain per tim? Mengelola dan memperbarui begitu banyak variabel akan sulit. Akan lebih baik menggunakan daftar. Untuk membuat daftar, Anda mengelilingi item yang ingin Anda simpan dengan tanda kurung siku. Cobalah daftar ini di shell.

3. Mendapatkan item dari daftar

Setelah data Anda ada dalam daftar, mudah untuk digunakan. Untuk mengeluarkan item dari daftar, ketik nama daftar terlebih dahulu. Kemudian tambahkan posisi item dalam daftar, letakkan di dalam tanda kurung siku. Hati-hati: Python mulai menghitung item daftar dari 0 daripada 1. Sekarang coba keluarkan nama pemain yang berbeda dari daftar tim Anda. Pemain pertama berada di posisi 0, sedangkan pemain terakhir berada di posisi 5.



Gambar 2.5 Keterangan item dari Daftar / List

2.3 MEMBUAT KEPUTUSAN

Setiap hari Anda membuat keputusan tentang apa yang harus dilakukan selanjutnya, berdasarkan jawaban atas pertanyaan yang Anda ajukan pada diri sendiri. Misalnya, "Apakah hujan?", "Apakah saya sudah mengerjakan pekerjaan rumah saya?", "Apakah saya seekor kuda?" Komputer juga membuat keputusan dengan mengajukan pertanyaan.

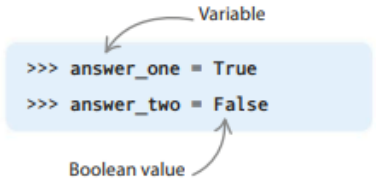
Pertanyaan yang membandingkan

Pertanyaan yang diajukan komputer pada diri mereka sendiri biasanya melibatkan membandingkan satu hal dengan hal lain. Misalnya, komputer mungkin menanyakan apakah satu nomor lebih besar dari yang lain. Jika ya, komputer mungkin memutuskan untuk menjalankan blok kode yang seharusnya dilewati.

Nilai Boolean

Jawaban atas pertanyaan yang diajukan komputer hanya memiliki dua kemungkinan nilai: Benar atau Salah. Python menyebut kedua nilai ini sebagai nilai Boolean, dan

keduanya harus selalu dimulai dengan huruf kapital. Anda dapat menyimpan nilai Boolean dalam sebuah variabel.



Operator logika

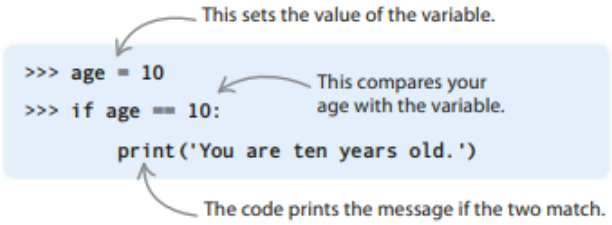
Simbol-simbol ini memberitahu komputer untuk membuat perbandingan. Pemrogram menyebutnya sebagai operator logika. Anda mungkin telah menggunakan beberapa dari mereka dalam matematika. Kata “dan” dan “atau” juga dapat digunakan sebagai operator logika dalam kode komputer.

Tabel 2.1 Operator Simbol Logika

Simbol	Berarti
==	sama dengan
!=	tidak sama dengan
<	kurang dari
>	lebih besar dari

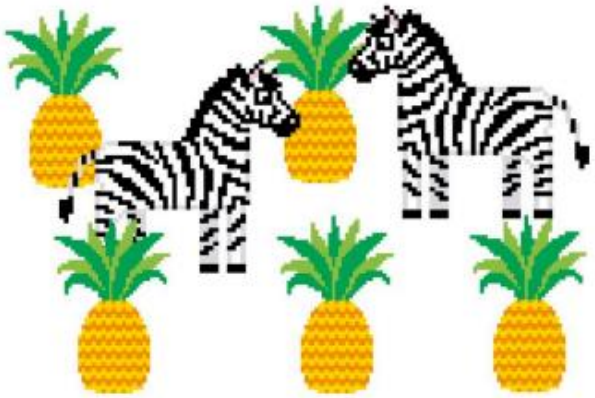
Tanda sama dengan

Dalam Python, Anda dapat menggunakan satu tanda sama dengan, =, atau tanda sama dengan ganda, ==. Mereka berarti hal-hal yang sedikit berbeda. Gunakan satu tanda sama dengan ketika Anda ingin menetapkan nilai variabel. Mengetik usia = 10, misalnya, menyetel nilai variabel usia menjadi 10. Gunakan tanda sama dengan ganda bila Anda ingin membandingkan dua nilai, seperti pada contoh di bawah ini.



Nanas dan zebra

Mari kita coba contoh menggunakan shell. Kita dapat merepresentasikan memiliki lima nanas dan dua zebra dengan menggunakan variabel nanas dan zebra. Ketik garis-garis ini ke dalam shell.



Gambar 2.6 Nanas dan Zebra

```
>>> pineapples = 5
>>> zebras = 2
```

This variable stores the number of pineapples.

This variable stores the number of zebras.

Buat perbandingan

Sekarang coba ketikkan baris kode berikut untuk membandingkan nilai kedua variabel tersebut. Setelah Anda mengetik setiap baris, tekan tombol kembali dan Python akan memberi tahu Anda apakah pernyataan itu Benar atau Salah.

The number of pineapples is greater than the number of zebras.

```
>>> pineapples > zebras
True
```

```
>>> zebras < pineapples
True
```

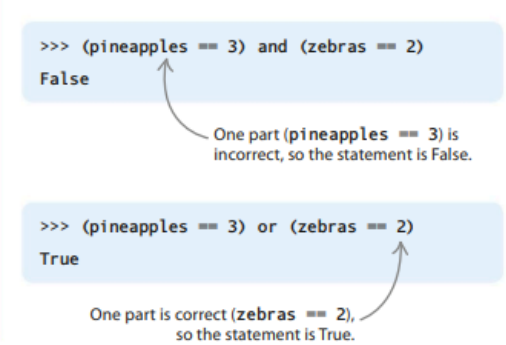
The number of zebras is less than the number of pineapples.

The number of pineapples and the number of zebras aren't equal.

```
>>> pineapples == zebras
False
```

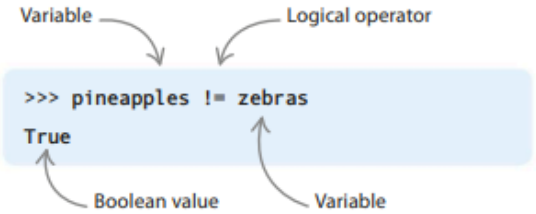
Beberapa perbandingan

Anda dapat menggunakan dan dan atau untuk menggabungkan lebih dari satu perbandingan. Jika Anda menggunakan dan, kedua bagian perbandingan harus benar agar pernyataan menjadi Benar. Jika Anda menggunakan atau, hanya satu bagian yang harus benar.



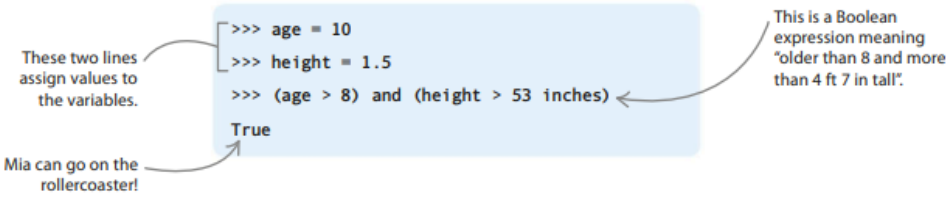
Ekspresi Boolean

Pernyataan tentang variabel dan nilai yang menggunakan operator logika selalu memberikan nilai Boolean, seperti True atau False. Karena itu, pernyataan ini disebut ekspresi Boolean. Semua pernyataan kami tentang nanas dan zebra adalah ekspresi Boolean.



Naik rollercoaster

Sebuah tanda di taman hiburan mengatakan Anda harus berusia di atas 8 tahun dan lebih tinggi dari 4 kaki 7 inci untuk naik rollercoaster. Mia berusia 10 tahun dan tingginya 5 kaki. Mari kita gunakan cangkangnya untuk memeriksa apakah dia bisa pergi jalan-jalan. Ketik baris kode berikut untuk membuat variabel untuk usia dan tinggi Mia dan menetapkan nilai yang benar untuk mereka. Ketik aturan untuk naik rollercoaster sebagai ekspresi Boolean, lalu tekan tombol enter/return.



Percabangan

Komputer sering kali perlu membuat keputusan tentang bagian mana dari program yang akan dijalankan. Ini karena sebagian besar program dirancang untuk melakukan hal yang berbeda dalam situasi yang berbeda. Rute melalui program terbagi seperti jalur yang bercabang menjadi jalur samping, masing-masing mengarah ke tempat yang berbeda.

Sekolah atau taman?

Bayangkan Anda harus memutuskan rute apa yang harus dilalui setiap hari berdasarkan jawaban atas pertanyaan "Apakah hari ini adalah hari kerja?" Jika hari kerja, Anda mengambil rute ke sekolah; jika tidak, Anda mengambil rute ke taman. Dalam Python, rute yang berbeda melalui program mengarah ke blok kode yang

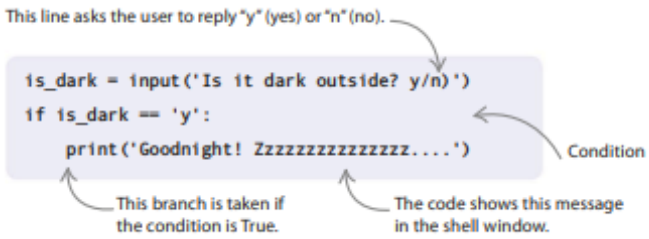
berbeda. Sebuah blok dapat berupa satu atau beberapa pernyataan, semuanya diindentasi oleh empat spasi. Komputer menggunakan tes yang disebut kondisi untuk mengetahui blok mana yang harus dijalankan selanjutnya.

Kondisi

Kondisi adalah ekspresi Boolean (perbandingan Benar-atau-Salah) yang membantu komputer memutuskan rute mana yang harus diambil ketika mencapai cabang dalam kode.

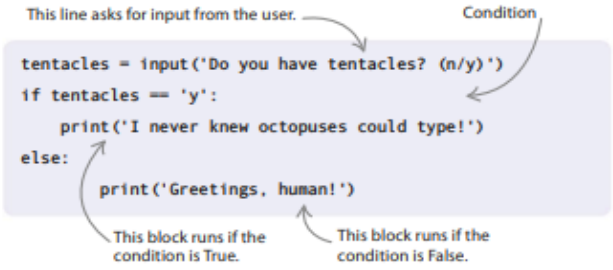
Satu cabang

Perintah percabangan paling sederhana adalah pernyataan if. Ini hanya memiliki satu cabang, yang diambil komputer jika kondisinya Benar. Program ini meminta pengguna untuk mengatakan apakah di luar gelap. Jika ya, program berpura-pura bahwa komputer akan tidur! Jika tidak gelap, is_dark == 'y' adalah False, jadi "Selamat malam!" pesan tidak ditampilkan.



Dua cabang

Apakah Anda ingin program melakukan satu hal jika suatu kondisi Benar dan hal lain jika salah? Jika demikian, Anda memerlukan perintah dengan dua cabang, yang disebut pernyataan if-else. Program ini menanyakan apakah pengguna memiliki tentakel. Jika mereka menjawab "Ya", itu memutuskan bahwa mereka pasti seekor gurita! Jika mereka menjawab "Tidak", itu berarti mereka manusia. Setiap keputusan mencetak pesan yang berbeda.

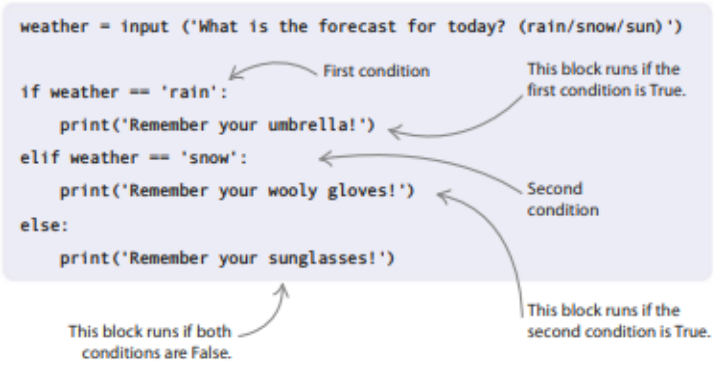


Banyak cabang

Ketika ada lebih dari dua jalur yang mungkin, pernyataan elif (kependekan dari "else-if") akan berguna. Program ini meminta pengguna untuk mengetikkan ramalan cuaca: baik "hujan", "salju", atau "matahari". Kemudian memilih salah satu dari tiga cabang dan kondisi cuaca.

Cara kerjanya

Pernyataan elif harus selalu muncul setelah if dan sebelum else. Dalam kode ini, elif memeriksa salju hanya ketika kondisi yang ditetapkan oleh pernyataan if adalah False. Anda bisa memasukkan pernyataan elif tambahan untuk memeriksa lebih banyak jenis cuaca.

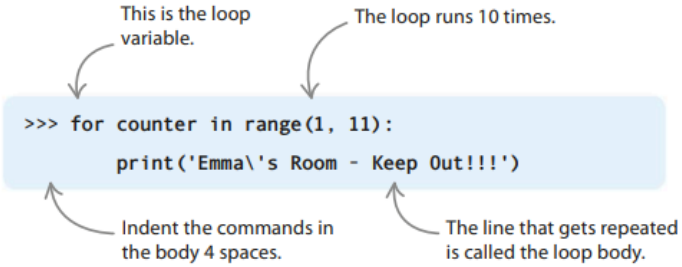


2.4 LOOP

Komputer hebat dalam melakukan tugas-tugas membosankan tanpa mengeluh. Pemrogram tidak, tetapi mereka pandai membuat komputer melakukan pekerjaan berulang untuk mereka — dengan menggunakan loop. Sebuah loop menjalankan blok kode yang sama berulang-ulang. Ada beberapa jenis loop yang berbeda.

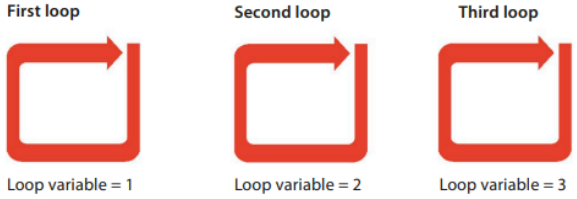
Untuk loop

Ketika Anda tahu berapa kali Anda ingin menjalankan blok kode, Anda dapat menggunakan for loop. Dalam contoh ini, Emma telah menulis sebuah program untuk membuat tanda untuk pintunya. Itu mencetak "Kamar Emma — Jauhkan !!!" sepuluh kali. Cobalah kodenya sendiri di shell. (Setelah mengetik kode dan menekan enter/return, tekan *backspace* untuk menghapus indentasi lalu tekan enter/return lagi.)



Variabel Loop

Variabel loop melacak berapa kali kita telah melewati loop sejauh ini. Putaran pertama sama dengan angka pertama dalam daftar yang ditentukan oleh rentang (1, 11). Kali kedua di sekitar itu sama dengan nomor kedua dalam daftar, dan seterusnya. Ketika kami telah menggunakan semua nomor dalam daftar, kami berhenti mengulang.



Gambar 2.7 Variabel Loop

Rentang *(Range)*

Dalam kode Python, kata "*Range*" diikuti oleh dua angka dalam tanda kurung berarti "semua angka dari angka pertama hingga satu kurang dari angka kedua". Jadi range(1, 4)

berarti angka 1, 2, dan 3—tapi bukan 4. Dalam program “Keep Out” Emma, rentang (1, 11) adalah angka 1, 2, 3, 4, 5, 6, 7, 8, 9, dan 10.

Karakter pelarian (\)

Garis miring terbalik di Emma\'s Room memberitahu Python untuk mengabaikan apostrof sehingga tidak memperlakukannya sebagai tanda kutip yang menutup seluruh string. Garis miring terbalik yang digunakan seperti ini disebut karakter pelarian. Ini memberitahu Python untuk tidak menghitung karakter berikutnya saat bekerja jika baris masuk akal atau mengandung kesalahan.

While

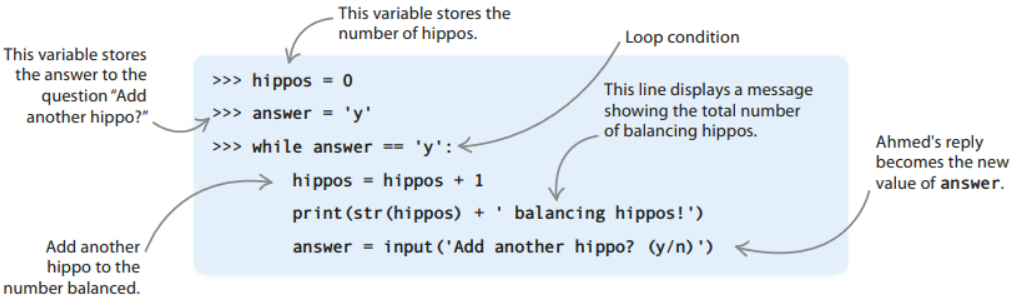
Apa yang terjadi jika Anda tidak tahu berapa kali Anda ingin mengulang kode? Apakah Anda memerlukan bola kristal atau cara lain untuk melihat masa depan? Tidak apa-apa! Anda dapat menggunakan loop sementara.

Kondisi lingkaran

Perulangan while tidak memiliki variabel perulangan yang disetel ke rentang nilai. Sebaliknya ia memiliki kondisi loop. Ini adalah ekspresi Boolean yang dapat berupa True atau False. Ini seperti penjaga di disko yang menanyakan apakah Anda punya tiket. Jika Anda memilikinya (Benar), langsung menuju ke lantai dansa; jika tidak (False), bouncer tidak akan mengizinkan Anda masuk. Dalam pemrograman, jika kondisi loop tidak True, Anda tidak akan masuk ke loop!

Balancing

Dalam contoh ini, Ahmed telah menulis sebuah program untuk melacak berapa banyak rombongan kuda nil akrobatiknya yang telah seimbang di atas satu sama lain untuk membuat sebuah menara. Baca kodenya dan lihat apakah Anda bisa mengetahui cara kerjanya.



Cara kerjanya

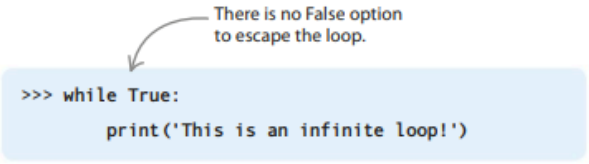
Kondisi loop pada program Ahmed adalah answer == 'y'. Artinya pengguna ingin menambahkan kuda nil. Di badan loop kita menambahkan satu ke jumlah kuda nil yang seimbang, lalu bertanya kepada pengguna apakah mereka ingin menambahkan yang lain. Jika mereka menjawab dengan mengetikkan “y” (untuk ya), kondisi loop adalah True jadi kita memutar loop lagi. Jika mereka menjawab “n” (tidak), kondisi loop adalah False dan program meninggalkan loop.

Loop tak terbatas

Terkadang Anda mungkin ingin loop sementara terus berjalan selama program berjalan. Loop semacam ini disebut loop tak terbatas. Banyak program video-game menggunakan loop tak terbatas yang dikenal sebagai loop utama.

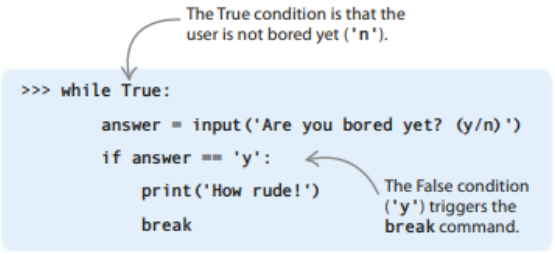
Menuju tak terhingga

Anda membuat infinite loop dengan menyetel kondisi loop ke nilai konstan: True. Karena nilai ini tidak pernah berubah, loop tidak akan pernah keluar. Coba ini while loop di Shell. Ini tidak memiliki opsi False, sehingga loop akan mencetak "Ini adalah loop tak terbatas!" nonstop sampai Anda keluar dari program.



Melarikan diri tanpa batas

Anda dapat dengan sengaja menggunakan infinite loop untuk mendapatkan input dari pengguna. Program (mengganggu) ini menanyakan apakah pengguna bosan. Selama mereka mengetik "n", itu terus mengajukan pertanyaan. Jika mereka muak dan mengetik "y", itu memberi tahu mereka bahwa mereka kasar dan menggunakan perintah break untuk meninggalkan loop!



Menghentikan loop

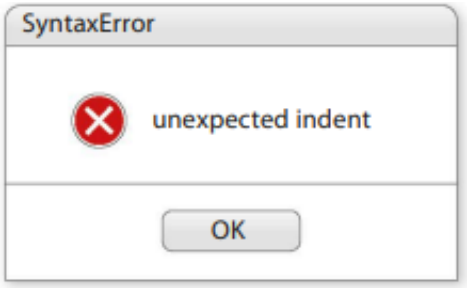
Jika Anda tidak menginginkan perulangan tak terbatas, penting untuk memastikan bahwa badan perulangan while melakukan sesuatu yang dapat membuat kondisi perulangan False. Namun jangan khawatir—jika Anda secara tidak sengaja mengkodekan infinite loop, Anda dapat menghindarinya dengan menekan tombol C sambil menahan tombol Ctrl (kontrol). Anda mungkin harus menekan Ctrl-C beberapa kali sebelum keluar dari loop.

Loop di dalam loop

Bisakah tubuh loop memiliki loop lain di dalamnya? Ya! Ini disebut loop bersarang. Ini seperti boneka Rusia, di mana setiap boneka muat di dalam boneka yang lebih besar. Dalam loop bersarang, loop dalam berjalan di dalam loop luar.

Indentasi tubuh

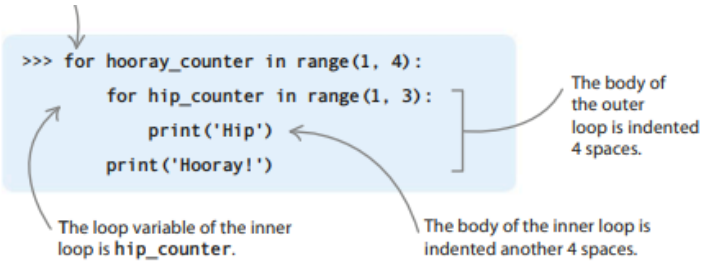
Kode di badan loop harus diindentasi empat spasi. Jika tidak, Python akan menampilkan pesan kesalahan dan kode tidak akan berjalan. Dengan loop bersarang (satu loop di dalam loop lain), badan loop dalam harus diindentasi empat spasi tambahan. Python secara otomatis membuat indentasi baris baru dalam loop, tetapi Anda harus selalu memeriksa bahwa setiap baris diindentasi dengan jumlah spasi yang benar.



Gambar 2.9 *Pop-up error* yang muncul

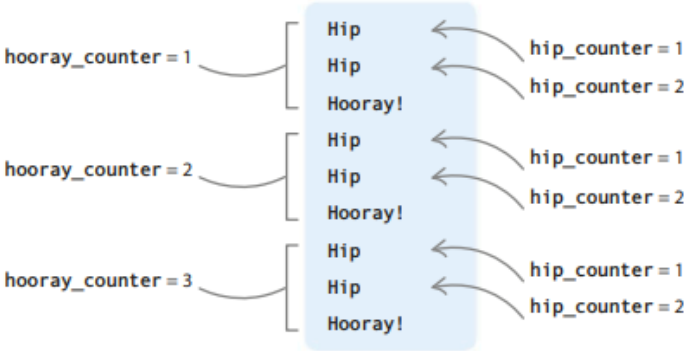
Satu lingkaran di dalam lingkaran lainnya

Dalam contoh ini, Emma telah mengubah program "*Keep Out*" menjadi program "*Three Cheers*" yang menampilkan "Hip, Hip, Hore!" tiga kali. Karena setiap sorakan menyertakan kata "Pinggul" dua kali, dia menggunakan loop bersarang untuk mencetaknya.



Cara kerjanya

Seluruh loop for dalam berada di dalam badan loop for luar. Setiap kali kita melakukan satu pengulangan loop luar, kita harus melakukan dua pengulangan loop dalam. Ini berarti tubuh loop luar dijalankan tiga kali secara total, tetapi tubuh loop dalam dijalankan enam kali.



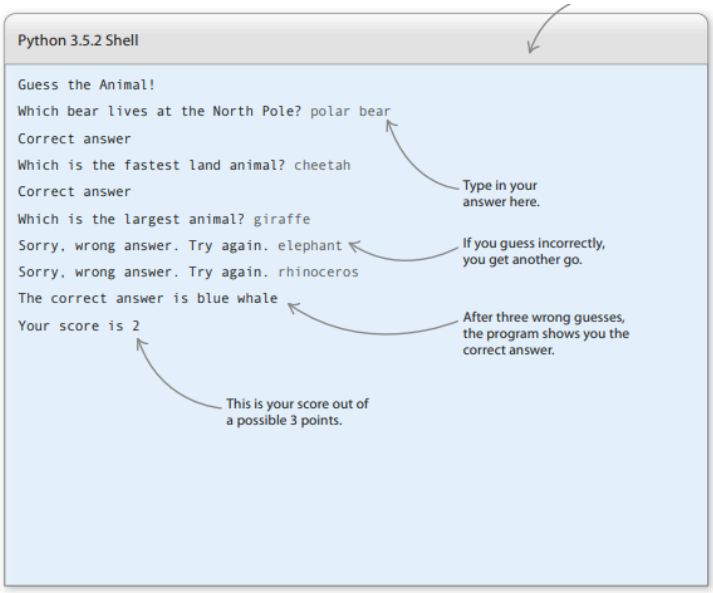
2.5 MEMBUAT KUIS HEWAN

Apakah Anda penggemar kuis? Apakah Anda ingin membuatnya sendiri? Dalam proyek ini, Anda akan membuat kuis hewan. Meskipun pertanyaannya tentang hewan, proyek ini dapat dengan mudah dimodifikasi menjadi topik lain.

Apa yang terjadi

Program ini mengajukan beberapa pertanyaan kepada pemain tentang hewan. Mereka mendapat tiga kesempatan untuk menjawab setiap pertanyaan—Anda tidak ingin

membuat kuis ini terlalu sulit! Setiap jawaban yang benar akan mendapatkan satu poin. Di akhir kuis, program mengungkapkan skor akhir pemain.



Gambar 2.10 Contoh Kuis yang sederhana menggunakan python

Bagaimana itu bekerja

Proyek ini menggunakan fungsi—blok kode dengan nama yang melakukan tugas tertentu. Sebuah fungsi memungkinkan Anda menggunakan kode yang sama berulang kali, tanpa harus mengetik semuanya setiap saat. Python memiliki banyak fungsi bawaan, tetapi juga memungkinkan Anda membuat fungsi sendiri.

Fungsi panggilan

Saat Anda ingin menggunakan suatu fungsi, Anda "memanggilnya" dengan mengetikkan namanya di kode Anda. Di Kuis Hewan, Anda akan membuat fungsi yang membandingkan tebakan pemain dengan jawaban yang benar untuk melihat apakah itu benar. Anda akan menyebutnya untuk setiap pertanyaan dalam kuis.

Abaikan kasusnya!

Saat membandingkan tebakan pemain dengan jawaban yang benar, tidak masalah apakah pemain mengetik huruf kapital atau huruf kecil—yang penting kata-katanya sama. Ini tidak berlaku untuk semua program. Misalnya, jika program yang memeriksa kata sandi mengabaikan huruf besar-kecil, kata sandi mungkin menjadi lebih mudah ditebak, dan kurang aman. Namun, di Kuis Hewan, tidak masalah jika pemain menjawab "beruang" atau "Beruang"—keduanya akan diakui benar.

Bagan alur Kuis Hewan

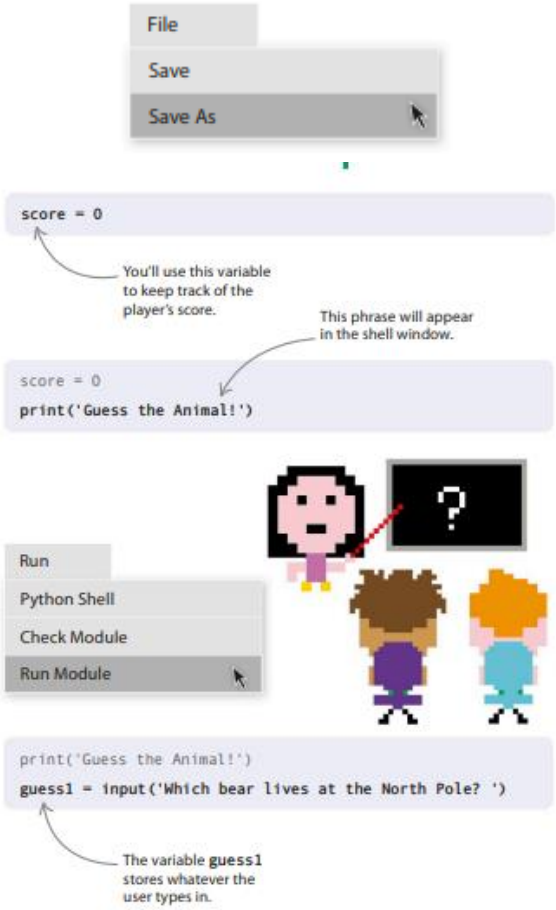
Program terus memeriksa apakah ada pertanyaan yang tersisa untuk ditanyakan dan apakah pemain telah menggunakan semua peluang mereka. Skor disimpan dalam variabel selama pertandingan. Setelah semua pertanyaan terjawab, permainan berakhir.

Gambar 2.11 Flowchart Kuis Hewan

Menyatukannya

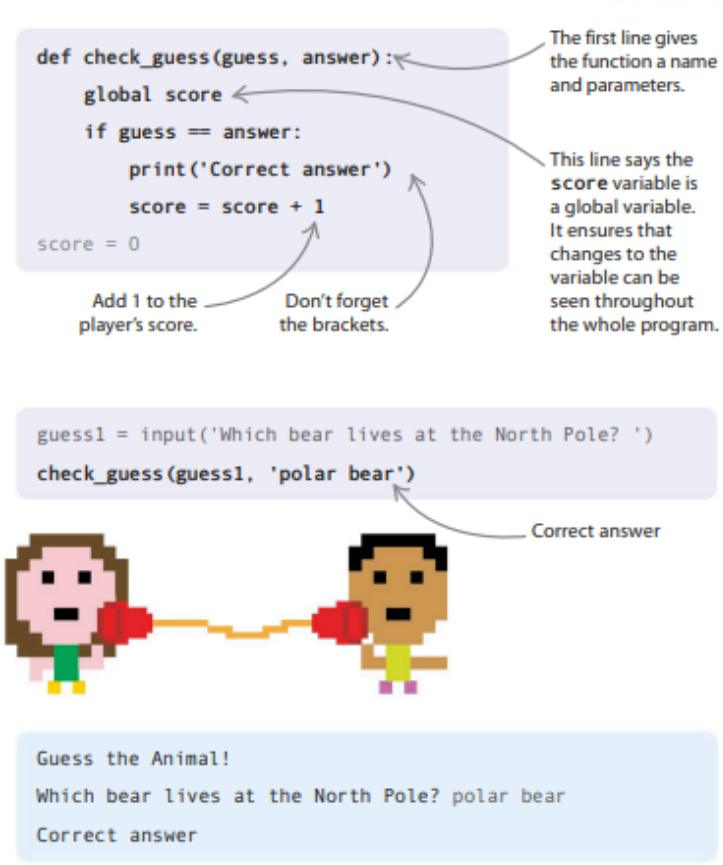
Sekarang saatnya membuat kuis Anda! Pertama, Anda akan membuat pertanyaan dan mekanisme untuk memeriksa jawaban. Kemudian Anda akan menambahkan kode yang memberi pemain tiga upaya untuk menjawab setiap pertanyaan.

1. Buat file baru
   Buka IDLE. Di bawah menu File, pilih File Baru. Simpan file sebagai "animal_quiz.py".
2. Buat variabel skor
   Ketik kode yang ditampilkan di sini untuk membuat variabel yang disebut skor dan atur nilai awalnya ke 0.
3. Perkenalkan permainannya
   Selanjutnya, buat pesan untuk memperkenalkan game kepada pemain. Ini akan menjadi hal pertama yang dilihat pemain di layar.
4. Jalankan kodenya
   Sekarang coba jalankan kodenya. Dari menu Jalankan, pilih Jalankan Modul. Apa yang terjadi selanjutnya? Anda akan melihat pesan selamat datang di jendela shell.
5. Ajukan pertanyaan (masukan pengguna)
   Baris kode berikutnya mengajukan pertanyaan dan menunggu tanggapan pemain. Jawabannya (input pengguna) disimpan dalam variabel tebak1. Jalankan kode untuk memastikan pertanyaan muncul.

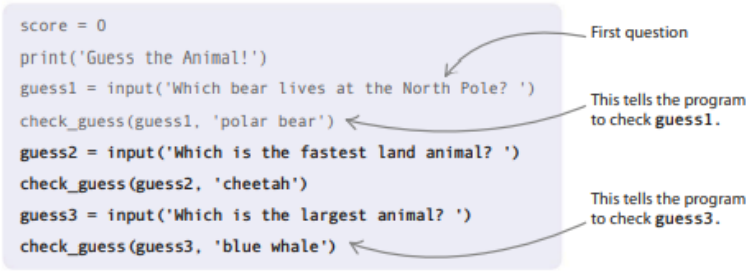


Gambar 2.12 Kuis Sederhana menggunakan Python

6. Bangun fungsi cek
   Tugas selanjutnya adalah memeriksa apakah tebakan pemain benar. Ketik kode ini di bagian atas skrip Anda, sebelum skor = 0. Kode membuat fungsi, yang disebut check_guess(), yang akan memeriksa apakah tebakan pemain cocok dengan jawaban yang benar. Dua kata dalam kurung adalah "parameter"—bit informasi yang dibutuhkan fungsi. Saat Anda memanggil (menjalankan) suatu fungsi, Anda menetapkan (memberi) nilai ke parameternya.
7. Panggil fungsinya
   Sekarang tambahkan baris di akhir skrip untuk memanggil (menjalankan) fungsi check_guess(). Kode ini memberi tahu fungsi untuk menggunakan tebakan pemain sebagai parameter pertama dan frasa "beruang kutub" sebagai parameter kedua.
8. Uji kodenya
   Coba jalankan kode lagi dan ketik jawaban yang benar. Jendela shell akan terlihat seperti ini.
9. Tambahkan beberapa pertanyaan lagi
   Dibutuhkan lebih dari satu pertanyaan untuk membuat kuis! Tambahkan dua pertanyaan lagi ke program, ikuti langkah yang sama seperti sebelumnya. Kami akan menyimpan jawaban pemain dalam variabel tebak2 dan tebak3.



Gambar 2.13 Pernyataan Benar / Salah



10. Tampilkan skor

    Baris kode berikutnya akan mengungkapkan skor pemain dalam pesan saat kuis berakhir. Tambahkan ke bagian bawah file, di bawah pertanyaan terakhir.

```
guess3 = input('Which is the largest animal? ')
check_guess(guess3, 'blue whale')

print('Your score is ' + str(score))
```

Cara kerjanya

Untuk langkah ini, Anda harus menggunakan fungsi str() untuk mengubah angka menjadi string. Ini karena Python menunjukkan kesalahan jika Anda mencoba menambahkan string dan integer (bilangan bulat) secara bersamaan.

11. Abaikan kasus

    Apa yang terjadi jika pemain mengetik "Singa" alih-alih "singa"? Akankah mereka tetap mendapatkan poin? Tidak, kode akan memberi tahu mereka bahwa itu adalah jawaban yang salah! Untuk memperbaikinya, Anda perlu membuat kode Anda lebih cerdas. Python memiliki fungsi yang lebih rendah (), yang mengubah kata menjadi semua karakter huruf kecil. Dalam kode Anda, ganti if guess == answer: dengan baris yang ditunjukkan di sebelah kanan dalam huruf tebal.

12. Uji kode lagi

    Jalankan kode Anda untuk ketiga kalinya. Coba ketik jawaban yang benar menggunakan campuran huruf besar dan huruf kecil dan lihat apa yang terjadi.

```
def check_guess(guess, answer):
    global score
    if guess.lower() == answer.lower():
        print('Correct answer')
        score = score + 1
```

Cara kerjanya

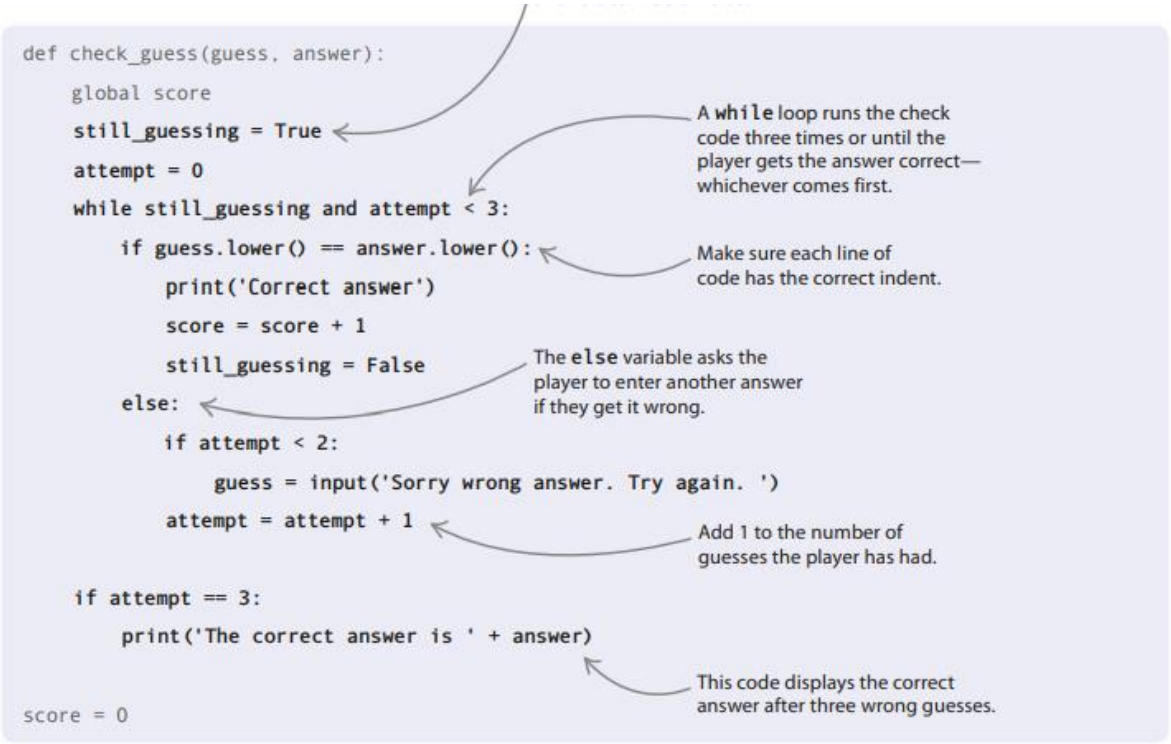
Baik tebakan maupun jawaban akan diubah menjadi karakter huruf kecil sebelum diperiksa. Ini memastikan bahwa kode berfungsi baik pemain menggunakan semua huruf kapital, semua huruf kecil, atau campuran keduanya.

```
Guess the animal!
Which bear lives at the North Pole? polar bear
Correct answer
Which is the fastest land animal? Cheetah
Correct answer
Which is the largest animal? BLUE WHALE
Correct answer
Your score is 3
```

The case is ignored when deciding whether an answer is correct or not.

13. Beri pemain lebih banyak peluang

    Pemain saat ini hanya memiliki satu kesempatan untuk mendapatkan jawaban yang benar. Anda dapat membuatnya sedikit lebih mudah bagi mereka dengan memberi mereka tiga kesempatan untuk menjawab pertanyaan. Ubah fungsi check_guess() menjadi seperti ini.



Cara kerjanya

Untuk mengetahui apakah pemain mendapatkan jawaban yang benar, Anda perlu membuat variabel bernama still_guessing. Anda kemudian mengatur variabel ke True untuk menunjukkan bahwa jawaban yang benar belum ditemukan. Ini disetel ke Salah ketika pemain mendapatkan jawaban yang benar.

Peretasan dan penyesuaian

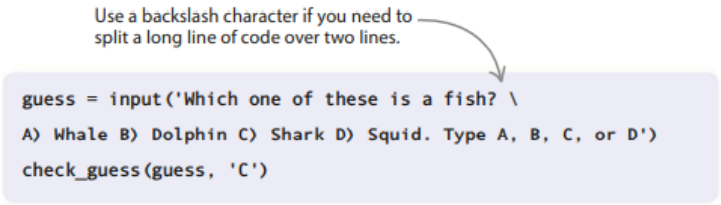
Campurkan kuis Anda! Buat lebih panjang atau lebih sulit, gunakan jenis pertanyaan yang berbeda, atau bahkan ubah topik kuis. Anda dapat mencoba salah satu atau semua peretasan dan penyesuaian ini, tetapi ingat untuk menyimpan masing-masing sebagai file Python terpisah sehingga Anda tidak mengacaukan gim aslinya.

Buat lebih lama

Tambahkan lebih banyak pertanyaan ke kuis. Beberapa contoh dapat berupa “Hewan apa yang memiliki belalai panjang?” (gajah) atau “Mamalia jenis apa yang bisa terbang?” (kelelawar). Atau, sedikit lebih sulit: "Berapa banyak hati yang dimiliki gurita?" (tiga).

Buatlah kuis pilihan ganda

Kode ini menunjukkan cara membuat pertanyaan pilihan ganda, yang memberi pemain beberapa kemungkinan jawaban untuk dipilih.



```
guess = input('Which one of these is a fish?\n \
A) Whale\n B) Dolphin\n C) Shark\n D) Squid\n \
Type A, B, C, or D ')
check_guess(guess, 'C')
```

```
Which one of these is a fish?
 A) Whale
 B) Dolphin
 C) Shark
 D) Squid
Type A, B, C, or D
```

This is how the question appears in the shell window.

Melanggar garis

Anda dapat menggunakan \n untuk membuat baris baru di mana saja. Soal pilihan ganda lebih mudah dipahami jika soal dan kemungkinan jawaban muncul pada baris yang berbeda. Untuk menampilkan pertanyaan ikan sebagai daftar opsi, ketikkan seperti ini.

Skor yang lebih baik untuk upaya yang lebih sedikit

Hadiahi pemain karena mendapatkan jawaban yang benar dengan tebakan yang lebih sedikit. Berikan 3 poin jika mereka mendapatkannya dalam satu percobaan, 2 poin karena membutuhkan dua upaya, dan 1 poin untuk menggunakan ketiga peluang. Buat perubahan ini pada baris yang memperbarui skor. Sekarang itu akan memberikan 3 poin dikurangi jumlah upaya yang gagal. Jika pemain mendapat jawaban yang benar pertama kali, 3 – 0 = 3 poin ditambahkan ke skor mereka; pada tebakan kedua, 3 – 1 = 2 poin; dan pada tebakan ketiga, 3 – 2 = 1 poin.

```
while still_guessing and attempt < 3:
    if guess.lower() == answer.lower():
        print('Correct Answer')
        score = score + 3 - attempt
        still_guessing = False
    else:
        if attempt < 2:
```

This line replaces **score + 1**.

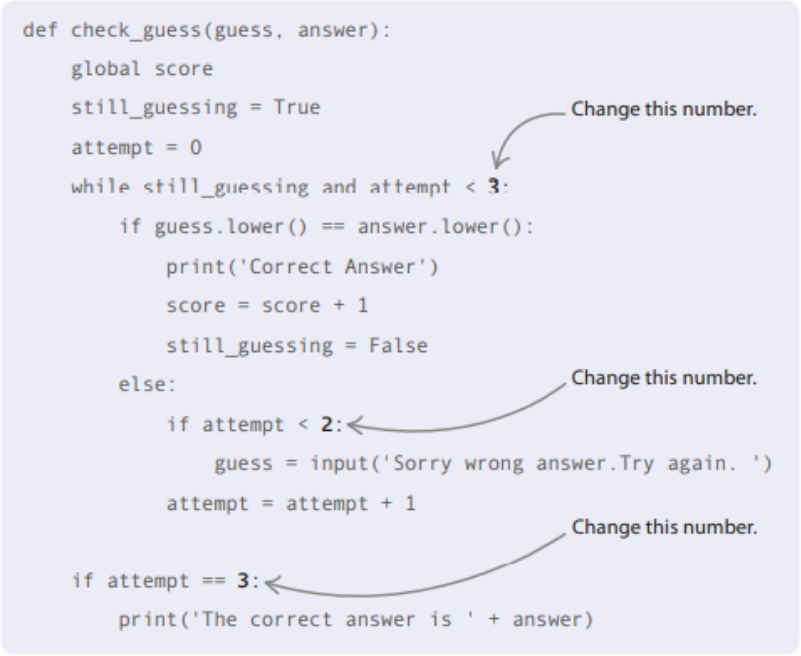
Buat kuis benar-salah

Kode ini menunjukkan cara membuat pertanyaan benar atau salah, yang hanya memiliki dua kemungkinan jawaban.

```
guess = input('Mice are mammals. True or False? ')
check_guess(guess, 'True')
```

Ubah kesulitan

Untuk membuat kuis lebih sulit, beri pemain lebih sedikit kesempatan untuk mendapatkan jawaban yang benar. Jika Anda membuat kuis benar atau salah, Anda hanya ingin pemain memiliki satu tebakan per pertanyaan, dan mungkin tidak lebih dari dua tebakan per pertanyaan jika itu adalah kuis pilihan ganda. Bisakah Anda mencari tahu apa yang Anda perlukan untuk mengubah nomor yang disorot untuk pertanyaan benar-salah atau pilihan ganda?



Pilih topik lain

Buat kuis tentang subjek yang berbeda, seperti pengetahuan umum, olahraga, film, atau musik. Anda bahkan dapat membuat kuis tentang keluarga atau teman Anda dan memasukkan beberapa pertanyaan nakal, seperti “Siapa yang tertawa paling menyebalkan?”

2.6 FUNGSI

Pemrogram menyukai pintasan yang membuat penulisan kode lebih mudah. Salah satu jalan pintas yang paling umum adalah memberi nama pada blok kode yang melakukan pekerjaan yang sangat berguna. Kemudian, daripada harus mengetikkan seluruh blok setiap kali Anda membutuhkannya, Anda cukup mengetikkan namanya. Blok kode bernama ini disebut fungsi.

Cara menggunakan fungsi

Menggunakan fungsi juga dikenal sebagai "memanggil" itu. Untuk memanggil suatu fungsi, Anda cukup mengetikkan nama fungsi, diikuti dengan serangkaian tanda kurung yang berisi parameter apa pun yang Anda inginkan agar fungsi tersebut berfungsi. Parameter agak mirip dengan variabel yang dimiliki oleh fungsi, dan memungkinkan Anda untuk melewatkan data di antara berbagai bagian program Anda. Ketika suatu fungsi tidak memerlukan parameter apa pun, tanda kurung dibiarkan kosong.

Istilah fungsi

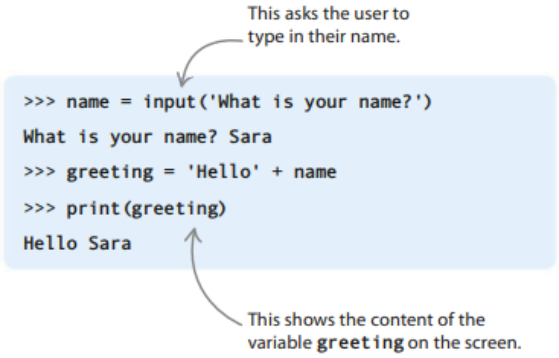
Ada sejumlah kata khusus yang digunakan pembuat kode ketika berbicara tentang fungsi.

- **Panggilan** Untuk menggunakan suatu fungsi.
- **Definisikan** Saat Anda menggunakan kata kunci def dan menulis kode untuk suatu fungsi, pembuat kode mengatakan Anda "mendefinisikan" fungsi tersebut. Kamu juga mendefinisikan variabel ketika Anda pertama kali menetapkan nilainya.
- **Parameter** Sepotong data (informasi) yang Anda berikan ke fungsi untuk digunakan.

- **Mengembalikan nilai** Data yang Anda lewati dari suatu fungsi kembali ke kode utama. Anda mendapatkannya menggunakan kata kunci kembali.

Fungsi bawaan

Python memiliki sejumlah fungsi bawaan yang dapat Anda gunakan dalam kode Anda. Ini adalah alat bantu yang memungkinkan Anda melakukan banyak tugas, mulai dari memasukkan informasi dan menampilkan pesan di layar hingga mengubah satu jenis data menjadi jenis data lainnya. Anda telah menggunakan beberapa fungsi bawaan Python, seperti print() dan input(). Lihatlah contoh-contoh ini. Mengapa tidak mencobanya di shell?

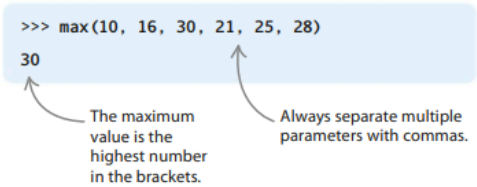


input() dan print()

Kedua fungsi ini seperti berlawanan. Fungsi input() memungkinkan pengguna memberikan instruksi atau data ke program dengan mengetikkannya. Fungsi print() mengirimkan output ke pengguna dengan menampilkan pesan atau hasil di layar.

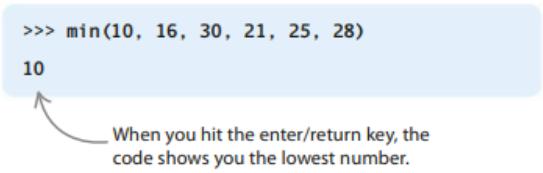
maks()

Fungsi max() memilih nilai maksimum dari parameter yang Anda berikan. Tekan tombol enter/return untuk melihat nilai di layar. Fungsi ini mengambil beberapa parameter, yang harus dipisahkan dengan koma.



min()

Fungsi min() melakukan kebalikan dari max(). Ini memilih nilai minimum dari parameter yang Anda masukkan ke dalam tanda kurung. Bereksperimenlah sendiri dengan fungsi max() dan min().



