

Clase 2 Condicionales y estructuras cíclicas

Estructuras Condicionales y Estructuras Cíclicas.

Estructuras Condicionales

Las estructuras de control condicionales, son aquellas que nos permiten evaluar si una o más condiciones se cumplen, para decir qué acción vamos a ejecutar. La evaluación de condiciones, solo puede arrojar 1 de 2 resultados: verdadero o falso (True o False).

Para describir la evaluación a realizar sobre una condición, se utilizan **operadores relacionales** (o de comparación):

Símbolo	Significado	Ejemplo	Resultado
==	Igual que	5 == 7	False
!=	Distinto que	rojo != verde	True
<	Menor que	8 < 12	True
>	Mayor que	12 > 7	True
<=	Menor o igual que	12 <= 12	True
>=	Mayor o igual que	4 >= 5	False

Y para evaluar más de una condición simultáneamente, se utilizan operadores lógicos:

Operador	Ejemplo	Explicación	Resultado
and	5 == 7 and 7 < 12	False and False	False
and	9 < 12 and 12 > 7	True and True	True
and	9 < 12 and 12 > 15	True and False	False
or	12 == 12 or 15 < 7	True or False	True
or	7 > 5 or 9 < 12	True or True	True
xor	4 == 4 xor 9 > 3	True o True	False
xor	4 == 4 xor 9 < 3	True o False	True

Las estructuras de control de flujo condicionales, se definen mediante el uso de tres palabras claves reservadas, del lenguaje: if (si), elif (sino, si) y else (sino).

Ejemplo: Si gasto hasta 10000, pago con dinero en efectivo. Si no, si gasto más de 10000 pero menos de 30000, pago con tarjeta de débito. Si no, pago con tarjeta de crédito.

```
if compra <= 10000:
    print ("Pago en efectivo")
elif compra > 10000 and compra < 30000:
    print ("Pago con tarjeta de débito")
else:
    print ("Pago con tarjeta de crédito")
```

Estructuras Cíclicas

A diferencia de las estructuras de control condicionales, las iterativas (también llamadas cíclicas o bucles), nos permiten ejecutar un mismo código, de manera repetida, mientras se cumpla una condición.

En Python se dispone de dos estructuras cíclicas:

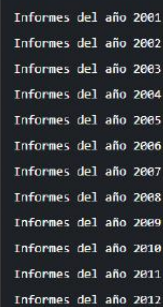
- El bucle while
- El bucle for

Bucle While

- Este bucle, se encarga de ejecutar una misma acción "mientras que" una determinada condición se cumpla. Ejemplo: Mientras que año sea menor o igual a 2012, imprimir la frase "Informes del Año año".

```
anio = 2001
while anio <= 2012:
    print ("Informes del Año", str(anio))
    anio += 1
```

La iteración anterior, generará la siguiente salida:



```
Informes del año 2001
Informes del año 2002
Informes del año 2003
Informes del año 2004
Informes del año 2005
Informes del año 2006
Informes del año 2007
Informes del año 2008
Informes del año 2009
Informes del año 2010
Informes del año 2011
Informes del año 2012
```


Si miras la última línea:

```
anio += 1
```

Podrás notar que en cada iteración, incrementamos el valor de la variable que condiciona el bucle (anio). Si no lo hiciéramos, esta variable siempre sería igual a 2001 y el bucle se ejecutaría de forma infinita, ya que la condición (anio <= 2012) siempre se estaría cumpliendo.

Pero ¿Qué sucede si el valor que condiciona la iteración no es numérico y no puede incrementarse? En ese caso, podremos utilizar una estructura de control condicional, anidada dentro del bucle, y frenar la ejecución cuando el condicional deje de cumplirse, con la palabra clave reservada break:

```
while True:
    nombre = input("Indique su nombre: ")
    if nombre:
        break
```

El bucle anterior, incluye un condicional anidado que verifica si la variable nombre es verdadera (solo será verdadera si el usuario tepea un texto en pantalla cuando el nombre le es solicitado). Si es verdadera, el bucle para (break). Sino, seguirá ejecutándose hasta que el usuario, ingrese un texto en pantalla.

