Un Problema menos Con Python

Christian Gimenez

18 Sep 2019

Outline

Outline

Outline

¿Qué hace falta?

¿Qué hace falta?

- Python 3.2
- Biblioteca tkinter (viene por defecto)
- Geany

Dos versiones

Python 2.x entrará en desuso en el próximo año.
Usaremos Python 3.x

¿Qué hace falta?

¿Qué hace falta?

- Python 3.2
- Biblioteca tkinter (viene por defecto)
- Geany

Dos versiones

Python 2.x entrará en desuso en el próximo año.
Usaremos Python 3.x

¿Qué hace falta?

¿Qué hace falta?

- Python 3.2
- Biblioteca tkinter (viene por defecto)
- Geany

Dos versiones

Python 2.x entrará en desuso en el próximo año.
Usaremos Python 3.x

¿Qué hace falta?

¿Qué hace falta?

- Python 3.2
- Biblioteca tkinter (viene por defecto)
- Geany

Dos versiones

Python 2.x entrará en desuso en el próximo año.
Usaremos Python 3.x

¿Qué hace falta?

¿Qué hace falta?

- Python 3.2
- Biblioteca tkinter (viene por defecto)
- Geany

Dos versiones

Python 2.x entrará en desuso en el próximo año.
Usaremos Python 3.x

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación
  - Determina el cuerpo
  - Se usan espacios (4 es lo ideal)
  - Nunca usar caracteres de tabulación
  - Ante la duda, escribirlos a mano

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación
  - Determina el cuerpo
  - Se usan espacios (4 es lo ideal)
  - Nunca usar caracteres de tabulación
  - Ante la duda, escribirlos a mano

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación
  - Determina el cuerpo
  - Se usan espacios (4 es lo ideal)
  - Nunca usar caracteres de tabulación
  - Ante la duda, escribirlos a mano

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación
  - Determina el cuerpo
  - Se usan espacios (4 es lo ideal)
  - Nunca usar caracteres de tabulación
  - Ante la duda, escribirlos a mano

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación
  - Determina el cuerpo
  - Se usan espacios (4 es lo ideal)
  - Nunca usar caracteres de tabulación
  - Ante la duda, escribirlos a mano

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación
  - Determina el cuerpo
  - Se usan espacios (4 es lo ideal)
  - Nunca usar caracteres de tabulación
  - Ante la duda, escribirlos a mano

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Tener en cuenta al programar

¿Por qué Python?

- Pensado para leer mejor
- Pensado para que sea dinámico
- Pensado para que sea lo más rápido posible

¿Qué se debe tener en cuenta?

- Ojo con la indentación
  - Determina el cuerpo
  - Se usan espacios (4 es lo ideal)
  - Nunca usar caracteres de tabulación
  - Ante la duda, escribirlos a mano

Zen de Python

Python tiene su propia *Filosofía de Python*.
Probar: `import this`

Outline

Outline

Un Pequeño Problema

Se acerca el día de la primavera y un grupo de estudiantes querían juntarse a comer pizas. Normalmente, sacar cuentas para poder deducir cuánto tiene que pagar cada uno sería fácil, pero querían que participen a los primeros 20 que lleguen al aula (se suponen que son los más aplicados).

Entonces...

- Van a dejar una computadora a la entrada del curso.
- Aquel que quiere participar (y si hay cupo), debe escribir su nombre y registrarse.

Un Pequeño Problema

Se acerca el día de la primavera y un grupo de estudiantes querían juntarse a comer pizas. Normalmente, sacar cuentas para poder deducir cuánto tiene que pagar cada uno sería fácil, pero querían que participen a los primeros 20 que lleguen al aula (se suponen que son los más aplicados).

Entonces...

- Van a dejar una computadora a la entrada del curso.
- Aquel que quiere participar (y si hay cupo), debe escribir su nombre y registrarse.

¿Qué debe hacer el programa?

- Preguntar el nombre
- Calcular para 20
- Decir cuánto le sale

¿Qué aprenderemos?

- Entrada y Salida
- Variables
- Operadores matemáticos

¿Qué debe hacer el programa?

- Preguntar el nombre
- Calcular para 20
- Decir cuánto le sale

¿Qué aprenderemos?

- Entrada y Salida
- Variables
- Operadores matemáticos

¿Qué debe hacer el programa?

- Preguntar el nombre
- Calcular para 20
- Decir cuánto le sale

¿Qué aprenderemos?

- Entrada y Salida
- Variables
- Operadores matemáticos

¿Qué debe hacer el programa?

- Preguntar el nombre
- Calcular para 20
- Decir cuánto le sale

¿Qué aprenderemos?

- Entrada y Salida
- Variables
- Operadores matemáticos

¿Qué debe hacer el programa?

- Preguntar el nombre
- Calcular para 20
- Decir cuánto le sale

¿Qué aprenderemos?

- Entrada y Salida
- Variables
- Operadores matemáticos

¿Qué debe hacer el programa?

- Preguntar el nombre
- Calcular para 20
- Decir cuánto le sale

¿Qué aprenderemos?

- Entrada y Salida
- Variables
- Operadores matemáticos

¿Qué debe hacer el programa?

- Preguntar el nombre
- Calcular para 20
- Decir cuánto le sale

¿Qué aprenderemos?

- Entrada y Salida
- Variables
- Operadores matemáticos

Primero, saludemos

Pedimos el nombre

- Importamos lo que necesitamos
  - tkinter
  - `askstring` con otro nombre: `pedir_dato`
- Pedimos el nombre

Primero, saludemos

Pedimos el nombre

- Importamos lo que necesitamos
  - tkinter
  - `askstring` con otro nombre: `pedir_dato`
- Pedimos el nombre

Primero, saludemos

Pedimos el nombre

- Importamos lo que necesitamos
  - tkinter
  - `askstring` con otro nombre: `pedir_dato`
- Pedimos el nombre

Primero, saludemos

Pedimos el nombre

- Importamos lo que necesitamos
  - tkinter
  - `askstring` con otro nombre: `pedir_dato`
- Pedimos el nombre

Pedimos el nombre

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato
root = Tk()
root.withdraw()

nombre = pedir_dato('¡Pizzas!', 'Indique su nombre y apellido')
```

¿Por qué root. . . ?

`root` es la ventana principal. Con `withdraw()` la ocultamos.
Toda aplicación de Tk necesita de su intérprete y una ventana principal.

¿Por qué from . . . import?

`from ... import` nos permite importar de una biblioteca alguna función o clase.
Elementos que se definen en otros archivos y que se pueden descargar e instalar.

Pedimos el nombre

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato
root = Tk()
root.withdraw()

nombre = pedir_dato('¡Pizzas!', 'Indique su nombre y apellido')
```

¿Por qué root...?

`root` es la ventana principal. Con `withdraw()` la ocultamos.
Toda aplicación de Tk necesita de su intérprete y una ventana principal.

¿Por qué from ... import?

`from ... import` nos permite importar de una biblioteca alguna función o clase.
Elementos que se definen en otros archivos y que se pueden descargar e instalar.

Pedimos el nombre

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato
root = Tk()
root.withdraw()

nombre = pedir_dato('¡Pizzas!', 'Indique su nombre y apellido')
```

¿Por qué root...?

`root` es la ventana principal. Con `withdraw()` la ocultamos.
Toda aplicación de Tk necesita de su intérprete y una ventana principal.

¿Por qué from ... import?

`from ... import` nos permite importar de una biblioteca alguna función o clase.
Elementos que se definen en otros archivos y que se pueden descargar e instalar.

¿y si saludamos primero?

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato

root = Tk()
root.withdraw()

nombre = pedir_dato('¡Pizzas!', 'Indique su nombre y apellido')
```

¿Y si primero saludamos?

- Intente agregar un saludo personalizado, algo como "Hola Pepe"
  - a. Pruebe con "Hola " + nombre
- Se puede agregar una breve explicación del programa.

¿y si saludamos primero?

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato

root = Tk()
root.withdraw()

nombre = pedir_dato('¡Pizzas!', 'Indique su nombre y apellido')
```

¿Y si primero saludamos?

- Intente agregar un saludo personalizado, algo como "Hola Pepe"
  - Pruebe con `'Hola ' + nombre`.
- Se puede agregar una breve explicación del programa.

¿y si saludamos primero?

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato

root = Tk()
root.withdraw()

nombre = pedir_dato('¡Pizzas!', 'Indique su nombre y apellido')
```

¿Y si primero saludamos?

- Intente agregar un saludo personalizado, algo como "Hola Pepe"
  - Pruebe con `'Hola ' + nombre`.
- Se puede agregar una breve explicación del programa.

¿y si saludamos primero?

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato

root = Tk()
root.withdraw()

nombre = pedir_dato('¡Pizzas!', 'Indique su nombre y apellido')
```

¿Y si primero saludamos?

- Intente agregar un saludo personalizado, algo como "Hola Pepe"
  - Pruebe con `'Hola ' + nombre`.
- Se puede agregar una breve explicación del programa.

Calculamos el promedio

Supongamos que las pizzas nos salen $500 y el cupo es para 20 personas.