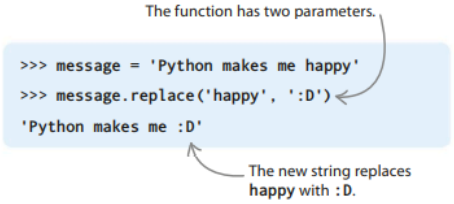
Cara lain untuk memanggil

Beberapa jenis data berbeda yang telah kita temui sejauh ini, seperti bilangan bulat, string, dan daftar, memiliki fungsinya sendiri. Fungsi-fungsi ini harus dipanggil dengan cara khusus. Anda mengetik data atau nama variabel yang menyimpan data,

diikuti dengan titik, nama fungsi, dan terakhir tanda kurung. Uji cuplikan kode ini di shell.

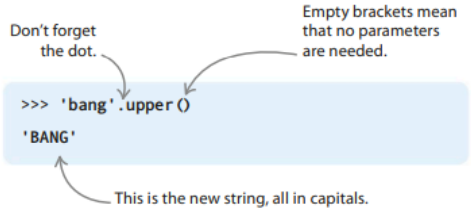
replace()

Dua parameter diperlukan untuk fungsi ini: yang pertama adalah bagian dari string yang ingin Anda ganti, sedangkan yang kedua adalah string yang ingin Anda masukkan ke tempatnya. Fungsi mengembalikan string baru dengan penggantian yang dibuat.



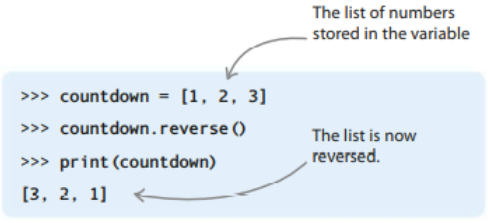
upper()

Fungsi upper() mengambil string yang ada dan mengembalikan string baru di mana semua karakter huruf kecil diubah menjadi huruf besar (kapital).



reverse()

Gunakan fungsi ini saat Anda ingin membalik urutan item dalam daftar. Di sini, ini digunakan untuk membalikkan daftar angka yang disimpan dalam variabel hitung mundur. Alih-alih mencetak daftar sebagai [1, 2, 3], fungsi membuatnya mencetak [3, 2, 1].

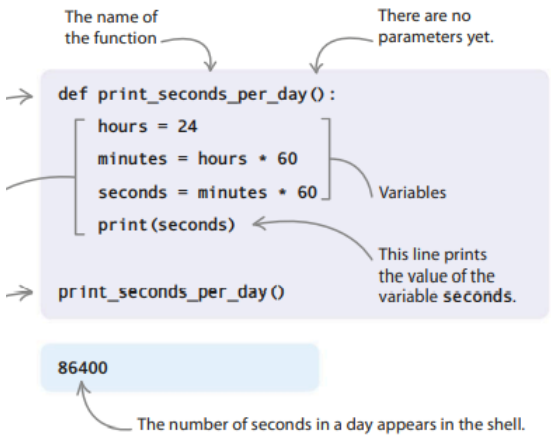


Membuat fungsi

Fungsi terbaik memiliki tujuan yang jelas dan nama bagus yang menjelaskan fungsinya—pikirkan fungsi check_guess() yang Anda gunakan di Kuis Hewan. Ikuti petunjuk ini untuk membuat, atau "menentukan", fungsi yang menghitung jumlah detik dalam sehari dan kemudian mencetak jawabannya di layar.

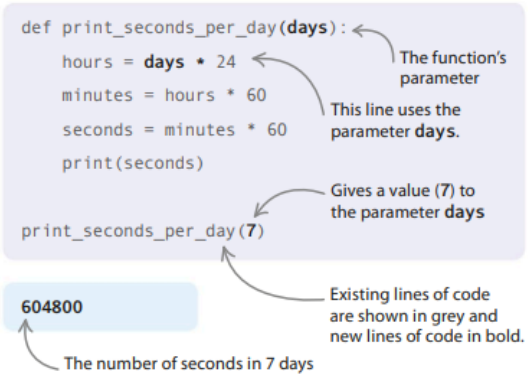
1. Tentukan fungsi

   Buat files baru di IDLE. Simpan sebagai "functions.py". Ketik baris ini ke dalam jendela editor. Indentasi ditambahkan di awal setiap baris dalam fungsi. Simpan file lagi, lalu jalankan kode untuk melihat apa yang terjadi.



2. Tambahkan parameter

   Jika Anda ingin memberi fungsi Anda nilai apa pun untuk dikerjakan, Anda meletakkannya di dalam tanda kurung sebagai parameter. Misalnya, untuk mengetahui jumlah detik dalam jumlah hari tertentu, ubah kode Anda menjadi seperti ini. Fungsi sekarang memiliki parameter hari. Anda dapat menentukan jumlah hari saat Anda memanggil fungsi tersebut. Cobalah sendiri.

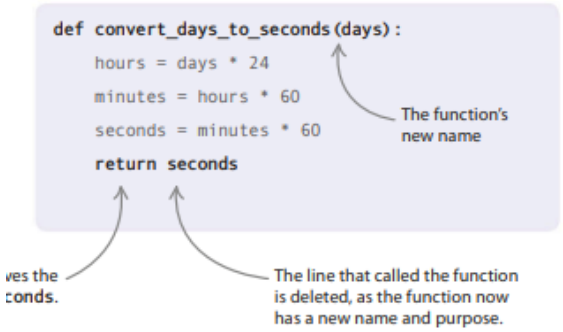


   Saran teratas

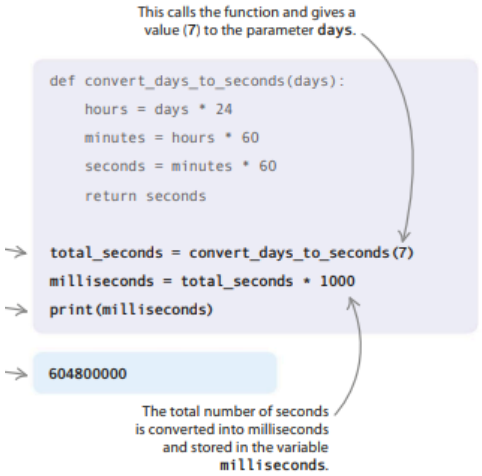
   Penting untuk mendefinisikan fungsi Anda sebelum menggunakannya dalam kode utama Anda. Saat Anda belajar membuat kode dengan Python, akan sangat membantu jika Anda meletakkan fungsi di bagian atas file Anda, setelah pernyataan impor apa pun. Dengan melakukan ini, Anda tidak akan membuat kesalahan dengan mencoba memanggil fungsi sebelum Anda mendefinisikannya.

3. Mengembalikan nilai

   Setelah Anda memiliki fungsi yang melakukan sesuatu yang berguna, Anda akan ingin menggunakan hasil dari fungsi itu di sisa kode Anda. Anda bisa mendapatkan nilai dari suatu fungsi dengan "mengembalikannya". Ubah kode Anda seperti yang ditunjukkan di sini untuk mendapatkan nilai balik dari fungsi Anda. Anda harus mengganti nama fungsi agar sesuai dengan tujuan barunya. Jangan mencoba menjalankan kode dulu.



4. Simpan dan gunakan nilai pengembalian
   Anda dapat menyimpan nilai kembalian dari suatu fungsi dalam variabel untuk digunakan nanti dalam kode Anda. Tambahkan kode ini di bawah fungsi Anda. Ini menyimpan nilai kembalian dan menggunakannya untuk menghitung jumlah milidetik (seperseribu detik). Cobalah dan bereksperimen dengan jumlah hari.



Memberi nama fungsi Anda

Pada Langkah 3, Anda mengubah nama fungsi Anda dari print_seconds_per_day() menjadi convert_days_to_seconds(). Sama seperti dengan variabel, penting bahwa nama yang Anda gunakan secara akurat menjelaskan apa fungsinya. Ini membuat kode Anda lebih mudah dipahami. Aturan penamaan fungsi mirip dengan aturan untuk variabel. Nama fungsi dapat berisi huruf, angka, dan garis bawah, tetapi harus dimulai dengan huruf. Jika ada beberapa kata dalam nama, kata-kata tersebut harus dipisahkan dengan garis bawah.

2.7 MEMPERBAIKI BUG

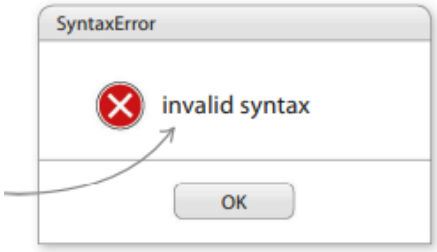
Jika ada yang salah dengan kode Anda, Python akan mencoba membantu dengan menampilkan pesan kesalahan. Pesan-pesan ini mungkin tampak sedikit membingungkan pada awalnya, tetapi mereka akan memberi Anda petunjuk tentang mengapa program Anda tidak berfungsi dan bagaimana cara memperbaikinya.

Pesan kesalahan

Editor IDLE dan jendela shell dapat menampilkan pesan kesalahan jika kesalahan terdeteksi. Pesan kesalahan memberi tahu Anda jenis kesalahan apa yang telah terjadi dan di mana mencarinya di kode Anda.

Pesan di dalam cangkang

Python menampilkan pesan kesalahan dalam teks merah di jendela shell. Program berhenti bekerja ketika pesan kesalahan muncul. Pesan tersebut memberi tahu Anda baris kode mana yang menyebabkan kesalahan terjadi.



Gambar 2.14 Jendela pop-up Error

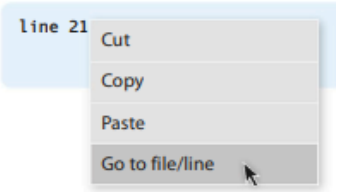
Pesan di editor IDLE

Kotak pop-up memperingatkan Anda bahwa ada kesalahan. Klik OK untuk kembali ke program Anda. Akan ada sorotan merah pada atau di dekat kesalahan.

```
>>>
Traceback (most recent call last):
    File "Users/Craig/Developments/top-secret-python-book/age.py", line 21, in module>
        print('I am'+ age + 'years old')
TypeError: Can't convert 'int' object to str implicitly
```

Menemukan bug

Ketika pesan kesalahan muncul di shell, klik kanan dan pilih "*Go to file/line*" pada menu *drop-down*. Editor IDLE melompat langsung ke baris kode itu sehingga Anda dapat mulai men-debug.



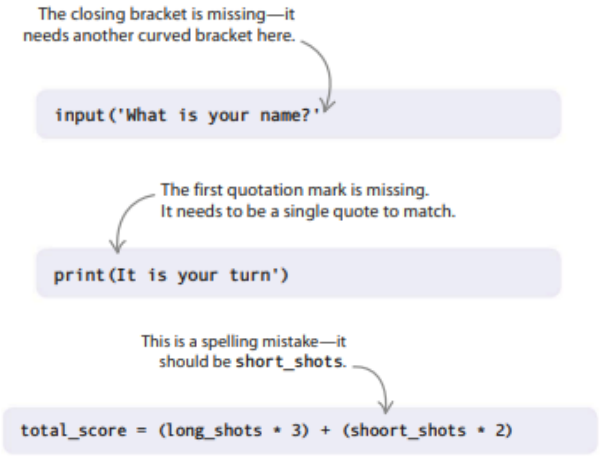
Gambar 2.15 *Pop-up Bug* yang muncul

Kesalahan sintaks

Saat Anda mendapatkan pesan kesalahan sintaks, itu adalah petunjuk bahwa Anda salah mengetik. Mungkin jari Anda terpeleset dan salah ketik? Jangan khawatir— ini adalah kesalahan termudah untuk diperbaiki. Periksa kode Anda dengan cermat dan coba temukan apa yang salah.

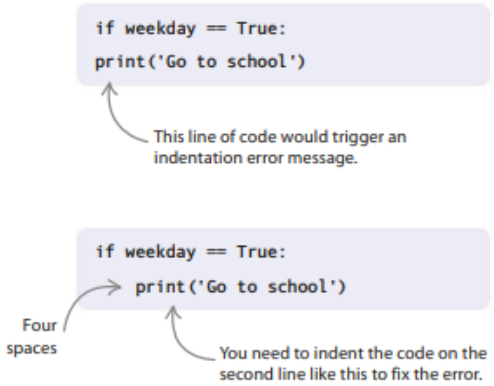
Hal-hal yang harus diperhatikan

Apakah Anda melewatkan tanda kurung atau tanda kutip? Apakah pasangan tanda kurung dan tanda kutip Anda cocok? Apakah Anda membuat kesalahan ejaan? Semua hal ini dapat menyebabkan kesalahan sintaks.



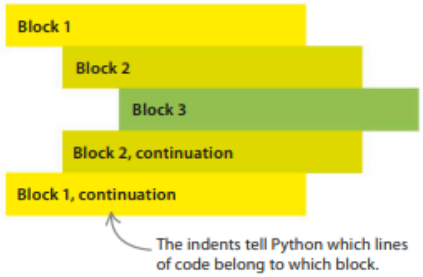
Kesalahan indentasi

Python menggunakan lekukan untuk memahami di mana blok kode mulai dan berhenti. Kesalahan lekukan berarti ada yang salah dengan cara Anda menyusun kode. Ingat: jika baris kode diakhiri dengan titik dua (:), baris berikutnya harus menjorok. Tekan bilah spasi empat kali untuk membuat indentasi garis secara manual.



Indentasi setiap blok baru

Dalam program Python Anda, Anda akan sering memiliki satu blok kode di dalam blok lain, seperti loop yang berada di dalam suatu fungsi. Setiap baris dalam blok tertentu harus diindentasi dengan jumlah yang sama. Meskipun Python membantu dengan membuat indentasi otomatis setelah titik dua, Anda masih perlu memeriksa bahwa setiap blok diindentasi dengan benar.



Gambar 2.16 Blok kode pemrograman Python

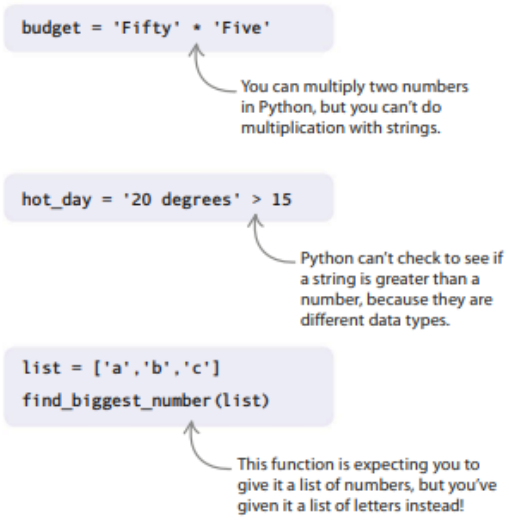
Kesalahan ketik

Kesalahan ketik bukanlah kesalahan pengetikan—ini berarti kode Anda telah mencampuradukkan satu jenis data dengan yang lain, seperti membingungkan angka dengan

string. Ini seperti mencoba memanggang kue di lemari es Anda — itu tidak akan berhasil, karena lemari es tidak dimaksudkan untuk memanggang! Jika Anda meminta Python untuk melakukan sesuatu yang mustahil, jangan kaget jika itu tidak mau bekerja sama!

Contoh kesalahan ketik

Kesalahan ketik terjadi ketika Anda meminta Python untuk melakukan sesuatu yang tidak masuk akal, seperti mengalikan dengan string, membandingkan dua jenis data yang sama sekali berbeda, atau menyuruhnya menemukan angka dalam daftar huruf.

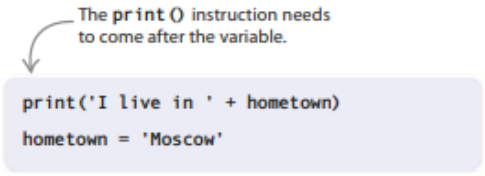


Kesalahan nama

Pesan kesalahan nama muncul jika kode Anda menggunakan nama variabel atau fungsi yang belum dibuat. Untuk menghindari hal ini, selalu tentukan variabel dan fungsi Anda sebelum Anda menulis kode untuk menggunakannya. Ini adalah praktik yang baik untuk mendefinisikan semua fungsi Anda di bagian atas program Anda.

Kesalahan nama

Kesalahan nama dalam kode ini menghentikan Python menampilkan pesan "Saya tinggal di Moskow". Anda harus membuat variabel kampung halaman terlebih dahulu, sebelum Anda menggunakan fungsi print().

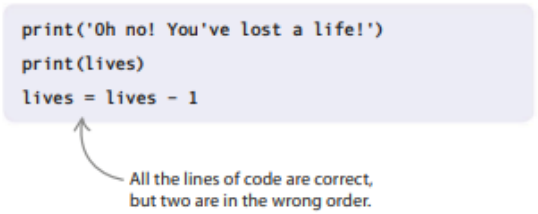


Kesalahan logika

Terkadang Anda dapat mengetahui ada yang tidak beres meskipun Python tidak memberi Anda pesan kesalahan, karena program Anda tidak melakukan apa yang Anda harapkan. Bisa jadi Anda memiliki kesalahan logika. Anda mungkin telah mengetikkan kode dengan benar, tetapi jika Anda melewatkan baris penting atau memasukkan instruksi dalam urutan yang salah, itu tidak akan berjalan dengan benar.

Dapatkah Anda menemukan bug?

Kode ini akan berjalan tanpa pesan kesalahan, tetapi ada kesalahan logika di dalamnya. Nilai nyawa ditampilkan di layar sebelum jumlah nyawa dikurangi satu. Pemain game ini akan melihat jumlah nyawa yang tersisa salah! Untuk memperbaikinya, pindahkan print instruksi (hidup) ke akhir.



Baris demi baris

Kesalahan logika mungkin sulit ditemukan, tetapi seiring bertambahnya pengalaman, Anda akan mahir melacaknya. Cobalah untuk mengidentifikasi kesalahan logika dengan memeriksa kode Anda secara perlahan, baris demi baris. Bersabarlah dan luangkan waktu Anda—Anda akan menemukan masalahnya pada akhirnya.

Daftar periksa penghilang bug

Terkadang Anda mungkin berpikir bahwa Anda tidak akan pernah mendapatkan program untuk bekerja, tetapi jangan menyerah! Jika Anda mengikuti tip dalam daftar periksa yang praktis ini, Anda akan dapat mengidentifikasi sebagian besar kesalahan.

Bertanya pada diri sendiri...

- Jika Anda membuat salah satu proyek dalam buku ini dan tidak berhasil, periksa apakah kode yang Anda ketikkan sama persis dengan buku tersebut.
- Apakah semuanya dieja dengan benar?
- Apakah Anda memiliki spasi yang tidak perlu di awal baris?
- Apakah Anda bingung dengan angka untuk huruf, seperti 0 dan O?
- Sudahkah Anda menggunakan huruf besar dan huruf kecil di tempat yang benar?
- Apakah semua kurung buka memiliki kurung tutup yang cocok? ( ) [ ] {}
- Apakah semua kutipan tunggal dan ganda memiliki kutipan penutup yang cocok? '' ""
- Pernahkah Anda meminta orang lain untuk mencocokkan kode Anda dengan buku?
- Sudahkah Anda menyimpan kode Anda sejak terakhir kali Anda melakukan perubahan?

2.8 MEMILIH KATA SANDI

Kata sandi menghentikan orang lain mengakses komputer, email pribadi, dan detail login situs web kami. Dalam proyek ini, Anda akan membuat alat yang membuat kata sandi yang aman dan mudah diingat untuk membantu menjaga keamanan informasi pribadi Anda.

Kiat kata sandi

Kata sandi yang baik mudah diingat tetapi sulit ditebak oleh seseorang atau pembobol kata sandi.

Gambar 2.17 Kata Kunci untuk *user* yang berbeda

Apa yang terjadi

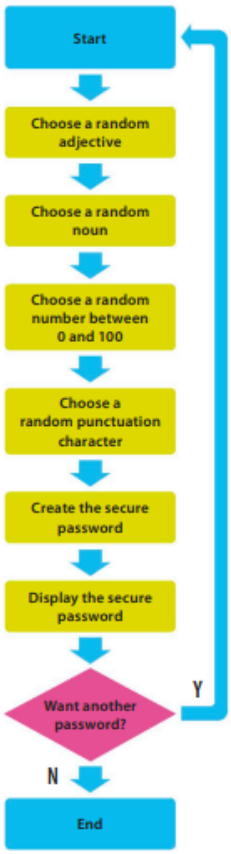
Password Picker akan memungkinkan Anda membuat kata sandi yang kuat dengan menggabungkan kata, angka, dan karakter. Saat Anda menjalankan program, itu akan membuat kata sandi baru dan menampilkannya di layar. Anda dapat memintanya untuk terus membuat kata sandi baru sampai Anda menemukan kata sandi yang Anda sukai.

Pembobol kata sandi

Cracker adalah program yang digunakan oleh *hacker* untuk menebak password. Beberapa *cracke*r dapat membuat jutaan tebakan setiap detik. Seorang *cracker* biasanya memulai dengan menebak kata dan nama yang umum digunakan. Kata sandi yang tidak biasa yang terdiri dari beberapa bagian berbeda akan membantu melindungi dari *cracker.*

Diagram alur Pemilih Kata Sandi

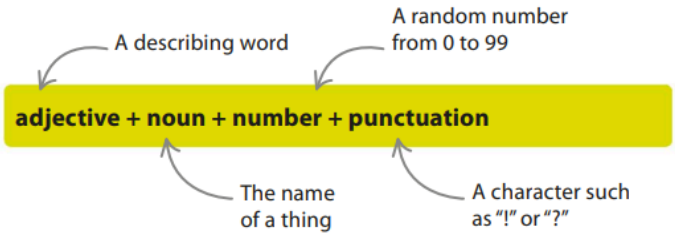
Program secara acak memilih masing-masing dari empat bagian kata sandi, menggabungkannya, dan menampilkan kata sandi di jendela shell. Jika Anda menginginkan kata sandi lain, itu mengulangi langkah-langkah itu lagi. Jika tidak, program berakhir.



Gambar 2.18 *Flowchart Password*

Bagaimana itu bekerja

Proyek ini akan menunjukkan cara menggunakan modul acak Python. Program ini menggunakan pilihan acak dari kelompok kata sifat, kata benda, angka, dan karakter tanda baca untuk menyusun setiap kata sandi. Anda akan segera membuat kata sandi yang gila dan sulit dilupakan, seperti "fluffyapple14" atau "smellygoat&".



Gambar 2.19 Susunan Kata sandi yang digunakan

Pintar namun sederhana!

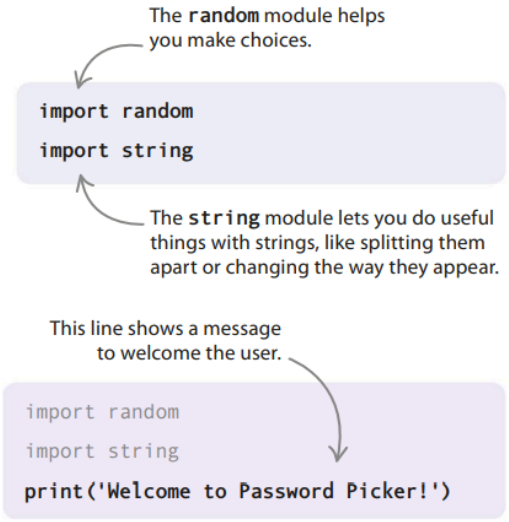
Program ini melakukan hal-hal cerdas dengan kata sandi, tetapi tidak ada banyak kode di dalamnya, jadi tidak perlu waktu lama untuk membuatnya.

1. Buat file baru

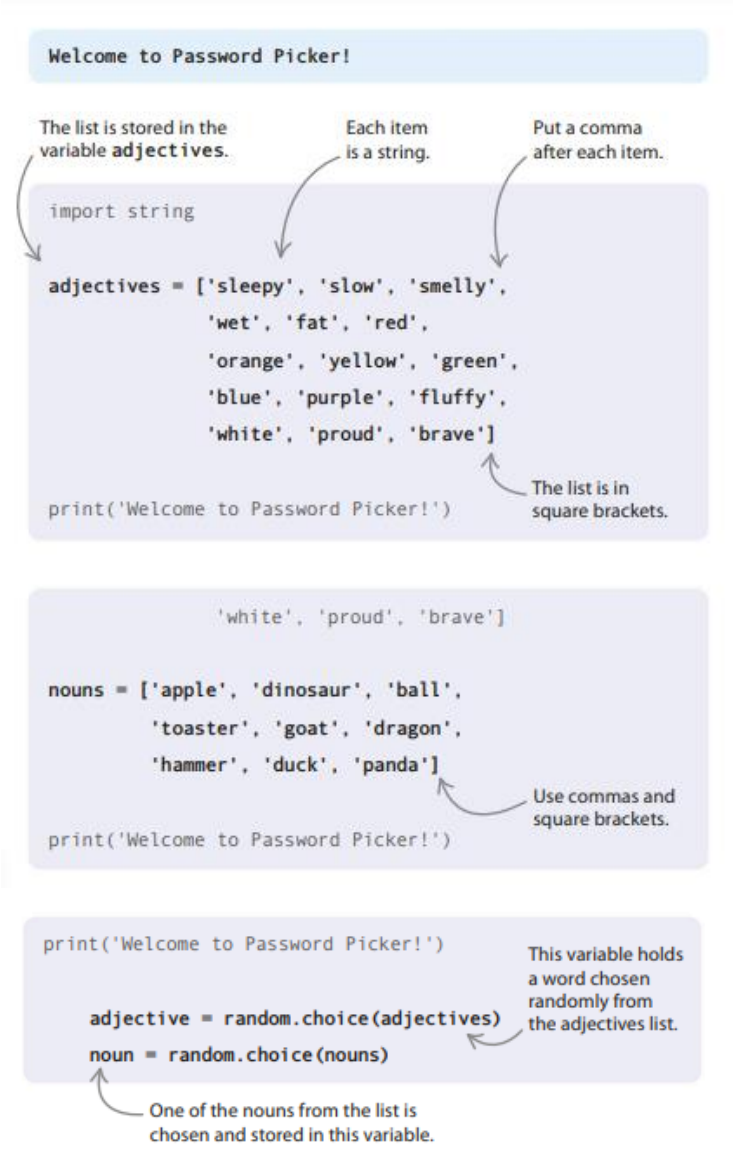
   Buka IDLE. Di bawah menu File, pilih File Baru. Simpan file sebagai "password_picker.py".
2. Tambahkan modul

Impor string dan modul acak dari pustaka Python. Ketik dua baris ini di bagian atas file Anda, sehingga Anda dapat menggunakan modul nanti.

3. Selamat datang pengguna
   Pertama buat pesan untuk menyambut pengguna ke program.

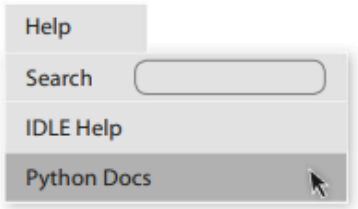


4. Coba kodenya
   Jalankan kode Anda. Pesan selamat datang akan muncul di jendela shell.
5. Buat daftar kata sifat
   Anda memerlukan kata sifat dan kata benda untuk menghasilkan kata sandi baru. Dengan Python, Anda dapat menyimpan sekelompok hal terkait sebagai sebuah daftar. Pertama buat kata sifat variabel untuk menyimpan daftar Anda dengan mengetikkan blok kode baru ini di antara perintah print() dan pernyataan impor. Letakkan seluruh daftar dalam tanda kurung siku, dan pisahkan setiap item dengan koma.
6. Buat daftar kata benda
   Selanjutnya buat variabel yang menampung daftar kata benda. Letakkan di bawah daftar kata sifat dan di atas perintah print(). Ingatlah untuk menggunakan koma dan tanda kurung siku, seperti yang Anda lakukan pada Langkah 5.
7. Pilih kata-kata
   Untuk membuat kata sandi, Anda harus memilih kata sifat acak dan kata benda acak. Anda melakukan ini menggunakan fungsi choice() dari modul acak. Ketik kode ini di bawah perintah print(). (Anda dapat menggunakan fungsi ini kapan saja Anda ingin memilih item acak dari daftar. Berikan saja variabel yang berisi item tersebut.)



Nomor acak

Melempar dadu, mengambil kartu dari dek, atau melempar koin adalah semua hal yang dapat Anda simulasikan dengan menghasilkan angka acak. Anda dapat membaca lebih lanjut tentang cara menggunakan modul acak Python di bagian "Dokumen" pada menu "Bantuan".



8. Pilih nomor

   Sekarang gunakan fungsi randrange() dari modul acak untuk memilih angka acak dari 0 hingga 99. Letakkan baris ini di bagian bawah kode Anda.

```
noun = random.choice(nouns)
number = random.randrange(0, 100)
```

9. Pilih karakter khusus

   Menggunakan fungsi random.choice() lagi, tambahkan baris ini untuk memilih karakter tanda baca acak. Ini akan membuat kata sandi Anda lebih sulit untuk dipecahkan!

```
number = random.randrange(0, 100)
special_char = random.choice(string.punctuation)
```

This is a constant.

10. Buat kata sandi aman baru

    Saatnya untuk merakit semua bagian yang berbeda untuk membuat kata sandi aman yang baru. Ketik dua baris kode ini di akhir program Anda.

Your secure password will be stored in this variable.

This changes the random number into a string.

```
password = adjective + noun + str(number) + special_char
print('Your new password is: %s' % password)
```

This displays the new password in the shell.

Konstanta

Konstanta adalah jenis variabel khusus yang isinya tidak dapat diubah. String.punctuation konstan memegang string karakter yang digunakan untuk tanda baca. Untuk melihat isinya, ketik import string ke dalam shell, diikuti dengan print(string.punctuation).

```
>>> import string
>>> print(string.punctuation)
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

Characters in this constant

String dan bilangan bulat

Fungsi str() mengubah bilangan bulat (integer) menjadi string. Jika Anda tidak menggunakan fungsi ini, Python menunjukkan kesalahan saat Anda mencoba menambahkan integer ke string. Uji: ketik print('route '+66) ke dalam jendela shell. Untuk menghindari kesalahan ini, gunakan fungsi str() untuk mengubah angka menjadi string terlebih dahulu.

```
>>> print('route '+66)
Traceback (most recent call last):
    File '<pyshell#0>', line 1, in <module>
        print('route '+66)
TypeError: Can't convert 'int' object to str implicitly
```

Error message

```
>>>  print('route '+str(66))
route 66
```

The number goes inside the brackets of the `str()` function.

11. Uji programnya

    Ini adalah poin yang bagus untuk menguji kode Anda. Jalankan dan lihat di shell untuk melihat hasilnya. Jika Anda memiliki kesalahan, jangan khawatir. Lihat kembali kode Anda dengan cermat untuk menemukan kesalahan.

12. Yang lainnya?

    Anda dapat menggunakan loop sementara untuk membuat kata sandi lain jika pengguna mengatakan mereka menginginkan kata sandi yang berbeda. Tambahkan kode ini ke program Anda. Ini menanyakan pengguna apakah mereka memerlukan kata sandi baru, lalu menyimpan balasan dalam variabel yang disebut respons.

```
Welcome to Password Picker!
Your new password is: bluegoat92=
```

```
print('Welcome to Password Picker!')

while True:
    adjective = random.choice(adjectives)
    noun = random.choice(nouns)
    number = random.randrange(0, 100)
    special_char = random.choice(string.punctuation)

    password = adjective + noun + str(number) + special_char
    print('Your new password is: %s' % password)

    response = input('Would you like another password? Type y or n: ')
    if response == 'n':
        break
```

The `while` loop starts here.

You need to indent these existing lines to make sure they're in the `while` loop.

The `while` loop ends here.

The `input()` function asks the user to enter a response into the shell.

If the answer's "yes" (y), the loop returns to the start. If it's "no" (n), the program exits the loop.

13. Pilih kata sandi yang sempurna

    Itu saja – Anda sudah selesai. Sekarang Anda dapat membuat kata sandi yang sulit diretas yang menyenangkan untuk diingat!

```
Welcome to Password Picker!
Your new password is: yellowapple42}
Would you like another password? Type y or n: y
Your new password is: greenpanda13*
Would you like another password? Type y or n: n
```

Type "y" at this prompt to get a new password.

Type "n" at this prompt to quit the program.

Peretasan dan penyesuaian

Coba remix program Anda untuk menambahkan fitur tambahan ini. Bisakah Anda memikirkan cara lain untuk membuatnya lebih tahan *cracker*?

Tambahkan lebih banyak kata

Untuk menambah jumlah kemungkinan kata sandi, tambahkan lebih banyak kata ke daftar kata benda dan kata sifat. Pikirkan kata-kata yang tidak biasa atau konyol yang akan melekat di benak Anda jika muncul dalam kata sandi.

```
nouns = ['apple', 'dinosaur', 'ball',
         'toaster', 'goat', 'dragon',
         'hammer', 'duck', 'panda',
         'telephone', 'banana', 'teacher']
```

Dapatkan banyak kata sandi

Ubah kodenya sehingga program Anda akan membuat dan menampilkan tiga kata sandi sekaligus. Anda harus menggunakan for loop. Masukkan ke dalam loop while.

```
while True:

    for num in range(3):
        adjective = random.choice(adjectives)
        noun = random.choice(nouns)
        number = random.randrange(0, 100)
        special_char = random.choice(string.punctuation)

        password = adjective + noun + str(number) + special_char
        print('Your new password is: %s' % password)

    response = input('Would you like more passwords? Type y or n: ')
```

The for loop runs 3 times, and selects 3 different passwords.

Keep these lines indented.

Buat lebih lama

Buat kata sandi lebih panjang dan lebih aman dengan menambahkan kata lain ke setiap kata sandi. Anda dapat membuat daftar warna, lalu memilih warna acak untuk ditambahkan ke setiap kata sandi.

Add a random colour.

```
Your new password is: hairybluepotato33%
```

2.9 MODUL

Modul adalah kumpulan kode yang membantu Anda mengatasi tantangan pengkodean umum. Modul memberikan potongan kode yang kurang menarik, memungkinkan Anda fokus pada hal-hal yang menyenangkan. Selain itu, karena modul digunakan oleh banyak orang, modul tersebut cenderung berfungsi dengan baik dan bebas dari bug.

Modul bawaan

Ada banyak modul berguna yang disertakan dengan Python. Kumpulan modul ini dikenal sebagai Perpustakaan Standar. Berikut adalah beberapa modul menarik dari perpustakaan yang mungkin ingin Anda coba.

statistik

Gunakan statistik untuk menghitung rata-rata atau menemukan nilai paling umum dalam daftar angka. Ini berguna jika Anda perlu menghitung skor rata-rata dalam permainan.

acak

Anda menggunakan modul ini untuk membuat pilihan acak di Password Picker. Sangat bagus untuk menambahkan elemen kesempatan untuk
sebuah permainan atau program.

soket

Modul soket memungkinkan program untuk berkomunikasi melalui jaringan dan Internet. Itu bisa digunakan untuk membuat game online.

tanggal waktu

Modul ini memungkinkan Anda bekerja dengan tanggal. Anda bisa mendapatkan tanggal hari ini, atau mencari tahu berapa lama sampai hari istimewa.

browser web

Anda dapat mengontrol browser web komputer dengan modul ini, memungkinkan Anda untuk membuka halaman web langsung dari kode Anda.

Menggunakan modul

Untuk menggunakan modul dalam kode Anda, Anda harus memberi tahu Python bahwa Anda ingin memasukkannya. Anda menginstruksikan Python modul mana yang harus disertakan menggunakan pernyataan impor. Ada beberapa cara berbeda yang dapat Anda lakukan, tergantung pada apa yang Anda butuhkan dari modul.

impor...

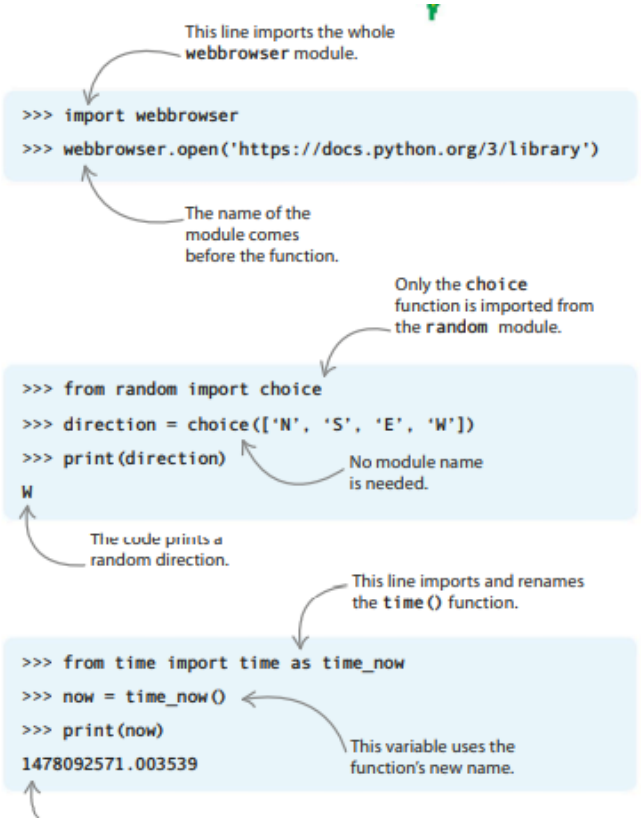
Mengetik impor kata kunci memungkinkan Anda menggunakan semua konten modul. Namun, Anda harus meletakkan nama modul sebelum fungsi apa pun yang Anda gunakan. Kode ini mengimpor semua modul webbrowser() dan menggunakan fungsi open() untuk membuka situs web Python di browser komputer.

dari... impor...

Jika Anda hanya ingin menggunakan bagian tertentu dari modul, Anda dapat mengimpor bagian itu saja dengan menambahkan kata kunci from. Sekarang Anda bisa menggunakan nama fungsi itu sendiri. Kode ini mengimpor fungsi choice() modul acak. Fungsi mengambil item acak dari daftar apa pun yang Anda berikan.

dari... impor... sebuah...

Terkadang Anda mungkin ingin mengubah nama modul atau fungsi yang diimpor, mungkin karena Anda sudah menggunakan nama itu atau mungkin tidak cukup jelas. Untuk melakukan ini, gunakan kata kunci as diikuti dengan nama baru. Dalam contoh yang ditunjukkan di sini, fungsi time(), yang telah kita beri nama time_now(), memberi kita waktu saat ini. Waktu yang diberikan adalah jumlah detik yang tepat sejak 00:00 pada 1 Januari 1970—tanggal yang digunakan oleh sebagian besar komputer sebagai awal jam mereka.

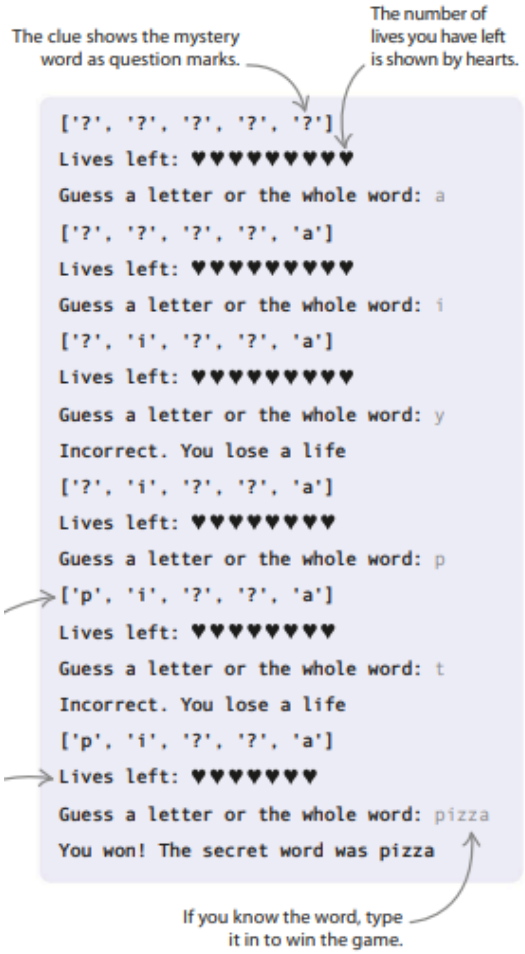


2.10 KESEMPATAN MENEBAK

Dalam game yang menegangkan ini, Anda harus menebak kata rahasia satu huruf dalam satu waktu. Jika tebakan Anda salah, Anda kehilangan nyawa. Pilih huruf Anda dengan hati-hati, karena Anda hanya memiliki sembilan nyawa. Kehilangan seluruh hidup Anda, dan permainan berakhir!

Apa yang terjadi

Program ini menunjukkan kepada Anda sebuah kata misteri dengan huruf-hurufnya diganti dengan tanda tanya. Jika Anda menebak huruf dengan benar, program akan mengganti tanda tanya dengan huruf yang benar. Ketika Anda merasa tahu apa kata itu, ketikkan secara lengkap. Permainan berakhir setelah Anda memasukkan kata yang benar atau tidak memiliki nyawa tersisa.



Bagaimana itu bekerja

Pertama, Anda akan membuat dua daftar: satu untuk menyimpan kata-kata rahasia dan satu lagi untuk menyimpan petunjuk, yang terdiri dari tanda tanya. Kemudian, dengan menggunakan modul acak, Anda akan membuat pilihan acak dari daftar kata-kata rahasia. Selanjutnya Anda akan membuat lingkaran untuk memeriksa tebakan pemain, dan juga membuat fungsi untuk memperbarui petunjuk saat kata perlahan terungkap.

Karakter Unicode

Huruf, angka, tanda baca, dan simbol yang dapat ditampilkan di komputer dikenal sebagai karakter. Ada karakter untuk sebagian besar bahasa dunia dan karakter khusus untuk gambar sederhana, termasuk emoji. Karakter datang dalam set. Misalnya, set karakter ASCII (Kode Standar Amerika untuk Pertukaran Informasi) digunakan untuk bahasa Inggris. Untuk hati dalam proyek ini, Anda akan menggunakan set karakter Unicode, yang berisi banyak simbol berbeda, termasuk yang di bawah ini.

Diagram alir Sembilan Kehidupan

Flowchartnya terlihat rumit, tetapi kode untuk game ini relatif singkat. Bagian utama dari program ini adalah loop yang memeriksa huruf-huruf yang ditebak untuk melihat apakah itu bagian dari kata rahasia, dan apakah pemain memiliki nyawa yang tersisa.

Gambar 2.20 Flowchart kesempatan Menebak dalam Game

Pengaturan

Anda akan membangun Nine Lives dalam dua tahap. Pertama Anda akan mengimpor modul yang Anda butuhkan untuk program dan membuat beberapa variabel. Kemudian Anda akan menulis kode utama untuk program tersebut.

1. Buat file baru

   Buka IDLE dan buat file baru. Simpan sebagai "nine_lives.py".

2. Impor modul

   Proyek ini menggunakan modul acak Python, jadi mulailah dengan mengetikkan baris kode yang ditampilkan di sini untuk mengimpornya.

```
import random
```

3. Buat variabel

   Di bawah garis impor, buat variabel yang disebut nyawa untuk melacak jumlah nyawa (tebakan) yang tersisa dari pemain.

4. Buat daftar

   Program hanya akan mengetahui kata-kata yang Anda berikan. Anda harus memasukkan kata-kata ini ke dalam daftar, lalu menyimpan daftar tersebut dalam variabel yang disebut kata. Tambahkan baris ini di bawah variabel hidup Anda.

5. Pilih kata rahasia

   Di awal setiap permainan, program akan secara acak memilih kata yang harus ditebak pemain dan menyimpannya dalam variabel yang disebut secret_word. Tambahkan baris untuk membuat variabel baru ini.

```
import random

lives = 9
```

The player starts with nine lives.

```
lives = 9
words = ['pizza', 'fairy', 'teeth', 'shirt',
         'otter', 'plane']
```

Each item in the list is a string made up of five characters.

```
words = ['pizza', 'fairy', 'teeth', 'shirt',
         'otter', 'plane']
secret_word = random.choice(words)
```

6. Simpan petunjuknya

   Sekarang buat daftar lain untuk menyimpan petunjuk. Huruf yang tidak diketahui disimpan sebagai tanda tanya. Ini akan diganti ketika pemain menebak huruf dengan benar. Di awal permainan, seluruh daftar adalah tanda tanya. Anda bisa menulisnya sebagai petunjuk = daftar['?', '?', '?', '?', '?'], mengetik satu tanda tanya untuk setiap huruf dalam kata rahasia, tetapi kode di bawah ini adalah cara yang lebih cepat untuk menulisnya. Tambahkan baris ini setelah variabel secret_word.

```
secret_word = random.choice(words)
clue = list('?????')
```

The five question marks are stored as a list in the variable **clue**.

7. Tunjukkan nyawa yang tersisa

   Proyek ini menggunakan karakter hati Unicode untuk menampilkan berapa banyak nyawa yang tersisa. Untuk membuat program Anda lebih mudah dibaca dan ditulis, tambahkan baris kode berikutnya untuk menyimpan karakter dalam variabel.

```
clue = list('?????')
heart_symbol = u'\u2764'
```

8. Ingat hasilnya

   Sekarang buat variabel untuk menyimpan apakah pemain sudah menebak kata dengan benar atau belum. Variabel ditetapkan sebagai False untuk memulai karena pemain tidak mengetahui kata saat permainan dimulai. Ketik baris ini di bawah kode untuk simbol hati.

```
heart_symbol = u'\u2764'
guessed_word_correctly = False
```

This is a Boolean (True or False) value.

   Berhati-hatilah untuk hanya menambahkan kata-kata yang panjangnya lima huruf. Daftar yang menyimpan petunjuk hanya memiliki ruang untuk lima karakter. Jika Anda menambahkan kata lebih dari lima huruf, Anda akan melihat pesan kesalahan saat program mencoba memasukkan huruf apa pun yang melewati huruf kelima dalam petunjuk.

```
Index error: list assignment index
out of range
```

   Jika Anda mencoba menambahkan kata yang panjangnya kurang dari lima huruf, program akan bekerja, tetapi pemain masih akan melihat lima tanda tanya. Mereka akan berpikir bahwa jawabannya harus terdiri dari lima huruf. Misalnya, jika Anda menggunakan "mobil", programnya akan terlihat seperti ini.

```
['?', '?', '?', '?', '?']
Lives left: ♥♥♥♥♥♥♥♥♥
Guess a letter or the whole word: c
['c', '?', '?', '?', '?']
Lives left: ♥♥♥♥♥♥♥♥♥
Guess a letter or the whole word: a
['c', 'a', '?', '?', '?']
Lives left: ♥♥♥♥♥♥♥♥♥
Guess a letter or the whole word: r
['c', 'a', 'r', '?', '?']
Lives left: ♥♥♥♥♥♥♥♥♥
Guess a letter or the whole word:
```

The last two question marks don't represent any letters, so they never disappear.

   Pemain tidak akan pernah bisa menang, karena dua tanda tanya terakhir akan tetap ada tidak peduli huruf apa yang mereka tebak!

   Kode utama

   Bagian utama dari kode adalah loop yang mendapat surat dari pemain dan memeriksa apakah itu ada di kata rahasia. Jika ya, kode menggunakan fungsi untuk memperbarui petunjuk. Anda akan membuat fungsi itu, lalu membuat loop utama.

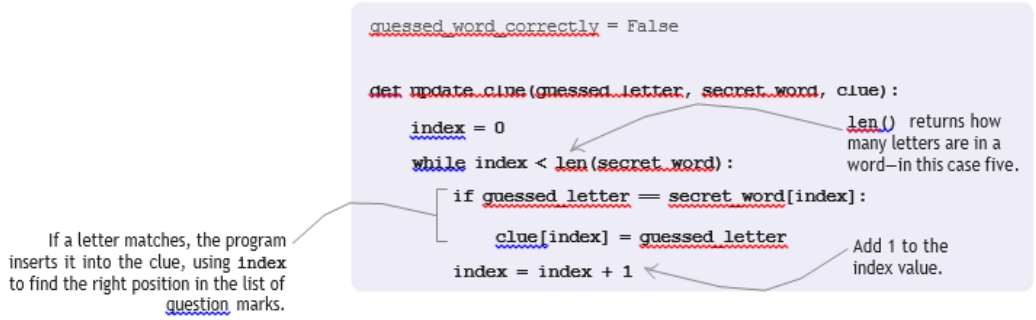
9. Apakah surat itu dalam kata rahasia?

   Jika huruf yang ditebak ada di kata rahasia, Anda harus memperbarui petunjuknya. Untuk melakukan ini, Anda akan menggunakan fungsi yang disebut update_clue().

Fungsi ini memiliki tiga parameter: huruf yang ditebak, kata rahasia, dan petunjuk. Tambahkan kode ini setelah variabel guessed_word_correctly.

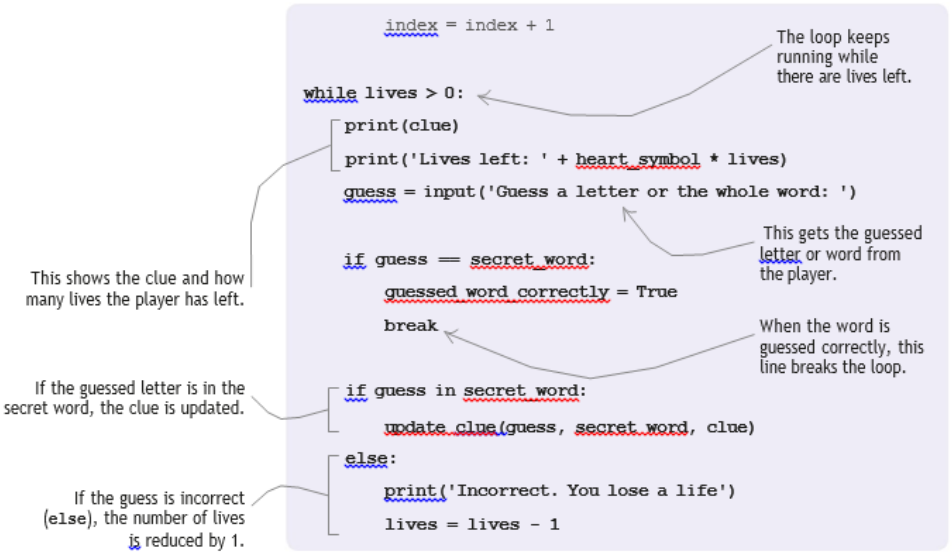
Cara kerjanya

Fungsi ini berisi loop while yang bekerja melalui kata rahasia satu huruf pada satu waktu, memeriksa apakah setiap huruf cocok dengan huruf yang ditebak. Variabel indeks terus menghitung huruf saat ini saat program memindai kata.



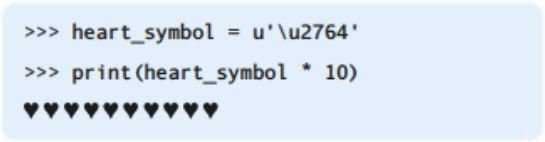
10. Tebak huruf atau kata

Program Anda harus terus meminta pengguna untuk menebak huruf atau keseluruhan kata sampai mereka mendapatkan jawaban yang benar atau kehabisan nyawa. Inilah yang dilakukan loop utama. Tambahkan kode ini di bawah fungsi update_clue().



Mengulang string

Kode print('Lives left: ' + heart_ symbol *lives) menggunakan trik yang rapi untuk menampilkan hati untuk setiap sisa hidup. Anda dapat memberi tahu Python untuk mengulangi string beberapa kali dengan mengalikannya dengan angka. Misalnya, print(heart_symbol * 10) akan menampilkan sepuluh hati. Coba kode ini di shell.



11. Apakah kamu menang?

    Saat permainan berakhir, Anda perlu mencari tahu apakah pemain telah menang. Jika variabel guessed_word_correctly adalah True, Anda tahu loop berakhir sebelum pemain kehabisan nyawa—jadi mereka memenangkan permainan. Jika tidak (lain), mereka telah kalah. Tambahkan kode ini di akhir program Anda.

```
        lives = lives - 1

if guessed_word_correctly:
    print('You won! The secret word was ' + secret_word)
else:
    print('You lost! The secret word was ' + secret_word)
```

This is shorthand for "if guessed_word_correctly = True"

12. Uji kode Anda

    Coba game untuk memastikan itu berjalan dengan baik. Jika ada masalah, hati-hati memeriksa kode Anda untuk bug. Saat berhasil, undang teman Anda untuk mengikuti tantangan Sembilan Nyawa!

```
['?', '?', '?', '?', '?']
Lives left: ♥♥♥♥♥♥♥♥♥
Guess a letter or the whole word:
```

Just type a letter to start playing!

Peretasan dan penyesuaian

Ada banyak cara Anda dapat me-remix dan mengadaptasi game ini. Anda dapat menambahkan kata baru, mengubah panjang kata, atau membuatnya lebih mudah atau lebih sulit.

Tambahkan lebih banyak kata

Coba tambahkan lebih banyak kata ke daftar kata program. Anda dapat menambahkan sebanyak yang Anda inginkan, tetapi ingatlah untuk hanya menggunakan kata-kata yang panjangnya lima huruf.

```
words = ['pizza', 'fairy', 'teeth', 'shirt', 'otter', 'plane', 'brush', 'horse', 'light']
```

Ubah jumlah nyawa

Anda dapat membuatnya lebih mudah atau lebih sulit bagi pemain dengan memberi mereka lebih banyak atau lebih sedikit nyawa. Untuk melakukan ini, cukup ubah variabel kehidupan yang Anda buat di Langkah 3.

Gunakan kata-kata yang lebih panjang

Jika menurut Anda hanya menggunakan kata-kata lima huruf membuat permainan terlalu mudah, beralihlah ke kata-kata yang sedikit lebih panjang—tetapi ingatlah untuk membuat semuanya sama panjangnya. Untuk membuat game ini sangat sulit, carilah kamus untuk kata-kata terpanjang dan paling tidak biasa yang dapat Anda temukan!

2.11 TINGKAT KESULITAN

Untuk membuat permainan lebih menarik, biarkan pemain memilih tingkat kesulitan di awal permainan. Pengaturan yang lebih mudah memberi pemain lebih banyak kehidupan.

1. Dapatkan levelnya
   Letakkan kode ini di awal program utama Anda, tepat di atas loop while. Ini meminta pemain untuk memilih level.

```
difficulty = input('Choose difficulty (type 1, 2 or 3):\n 1 Easy\n 2 Normal\n 3 Hard\n')
difficulty = int(difficulty)

while lives > 0:
```

`difficulty` is currently a string.
This line changes it to an integer.

2. Uji kodenya
   Jalankan program untuk memeriksa apakah perubahan ini berhasil. Anda akan melihat pesan ini muncul di jendela shell.

```
Choose difficulty (type 1, 2, or 3):
 1 Easy
 2 Normal
 3 Hard
```

3. Tetapkan level
   Sekarang gunakan pernyataan if, elif, dan else untuk mengatur jumlah nyawa untuk setiap level. Coba gunakan 12 nyawa untuk mudah, 9 untuk normal, dan 6 untuk keras. Jika Anda tidak senang dengan seberapa mudah atau sulitnya levelnya, Anda dapat mengubah jumlah nyawa setelah Anda mengujinya. Tambahkan kode ini setelah baris yang meminta pemain untuk memilih level.

```
difficulty = input('Choose difficulty (type 1, 2 or 3):\n 1 Easy\n 2 Normal\n 3 Hard\n')
difficulty = int(difficulty)

if difficulty == 1:
    lives = 12
elif difficulty == 2:
    lives = 9
else:
    lives = 6
```

Kata-kata dengan panjang yang bervariasi

Bagaimana jika Anda ingin bermain game dengan panjang kata yang bervariasi? Jika Anda tidak tahu panjang kata rahasia sebelum program dijalankan, Anda tidak akan tahu berapa lama untuk membuat daftar untuk menyimpan petunjuk. Ada perbaikan pintar yang dapat Anda gunakan untuk menyelesaikan masalah ini.

1. Gunakan daftar kosong
   Saat Anda membuat daftar yang berisi petunjuk, jangan mengisinya dengan tanda tanya—kosongkan saja daftarnya. Buat perubahan ini pada daftar petunjuk.

```
clue = []
```
There's nothing inside the brackets.

2. Tambahkan lingkaran baru

   Untuk membuat petunjuk dengan panjang yang benar setelah kata rahasia dipilih, gunakan loop sederhana ini. Ini menghitung berapa banyak huruf dalam kata dan menambahkan tanda tanya untuk setiap huruf.

```
clue = []
index = 0
while index < len(secret_word):
    clue.append('?')
    index = index + 1
```
The **append()** function simply adds an item to the end of the list.

Jadikan akhir lebih cerdas

Saat ini, permainan tidak berakhir sampai Anda mengetikkan kata secara lengkap. Mari kita buat kode lebih pintar sehingga permainan berakhir saat Anda menebak huruf terakhir.

1. Buat variabel lain

   Pertama buat variabel untuk menghitung berapa banyak huruf yang tidak diketahui. Tambahkan kode ini di atas fungsi update_clue.

At first all the letters are unknown.

```
unknown_letters = len(secret_word)
```

2. Mengedit fungsi

   Selanjutnya ubah fungsi update_clue() seperti gambar di bawah ini. Setiap kali pemain menebak huruf dengan benar, program sekarang akan menghilangkan berapa kali huruf itu muncul di kata rahasia dari unknown_letters.

```
def update_clue(guessed_letter, secret_word, clue, unknown_letters):
    index = 0
    while index < len(secret_word):
        if guessed_letter == secret_word[index]:
            clue[index] = guessed_letter
            unknown_letters = unknown_letters - 1
        index = index + 1

    return unknown_letters
```
Add this new parameter to the **update_clue** function.

The code subtracts 1 from **unknown_letters** each time a guessed letter appears in the word.

This line makes the function return the number of unknown letters.

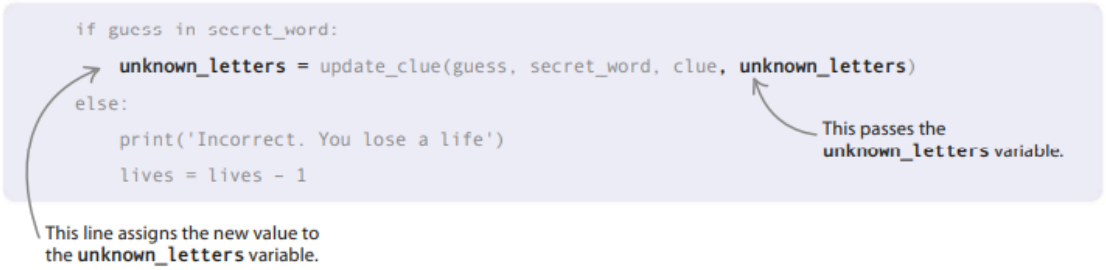
Cara kerjanya

Mengapa Anda harus memperbarui unknown_letters di fungsi update_clue ()? Mengapa Anda tidak bisa mengurangi 1 ketika Anda tahu bahwa huruf yang ditebak ada di kata rahasia? Ini akan berhasil jika setiap huruf hanya muncul sekali dalam kata rahasia. Tapi jika surat itu muncul berkali-kali, itu akan membuat hitungan Anda salah.

Dengan memperbarui variabel dalam fungsi, kode akan mengurangi 1 dari unknown_letters setiap kali huruf muncul di kata rahasia. Ini karena fungsinya memeriksa setiap huruf dalam kata rahasia untuk melihat apakah itu cocok dengan huruf yang ditebak.

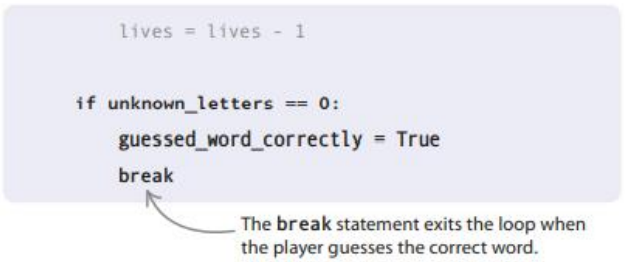
3. Memanggil fungsi

   Anda juga harus mengubah fungsi update_clue() untuk meneruskan variabel unknown_letters dan menyimpan nilai baru.



4. Memenangkan permainan

   Ketika unknown_letters mencapai 0, pengguna telah menebak kata dengan benar. Tambahkan kode ini di akhir loop utama. Sekarang permainan akan secara otomatis mengumumkan Anda sebagai pemenang ketika Anda telah menebak semua huruf.



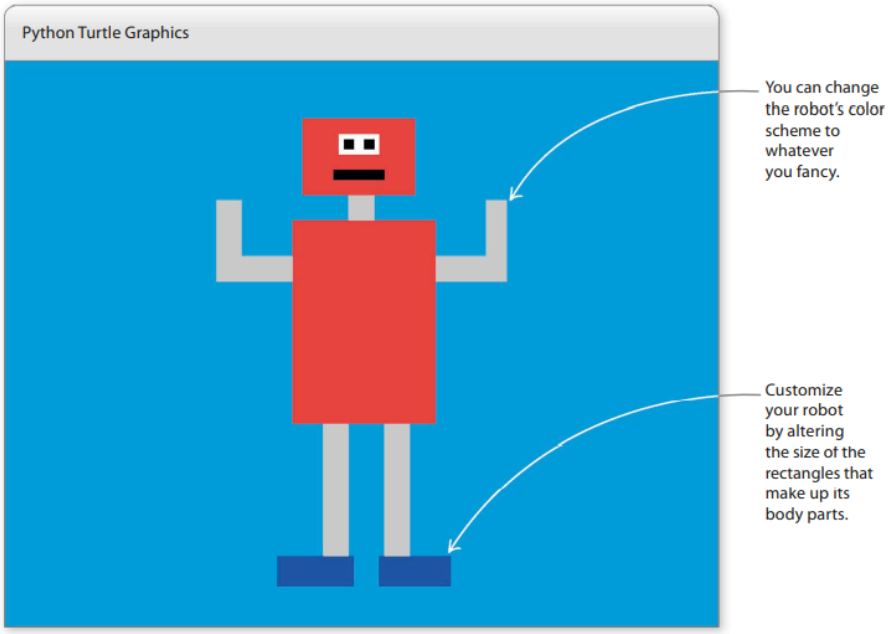
BAB 3
GRAFIK KURA-KURA

3.1 PEMBUAT ROBOT

Membuat grafik dengan Python itu mudah. Modul kura-kura Python memungkinkan Anda memindahkan "kura-kura" robot di sekitar layar, menggambar gambar dengan pena saat berjalan. Dalam proyek ini, Anda akan memprogram kura-kura untuk membuat lebih banyak robot—atau setidaknya gambar robot!

Apa yang terjadi

Saat Anda menjalankan program, kura-kura Python bergerak, berlarian di sekitar layar saat ia menggambar robot yang ramah. Perhatikan saat robot merakit potongan demi potongan, menggunakan warna yang berbeda.



Bagaimana itu bekerja

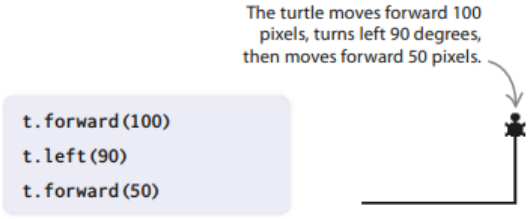
Anda akan mulai dengan menulis fungsi yang menggambar persegi panjang. Kemudian Anda akan menyatukan persegi panjang untuk membuat robot. Anda dapat mengubah ukuran dan warna persegi panjang dengan mengubah parameter yang Anda berikan ke fungsi. Jadi Anda bisa memiliki balok panjang dan tipis untuk kaki, balok persegi untuk mata, dan seterusnya.

Jangan panggil aku kura-kura!

Berhati-hatilah untuk tidak pernah menamai program penyu Anda "turtle.py". Jika Anda melakukannya, Python akan menjadi sangat bingung dan memberi Anda banyak pesan kesalahan.

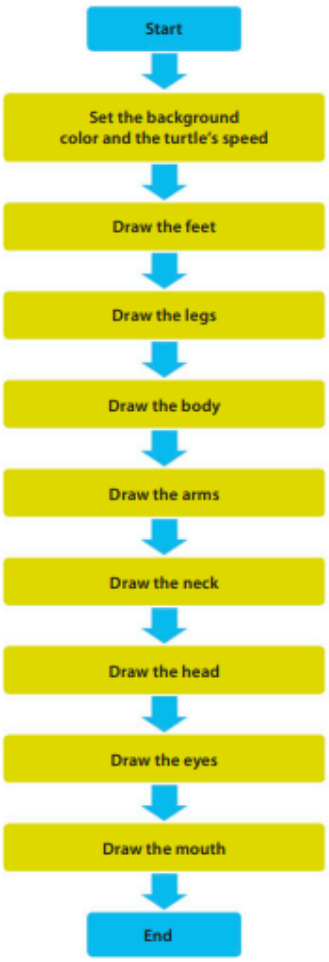
Menggambar dengan kura-kura

Modul kura-kura memungkinkan Anda untuk mengontrol kura-kura robot pembawa pena. Dengan memberikan instruksi kepada kura-kura tentang bagaimana ia harus bergerak di sekitar layar, Anda dapat menggambar berbagai gambar dan desain. Anda juga dapat memberi tahu kura-kura kapan harus meletakkan pena dan mulai menggambar, atau kapan harus menariknya ke atas agar dapat berpindah ke bagian layar yang berbeda tanpa meninggalkan jejak yang tidak rapi.



Bagan alir Pembuat Robot

Flowchart menunjukkan bagaimana kode untuk proyek ini cocok satu sama lain. Pertama program mengatur warna latar belakang dan seberapa cepat kura-kura bergerak. Kemudian ia menggambar robot satu per satu, mulai dari kakinya dan bergerak ke atas ke kepalanya.



Gambar 3.1 Flowchart pembuat robot

Menggambar persegi panjang

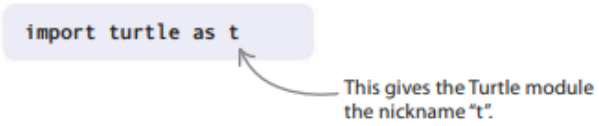
Mari kita mulai dengan mengimpor modul turtle dan menggunakannya untuk membuat fungsi yang menggambar persegi panjang.

1. Buat file baru

   Buka IDLE dan buat file baru. Simpan sebagai "robot_builder.py".

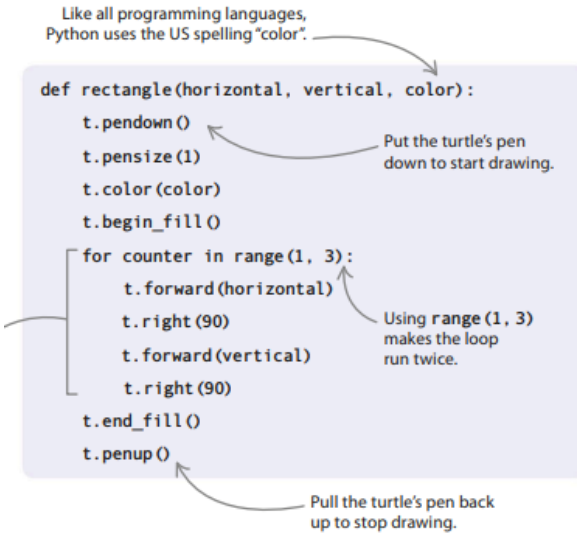
2. Impor modul Turtle

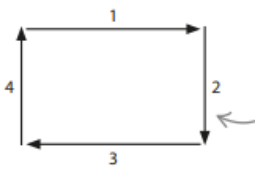
   Ketik baris ini di bagian atas program Anda. Perintah import turtle as t memungkinkan Anda menggunakan fungsi dari modul turtle tanpa harus mengetik "turtle" secara penuh setiap kali. Ini seperti memanggil seseorang yang namanya Benjamin "Ben" singkatnya.



3. Buat fungsi persegi panjang

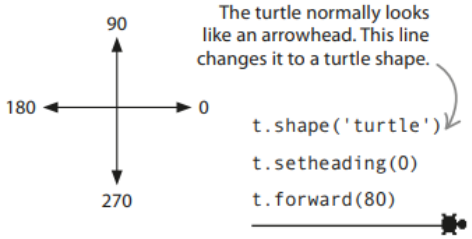
   Sekarang buat fungsi untuk menggambar balok yang akan Anda gunakan untuk membangun robot Anda. Fungsi ini memiliki tiga parameter: panjang sisi horizontal; panjang sisi vertikal; dan warna. Anda akan menggunakan lingkaran yang menggambar satu sisi horizontal dan satu sisi vertikal setiap kali berjalan, dan Anda akan membuatnya berjalan dua kali. Letakkan fungsi persegi panjang ini di bawah kode yang Anda tambahkan pada Langkah 2.





Modus kura-kura

Anda akan menggunakan kura-kura dalam mode standarnya. Ini berarti kura-kura mulai menghadap ke sisi kanan layar. Jika Anda mengatur heading (kata lain untuk arah) ke 0, itu akan menghadap ke kanan. Mengatur heading ke 90 membuatnya mengarah ke atas layar, 180 mengarah ke kiri, dan 270 membuatnya mengarah ke bawah layar.

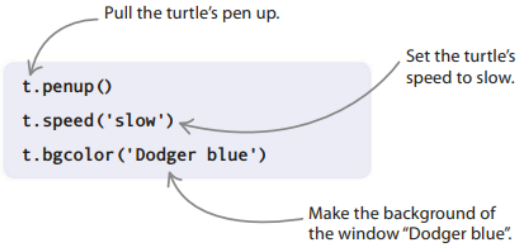


Kecepatan kura-kura

Anda dapat mengontrol seberapa cepat kura-kura menggambar dengan menggunakan perintah t.speed() untuk menyetel kecepatannya ke salah satu nilai berikut: “paling lambat”, “lambat”, “normal”, “cepat”, dan “tercepat”.

4. Mengatur latar belakang

   Selanjutnya siapkan kura-kura untuk mulai menggambar, dan atur warna latar belakang jendela. Anda perlu kura-kura untuk memulai dengan penanya sehingga tidak menarik garis sampai Anda menginginkannya. Itu hanya akan mulai menggambar ketika mencapai kaki robot (Langkah 5). Ketik kode berikut di bawah kode yang Anda tambahkan di Langkah 3.

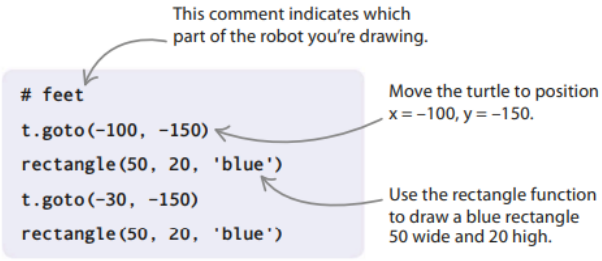


Membangun robot

Sekarang Anda siap untuk mulai membuat robot. Anda akan membuatnya sepotong demi sepotong, dimulai dengan kaki dan terus ke atas. Seluruh robot akan dibuat menggunakan persegi panjang dengan ukuran dan warna yang berbeda, masing-masing diambil dari titik awal yang berbeda di jendela Turtle.

5. Gambar kaki

   Anda perlu memindahkan kura-kura ke tempat Anda ingin mulai menggambar kaki pertama, dan kemudian menggunakan fungsi persegi panjang untuk menggambarnya. Anda harus melakukan hal yang sama untuk kaki kedua. Ketik baris ini di bawah kode yang Anda tambahkan pada Langkah 4, lalu jalankan program untuk melihat kaki robot Anda muncul.

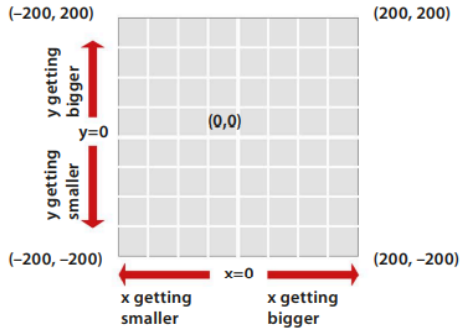


Komentar

Anda akan melihat bahwa ada beberapa baris dalam program ini yang dimulai dengan simbol #. Kata-kata setelah # adalah komentar, ditambahkan untuk membuat kode lebih mudah dibaca dan dipahami pengguna. Python tahu bahwa itu harus mengabaikan mereka.

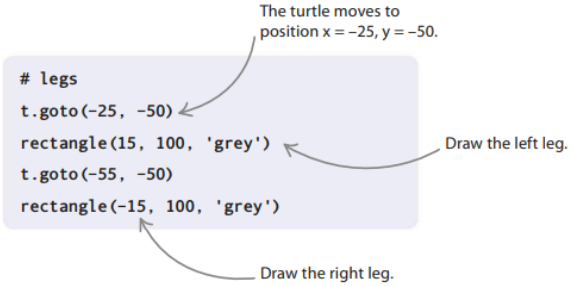
Koordinat penyu

Python akan menyesuaikan jendela Turtle agar sesuai dengan layar Anda, tetapi mari kita gunakan contoh 400 piksel kali 400 piksel. Python menggunakan koordinat untuk mengidentifikasi semua tempat di jendela tempat kura-kura berada. Ini berarti bahwa setiap tempat di jendela dapat ditemukan dengan menggunakan dua angka. Angka pertama, koordinat x, menunjukkan seberapa jauh ke kiri atau kanan dari pusat kura-kura. Angka kedua, koordinat y, menunjukkan seberapa jauh ke atas atau ke bawah dari pusatnya. Koordinat ditulis dalam tanda kurung, dengan koordinat x terlebih dahulu, seperti ini: (x, y).



6. Gambar kaki

   Bagian program berikutnya membuat kura-kura bergerak ke tempat ia akan mulai menggambar kakinya. Ketik baris ini di bawah kode yang Anda tambahkan pada Langkah 5. Sekarang jalankan kode lagi.



7. Menggambar tubuh

   Ketik kode ini di bawah kode yang Anda tambahkan pada Langkah 6. Jalankan program dan Anda akan melihat badan muncul.

8. Gambar lengannya

   Setiap lengan digambar menjadi dua bagian: pertama lengan atas, dari bahu robot hingga siku; lalu lengan bawah, dari siku ke pergelangan tangan. Ketik ini di bawah kode yang Anda tambahkan pada Langkah 7, lalu jalankan untuk melihat lengan muncul.

9. Gambar lehernya

   Saatnya memberi robot Anda leher. Ketik perintah menggambar leher ini di bawah kode yang Anda tambahkan di Langkah 8.

```
# neck
t.goto(-50, 120)
rectangle(15, 20, 'grey')
```

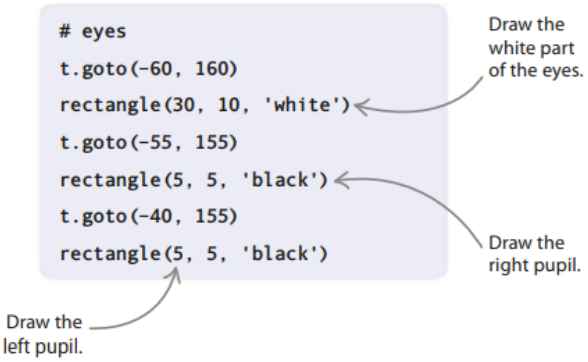
10. Gambar kepala

    Ups—Anda telah menggambar robot tanpa kepala! Untuk memberi kepala robot Anda yang malang, ketik perintah ini di bawah kode yang Anda tambahkan pada Langkah 9.

```
# head
t.goto(-85, 170)
rectangle(80, 50, 'red')
```

11. Menggambar mata

    Mari tambahkan beberapa mata sehingga robot dapat melihat ke mana arahnya. Untuk melakukan ini, Anda akan menggambar persegi panjang putih besar dengan dua kotak kecil di dalamnya (untuk pupil). Anda tidak perlu menulis fungsi baru untuk menggambar persegi, karena persegi adalah persegi panjang dengan semua sisinya sama panjang. Masukkan perintah ini di bawah kode yang Anda tambahkan di Langkah 10.



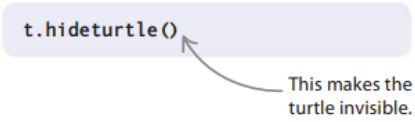
12. Gambar mulut

    Sekarang beri robot itu mulut. Ketik perintah ini di bawah kode yang Anda tambahkan di Langkah 11.

```
# mouth
t.goto(-65, 135)
rectangle(40, 5, 'black')
```

13. Sembunyikan kura-kura

    Terakhir, sembunyikan kura-kura agar tidak terlihat aneh duduk di wajah robot. Ketik baris ini setelah kode yang Anda tambahkan pada Langkah 12. Jalankan program untuk melihat keseluruhan robot yang sedang dibangun.

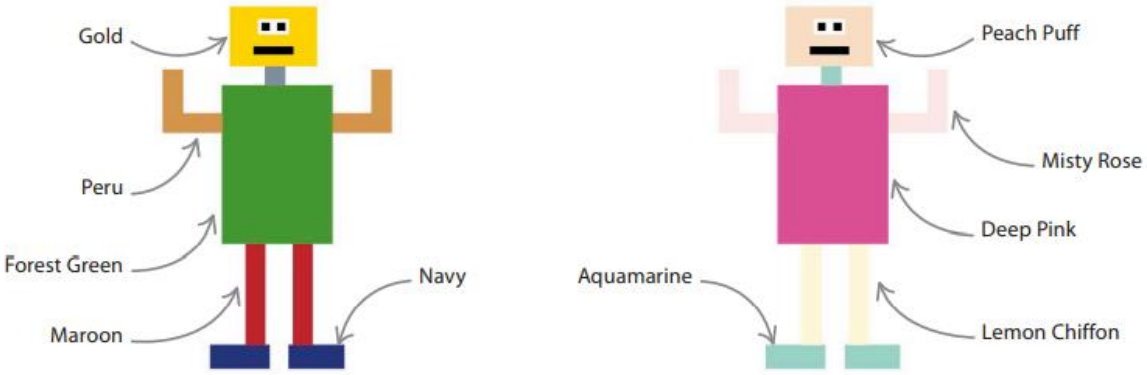


Peretasan dan penyesuaian

Sekarang proyek Anda sudah berjalan, berikut adalah beberapa ide untuk memodifikasi kode sehingga Anda dapat menyesuaikan robot yang Anda buat.

Ubah warna

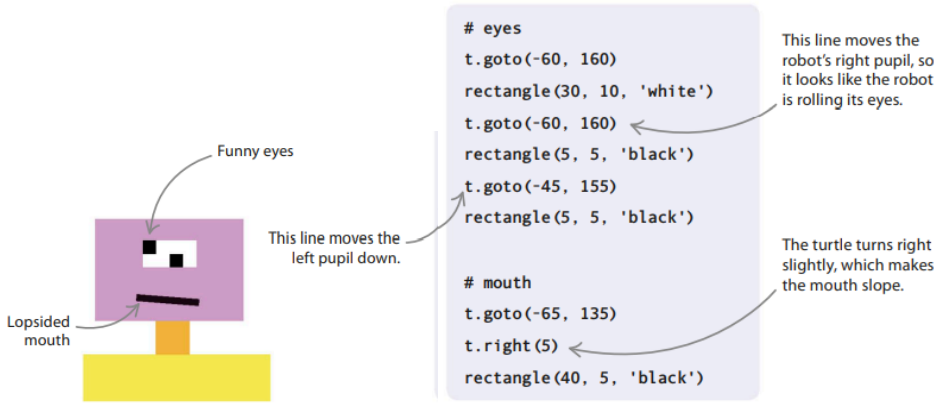
Robot yang Anda buat cukup berwarna, tetapi pasti ada ruang untuk perbaikan. Anda dapat mengubah kode untuk membuat robot yang sesuai dengan warna ruangan atau baju tim sepak bola favorit Anda, atau membuat robot yang benar-benar beraneka warna! Di sebelah kanan ada beberapa warna yang dikenali kura-kura.



Gambar 3.2 Macam-macam Nama Warna dalam Python

Ubah wajah

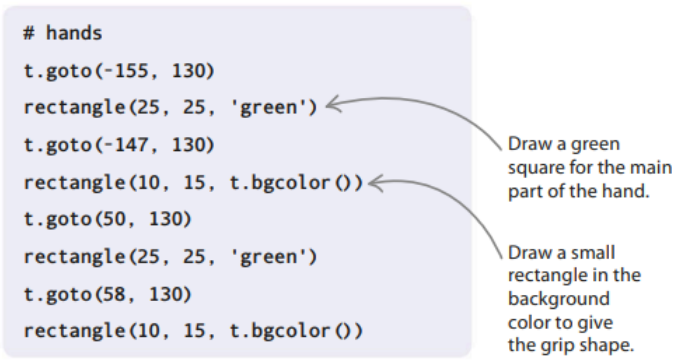
Anda dapat mengubah ekspresi wajah robot dengan mengatur ulang fitur-fiturnya. Untuk memberikan mata dan mulut yang miring, gunakan kode di sebelah kanan.



Gambar 3.3 Gambar Sketas Wajah dan pelokasian bagiannya

Sebuah uluran tangan

Tambahkan kode ini untuk memberikan tangan mencengkeram robot berbentuk U Anda. Anda dapat membentuk kembali tangan agar terlihat seperti pengait, penjepit, atau apa pun yang Anda suka. Biarkan imajinasi Anda menjadi liar dan buat versi Anda sendiri!

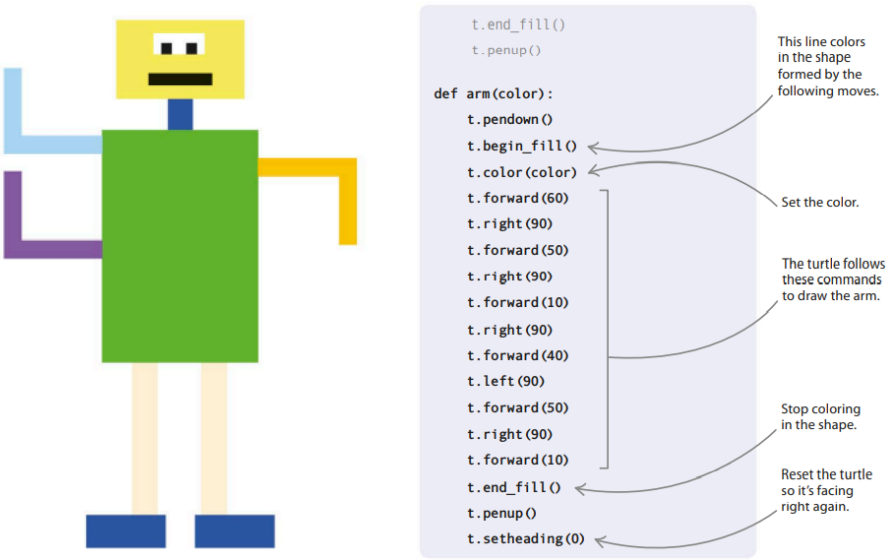


Lengan *all-in-one*

Menggambar lengan di beberapa bagian membuatnya canggung untuk mengubah posisinya atau menambahkan lengan ekstra. Dalam peretasan ini, Anda akan menulis fungsi yang menarik lengan sekaligus.

1. Buat fungsi lengan

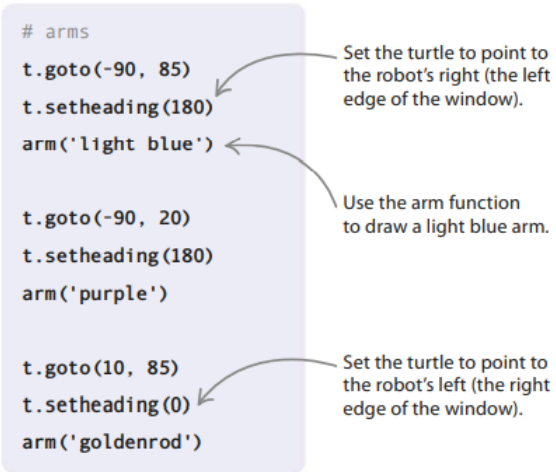
   Pertama tambahkan fungsi baru ini, yang menggambar bentuk lengan dan memberinya warna.



Gambar 3.4 pelokasian lengan karakter

2. Tambahkan lengan

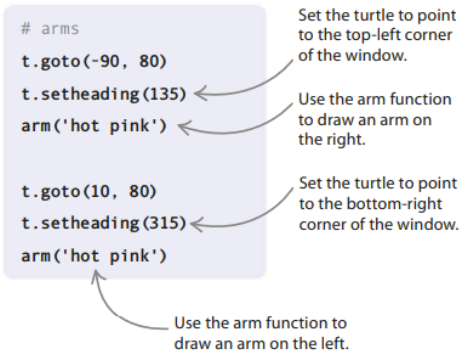
   Selanjutnya ganti kode yang semula Anda miliki di antara baris komentar # lengan dan leher baris komentar # dengan kode yang ditampilkan di sini. Ini menggunakan fungsi lengan untuk menggambar tiga lengan.



Menggerakkan lengan

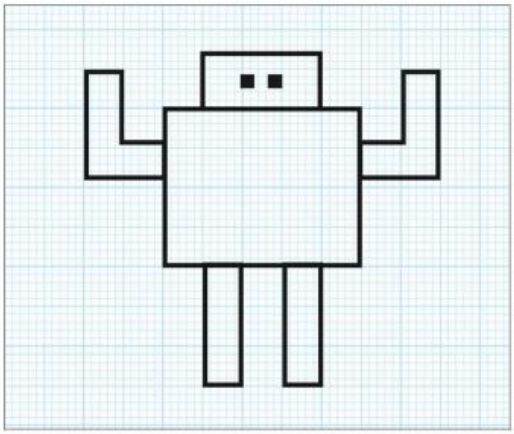
Sekarang setelah Anda dapat menggambar seluruh lengan sekaligus, Anda dapat mengubah posisinya sehingga robot terlihat seperti sedang menggaruk kepalanya atau mungkin sedang menari Highland Fling! Untuk melakukannya, gunakan fungsi

setheading() untuk mengubah arah kura-kura menghadap saat mulai menggambar lengan.



Percobaan dan kesalahan

Saat Anda mendesain robot atau menambahkan fitur baru ke robot yang sudah ada, mungkin diperlukan sedikit percobaan dan kesalahan untuk mendapatkan hal-hal yang Anda inginkan. Jika Anda menambahkan baris print(t.window_ width()) dan print(t.window_height()) setelah baris t.speed('slowest'), Python akan menampilkan tinggi dan lebar jendela Turtle Anda di shell. Kemudian tandai kotak dengan ukuran tersebut pada kertas grafik untuk membantu Anda menentukan koordinat setiap bagian tubuh.



Gambar 3.5 Karakter Robot yang dihasilkan

3.2 KALEIDO-SPIRAL

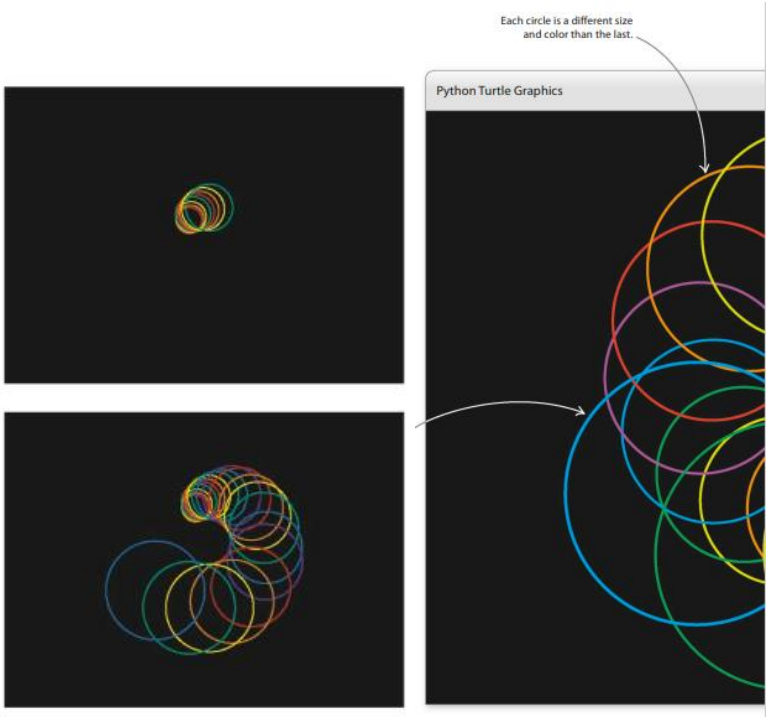
Dengan cara yang sama bahwa baris kode sederhana dapat membentuk program yang baik, bentuk sederhana dapat membentuk gambar yang kompleks. Dengan menggabungkan bentuk dan warna melalui kode, Kaleido-spiral akan membantu Anda menciptakan mahakarya seni digital yang layak untuk galeri seni!

Apa yang terjadi

Kura-kura Python menggambar lingkaran di layar, satu demi satu. Setiap kali sebuah lingkaran digambar, kura-kura mengubah posisi, sudut, warna, dan ukuran lingkaran berikutnya yang digambarnya. Sebuah pola secara bertahap muncul.

Pergeseran spiral

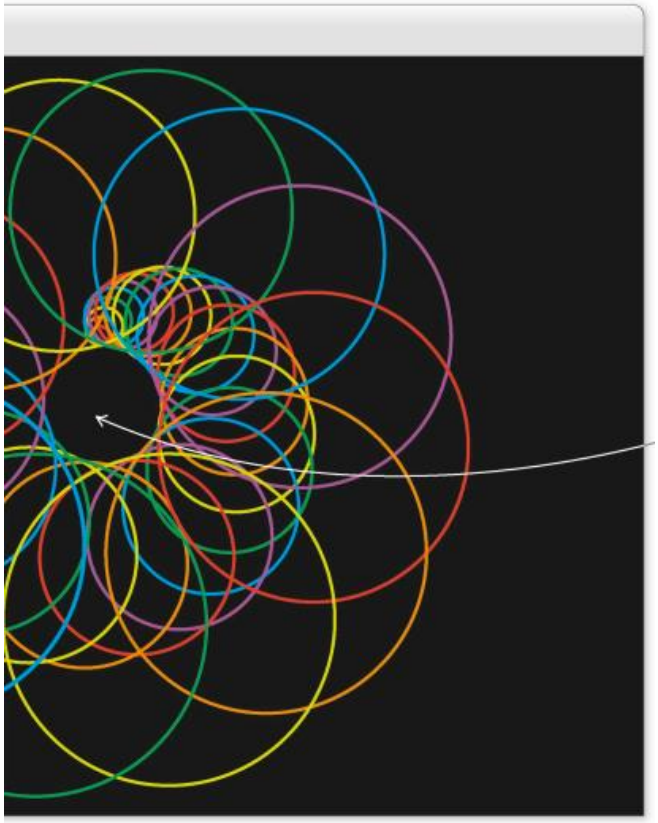
Saat lapisan lingkaran di atas satu sama lain, posisi pergeserannya membentuk spiral yang meliuk keluar dari pusat.



Gambar 3.6 Grafik Kura-kura Python

Program yang dapat disesuaikan

Semakin lama Anda membiarkan Kaleido-spiral berjalan, semakin rumit pola di layar. Dengan mengubah parameter fungsi yang menggambar lingkaran, Anda dapat membuat pola yang lebih membingungkan.



Gambar 3.7 Kaleido-Spiral python

Bagaimana itu bekerja

Dalam proyek ini, Anda akan menggunakan modul kura-kura dan teknik perulangan yang cerdas untuk melapisi lingkaran di atas satu sama lain dalam pola spiral. Setiap kali lingkaran digambar, program sedikit meningkatkan parameter kode gambar lingkaran. Setiap lingkaran baru berbeda dari yang terakhir digambar, membuat polanya lebih menarik.

Cycle

Untuk membuat pola berwarna, proyek ini menggunakan fungsi yang disebut cycle() dari modul itertools. Fungsi cycle() memungkinkan Anda untuk menggilir daftar warna yang berbeda berulang kali. Ini memudahkan penggunaan warna pena yang berbeda untuk setiap lingkaran.

Dapatkan menggambar!

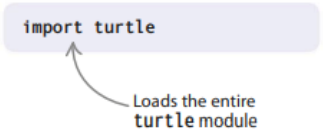
Hal pertama yang akan Anda gambar di layar adalah lingkaran sederhana. Selanjutnya Anda akan mengulangi lingkaran ini, tetapi dengan sedikit perubahan. Terakhir, Anda akan mengubah kode untuk membuat pola lebih berwarna dan menarik.

1. Buat file baru

   Buka IDLE dan buat file baru. Simpan sebagai "kaleido-spiral.py".

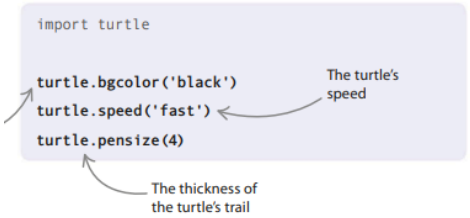
2. Impor Kura-kura

   Pertama, Anda perlu mengimpor modul kura-kura. Ini akan menjadi modul utama yang Anda gunakan. Ketik baris ini di bagian atas program.



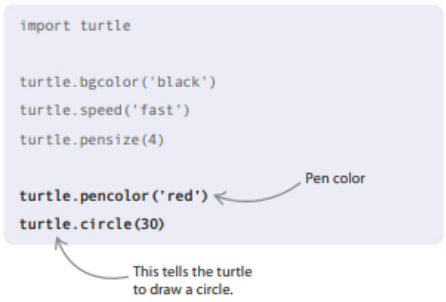
3. Siapkan kura-kura

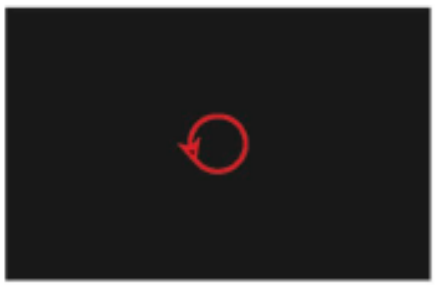
   Kode yang ditampilkan di sini memanggil fungsi dalam modul kura-kura untuk mengatur warna latar belakang, serta kecepatan dan ukuran kura-kura.



4. Pilih warna pena, gambar lingkaran

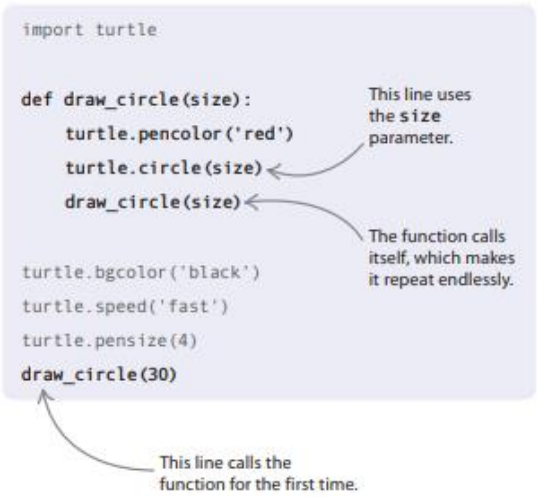
   Selanjutnya atur warna jejak kura-kura dan uji kodenya dengan menggambar lingkaran. Tambahkan dua baris ini ke akhir kode Anda dan jalankan program.





5. Gambar lebih banyak lingkaran

   Anda sekarang akan melihat satu lingkaran, tetapi kami membutuhkan lebih banyak lagi. Di sinilah sedikit pintar. Letakkan perintah untuk menggambar lingkaran merah di dalam suatu fungsi, tetapi tambahkan garis sehingga fungsi tersebut memanggil dirinya sendiri. Trik ini, yang dikenal sebagai rekursi, membuat fungsi berulang. Ingat, fungsi perlu didefinisikan sebelum digunakan, jadi Anda harus memindahkan fungsi di atas garis yang memanggilnya.



   Pengulangan

   Ketika suatu fungsi memanggil dirinya sendiri, ini dikenal sebagai rekursi. Ini adalah cara lain untuk membuat loop dalam program Anda. Dalam sebagian besar penggunaan rekursi, parameter fungsi berubah setiap kali fungsi dipanggil. Dalam Kaleido-spiral, misalnya, ukuran, sudut, dan posisi lingkaran berubah setiap kali fungsi memanggil dirinya sendiri.

6. Uji kode Anda

   Jalankan programnya. Anda akan melihat kura-kura menggambar lingkaran yang sama berulang-ulang. Jangan khawatir—Anda akan memperbaikinya di langkah berikutnya.

7. Ubah warnanya, tambah ukurannya

   Untuk membuat pola yang lebih menarik, buat perubahan ini pada kode untuk memperbesar ukuran lingkaran dan mengubah warnanya. Kode ini menggunakan fungsi cycle(), yang mengambil daftar nilai sebagai parameternya dan mengembalikan jenis daftar khusus yang dapat Anda putar tanpa henti menggunakan fungsi next(). Jalankan kode lagi.

```
import turtle
from itertools import cycle

colors = cycle(['red', 'orange', 'yellow', 'green', 'blue', 'purple'])

def draw_circle(size):
    turtle.pencolor(next(colors))
    turtle.circle(size)
    draw_circle(size + 5)

turtle.bgcolor('black')
turtle.speed('fast')
turtle.pensize(4)
draw_circle(30)
```

Import the `cycle()` function.

This line creates a cycle of the colors in the list.

Use the next color in the cycle.

Add 5 to the previous circle size.

8. Perbaiki polanya

   Sekarang setelah Anda mengubah warna dan ukuran lingkaran, Anda dapat mencoba beberapa hal lagi untuk memperbaiki polanya. Mari kita beri sentuhan lucu dengan mengubah sudut dan posisi di mana setiap lingkaran digambar. Buat perubahan yang disorot dalam kode di bawah ini, lalu jalankan program dan lihat apa yang terjadi.

```
def draw_circle(size, angle, shift):
    turtle.pencolor(next(colors))
    turtle.circle(size)
    turtle.right(angle)
    turtle.forward(shift)
    draw_circle(size + 5, angle + 1, shift + 1)

turtle.bgcolor('black')
turtle.speed('fast')
turtle.pensize(4)
draw_circle(30, 0, 1)
```

Add these new parameters.

The turtle turns clockwise.

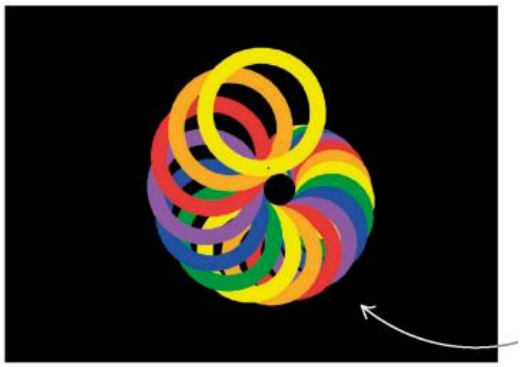
The turtle moves forward.

The angle and shift increase with every circle drawn.

Set the starting values of the new parameters.

3.3 PERETASAN DAN PENYESUAIAN

Setelah semuanya bekerja dengan lancar, Anda dapat bermain-main dengan kode dan membuat polanya menjadi lebih fantastis.



Gambar 3.8 Spiral Pena Tebal

Pena tebal

Coba tambah ukuran pena dan lihat apa pengaruhnya pada pola Anda. Anda awalnya mengaturnya ke 4 dengan kode di bawah ini. Seperti apa tampilan 40?

```
turtle.pensize(40)
```

Penuh Warna

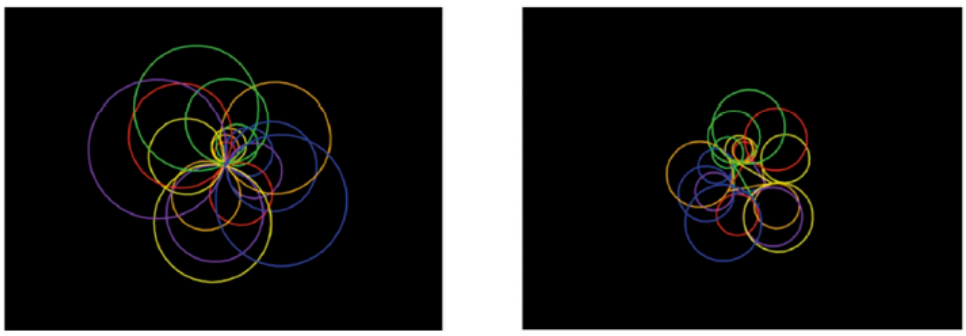
Bagaimana jika Anda mengubah warna latar belakang pada setiap loop, serta warna pena? Ini mungkin memberi Anda beberapa hasil yang liar! Untuk mengubah warna latar belakang setiap kali, pindahkan garis yang menyetelnya ke dalam fungsi draw_circle(). Anda juga harus menggunakan siklus warna untuk memilih warna baru pada setiap loop.

```
def draw_circle(size, angle, shift):
    turtle.bgcolor(next(colors))
    turtle.pencolor(next(colors))
    turtle.circle(size)
    turtle.right(angle)
    turtle.forward(shift)
    draw_circle(size + 5, angle + 1, shift + 1)

turtle.speed('fast')
turtle.pensize(4)
draw_circle(30, 0, 1)
```

The background color is now set inside the loop.

Temukan pola baru

Tampilan pola ditentukan oleh seberapa banyak Anda menambahkan parameter fungsi setiap kali dipanggil. Coba tambahkan lebih banyak atau lebih sedikit pada ukuran, pergeseran, dan sudut daripada yang Anda lakukan saat ini, untuk mengetahui bagaimana perubahan ini memengaruhi pola.



Size +10, angle +10, shift +1 | Size +5, angle –20, shift –10

Gambar 3.9 Perbandingan 2 Pola dengan letak tertentu

Perubahan bentuk

Bagaimana polanya jika program dapat menggambar bentuk lain dan juga lingkaran? Menambahkan persegi setiap waktu dapat menciptakan pola yang menarik. Berikut beberapa kode untuk membantu Anda. Hati-hati—nama fungsi telah berubah!

```
import turtle
from itertools import cycle

colors = cycle(['red', 'orange', 'yellow', 'green', 'blue', 'purple'])

def draw_shape(size, angle, shift, shape):
    turtle.pencolor(next(colors))
    next_shape = ''
    if shape == 'circle':
        turtle.circle(size)
        next_shape = 'square'
    elif shape == 'square':
        for i in range(4):
            turtle.forward(size * 2)
            turtle.left(90)
        next_shape = 'circle'
    turtle.right(angle)
    turtle.forward(shift)
    draw_shape(size + 5, angle + 1, shift + 1, next_shape)

turtle.bgcolor('black')
turtle.speed('fast')
turtle.pensize(4)
draw_shape(30, 0, 1, 'circle')
```

Add a new parameter, **shape**.

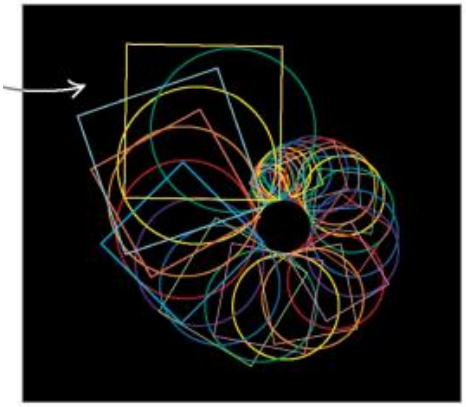
The loop runs 4 times, once for each side of the square.

The turtle rotates.

The turtle moves forward.

This makes the turtle alternate between circles and squares.

The first shape is a circle.



Gambar 3.10 Pola perubahan Bentuk

3.4 MALAM BERBINTANG

Isi layar Anda dengan bintang-bintang yang indah! Proyek ini menggunakan modul kura-kura Python untuk menggambar bentuk bintang. Angka acak menyebarkan bintang di atas layar dan memvariasikan warna, ukuran, dan bentuknya.

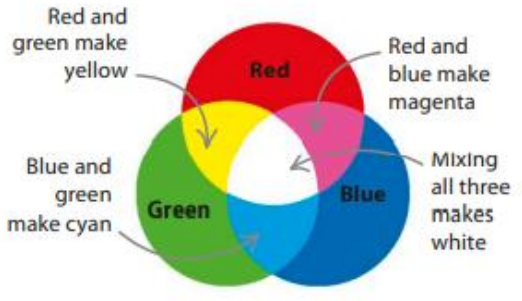
Apa yang terjadi

Pertama, langit malam hari digambar, lalu satu bintang muncul di langit. Saat program berlanjut, langit mulai dipenuhi dengan lebih banyak bintang dalam berbagai gaya yang berbeda. Semakin lama Anda membiarkan program berjalan, semakin fantastis dan berwarna langit menjadi.

Gambar 3.11 Jenis dan Bentuk Bintang

Membuat warna

Gambar dan grafik pada layar komputer terdiri dari titik-titik kecil yang disebut piksel, yang dapat mengeluarkan cahaya merah, hijau, dan biru. Dengan mencampur warna-warna ini bersama-sama, Anda dapat membuat warna apa pun yang bisa dibayangkan. Dalam proyek ini, warna setiap bintang disimpan sebagai tiga angka. Angka-angka mewakili jumlah cahaya merah, hijau, dan biru yang digabungkan untuk memberikan warna akhir.



Gambar 3.12 Pencampuran Warna RGB

Layar penuh bintang

Proyek malam berbintang akan menggambar bintang satu per satu, tetapi karena menggunakan *loop while* tak terbatas, itu akan menggambar bintang selamanya! Anda dapat mengubah rentang ukuran bintang dengan menyesuaikan batas angka acak dalam kode.

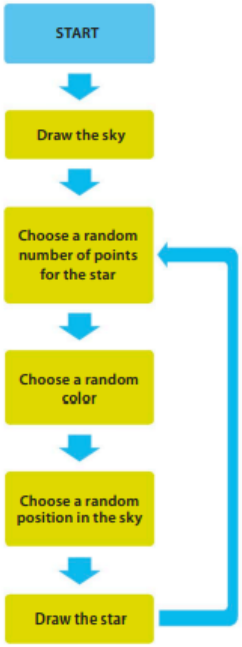
Gambar 3.13 pembatasan rentang ukuran dan warna bintang yang dibuat

Bagaimana itu bekerja

Kode untuk proyek ini menggambar bentuk bintang di lokasi acak di jendela Turtle Graphics. Anda akan menulis kode Python untuk membuat fungsi yang dapat menggambar satu bintang. Kemudian Anda akan membuat lingkaran yang mengulanginya berulang-ulang, menggambar banyak bintang berbeda di seluruh layar.

Bagan alir Malam Berbintang

Flowchartnya cukup sederhana, tanpa pertanyaan atau keputusan yang harus dibuat. Setelah kura-kura menggambar bintang pertama, program akan mengulang kembali dan mengulangi langkah menggambar bintang tanpa henti sampai Anda berhenti.



Gambar 3.14 Diagram Alur membuat Bintang

Menghitung bintang

Pada malam yang cerah ada sekitar 4.500 bintang yang terlihat di langit. Agar program Anda menarik banyak bintang, Anda harus membiarkannya berjalan selama lebih dari 3 jam.

Menggambar bintang

Sebelum Anda membuat fungsi Anda, Anda perlu mengetahui cara menggambar bintang di kura-kura. Setelah Anda menguasainya, Anda akan dapat membuat sisa kode untuk proyek tersebut.

1. Buat file baru

   Buka IDLE. Masuk ke menu File, lalu pilih File Baru. Simpan file sebagai "starry_night.py".

2. Impor Kura-kura

   Ketik baris ini ke dalam jendela editor yang muncul. Ini memuat modul kura-kura, siap untuk Anda mulai menggambar bintang Anda.

```
import turtle as t
```

Loads the turtle

3. Tulis beberapa instruksi

   Sekarang tambahkan kode ini di bawah perintah untuk mengimpor penyu. Ini menciptakan variabel yang mengatur ukuran dan bentuk bintang. Ini juga memberi tahu kura-kura cara bergerak melewati jendela untuk menggambar bintang.

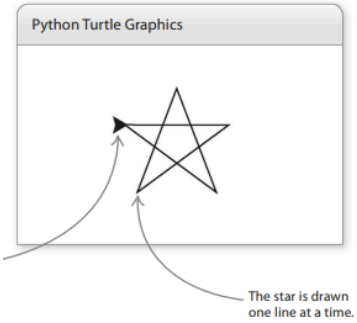
```
import turtle as t

size = 300
points = 5
angle = 144

for i in range(points):
    t.forward(size)
    t.right(angle)
```

These are the instructions for the size and shape of the star.

This **for** loop makes the turtle repeat the same movement for each point of the star.

4. Gambarlah bintang percobaan

   Dari menu IDLE, pilih Jalankan lalu Jalankan Modul untuk menguji proyek. Jendela Turtle Graphics akan muncul (mungkin di belakang jendela lain) dan Anda akan melihat panah kura-kura mulai menggambar bintang Anda.



Gambar 3.15 PopUp Python Turtle Graphics

5. Tambahkan kalkulator sudut
   Akan lebih baik untuk dapat menggambar bintang dengan jumlah poin yang berbeda. Buat perubahan ini pada kode. Ini akan menghitung sudut belokan yang harus dibuat kura-kura untuk menggambar bintang dengan banyak titik yang Anda pilih.

```
import turtle as t

size = 300
points = 5
angle = 180 - (180 / points)

for i in range(points):
    t.forward(size)
    t.right(angle)
```

The angle depends on the number of points the star has.

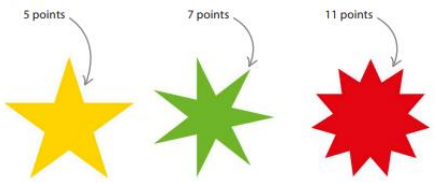
6. Warnai itu!
   Anda telah menggambar bintang yang bagus dan rapi, tetapi saat ini terlihat agak kusam. Mari tambahkan beberapa warna agar lebih menarik. Ubah kode seperti yang ditunjukkan di sebelah kanan untuk mengecat kuning bintang Anda.
7. Jalankan proyek
   Kura-kura harus menggambar bintang kuning. Lihat apakah Anda dapat mengubah warna bintang dengan mengedit kode.

```
import turtle as t

size = 300
points = 5
angle = 180 - (180 / points)

t.color('yellow')
t.begin_fill()
for i in range(points):
    t.forward(size)
    t.right(angle)

t.end_fill()
```

This sets the star's color to yellow.

This fills the star with color.

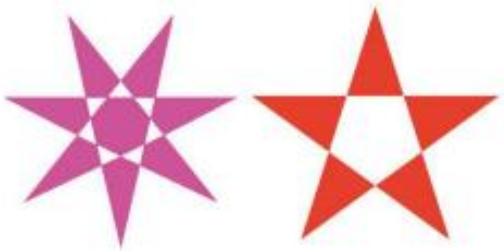
8. Gambar bintang yang berbeda
   Coba ubah angka setelah tanda sama dengan di titik variabel dan Anda akan melihat bahwa Anda dapat menggambar bintang yang berbeda. Perhatikan bahwa kode hanya berfungsi untuk bintang dengan jumlah poin ganjil. Angka genap akan mengacaukan segalanya.



Gambar 3.16 bentuk dan warna bintang yang berbeda

Bintang berlubang

Pada beberapa komputer, bintang Anda mungkin terlihat sedikit berbeda atau bahkan memiliki lubang di tengahnya. Tampilan Grafik Turtle Python dapat bervariasi tergantung pada jenis komputer yang Anda gunakan, tetapi ini tidak berarti bahwa kode Anda salah.



Gambar 3.17 Bintang Macam yang berlubang

Langit berbintang

Langkah selanjutnya akan membungkus bintang Anda sebagai fungsi Python. Anda kemudian dapat menggunakan fungsi itu untuk menggambar langit yang penuh dengan bintang.

9. Buat fungsi bintang

   Edit kode seperti yang ditunjukkan di sini. Ini menggantikan hampir semua kode yang ada dengan versi baru. Blok besar membungkus semua instruksi menggambar bintang dan menyimpannya dengan rapi sebagai sebuah fungsi. Anda sekarang dapat menggunakan fungsi ini untuk menggambar bintang di kode utama Anda dengan satu baris Python, draw_star().

```
import turtle as t

def draw_star(points, size, col, x, y):
    t.penup()
    t.goto(x, y)
    t.pendown()
    angle = 180 - (180 / points)
    t.color(col)
    t.begin_fill()
    for i in range(points):
        t.forward(size)
        t.right(angle)
    t.end_fill()

# Main code
t.Screen().bgcolor('dark blue')
draw_star(5, 50, 'yellow', 0, 0)
```

The x and y coordinates set the position of the star on the screen.

This sets the background color to dark blue.

The turtle draws a yellow, five-pointed star, size 50, in the center of the window.

10. Jalankan proyek

    Kura-kura harus menggambar satu bintang kuning dengan latar belakang biru.

Gambar 3.18 Jendela Bintang yang akan dibuat

Komentar

Pemrogram sering memberikan komentar dalam kode mereka untuk mengingatkan mereka apa yang dilakukan oleh bagian yang berbeda dari suatu program atau untuk menjelaskan bagian yang rumit dari suatu proyek. Komentar harus dimulai dengan #. Python mengabaikan apa pun yang Anda ketik di baris yang sama setelahnya # dan tidak memperlakukannya sebagai bagian dari kode. Menulis komentar di proyek Anda sendiri (seperti baris # Kode utama yang ditunjukkan di atas) dapat sangat membantu ketika Anda kembali untuk melihat program setelah meninggalkannya sebentar.

11. Tambahkan nomor acak

    Sekarang campur semuanya dengan menambahkan beberapa angka acak ke kode Anda. Ketik baris ini di bawah baris yang mengimpor kura-kura. Ini membawa fungsi randint() dan random() dari modul acak Python.

```
import turtle as t
from random import randint, random

def draw_star(points, size, col, x, y):
```

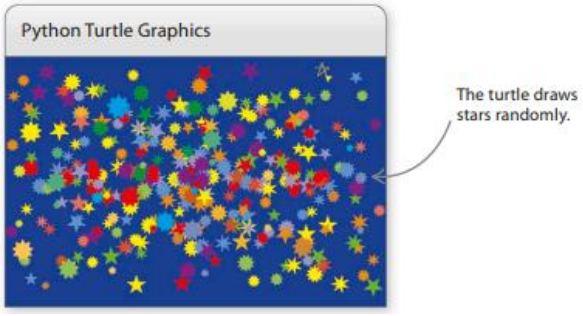
12. Buat lingkaran

    Lakukan perubahan ini pada bagian #Kode utama. Itu menambahkan loop sementara yang terus-menerus mengacak parameter yang digunakan untuk mengatur ukuran, bentuk, warna, dan posisi bintang.

```
# Main code
t.Screen().bgcolor('dark blue')

while True:
    ranPts = randint(2, 5) * 2 + 1
    ranSize = randint(10, 50)
    ranCol = (random(), random(), random())
    ranX = randint(-350, 300)
    ranY = randint(-250, 250)

    draw_star(ranPts, ranSize, ranCol, ranX, ranY)
```

13. Jalankan proyek lagi

    Jendela harus perlahan terisi saat kura-kura menggambar bintang demi bintang dalam berbagai warna, bentuk, dan ukuran.



Gambar 3.19 menggambar bintang secara acak

3.5 KURA-KURA TAK TERLIHAT

Jika Anda lebih suka tidak melihat kura-kura, ingatlah ada perintah yang dapat Anda gunakan untuk membuatnya tidak terlihat. Tambahkan baris ini ke program Anda dan bintang Anda akan muncul secara ajaib, digambar oleh kura-kura yang tidak terlihat!

```
# Main code
t.hideturtle()
```

Peretasan dan penyesuaian

Anda sekarang dapat membuat bintang sesuai permintaan. Mengapa tidak mencoba menggunakan kode draw_star() di proyek Anda sendiri. Berikut adalah beberapa ide.

Ubah bintang Anda

Untuk mengubah seberapa bervariasi tampilan bintang Anda, ubah angka dalam kurung variabel ranPts dan ranSize dalam loop while.

Rancang rasi bintang

Rasi bintang adalah pola bintang di langit malam. Coba buat daftar posisi (x, y) untuk bintang di konstelasi desain Anda sendiri. Kemudian gunakan for loop untuk menggambar bintang di lokasi tersebut.

Gambar beberapa planet

Selidiki fungsi turtle.circle() dan lihat apakah Anda dapat menggunakannya untuk membuat beberapa kode gambar planet. Berikut beberapa kode untuk membantu Anda memulai.

```
def draw_planet(col, x, y):
    t.penup()
    t.goto(x, y)
    t.pendown()
    t.color(col)
    t.begin_fill()
    t.circle(50)
    t.end_fill()
```

Klik untuk bintang

Daripada membiarkan kura-kura menggambar bintang secara acak, coba gunakan fungsi turtle.onScreenClick() untuk menggambar bintang di mana pun Anda mengklik dengan mouse.

Percepat kura-kura

Anda dapat mengubah seberapa cepat kura-kura menggambar bintang dengan membuat fungsi speed(). Cukup tambahkan t.speed(0) di awal kode utama untuk memberikan lebih banyak zip pada turtle. Anda dapat melihat semua fungsi modul kura-kura di bagian "Bantuan" Python.

3.6 PELANGI ACAK

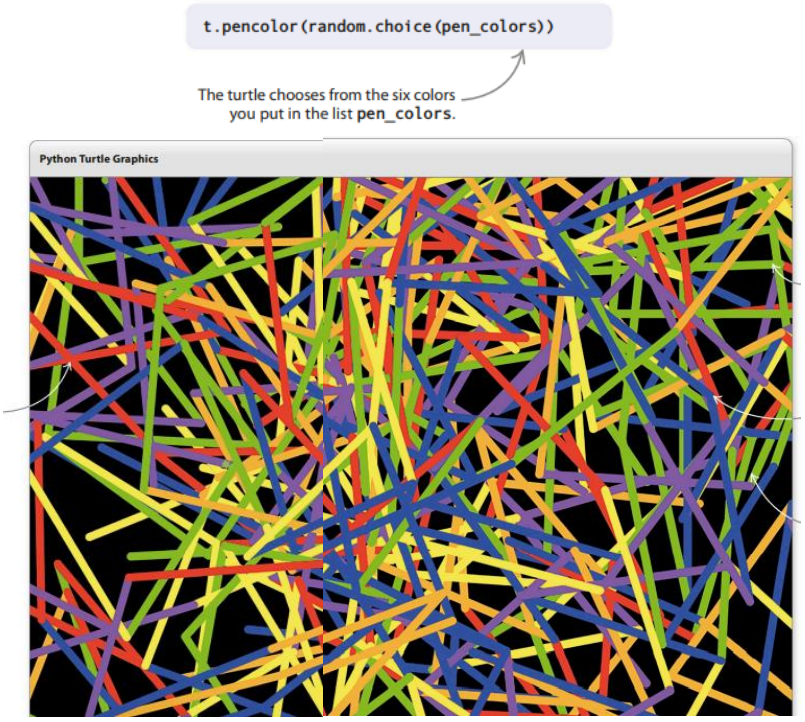
Anda dapat memprogram kura-kura Python untuk menggambar segala macam pola dan desain. Tapi hati-hati! Sepertinya kura-kura dalam proyek ini menjadi sedikit liar— Anda tidak akan melihat pelangi seperti ini di langit!

Apa yang terjadi

Program akan meminta Anda untuk memilih panjang dan ketebalan garis yang dilukis oleh kura-kura. Kura-kura kemudian berlarian di sekitar layar sampai Anda menghentikan program, melukis garis-garis berwarna saat berjalan. Jenis pola yang dibuat akan berubah, tergantung pada panjang dan ketebalan garis.

Warna apa selanjutnya?

Di Mutant Rainbow, Anda akan menggunakan fungsi choice() dari modul acak Python untuk memilih warna saat Anda memberi tahu kura-kura untuk menggambar garis. Ini berarti Anda tidak dapat benar-benar memprediksi warna mana yang akan digunakan kura-kura setiap saat.



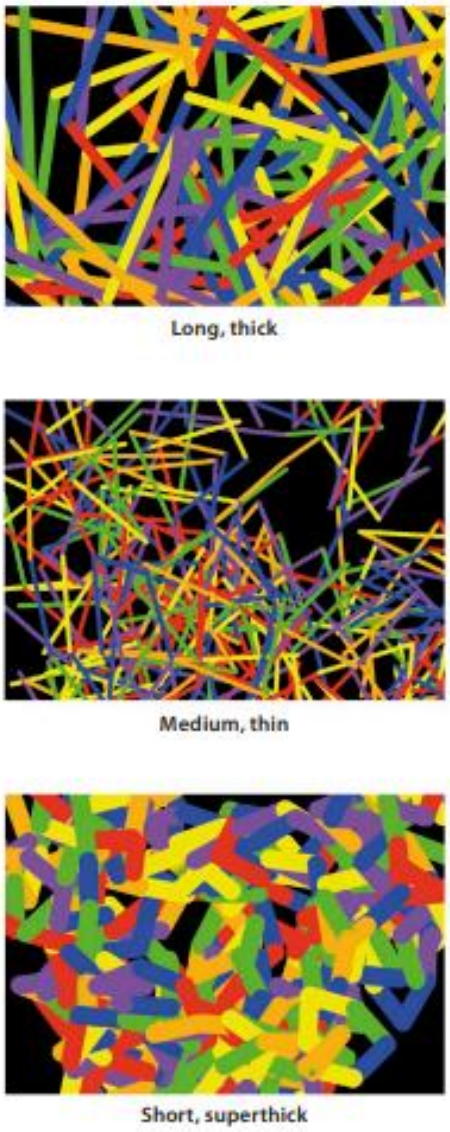
Gambar 3.20 Macam-macam pena berwarna

Tampilan warna

Karena program ini menggunakan perulangan while tak terbatas, kura-kura terus menggambar sampai Anda menutup jendelanya. Anda tidak hanya dapat mengubah warna, lebar, dan panjang garis, tetapi juga bentuk, warna, dan kecepatan kura-kura itu sendiri.

Bagaimana itu bekerja

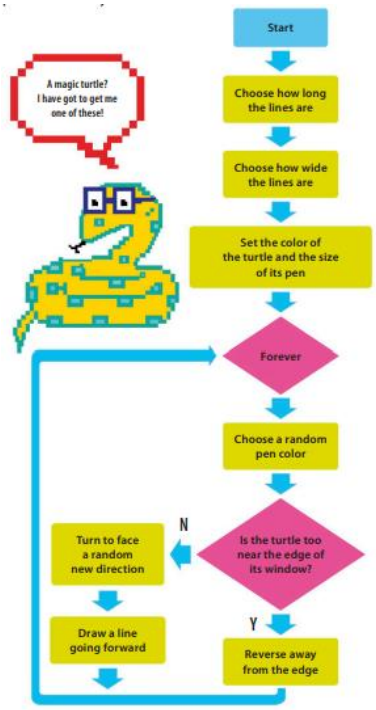
Setiap pola dalam proyek ini berbeda karena program memberitahu kura-kura untuk menghadap ke arah baru secara acak sebelum melukis setiap garis. Warna untuk setiap baris juga dipilih secara acak dari daftar kemungkinan warna yang telah Anda kodekan. Jadi Anda tidak akan pernah bisa memprediksi dengan tepat apa yang akan dilakukan kura-kura!



Gambar 3.21 Macam-macam ketebalan Pena

Diagram alir Pelangi Mutan

Program menggunakan loop tak terbatas yang terus melukis garis berwarna selama program berjalan. Hanya ketika Anda menutup jendela, kura-kura akan menghentikan pengembaraannya yang gila.



Gambar 3.22 Diagram alur Pelangi Mutan

Kura-kura pelarian!

Diberi kebebasan penuh untuk berkeliaran, kura-kura cenderung berkeliaran di luar jendela. Saat Anda menyusun program, Anda akan menulis beberapa kode untuk memeriksa posisi kura-kura dan menghentikannya agar tidak menyimpang terlalu jauh. Jika tidak, ini akan berubah menjadi proyek penyu yang menghilang!

Memulai

Mulailah dengan menyiapkan dan menyimpan file baru, mengimpor modul yang dibutuhkan program, dan membuat beberapa fungsi yang berguna untuk mendapatkan masukan pengguna.

1. Buat file baru

   Buka IDLE dan buat file baru. Simpan sebagai "pelangi.py".

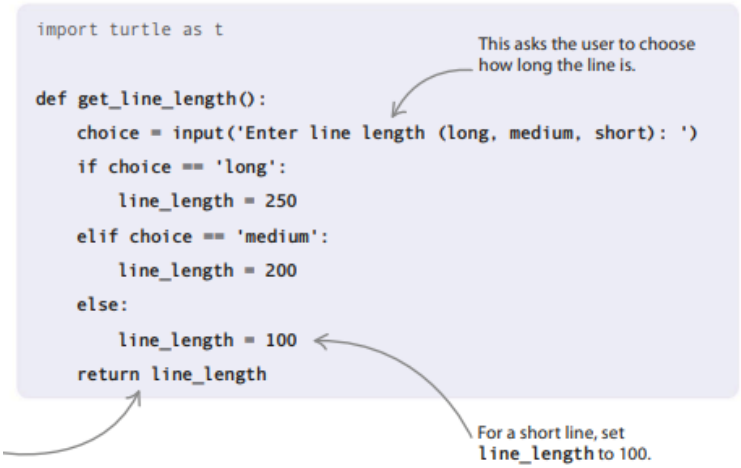
2. Tambahkan modul

   Ketik dua baris ini di bagian atas file Anda untuk mengimpor modul Turtle dan modul acak. Ingatlah untuk menggunakan import turtle sebagai t, sehingga Anda tidak perlu mengetikkan kata “turtle” setiap kali ingin menggunakan fungsi dari modul Turtle. Anda bisa menyebutnya t.

```
import random
import turtle as t
```

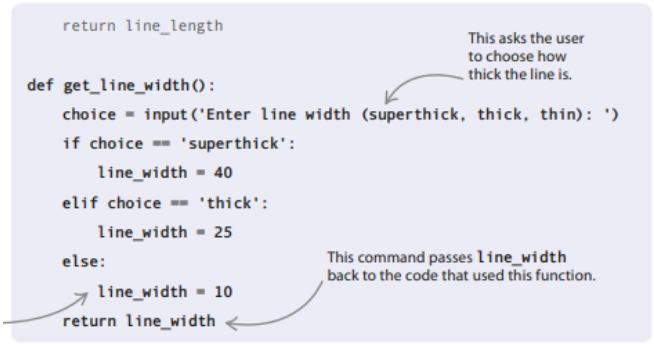
3. Tetapkan panjang garis

   Selanjutnya buat fungsi yang akan membiarkan pengguna memutuskan apakah kura-kura melukis garis panjang, sedang, atau pendek. Anda tidak akan menggunakannya sampai Langkah 4, tetapi ini akan membuat program siap saat Anda membutuhkannya. Ketik sedikit kode ini di bawah kode pada Langkah 1.



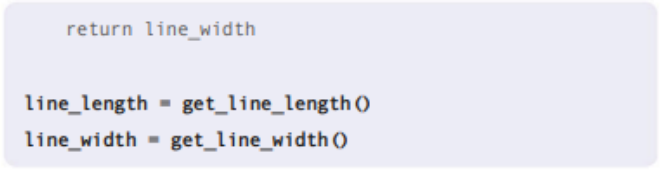
4. Tentukan ketebalan

   Pada langkah ini, Anda akan membuat fungsi yang memungkinkan pengguna memilih apakah kura-kura melukis garis super tebal, tebal, atau tipis. Seperti fungsi get_line_length(), Anda tidak akan menggunakannya hingga Langkah 5. Ketik kode yang ditampilkan di sini, di bawah kode yang Anda tambahkan di Langkah 3.



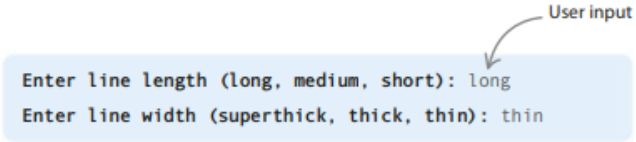
5. Gunakan fungsi

   Sekarang setelah Anda membangun dua fungsi, Anda dapat menggunakannya untuk mendapatkan pilihan pengguna untuk panjang dan lebar garis. Ketik baris ini di akhir kode Anda, lalu simpan pekerjaan Anda.



6. Uji programnya

   Jalankan kode untuk melihat fungsi baru beraksi di shell. Mereka akan meminta Anda untuk memilih panjang dan lebar garis.



Panggil kura-kura!

Saatnya menulis kode yang akan membuat jendela grafik dan membawa kura-kura untuk menggambar.

7. Buka jendela

   Ketik garis yang ditampilkan di sini di bawah kode yang Anda tambahkan pada Langkah 5. Kode ini mendefinisikan warna latar belakang jendela, bentuk, warna, dan kecepatan kura-kura, dan lebar pena yang akan digunakan kura-kura untuk menggambar garis.

```
line_width = get_line_width()

t.shape('turtle')
t.fillcolor('green')
t.bgcolor('black')
t.speed('fastest')
t.pensize(line_width)
```

This makes the turtle green.

This sets the background to black.

This sets the turtle's speed.

8. Jalankan proyek

   Jika Anda menjalankan kode sekali lagi, sebuah jendela akan muncul setelah Anda memasukkan ukuran garis di jendela shell. Anda sekarang akan melihat kura-kura. Perhatikan baik-baik, karena itu tidak akan duduk diam terlalu lama!

9. Tetap dalam batas!

   Untuk menghentikan kura-kura agar tidak tersesat, mari kita tetapkan batas 100 langkah dari tepi jendela. Buat fungsi ini untuk memeriksa apakah kura-kura berada di dalam batas atau tidak. Ketik kode yang ditunjukkan di sini di bawah kode di Langkah 4 dan di atas kode di Langkah 5.

```
    return line_width


def inside_window():
    left_limit = (-t.window_width() / 2) + 100
    right_limit = (t.window_width() / 2) - 100
    top_limit = (t.window_height() / 2) - 100
    bottom_limit = (-t.window_height() / 2) + 100
    (x, y) = t.pos()
    inside = left_limit < x < right_limit and bottom_limit < y < top_limit
    return inside


line_length = get_line_length()
```

This is 100 steps to the right of the left edge.

100 steps from the right edge

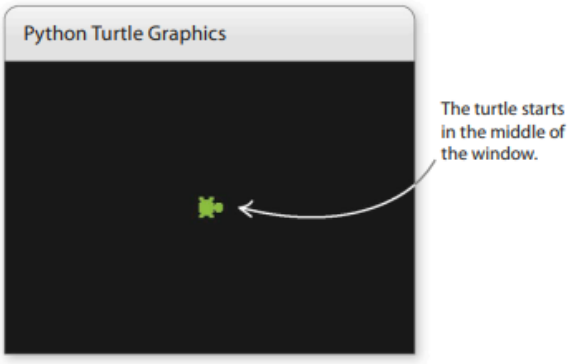
100 steps from the top

100 steps from the bottom

This sets **inside** to True if the turtle is inside the limits and False if it isn't.

With this line, the program gets the turtle's current x and y coordinates.

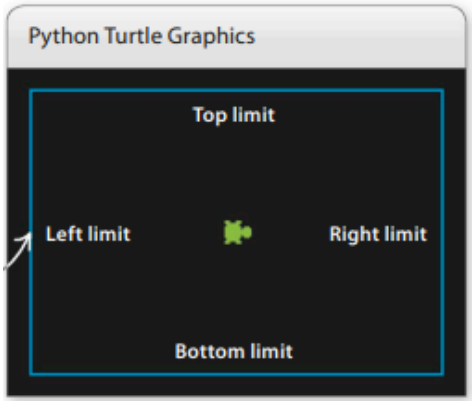
This command passes **inside** back to the code that used this function.



Gambar 3.23 Hasil dari list program

Cara kerjanya

Kode memeriksa apakah koordinat x kura-kura berada di antara batas kanan dan kiri, dan apakah koordinat y berada di antara batas atas dan bawah.



Gambar 3.24 memeriksa letak bintang yang muncul

Pindahkan kura-kura itu!

Sekarang Anda siap untuk menulis fungsi yang membuat kura-kura Anda bergerak. Bagian terakhir dari kode akan menjadi loop sementara yang membuat kura-kura menggambar pelangi mutan!

10. Garis mutan

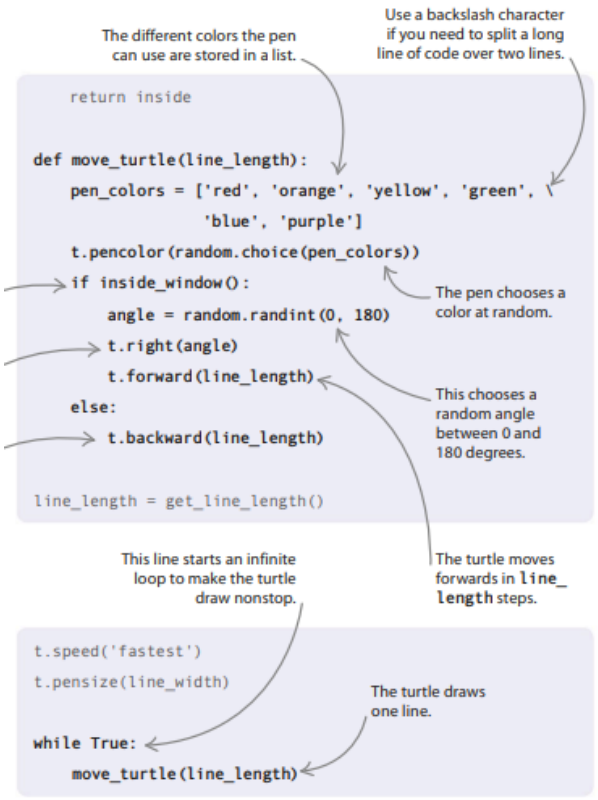
    Tambahkan kode ini di bawah kode yang Anda ketik di Langkah 9, dan di atas kode yang Anda ketik di Langkah 5. Fungsi ini membuat kura-kura berputar dan bergerak maju ke arah yang baru, menggambar satu baris warna acak saat berjalan. Program utama Anda akan menggunakannya berulang kali untuk menggambar pelangi mutan. Jika kura-kura tersesat di luar batas yang Anda tetapkan di Langkah 9, fungsi ini akan mengembalikannya.

    Cara kerjanya

    Kode memanggil fungsi inside_window() untuk melihat apakah kura-kura berada dalam batas jendela. Jika ya, kura-kura berbelok ke kanan dengan jumlah acak antara 0 derajat (tidak berbelok sama sekali) dan 180 derajat (menghadap ke arah yang berlawanan), lalu bergerak lagi. Jika sudah terlalu jauh dari batas, ia bergerak mundur.

11. Pergi, Kura-Kura, Pergi!

    Terakhir, tambahkan kode yang sebenarnya akan memulai menggambar kura-kura Anda. Ketik dua baris ini tepat di bagian bawah kode Anda, di bawah perintah yang Anda tambahkan di Langkah 7. Kemudian simpan dan jalankan kode untuk melihat pelangi mutan pertama Anda!

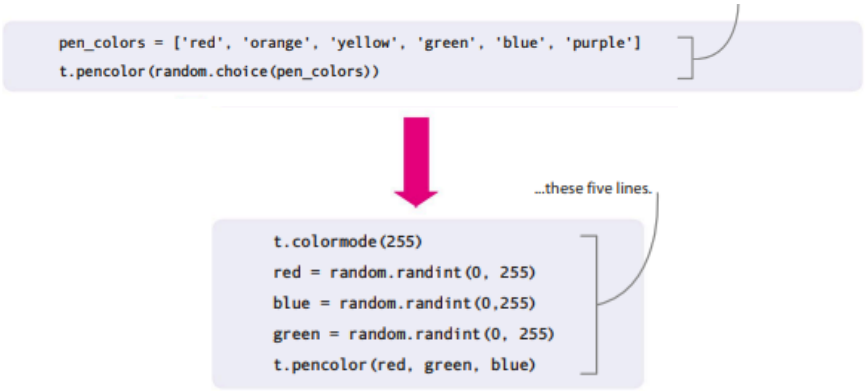


Peretasan dan penyesuaian

Apakah mutan pelangi Anda cukup? Tidak? Berikut adalah beberapa ide yang bisa Anda coba untuk membuatnya lebih aneh!

Kejutan warna!

Dalam Python, warna juga dapat dideskripsikan dengan menggunakan nilai RGB— ini singkatan dari merah, hijau, biru. Memilih nilai secara acak untuk jumlah merah, hijau, dan biru dalam suatu warna berarti warna itu sendiri akan sepenuhnya acak. Coba ganti kode di fungsi move_turtle() dengan beberapa kode baru yang menggunakan nilai RGB alih-alih nama warna. Sekarang jalankan kode untuk melihat warna apa yang muncul!



Campurkan garis

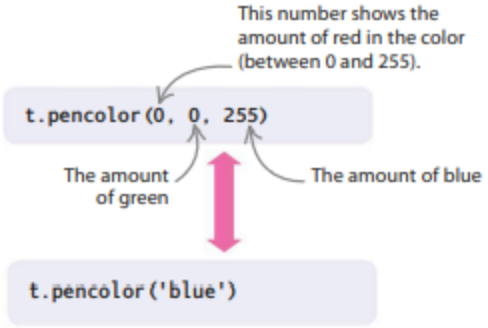
Jangan hanya berpegang pada satu lebar untuk garis—gambar pelangi yang lebih acak dengan retasan ini! Garis akan berubah secara acak dari sangat tipis menjadi sangat

tebal dan semua lebar di antaranya. Tambahkan kode ini ke fungsi move_ turtle() setelah Anda menyetel t.pencolor.

```
t.pensize(random.randint(1,40))
```

3.7 WARNA RGB

Dalam kura-kura, warna "biru" adalah (0, 0, 255) dalam nilai RGB, karena terdiri dari jumlah maksimum biru, tanpa merah atau hijau. Jika Anda ingin menggunakan nilai RGB untuk warna pena kura-kura, Anda perlu memberi tahu Python dengan menggunakan perintah t.colormode(255), atau ia akan mengharapkan string dan memberi Anda kesalahan.



Cap kura-kura!

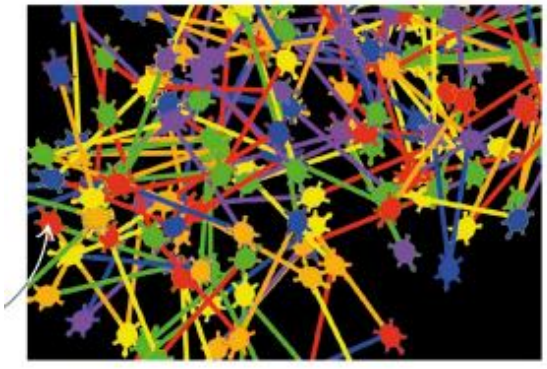
"Keling" garis pelangi Anda bersama-sama dengan menggunakan fungsi stamp() modul turtle untuk menambahkan gambar turtle ke awal setiap baris. (Anda juga dapat menulis sebuah fungsi untuk menggambar garis yang seluruhnya terdiri dari kura-kura yang dicap dan menggunakannya sebagai ganti t.forward dan t.backward.) Tambahkan baris kode baru ini ke fungsi move_turtle(), setelah perintah pena, ke mulai memukau.

```
def move_turtle(line_length):
    pen_colors = ['red', 'orange', 'yellow', 'green', 'blue', 'purple']
    t.pencolor(random.choice(pen_colors))
    t.fillcolor(random.choice(pen_colors))
    t.shapesize(3,3,1)
    t.stamp()
    if inside_window():
```

This sets the color of the turtle to a random color.

Type this to stamp a turtle picture on the screen.

This makes the turtle three times bigger than usual.



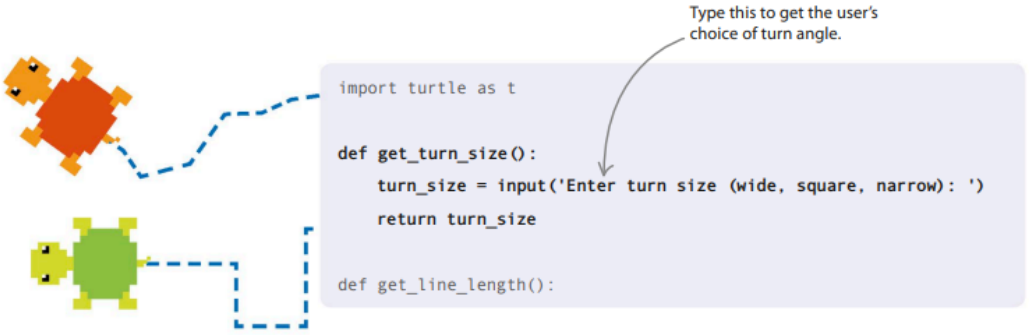
Gambar 3.25 Bintang berekor dengan variasi warna

Belokan besar atau kecil?

Anda dapat menambahkan prompt yang memungkinkan pengguna untuk memutuskan sudut belokan yang dibuat kura-kura. Mereka bisa lebar, persegi, atau sempit. Ikuti langkah-langkah ini untuk melihat bagaimana ini mengubah pola.

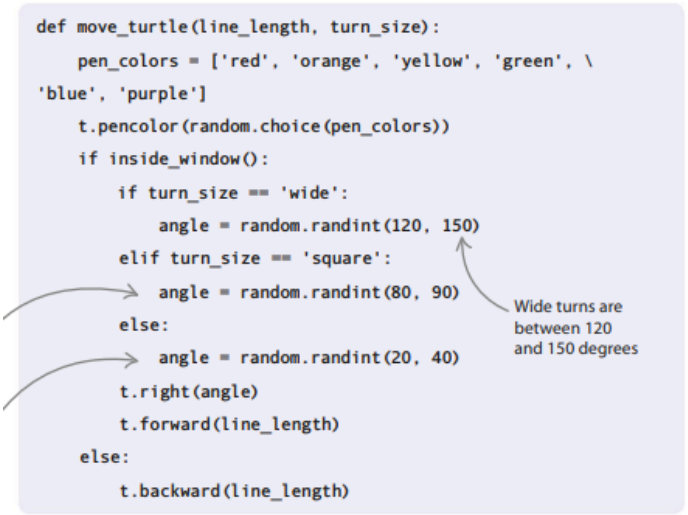
1. Buatlah fungsi

   Buat fungsi yang memungkinkan pengguna memilih ukuran belokan. Tambahkan ini di atas fungsi get_line_length() yang Anda tambahkan di Langkah 3 proyek utama.



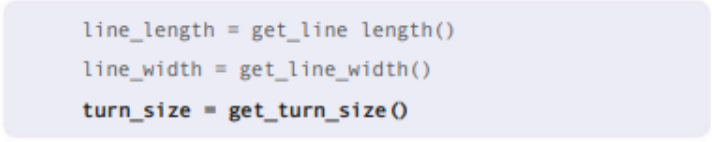
2. Gerakan yang berbeda

   Ganti fungsi move_turtle() dengan versi baru yang ditampilkan di sini. Itu menambahkan turn_size ke nilai yang Anda berikan ke fungsi saat Anda menggunakannya. Itu juga menggantikan sudut garis = acak. randint(0, 180) dengan kode yang memilih derajat berbeda untuk berbelok tergantung pada nilai turn_size.



3. Masukan pengguna

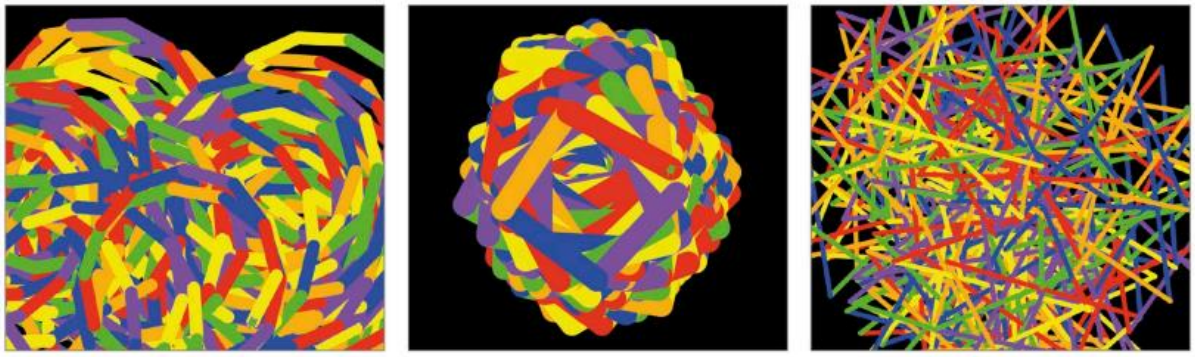
   Selanjutnya tambahkan baris ke bagian utama program untuk menggunakan fungsi get_turn_size() untuk mendapatkan pilihan ukuran giliran pemain.



4. Program utama

   Terakhir, ubah baris tempat Anda menggunakan fungsi move_turtle() untuk menyertakan turn_size.

```
while True:
        move_turtle(line_length, turn_size)
```



Gambar 3.26 Warna dan Ketebalan Pena yang berbeda

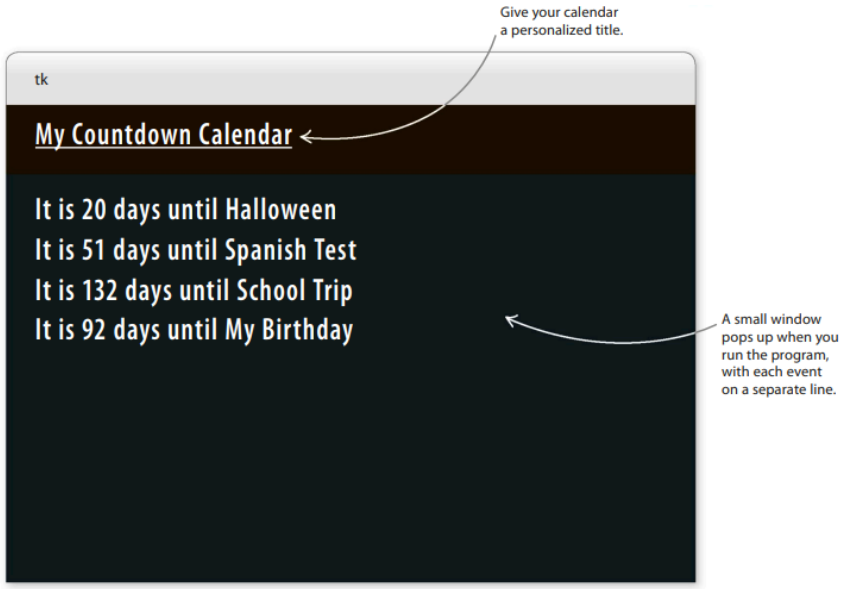
BAB 4
APLIKASI BERMAIN

4.1 KALENDER HITUNG MUNDUR

Saat Anda menantikan acara yang menarik, ada baiknya Anda mengetahui berapa lama lagi Anda harus menunggu. Dalam proyek ini, Anda akan menggunakan modul Tkinter Python untuk membangun program praktis yang menghitung mundur hingga hari besar.

Apa yang terjadi

Saat Anda menjalankan program, program ini menampilkan daftar acara mendatang dan memberi tahu Anda berapa hari yang tersisa hingga setiap acara. Jalankan lagi keesokan harinya dan Anda akan melihat bahwa itu telah mengurangi satu hari dari masing-masing angka "hari sampai". Isi dengan tanggal petualangan Anda yang akan datang dan Anda tidak akan pernah melewatkan hari penting—atau tenggat waktu pekerjaan rumah—lagi!



Gambar 4.1 Kalender Perhitungan Mundur

Bagaimana itu bekerja

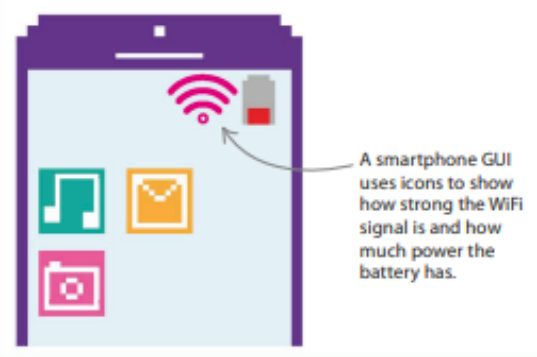
Program mempelajari tentang peristiwa penting dengan membaca informasi dari file teks—ini disebut "input file". File teks berisi nama dan tanggal setiap acara. Kode menghitung jumlah hari dari hari ini hingga setiap acara menggunakan modul datetime Python. Ini menampilkan hasil di jendela yang dibuat oleh modul Tkinter Python.

Menggunakan Tkinter

Modul Tkinter adalah seperangkat alat yang digunakan programmer Python untuk menampilkan grafik dan mendapatkan input dari pengguna. Alih-alih menampilkan output di shell, Tkinter dapat menampilkan hasil di jendela terpisah yang dapat Anda rancang dan gaya sendiri.

4.2 ANTARMUKA PENGGUNA GRAFIS

Tkinter berguna untuk membuat apa yang oleh pembuat kode disebut GUI (diucapkan "lengket"). GUI (antarmuka pengguna grafis) adalah bagian yang terlihat dari program yang berinteraksi dengan seseorang, seperti sistem ikon dan menu yang Anda gunakan pada ponsel cerdas. Tkinter membuat jendela sembulan yang dapat Anda tambahkan tombol, penggeser, dan menu.



Gambar 4.2 Indikator Baterai dan kuat sinyal WiFi

Diagram Alir Kalender Hitung Mundur

Dalam proyek ini, daftar peristiwa penting dibuat secara terpisah dari kode sebagai file teks. Program dimulai dengan membaca semua kejadian dari file ini. Setelah semua hari dihitung dan ditampilkan, program berakhir.

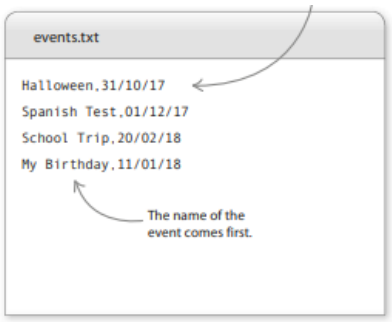
Gambar 4.3 Diagram Alir Kalender hitung mundur

Membuat dan membaca file teks

Semua informasi untuk Kalender Hitung Mundur Anda harus disimpan dalam file teks. Anda dapat membuatnya menggunakan IDLE.

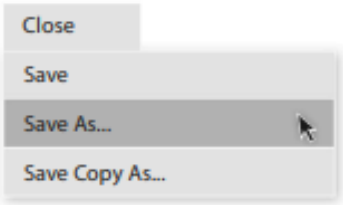
1. Buat file baru

   Buka file IDLE baru, lalu ketik beberapa acara mendatang yang penting bagi Anda. Letakkan setiap acara pada baris terpisah dan ketik koma di antara acara dan tanggalnya. Pastikan tidak ada spasi antara koma dan tanggal acara.



2. Simpan sebagai file teks

   Selanjutnya simpan file tersebut sebagai file teks. Klik menu File, pilih Save As, dan panggil file “events.txt”. Sekarang Anda siap untuk mulai mengkode program Python.



3. Buka file Python baru

   Anda sekarang perlu membuat file baru untuk kode tersebut. Simpan sebagai "countdown_calendar.py" dan pastikan itu berada di folder yang sama dengan file "events.txt" Anda.

Gambar 4.4 Ilustrasi Membuka file

4. Siapkan modul

   Proyek ini membutuhkan dua modul: Tkinter dan datetime. Tkinter akan digunakan untuk membangun GUI sederhana, sedangkan datetime akan memudahkan perhitungan menggunakan tanggal. Impor mereka dengan mengetikkan dua baris ini di bagian atas program baru Anda.

```
from tkinter import Tk, Canvas
from datetime import date, datetime
```

Import the **Tkinter** and **datetime** modules.

5. Buat kanvas

   Sekarang atur jendela yang akan menampilkan acara penting Anda dan jumlah hari hingga setiap acara. Letakkan kode ini di bawah garis yang Anda tambahkan pada Langkah 4. Ini menciptakan jendela yang berisi "kanvas"—persegi panjang kosong yang dapat Anda tambahkan teks dan gambar.

This command packs the canvas into the **Tkinter** window.

Create a **Tkinter** window.

Create a canvas called **c** that is 800 pixels wide by 800 pixels high.

```
root = Tk()
c = Canvas(root, width=800, height=800, bg='black')
c.pack()
c.create_text(100, 50, anchor='w', fill='orange', \
font='Arial 28 bold underline', text='My Countdown Calendar')
```

This line adds text onto the **c** canvas. The text starts at x = 100, y = 50. The starting coordinate is at the left (west) of the text.

Kanvas

Di Tkinter, kanvas adalah area, biasanya persegi panjang, tempat Anda dapat menempatkan berbagai bentuk, grafik, teks, atau gambar yang dapat dilihat atau berinteraksi dengan pengguna. Anggap saja seperti kanvas seniman—kecuali Anda menggunakan kode untuk membuat sesuatu, bukan kuas!

6. Jalankan kodenya

   Sekarang coba jalankan kodenya. Anda akan melihat jendela muncul dengan judul program. Jika tidak berhasil, ingatlah untuk membaca pesan kesalahan apa pun dan periksa kode Anda dengan cermat untuk menemukan kemungkinan kesalahan.

Gambar 4.5 jendel hasil saat *run* program

7. Baca file teks

   Selanjutnya buat fungsi yang akan membaca dan menyimpan semua kejadian dari file teks. Di bagian atas kode Anda, setelah mengimpor modul, buat fungsi baru bernama get_events. Di dalam fungsi tersebut terdapat daftar kosong yang akan menyimpan kejadian ketika file telah dibaca.

```
from datetime import date, datetime
def get_events():
    list_events = []
root = Tk()
```

Create an empty list called **list_events**.

8. Buka file teks

   Potongan kode berikutnya akan membuka file bernama events.txt sehingga program dapat membacanya. Ketik baris ini di bawah kode Anda dari Langkah 7.

```
def get_events():
    list_events = []
    with open('events.txt') as file:
```

This line opens the text file.

9. Mulai lingkaran

   Sekarang tambahkan for loop untuk membawa informasi dari file teks ke dalam program Anda. Loop akan dijalankan untuk setiap baris dalam file events.txt.

```
def get_events():
    list_events = []
    with open('events.txt') as file:
        for line in file:
```

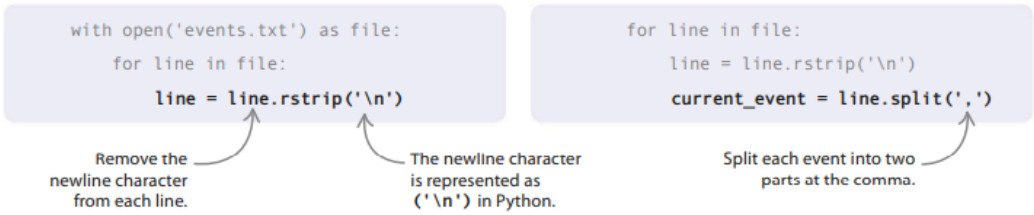
Run the loop for each line in the text file.

10. Hapus karakter yang tidak terlihat

    Saat Anda mengetik informasi ke dalam file teks di Langkah 1, Anda menekan tombol enter/return di akhir setiap baris. Ini menambahkan karakter "baris baru" yang tidak terlihat di akhir setiap baris. Meskipun Anda tidak dapat melihat karakter ini, Python bisa. Tambahkan baris kode ini, yang memberi tahu Python untuk mengabaikan karakter tak terlihat ini saat membaca file teks.

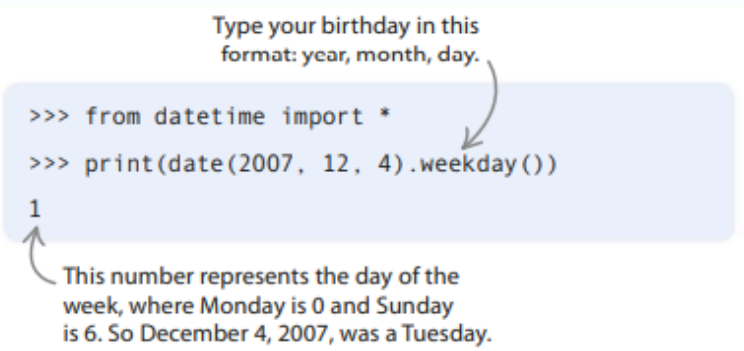
11. Simpan detail acara

    Pada titik ini, variabel yang disebut garis menyimpan informasi tentang setiap peristiwa sebagai string, seperti Halloween,31/10/2017. Gunakan perintah split() untuk memotong string ini menjadi dua bagian. Bagian sebelum dan sesudah koma akan menjadi item terpisah yang dapat Anda simpan dalam daftar yang disebut current_event. Tambahkan baris ini setelah kode Anda di Langkah 10.



Modul waktu-tanggal

Modul *datetime* Python sangat berguna jika Anda ingin melakukan perhitungan yang melibatkan tanggal dan waktu. Misalnya, apakah Anda tahu hari apa dalam seminggu Anda lahir? Coba ketik ini ke dalam shell Python untuk mencari tahu.

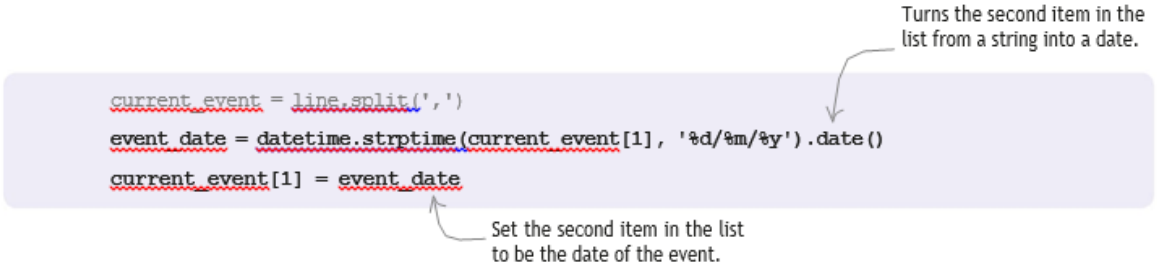


Daftar posisi

Ketika Python memberi nomor item dalam daftar, itu dimulai dari 0. Jadi item pertama dalam daftar current_event Anda, “Halloween”, ada di posisi 0, sedangkan item kedua, “31/10/2017”, ada di posisi 1. Itu sebabnya kode mengubah current_event[1] menjadi tanggal.

12. Menggunakan waktu tanggal

    Acara Halloween disimpan di current_event sebagai daftar yang berisi dua item: "Halloween" dan "31/10/2017". Gunakan modul datetime untuk mengonversi item kedua dalam daftar (di posisi 1) dari string menjadi bentuk yang dapat dipahami Python sebagai tanggal. Tambahkan baris kode ini di bagian bawah fungsi.



13. Tambahkan acara ke daftar

    Sekarang daftar current_event menampung dua hal: nama acara (sebagai string) dan tanggal acara. Tambahkan current_event ke daftar acara. Berikut seluruh kode untuk fungsi get_events().

```
def get_events():
    list_events = []
    with open('events.txt') as file:
        for line in file:
            line = line.rstrip('\n')
            current_event = line.split(',')
            event_date = datetime.strptime(current_event[1], '%d/%m/%y').date()
            current_event[1] = event_date
            list_events.append(current_event)
    return list_events
```

After this line is run, the program loops back to read the next line from the file.

After all the lines have been read, the function hands over the complete list of events to the program.

Mengatur hitungan mundur

Pada tahap pembuatan Kalender Hitung Mundur berikutnya, Anda akan membuat fungsi untuk menghitung jumlah hari antara hari ini dan acara penting Anda. Anda juga akan menulis kode untuk menampilkan acara di kanvas Tkinter.

14. Hitung hari

    Buat fungsi untuk menghitung jumlah hari antara dua tanggal. Modul datetime membuatnya mudah, karena dapat menambahkan tanggal bersama-sama atau mengurangi satu dari yang lain. Ketik kode yang ditampilkan di sini di bawah fungsi get_events() Anda. Ini akan menyimpan jumlah hari sebagai string dalam variabel time_between.

```
def days_between_dates(date1, date2):
    time_between = str(date1-date2)
```

This variable stores the result as a string.

The dates are subtracted to give the number of days between them.

15. Pisahkan talinya

    Jika Halloween tinggal 27 hari lagi, string yang disimpan di time_between akan terlihat seperti ini: '27 hari, 0:00:00' (nol mengacu pada jam, menit, dan detik). Hanya angka di awal string yang penting, jadi Anda bisa menggunakan perintah split() lagi untuk mendapatkan bagian yang Anda butuhkan. Ketik kode yang disorot di bawah ini setelah kode pada Langkah 14. Ini mengubah string menjadi daftar tiga item: '27', 'hari', '0:00:00'. Daftar disimpan dalam number_of_days.

```
def days_between_dates(date1, date2):
    time_between = str(date1-date2)
    number_of_days = time_between.split(' ')
```

This time the string is split at each blank space.

16. Kembalikan jumlah hari

    Untuk menyelesaikan fungsi ini, Anda hanya perlu mengembalikan nilai yang disimpan di posisi 0 dari daftar. Dalam kasus Halloween, itu adalah 27. Tambahkan baris kode ini ke akhir fungsi.

```
def days_between_dates(date1, date2):
    time_between = str(date1-date2)
    number_of_days = time_between.split(' ')
    return number_of_days[0]
```

17. Dapatkan acaranya

    Sekarang setelah Anda menulis semua fungsi, Anda dapat menggunakannya untuk menulis bagian utama program. Letakkan dua baris ini di bagian bawah file Anda. Baris pertama memanggil (menjalankan) fungsi get_events() dan menyimpan daftar acara kalender dalam variabel yang disebut acara. Baris kedua menggunakan modul datetime untuk mendapatkan tanggal hari ini dan menyimpannya dalam variabel yang disebut hari ini.

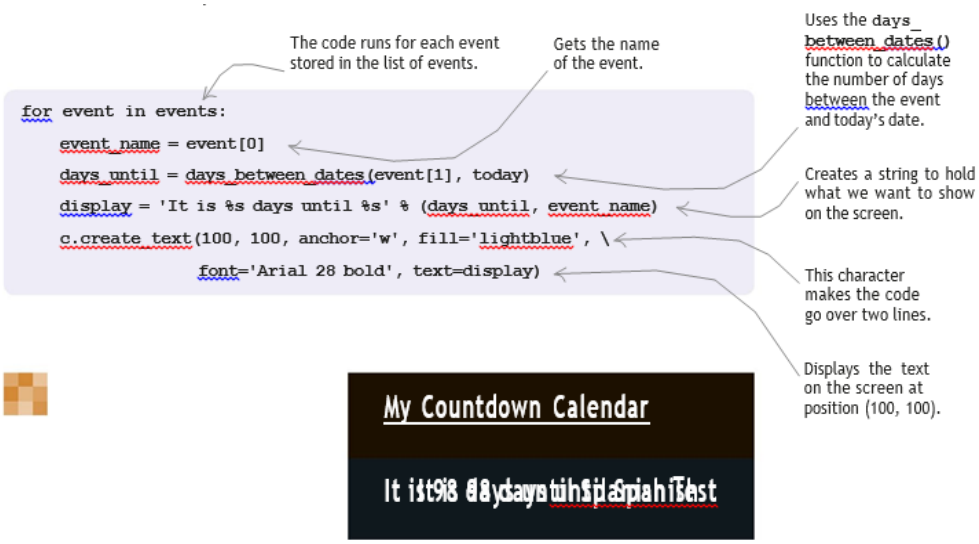
```
c.create_text(100, 50, anchor='w', fill='orange', \
font='Arial 28 bold underline', text='My Countdown Calendar')

events = get_events()
today = date.today()
```

18. Tampilkan hasilnya

    Selanjutnya hitung jumlah hari hingga setiap peristiwa dan tampilkan hasilnya di layar. Anda perlu melakukan ini untuk setiap acara dalam daftar, jadi gunakan for loop. Untuk setiap peristiwa dalam daftar, panggil fungsi days_between_dates() dan simpan hasilnya dalam variabel yang disebut days_until. Kemudian gunakan fungsi Tkinter create_text() untuk menampilkan hasilnya di layar. Tambahkan kode ini tepat setelah kode dari Langkah 17.

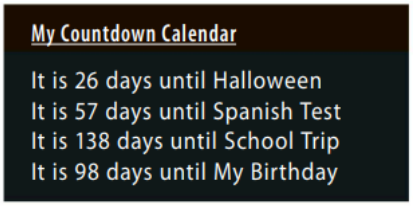
19. Uji programnya

    Sekarang coba jalankan kodenya. Sepertinya semua baris teks ditulis di atas satu sama lain. Bisakah Anda mencari tahu apa yang salah? Bagaimana Anda bisa menyelesaikannya?



20. Sebarkan

    Masalahnya adalah semua teks ditampilkan di lokasi yang sama (100, 100). Jika kita membuat variabel bernama vertical_space dan meningkatkan nilainya setiap kali program melewati perulangan for, itu akan meningkatkan nilai koordinat y dan spasi teks lebih jauh ke bawah layar. Itu akan menyelesaikannya!



Gambar 4.7 PopUp Kalender yang muncul

21. Mulai hitungan mundur!

    Itu saja—Anda telah menulis semua kode yang Anda butuhkan untuk Kalender Hitung Mundur. Sekarang jalankan program Anda dan cobalah.

```
vertical_space = 100

for event in events:
    event_name = event[0]
    days_until = days_between_dates(event[1], today)
    display = 'It is %s days until %s' % (days_until, event_name)
    c.create_text(100, vertical_space, anchor='w', fill='lightblue', \
                  font='Arial 28 bold', text=display)

    vertical_space = vertical_space + 30
```

Peretasan dan penyesuaian

Cobalah peretasan dan penyesuaian ini untuk membuat Kalender Hitung Mundur lebih berguna. Beberapa di antaranya lebih sulit daripada yang lain, jadi ada beberapa tips berguna untuk membantu Anda.

Cat ulang kanvas

Anda dapat mengedit warna latar belakang kanvas Anda dan benar-benar meramaikan tampilan tampilan program. Ubah baris kode c = Canvas.

```
c = Canvas(root, width=800, height=800, bg='green')
```

You can change the background color to any color of your choice.

Urutkan!

Anda dapat mengubah kode Anda sehingga acara diurutkan ke dalam urutan yang akan terjadi. Tambahkan baris kode ini sebelum perulangan for. Ini menggunakan fungsi sort() untuk mengatur acara dalam urutan menaik, dari jumlah hari terkecil hingga yang terbesar.

```
vertical_space = 100
events.sort(key=lambda x: x[1])
for event in events:
```

Sort the list in order of days to go and not by the name of the events.

Ubah gaya teks

Berikan antarmuka pengguna Anda tampilan baru yang segar dengan mengubah ukuran, warna, dan gaya teks judul.

```
c.create_text(100, 50, anchor='w', fill='pink', font='Courier 36 bold underline', \
              text='Sanjay\'s Diary Dates')
```

Setel pengingat

Mungkin berguna untuk menyoroti peristiwa yang akan segera terjadi. Retas kode Anda sehingga acara apa pun yang terjadi di minggu depan ditampilkan dengan warna merah.

```
for event in events:
    event_name = event[0]
    days_until = days_between_dates(event[1], today)
    display = 'It is %s days until %s' % (days_until, event_name)
    if (int(days_until) <= 7):
        text_col = 'red'
    else:
        text_col = 'lightblue'
    c.create_text(100, vertical_space, anchor='w', fill=text_col, \
                  font='Arial 28 bold', text=display)
```

The symbol <= means "is less than or equal to".

Display the text using the correct color.

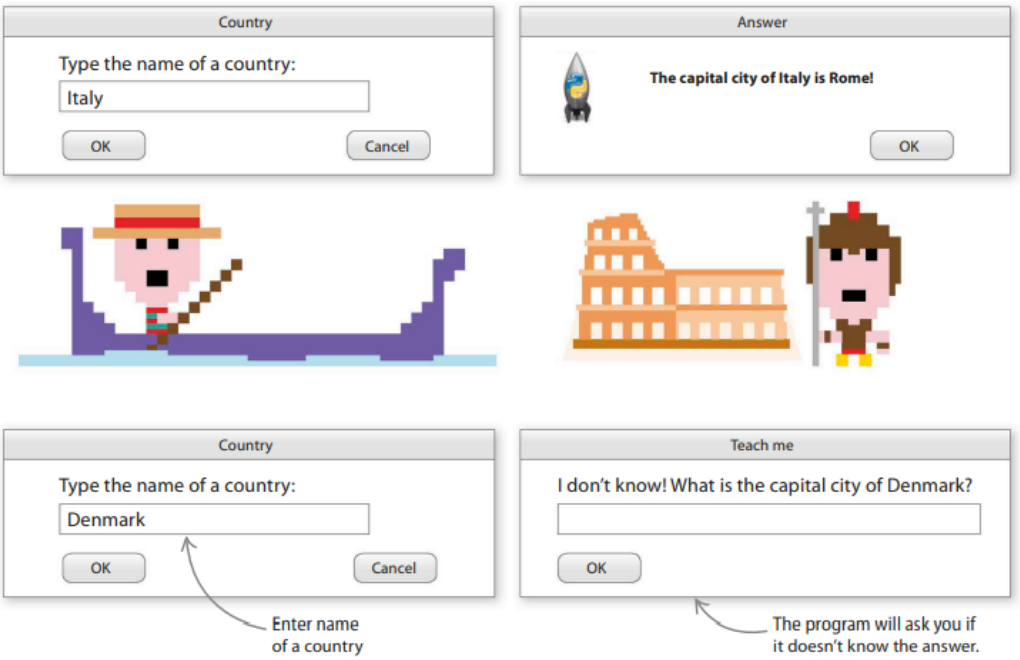
The `int()` function changes a string into a number. For example, it turns the string '5' into the number 5.

4.3 TANYAKAN PADA AHLINYA

Bisakah Anda menyebutkan semua ibu kota di dunia? Atau para pemain di tim olahraga favorit Anda? Setiap orang ahli dalam sesuatu. Dalam proyek ini, Anda akan membuat kode program yang tidak hanya dapat menjawab pertanyaan Anda, tetapi juga mempelajari hal-hal baru dan menjadi seorang ahli.

Apa yang terjadi

Kotak input meminta Anda memasukkan nama negara. Saat Anda mengetikkan jawaban Anda, program akan memberi tahu Anda apa ibu kotanya. Jika program tidak tahu, ia meminta Anda untuk mengajarinya jawaban yang benar. Semakin banyak orang menggunakan program ini, semakin pintar!



Gambar 4.8 Pop-Up Pertanyaan ketika salah menjawab

Bagaimana itu bekerja

Program mendapatkan informasi tentang ibu kota dari file teks. Anda akan menggunakan modul Tkinter untuk membuat kotak popup yang memungkinkan program dan pengguna berkomunikasi. Ketika ibu kota baru dimasukkan oleh pengguna, informasi ditambahkan ke dalam file teks.

Komunikasi

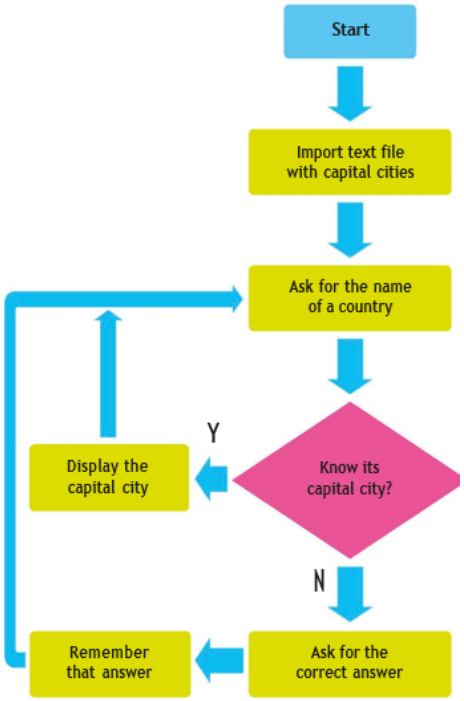
Program ini menggunakan dua widget Tkinter baru. Yang pertama, simpledialog(), membuat kotak popup yang meminta pengguna untuk memasukkan nama negara. Yang kedua, messagebox(), menampilkan ibu kota.

Kamus

Anda akan menyimpan nama negara dan ibu kotanya dalam kamus. Kamus bekerja sedikit seperti daftar, tetapi setiap item dalam kamus memiliki dua bagian, yang disebut kunci dan nilai. Biasanya lebih cepat untuk mencari sesuatu di kamus daripada menemukan sesuatu dalam daftar panjang.

Tanyakan diagram alur Pakar

Ketika program dimulai, ia membaca data dari file teks. Kemudian menggunakan loop tak terbatas untuk terus mengajukan pertanyaan, dan hanya berhenti ketika pengguna keluar dari program.



Gambar 4.9 Diagram alur membuat pertanyaan

4.4 SISTEM PAKAR

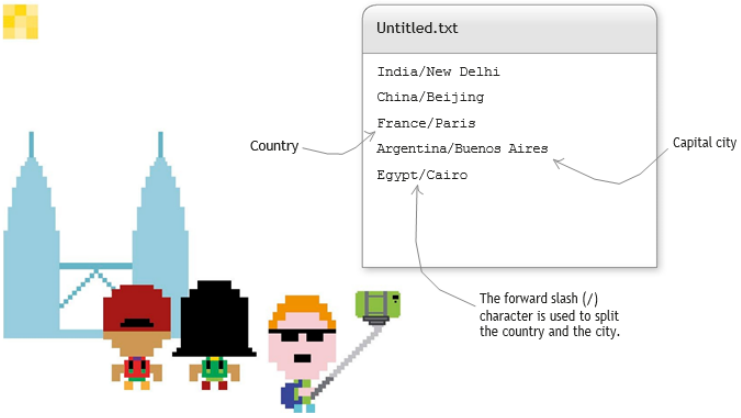
Sistem pakar adalah program komputer yang ahli pada topik tertentu. Sama seperti seorang ahli manusia, ia tahu jawaban atas banyak pertanyaan, dan juga dapat membuat keputusan dan memberi nasihat. Hal ini dapat dilakukan karena seorang programmer telah mengkodekannya dengan semua data yang dibutuhkannya dan aturan tentang cara menggunakan data tersebut.

Langkah pertama

Ikuti langkah-langkah ini untuk membangun sistem pakar Anda sendiri menggunakan Python. Anda harus menulis file teks ibukota negara, membuka jendela Tkinter, dan membuat kamus untuk menyimpan semua pengetahuan.

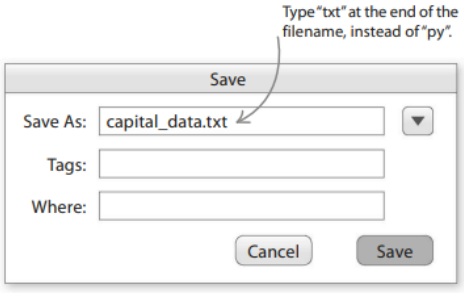
1. Siapkan file teks

   Pertama buat file teks untuk menampung daftar ibu kota dunia. Buat file baru di IDLE dan ketik fakta berikut.



Gambar 4.10 membuat file

2. Simpan file teks
   Simpan file sebagai "capital_data.txt". Program akan mendapatkan pengetahuan khusus dari file ini.
3. Buat file Python
   Untuk menulis program, buat file baru dan simpan sebagai "ask_expert.py". Pastikan Anda menyimpannya di folder yang sama dengan file teks Anda.



Gambar 4.11 Menyimpan File

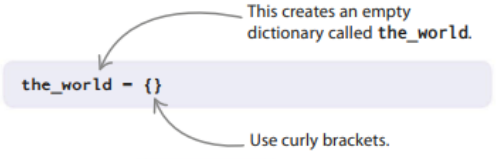
4. Impor alat Tkinter
   Untuk membuat program ini, Anda memerlukan beberapa widget dari modul Tkinter. Ketik baris ini di bagian atas program Anda.

```
from tkinter import Tk, simpledialog, messagebox
```

5. Mulai Tkinter
   Selanjutnya tambahkan kode berikut untuk menampilkan judul proyek di shell. Tkinter secara otomatis membuat jendela kosong. Anda tidak memerlukannya untuk proyek ini, jadi sembunyikan dengan baris kode yang cerdas.

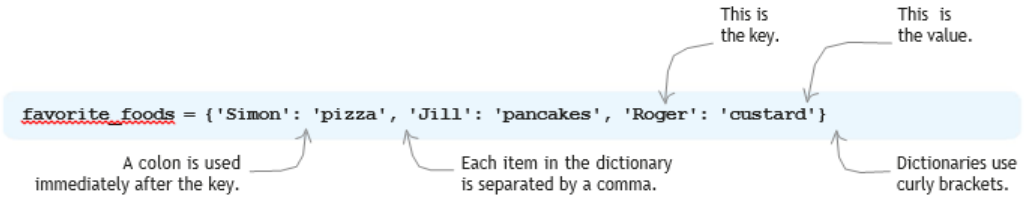
```
print('Ask the Expert - Capital Cities of the World')
root = Tk()
root.withdraw()
```

6. Uji kodenya
   Coba jalankan kode Anda. Anda akan melihat nama proyek yang ditampilkan di shell.
7. Siapkan kamus
   Sekarang ketik baris kode ini setelah kode yang Anda tulis untuk Langkah 5. Kode baru mengatur kamus yang akan menyimpan nama negara dan ibu kotanya.

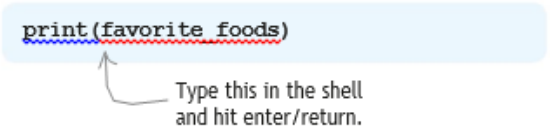


Menggunakan kamus

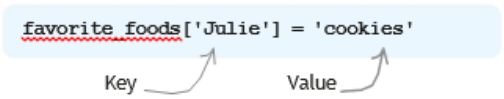
Kamus adalah cara lain Anda dapat menyimpan informasi dengan Python. Ini mirip dengan daftar, tetapi setiap item memiliki dua bagian: kunci dan nilai. Anda dapat mengujinya dengan mengetik ini di jendela shell.



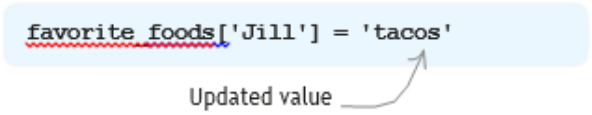
1. Untuk menampilkan isi kamus, Anda harus mencetaknya. Coba cetak favorite_foods.



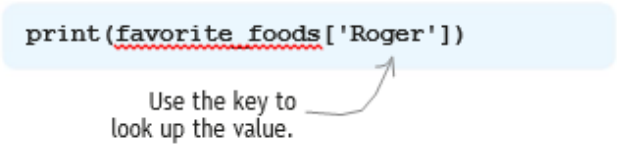
2. Sekarang tambahkan item baru ke kamus: Julie dan makanan favoritnya. Dia suka kue.



3. Jill berubah pikiran—makanan favoritnya sekarang adalah taco. Anda dapat memperbarui informasi ini di kamus.



4. Terakhir, kamu bisa mencari makanan favorit Roger di kamus hanya dengan menggunakan namanya sebagai kunci.

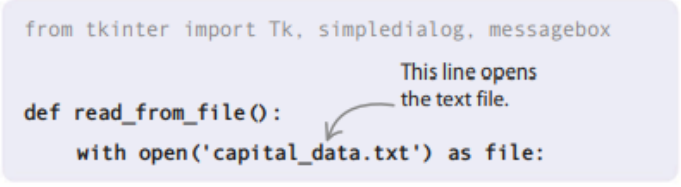


Saatnya fungsi!

Tahap proyek selanjutnya melibatkan pembuatan fungsi yang perlu Anda gunakan dalam program Anda.

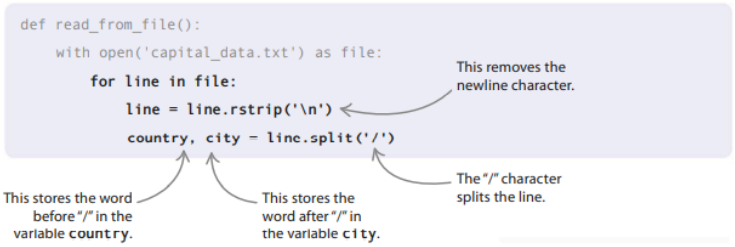
8. Masukan file

Anda memerlukan fungsi untuk membaca semua informasi yang tersimpan dalam file teks Anda. Ini akan mirip dengan yang Anda gunakan di Kalender Hitung Mundur untuk membaca data dari file acara Anda. Tambahkan kode ini setelah baris impor Tkinter.



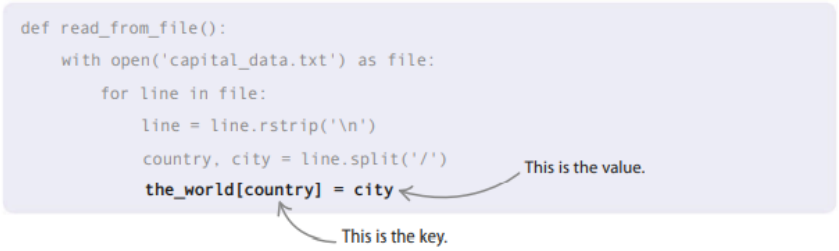
9. Baris demi baris

   Sekarang gunakan for loop untuk menelusuri file baris demi baris. Sama seperti di Kalender Hitung Mundur, Anda harus menghapus karakter baris baru yang tidak terlihat. Maka Anda perlu menyimpan nilai negara dan kota dalam dua variabel. Menggunakan perintah split, kode akan mengembalikan dua nilai. Anda dapat menyimpan nilai-nilai ini dalam dua variabel menggunakan satu baris kode.



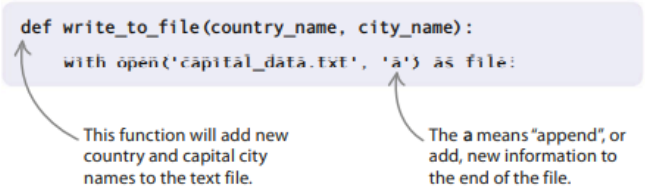
10. Tambahkan data ke kamus

    Pada tahap ini, variabel negara dan kota menyimpan informasi yang perlu Anda tambahkan ke dalam kamus. Untuk baris pertama dalam file teks Anda, negara akan memegang "India" dan kota akan memegang "New Delhi". Baris kode berikutnya menambahkannya ke dalam kamus.



11. Keluaran berkas

    Saat pengguna mengetik di ibu kota yang tidak diketahui program, Anda ingin program memasukkan informasi baru ini ke dalam file teks. Ini disebut keluaran file. Ini bekerja dengan cara yang mirip dengan input file, tetapi alih-alih membaca file, Anda menulis ke dalamnya. Ketik fungsi baru ini setelah kode yang Anda ketik di Langkah 10.



12. Tulis ke file

    Sekarang tambahkan satu baris kode untuk menulis informasi baru ke dalam file. Pertama kode akan menambahkan karakter baris baru, yang memberitahu program untuk memulai baris baru dalam file teks. Kemudian ditulis nama negara diikuti dengan garis miring (/) dan nama ibu kota, seperti Mesir/Kairo. Python secara otomatis menutup file teks setelah informasi ditulis ke dalamnya.

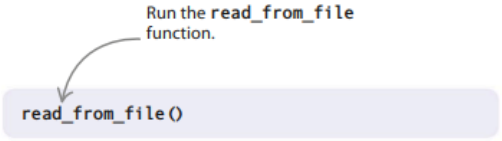
```
def write_to_file(country_name, city_name):
    with open('capital_data.txt', 'a') as file:
        file.write('\n' + country_name + '/' + city_name)
```

Kode program utama

Anda telah menulis semua fungsi yang Anda butuhkan, jadi inilah saatnya untuk mulai mengkode program utama.

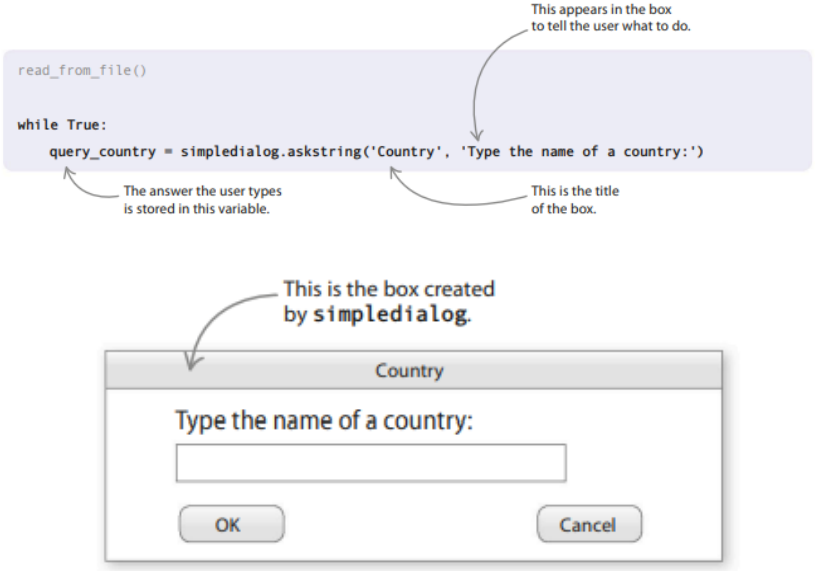
13. Baca file teks

    Hal pertama yang Anda ingin program lakukan adalah membaca informasi dari file teks. Tambahkan baris ini setelah kode yang Anda tulis di Langkah 7.



14. Mulai loop tak terbatas

    Selanjutnya tambahkan kode di bawah ini untuk membuat infinite loop. Di dalam loop adalah fungsi dari modul Tkinter: simpledialog.askstring(). Fungsi ini membuat kotak di layar yang menampilkan informasi dan memberi ruang bagi pengguna untuk mengetik jawaban. Uji kode lagi. Sebuah kotak akan muncul menanyakan Anda untuk nama negara. Mungkin tersembunyi di balik jendela lain.



Gambar 4.12 Jendela pertanyaan

15. Tahu jawabannya?

    Sekarang tambahkan pernyataan if untuk melihat apakah program mengetahui jawabannya. Ini akan memeriksa apakah negara dan ibu kotanya disimpan dalam kamus.

```
while True:
    query_country = simpledialog.askstring('Country', 'Type the name of a country:')

    if query_country in the_world:
```

Will return True if the country input by the user is stored in **the_world**.

16. Tampilkan jawaban yang benar

    Jika negara berada di the_world, Anda ingin program mencari jawaban yang benar dan menampilkannya di layar. Untuk melakukan ini, gunakan kotak pesan. showinfo() dari modul Tkinter. Ini menampilkan pesan dalam kotak dengan tombol OK. Ketik ini di dalam pernyataan if.

```
    if query_country in the_world:
        result = the_world[query_country]
        messagebox.showinfo('Answer',
                            'The capital city of ' + query_country + ' is ' + result + '!')
```

This is the title of the box.

This variable stores the answer (the value from the dictionary).

This message will be displayed inside the box.

17. Ujilah

    Jika kode Anda memiliki bug, sekarang adalah saat yang tepat untuk menangkapnya. Saat diminta untuk menyebutkan nama negara, ketik "Prancis". Apakah itu memberi Anda jawaban yang benar? Jika tidak, lihat kembali kode Anda dengan hati-hati dan lihat apakah Anda dapat menemukan di mana kesalahannya. Apa yang akan terjadi jika Anda mengetik di negara yang tidak ada dalam file teks? Cobalah untuk melihat bagaimana program merespons.

18. Ajarkan itu

    Terakhir, tambahkan beberapa baris lagi setelah pernyataan if. Jika negara tidak ada dalam kamus, program akan meminta pengguna untuk memasukkan nama ibu kotanya. Ibu kota ini ditambahkan ke kamus, sehingga program mengingatnya untuk waktu berikutnya. Kemudian fungsi write_to_file() menambahkan kota ke file teks.

```
    if query_country in the_world:
        result = the_world[query_country]
        messagebox.showinfo('Answer',
                            'The capital city of ' + query_country + ' is ' + result + '!')
    else:
        new_city = simpledialog.askstring('Teach me',
                                          'I don\'t know!' +
                                          'What is the capital city of' + query_country + '?')
        the_world[query_country] = new_city
        write_to_file(query_country, new_city)

root.mainloop()
```

Ask the user to type in the capital city and store it in **new_city**.

This adds **new_city** to the dictionary, using **query_country** as the key.

19. Menjalankannya

    Itu dia. Anda telah menciptakan pakar digital! Sekarang jalankan kodenya dan mulai kuis!

Peretasan dan penyesuaian

Bawa program Anda ke tingkat berikutnya dan buat lebih pintar dengan mencoba saran-saran ini.

Di seluruh dunia

Ubah program Anda menjadi jenius geografis dengan membuat file teks yang berisi setiap negara di dunia dan ibu kotanya. Ingatlah untuk meletakkan setiap entri pada baris baru dalam format ini: nama negara/ibu kota.

Kapitalisasi

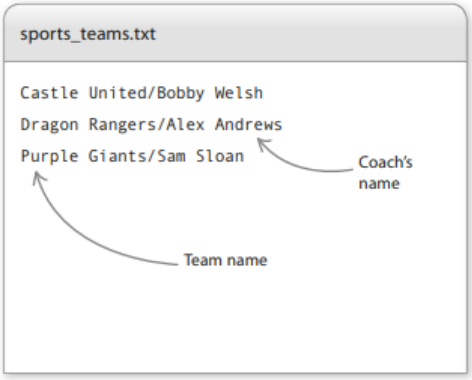
Jika pengguna lupa menggunakan huruf kapital untuk menyebut negara, program tidak akan menemukan ibu kota. Bagaimana Anda bisa memecahkan masalah ini menggunakan kode? Inilah salah satu cara untuk melakukannya.

```
query_country = simpledialog.askstring('Country', 'Type the name of a country:')
query_country = query_country.capitalize()
```

This function turns the first letter in a string into a capital letter.

Data yang berbeda

Saat ini, program hanya mengetahui tentang ibu kota dunia. Anda dapat mengubahnya dengan mengedit file teks sehingga menyimpan fakta tentang subjek yang menjadi keahlian Anda. Misalnya, Anda bisa mengajarinya nama tim olahraga terkenal dan pelatihnya.



Gambar 4.13 Jendela Penulisan data

Cek fakta

Program Anda saat ini menambahkan jawaban baru langsung ke file teks, tetapi tidak dapat memeriksa apakah jawabannya benar. Tweak kode sehingga jawaban baru disimpan dalam file teks terpisah. Kemudian Anda dapat memeriksanya nanti sebelum menambahkannya ke file teks utama. Inilah cara Anda dapat mengubah kode.

```
def write_to_file(country_name, city_name):
    with open('new_data.txt', 'a') as file:
        file.write('\n' + country_name + '/' + city_name)
```

4.5 PESAN RAHASIA

Tukar pesan dengan teman Anda menggunakan seni kriptografi—mengubah teks pesan sehingga orang yang tidak mengetahui metode rahasia Anda tidak dapat memahaminya!

Apa yang terjadi

Program akan menanyakan apakah Anda ingin membuat pesan rahasia atau mengungkapkan isi pesan rahasia. Ini kemudian akan meminta Anda untuk mengetikkan pesan. Jika Anda memilih untuk membuat pesan rahasia, pesan Anda akan berubah menjadi omong kosong belaka. Tetapi jika Anda memilih untuk mengungkapkan pesan, omong kosong akan berubah menjadi teks yang dapat Anda baca!

Kriptografi

Kata kriptografi berasal dari kata Yunani kuno untuk “tersembunyi” dan “menulis.” Orang-orang telah menggunakan teknik ini untuk mengirim pesan rahasia selama hampir 4.000 tahun. Berikut adalah beberapa istilah khusus yang digunakan dalam kriptografi—

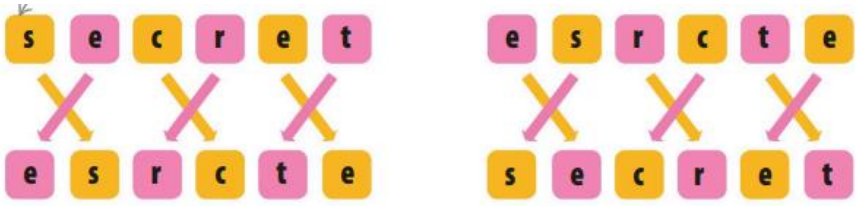
- **Cipher:** satu set instruksi untuk mengubah pesan untuk menyembunyikan maknanya.
- **Encrypt:** untuk menyembunyikan pesan rahasia.
- **Decrypt:** untuk mengungkapkan pesan rahasia.
- **Ciphertext:** pesan setelah dienkripsi.
- **Plaintext:** pesan sebelum dienkripsi.

Bagikan kodenya

Jika Anda membagikan kode Python Anda dengan seorang teman, Anda akan dapat saling mengirimkan pesan rahasia.

Bagaimana itu bekerja

Program mengatur ulang urutan huruf dalam pesan sehingga tidak dapat dipahami. Ini dilakukan dengan mencari tahu huruf mana yang berada di posisi genap atau ganjil. Kemudian ia menukar posisi setiap pasangan huruf dalam pesan, dimulai dengan dua yang pertama, lalu dua berikutnya, dan seterusnya. Program ini juga membuat pesan terenkripsi dapat dibaca kembali dengan mengalihkan huruf kembali ke tempat awal.



Gambar 4.14 Pesan Rahasia dalam sebuah Text

Enkripsi

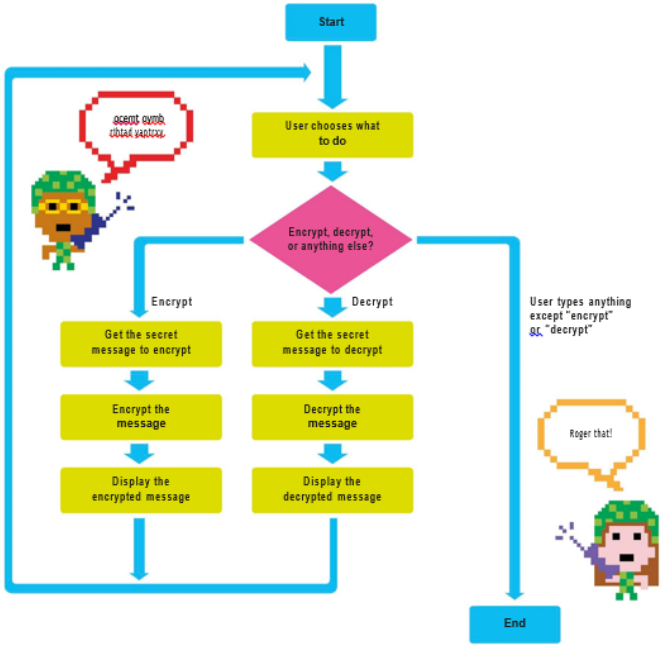
Saat Anda menjalankan kode pada pesan Anda, program menukar setiap pasangan huruf, mengacak artinya.

Dekripsi

Saat Anda atau teman mendekripsi pesan, program akan mengembalikan huruf ke posisi semula.

Bagan alir Pesan Rahasia

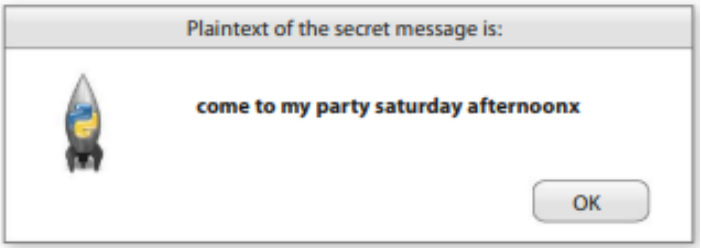
Program ini menggunakan infinite loop yang menanyakan pengguna apakah mereka ingin mengenkripsi atau mendekripsi. Pilihan pengguna menentukan jalur mana yang kemudian diambil program. Kotak dialog mendapatkan teks dari pengguna, sedangkan kotak info menampilkan pesan terenkripsi dan dekripsi kepada mereka. Program berakhir jika pengguna mengetik apa pun kecuali "enkripsi" atau "dekripsi".



Gambar 4.16 Diagram alir pesan rahasia

Misteri x

Program membutuhkan pesan untuk memiliki jumlah karakter yang genap. Ini memeriksa pesan dan menghitung karakter. Jika ada jumlah karakter yang ganjil, ia menambahkan x di akhir untuk membuatnya genap. Anda dan rekan agen rahasia Anda akan tahu untuk mengabaikan x, jadi Anda tidak akan tertipu!



Gambar 4.17 Hasil Pesan rahasia yang muncul

4.6 MEMBUAT GUI

Anda akan menulis kode Anda dalam dua bagian. Pertama, Anda akan mengatur beberapa fungsi untuk mendapatkan masukan dari pengguna; maka Anda akan menulis kode yang melakukan enkripsi dan dekripsi. Sekarang mari kita mulai—Anda tidak pernah tahu kapan Anda mungkin perlu mengirim pesan rahasia kepada seseorang!

1. Buat file baru
   Buka IDLE dan buat file baru. Simpan sebagai "secret_messages.py".

2. Tambahkan modul
   Anda perlu mengimpor beberapa widget dari modul Tkinter Python. Ini akan memungkinkan Anda menggunakan beberapa fitur GUI-nya, seperti kotak pesan untuk menampilkan informasi kepada pengguna, dan dialog sederhana untuk mengajukan pertanyaan kepada mereka. Ketik baris ini di bagian atas file Anda.

```
from tkinter import messagebox, simpledialog, Tk
```

3. Enkripsi atau dekripsi?
   Sekarang buat fungsi, get_task(), untuk membuka kotak dialog yang menanyakan pengguna apakah mereka ingin mengenkripsi atau mendekripsi pesan. Tambahkan fungsi di bawah kode yang Anda tambahkan di Langkah 2.

```
def get_task():
    task = simpledialog.askstring('Task', 'Do you want to encrypt or decrypt?')
    return task
```

Pass the value in **task** back to the code that used this function.

This word will appear as a title in the dialogue box.

4. Dapatkan pesannya
   Buat fungsi baru, get_message(), untuk membuka kotak dialog yang meminta pengguna mengetikkan pesan yang ingin mereka enkripsi atau dekripsi. Tambahkan fungsi ini di bawah kode yang Anda tambahkan di Langkah 3.

```
def get_message():
    message = simpledialog.askstring('Message', 'Enter the secret message: ')
    return message
```

Pass the value in **message** back to the code that used this function.

5. Mulai Tkinter
   Perintah ini memulai Tkinter dan membuat jendela Tkinter. Ketik di bawah fungsi yang Anda buat di Langkah 4.

```
root = Tk()
```

If you find the **Tkinter** window distracting, add the **root.withdraw** line you used in Ask the Expert.

6. Mulai putaran
   Sekarang setelah Anda membuat fungsi antarmuka, tambahkan loop while tak terbatas ini untuk memanggil (menjalankan) fungsi tersebut dalam urutan yang benar. Masukkan kode ini di bawah perintah yang Anda ketik di Langkah 5.

```
while True:
    task = get_task()
    if task == 'encrypt':
        message = get_message()
        messagebox.showinfo('Message to encrypt is:', message)
    elif task == 'decrypt':
        message = get_message()
        messagebox.showinfo('Message to decrypt is:', message)
    else:
        break
root.mainloop()
```

Find out what the user wants to do.

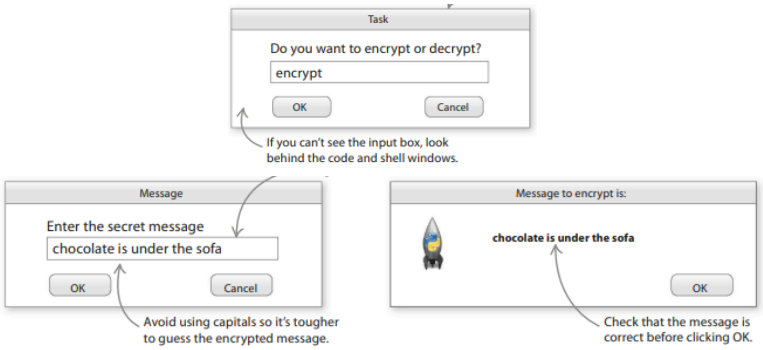
Get the secret message for encryption.

Gets the secret message for decryption

Stop looping if the user doesn't type "encrypt" or "decrypt".

7. Uji kodenya

   Coba jalankan kodenya. Ini pertama-tama akan menampilkan kotak input yang menanyakan apakah Anda ingin mengenkripsi atau mendekripsi. Kemudian akan muncul kotak input lain sehingga Anda dapat mengetikkan pesan rahasia tersebut. Terakhir, itu akan menampilkan pesan terenkripsi atau didekripsi di kotak info. Jika ada masalah, periksa kode Anda dengan cermat.



Gambar 4.18 Hasil pengujian

Catat pesannya!

Sekarang setelah antarmuka Anda berfungsi, saatnya untuk menulis kode yang akan mengenkripsi dan kemudian mendekripsi pesan rahasia Anda.

8. Apakah itu genap?

   Anda perlu membuat fungsi untuk memberi tahu program apakah jumlah karakter dalam pesan Anda genap atau tidak. Fungsi akan menggunakan operator modulo (%) untuk memeriksa apakah dapat membagi bilangan dengan 2 tanpa meninggalkan sisa. Jika bisa (Benar), maka angkanya genap. Tambahkan fungsi ini di bawah kode yang Anda ketik di Langkah 2.

```
def is_even(number):
    return number % 2 == 0
```

This will be True if the number is even.

Pass the True or False value back to the code.

9. Dapatkan huruf genap

   Pada langkah ini, Anda akan membuat fungsi yang mengambil pesan dan menghasilkan daftar yang berisi semua huruf genap. Fungsi ini menggunakan perulangan for dengan

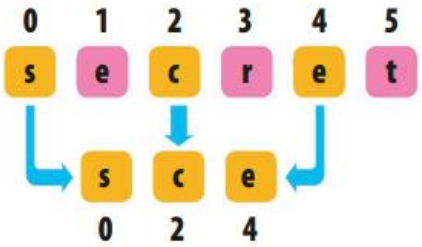
rentang dari 0 hingga len(pesan), sehingga ia memeriksa semua huruf dalam string. Tambahkan fungsi ini di bawah kode di Langkah 8.

```
def get_even_letters(message):
    even_letters = []
    for counter in range(0, len(message)):
        if is_even(counter):
            even_letters.append(message[counter])
    return even_letters
```

Loop through every letter in the message.

If this is a letter in an even position, Python adds it to the end of the list of letters.

Pass the list of letters back to the code that called this function.

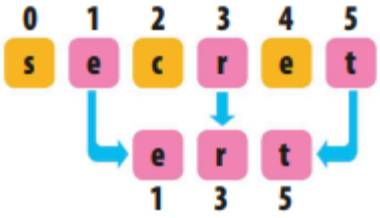


Gambar 4.19 Pengambilan posisi genap

10. Dapatkan huruf ganjil

    Selanjutnya Anda perlu membuat fungsi serupa untuk menghasilkan daftar semua huruf bernomor ganjil dalam pesan Anda. Letakkan fungsi ini di bawah kode pada Langkah 9.

```
def get_odd_letters(message):
    odd_letters = []
    for counter in range(0, len(message)):
        if not is_even(counter):
            odd_letters.append(message[counter])
    return odd_letters
```



Gambar 4.20 Pengambilan Posisi Ganjil

11. Tukar huruf bulat

    Sekarang Anda memiliki huruf genap dalam satu daftar dan ganjil di tempat lain, Anda dapat menggunakannya untuk mengenkripsi pesan Anda. Fungsi selanjutnya akan mengambil huruf secara bergantian dari daftar ini dan memasukkannya ke dalam daftar baru. Tetapi alih-alih menyusunnya dalam urutan aslinya, dimulai dengan huruf genap, pesan akan dimulai dengan huruf ganjil. Ketik fungsi ini di bawah kode di Langkah 10.

```
def swap_letters(message):
    letter_list = []
    if not is_even(len(message)):
        message = message + 'x'
    even_letters = get_even_letters(message)
    odd_letters = get_odd_letters(message)
    for counter in range(0, int(len(message)/2)):
        letter_list.append(odd_letters[counter])
        letter_list.append(even_letters[counter])
    new_message = ''.join(letter_list)
    return new_message
```

Add an extra x to any message with an odd number of letters.

Loop through the lists of odd and even letters.

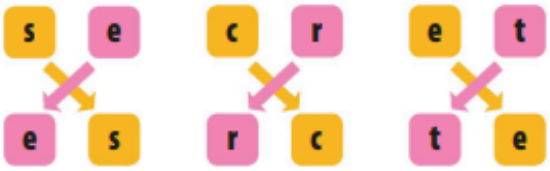
Add the next odd letter to the final message.

Add the next even letter to the final message.

The `join()` function turns the list of letters into a string.

Cara kerjanya

Fungsi swap_letters() menempatkan semua angka ganjil dan genap ke dalam daftar baru, menambahkannya secara bergantian. Itu memulai daftar dengan huruf kedua dalam kata, yang dihitung Python sebagai angka ganjil.



Gambar 4.21 Pengambilan posisi menyilang

Posisi bilangan bulat

Anda menggunakan nilai len(message)/2 dalam rentang loop Anda karena daftar huruf genap dan ganjil keduanya setengah dari panjang pesan asli. Anda memastikan panjang pesan Anda akan selalu genap dengan meminta program untuk menambahkan x bila perlu, sehingga dapat dibagi dengan 2. Namun, hasilnya akan menjadi nilai float (dengan titik desimal, seperti 3.0 atau 4.0) daripada bilangan bulat (bilangan bulat, seperti 3 atau 4). Python memberikan kesalahan jika Anda mencoba menggunakan float untuk posisi item dalam daftar, jadi gunakan fungsi int() untuk mengubahnya menjadi integer.

```
>>> mystring = 'secret'
>>> mystring[3.0]
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    mystring[3.0]
TypeError: string indices must be integers
```

This is the error message Python will give you if you use a float, such as 3.0, instead of an integer, such as 3.

12. Perbarui loop

    Fungsi swap_letters() memiliki fitur yang sangat berguna: jika Anda menjalankannya pada pesan terenkripsi, itu akan mendekripsinya. Jadi Anda dapat menggunakan fungsi ini untuk mengenkripsi atau mendekripsi pesan tergantung pada apa yang ingin dilakukan pengguna. Buat perubahan berikut pada loop sementara yang Anda buat di Langkah 6.

```
while True:
    task = get_task()
    if task == 'encrypt':
        message = get_message()
        encrypted = swap_letters(message)
        messagebox.showinfo('Ciphertext of the secret message is:', encrypted)
    elif task == 'decrypt':
        message = get_message()
        decrypted = swap_letters(message)
        messagebox.showinfo('Plaintext of the secret message is:', decrypted)
    else:
        break
root.mainloop()
```

Use swap_letters() to encrypt the message.

Display the encrypted message.

Uses swap_letters() to decrypt the message.

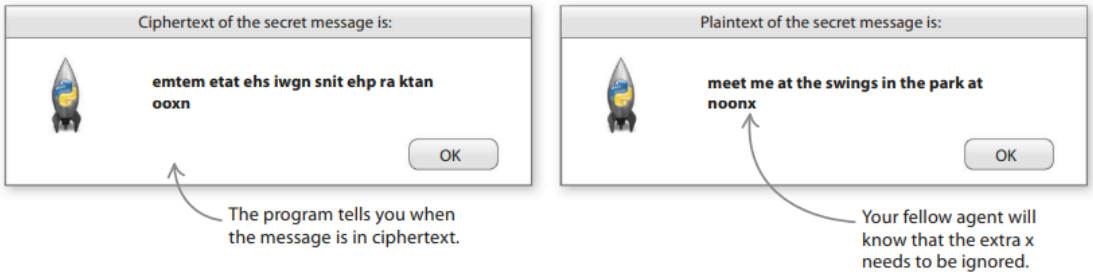
Display the decrypted message.

13. Jalankan enkripsi

    Uji program Anda. Pilih "enkripsi" di jendela tugas. Saat jendela pesan muncul, masukkan jenis pesan yang mungkin ingin dirahasiakan oleh mata-mata. Coba: “Temui aku di ayunan di taman pada siang hari”!

14. Jalankan dekripsi

    Jika Anda memilih teks terenkripsi dan menyalinnya, Anda dapat memilih opsi "dekripsi" pada putaran berikutnya. Di jendela pesan, rekatkan pesan terenkripsi dan klik OK. Anda kemudian akan melihat pesan asli lagi.



Gambar 4.22 Dialog Keterangan yang muncul

15. Dekripsi ini!

    Program cipher Anda sekarang seharusnya berfungsi. Untuk memastikan, coba dekripsi teks yang ditampilkan di sini. Anda sekarang dapat membagikan kode Python Anda dengan teman dan mulai mengirim pesan rahasia!

ewlld no eoy uahevd ceyrtpdet ih sesrctem seaseg

oy uac nsu eelom nujci erom li ksai vnsibieli kn

Peretasan dan penyesuaian

Berikut adalah beberapa ide untuk membuat pesan rahasia Anda semakin sulit dibaca jika dicegat oleh agen musuh— seperti saudara atau saudari yang usil!

Hapus spasi

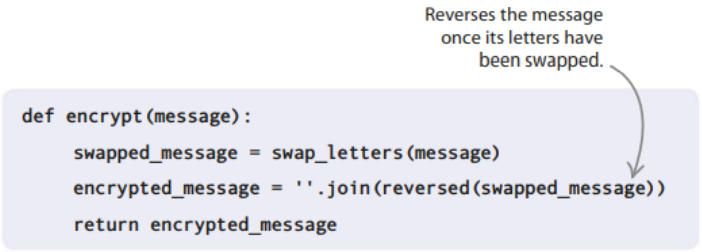
Salah satu cara untuk membuat sandi Anda lebih aman adalah dengan menghilangkan spasi dan karakter tanda baca, seperti titik dan koma. Untuk melakukannya, ketik pesan Anda tanpa spasi dan tanda baca. Pastikan teman yang bertukar pesan dengan Anda tahu bahwa ini adalah rencananya.

4.7 MEMBALIKKAN SETELAH BERTUKAR

Untuk mempersulit orang untuk memecahkan enkripsi Anda, balikkan pesan setelah mengenkripsinya dengan swap_letters(). Untuk melakukan ini, Anda harus membuat dua fungsi berbeda—satu untuk mengenkripsi dan satu lagi untuk mendekripsi.

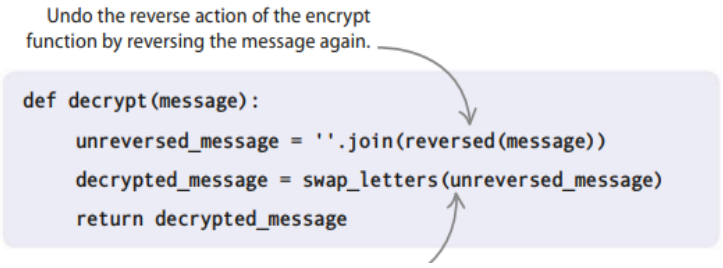
1. Fungsi enkripsi

   Fungsi encrypt() menukar huruf dan kemudian membalikkan string. Ketik baris ini di bawah fungsi swap_letters().



2. Fungsi dekripsi

   Tambahkan fungsi decrypt() ini di bawah fungsi encrypt(). Ini dimulai dengan membalikkan pesan terenkripsi, dan kemudian menggunakan swap_letters() untuk mengembalikan huruf ke urutan yang benar.



3. Gunakan fungsi baru

   Sekarang Anda perlu memperbarui bagian infinite loop dari program Anda untuk menggunakan fungsi-fungsi ini alih-alih fungsi swap_letters().

```
while True:
    task = get_task()
    if task == 'encrypt':
        message = get_message()
        encrypted = encrypt(message)
        messagebox.showinfo('Ciphertext of the secret message is:', encrypted)
    elif task == 'decrypt':
        message = get_message()
        decrypted = decrypt(message)
    messagebox.showinfo('Plaintext of the secret message is:', decrypted)
    else:
       break
```

The new **encrypt()** function replaces **swap_letters()**.

The new **decrypt()** function replaces swap_letters().

Tambahkan huruf "palsu"

Cara lain untuk mengenkripsi pesan adalah dengan menyisipkan huruf acak di antara setiap pasangan huruf. Jadi kata "rahasia" bisa menjadi "stegciraelta" atau "shevcarieste". Sama seperti dalam peretasan “Mundur setelah bertukar”, Anda memerlukan dua fungsi berbeda—satu untuk mengenkripsi dan satu lagi untuk mendekripsi.

Gambar 4.23 Penambahan huruf palsu

1. Tambahkan modul lain

   Impor fungsi choice() dari modul acak. Ini akan memungkinkan Anda memilih surat palsu dari daftar surat. Ketik baris ini di dekat bagian atas file Anda, di bawah perintah untuk mengimpor fungsi Tkinter.

```
from tkinter import messagebox, simpledialog, Tk
from random import choice
```

2. Enkripsi

   Untuk mengenkripsi pesan, Anda perlu mengatur daftar surat palsu untuk disisipkan di antara yang asli. Kode yang ditunjukkan di bawah ini akan mengulang pesan, menambahkan satu huruf asli dan satu huruf palsu ke daftar_enkripsi setiap kali putaran.

```
def encrypt(message):
    encrypted_list = []
    fake_letters = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'i', 'r', 's', 't', 'u', 'v']
    for counter in range(0, len(message)):
        encrypted_list.append(message[counter])
        encrypted_list.append(choice(fake_letters))
    new_message = ''.join(encrypted_list)
    return new_message
```

Add fake letters between real letters.

Add a letter from the message to `encrypted_list`.

Add a fake letter to the `encrypted_list`.

Join the letters in `encrypted_list` into a string.

3. Dekripsi

   Mendekripsi pesan sangat mudah. Dalam versi pesan terenkripsi, semua huruf dalam posisi genap adalah huruf dari pesan asli. Jadi Anda bisa menggunakan fungsi get_even_letters() untuk mendapatkannya.

```
def decrypt(message):
    even_letters = get_even_letters(message)
    new_message = ''.join(even_letters)
    return new_message
```

Get the original message's letters.

Join the letters in `even_letters` into a string.

4. Gunakan fungsi baru

   Sekarang Anda perlu memperbarui bagian infinite loop dari program Anda untuk menggunakan fungsi encrypt() dan decrypt() yang baru, alih-alih swap_letters(). Untuk melakukannya, buat perubahan ini pada kode Anda.

```
while True:
    task = get_task()
    if task == 'encrypt':
        message = get_message()
        encrypted = encrypt(message)
        messagebox.showinfo('Ciphertext of the secret message is:', encrypted)
    elif task == 'decrypt':
        message = get_message()
        decrypted = decrypt(message)
        messagebox.showinfo('Plaintext of the secret message is:', decrypted)
    else:
        break
root.mainloop()
```

The new **encrypt()** function replaces **swap_letters()**.

The new **decrypt()** function replaces **swap_letters()**.

Multienkripsi

Untuk membuat segalanya lebih rumit, Anda dapat memodifikasi kode Anda sehingga menggabungkan semua peretasan dan penyesuaian yang berbeda dari bagian ini. Misalnya, itu bisa menambahkan huruf palsu, menukar huruf, dan kemudian membalikkannya!

4.8 LAYAR HEWAN PELIHARAAN

Pernahkah Anda berharap memiliki hewan peliharaan untuk menemani Anda saat mengerjakan pekerjaan rumah di komputer Anda? Dalam proyek ini, Anda akan membuat hewan peliharaan yang "hidup" di sudut layar komputer Anda. Ini akan membuat Anda sibuk, karena Anda harus merawat hewan peliharaan Anda agar tetap bahagia.

Apa yang terjadi

Saat Anda memulai program, Screen Pet akan duduk di sana, dengan sedikit senyum di wajahnya, berkedip pada Anda. Teman Anda yang imut dan biru langit akan mengubah ekspresinya dari normal (bawah) menjadi bahagia, nakal, atau sedih, tergantung bagaimana Anda berinteraksi dengannya di layar. Tapi jangan khawatir, ini ramah—tidak akan menggigit jika bosan!

Wajah bahagia

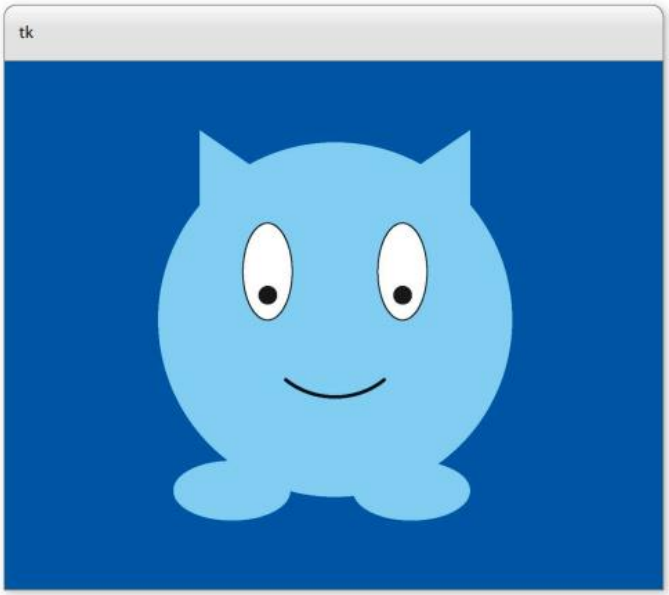
Jika Anda "mengelusnya" dengan penunjuk tetikus, Layar Pet bersinar dan memerah.

Wajah nakal

Jika Anda mengklik dua kali untuk "menggelitik", hewan peliharaan yang nakal itu menjulurkan lidahnya.

Wajah sedih

Jika Anda mengabaikannya, Screen Pet akan menjadi sedih. Mengelusnya akan menghiburnya lagi.



Gambar 4.24 Ekpresi Tersenyum

Bagaimana itu bekerja

Menjalankan fungsi root.mainloop() Tkinter akan menyiapkan loop sementara yang terus memeriksa input dari pengguna. Loop terus berjalan sampai Anda menutup jendela Tkinter utama. Ini juga bagaimana Anda dapat membuat GUI (antarmuka pengguna grafis) yang bereaksi terhadap pengguna yang mengklik tombol atau memasukkan teks di Ask the Expert.

Animasi putaran utama

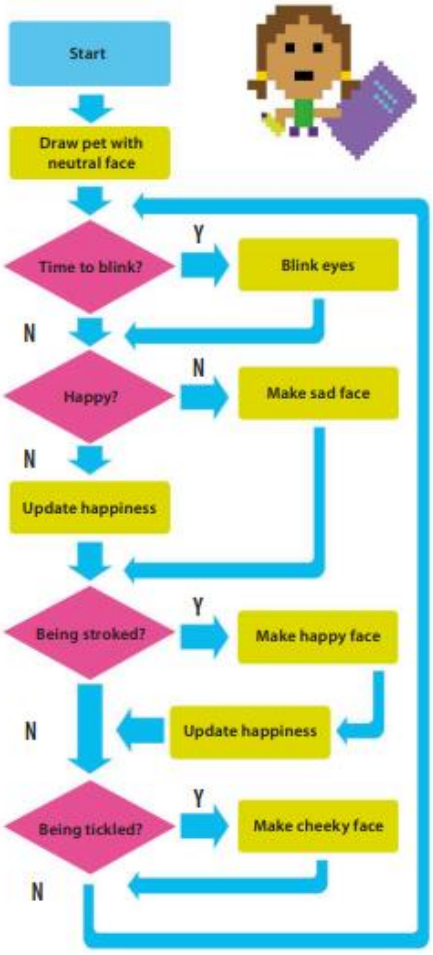
Anda juga dapat menganimasikan gambar di jendela Tkinter menggunakan fungsi root.mainloop(). Dengan menyuruhnya menjalankan fungsi yang mengubah gambar pada waktu yang ditentukan, Anda dapat membuat Screen Pet tampak bergerak dengan sendirinya.

Program berbasis acara

Screen Pet adalah program yang digerakkan oleh peristiwa, yang berarti bahwa hal-hal yang dilakukannya dan urutannya bergantung pada masukan dari pengguna. Program mencari input, seperti penekanan tombol dan klik mouse, kemudian memanggil fungsi yang berbeda untuk menangani masing-masing. Program pengolah kata, video game, dan program menggambar adalah contoh program yang digerakkan oleh peristiwa.

Diagram alur hewan peliharaan layar

Diagram alir menunjukkan urutan tindakan dan keputusan, dan bagaimana masukan pengguna memengaruhinya. Program berjalan dalam lingkaran tanpa akhir. Ia menggunakan variabel kebahagiaan yang selalu berubah untuk melacak suasana hati hewan peliharaan.



Gambar 4.25 Diagram Alur Membuat Hewan Piaraan

Gambar Hewan Peliharaan Layar Anda

Mari kita mulai. Pertama, Anda perlu membuat jendela tempat Screen Pet Anda akan hidup. Kemudian Anda akan menulis beberapa kode untuk menggambar hewan peliharaan di layar.

1. Buat file baru

   Buka IDLE. Buka menu File dan pilih File Baru, lalu simpan file sebagai “screen_pet.py”.
2. Tambahkan modul Tkinter

   Anda perlu mengimpor bagian dari modul Tkinter Python di awal program Anda. Ketik kode ini untuk memasukkan Tkinter dan buka jendela tempat Screen Pet Anda akan hidup.
3. Buat kanvas baru

   Di jendela, buat kanvas biru tua yang disebut "c", di mana Anda akan menggambar hewan peliharaan Anda. Tambahkan kode ini setelah baris yang membuka jendela Tkinter. Empat baris kode baru ini adalah awal dari bagian utama program Anda.

This line imports the parts of the **Tkinter** module that you'll need in this project.

```
from tkinter import HIDDEN, NORMAL, Tk, Canvas
root = Tk()
```

This line starts **Tkinter** and opens a window.

The canvas will be 400 pixels wide and 400 pixels high.

The background colour will be dark blue.

```
from tkinter import HIDDEN, NORMAL, Tk, Canvas
root = Tk()
c = Canvas(root, width=400, height=400)
c.configure(bg='dark blue', highlightthickness=0)
c.pack()
root.mainloop()
```

This command arranges things within the **Tkinter** window.

4. Menjalankannya

   Sekarang coba jalankan programnya. Apa yang Anda perhatikan? Kode seharusnya hanya menampilkan jendela biru tua yang polos. Kelihatannya agak membosankan dan kosong saat ini—yang Anda butuhkan adalah hewan peliharaan!

Gambar 4.26 Program dijalankan

5. Dapatkan menggambar

   Untuk menggambar hewan peliharaan Anda, tambahkan instruksi ini di atas dua baris kode terakhir. Ada perintah terpisah untuk setiap bagian tubuh. Angka-angka, yang disebut koordinat, memberi tahu Tkinter apa yang harus digambar dan di mana menggambarnya.

```
c.configure(bg='dark blue', highlightthickness=0)
c.body_color = 'SkyBlue1'
body = c.create_oval(35, 20, 365, 350, outline=c.body_color, fill=c.body_color)
ear_left = c.create_polygon(75, 80, 75, 10, 165, 70, outline=c.body_color, fill=c.body_color)
ear_right = c.create_polygon(255, 45, 325, 10, 320, 70, outline=c.body_color, \
                             fill=c.body_color)
foot_left = c.create_oval(65, 320, 145, 360, outline=c.body_color, fill= c.body_color)
foot_right = c.create_oval(250, 320, 330, 360, outline=c.body_color, fill= c.body_color)

eye_left = c.create_oval(130, 110, 160, 170, outline='black', fill='white')
pupil_left = c.create_oval(140, 145, 150, 155, outline='black', fill='black')
eye_right = c.create_oval(230, 110, 260, 170, outline='black', fill='white')
pupil_right = c.create_oval(240, 145, 250, 155, outline='black', fill='black')

mouth_normal = c.create_line(170, 250, 200, 272, 230, 250, smooth=1, width=2, state=NORMAL)

c.pack()
```

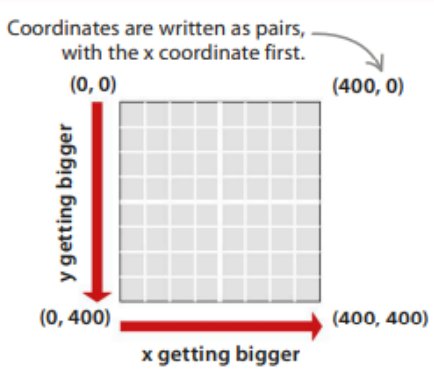
In the code, "left" and "right" refer to the left and right of the window as you look at it.

These pairs of coordinates define the start, mid-point, and end of the mouth.

The mouth is a smooth line, 2 pixels wide.

Koordinat Tkinter

Petunjuk menggambar menggunakan koordinat x dan y. Di Tkinter, koordinat x dimulai dari 0 di sebelah kiri dan meningkat saat Anda bergerak melintasi jendela, hingga mencapai 400 di paling kanan. Koordinat y juga dimulai dari 0 di sebelah kiri. Mereka menjadi lebih besar saat Anda bergerak ke bawah, hingga mencapai 400 di bagian bawah.



Gambar 4.27 Koordinat tkinter

6. Jalankan lagi

   Jalankan program lagi dan Anda akan melihat Screen Pet duduk di tengah jendela Tkinter.

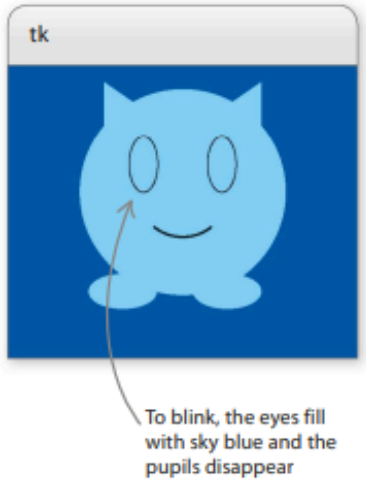
Gambar 4.28 gambar yang sudah ditentukan koordinat

Hewan peliharaan berkedip

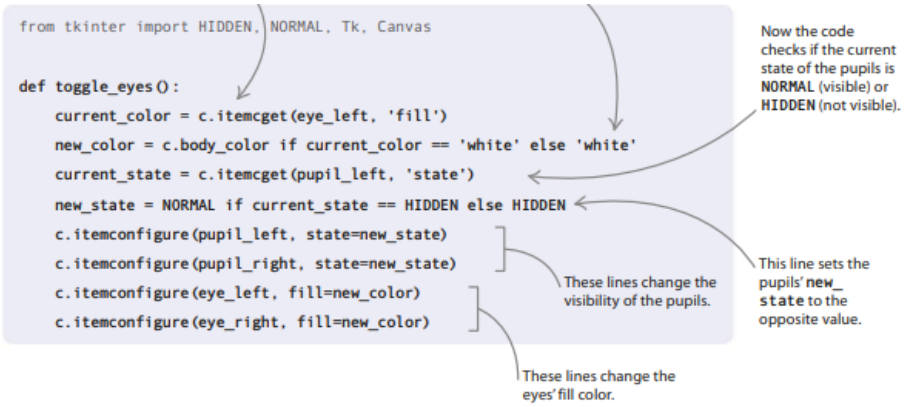
Screen Pet Anda terlihat lucu, tetapi tidak melakukan apa-apa! Mari kita menulis beberapa kode untuk membuatnya berkedip. Anda harus membuat dua fungsi: satu untuk membuka dan menutup mata, yang lain untuk memberi tahu mereka berapa lama untuk tetap terbuka dan tertutup.

7. Buka dan tutup mata

   Buat fungsi ini, toggle_eyes(), di bagian atas file Anda, di bawah baris kode pertama. Itu membuat mata terlihat tertutup dengan menyembunyikan pupil dan mengisi mata dengan warna yang sama dengan tubuh. Itu juga mengubah mata antara terbuka dan tertutup.



Gambar 4.29 Karakter yang berkedip



Beralih

Beralih di antara dua status dikenal sebagai "beralih." Jadi Anda "menghidupkan" lampu di rumah Anda saat Anda menyalakan dan mematikannya. Kode berkedip beralih, atau matikan, antara mata Layar Pet yang terbuka dan tertutup. Jika mata tertutup saat Anda menjalankannya, mereka akan berubah menjadi terbuka. Jika mereka terbuka, mereka akan berubah menjadi tertutup.

8. Berkedip realistis

   Mata hanya perlu menutup sebentar dan tetap terbuka beberapa saat di antara kedipan. Tambahkan fungsi ini, blink(), di bawah kode yang Anda ketik di Langkah 7. Ini mengedipkan mata selama seperempat detik (250 milidetik), lalu diakhiri dengan perintah yang memberi tahu mainloop() untuk memanggilnya lagi setelah 3 detik (3.000 milidetik).

```
    c.itemconfigure(eye_right, fill=new_color)

def blink():
    toggle_eyes()
    root.after(250, toggle_eyes)
    root.after(3000, blink)

root = Tk()
```

Close the eyes.

Wait 250 milliseconds, then open the eyes.

Wait 3,000 milliseconds, then blink again.

9. Menghidupkan!

   Letakkan baris ini di bagian utama program Anda, tepat di atas baris terakhir. Sekarang jalankan programnya. Hewan peliharaan Anda akan hidup kembali setelah 1 detik (1.000 milidetik) dan duduk di sana berkedip sampai Anda menutup jendela.

```
root.after(1000, blink)
root.mainloop()
```

Wait 1,000 milliseconds, then start blinking.

Mengubah suasana hati

Screen Pet terlihat cukup bahagia sekarang, dengan senyum kecilnya, tapi mari kita lebih menghiburnya. Kami akan memberikan senyum yang lebih besar, berseri-seri dan pipi yang cerah dan kemerahan.

10. Buat wajah bahagia

    Tambahkan kode ini ke bagian program yang menggambar Screen Pet, setelah garis yang membuat mulut "normal". Selain mulut bahagia dan pipi merah muda, itu juga menarik mulut sedih. Mereka semua akan tetap tersembunyi untuk saat ini.

```
mouth_normal = c.create_line(170, 250,200, 272, 230, 250, smooth=1, width=2, state=NORMAL)
mouth_happy = c.create_line(170, 250, 200, 282, 230, 250, smooth=1, width=2, state=HIDDEN)
mouth_sad = c.create_line(170, 250, 200, 232, 230, 250, smooth=1, width=2, state=HIDDEN)

cheek_left = c.create_oval(70, 180, 120, 230, outline='pink', fill='pink', state=HIDDEN)
cheek_right = c.create_oval(280, 180, 330, 230, outline='pink', fill='pink', state=HIDDEN)

c.pack()
```

These lines create pink, blushing cheeks.

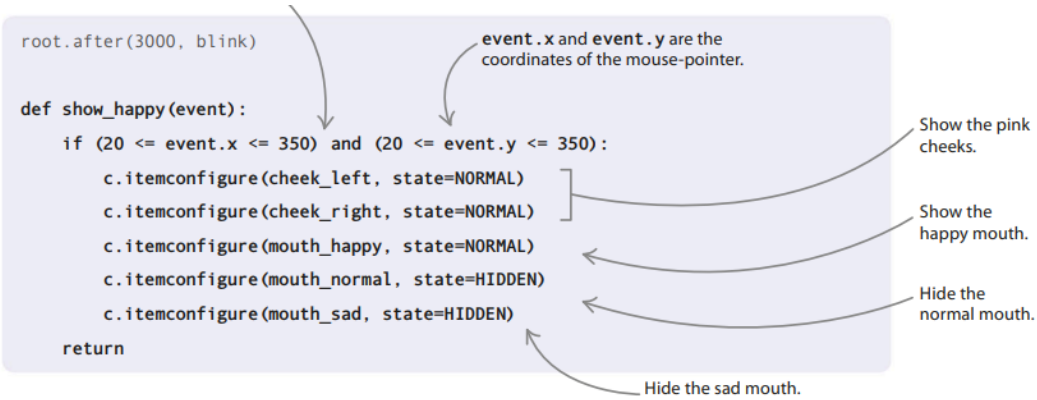
Gambar 4.30 Membuat wajah bahagia

11. Tunjukkan wajah bahagia

    Selanjutnya, buat fungsi bernama show_happy() untuk menampilkan ekspresi bahagia saat Anda menggerakkan penunjuk mouse di atas Layar Hewan Peliharaan seolah-olah Anda sedang mengelusnya. Ketik kode ini di bawah fungsi blink() yang Anda tambahkan di Langkah 8.

    Pengendali acara

    Fungsi show_happy() adalah event handler. Ini berarti hanya dipanggil ketika peristiwa tertentu terjadi, sehingga dapat menanganinya. Dalam kode Anda, membelai hewan peliharaan Anda memanggil show_happy(). Dalam kehidupan nyata, Anda dapat memanggil fungsi "pel lantai" untuk menangani acara "minuman tumpah"!

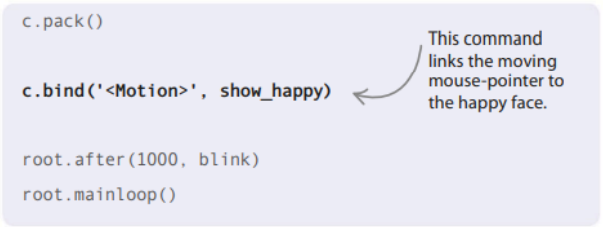


    Fokus

    Tkinter tidak akan dapat melihat Anda menggerakkan penunjuk tetikus di atas jendela untuk mengelus Screen Pet kecuali jika jendela "dalam fokus". Anda bisa membuatnya fokus dengan mengklik sekali di mana saja di jendela.

12. Selamat bergerak

    Saat program dimulai, Hewan Peliharaan Layar berkedip tanpa Anda melakukan apa pun. Tetapi untuk membuatnya terlihat bahagia saat dibelai, Anda perlu memberi tahu acara apa yang harus diwaspadai. Tkinter menyebut penunjuk mouse yang bergerak di atas jendelanya sebagai peristiwa <Motion>. Anda perlu menautkan ini ke fungsi handler dengan menggunakan perintah bind() Tkinter. Tambahkan baris ini ke bagian utama program Anda. Kemudian jalankan kode dan usap hewan peliharaan untuk mencobanya.



13. Sembunyikan wajah bahagia

    Anda hanya ingin Screen Pet terlihat sangat bahagia saat Anda benar-benar mengelusnya. Tambahkan fungsi baru, hide_happy(), di bawah kode untuk show_happy(). Kode baru ini akan mengatur ekspresi Screen Pet kembali normal.

```
def hide_happy(event):
    c.itemconfigure(cheek_left, state=HIDDEN)
    c.itemconfigure(cheek_right, state=HIDDEN)
    c.itemconfigure(mouth_happy, state=HIDDEN)
    c.itemconfigure(mouth_normal, state=NORMAL)
    c.itemconfigure(mouth_sad, state=HIDDEN)
    return
```

Hide the pink cheeks.
Hide the happy mouth.
Show the normal mouth.
Hide the sad mouth.

14. Panggil fungsinya

    Ketik baris ini untuk memanggil hide_happy() saat penunjuk mouse meninggalkan jendela. Ini menautkan acara <Leave> Tkinter ke hide_happy(). Sekarang uji kode Anda.

```
c.bind('<Motion>', show_happy)
c.bind('<Leave>', hide_happy)

root.after(1000, blink)
```

Expresi Hewan Piaraan

Sejauh ini, hewan peliharaan Anda berperilaku sangat baik. Mari kita berikan kepribadian yang nakal! Anda dapat menambahkan beberapa kode yang akan membuat Screen Pet menjulurkan lidahnya dan menyilangkan matanya saat Anda menggelitiknya dengan mengklik dua kali pada kode tersebut.

Gambar 4.31 Membuat Expresi Karakter

15. Menggambar lidah

    Tambahkan baris ini ke kode yang menggambar Screen Pet, di bawah garis yang membuat mulut sedih. Program akan menggambar lidah menjadi dua bagian, persegi panjang dan oval.

```
mouth_sad = c.create_line(170, 250, 200, 232, 230, 250, smooth=1, width=2, state=HIDDEN)
tongue_main = c.create_rectangle(170, 250, 230, 290, outline='red', fill='red', state=HIDDEN)
tongue_tip = c.create_oval(170, 285, 230, 300, outline='red', fill='red', state=HIDDEN)

cheek_left = c.create_oval(70, 180, 120, 230, outline='pink', fill='pink', state=HIDDEN)
```

16. Siapkan bendera

    Tambahkan dua variabel bendera ke kode untuk melacak apakah mata Screen Pet disilangkan atau lidahnya keluar. Ketik tepat di atas baris yang memberi tahu Screen Pet untuk mulai berkedip, yang Anda tambahkan ke bagian utama kode di Langkah 9.

```
c.eyes_crossed = False
c.tongue_out = False


root.after(1000, blink)
```

These are the flag variables for the pupils and the tongue.

17. Alihkan lidah

    Fungsi ini mengubah lidah Hewan Peliharaan Layar antara keluar dan masuk. Letakkan kode yang ditunjukkan di bawah ini di atas fungsi show_happy() yang Anda buat pada Langkah 11.

```
def toggle_tongue():
    if not c.tongue_out:
        c.itemconfigure(tongue_tip, state=NORMAL)
        c.itemconfigure(tongue_main, state=NORMAL)
        c.tongue_out = True
    else:
        c.itemconfigure(tongue_tip, state=HIDDEN)
        c.itemconfigure(tongue_main, state=HIDDEN)
        c.tongue_out = False

def show_happy(event):
```

The code checks to to see if the tongue is out already.

This line sets a flag variable saying the tongue isn't out.

These lines hide the tongue again.

Menggunakan variabel bendera

Variabel flag membantu Anda melacak sesuatu dalam program Anda yang dapat berada di salah satu dari dua status. Saat Anda mengubah status, Anda memperbarui bendera. Tanda "Terlibat / Kosong" di pintu toilet adalah bendera—Anda menyetelnya ke "Terlibat" saat Anda mengunci pintu dan kembali ke "Kosong" saat Anda membukanya.

18. Beralih murid

    Untuk tampilan juling, murid harus menunjuk ke dalam. Fungsi toggle_pupils() ini akan mengalihkan pupil Screen Pet antara menunjuk ke dalam dan terlihat normal. Ketik di bawah fungsi blink() yang Anda tambahkan di Langkah 8.

```
    root.after(3000, blink)


def toggle_pupils():
    if not c.eyes_crossed:
        c.move(pupil_left, 10, -5)
        c.move(pupil_right, -10, -5)
        c.eyes_crossed = True
    else:
        c.move(pupil_left, -10, 5)
        c.move(pupil_right, 10, 5)
        c.eyes_crossed = False
```

The code checks to see if the eyes are crossed already.

If the pupils aren't crossed, this line moves them in.

These lines move the pupils back to normal.

19. Mengkoordinasikan kecerobohan

    Sekarang buat fungsi untuk membuat Screen Pet menjulurkan lidah dan menyilangkan matanya secara bersamaan. Ketik kode ini di bawah fungsi toggle_tongue() yang Anda tambahkan di Langkah 17. Gunakan fungsi root.after() untuk membuat Screen Pet kembali normal setelah 1 detik (1.000 milidetik), seperti yang Anda lakukan di blink().

```
def cheeky(event):
    toggle_tongue()
    toggle_pupils()
    hide_happy(event)
    root.after(1000, toggle_tongue)
    root.after(1000, toggle_pupils)
    return
```

Stick the tongue out.
Cross the pupils.
Hide the happy face.
Put the tongue back in after 1,000 milliseconds.
Uncross the pupils after 1,000 milliseconds.

20. Tautkan klik dua kali ke kecerobohan

    Untuk memicu ekspresi nakal Screen Pet, tautkan acara klik dua kali ke fungsi nakal(). Letakkan baris baru ini tepat di bawah garis yang Anda tambahkan pada Langkah 14 untuk menyembunyikan wajah bahagia Screen Pet. Jalankan kodenya dan klik dua kali untuk melihat keanehannya!

```
c.bind('<Motion>', show_happy)
c.bind('<Leave>', hide_happy)
c.bind('<Double-1>', cheeky)
```

`<Double-1>` is Tkinter's name for a double-click in the window with the mouse.

Hewan peliharaan yang sedih

Terakhir, beri tahu Screen Pet jika Anda tidak memperhatikannya. Setelah hampir satu menit tanpa dibelai, hewan peliharaan Anda yang malang dan terabaikan akan menunjukkan wajah sedihnya!

21. Siapkan tingkat kebahagiaan

    Letakkan baris kode ini tepat di atas variabel flag yang Anda tambahkan ke bagian utama program di Langkah 16. Ini menciptakan tingkat kebahagiaan untuk Screen Pet dan menetapkan level 10 saat Anda menjalankan program dan menggambar hewan peliharaan.

```
c.happy_level = 10
c.eyes_crossed = False
```

Screen Pet starts with a happiness level of 10.

Gambar 4.32 Skala level kebahagiaan

22. Buat perintah baru

    Ketik baris ini di bawah perintah yang Anda tambahkan pada Langkah 9 yang memulai Screen Pet berkedip. Ini memberitahu mainloop() untuk memanggil fungsi sad(), yang akan Anda tambahkan di Langkah 23, setelah 5 detik (5.000 milidetik).

```
root.after(1000, blink)
root.after(5000, sad)
root.mainloop()
```

23. Tulis fungsi sedih

    Tambahkan fungsi ini, sad(), di bawah hide_happy(). Ia memeriksa untuk melihat apakah c.happy_level masih 0. Jika ya, itu akan mengubah ekspresi Screen Pet menjadi sedih. Jika tidak, itu mengurangi 1 dari c.happy_level. Seperti blink(), ini mengingatkan mainloop() untuk memanggilnya lagi setelah 5 detik.

```
def sad():
    if c.happy_level == 0:
        c.itemconfigure(mouth_happy, state=HIDDEN)
        c.itemconfigure(mouth_normal, state=HIDDEN)
        c.itemconfigure(mouth_sad, state=NORMAL)
    else:
        c.happy_level -= 1
    root.after(5000, sad)
```

This line checks to see if the value of `c.happy_level` is 0.

If `c.happy_level` equals 0, the code hides the happy and normal expressions.

This line sets Screen Pet's expression to sad.

The value of `c.happy_level` is greater than 0 (`else`).

Subtract 1 from the value of `c.happy_level`.

Call `sad()` again after 5,000 milliseconds.

24. Bergembiralah, Layar Pet!

    Apakah ada cara untuk menghentikan Screen Pet agar tidak sedih? Atau menghiburnya ketika itu menyedihkan? Untungnya ada—Anda cukup mengeklik jendelanya dan mengelusnya. Tambahkan baris kode ini ke fungsi show_happy() yang Anda tulis di Langkah 11. Sekarang fungsi tersebut akan mereset nilai variabel c.happy_level kembali ke 10 dan membuat Screen Pet menunjukkan wajah bahagianya lagi. Jalankan kode untuk melihat hewan peliharaan Anda sedih, lalu menghiburnya dengan membelainya.

```
        c.itemconfigure(mouth_normal, state = HIDDEN)
        c.itemconfigure(mouth_sad, state = HIDDEN)
        c.happy_level = 10
    return
```

This line puts the happiness level back up to 10.

Peretasan dan penyesuaian

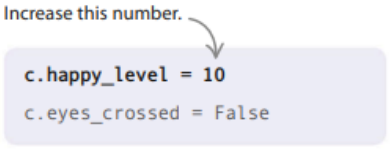
Apakah Screen Pet hewan peliharaan ideal Anda sekarang? Jika tidak, Anda dapat mengubah perilakunya atau menambahkan beberapa fitur tambahan! Berikut adalah beberapa ide untuk mempersonalisasi Hewan Peliharaan Layar Anda.

Bersikap ramah, tidak nakal

Mungkin Anda lebih suka tidak memiliki hewan peliharaan yang nakal? Dapatkan Screen Pet untuk memberi Anda kedipan ramah alih-alih membuat wajah kasar saat Anda mengekliknya dua kali.

Kebahagiaan ekstra

Mungkin akan mengganggu jika Anda harus terus membelai dan menggelitik Hewan Peliharaan Layar saat mengerjakan pekerjaan rumah. Untuk membuatnya lebih jarang sedih, setel nilai c.happy_level ke angka yang lebih tinggi di awal.



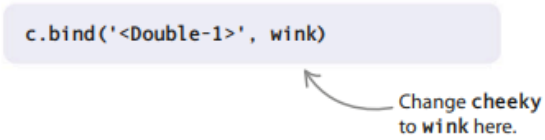
1. Tambahkan fungsi ini di bawah fungsi blink(). Ini mirip dengan kode blink(), tetapi hanya akan mengaktifkan satu mata

```
def toggle_left_eye():
    current_color = c.itemcget(eye_left, 'fill')
    new_color = c.body_color if current_color == 'white'   else 'white'
    current_state = c.itemcget(pupil_left, 'state')
    new_state = NORMAL if current_state == HIDDEN else HIDDEN
    c.itemconfigure(pupil_left, state=new_state)
    c.itemconfigure(eye_left, fill=new_color)
```

2. Fungsi berikutnya menutup dan membuka mata kiri sekali untuk membuat Screen Pet mengedipkan mata. Ketik di bawah toggle_left_eye().

```
def wink(event):
    toggle_left_eye()
    root.after(250, toggle_left_eye)
```

3. Ingatlah untuk mengubah perintah yang mengikat acara klik dua kali (<Double-1>) menjadi wink() alih-alih nakal() di bagian utama program.



4.9 HEWAN PELIHARAAN PELANGI

Sangat mudah untuk membuat *Screen Pet* warna yang berbeda dengan mengubah nilai c.body_color. Jika Anda tidak dapat memutuskan warna apa yang akan dipilih, Anda dapat menambahkan fungsi yang terus mengubah warna *Screen Pet* tanpa henti!

1. Pertama tambahkan baris untuk mengimpor modul acak Python. Letakkan di bawah garis yang memuat fitur Tkinter proyek.

```
from tkinter import HIDDEN, NORMAL, Tk, Canvas
import random
```

2. Sekarang ketikkan fungsi baru, change_color(), tepat di atas bagian utama kode. Ini mengambil nilai baru untuk c.body_color dari daftar pet_colors. Kemudian menggambar ulang tubuh Screen Pet menggunakan warna baru. Karena menggunakan random.choice, kamu tidak akan pernah bisa memastikan warna pet selanjutnya!

```
def change_color():
    pet_colors = ['SkyBlue1', 'tomato', 'yellow', 'purple', 'green', 'orange']
    c.body_color = random.choice(pet_colors)]
    c.itemconfigure(body, outline=c.body_color, fill=c.body_color)
    c.itemconfigure(ear_left, outline=c.body_color, fill=c.body_color)
    c.itemconfigure(ear_right, outline=c.body_color, fill=c.body_color)
    c.itemconfigure(foot_left, outline=c.body_color, fill=c.body_color)
    c.itemconfigure(foot_right, outline=c.body_color, fill=c.body_color)
    root.after(5000, change_color)
```

3. Terakhir, tambahkan ini tepat di atas baris terakhir di bagian utama program untuk membuat mainloop() memanggil change_color() 5 detik (5.000 milidetik) setelah program dimulai.

```
root.after(5000, change_color)
```

Your pet will begin changing color 5 seconds after the program starts.

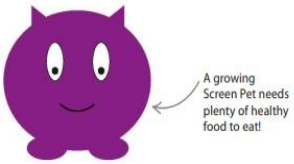
4. Anda mungkin ingin mengubah nilai dalam kode sehingga *Screen Pet* berubah warna lebih cepat. Anda juga dapat mengubah warna dalam daftar menjadi warna yang lebih Anda sukai, atau menambahkan warna tambahan.

Gambar 4.33 Macam Warna hewan piaraan

Beri aku makan!

Hewan peliharaan membutuhkan makanan, serta membelai dan menggelitik. Bisakah Anda menemukan cara untuk memberi makan hewan peliharaan Anda dan menjaganya tetap sehat?

1. Mungkin coba tambahkan "Beri saya makan!" tombol ke jendela *Screen Pet* dan fungsi feed() yang dipanggil saat Anda mengklik tombol.



Gambar 4.34 Gambar Pet ketika di klik

2. Anda bahkan bisa membuat *Screen Pet* tumbuh jika Anda mengklik "Feed me!" beberapa kali. Baris kode ini membuat tubuhnya lebih besar.