Bucle For

El bucle for, en Python, es aquel que nos permitirá iterar sobre una variable compleja, del tipo lista o tupla:

1) Por cada nombre en mi_lista, imprimir nombre

```
mi_lista = ['Juan', 'Antonio', 'Pedro', 'Herminio']
for nombre in mi_lista:
    print(nombre)
```

2) Por cada color en mi_tupla, imprimir color:

```
mi_tupla = ('rosa', 'verde', 'celeste', 'amarillo')
for color in mi_tupla:
    print (color)
```

Bucle For

En los ejemplos anteriores, nombre y color, son dos variables declaradas en tiempo de ejecución (es decir, se declaran dinámicamente durante el bucle), asumiendo como valor, el de cada elemento de la lista (o tupla) en cada iteración.

Otra forma de iterar con el bucle for, puede emular a while:

3) Por cada año en el rango 2001 a 2019, imprimir la frase "Informes del Año año":

```
# -*- coding: utf-8 -*-
```

```
for anio in range(2001, 2019):
```

```
    print ("Informes del Año", str(anio))
```

Un poco acerca de Estructuras de Datos en Python

Listas

Una lista es una estructura de datos que contiene una colección o secuencia de datos. Los datos o elementos de una lista deben ir separados con una coma y todo el conjunto entre corchetes. Se dice que una lista es una estructura mutable porque además de permitir el acceso a los elementos, pueden suprimirse o agregarse nuevos.

```
ListaEstaciones = ["Invierno", "Primavera", "Verano", "Otoño"] # Declara lista
```

EJERCITARIO DE CLASE:

PARTE 1

Ejercitario – Estructuras Condicionales

Ejercicio Resuelto

1. Escribir un programa que pida al usuario un número entero y muestre por pantalla si es par o impar.

```

ejemplo1.py > ...
1  # Solicitar al usuario que ingrese un número entero
2  numero = int(input("Introduce un número entero: "))
3
4  # Verificar si el número es par o impar
5  if numero % 2 == 0:
6      print(f"El número {numero} es par.")
7  else:
8      print(f"El número {numero} es impar.")

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```

PS E:\MGR\AMERICANA\UA\OPTATIVO_II\EJERCICIOS_CLASE2> & C:/Users/Usuario/AppData/Local/Programs/Python/Python39-64/Scripts/python.exe E:\MGR\AMERICANA\UA\OPTATIVO_II\EJERCICIOS_CLASE2/ejemplo1.py
Introduce un número entero: 14
El número 14 es par.
PS E:\MGR\AMERICANA\UA\OPTATIVO_II\EJERCICIOS_CLASE2>

```

```

ejem_1_forma2.py > ...
1  n = int(input("Introduce un número entero: "))
2  if n % 2 == 0:
3      print("El número " + str(n) + " es par")
4  else:
5      print("El número " + str(n) + " es impar")

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

```

PS E:\MGR\AMERICANA\UA\OPTATIVO_II\EJERCICIOS_CLASE2> & C:/Users/Usuario/AppData/Local/Programs/Python/Python39-64/Scripts/python.exe E:\MGR\AMERICANA\UA\OPTATIVO_II\EJERCICIOS_CLASE2/ejem_1_forma2.py
Introduce un número entero: 17
El número 17 es impar
PS E:\MGR\AMERICANA\UA\OPTATIVO_II\EJERCICIOS_CLASE2>

```

Uso de :

```
print(f"El número {numero} es impar.")
```

f"El número {numero} es impar." es una f-string.

{numero} es una expresión dentro de la f-string que se evalúa y se reemplaza por el valor de la variable numero.

```
print("El número " + str(n) + " es par")
```

Se utiliza str(n) para convertir el número n a una cadena de texto y concatenarlo con el mensaje.

**** Probar el código de las dos maneras**

Resolver:

2. Pedir por teclado dos números y decir si son iguales o no.
3. Escribir un programa que pregunte al usuario su edad y muestre por pantalla si es mayor de edad o no.
4. Escribir un programa que almacene la cadena de caracteres contraseña en una variable, pregunte al usuario por la contraseña e imprima por pantalla si la contraseña introducida por el usuario coincide con la guardada en la variable sin tener en cuenta mayúsculas y minúsculas.
5. Escribir un programa para una empresa que tiene salas de juegos para todas las edades y quiere calcular de forma automática el precio que debe cobrar a sus clientes por entrar. El programa debe preguntar al usuario la edad del cliente y mostrar el precio de la entrada. Si el cliente es menor de 4 años puede entrar gratis, si tiene entre 4 y 18 años debe pagar 10.000 Gs. y si es mayor de 18 años, 15.000 Gs.

PARTE 2:

Ejercitario – Estructuras Cíclicas

Ejercicios Resueltos

1. Escribir un programa que muestre por pantalla la tabla de multiplicar del 1 al 10.