```
plata = 500
cant_personas = 20
cada_uno = plata / cant_personas
```

¿Y si se agregan las bebidas?

Cada estudiantes debe pagar su suma más un plus por las bebidas.
¿Qué se debe cambiar?

Calculamos el promedio

Supongamos que las pizzas nos salen $500 y el cupo es para 20 personas.

```
plata = 500
cant_personas = 20
cada_uno = plata / cant_personas
```

¿Y si se agregan las bebidas?

Cada estudiantes debe pagar su suma más un plus por las bebidas.
¿Qué se debe cambiar?

Calculamos el promedio

Supongamos que las pizzas nos salen $500 y el cupo es para 20 personas.

```
plata = 500
cant_personas = 20
cada_uno = plata / cant_personas
```

¿Y si se agregan las bebidas?

Cada estudiantes debe pagar su suma más un plus por las bebidas.
¿Qué se debe cambiar?

Calculamos

Mostramos el resultado.

```
from tkinter.messagebox import showinfo as mostrar_info

mostrar_info('¡Pizzas!', nombre + ' Te va a salir: '
             + str(cada_uno))
```

Consejo

Lo ideal es que todos los `from ... import` estén al principio.

Calculamos

Mostramos el resultado.

```
from tkinter.messagebox import showinfo as mostrar_info

mostrar_info('¡Pizzas!', nombre + ' Te va a salir: '
             + str(cada_uno))
```

Consejo

Lo ideal es que todos los `from ... import` estén al principio.

Calculamos

Mostramos el resultado.

```
from tkinter.messagebox import showinfo as mostrar_info

mostrar_info('¡Pizzas!', nombre + ' Te va a salir: '
             + str(cada_uno))
```

Consejo

Lo ideal es que todos los `from ... import` estén al principio.

Outline

From . . . import

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato
from tkinter.messagebox import showinfo as mostrar_info

root = Tk()
root.withdraw()
```

Cuerpo

```
nombre = pedir_dato('¡Pizzas!',
                    'Indique su nombre y apellido')

plata = 500
cant_personas = 20
cada_uno = plata / cant_personas

mostrar_info('¡Pizzas!', nombre + ' Te va a salir: '
             + str(cada_uno))
```

Outline

¿Qué pasa si. . . ?

Además, los estudiantes que organizan el evento quieren ofrecer dos tipos de pizzas: con palmitos y común. Como hay varias personas que no les gustan los palmitos, deciden preguntarle a cada una si quieren con palmitos o no. Lo bueno es que no afecta al precio final y sólo debe avisarle a Juan que no le ponga palmitos.

¿Qué debe hacer el programa?

Hacer lo mismo que antes y agregar:

- Preguntar si se quiere con palmitos
- Si no se quiere
  - Mostrar un cartel "¡No te olvides de avisarle a Juan!"
- Si se quiere
  - Mostrar un cartel "Genial"

¿Qué veremos?

- Condicionales

¿Qué pasa si. . . ?

Además, los estudiantes que organizan el evento quieren ofrecer dos tipos de pizzas: con palmitos y común. Como hay varias personas que no les gustan los palmitos, deciden preguntarle a cada una si quieren con palmitos o no. Lo bueno es que no afecta al precio final y sólo debe avisarle a Juan que no le ponga palmitos.

¿Qué debe hacer el programa?

Hacer lo mismo que antes y agregar:

- Preguntar si se quiere con palmitos
- Si no se quiere
  - Mostrar un cartel "¡No te olvidés de avisarle a Juan!"
- Si se quiere
  - Mostrar un cartel "¡Genial!"

¿Qué veremos?

- Condicionales

¿Qué pasa si. . . ?

Además, los estudiantes que organizan el evento quieren ofrecer dos tipos de pizzas: con palmitos y común. Como hay varias personas que no les gustan los palmitos, deciden preguntarle a cada una si quieren con palmitos o no. Lo bueno es que no afecta al precio final y sólo debe avisarle a Juan que no le ponga palmitos.

¿Qué debe hacer el programa?

Hacer lo mismo que antes y agregar:

- Preguntar si se quiere con palmitos
- Si no se quiere
  - Mostrar un cartel "¡No te olvidés de avisarle a Juan!"
- Si se quiere
  - Mostrar un cartel "¡Genial!"

¿Qué veremos?

- Condicionales

¿Qué pasa si. . . ?

Además, los estudiantes que organizan el evento quieren ofrecer dos tipos de pizzas: con palmitos y común. Como hay varias personas que no les gustan los palmitos, deciden preguntarle a cada una si quieren con palmitos o no. Lo bueno es que no afecta al precio final y sólo debe avisarle a Juan que no le ponga palmitos.

¿Qué debe hacer el programa?

Hacer lo mismo que antes y agregar:

- Preguntar si se quiere con palmitos
- Si no se quiere
  - Mostrar un cartel "¡No te olvidés de avisarle a Juan!"
- Si se quiere
  - Mostrar un cartel "¡Genial!"

¿Qué veremos?

- Condicionales

¿Qué pasa si. . . ?

Además, los estudiantes que organizan el evento quieren ofrecer dos tipos de pizzas: con palmitos y común. Como hay varias personas que no les gustan los palmitos, deciden preguntarle a cada una si quieren con palmitos o no. Lo bueno es que no afecta al precio final y sólo debe avisarle a Juan que no le ponga palmitos.

¿Qué debe hacer el programa?

Hacer lo mismo que antes y agregar:

- Preguntar si se quiere con palmitos
- Si no se quiere
  - Mostrar un cartel "¡No te olvidés de avisarle a Juan!"
- Si se quiere
  - Mostrar un cartel "¡Genial!"

¿Qué veremos?

- Condicionales

¿Qué pasa si. . . ?

Además, los estudiantes que organizan el evento quieren ofrecer dos tipos de pizzas: con palmitos y común. Como hay varias personas que no les gustan los palmitos, deciden preguntarle a cada una si quieren con palmitos o no. Lo bueno es que no afecta al precio final y sólo debe avisarle a Juan que no le ponga palmitos.

¿Qué debe hacer el programa?

Hacer lo mismo que antes y agregar:

- Preguntar si se quiere con palmitos
- Si no se quiere
  - Mostrar un cartel "¡No te olvidés de avisarle a Juan!"
- Si se quiere
  - Mostrar un cartel "¡Genial!"

¿Qué veremos?

- Condicionales

¿Qué pasa si. . . ?

Además, los estudiantes que organizan el evento quieren ofrecer dos tipos de pizzas: con palmitos y común. Como hay varias personas que no les gustan los palmitos, deciden preguntarle a cada una si quieren con palmitos o no. Lo bueno es que no afecta al precio final y sólo debe avisarle a Juan que no le ponga palmitos.

¿Qué debe hacer el programa?

Hacer lo mismo que antes y agregar:

- Preguntar si se quiere con palmitos
- Si no se quiere
  - Mostrar un cartel "¡No te olvidés de avisarle a Juan!"
- Si se quiere
  - Mostrar un cartel "¡Genial!"

¿Qué veremos?

- Condicionales

¿Qué pasa si. . . ?

Además, los estudiantes que organizan el evento quieren ofrecer dos tipos de pizzas: con palmitos y común. Como hay varias personas que no les gustan los palmitos, deciden preguntarle a cada una si quieren con palmitos o no. Lo bueno es que no afecta al precio final y sólo debe avisarle a Juan que no le ponga palmitos.

¿Qué debe hacer el programa?

Hacer lo mismo que antes y agregar:

- Preguntar si se quiere con palmitos
- Si no se quiere
  - Mostrar un cartel "¡No te olvidés de avisarle a Juan!"
- Si se quiere
  - Mostrar un cartel "¡Genial!"

¿Qué veremos?

- Condicionales

¿Qué pasa si. . . ?

Agregar el siguiente código:

```
from tkinter.messagebox import askyesno as preg_sino

con_palmitos = preg_sino('¡Pizzas!', '¿Con palmitos?')

if con_palmitos:
    mostrar_info('¡Pizzas!', '¡Genial!')
else:
    mostrar_info('¡Pizzas!',
                 '¡No te olvidés de avisarle a Juan!')
```

¿Qué pasa si. . . ?

Agregar el siguiente código:

```
from tkinter.messagebox import askyesno as preg_sino

con_palmitos = preg_sino('¡Pizzas!', '¿Con palmitos?')

if con_palmitos:
    mostrar_info('¡Pizzas!', '¡Genial!')
else:
    mostrar_info('¡Pizzas!',
                 '¡No te olvidés de avisarle a Juan!')
```

