PROGRAMACIÓN

Guía de estudio 14: Todo de nuevo, otra vez...

Bucles/Ciclos while y for en Python

Nivel: Secundario - Modalidad Educación Técnico-Profesional.
Ciclo: Segundo ciclo.
Especialidades: Programación, Informática.

Introducción

Así como vimos la estructura condicional, veremos otro tipo de estructuras, las iterativas o cíclicas. Estas estructuras, también llamadas bucles, nos permiten ejecutar muchas veces un mismo código sin tener que escribirlo esa cantidad de veces. Como ya se podrán dar cuenta, esto es de gran utilidad y de gran ayuda para nuestras cansadas manos.

¿Qué estamos aprendiendo? Qué son los bucles. Cómo utilizar el while y el for, diferencias y ejemplos de utilización.

inet | Instituto Nacional de Educación Tecnológica

Ministerio de Educación Argentina

Recursos

Video nro 37 del curso: https://youtu.be/YEWxlbffgxE	
Video nro 38 del curso: https://youtu.be/mRI8C2ZhDkg	

Secuencia didáctica

Mirá los siguientes videos para entender los bucles:

https://youtu.be/YEWxlbffgxE	
https://youtu.be/mRI8C2ZhDkg	

Actividades

Después de mirar el video del link o del QR, te invitamos a que reflexiones con las siguientes preguntas.

No olvides escribir tus respuestas, te facilitarán realizar trabajos prácticos y repasar los temas que hayas estudiado con esta guía.

1) Escribir un programa que muestre todo lo que el usuario introduzca hasta que el usuario escriba "salir" que terminará.

2) Creá un programa que pida introducir un número y un modo. Los modos serán "par" o "impar". Luego se imprimirán todos los números pares o impares (según corresponda) desde 0 hasta el número introducido.

3) Creá un programa que comience con el siguiente menú:

1 - Registrar usuario
2 - Mostrar usuario
3 - Salir.

El operador deberá ingresar un número para determinar la opción del menú.

Si se registra un usuario nuevo, se deberá pedir nombre, mail y celular, guardarlo en un diccionario y volver al menú. En cambio, si se desea mostrar el usuario, se debe preguntar el nombre y, al ingresarlo, mostrar todos sus datos. Luego debe volver al menú. La única forma de terminar el programa es eligiendo la opción de salir.

El diccionario será de la forma diccionario = {"nombre":{"email":"a@b.c", "teléfono":123456}}

4) Desarrollá un código que comience con una lista de diez "0" (lista = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) y que pida dos números N y M. Si N está entre 1 y 10, se deberá poner el número M en la posición N correspondiente. Si N es 0, se le sumará el número M a todas las celdas (es decir, todos los elementos sumarán M a sus valores). Si N es -1 y el número M es mayor o igual a 0, se deberá mostrar la cantidad de veces que está ese número en la lista. El programa terminará solo cuando N y M sean ambos -1.

5) Escribir un programa en el cual se deba ingresar un número entero y muestre por pantalla un triángulo con la altura (en caracteres) del número introducido y de ancho 2 por el número de fila (exceptuando la fila 0, que tendrá 1 carácter). Por ejemplo, si se introdujo 5, se deberá mostrar:

```
*           (fila 0, 1 "*")
**          (fila 1, 2 "*")
****        (fila 2, 4 "*")
******      (fila 3, 6 "*")
********    (fila 4, 8 "*")
```

Para seguir aprendiendo...

En el último video se menciona la función "range" y la posibilidad de utilizarlo en el ciclo "for". Primero analicemos la función range(N, M). Esta función genera rangos entre N y M-1. Por ejemplo, si escribiéramos range(5, 9) eso crearía un rango entre

esos 2 números (5, 6, 7, 8). Vale aclarar que si quisiera un rango entre 0 y un número positivo, se podría simplemente escribir range(3) (siendo el rango 0, 1, 2)

¿Y cómo podemos utilizar esto? Bueno, esto tiene varias aplicaciones. Veremos 2 casos, uno primero con el cual están un poco más familiarizados, y luego, lo veremos aplicado al for.

Si quisiéramos crear una lista con números del "-5" al "5", en vez de escribir todos los números manualmente, podríamos escribir:

```
lista = list(range(-5, 6))
```

¿De qué no sirve esto para el for? Bueno, tomemos el ejemplo de escribir los números pares de 0 a x y reescribamos usando un for y range

```
for i in range(x):
    if (i%2) == 0:
        print(f"{i} es par")
```

Como se puede ver (y más si ejecutan el código), directamente "i" va tomando todos los valores que estén en el rango, en este caso, de 0 a x-1.