```
body = c.create_oval(15, 20, 395, 350, outline=c.body_color, fill=c.body_color)
```

3. Kemudian coba tulis beberapa kode agar tubuh hewan peliharaan Anda menyusut kembali ke ukuran aslinya lagi jika tidak mendapatkan cukup makanan.

   Bersihkan itu!

   Masalah dengan memberi makan Screen Pet adalah ia juga perlu buang air besar! Tulis beberapa kode yang membuatnya kotoran beberapa saat setelah Anda memberinya makan. Kemudian tambahkan tombol "Bersihkan". Mengklik "Bersihkan" akan memanggil fungsi penangan yang menghapus kotoran.

   Jendela yang lebih besar

   Jika Anda menambahkan tombol atau fitur tambahan lainnya ke jendela Screen Pet, mungkin akan sedikit sesak dan tidak nyaman bagi hewan peliharaan Anda. Jika demikian, Anda dapat memperbesar jendela Tkinter. Untuk melakukannya, ubah nilai lebar dan tinggi dalam perintah yang membuat kanvas di awal program utama.

BAB 5
GAME DENGAN PYTHON

5.1 ULAT

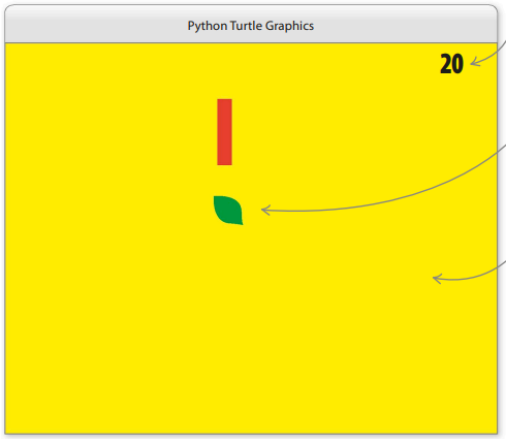
Jika semua pengkodean ini telah membangkitkan selera Anda, Anda tidak sendirian—bintang dari proyek ini adalah ulat yang lapar. Dengan menggunakan modul kura-kura Python, Anda akan mengetahui cara menganimasikan karakter game dan mengontrolnya di layar dengan keyboard.

Apa yang terjadi

Anda menggunakan empat tombol panah untuk mengarahkan ulat di sekitar layar dan membuatnya "memakan" daun. Setiap daun memberi Anda poin, tetapi juga membuat ulat lebih besar dan lebih cepat, membuat permainan lebih sulit. Jauhkan ulat di dalam jendela permainan, atau permainan berakhir!

Meningkatkan kesulitan

Semakin banyak daun yang dimakan ulat, semakin sulit permainannya. Saat ulat semakin panjang dan cepat, reaksi Anda juga harus semakin cepat; jika tidak, ulat Anda akan memperkecil layar.



Gambar 5.1 Permainan Ular

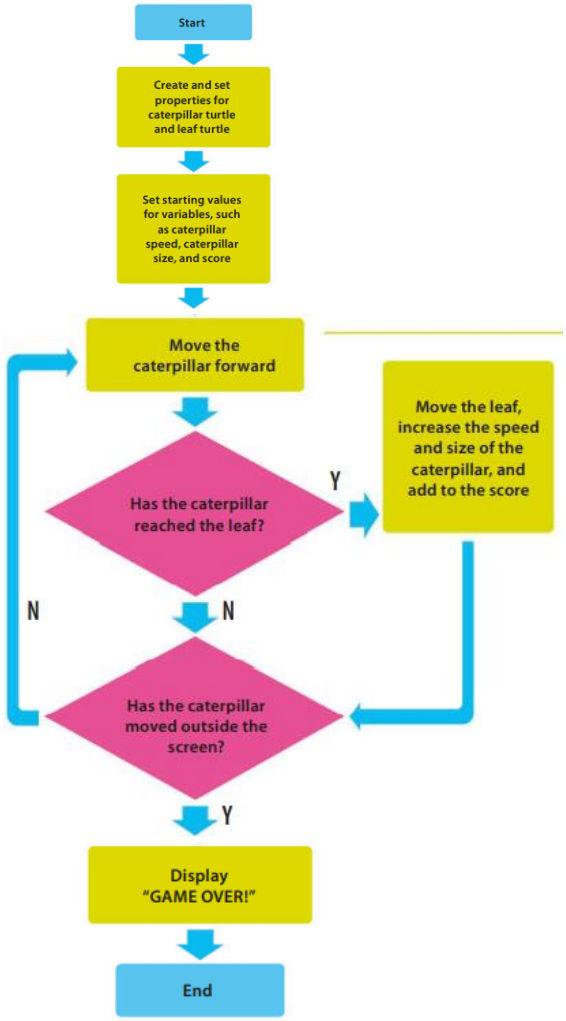
- Skor Anda ditampilkan di sudut kanan atas jendela permainan.
- Daun menghilang saat dimakan, dan daun baru kemudian muncul di tempat lain.
- Untuk memulai permainan, pemain harus mengklik layar terlebih dahulu dan kemudian menekan bilah spasi.

Bagaimana itu bekerja

Proyek ini menggunakan dua kura-kura utama: satu untuk menggambar ulat dan satu lagi untuk menggambar daun. Kode menempatkan setiap daun baru di lokasi acak. Ketika program mendeteksi bahwa daun telah dimakan, variabel yang menyimpan skor, kecepatan ulat, dan panjangnya meningkat. Sebuah fungsi mengetahui apakah ulat telah pindah ke luar jendela, yang akan menandakan akhir dari permainan.

Bagan alir Ulat

Untuk membuat ulat bergerak melintasi layar, Anda akan menggunakan loop tak terbatas. Setiap kali putaran berputar, ulat bergerak sedikit ke depan. Saat loop berulang dengan cepat, gerakan kecil ini menciptakan ilusi bahwa ulat Anda sedang merangkak.



Gambar 5.2 Diagram Alur Permainan Ulat

5.2 LANGKAH PERTAMA

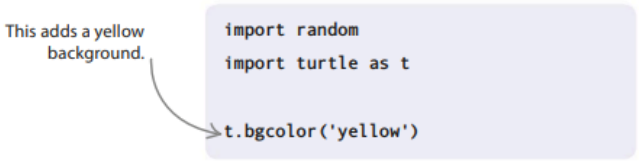
Untuk permainan yang menyenangkan, kodenya sangat mudah. Anda akan mulai dengan menyiapkan kura-kura, sebelum beralih ke loop permainan utama dan akhirnya kontrol keyboard.

1. Mulai

   Buka IDLE dan buat file baru. Simpan sebagai "caterpillar.py".

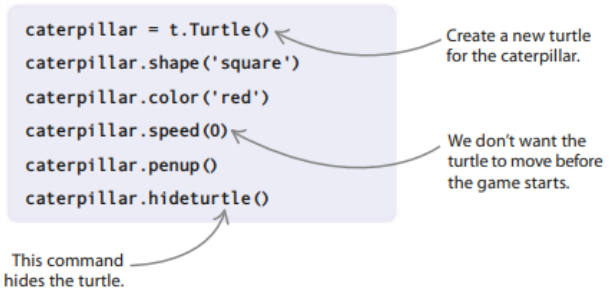
2. Impor modul

   Tambahkan dua pernyataan impor ini untuk memberi tahu Python bahwa Anda memerlukan modul kura-kura dan acak. Baris ketiga mengatur warna latar belakang untuk jendela permainan.



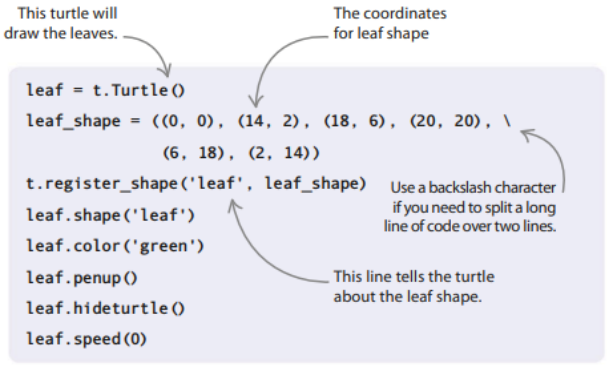
3. Buat ulat kura-kura

   Sekarang buat kura-kura yang akan menjadi ulat Anda. Tambahkan kode yang ditampilkan di sini. Ini menciptakan kura-kura dan mengatur warna, bentuk, dan kecepatannya. Fungsi caterpillar.penup() menonaktifkan pena kura-kura, memungkinkan Anda untuk memindahkan kura-kura di sekitar layar tanpa menggambar garis di sepanjang jalan.



4. Buat kura-kura daun

   Di bawah kode untuk Langkah 3, ketik baris ini untuk mengatur kura-kura kedua, yang akan menggambar daunnya. Kode menggunakan daftar enam pasangan koordinat untuk menggambar bentuk daun. Setelah Anda memberi tahu kura-kura tentang bentuk ini, ia dapat menggunakan kembali detailnya untuk menggambar lebih banyak daun. Panggilan ke hideturtle di sini membuat kura-kura ini tidak terlihat di layar.



5. Tambahkan beberapa teks

   Sekarang siapkan dua kura-kura lagi untuk menambahkan teks ke permainan. Seseorang akan menampilkan pesan sebelum aksi dimulai, memberi tahu pemain untuk menekan bilah spasi untuk memulai. Yang lain akan menulis skor di sudut jendela. Tambahkan baris ini setelah kode penyu daun.

```
game_started = False
text_turtle = t.Turtle()
text_turtle.write('Press SPACE to start', align='center',\
                  font=('Arial', 16, 'bold'))
text_turtle.hideturtle()

score_turtle = t.Turtle()
score_turtle.hideturtle()
score_turtle.speed(0)
```

This line draws some text on the screen.

This hides the turtle but not the text.

Lingkaran utama

Kura-kura Anda sekarang sudah siap dan siap untuk pergi. Mari tulis kode yang membuat game menjadi hidup.

6. Fungsi *placeholder*

   Anda dapat menunda pendefinisian suatu fungsi sampai nanti dengan menggunakan kata kunci pass. Di bawah kode untuk kura-kura, tambahkan placeholder berikut untuk fungsi yang akan Anda isi dengan kode di langkah selanjutnya.

```
def outside_window():
    pass

def game_over():
    pass

def display_score(current_score):
    pass

def place_leaf():
    pass
```

To get a basic version of the program running sooner, you can use placeholders for functions that you'll finish coding later.

Lulus

Dengan Python, jika Anda belum yakin kode apa yang Anda inginkan di dalam suatu fungsi, Anda cukup mengetikkan kata kunci pass dan kembali lagi nanti. Ini seperti menyampaikan pertanyaan dalam kuis.

7. Pemula permainan

   Setelah empat fungsi placeholder, muncul fungsi start_game(), yang menyiapkan beberapa variabel dan menyiapkan layar sebelum loop animasi utama dimulai. Anda akan menambahkan kode untuk loop utama, yang membentuk sisa fungsi ini, di langkah berikutnya.

```
def start_game():
    global game_started
    if game_started:
        return
    game_started = True

    score = 0
    text_turtle.clear()

    caterpillar_speed = 2
    caterpillar_length = 3
    caterpillar.shapesize(1, caterpillar_length, 1)
    caterpillar.showturtle()
    display_score(score)
    place_leaf()
```

If the game has already started, the return command makes the function quit so it doesn't run a second time.

Clear the text from the screen.

8. Bergerak

   Putaran utama menggerakkan ulat sedikit ke depan, sebelum melakukan dua pemeriksaan. Ini pertama memeriksa apakah ulat telah mencapai daun. Jika daun telah dimakan, skornya meningkat, daun baru akan ditarik, dan ulat menjadi lebih panjang dan lebih cepat. Loop kemudian memeriksa apakah ulat telah meninggalkan jendela—jika demikian, permainan berakhir. Tambahkan loop utama di bawah kode yang Anda ketik di Langkah 7.

```
place_leaf()

while True:
    caterpillar.forward(caterpillar_speed)
    if caterpillar.distance(leaf) < 20:
        place_leaf()
        caterpillar_length = caterpillar_length + 1
        caterpillar.shapesize(1, caterpillar_length, 1)
        caterpillar_speed = caterpillar_speed + 1
        score = score + 10
        display_score(score)
    if outside_window():
        game_over()
        break
```

The caterpillar eats the leaf when it's less than 20 pixels away.

The current leaf has been eaten, so add a new leaf.

This will make the caterpillar grow longer.

9. Ikat dan dengarkan

   Sekarang letakkan garis-garis ini di bawah fungsi yang baru saja Anda buat. Fungsi onkey() mengikat bilah spasi ke start_game(), sehingga Anda dapat menunda mulai hingga pemain menekan spasi. Fungsi listen() memungkinkan program menerima sinyal dari keyboard.

```
t.onkey(start_game, 'space')
t.listen()
t.mainloop()
```

When you press the space bar, the game begins.

10. Uji kode Anda

    Jalankan programnya. Jika kode Anda benar, Anda akan melihat ulat bergerak setelah Anda menekan spasi. Akhirnya, itu harus merangkak dari layar. Jika program tidak bekerja, periksa kode Anda dengan hati-hati untuk bug.

    Mengisi kekosongan

    Saatnya mengganti pass di fungsi placeholder dengan kode yang sebenarnya. Setelah menambahkan kode untuk setiap fungsi, jalankan game untuk melihat perbedaannya.

11. Tetap didalam

    Isi fungsi outside_window() dengan kode ini. Pertama menghitung posisi masing-masing dinding. Kemudian ia meminta ulat untuk posisinya saat ini. Dengan membandingkan koordinat ulat dengan koordinat dinding, dapat diketahui apakah ulat telah meninggalkan jendela. Jalankan program untuk memeriksa fungsinya—ulat harus berhenti ketika mencapai tepi.

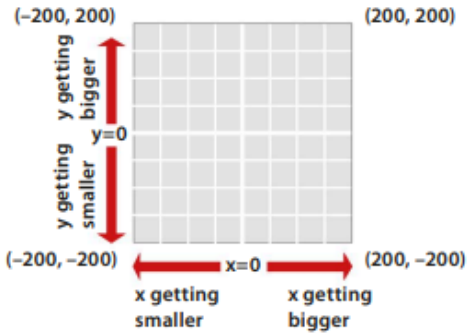
```
def outside_window():
    left_wall = -t.window_width() / 2
    right_wall = t.window_width() / 2
    top_wall = t.window_height() / 2
    bottom_wall = -t.window_height() / 2
    (x, y) = caterpillar.pos()
    outside = \
            x< left_wall or \
            x> right_wall or \
            y< bottom_wall or \
            y> top_wall
    return outside
```

This function returns two values (a "tuple").

If any of the four conditions above is True, then `outside` is True.

Bagaimana itu bekerja

Bagian tengah jendela memiliki koordinat (0, 0). Karena lebar jendela 400, lebar dinding kanan setengah dari tengah, yaitu 200. Kode mendapatkan posisi dinding kiri dengan mengurangkan setengah lebar dari 0. Dengan kata lain, 0–200, yaitu –200. Ia menemukan posisi dinding atas dan bawah dengan metode yang sama.



Gambar 5.3 Koordinat pada *Game*

12. TAMAT!

    Ketika ulat telah meninggalkan layar, tampilkan pesan untuk memberi tahu pemain bahwa permainan telah berakhir. Isi fungsi game_over() dengan kode ini. Saat dipanggil, fungsi tersebut akan menyembunyikan ulat dan daun, dan menulis "GAME OVER!" di layar.

```
def game_over():
    caterpillar.color('yellow')
    leaf.color('yellow')
    t.penup()
    t.hideturtle()
    t.write('GAME OVER!', align='center', font=('Arial', 30, 'normal'))
```

The text should be centered.

13. Tunjukkan skornya

    Fungsi display_score() menginstruksikan kura-kura skor untuk menulis ulang skor, menempatkan total terbaru di layar. Fungsi ini dipanggil setiap kali ulat mencapai daun.

```
def display_score(current_score):
    score_turtle.clear()
    score_turtle.penup()
    x = (t.window_width() / 2) - 50
    y = (t.window_height() / 2) - 50
    score_turtle.setpos(x, y)
    score_turtle.write(str(current_score), align='right', \
                       font=('Arial', 40, 'bold'))
```

50 pixels from the right

14. Daun baru

    Saat daun tercapai, fungsi place_leaf() dipanggil untuk memindahkan daun ke lokasi baru yang acak. Ia memilih dua angka acak antara –200 dan 200. Angka-angka ini menjadi koordinat x dan y untuk daun berikutnya.

```
def place_leaf():
    leaf.ht()
    leaf.setx(random.randint(-200, 200))
    leaf.sety(random.randint(-200, 200))
    leaf.st()
```

ht is short for hideturtle.

Chooses random coordinates to move the leaf.

st is short for showturtle.

15. Menghidupkan ulat

    Selanjutnya, untuk menghubungkan tombol keyboard ke caterpillar, tambahkan empat fungsi arah baru setelah fungsi start_game(). Untuk membuat permainan ini sedikit lebih rumit, ulat hanya bisa berbelok 90 derajat. Akibatnya, setiap fungsi pertama-tama memeriksa untuk melihat ke arah mana ulat itu bergerak sebelum mengubah arahnya. Jika ulat berjalan ke arah yang salah, fungsi menggunakan setheading() untuk membuatnya menghadap ke arah yang benar.

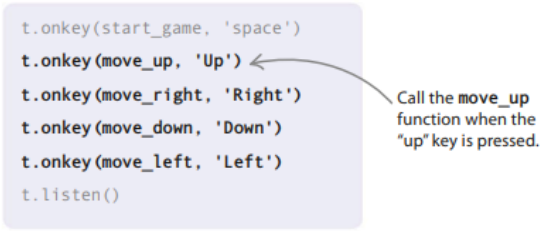
```
            game_over()
            break

def move_up():
    if caterpillar.heading() == 0 or caterpillar.heading() == 180:
        caterpillar.setheading(90)

def move_down():
    if caterpillar.heading() == 0 or caterpillar.heading() == 180:
        caterpillar.setheading(270)

def move_left():
    if caterpillar.heading() == 90 or caterpillar.heading() == 270:
        caterpillar.setheading(180)

def move_right():
    if caterpillar.heading() == 90 or caterpillar.heading() == 270:
        caterpillar.setheading(0)
```

Check if the caterpillar is heading left or right.

A heading of 270 sends the caterpillar down the screen.

16. Mendengarkan pers

    Terakhir, gunakan onkey() untuk menautkan fungsi arah ke tombol keyboard. Tambahkan baris ini setelah panggilan onkey() yang Anda buat di Langkah 9. Dengan kode kemudi di tempat, permainan selesai. Bersenang-senang bermain dan mencari tahu skor tertinggi Anda!



Peretasan dan penyesuaian

Sekarang setelah permainan ulat Anda berfungsi, tidak akan terlalu sulit untuk memodifikasinya atau bahkan memperkenalkan ulat pembantu atau saingan!

Jadikan ini permainan dua pemain

Dengan membuat kura-kura ulat kedua dengan kontrol keyboard terpisah, Anda dan seorang teman dapat bekerja sama untuk membuat ulat memakan lebih banyak daun!

1. Buat ulat baru

   Pertama, Anda harus menambahkan ulat baru. Ketik baris ini di dekat bagian atas program Anda, di bawah kode yang membuat ulat pertama.

```
caterpillar2 = t.Turtle()
caterpillar2.color('blue')
caterpillar2.shape('square')
caterpillar2.penup()
caterpillar2.speed(0)
caterpillar2.hideturtle()
```

2. Tambahkan parameter

   Untuk menggunakan kembali fungsi outside_window() untuk kedua ulat, tambahkan parameter ke dalamnya. Sekarang Anda dapat mengetahui ulat mana yang ingin Anda periksa.

```
def outside_window(caterpillar):
```

3. Sembunyikan ulat2

   Saat fungsi game_over() dipanggil, ia menyembunyikan ulat pertama. Mari tambahkan garis untuk menyembunyikan ulat kedua juga.

```
def game_over():
    caterpillar.color('yellow')
    caterpillar2.color('yellow')
    leaf.color('yellow')
```

4. Ubah fungsi utama

   Anda harus menambahkan kode untuk caterpillar2 ke fungsi start_game() utama. Pertama atur bentuk awalnya dan buat menghadap ke arah yang berlawanan dari ulat pertama. Kemudian tambahkan ke loop while untuk membuatnya bergerak, dan tambahkan tanda centang pada pernyataan if sehingga bisa memakan daunnya. Anda juga perlu menambahkan garis untuk membuatnya tumbuh. Terakhir, edit panggilan

ke fungsi outside_window() dalam pernyataan if kedua Anda untuk melihat apakah permainan sudah selesai.

```
score = 0
text_turtle.clear()

caterpillar_speed = 2
caterpillar_length = 3
caterpillar.shapesize(1, caterpillar_length, 1)
caterpillar.showturtle()
caterpillar2.shapesize(1, caterpillar_length, 1)
caterpillar2.setheading(180)
caterpillar2.showturtle()
display_score(score)
place_leaf()
```

This sets caterpillar2's starting shape.

Caterpillar2 starts heading left.

```
while True:
    caterpillar.forward(caterpillar_speed)
    caterpillar2.forward(caterpillar_speed)
    if caterpillar.distance(leaf) < 20 or leaf.distance(caterpillar2) < 20:
        place_leaf()
        caterpillar_length = caterpillar_length + 1
        caterpillar.shapesize(1, caterpillar_length, 1)
        caterpillar2.shapesize(1, caterpillar_length, 1)
        caterpillar_speed = caterpillar_speed + 1
        score = score + 10
        display_score(score)
    if outside_window(caterpillar) or outside_window(caterpillar2):
        game_over()
```

Each time the program loops, caterpillar2 moves forward.

This checks if caterpillar2 has eaten the leaf.

Caterpillar2 gets longer.

Has caterpillar2 left the screen?

5. Kontrol ekstra

   Sekarang tetapkan kunci yang akan digunakan pemain kedua untuk mengontrol ulat baru. Kode di sini menggunakan "w" untuk atas, "a" untuk kiri, "s" untuk bawah, dan "d" untuk kanan, tetapi jangan ragu untuk mencoba pilihan yang berbeda. Anda memerlukan empat fungsi baru dan empat penggunaan onkey untuk mengikat tombol baru ke fungsi baru.

```
def caterpillar2_move_up():
    if caterpillar2.heading() == 0 or caterpillar2.heading() == 180:
        caterpillar2.setheading(90)

def caterpillar2_move_down():
    if caterpillar2.heading() == 0 or caterpillar2.heading() == 180:
        caterpillar2.setheading(270)

def caterpillar2_move_left():
    if caterpillar2.heading() == 90 or caterpillar2.heading() == 270:
        caterpillar2.setheading(180)

def caterpillar2_move_right():
    if caterpillar2.heading() == 90 or caterpillar2.heading() == 270:
        caterpillar2.setheading(0)

t.onkey(caterpillar2_move_up, 'w')
t.onkey(caterpillar2_move_right, 'd')
t.onkey(caterpillar2_move_down, 's')
t.onkey(caterpillar2_move_left, 'a')
```

Buatlah kompetitif

Lihat apakah Anda dapat mengetahui cara mengadaptasi permainan dua pemain untuk mencatat skor setiap pemain dan kemudian mengumumkan pemenangnya di akhir. Berikut tipnya: Anda memerlukan variabel baru untuk melacak skor pemain kedua. Saat ulat memakan daun, Anda hanya perlu menambahkan poin ke skor ulat itu. Terakhir, saat permainan selesai, Anda dapat membandingkan skor untuk melihat siapa yang menang.

Buat lebih sulit atau lebih mudah

Jika Anda mengubah nilai di dalam lingkaran yang menambah panjang (+1) dan kecepatan (+2) ulat, Anda dapat mengubah tingkat kesulitan permainan. Angka yang lebih tinggi akan membuat permainan lebih sulit, sedangkan angka yang lebih rendah akan membuatnya lebih mudah.

5.3 SNAP

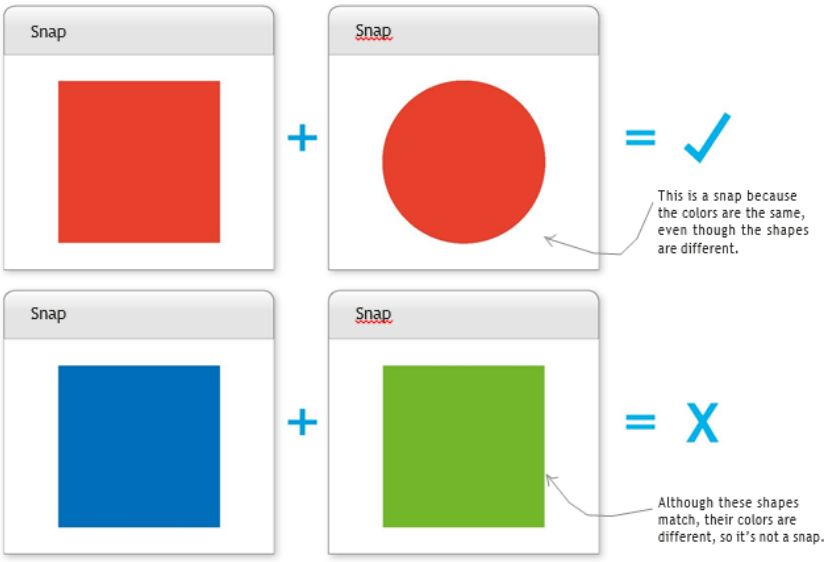
Tantang teman Anda ke permainan jepret digital. Gim dua pemain yang serba cepat ini membutuhkan mata yang tajam dan reaksi secepat kilat. Ini bekerja seperti permainan kartu tetapi menggunakan bentuk berwarna yang muncul di layar daripada kartu yang dibagikan.

Apa yang terjadi

Berbagai bentuk muncul di layar secara acak dalam warna hitam, merah, hijau, atau biru. Jika warna muncul dua kali berturut-turut, tekan tombol jepret. Pemain 1 menekan tombol "q" untuk mengambil dan pemain 2 menekan tombol "p". Setiap jepretan yang benar mencetak poin. Snap pada waktu yang salah dan Anda kehilangan poin. Pemain dengan skor tertinggi adalah pemenangnya.

Memulai permainan

Game ini bekerja di jendela Tkinter. Saat Anda memulai program, jendela Tkinter mungkin tersembunyi di balik jendela IDLE di desktop Anda. Pindahkan mereka agar Anda dapat melihat permainannya. Namun, cepatlah: bentuk jepret mulai muncul 3 detik setelah Anda menjalankan program.



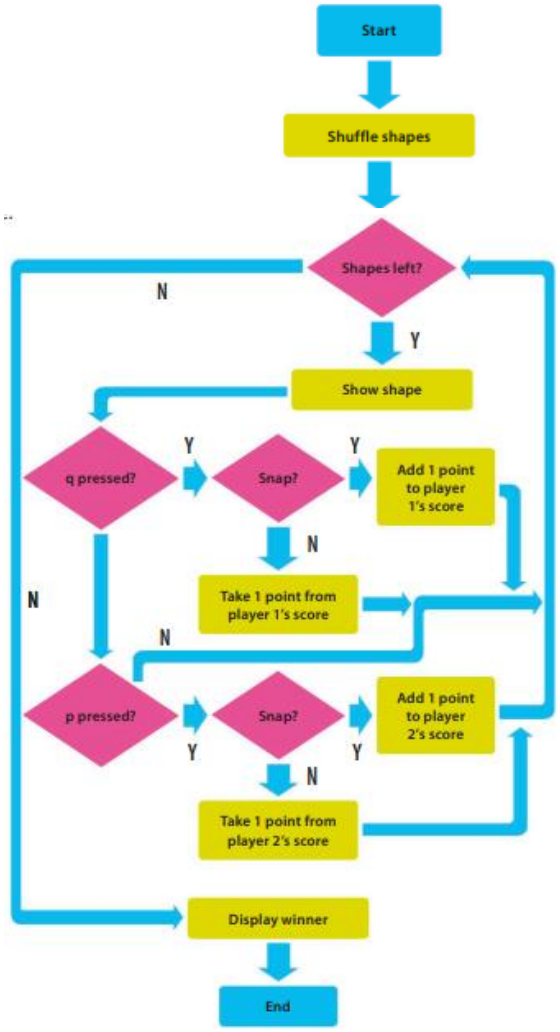
Gambar 5.4 Snap yang sama akan memulai permainan

Bagaimana itu bekerja

Proyek ini menggunakan Tkinter untuk membuat bentuk. Fungsi mainloop() Tkinter menjadwalkan fungsi yang akan Anda buat untuk menampilkan bentuk berikutnya. Fungsi acak modul shuffle() memastikan bentuk selalu muncul dalam urutan yang berbeda. Tombol "q" dan "p" terikat (ditautkan) ke fungsi snap(), sehingga setiap kali salah satu tombol ini ditekan, skor pemain yang bersangkutan akan diperbarui.

Jepret diagram alur

Program berjalan selama masih ada bentuk yang tersisa untuk diungkap. Ini bereaksi terhadap penekanan tombol pemain ketika mereka berpikir mereka melihat snap. Ketika tidak ada lagi bentuk yang tersisa, pemenang diumumkan dan permainan berakhir.



Gambar 5.5 Flowchart memulai pertandingan

Mulai

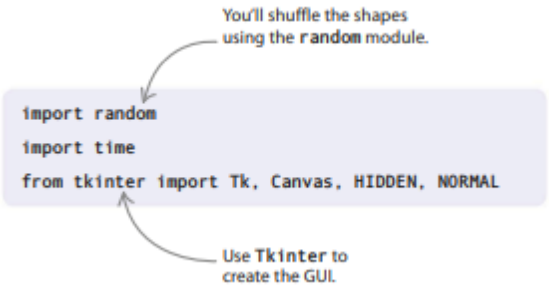
Pertama, Anda perlu mengimpor modul yang relevan dan membuat antarmuka pengguna grafis (GUI). Kemudian Anda perlu membuat kanvas untuk menggambar bentuk.

1. Buat file baru

   Buka IDLE. Buat file baru dan simpan sebagai "snap.py".

2. Tambahkan modul

   Pertama-tama impor modul acak dan waktu, dan bagian dari Tkinter. Waktu memungkinkan Anda membuat penundaan sehingga pemain dapat membaca "SNAP!" atau "SALAH!" pesan sebelum bentuk berikutnya ditampilkan. HIDDEN memungkinkan Anda menyembunyikan setiap bentuk hingga Anda ingin menampilkannya dengan NORMAL—jika tidak, semua bentuk akan muncul di layar pada awal permainan.



3. Siapkan GUI

   Sekarang ketik kode yang ditampilkan di sini untuk membuat jendela Tkinter (juga disebut widget root) dengan judul "Snap". Jalankan kode untuk memeriksanya. Jendela mungkin tersembunyi di balik jendela lain di desktop.

```
from tkinter import Tk, Canvas, HIDDEN, NORMAL

root = Tk()
root.title('Snap')
```

4. Buat kanvas

   Ketik baris ini untuk membuat kanvas—ruang kosong tempat bentuk akan muncul.

```
root.title('Snap')
c = Canvas(root, width=400, height=400)
```

Membuat bentuk

Tahap selanjutnya adalah membuat bentuk berwarna menggunakan fungsi dari widget Canvas Tkinter. Anda akan menggambar lingkaran, kotak, dan persegi panjang, masing-masing dalam empat warna berbeda.

5. Buat toko untuk bentuknya

   Anda perlu membuat daftar sehingga Anda dapat menyimpan semua bentuk di suatu tempat. Tambahkan baris ini di bagian bawah file Anda.

```
c = Canvas(root, width=400, height=400)

shapes = []
```

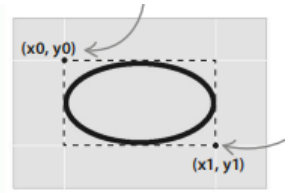
6. Buat lingkaran

   Untuk menggambar lingkaran, gunakan fungsi create_oval() widget Canvas. Ketik kode berikut di bawah daftar bentuk. Ini membuat empat lingkaran dengan ukuran yang sama—masing-masing berwarna hitam, merah, hijau, dan biru—dan menambahkannya ke daftar bentuk.

```
shapes = []

circle = c.create_oval(35, 20, 365, 350, outline='black', fill='black', state=HIDDEN)
shapes.append(circle)
circle = c.create_oval(35, 20, 365, 350, outline='red', fill='red', state=HIDDEN)
shapes.append(circle)
circle = c.create_oval(35, 20, 365, 350, outline='green', fill='green', state=HIDDEN)
shapes.append(circle)
circle = c.create_oval(35, 20, 365, 350, outline='blue', fill='blue', state=HIDDEN)
shapes.append(circle)
c.pack()
```


Buat oval

Fungsi create.oval() menggambar oval seolah-olah berada di dalam kotak tak terlihat. Empat angka di dalam tanda kurung menentukan posisi lingkaran di layar. Mereka adalah koordinat dari dua sudut kotak yang berlawanan. Semakin besar perbedaan antara dua pasang angka, semakin besar lingkarannya.



Gambar 5.6 Membentuk Oval

7. Tunjukkan lingkarannya

   Coba jalankan programnya. Apakah Anda melihat bentuk apapun? Ingatlah bahwa Anda menyetel statusnya ke TERSEMBUNYI. Ubah status satu bentuk menjadi NORMAL dan jalankan kode lagi. Anda sekarang seharusnya dapat melihat bentuk itu di layar. Berhati-hatilah untuk tidak mengatur lebih dari satu bentuk ke NORMAL. Jika Anda melakukannya, semuanya akan ditampilkan sekaligus, digambar satu di atas yang lain.

8. Tambahkan beberapa persegi panjang

   Sekarang buat empat persegi panjang berwarna berbeda menggunakan fungsi create_rectangle() Canvas. Sisipkan blok kode ini di antara kode gambar lingkaran dan c.pack(). Untuk menghindari mengetik semuanya, cukup ketik dua baris pertama, lalu salin dan tempel tiga kali dan ubah warnanya.

```
shapes.append(circle)

rectangle = c.create_rectangle(35, 100, 365, 270, outline='black', fill='black', state=HIDDEN)
shapes.append(rectangle)
rectangle = c.create_rectangle(35, 100, 365, 270, outline='red', fill='red', state=HIDDEN)
shapes.append(rectangle)
rectangle = c.create_rectangle(35, 100, 365, 270, outline='green', fill='green', state=HIDDEN)
shapes.append(rectangle)
rectangle = c.create_rectangle(35, 100, 365, 270, outline='blue', fill='blue', state=HIDDEN)
shapes.append(rectangle)
c.pack()
```

9. Tambahkan beberapa kotak

   Selanjutnya menggambar kotak. Anda dapat menggunakan fungsi yang sama yang Anda gunakan untuk membuat persegi panjang, tetapi kali ini Anda akan mengubah

persegi panjang menjadi persegi dengan membuat semua sisinya sama panjang. Tambahkan blok kode ini di antara kode persegi panjang dan c.pack().

```
shapes.append(rectangle)

square = c.create_rectangle(35, 20, 365, 350, outline='black', fill='black', state=HIDDEN)
shapes.append(square)
square = c.create_rectangle(35, 20, 365, 350, outline='red', fill='red', state=HIDDEN)
shapes.append(square)
square = c.create_rectangle(35, 20, 365, 350, outline='green', fill='green', state=HIDDEN)
shapes.append(square)
square = c.create_rectangle(35, 20, 365, 350, outline='blue', fill='blue', state=HIDDEN)
shapes.append(square)
c.pack()
```

10. Acak bentuknya

    Untuk memastikan bahwa bentuk tidak muncul dalam urutan yang sama setiap kali, Anda perlu mengocoknya – seperti yang akan Anda lakukan dengan setumpuk kartu. Fungsi shuffle() secara acak dapat melakukan ini untuk Anda. Sisipkan baris ini setelah c.pack().

```
random.shuffle(shapes)
```

Bersiap-siap

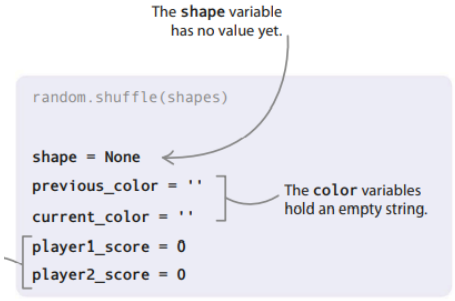
Di bagian build selanjutnya, Anda akan menyiapkan beberapa variabel dan menulis beberapa bit kode yang menyiapkan game untuk dimainkan. Namun, itu tidak akan berfungsi sampai kami menambahkan fungsi di tahap terakhir.

Tidak ada yang benar-benar penting

Pembuat kode sering kali perlu mengatur variabel dengan nilai awal nol, seperti skor dalam game ini. Tetapi bagaimana Anda melakukan ini jika variabel memegang string daripada angka? Jawabannya adalah dengan menggunakan sepasang tanda kutip tanpa apa-apa di antara mereka. Beberapa variabel, bagaimanapun, tidak memiliki nilai default yang jelas seperti 0 atau string kosong. Dalam hal ini, Anda dapat menggunakan kata "Tidak Ada", seperti yang kami lakukan di bawah ini.

11. Mengatur variabel

    Anda memerlukan variabel untuk melacak berbagai hal saat program sedang berjalan, termasuk bentuk saat ini, warna sebelumnya dan saat ini, dan skor kedua pemain.



12. Tambahkan penundaan

    Sekarang tambahkan garis untuk membuat penundaan 3 detik sebelum bentuk pertama muncul. Ini memberi pemain waktu untuk menemukan jendela Tkinter jika

tersembunyi di balik jendela lain di desktop. Anda akan membuat fungsi next_shape() nanti, di Langkah 16 dan 17.

```
player2_score = 0

root.after(3000, next_shape)
```

The program waits for 3,000 milliseconds, or 3 seconds before showing the next shape.

13. Bereaksi terhadap bidikan

    Selanjutnya tambahkan dua baris ini ke kode Anda. Fungsi bind() memberi tahu GUI untuk mendengarkan tombol "q" atau "p" yang ditekan, dan untuk memanggil fungsi snap() setiap kali itu terjadi. Anda akan membuat fungsi snap() nanti.

```
root.after(3000, next_shape)
c.bind('q', snap)
c.bind('p', snap)
```

14. Kirim penekanan tombol ke GUI

    Fungsi focus_set() memberitahu penekanan tombol untuk pergi ke kanvas. GUI tidak akan bereaksi terhadap "q" dan "p" yang ditekan tanpa fungsi ini dipanggil. Ketik baris ini di bawah panggilan fungsi bind().

```
c.bind('q', snap)
c.bind('p', snap)
c.focus_set()
```

15. Mulai putaran utama

    Tambahkan baris ini tepat di akhir file Anda. Setelah kita menambahkan fungsi next_shape() dan snap(), loop utama akan memperbarui GUI dengan bentuk berikutnya dan mendengarkan penekanan tombol.

```
c.focus_set()

root.mainloop()
```

Variabel lokal dan global

Variabel dapat berupa lokal atau global. Variabel lokal hanya ada di dalam fungsi tertentu, yang berarti program lainnya tidak dapat menggunakannya. Variabel yang dibuat di program utama, di luar fungsi, disebut global dan dapat digunakan di bagian kode mana pun. Namun, jika Anda ingin menggunakan fungsi untuk menetapkan nilai baru ke variabel global, Anda perlu menambahkan kata kunci global sebelum nama variabel saat Anda mengetiknya di fungsi. Inilah yang kami lakukan di Langkah 16.

Mengkodekan fungsi

Tahap terakhir adalah membuat dua fungsi: satu untuk menunjukkan bentuk berikutnya, dan satu lagi untuk menangani kancing. Ketik di bagian atas program Anda, tepat di bawah pernyataan impor.

16. Buat fungsinya

Fungsi next_shape() menunjukkan bentuk berwarna satu demi satu, seperti kartu yang dibagikan. Mulailah mendefinisikan fungsi dengan mengetikkan kode di bawah ini. Ini melabeli beberapa variabel Anda sebagai global (lihat kotak, kiri) dan memperbarui warna_sebelumnya.

```
def next_shape():
    global shape
    global previous_color
    global current_color

    previous_color = current_color
```

This line sets `previous_color` to `current_color` before the code gets the next shape.

17. Lengkapi fungsinya

Sekarang ketik sisa fungsi. Untuk menampilkan bentuk baru, kita perlu mengubah statusnya dari HIDDEN ke NORMAL. Kode di bawah ini melakukan ini dengan menggunakan fungsi itemconfigure() Canvas. Ia menggunakan fungsi Canvas lain, itemcget(), untuk memperbarui variabel current_color, yang akan digunakan untuk memeriksa snap.

```
previous_color = current_color

c.delete(shape)
if len(shapes) > 0:
    shape = shapes.pop()
    c.itemconfigure(shape, state=NORMAL)
    current_color = c.itemcget(shape, 'fill')
    root.after(1000, next_shape)
else:
    c.unbind('q')
    c.unbind('p')
    if player1_score > player2_score:
        c.create_text(200, 200, text='Winner: Player 1')
    elif player2_score > player1_score:
        c.create_text(200, 200, text='Winner: Player 2')
    else:
        c.create_text(200, 200, text='Draw')
    c.pack()
```

Delete the current shape, so that the next shape doesn't show on top of it and so that it won't be shown again.

Get the next shape if there are any shapes left.

Make the new shape visible.

Assign `current_color` to the color of the new shape.

Wait 1 second before showing the next shape.

These lines stop the program responding to snaps after the game is over.

This code shows the winner on the screen or declares the game a tie.

Mengonfigurasi item Kanvas

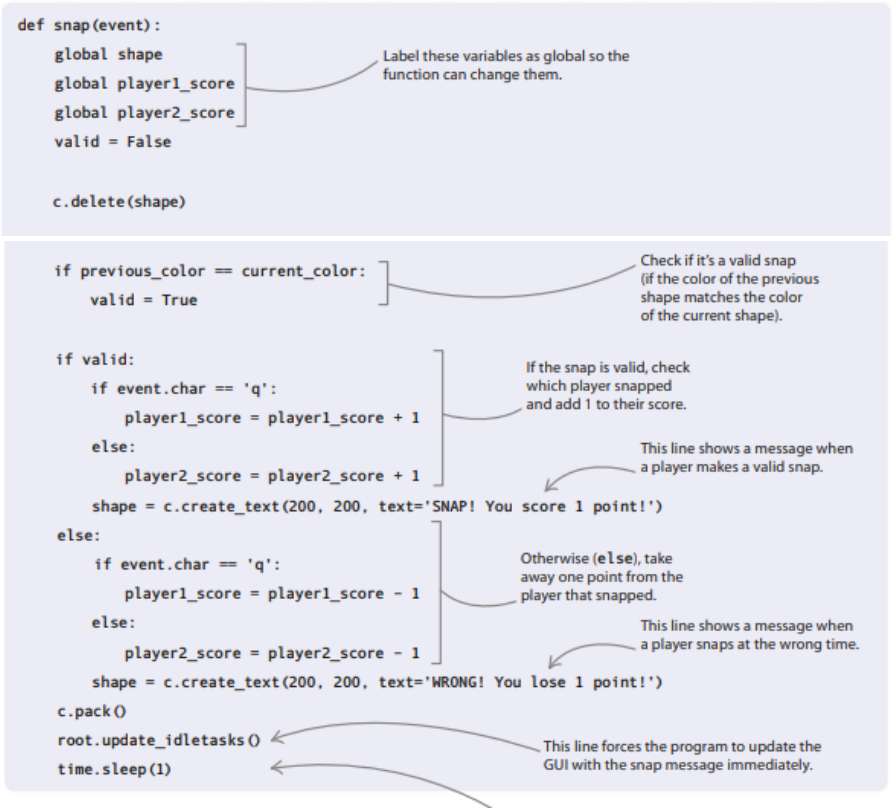
Anda dapat mengubah hal-hal yang muncul di kanvas dengan menggunakan fungsi itemconfigure() Canvas. Dalam game ini, misalnya, Anda menggunakan itemconfigure() untuk mengubah bentuk dari tersembunyi menjadi terlihat, tetapi Anda juga dapat menggunakannya untuk mengubah warna atau karakteristik lainnya. Untuk menggunakan itemconfigure(), masukkan nama item yang ingin Anda ubah dalam tanda kurung, diikuti dengan koma lalu karakteristik dan nilai barunya.

```
c.itemconfigure(shape, state=NORMAL)
```

18. Apakah itu sekejap?

Untuk menyelesaikan permainan, buat fungsi terakhir Anda: snap(). Fungsi ini akan memeriksa pemain mana yang telah menekan tombol mereka dan apakah snap itu

valid (benar). Ini kemudian akan memperbarui skor dan menampilkan pesan. Tambahkan kode ini di bawah fungsi next_shape().



19. Uji kode Anda

    Sekarang jalankan program untuk memeriksanya berfungsi. Ingat Anda perlu mengklik jendela Tkinter sebelum merespons tombol "q" dan "p".

Peretasan dan penyesuaian

Tkinter dapat menampilkan banyak warna dan bentuk yang berbeda selain lingkaran, kotak, dan persegi panjang, jadi ada banyak ruang untuk menyesuaikan permainan Anda. Berikut adalah beberapa ide untuk dicoba—termasuk membuat game ini anti-cheat!

Mempercepat permainan

Anda dapat membuat permainan sedikit lebih sulit dengan mengurangi waktu tunda antara setiap bentuk saat permainan berlangsung. Petunjuk: coba simpan waktu dalam variabel, mulai dari 1000 dan kurangi 25 darinya setiap kali bentuk ditampilkan. Angka-angka ini hanyalah saran—bereksperimenlah dengan angka-angka tersebut untuk melihat apa yang menurut Anda paling berhasil.

Garis besar berwarna

Program melihat parameter isian, bukan garis besar, ketika menilai apakah snap yang valid telah dibuat. Anda dapat memberikan garis luar berwarna yang berbeda pada bentuk dan mereka akan tetap membuat gambar selama warna isiannya cocok.

Tambahkan lebih banyak warna

Anda mungkin telah memperhatikan bahwa Snap adalah permainan yang cukup singkat. Untuk membuatnya lebih panjang, tambahkan kotak ekstra, persegi panjang, dan lingkaran menggunakan warna berbeda.

5.4 BUAT BENTUK BARU

Anda dapat mengubah parameter create_oval() untuk menghasilkan oval daripada lingkaran. Tkinter juga dapat menggambar busur, garis, dan poligon. Cobalah contoh yang ditampilkan di sini, dan mainkan parameternya. Ingatlah untuk menjaga status sebagai TERSEMBUNYI untuk menyembunyikan bentuk sampai saatnya untuk menunjukkannya.

1. Menggambar busur

   Gunakan fungsi create_arc() untuk menggambar busur. Busur padat digambar kecuali Anda memberi gaya pada busur Anda. Untuk menggunakan gaya busur Tkinter yang berbeda, impor CHORD dan ARC dengan mengubah baris ketiga program Anda, seperti yang ditunjukkan di bawah ini. Kemudian tambahkan beberapa akord dan busur ke daftar bentuk Anda, seperti yang ditunjukkan di atas.

```
from tkinter import Tk, Canvas, HIDDEN, NORMAL, CHORD, ARC
```

```
arc = c.create_arc(-235, 120, 365, 370, outline='black', \
                   fill='black', state=HIDDEN)
```

```
arc = c.create_arc(-235, 120, 365, 370, outline='red', \
                   fill='red', state=HIDDEN, style=CHORD)
```

```
arc = c.create_arc(-235, 120, 365, 370, outline='green', \
                   fill='green', state=HIDDEN, style=ARC)
```

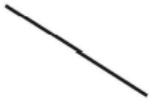
2. Menggambar garis

   Sekarang coba tambahkan beberapa baris ke daftar bentuk Anda menggunakan fungsi create_line().

```
line = c.create_line(35, 200, 365, 200, fill='blue', state=HIDDEN)
```



```
line = c.create_line(35, 20, 365, 350, fill='black', state=HIDDEN)
```



3. Menggambar poligon

   Selanjutnya coba buat beberapa poligon untuk koleksi bentuk Anda, menggunakan create_polygon(). Anda harus memberikan koordinat untuk setiap sudut poligon Anda.

```
polygon = c.create_polygon(35, 200, 365, 200, 200, 35, \
                           outline='blue', fill='blue', state=HIDDEN)
```

5.5 HENTIKAN KECURANGAN PEMAIN

Saat ini, jika snap valid dan kedua pemain menekan tombol snap mereka secara bersamaan, mereka masing-masing mendapatkan poin. Bahkan, mereka masih dapat

mencetak poin hingga bentuk berikutnya ditampilkan, karena sebelumnya dan saat ini akan tetap sama. Coba peretasan ini untuk menghentikan para pemain agar tidak curang.

1. Go global

   Pertama, Anda perlu mengatakan bahwa before_color adalah variabel global dalam fungsi snap(), karena Anda perlu mengubah nilainya. Tambahkan baris ini di bawah variabel global lainnya.

```
    global previous_color
```

2. Blokir beberapa jepretan

   Selanjutnya tambahkan baris berikut ke fungsi snap() untuk mengatur nilai prior_color ke string kosong ('') setelah snap yang benar. Sekarang jika seorang pemain menekan tombol mereka lagi sebelum bentuk berikutnya ditampilkan, mereka akan kehilangan satu poin. Ini karena '' tidak akan pernah sama dengan warna saat ini, kecuali sebelum bentuk pertama ditampilkan.

```
        shape = c.create_text(200, 200, text='SNAP! You scored 1 point!'
    previous_color = ''
```

3. Cegah jepretan dini

   Karena warna_sebelumnya dan warna_saat ini sama di awal permainan, pemain masih bisa curang dengan menekan tombol mereka sebelum bentuk pertama muncul. Untuk mengatasi ini, atur dua variabel ke string yang berbeda di awal. Ubah nilainya menjadi "a" dan "b".

```
previous_color = 'a'
current_color = 'b'
```

4. Ubah pesan

   Jika kedua pemain menekan tombol mereka pada waktu yang hampir bersamaan, mungkin akan membingungkan siapa yang mencetak atau kehilangan poin. Untuk memperbaikinya, Anda dapat mengubah pesan yang ditampilkan saat pemain mencoba melakukan snap.

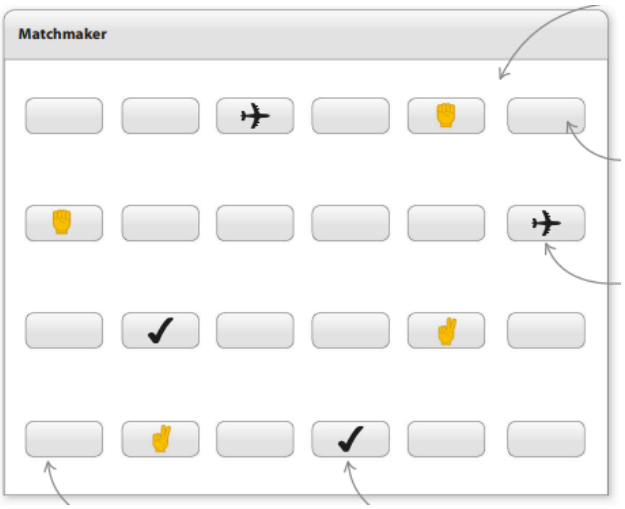
```
if valid:
    if event.char == 'q':
        player1_score = player1_score + 1
        shape = c.create_text(200, 200, text='SNAP! Player 1 scores 1 point!')
    else:
        player2_score = player2_score + 1
        shape = c.create_text(200, 200, text='SNAP! Player 2 scores 1 point!')
    previous_color = ''
else:
    if event.char == 'q':
        player1_score = player1_score - 1
        shape = c.create_text(200, 200, text='WRONG! Player 1 loses 1 point!')
    else:
        player2_score = player2_score - 1
        shape = c.create_text(200, 200, text='WRONG! Player 2 loses 1 point!')
```

5.6 PENCARI JODOH

Seberapa baik ingatanmu? Uji dalam permainan menyenangkan ini di mana Anda harus menemukan pasangan simbol yang cocok. Lihat seberapa cepat Anda dapat menemukan semua 12 pasangan yang cocok!

Apa yang terjadi

Saat Anda menjalankan program, itu akan membuka jendela yang menunjukkan kisi-kisi tombol. Klik pada mereka berpasangan untuk mengungkapkan simbol tersembunyi. Jika dua simbol sama, Anda telah menemukan kecocokan dan simbol tetap terlihat di layar. Jika tidak, kedua tombol akan diatur ulang. Cobalah untuk mengingat lokasi setiap simbol tersembunyi untuk menemukan semua pasangan dengan cepat.



Gambar 5.7 antarmuka pengguna grafis (GUI) dibuat oleh modul Tkinter Python

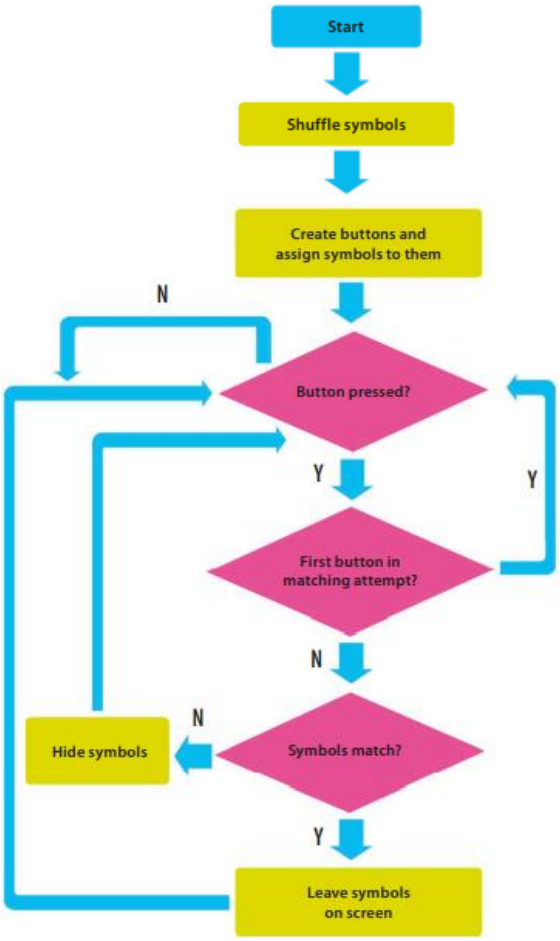
- Kotak menunjukkan 24 tombol yang disusun menjadi empat baris enam.
- Klik pada tombol untuk mengungkapkan simbol.
- Hanya ada dua dari setiap simbol.
- Jika Anda membuat kecocokan yang salah, simbol disembunyikan lagi.
- Simbol yang cocok dibiarkan ditampilkan di kisi.

Bagaimana itu bekerja

Proyek ini menggunakan modul Tkinter untuk menampilkan kisi tombol. Fungsi mainloop() Tkinter mendengarkan penekanan tombol dan menanganinya dengan jenis fungsi khusus, yang disebut fungsi lambda, yang mengungkapkan simbol. Jika simbol yang tidak cocok telah terungkap, program akan memeriksa untuk melihat apakah simbol kedua cocok. Proyek menyimpan tombol dalam kamus dan simbol dalam daftar.

Bagan alur mak comblang

Setelah mengacak simbol dan membuat kisi, program menghabiskan waktunya mendengarkan penekanan tombol. Itu berakhir ketika semua pasangan yang cocok telah ditemukan.



Gambar 5.8 Diagram alur Pencari jodoh

Fungsi Lambda

Seperti def, kata kunci lambda digunakan untuk mendefinisikan fungsi. Semua fungsi Lambda ditulis dalam satu baris dan dapat digunakan di mana pun Anda membutuhkan suatu fungsi. Misalnya, fungsi lambda x: x*2 menggandakan angka. Anda dapat menetapkannya ke variabel, seperti double = lambda x: x*2. Kemudian Anda menyebutnya menggunakan double(x), di mana x adalah angka.

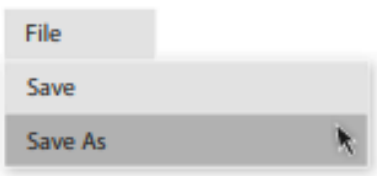
Jadi double(2) akan mengembalikan 4. Fungsi Lambda sangat berguna dalam pemrograman GUI, di mana beberapa tombol mungkin perlu memanggil fungsi yang sama menggunakan parameter yang berbeda. Tanpa fungsi lambda di Matchmaker, Anda harus membuat fungsi yang berbeda untuk setiap tombol— yaitu 24 fungsi!

Mulai

Di bagian pertama proyek, Anda akan menyiapkan antarmuka pengguna grafis (GUI) dan menambahkan pasangan simbol yang akan disembunyikan oleh tombol.

1. Buat file baru

   Buka IDLE. Buat file baru dan simpan sebagai "matchmaker.py".



2. Tambahkan modul

   Sekarang ketik kode ini di bagian atas file Anda untuk mengimpor modul yang Anda butuhkan untuk proyek ini. Anda akan menggunakan acak untuk mengacak simbol, waktu untuk menjeda program, dan Tkinter untuk membuat GUI.

```
import random
import time
from tkinter import Tk, Button, DISABLED
```

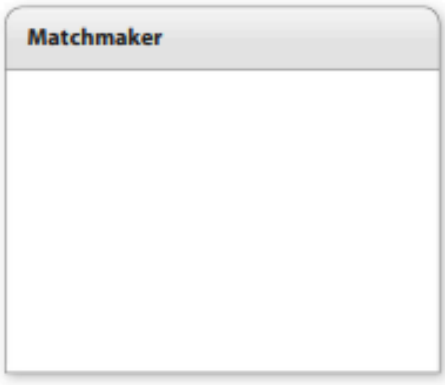
3. Siapkan GUI

   Di bawah perintah impor, tambahkan kode ini, yang akan mengatur GUI. Fungsi root.resizable() mencegah pemutar mengubah ukuran jendela. Ini penting, karena mengubah ukuran jendela akan mengacaukan tata letak tombol yang akan Anda buat nanti.

```
root = Tk()
root.title('Matchmaker')
root.resizable(width=False, height=False)
```

4. Uji kode Anda

   Sekarang jalankan kodenya. Anda akan melihat jendela Tkinter kosong dengan judul "Matchmaker". Jika Anda tidak dapat melihatnya, mungkin tersembunyi di balik jendela lain.



Gambar 5.9 Jendela Matchmaker

5. Buat beberapa variabel

   Di bawah kode untuk Langkah 3, tambahkan variabel yang dibutuhkan program, dan buat kamus untuk menyimpan tombol. Untuk setiap percobaan pertandingan, Anda perlu mengingat apakah itu simbol pertama atau kedua dalam pertandingan. Anda juga perlu melacak penekanan tombol pertama sehingga Anda dapat membandingkannya dengan penekanan tombol kedua.

```
root.resizable(width=False, height=False)

buttons = {}      # This is the dictionary.
first = True      # This variable is used to check if the symbol is the first in the match.
previousX = 0     # These two variables keep track
previousY = 0     # of the last button pressed.
```

6. Tambahkan simbol

   Selanjutnya ketik kode di bawah ini untuk menambahkan simbol yang akan digunakan game. Seperti dalam proyek Nine Lives, program ini menggunakan karakter Unicode. Ada 12 pasang, jadi totalnya 24. Tambahkan kode ini di bawah variabel yang ditambahkan pada Langkah 5.

```
previousY = 0
                    The symbol for each button
                    is stored in this dictionary.
button_symbols = {}
symbols = [u'\u2702', u'\u2702', u'\u2705', u'\u2705', u'\u2708', u'\u2708',
           u'\u2709', u'\u2709', u'\u270A', u'\u270A', u'\u270B', u'\u270B',
           u'\u270C', u'\u270C', u'\u270F', u'\u270F', u'\u2712', u'\u2712',
           u'\u2714', u'\u2714', u'\u2716', u'\u2716', u'\u2728', u'\u2728']
```

Gambar 5.10 Simbol pada python

7. Acak simbol-simbolnya

   Anda tidak ingin simbol muncul di tempat yang sama setiap saat. Setelah beberapa pertandingan, pemain akan mengingat posisi mereka dan akan dapat mencocokkan semuanya pada percobaan pertama mereka, setiap saat. Untuk mencegah hal ini, Anda perlu mengocok simbol sebelum setiap permainan dimulai. Tambahkan baris ini setelah daftar simbol.

```
random.shuffle(symbols)
```

   Bawa tombolnya!

   Pada tahap selanjutnya Anda akan membuat tombol dan menambahkannya ke GUI. Kemudian Anda akan membuat fungsi yang disebut show_symbol() untuk mengontrol apa yang terjadi saat pemain mengklik tombol.

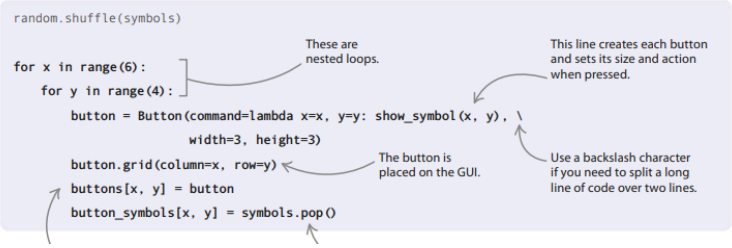
8. Membangun jaringan

   Grid akan terdiri dari 24 tombol yang disusun menjadi empat baris enam. Untuk menata grid, Anda akan menggunakan loop bersarang. Lingkaran x luar akan bekerja dari kiri ke kanan melintasi enam kolom, sedangkan lingkaran y dalam akan bekerja dari atas ke bawah ke bawah setiap kolom. Setelah loop berjalan, setiap tombol akan diberikan sepasang koordinat x dan y yang mengatur posisinya di grid. Letakkan blok kode ini setelah perintah shuffle.

   Tombol

   Tkinter memiliki widget built-in yang disebut Button, yang kami gunakan untuk membuat tombol GUI. Anda dapat melewati parameter yang berbeda untuk itu. Yang kita butuhkan adalah perintah, lebar, dan tinggi. Parameter perintah memberitahu program apa yang harus dilakukan ketika sebuah tombol ditekan. Ini adalah panggilan

fungsi. Dalam program kami, ini memanggil fungsi lambda. Parameter lebar dan tinggi digunakan untuk mengatur ukuran tombol.



Cara kerjanya

Setiap kali loop berjalan, fungsi lambda menyimpan nilai x dan y tombol saat ini (baris dan kolom tempatnya). Saat tombol ditekan, ia memanggil fungsi show_symbol() (yang akan Anda buat nanti) dengan nilai-nilai ini, jadi fungsi tombol mana yang telah ditekan dan simbol mana yang akan ditampilkan.

Loop bersarang

Anda mungkin ingat pernah membaca tentang loop bersarang di halaman 35. Anda dapat memasukkan loop sebanyak yang Anda inginkan. Dalam proyek ini, loop luar berjalan enam kali. Setiap kali putaran luar berjalan, putaran dalam berjalan empat kali. Jadi total loop dalam berjalan 6 x 4 = 24 kali.

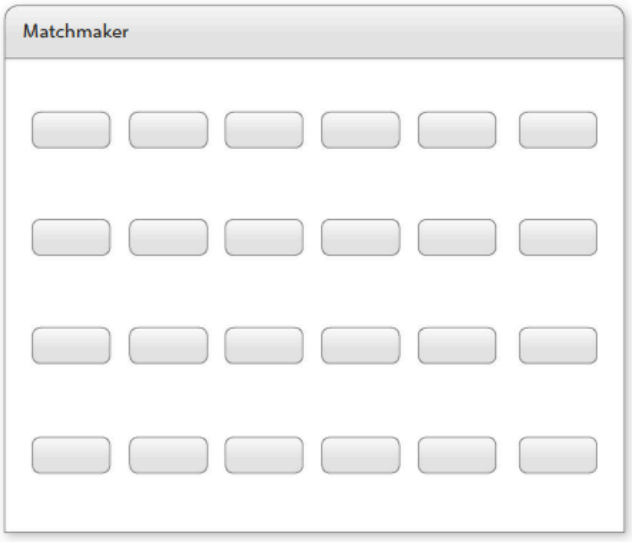
9. Mulai putaran utama

   Sekarang mulai mainloop Tkinter. Setelah loop ini dimulai, GUI akan ditampilkan dan akan mulai mendengarkan penekanan tombol. Ketik baris ini setelah kode yang Anda tambahkan di Langkah 8.

```
        button_symbols[x, y] = symbols.pop()

root.mainloop()
```

10. Uji kode Anda

    Jalankan program lagi. Jendela Tkinter Anda sekarang harus diisi dengan 24 tombol yang diatur dalam kotak. Jika tidak terlihat mirip dengan gambar yang ditampilkan di sini, periksa kode Anda dengan cermat untuk menemukan kesalahan.



Gambar 5.11 Jendela Permainan Matchmaker

11. Tunjukkan simbolnya

Terakhir, Anda perlu membuat fungsi yang menangani penekanan tombol. Fungsi ini akan selalu menampilkan simbol, tetapi cara kerjanya bergantung pada apakah itu giliran pertama atau kedua dalam upaya pencocokan. Jika sudah belokan pertama, fungsinya hanya perlu mengingat tombol mana yang ditekan. Jika giliran kedua, perlu memeriksa apakah simbolnya cocok. Simbol yang tidak cocok disembunyikan. Simbol yang cocok dibiarkan ditampilkan dan tombolnya dinonaktifkan.

```
from tkinter import Tk, Button, DISABLED

def show_symbol(x, y):
    global first
    global previousX, previousY
    buttons[x, y]['text'] = button_symbols[x, y]
    buttons[x, y].update_idletasks()

    if first:
        previousX = x
        previousY = y
        first = False
    elif previousX != x or previousY != y:
        if buttons[previousX, previousY]['text'] != buttons[x, y]['text']:
            time.sleep(0.5)
            buttons[previousX, previousY]['text'] = ''
            buttons[x, y]['text'] = ''
        else:
            buttons[previousX, previousY]['command'] = DISABLED
            buttons[x, y]['command'] = DISABLED
        first = True
```

The x and y values tell the function which button has been pressed.

These lines tell the program that the variables are global.

These lines show the symbol.

If it's the first turn, the code remembers the button press by storing the x and y coordinates.

Second turn. This line includes a check to stop the player cheating by pressing every button twice!

If the symbols don't match...

If the symbols match...

Disable the pair of matching buttons so the player can't press them again.

Bagaimana itu bekerja

Fungsi menunjukkan simbol tombol dengan mengubah label teksnya menjadi karakter Unicode yang kami tetapkan secara acak. Kami menggunakan update_idletasks() untuk memberi tahu Tkinter agar menunjukkan simbol ini sekarang. Jika giliran pertama, kita simpan saja koordinat tombol dalam variabel. Jika giliran kedua, kita perlu memeriksa bahwa pemain tidak mencoba menipu dengan menekan tombol yang sama dua kali. Jika tidak, kami memeriksa apakah simbolnya cocok. Jika simbol tidak cocok, kami

menyembunyikannya dengan menyetel teks ke string kosong; jika cocok, kami membiarkannya ditampilkan tetapi menonaktifkan tombol.

Peretasan dan penyesuaian

Anda bisa mengadaptasi game ini dengan banyak cara. Anda dapat menunjukkan jumlah gerakan yang dilakukan untuk menyelesaikan permainan, sehingga pemain dapat mencoba dan mengalahkan skor mereka sendiri atau menantang teman mereka. Anda juga bisa menambahkan lebih banyak simbol untuk membuat permainan lebih sulit.

Tunjukkan jumlah gerakan

Saat ini, pemain tidak memiliki cara untuk mengetahui seberapa baik yang telah mereka lakukan atau apakah mereka telah melakukan lebih baik daripada teman-teman mereka. Bagaimana kita bisa membuat permainan lebih kompetitif? Mari tambahkan variabel untuk menghitung berapa banyak giliran yang dibutuhkan pemain untuk menyelesaikan permainan. Kemudian pemain bisa bersaing untuk melihat siapa yang mendapat skor terendah.

1. Tambahkan modul baru

   Anda perlu mengimpor widget kotak pesan Tkinter untuk menampilkan jumlah gerakan di akhir permainan. Di baris impor, tambahkan kotak pesan kata setelah DINONAKTIFKAN.

```
from tkinter import Tk, Button, DISABLED, messagebox
```

2. Buat variabel baru

   Anda harus membuat dua variabel tambahan untuk peretasan ini. Satu variabel akan melacak jumlah gerakan yang dilakukan pemain, sementara yang lain akan mengingat berapa banyak pasangan yang mereka temukan. Beri keduanya nilai awal 0. Letakkan baris ini di bawah variabel sebelumnyaY.

```
previousY = 0
moves = 0
pairs = 0
```

3. Nyatakan mereka secara global

   Variabel move dan pair adalah variabel global, dan mereka harus diubah oleh fungsi show_symbol(). Biarkan show_symbol() mengetahui hal ini dengan meletakkan dua baris ini di dekat bagian atas fungsi.

```
def show_symbol(x, y):
    global first
    global previousX, previousY
    global moves
    global pairs
```

4. Hitung gerakannya

Sebuah langkah adalah dua penekanan tombol (satu upaya yang cocok). Jadi Anda hanya perlu menambahkan 1 ke variabel move saat fungsi show_symbol() dipanggil untuk penekanan tombol pertama atau kedua—bukan untuk keduanya. Mari kita lakukan untuk penekanan tombol pertama. Ubah fungsi show_symbol() menjadi seperti ini.

```
if first:
    previousX = x
    previousY = y
    first = False
    moves = moves + 1
```

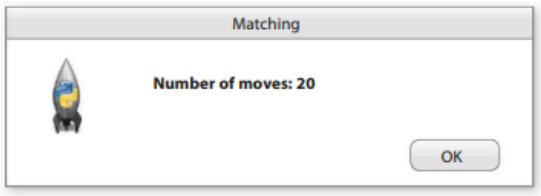
5. Menampilkan pesan

   Sekarang tambahkan kode berikut di dekat bagian bawah fungsi show_symbol(). Ini akan melacak pasangan yang cocok dan menunjukkan kotak pesan di akhir permainan yang memberi tahu pemain berapa banyak gerakan yang mereka ambil. Saat pemain mengklik tombol OK kotak, kode memanggil fungsi close_window(), yang akan kita tambahkan selanjutnya.

```
buttons[x, y]['command'] = DISABLED
pairs = pairs + 1
if pairs == len(buttons)/2:
    messagebox.showinfo('Matching', 'Number of moves: ' +
                        str(moves), command=close_window)
```

Add 1 to the number of pairs found.

This line displays a box showing the number of moves.



Gambar 5.12 *Massage box*

Bagaimana itu bekerja

Ada 12 pasang simbol, jadi Anda cukup mengetikkan pasangan == 12 di peretasan. Namun, kode Anda lebih pintar dari ini. Ini menghitung jumlah pasangan dengan menggunakan pasangan == len(tombol)/2. Ini memungkinkan Anda untuk menambahkan lebih banyak tombol ke gim tanpa harus memperbarui sedikit kode ini.

6. Menutup jendela

   Terakhir, Anda perlu membuat fungsi close_window(), untuk membuat program keluar dari game saat pemain mengklik tombol OK pada kotak pesan “Jumlah gerakan”. Tambahkan kode ini di bawah baris yang mengimpor modul.

```
def close_window(self):
    root.destroy()
```

Tambahkan lebih banyak tombol

Mari kita tantang memori pemain dengan menambahkan lebih banyak tombol dan simbol ke dalam *game.*

1. Simbol ekstra

   Pertama, Anda perlu menambahkan lebih banyak pasangan ke daftar simbol. Sertakan baris baru ini dalam kode.


```
symbols = [u'\u2702', u'\u2702', u'\u2705', u'\u2705', u'\u2708', u'\u2708',
           u'\u2709', u'\u2709', u'\u270A', u'\u270A', u'\u270B', u'\u270B',
           u'\u270C', u'\u270C', u'\u270F', u'\u270F', u'\u2712', u'\u2712',
           u'\u2714', u'\u2714', u'\u2716', u'\u2716', u'\u2728', u'\u2728',
           u'\u2733', u'\u2733', u'\u2734', u'\u2734', u'\u2744', u'\u2744']
```

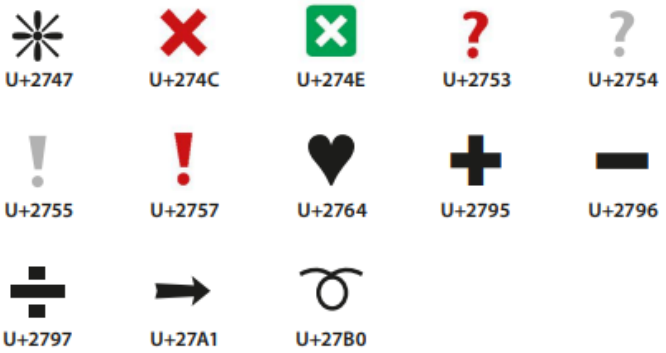
2. Tombol ekstra

   Sekarang tambahkan baris tombol tambahan. Untuk melakukan ini, Anda hanya perlu mengubah rentang y di loop bersarang dari 4 menjadi 5, seperti yang ditunjukkan di sebelah kanan.

```
for x in range(6):
    for y in range(5):
```

3. Bahkan lebih besar?

   Anda sekarang memiliki total 30 tombol. Jika Anda ingin menambahkan lebih banyak, pastikan jumlah tombol tambahan yang Anda tambahkan adalah kelipatan 6 sehingga Anda selalu menambahkan baris lengkap. Jika Anda merasa ingin bertualang, Anda dapat bereksperimen dengan tata letak tombol yang berbeda dengan mengubah loop bersarang.



Gambar 5.13 macam-macam symbol

5.7 PENANGKAP TELUR

Game ini akan menguji konsentrasi dan kecepatan refleks Anda. Jangan retak di bawah tekanan—tangkap saja telur sebanyak mungkin untuk mendapatkan skor tinggi. Tantang teman Anda untuk melihat siapa pemenang penangkap telur!

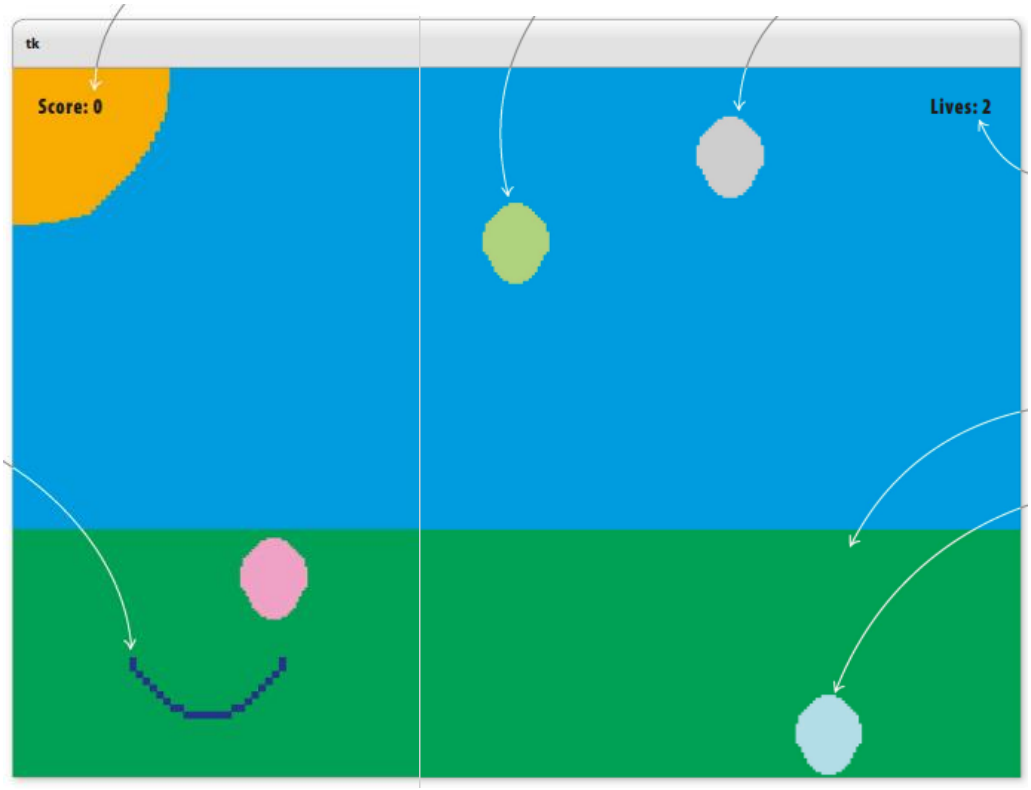
Apa yang terjadi

Pindahkan penangkap di sepanjang bagian bawah layar untuk menangkap setiap telur sebelum menyentuh tanah. Ketika Anda mengambil telur Anda mencetak poin, tetapi jika

Anda menjatuhkan telur Anda kehilangan kehidupan. Hati-hati: semakin banyak telur yang Anda tangkap, semakin sering telur baru muncul di bagian atas layar dan semakin cepat jatuhnya. Kehilangan ketiga nyawa dan permainan berakhir.

Waktu

Waktu aksi di layar itu penting. Pada awalnya, telur baru hanya ditambahkan setiap 4 detik; jika tidak, akan ada terlalu banyak telur. Awalnya, telur bergerak turun sedikit setiap setengah detik. Jika intervalnya lebih kecil, permainan akan terlalu sulit. Program memeriksa tangkapan sekali setiap sepersepuluh detik — lebih lambat, dan mungkin melewatkannya. Saat pemain mencetak lebih banyak poin, kecepatan dan jumlah telur meningkat untuk membuat permainan lebih menantang.



Gambar 5.14 Permainan Penangkap telur

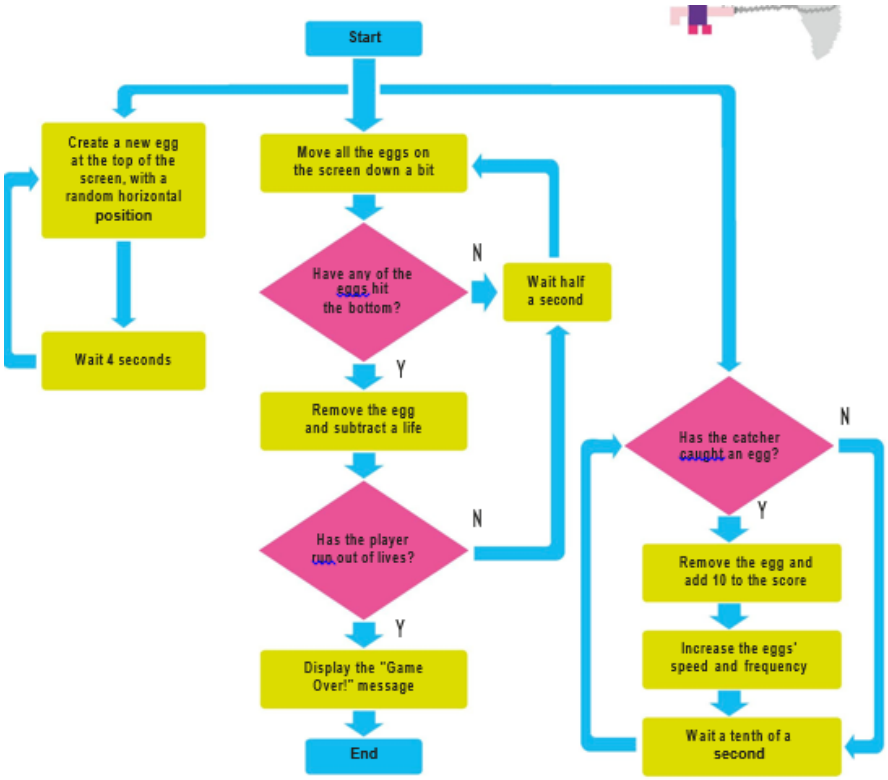
- Anda mencetak 10 poin untuk menangkap setiap telur.
- Gerakkan penangkap ke depan dan ke belakang dengan menekan tombol panah kiri dan kanan.
- Program ini menggunakan Tkinter untuk menggambar dan memindahkan bentuk, dan modul acak untuk menempatkannya di layar.
- Telur baru muncul di bagian atas layar, dalam posisi acak.
- Penghitung ini menunjukkan berapa banyak nyawa yang tersisa.
- Anda dapat menambahkan bentuk statis, seperti rumput, ke layar untuk membuat pemandangan latar belakang.
- Jika telur menyentuh bagian bawah layar, Anda kehilangan nyawa.

Game bergaya arcade

Proyek terakhir ini menyatukan semua keterampilan pengkodean Anda untuk membuat game bergaya arcade yang mengesankan. Kodenya cukup rumit, jadi periksa kode Anda dengan cermat untuk menemukan bug di setiap tahap dan jangan berkecil hati jika Anda membuat beberapa kesalahan di sepanjang prosesnya. Setelah Anda memecahkan *Egg Catcher*, Anda akan siap untuk mulai membuat game Anda sendiri.

Bagaimana itu bekerja

Setelah latar belakang dibuat, telur secara bertahap bergerak ke bawah layar, yang menciptakan ilusi bahwa mereka jatuh. Menggunakan loop, kode terus-menerus memeriksa koordinat telur untuk melihat apakah ada yang menyentuh dasar atau tertangkap di penangkap. Ketika telur ditangkap atau dijatuhkan, telur itu akan dihapus dan program menyesuaikan skor atau jumlah nyawa yang tersisa.



Gambar 5.15 Diagram alur Permainan Penangkap Telur

Bagan alur Penangkap Telur

Ada tiga putaran berbeda dalam permainan ini: satu untuk membuat telur baru, yang lain untuk memeriksa apakah penangkap telah menangkap telur, dan yang ketiga untuk memindahkan telur dan memeriksa apakah telur menyentuh dasar. Masing-masing dari tiga loop berulang dengan kecepatan yang berbeda.

Pengaturan

Pertama, Anda akan mengimpor bagian Python yang Anda butuhkan untuk proyek ini. Kemudian Anda akan mengatur semuanya sehingga Anda siap untuk menulis fungsi utama untuk game tersebut.

1. Buat file

   Buka IDLE dan buat file baru. Simpan sebagai "telur_catcher.py".

2. Impor modul

   Egg Catcher menggunakan tiga modul: itertools untuk menggilir beberapa warna; acak untuk membuat telur muncul di tempat acak; dan Tkinter untuk menganimasikan game dengan membuat bentuk di layar. Ketik baris ini di bagian atas file Anda.

```
from itertools import cycle
from random import randrange
from tkinter import Canvas, Tk, messagebox, font
```

3. Siapkan kanvas

   Tambahkan kode ini di bawah pernyataan impor. Itu membuat variabel untuk tinggi dan lebar kanvas, lalu menggunakannya untuk membuat kanvas itu sendiri. Untuk menambahkan sedikit pemandangan ke permainan Anda, itu menggambar persegi panjang untuk mewakili beberapa rumput dan oval untuk mewakili matahari.

```
from tkinter import Canvas, Tk, messagebox, font

canvas_width = 800
canvas_height = 400

root = Tk()
c = Canvas(root, width=canvas_width, height=canvas_height, \
background='deep sky blue')
c.create_rectangle(-5, canvas_height - 100, canvas_width + 5, \
canvas_height + 5, fill='sea green', width=0)
c.create_oval(-80, -80, 120, 120, fill='orange', width=0)
c.pack()
```

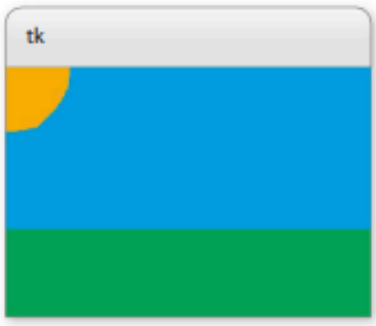
This creates a window.

The canvas will be sky blue and measure 800 x 400 pixels.

Use a backslash character if you need to split a long line of code over two lines.

4. Lihat kanvas Anda

   Jalankan kode untuk melihat tampilan kanvas. Anda akan melihat pemandangan dengan rumput hijau, langit biru, dan matahari yang cerah. Jika Anda merasa percaya diri, cobalah untuk membuat pemandangan sendiri dengan bentuk warna atau ukuran yang berbeda. Anda selalu dapat kembali ke kode di atas jika mengalami masalah.



Gambar 5.16 Tampilan Pada layar

5. Siapkan telur

   Sekarang buat beberapa variabel untuk menyimpan warna, lebar, dan tinggi telur. Anda juga memerlukan variabel untuk skor, kecepatan telur yang jatuh, dan interval antara telur baru yang muncul di layar. Jumlah perubahannya ditentukan oleh faktor-

kesulitan—nilai yang lebih rendah untuk variabel ini sebenarnya membuat permainan lebih sulit.

```
c.pack()

color_cycle = cycle(['light blue', 'light green', 'light pink', 'light yellow', 'light cyan'])
egg_width = 45
egg_height = 55
egg_score = 10
egg_speed = 500
egg_interval = 4000
difficulty_factor = 0.95
```

You score 10 points for catching an egg.

A new egg appears every 4,000 milliseconds (4 seconds).

This is how much the speed and interval change after each catch (closer to 1 is easier).

6. Siapkan penangkap

Selanjutnya tambahkan variabel untuk penangkap. Selain variabel untuk warna dan ukurannya, ada empat variabel yang menyimpan posisi awal penangkap. Nilai untuk ini dihitung menggunakan ukuran kanvas dan penangkap. Setelah ini dihitung, mereka digunakan untuk membuat busur yang digunakan game untuk penangkap.

```
difficulty_factor = 0.95

catcher_color = 'blue'
catcher_width = 100
catcher_height = 100
catcher_start_x = canvas_width / 2 - catcher_width / 2
catcher_start_y = canvas_height - catcher_height - 20
catcher_start_x2 = catcher_start_x + catcher_width
catcher_start_y2 = catcher_start_y + catcher_height

catcher = c.create_arc(catcher_start_x, catcher_start_y, \
                       catcher_start_x2, catcher_start_y2, start=200, extent=140, \
                       style='arc', outline=catcher_color, width=3)
```

This is the height of the circle that is used to draw the arc.

These lines make the catcher start near the bottom of the canvas, in the center of the window.

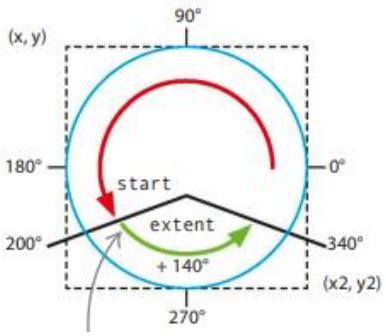
Start drawing at 200 degrees on the circle.

Draw for 140 degrees.

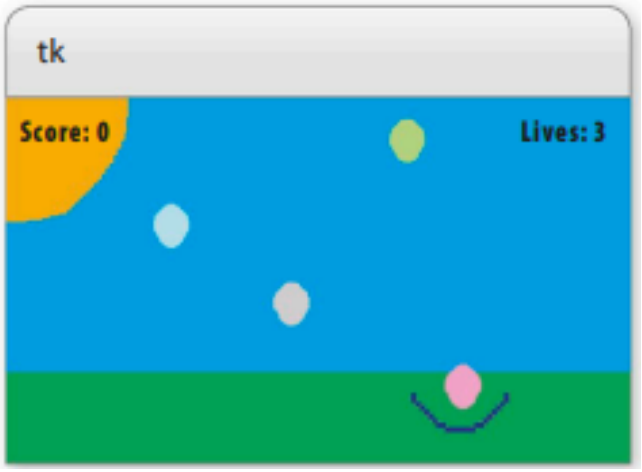
Draw the catcher.

Bagaimana itu bekerja

Anda menggunakan busur untuk mewakili penangkap. Busur adalah salah satu bagian dari seluruh lingkaran. Tkinter menggambar lingkaran di dalam kotak tak terlihat. Dua koordinat catcher_start pertama (x dan y) plot di mana salah satu sudut kotak seharusnya. Dua koordinat kedua (x2 dan y2) menggambarkan posisi sudut kotak yang berlawanan. Fungsi create_arc() memiliki dua parameter, keduanya diberikan dalam derajat (°), yang mengatakan di mana dalam lingkaran untuk menggambar busur: start mengatakan di mana harus mulai menggambar, sedangkan sejauh adalah berapa derajat untuk menggambar sebelum berhenti.



Gambar 5.17 Membuat penangkap telur



Gambar 5.18 Game penangkap Telur

7. Skor dan penghitung hidup

   Tambahkan kode ini di bawah baris yang mengatur penangkap. Ini menetapkan skor awal ke 0 dan membuat teks yang menunjukkan skor di layar. Itu juga mengatur sisa hidup menjadi tiga dan menampilkan nomor ini. Untuk memeriksa apakah kode berfungsi, tambahkan root.mainloop() tepat di akhir dan kemudian jalankan kode. Setelah Anda memeriksanya, hapus baris ini—Anda akan menambahkannya lagi nanti saat dibutuhkan.

```
catcher = c.create_arc(catcher_start_x, catcher_start_y, \
                       catcher_start_x2, catcher_start_y2, start=200, extent=140,
                       style='arc', outline=catcher_color, width=3)

game_font = font.nametofont('TkFixedFont')
game_font.config(size=18)

score = 0
score_text = c.create_text(10, 10, anchor='nw', font=game_font, fill='darkblue', \
                           text='Score: ' + str(score))

lives_remaining = 3
lives_text = c.create_text(canvas_width - 10, 10, anchor='ne', font=game_font, \
                           fill='darkblue', text='Lives ' + str(lives_remaining))
```

This line selects a cool computer-style font.

You can make the text larger or smaller by changing this number.

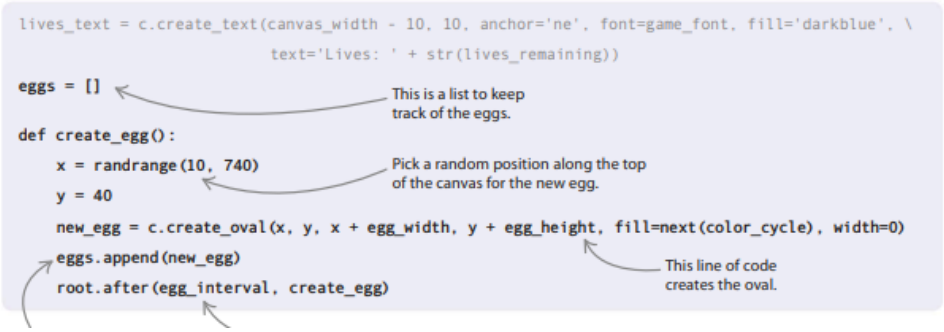
The player gets three lives.

Jatuh, mencetak gol, jatuh

Anda telah menyelesaikan semua tugas penyiapan, jadi saatnya menulis kode yang menjalankan game. Anda akan memerlukan fungsi untuk membuat telur dan membuatnya jatuh, dan beberapa fungsi lagi untuk menangani tangkapan telur dan tetesan telur.

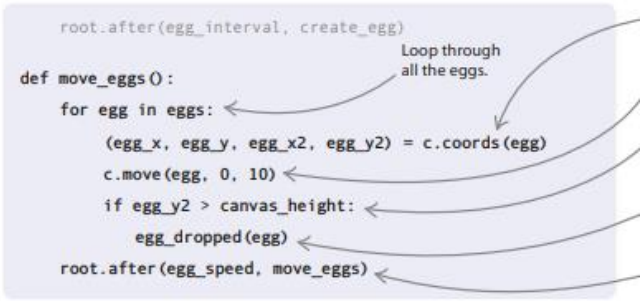
8. Buat telurnya

   Tambahkan kode ini. Daftar melacak semua telur di layar. Fungsi create_egg() menentukan koordinat setiap telur baru (koordinat x selalu dipilih secara acak). Kemudian ia membuat telur sebagai oval dan menambahkannya ke daftar telur. Akhirnya, ini mengatur timer untuk memanggil fungsi lagi setelah jeda.



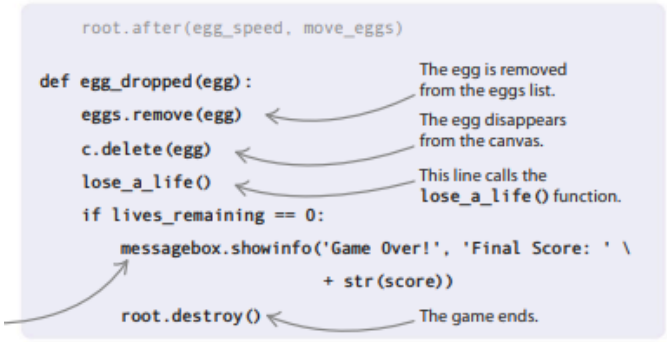
9. Pindahkan telur

   Setelah membuat telur, tambahkan fungsi berikutnya, move_eggs(), untuk menggerakkannya. Ini loop melalui daftar semua telur di layar. Untuk setiap telur, koordinat y meningkat, yang menggerakkan telur ke bawah layar. Setelah telur dipindahkan, program akan memeriksa apakah telur telah mencapai bagian bawah layar. Jika sudah, telur telah dijatuhkan dan fungsi egg_dropped() dipanggil. Terakhir, pengatur waktu diatur untuk memanggil fungsi move_eggs() lagi setelah jeda singkat.



10. Ups—telur jatuh!

    Selanjutnya tambahkan fungsi egg_dropped() setelah move_eggs(). Ketika telur dijatuhkan, itu akan dihapus dari daftar telur dan kemudian dihapus dari kanvas. Nyawa dikurangi menggunakan fungsi lose_a_life(), yang akan Anda buat di Langkah 11. Jika kehilangan nyawa berarti tidak ada nyawa yang tersisa, "Game Over!" pesan ditampilkan.



11. Kehilangan nyawa

    Kehilangan nyawa hanya melibatkan pengurangan nyawa dari variabel life_remaining dan kemudian menampilkan nilai baru di layar. Tambahkan baris ini setelah fungsi egg_dropped().

```
        root.destroy()

def lose_a_life():
    global lives_remaining
    lives_remaining -= 1
    c.itemconfigure(lives_text, text='Lives: ' \
                    + str(lives_remaining))
```

This variable needs to be global, as the function will modify it.

The player loses a life.

12. Periksa tangkapan

    Sekarang tambahkan fungsi check_catch(). Telur ditangkap jika berada di dalam busur penangkap. Untuk mengetahui apakah Anda telah menangkap, loop for mendapatkan koordinat setiap telur dan membandingkannya dengan koordinat penangkap. Jika ada kecocokan, telur ditangkap. Kemudian dihapus dari daftar, dihapus dari layar, dan skor ditingkatkan.

```
    c.itemconfigure(lives_text, text='Lives: ' + str(lives_remaining))

def check_catch():
    (catcher_x, catcher_y, catcher_x2, catcher_y2) = c.coords(catcher)
    for egg in eggs:
        (egg_x, egg_y, egg_x2, egg_y2) = c.coords(egg)
        if catcher_x < egg_x and egg_x2 < catcher_x2 and catcher_y2 - egg_y2 < 40:
            eggs.remove(egg)
            c.delete(egg)
            increase_score(egg_score)
    root.after(100, check_catch)
```

Get the coordinates of the catcher.

Get the coordinates of the eggs.

Is the egg inside the catcher horizontally and vertically?

Increase the score by 10 points.

Call this function again after 100 milliseconds (one-tenth of a second).

13. Tingkatkan skornya

    Pertama skor ditingkatkan dengan nilai parameter poin. Selanjutnya kecepatan dan interval baru telur dihitung dengan mengalikan nilainya dengan faktor kesulitan. Akhirnya, teks di layar diperbarui dengan skor baru. Tambahkan fungsi baru ini di bawah check_catch().

```
    root.after(100, check_catch)

def increase_score(points):
    global score, egg_speed, egg_interval
    score += points
    egg_speed = int(egg_speed * difficulty_factor)
    egg_interval = int(egg_interval * difficulty_factor)
    c.itemconfigure(score_text, text='Score: ' + str(score))
```

Add to the player's score.

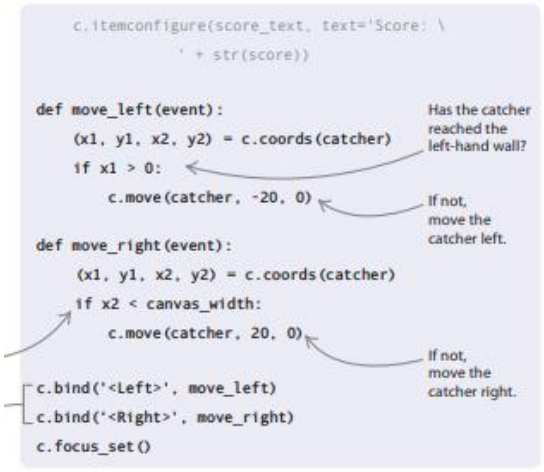
Tangkap telur itu!

Sekarang setelah Anda mendapatkan semua bentuk dan fungsi yang diperlukan untuk permainan, yang tersisa untuk ditambahkan hanyalah kontrol untuk penangkap telur dan perintah yang memulai permainan.

14. Siapkan kontrol

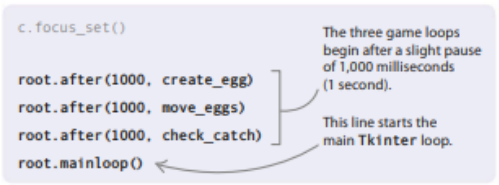
    Fungsi move_left() dan move_right() menggunakan koordinat penangkap untuk memastikan penangkap tidak akan meninggalkan layar. Jika masih ada ruang untuk dipindahkan, penangkap bergeser secara horizontal sebesar 20 piksel. Kedua fungsi ini ditautkan ke tombol panah kiri dan kanan pada keyboard menggunakan fungsi bind().

Fungsi focus_set() memungkinkan program mendeteksi penekanan tombol. Tambahkan fungsi baru di bawah fungsi peningkatan_score().



15. Memulai permainan

Tiga fungsi perulangan dimulai menggunakan timer. Ini memastikan mereka tidak dijalankan sebelum loop utama dimulai. Terakhir, fungsi mainloop() memulai loop Tkinter yang mengelola semua loop dan timer Anda. Semua selesai – nikmati permainannya, dan jangan biarkan telur-telur itu pecah!



Peretasan dan penyesuaian

Untuk membuat permainan terlihat lebih baik, Anda dapat mencoba menambahkan beberapa pemandangan keren Anda sendiri. Suara dan musik yang menyenangkan adalah cara hebat lainnya untuk membuat game lebih seru.

Memasang modul

Beberapa modul Python yang paling berguna— seperti Pygame—tidak disertakan sebagai bagian dari pustaka Python standar. Jika Anda ingin menggunakan salah satu modul lain ini, Anda harus menginstalnya terlebih dahulu. Tempat terbaik untuk mencari petunjuk tentang cara memasang modul adalah situs web modul. Ada petunjuk dan tip di https://docs.python.org/3/installing/.

Atur adegan

Tkinter memungkinkan gambar khusus untuk digunakan sebagai latar belakang untuk kanvas. Jika file Anda adalah GIF, Anda dapat menggunakan tkinter.PhotoImage untuk memuat file. Jika gambar Anda memiliki format yang berbeda, Anda mungkin ingin melihat ke Pillow—modul penanganan gambar yang berguna.

Gambar 5.19 Hasil Permainan penangkap Telur dengan background yang berbeda

Buat beberapa kebisingan

Untuk benar-benar menghidupkan permainan, tambahkan musik latar atau efek suara untuk menangkap telur atau kehilangan nyawa. Modul yang digunakan untuk menambahkan suara adalah pygame.mixer. Ingat, pygame bukan modul Python standar, jadi Anda harus menginstalnya terlebih dahulu. Anda juga harus memiliki salinan file suara yang ingin Anda putar, yang harus Anda tempatkan di folder yang sama dengan file kode Anda. Setelah itu, memutar suara hanya membutuhkan beberapa baris kode.

```
import time

from pygame import mixer

mixer.init()        # Get the mixer ready to play sounds.
beep = mixer.Sound("beep.wav")
beep.play()         # Tell the mixer which sound to play.
time.sleep(5)
```

DAFTAR PUSTAKA

Aldrich, Clark. Learning Online with Games, Simulations, and Virtual Worlds: Strategies for Online Instruction Hershey: IGI Global, 2009.

Bejiga, M. B., Zeggada, A., Nouffidj, A., & Melgani, F. (2017). *A convolutional neural network approach for assisting avalanche search and rescue operations with UAV imagery.* Remote Sensing, 9(2). https://doi.org/10.3390/rs9020100

Cavallaro, Dani. Anime and the Visual Novel: Narrative Structure, Design and Play at the Crossroads of Animation and Computer Games Jefferson: McFarland & Company, 2010.

Dastbaz, M. Designing Interactive Multimedia Systems New York: McGraw-Hill, 2003.

Devikar, P. 2016. Transfer Learning for Image Classification of Various Dog Breeds. *International Journal of Advanced Research in Computer Engineering and Technology* (IJARCET), Vol.5: 2707-2715.

Fahriza Azwar Muhammad, Rizky Arif Windiator,Yuridi Bintang Pratama, "pemrograman socket untuk koneksi Abtara Raspberry Pi dengan Referee Box",*Universitas Islam Indonesia*, 2016.

Fitri, Kiki Reski R, Ady Rahmansyah, dan Wahyuni. *Penggunaan Bahasa Pemrograman Python Sebagai Pusat Kendali Pada Robot 10-D*, 2017.

H. Abhirawa, Jondri, dan A. Arifianto, "Pengenalan wajah menggunakan convolutional neural network," Dalam *e-Proceeding of Engineering*, 2017.

Henry, Samuel. Panduan Praktis Membuat Game 3D Yogyakarta: Graha Ilmu, 2005.

K. P. Danukusumo, "*Implementasi deep learning menggunakan convolutional neural network untuk klasifikasi citra candi berbasis GPU*," Skripsi, Universitas Atma Jaya Yogyakarta, Yogyakarta, 2017.

Kadir, A. 2018. Dasar Logika Pemrograman Komputer. Cetakan Kedua. Elexmedia Komputindo.

Kim, J., Sangjun, O., Kim, Y., & Lee, M. (2016). Convolutional Neural Network with Biologically Inspired Retinal Structure. Procedia Computer Science, 88, 145–154. https://doi.org/10.1016/j.procs.2016.07.418

Krizhevsky, A., Sutskever I., & Hinton G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of NIPS.

Ravi, S. & Nayeem, S. (2013). A Study on Face Recognition Technique based on Eigenface. Foundation of Computer Science FCS, New York, USA Volume 5– No.4. International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868.

S. R. Dewi, “Deep learning object detection pada video menggunakan tensorflow dan convolutional network,” Skripsi, Universitas Islam Indonesia, Yogyakarta, 2018.

Suartika E. P, Arya Yudhi Wijaya Wijaya, dan Rully Soelaiman . “Klasifikasi Citra Menggunakan Convolutional Neural Network (Cnn) pada Caltech 101.” 2016.

Vaughan, Tay. Multimedia: Making it Work 6th ed. San Fransisco: McGraw-Hill, 2004.

Wahyono, Teguh. Fundamental Of Python For Mechibe Learning. Yogyakarta: Gava Media, 2018.

Yuliza, IncomTech, Jurnal Telekomunikasi dan Komputer, vol.4, no.1,2013.

Zhi, T., Duan, L. Y., Wang, Y., & Huang, T. (2016). Two- stage pooling of deep convolutional features for image retrieval. In 2016 IEEE International Conference on Image Processing (ICIP) (hal. 2465– 2469). https://doi.org/10.1109/ICIP.2016.7532802

Zufar, M. & Setiyono B. (2016). Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time. JURNAL SAINS DAN SENI ITS Vol. 5 No. 2 . A-72.