Outline

From . . . import

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato
from tkinter.messagebox import showinfo as mostrar_info
from tkinter.messagebox import askyesno as preg_sino

root = Tk()
root.withdraw()
```

Cuerpo

```
nombre = pedir_dato('¡Pizzas!',
                    'Indique su nombre y apellido')
plata = 500
cant_personas = 20
cada_uno = plata / cant_personas

mostrar_info('¡Pizzas!', nombre + ' Te va a salir: '
             + str(cada_uno))

con_palmitos = preg_sino('¡Pizzas!', '¿Con palmitos?')
if con_palmitos:
    mostrar_info('¡Pizzas!', '¡Genial!')
else:
    mostrar_info('¡Pizzas!',
                 '¡No te olvidés de avisarle a Juan!')
```

Outline

¡Pero son 20 estudiantes!

Ahora, no todos los estudiantes sabe cómo ejecutar el programa. Por eso, queremos repetir nuestro programa 20 veces.

¿Qué debe hacer el programa?

- De 1 a 20 (inclusive), repetir:
  - Lo mismo que antes.

¿Qué veremos?

- Repetitivas for
- `range()` o rangos

¡Pero son 20 estudiantes!

Ahora, no todos los estudiantes sabe cómo ejecutar el programa. Por eso, queremos repetir nuestro programa 20 veces.

¿Qué debe hacer el programa?

- De 1 a 20 (inclusive), repetir:
  - Lo mismo que antes.

¿Qué veremos?

- Repetitivas for
- `range()` o rangos

¡Pero son 20 estudiantes!

Ahora, no todos los estudiantes sabe cómo ejecutar el programa. Por eso, queremos repetir nuestro programa 20 veces.

¿Qué debe hacer el programa?

- De 1 a 20 (inclusive), repetir:
  - Lo mismo que antes.

¿Qué veremos?

- Repetitivas for
- `range()` o rangos

¡Pero son 20 estudiantes!

Ahora, no todos los estudiantes sabe cómo ejecutar el programa. Por eso, queremos repetir nuestro programa 20 veces.

¿Qué debe hacer el programa?

- De 1 a 20 (inclusive), repetir:
  - Lo mismo que antes.

¿Qué veremos?

- Repetitivas for
- `range()` o rangos

Probemos algo

Primero, probemos lo siguiente en Geany.

```
for i in range(20):
    print(i)
```

¿Qué sucede? ¿Cuántas veces se imprime en pantalla los números?

¿Por qué range?

`range` devuelve una secuencia de números del 0 hasta el que le digamos.

¿Y si probamos otros valores?

Intente cambiar los valores.

`range(20)` Secuencia de 0 a 19.

`range(1, 20)` Secuencia de 1 a 19.

Probemos algo

Primero, probemos lo siguiente en Geany.

```
for i in range(20):
    print(i)
```

¿Qué sucede? ¿Cuántas veces se imprime en pantalla los números?

¿Por qué range?

`range` devuelve una secuencia de números del 0 hasta el que le digamos.

¿Y si probamos otros valores?

Intente cambiar los valores.

`range(20)` Secuencia de 0 a 19.

`range(1, 20)` Secuencia de 1 a 19.

Probemos algo

Primero, probemos lo siguiente en Geany.

```
for i in range(20):
    print(i)
```

¿Qué sucede? ¿Cuántas veces se imprime en pantalla los números?

¿Por qué range?

`range` devuelve una secuencia de números del 0 hasta el que le digamos.

¿Y si probamos otros valores?

Intente cambiar los valores.

`range(20)` Secuencia de 0 a 19.

`range(1, 20)` Secuencia de 1 a 19.

Probemos algo

Primero, probemos lo siguiente en Geany.

```
for i in range(20):
    print(i)
```

¿Qué sucede? ¿Cuántas veces se imprime en pantalla los números?

¿Por qué range?

`range` devuelve una secuencia de números del 0 hasta el que le digamos.

¿Y si probamos otros valores?

Intente cambiar los valores.

`range(20)` Secuencia de 0 a 19.

`range(1, 20)` Secuencia de 1 a 19.

Probemos algo

Primero, probemos lo siguiente en Geany.

```
for i in range(20):
    print(i)
```

¿Qué sucede? ¿Cuántas veces se imprime en pantalla los números?

¿Por qué range?

`range` devuelve una secuencia de números del 0 hasta el que le digamos.

¿Y si probamos otros valores?

Intente cambiar los valores.

- `range(20)` Secuencia de 0 a 19.
- `range(1, 20)` Secuencia de 1 a 19.

Probemos algo

Primero, probemos lo siguiente en Geany.

```
for i in range(20):
    print(i)
```

¿Qué sucede? ¿Cuántas veces se imprime en pantalla los números?

¿Por qué range?

`range` devuelve una secuencia de números del 0 hasta el que le digamos.

¿Y si probamos otros valores?

Intente cambiar los valores.

- `range(20)` Secuencia de 0 a 19.
- `range(1, 20)` Secuencia de 1 a 19.

¿Código?

```
for i in range(20):
    nombre = pedir_dato('¡Pizzas!',
                        'Indique su nombre y apellido')
    plata = 500
    cant_personas = 20
    cada_uno = plata / cant_personas

    mostrar_info('¡Pizzas!', nombre + ' Te va a salir: '
                 + str(cada_uno))

    con_palmitos = preg_sino('¡Pizzas!', '¿Con palmitos?')

    if con_palmitos:
        mostrar_info('¡Pizzas!', '¡Genial!')
    else:
        mostrar_info('¡Pizzas!',
                     '¡No te olvidés de avisarle a Juan!')
```

Outline

¿Cuántos de palmitos?

Ahora, una vez terminada la aplicación, sería ideal que Juan sepa cuántos pidieron con y sin palmitos. ¿Cómo se puede hacer esto?

¿Qué debe hacer el programa?

- Contar la cantidad de pizzas con palmitos.
- Si el estudiante eligió "con palmitos", sumarle uno a un contador.
- Al final, mostrarle a Juan el resultado.

¿Qué veremos?

- Variables contadoras.
- La verdadera utilidad de las repetitivas for. :)

¿Cuántos de palmitos?

Ahora, una vez terminada la aplicación, sería ideal que Juan sepa cuántos pidieron con y sin palmitos. ¿Cómo se puede hacer esto?

¿Qué debe hacer el programa?

- Contar la cantidad de pizzas con palmitos.
- Si el estudiante eligió "con palmitos", sumarle uno a un contador.
- Al final, mostrarle a Juan el resultado.

¿Qué veremos?

- Variables contadoras.
- La verdadera utilidad de las repetitivas for. :)

¿Cuántos de palmitos?

Ahora, una vez terminada la aplicación, sería ideal que Juan sepa cuántos pidieron con y sin palmitos. ¿Cómo se puede hacer esto?

¿Qué debe hacer el programa?

- Contar la cantidad de pizzas con palmitos.
- Si el estudiante eligió "con palmitos", sumarle uno a un contador.
- Al final, mostrarle a Juan el resultado.

¿Qué veremos?

- Variables contadoras.
- La verdadera utilidad de las repetitivas for. :)

¿Cuántos de palmitos?

Ahora, una vez terminada la aplicación, sería ideal que Juan sepa cuántos pidieron con y sin palmitos. ¿Cómo se puede hacer esto?

¿Qué debe hacer el programa?

- Contar la cantidad de pizzas con palmitos.
- Si el estudiante eligió "con palmitos", sumarle uno a un contador.
- Al final, mostrarle a Juan el resultado.

¿Qué veremos?

- Variables contadoras.
- La verdadera utilidad de las repetitivas for. :)

¿Cuántos de palmitos?

Ahora, una vez terminada la aplicación, sería ideal que Juan sepa cuántos pidieron con y sin palmitos. ¿Cómo se puede hacer esto?

¿Qué debe hacer el programa?

- Contar la cantidad de pizzas con palmitos.
- Si el estudiante eligió "con palmitos", sumarle uno a un contador.
- Al final, mostrarle a Juan el resultado.

¿Qué veremos?

- Variables contadoras.
- La verdadera utilidad de las repetitivas for. :)

¿Cuántos de palmitos?

Ahora, una vez terminada la aplicación, sería ideal que Juan sepa cuántos pidieron con y sin palmitos. ¿Cómo se puede hacer esto?

¿Qué debe hacer el programa?

- Contar la cantidad de pizzas con palmitos.
- Si el estudiante eligió "con palmitos", sumarle uno a un contador.
- Al final, mostrarle a Juan el resultado.

¿Qué veremos?

- Variables contadoras.
- La verdadera utilidad de las repetitivas for. :)

¿Cuántos de palmitos?

Ahora, una vez terminada la aplicación, sería ideal que Juan sepa cuántos pidieron con y sin palmitos. ¿Cómo se puede hacer esto?

¿Qué debe hacer el programa?

- Contar la cantidad de pizzas con palmitos.
- Si el estudiante eligió "con palmitos", sumarle uno a un contador.
- Al final, mostrarle a Juan el resultado.

¿Qué veremos?

- Variables contadoras.
- La verdadera utilidad de las repetitivas for. :)

Variable contadora

¿Qué es un contador?

Es una variable que va contando algo. Normalmente se le suma uno a su contenido cuando una acción sucede.

Recordar siemrpe

Al principio del programa siempre se debe iniciar los contadores:

```
cant_palmitos = 0
```

Variable contadora

¿Qué es un contador?

Es una variable que va contando algo. Normalmente se le suma uno a su contenido cuando una acción sucede.

Recordar siemrpe

Al principio del programa siempre se debe iniciar los contadores:

```
cant_palmitos = 0
```

Variable contadora

¿Qué es un contador?

Es una variable que va contando algo. Normalmente se le suma uno a su contenido cuando una acción sucede.

Recordar siemrpe

Al principio del programa siempre se debe iniciar los contadores:

```
cant_palmitos = 0
```

¿Qué cambiamos?

El condicional

El `if` lo cambiamos así:

```
if con_palmitos:
    cant_palmitos = cant_palmitos + 1
    mostrar_info('¡Pizzas!', '¡Genial!')
else:
    cant_palmitos = cant_palmitos + 1
    mostrar_info('¡Pizzas!',
                 '¡No te olvidés de avisarle a Juan!')
```

Al salir del for

Al salir del `for`, mostramos la cantidad de pizzas con palmitos.

```
mostrar_info('¡Pizzas!', 'Juan, hay ' + str(cant_palmitos)
             + 'pizsas con palmitos')
```

¿Qué cambiamos?

El condicional

El `if` lo cambiamos así:

```
if con_palmitos:
    cant_palmitos = cant_palmitos + 1
    mostrar_info('¡Pizzas!', '¡Genial!')
else:
    cant_palmitos = cant_palmitos + 1
    mostrar_info('¡Pizzas!',
                 '¡No te olvidés de avisarle a Juan!')
```

Al salir del for

Al salir del `for`, mostramos la cantidad de pizzas con palmitos.

```
mostrar_info('¡Pizzas!', 'Juan, hay ' + str(cant_palmitos)
             + 'pizsas con palmitos')
```

¿Qué cambiamos?

El condicional

El `if` lo cambiamos así:

```
if con_palmitos:
    cant_palmitos = cant_palmitos + 1
    mostrar_info('¡Pizzas!', '¡Genial!')
else:
    cant_palmitos = cant_palmitos + 1
    mostrar_info('¡Pizzas!',
                 '¡No te olvidés de avisarle a Juan!')
```

Al salir del for

Al salir del `for`, mostramos la cantidad de pizzas con palmitos.

```
mostrar_info('¡Pizzas!', 'Juan, hay ' + str(cant_palmitos)
             + 'pizsas con palmitos')
```

¿Qué cambiamos?

El condicional

El `if` lo cambiamos así:

```
if con_palmitos:
    cant_palmitos = cant_palmitos + 1
    mostrar_info('¡Pizzas!', '¡Genial!')
else:
    cant_palmitos = cant_palmitos + 1
    mostrar_info('¡Pizzas!',
                 '¡No te olvidés de avisarle a Juan!')
```

Al salir del for

Al salir del `for`, mostramos la cantidad de pizzas con palmitos.

```
mostrar_info('¡Pizzas!', 'Juan, hay ' + str(cant_palmitos)
             + 'pizsas con palmitos')
```

Outline

Código final

```
from tkinter import Tk
from tkinter.simpledialog import askstring as pedir_dato
from tkinter.messagebox import showinfo as mostrar_info
from tkinter.messagebox import askyesno as preg_sino

root = Tk()
root.withdraw()
```

Cuerpo

```
cant_palmitos = 0

for i in range(20):
    nombre = pedir_dato('¡Pizzas!',
                        'Indique su nombre y apellido')
    plata = 500
    cant_personas = 20
    cada_uno = plata / cant_personas

    mostrar_info('¡Pizzas!', nombre + ' Te va a salir: '
                 + str(cada_uno))
```

Cuerpo 2

```
    con_palmitos = preg_sino('¡Pizzas!', '¿Con palmitos?')

    if con_palmitos:
        cant_palmitos = cant_palmitos + 1
        mostrar_info('¡Pizzas!', '¡Genial!')
    else:
        cant_palmitos = cant_palmitos + 1
        mostrar_info('¡Pizzas!',
                     '¡No te olvidés de avisarle a Juan!')

mostrar_info('¡Pizzas!', 'Juan, hay ' + str(cant_palmitos)
             + 'pizsas con palmitos')
```

Outline

Outline

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, publicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿Sin dnf de Fedora?)

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, cálculos en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿Yum de Fedora?)

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, cálculos en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿Yum de Fedora?)

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, cálculos en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿Yum de Fedora?)

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)...

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)...

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)...

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)...

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)...

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)...

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)...

¿Qué más se puede aprender?

- Arreglos, Hashs
- E/S de archivos
- Funciones
- Módulos

¿Y en el mundo?

- Ingeniería de Software: Hay muchas empresas y entidades que usan Python.
  - Webs, aplicaciones de escritorio (PyQt), etc.
- Científicos y académicos lo usan.
  - Minería de datos, estadística, aplicaciones en general.
- Expertos en seguridad lo están usando.
  - Exploits, scripts de automatización, herramientas, etc.
- Expertos en sistemas arman sus programas en Python.
  - Gestores de paquetes (¿dnf de Fedora?)...

Outline

¿Qué les parece Python?

¿Qué les pareció?

- ¿Les gustó?
- ¿Qué les pareció Python?
  - ¿Resulta fácil de leer?
  - ¿Resulta fácil de escribir?
- ¿Quieren probar más cosas?
- ¿Lo usarían en sus proyectos?

¡No se queden con dudas!

¿Preguntas?

¿Qué les parece Python?

¿Qué les pareció?

- ¿Les gustó?
- ¿Qué les pareció Python?
  - ¿Resulta fácil de leer?
  - ¿Resulta fácil de escribir?
- ¿Quieren probar más cosas?
- ¿Lo usarían en sus proyectos?

¡No se queden con dudas!

¿Preguntas?

¿Qué les parece Python?

¿Qué les pareció?

- ¿Les gustó?
- ¿Qué les pareció Python?
  - ¿Resulta fácil de leer?
  - ¿Resulta fácil de escribir?
- ¿Quieren probar más cosas?
- ¿Lo usarían en sus proyectos?

¡No se queden con dudas!

¿Preguntas?

¿Qué les parece Python?

¿Qué les pareció?

- ¿Les gustó?
- ¿Qué les pareció Python?
  - ¿Resulta fácil de leer?
  - ¿Resulta fácil de escribir?
- ¿Quieren probar más cosas?
- ¿Lo usarían en sus proyectos?

¡No se queden con dudas!

¿Preguntas?

¿Qué les parece Python?

¿Qué les pareció?

- ¿Les gustó?
- ¿Qué les pareció Python?
  - ¿Resulta fácil de leer?
  - ¿Resulta fácil de escribir?
- ¿Quieren probar más cosas?
- ¿Lo usarían en sus proyectos?

¡No se queden con dudas!

¿Preguntas?

¿Qué les parece Python?

¿Qué les pareció?

- ¿Les gustó?
- ¿Qué les pareció Python?
  - ¿Resulta fácil de leer?
  - ¿Resulta fácil de escribir?
- ¿Quieren probar más cosas?
- ¿Lo usarían en sus proyectos?

¡No se queden con dudas!

¿Preguntas?

¿Qué les parece Python?

¿Qué les pareció?

- ¿Les gustó?
- ¿Qué les pareció Python?
  - ¿Resulta fácil de leer?
  - ¿Resulta fácil de escribir?
- ¿Quieren probar más cosas?
- ¿Lo usarían en sus proyectos?

¡No se queden con dudas!

¿Preguntas?

¿Qué les parece Python?

¿Qué les pareció?

- ¿Les gustó?
- ¿Qué les pareció Python?
  - ¿Resulta fácil de leer?
  - ¿Resulta fácil de escribir?
- ¿Quieren probar más cosas?
- ¿Lo usarían en sus proyectos?

¡No se queden con dudas!

¿Preguntas?

Outline

Muchas Gracias

¡Muchas gracias!

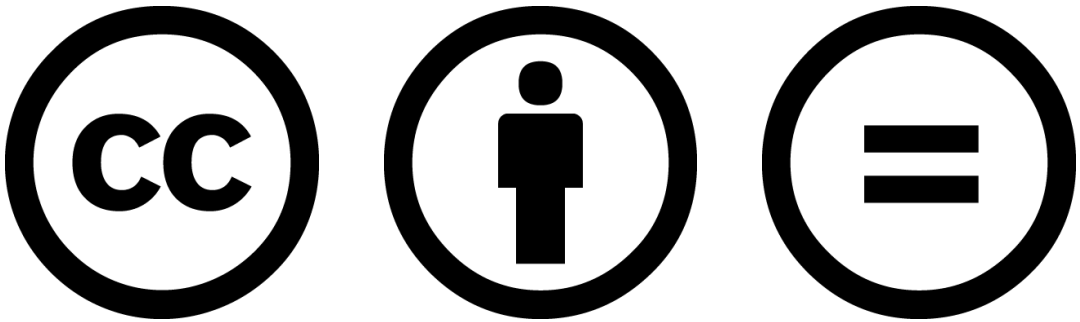
Outline

Outline

Licencia de Esta obra

Excepto en los lugares que se ha indicado lo contrario:

Un Problema menos con Python se distribuye bajo una Licencia Creative Commons Atribución-SinDerivadas 4.0 Internacional.



CC-By-ND