Módulo 3 - Valores booleanos, ejecución condicional, bucles, listas y su procesamiento, operaciones lógicas y de bit a bit

Tomando decisiones

- = asignación
- == comparación, es igual
- `!=` no es igual
- >,<,>=,⇐

if

```
if <exp>:
    linea1
    linea2
  elif <exp>:
    linea5
    linea6
  else:
    linea3
    linea4
```

```
if number1 > number2: larger_number = number1
else: larger_number = number2
```

- no mezclar tabs y espacios en blanco en la indentación
- `else` es la última rama de la cascada, opcionalmente

bucles (ciclos)

- algoritmo, pseudocódigo
- `max()`: máximo de X números → `min()`

ejercicio

```
Érase una vez una tierra de leche y miel, habitada por gente feliz y próspera. La
gente pagaba impuestos, por supuesto, su felicidad tenía límites. El impuesto más
importante, denominado Impuesto Personal de Ingresos (IPI, para abreviar) tenía que
pagarse una vez al año y se evaluó utilizando la siguiente regla:

Si el ingreso del ciudadano no era superior a 85,528 pesos, el impuesto era igual
al 18% del ingreso menos 556 pesos y 2 centavos (esta fue la llamada exención
fiscal ).
Si el ingreso era superior a esta cantidad, el impuesto era igual a 14,839 pesos y
2 centavos, más el 32% del excedente sobre 85,528 pesos.
Tu tarea es escribir una calculadora de impuestos.
```

```
Debe aceptar un valor de punto flotante: el ingreso.
A continuación, debe imprimir el impuesto calculado, redondeado a pesos totales.
Hay una función llamada round() que hará el redondeo por ti, la encontrarás en el
código de esqueleto del editor.
Nota: Este país feliz nunca devuelve dinero a sus ciudadanos. Si el impuesto
calculado es menor que cero, solo significa que no hay impuesto (el impuesto es
igual a cero). Ten esto en cuenta durante tus cálculos.

Observa el código en el editor: solo lee un valor de entrada y genera un resultado,
por lo que debes completarlo con algunos cálculos inteligentes.
```

```python
income = float(input("Introduce el ingreso anual:"))

if income<=85528:
    tax = (income * 18 / 100) - 556.2
else:
    tax = 14839.2 + (income - 85528) * 32 / 100

if tax < 0: tax = 0.0

tax = round(tax, 0)
print("El impuesto es:", tax, "pesos")
```

ejercicio

```
Como seguramente sabrás, debido a algunas razones astronómicas, el año pueden ser
bisiesto o común. Los primeros tienen una duración de 366 días, mientras que los
últimos tienen una duración de 365 días.

Desde la introducción del calendario Gregoriano (en 1582), se utiliza la siguiente
regla para determinar el tipo de año:

Si el número del año no es divisible entre cuatro, es un año común.
De lo contrario, si el número del año no es divisible entre 100, es un año
bisiesto.
De lo contrario, si el número del año no es divisible entre 400, es un año común.
De lo contrario, es un año bisiesto.
Observa el código en el editor: solo lee un número de año y debe completarse con
las instrucciones que implementan la prueba que acabamos de describir.

El código debe mostrar uno de los dos mensajes posibles, que son Año Bisiesto o Año
Común, según el valor ingresado.

Sería bueno verificar si el año ingresado cae en la era Gregoriana y emitir una
advertencia de lo contrario: No dentro del período del calendario Gregoriano.
Consejo: utiliza los operadores != y %.
```

```python
year = int(input("Introduce un año:"))

if year >= 1582:
    if year % 4 != 0: ano="común"
    elif year % 100 != 0: ano="bisiesto"
    elif year % 400 != 0: ano="comun"
```

```
    else: ano="bisiesto"
    print("Año "+ano)
else:
    print("no es gregoriano")
```

while

```
while conditional_expression:
    instruction
```

```
while True:
    print("Estoy atrapado dentro de un bucle.")
```

- Si deseas ejecutar más de una sentencia dentro de un while, debes (como con if) poner sangría a todas las instrucciones de la misma manera.
- Una instrucción o conjunto de instrucciones ejecutadas dentro del while se llama el cuerpo del bucle.
- Si la condición es False (igual a cero) tan pronto como se compruebe por primera vez, el cuerpo no se ejecuta ni una sola vez (ten en cuenta la analogía de no tener que hacer nada si no hay nada que hacer).
- El cuerpo debe poder cambiar el valor de la condición, porque si la condición es True al principio, el cuerpo podría funcionar continuamente hasta el infinito. Observa que hacer una cosa generalmente disminuye la cantidad de cosas por hacer.

ejercicio

```
Un mago junior ha elegido un número secreto. Lo ha escondido en una variable
llamada secret_number. Quiere que todos los que ejecutan su programa jueguen el
juego Adivina el número secreto, y adivina qué número ha elegido para ellos.
¡Quiénes no adivinen el número quedarán atrapados en un bucle sin fin para siempre!
Desafortunadamente, él no sabe cómo completar el código.

Tu tarea es ayudar al mago a completar el código en el editor de tal manera que el
código:

Pedirá al usuario que ingrese un número entero.
Utilizará un bucle while.
Comprobará si el número ingresado por el usuario es el mismo que el número escogido
por el mago. Si el número elegido por el usuario es diferente al número secreto del
mago, el usuario debería ver el mensaje "¡Ja, ja! ¡Estás atrapado en mi bucle!"  y
se le solicitará que ingrese un número nuevamente. Si el número ingresado por el
usuario coincide con el número escogido por el mago, el número debe imprimirse en
la pantalla, y el mago debe decir las siguientes palabras: "¡Bien hecho, muggle!
Eres libre ahora".
¡El mago está contando contigo! No lo decepciones.


INFO EXTRA

Por cierto, observa la función print(). La forma en que lo hemos utilizado aquí se
llama impresión multilínea. Puede utilizar comillas triples para imprimir cadenas
en varias líneas para facilitar la lectura del texto o crear un diseño especial
basado en texto. Experimenta con ello.
```

```
secret_number = 777
```

```
print(
"""
+=================================+
| ¡Bienvenido a mi juego, muggle!  |
| Introduce un número entero       |
| y adivina qué número he          |
| elegido para ti.                 |
| Entonces,                        |
| ¿Cuál es el número secreto?      |
+=================================+
""")
num = 0

while num!=secret_number:
    num = int(input())
    if num!=secret_number: print("¡Ja, ja! ¡Estás atrapado en mi bucle!")
    else: print("¡Bien hecho, muggle! Eres libre ahora")
```

for

```
for i in range(100):
    # do_something()
    pass
```

```
for i in range(2, 8):
    print("El valor de i es actualmente", i)
```

- La palabra reservada for abre el bucle for; nota - No hay condición después de eso; no tienes que pensar en las condiciones, ya que se verifican internamente, sin ninguna intervención.
- Cualquier variable después de la palabra reservada for es la variable de control del bucle; cuenta los giros del bucle y lo hace automáticamente.
- La palabra reservada in introduce un elemento de sintaxis que describe el rango de valores posibles que se asignan a la variable de control.
- La función **range()** (esta es una función muy especial) es responsable de generar todos los valores deseados de la variable de control; en nuestro ejemplo, la función creará (incluso podemos decir que alimentará el bucle con) valores subsiguientes del siguiente conjunto: 0, 1, 2 .. 97, 98, 99; nota: en este caso, la función range() comienza su trabajo desde 0 y lo finaliza un paso (un número entero) antes del valor de su argumento.
- Nota la palabra clave **pass** dentro del cuerpo del bucle - no hace nada en absoluto; es una instrucción vacía : la colocamos aquí porque la sintaxis del bucle for exige al menos una instrucción dentro del cuerpo (por cierto, if, elif, else y while expresan lo mismo).

```
for i in range(2, 8, 3):
    print("El valor de i es actualmente", i)
```

- La función range() también puede aceptar tres argumentos: Echa un vistazo al código del editor.
- El tercer argumento es un incremento: es un valor agregado para controlar la variable en cada giro del bucle (como puedes sospechar, el valor predeterminado del incremento es 1).
- el conjunto generado por range() debe ordenarse en un orden ascendente. No hay forma de forzar el range() para crear un conjunto en una forma diferente. Esto significa que el segundo argumento de range() debe ser mayor que el primero.

ejercicio

```
¿Sabes lo que es Mississippi? Bueno, es el nombre de uno de los estados y ríos en
los Estados Unidos. El río Mississippi tiene aproximadamente 2,340 millas de largo,
lo que lo convierte en el segundo río más largo de los Estados Unidos (el más largo
es el río Missouri). ¡Es tan largo que una sola gota de agua necesita 90 días para
recorrer toda su longitud!

La palabra Mississippi también se usa para un propósito ligeramente diferente: para
contar mississippily (mississippimente).

Si no estás familiarizado con la frase, estamos aquí para explicarte lo que
significa: se utiliza para contar segundos.

La idea detrás de esto es que agregar la palabra Mississippi a un número al contar
los segundos en voz alta hace que suene más cercano al reloj, y por lo tanto "un
Mississippi, dos Mississippi, tres Mississippi" tomará aproximadamente unos tres
segundos reales de tiempo. A menudo lo usan los niños que juegan al escondite para
asegurarse de que el buscador haga un conteo honesto.


Tu tarea es muy simple aquí: escribe un programa que use un bucle for para "contar
de forma mississippi" hasta cinco. Habiendo contado hasta cinco, el programa
debería imprimir en la pantalla el mensaje final "¡Listos o no, ahí voy!"

Utiliza el esqueleto que hemos proporcionado en el editor.

INFO EXTRA

Ten en cuenta que el código en el editor contiene dos elementos que pueden no ser
del todo claros en este momento: la sentencia import time y el método sleep().
Vamos a hablar de ellos pronto.

Por el momento, nos gustaría que supieras que hemos importado el módulo time y
hemos utilizado el método sleep() para suspender la ejecución de cada función
posterior de print() dentro del bucle for durante un segundo, de modo que el
mensaje enviado a la consola se parezca a un conteo real. No te preocupes, pronto
aprenderás más sobre módulos y métodos.
```

```python
import time

for count in range(1,6):
    print(count,"Mississippi")
    time.sleep(1)
print("¡Listos o no, ahí voy!")
```

break y continue

From:
https://miguelangel.torresegea.es/wiki/ - **miguel angel torres egea**

Permanent link:
https://miguelangel.torresegea.es/wiki/info:cursos:netacad:python:pe1m3?rev=1654795588

Last update: **09/06/2022 10:26**