```

1 for i in range(1, 11):
2     for j in range(1, 11):
3         print(i*j, end="\t")
4     print("\n")

```

```

PS C:\Users\zulma\OneDrive\Desktop\PYTHON> & C:/Users/zulma/AppData/Local/Programs/Python/Python39/python.exe c:/Users/zulma/OneDrive/Desktop/PYTHON/Ciclo1.py
1 2 3 4 5 6 7 8 9 10
2 4 6 8 10 12 14 16 18 20
3 6 9 12 15 18 21 24 27 30
4 8 12 16 20 24 28 32 36 40
5 10 15 20 25 30 35 40 45 50
6 12 18 24 30 36 42 48 54 60
7 14 21 28 35 42 49 56 63 70
8 16 24 32 40 48 56 64 72 80
9 18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100
PS C:\Users\zulma\OneDrive\Desktop\PYTHON>

```

2. Escribir un programa que pida al usuario un número entero positivo y muestre por pantalla todos los números impares desde 1 hasta ese número separados por comas.

```

1 n = int(input("Introduce un número entero positivo: "))
2 for i in range(1, n+1, 2):
3     print(i, end=", ")

```

```

Microsoft PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

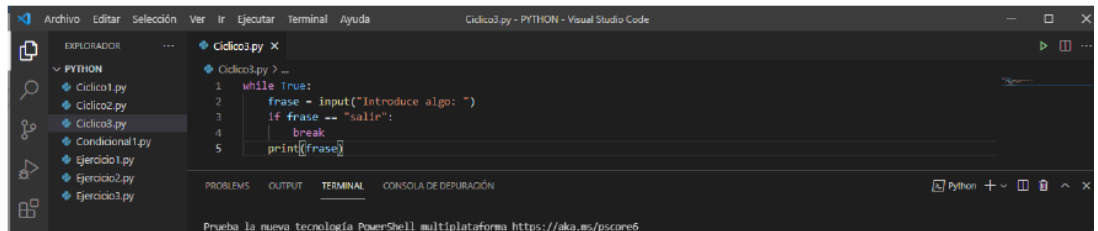
PS C:\Users\zulma\OneDrive\Desktop\PYTHON> & C:/Users/zulma/AppData/Local/Programs/Python/Python39/python.exe c:/Users/zulma/OneDrive/Desktop/PYTHON/Ciclo2.py
Introduce un número entero positivo: 14
1, 3, 5, 7, 9, 11, 13,
PS C:\Users\zulma\OneDrive\Desktop\PYTHON>

```

Aclaraciones del código:

- Se utiliza un bucle `for` para iterar desde 1 hasta `n` (inclusive) con un paso de 2.
- El tercer argumento 2 en `range(1, n+1, 2)` indica que el bucle incrementará `i` en 2 en cada iteración, lo que significa que solo se considerarán los números impares.

3. Escribir un programa que muestre el eco de todo lo que el usuario introduzca hasta que el usuario escriba “salir” que terminará.



The screenshot shows the Visual Studio Code interface with a Python file named 'Ciclco3.py' open. The code implements a while loop that prompts the user to 'Introduce algo: ' and prints the input until the user enters 'salir'.

```
1 while True:
2     frase = input("Introduce algo: ")
3     if frase == "salir":
4         break
5     print(frase)
```

Resolver:

4. Pedir por teclado 10 números. Indicar cuántos de ellos son números positivos, cuantos son números negativos y la cantidad de ceros.
5. Escribir un programa que visualice en pantalla los números pares entre 1 y 25.
6. Leer un número y mostrar su cuadrado, repetir el proceso hasta que se introduzca un número negativo.
7. Dadas 6 notas, escribir la cantidad de alumnos aprobados y los no aprobados. (Aprobados mayores a 1)(No aprobados menor o igual a 1).

PARTE 3 – PLANTEAR UNA SITUACION DE LA VIDA REAL Y PRESENTAR LA SOLUCION MEDIANTE ESTRUCTURAS CONDICIONALES Y ESTRUCTURAS CÍCLICAS.

- PLANTEAMIENTO:

- SOLUCIÓN: