Python es un lenguaje de programación de propósito general de alto nivel. Para acceder a este programa se puede optar por una de las siguientes maneras:

• Python en internet: https: //www.programiz.com/ python-programming/online-compiler/

Sintaxis básica de Python

En la escritura de programas usualmente tenemos que emplear: comentarios, operaciones aritméticas, operaciones lógicas y de comparación, estructuras condicionales, estructuras repetitivas, funciones y una diversidad de tipos de datos.

Los comentarios permiten explicar las distintas partes del código. En Python los comentarios se inician con el símbolo #.

Las operaciones aritméticas en Python se expresan de la siguiente forma:

1.2. Python

• adición A+B;

• substracción A-B;

• multiplicacón A*B;

• división A/B;

• exponenciación A**B.

Operaciones lógicas y de comparación.

Las operaciones lógicas y de comparación dan como resultado los valores booleanos: True (la expresión es ver dadera) y False (la expresión es falsa).

Los operadores de comparación en Python se expresan de la siguiente manera:

• igual que a == b;

• no es igual que a! = b;

• menor que a < b;

• menor o igual que a <= b;

• mayor que a > b;

• mayor o igual que a >= b. Los operadores lógicos:

Python

• conjunción and;

• disyunción or;

• negación not.

Los bloques de código en las estructuras de control se escriben con sangría. La estructura condicional se usa en las operaciones de decisión.

En Python tiene la forma:

```
if condición:
        grupo de ordenes
 else :
        grupo de ordenes
```

Ejemplo 2. Escriba un código para elegir el menor de 2 números.

Resolución

Algoritmo 1.2 Estructura condicional

```
#asignación de valores
x = 2
y = 4 i f x < y:
print (’El número menor es:’, x)
else: print (’El número menor es:’, y)
```

A veces es necesario ejecutar un bloque de código si una de varias condiciones se cumple.

En este caso se usan las estructuras condicionales en cascada las cuales tienen la siguiente forma:

```
i f condición 1:
        grupo de ordenes
e l i f condición 2:
        grupo de ordenes
e l i f condición 3:
        grupo de ordenes
else:
        grupo de ordenes
```

Ejemplo 3. Escriba un código para determinar en qué cuadrante se encuentra un punto ingresado por teclado.

Resolución

Algoritmo 1.3 Estructura condicional en cascada

```
x, y = int(input()) , int(input())
i f x > 0 and y > 0:
        print (" primer cuadrante ")
e l i f x < 0 and y > 0:
        print (" segundo cuadrante ")
e l i f x < 0 and y < 0:
        print (" tercer cuadrante ")
e l i f x > 0 and y < 0:
        print (" cuarto cuadrante ")
else:
        print ("P se encuentra en los ejes ... o en el centro de coordenadas.")
```

Las estructuras de control de repetición ejecutan ciclica mente un grupo de órdenes según ciertas condiciones. La estructura de repetición while tiene la forma:

while condición:

    grupo de ordenes

Ejemplo 4. Escriba un código para sumar los primeros 6 números naturales.

Resolución

Algoritmo 1.4 Estructura de repetición 'while'

```
número, suma = 1, 1
while número <= 5:
    número += 1
    suma = suma + número
    print (f 'La suma de los {número}...
    primeros números es {suma}')
```

El resultado de la ejecución de este código es:

La suma de los 2 primeros números es 3

La suma de los 3 primeros números es 6

La suma de los 4 primeros números es 10

La suma de los 5 primeros números es 15

La suma de los 6 primeros números es 21

La repetición también se puede implementar con los ciclos for.

La diferencia con un ciclo while consiste en que en el ciclo for se recorre los elementos de un objeto iterable1.

Una estructura de repetición 'for' tiene la forma:

f or ’elemento ’ in ’iterable ’:

grupo de ordenes

Análisis del código

• frommódulo1 import función- importa solo una función de un módulo;

• itertools- módulo que contiene funciones iterativas;

• product- función que implementa el producto cartesiano;

• repeat = n- establece las veces que se ejecuta ’product’

• list()- crea una lista1 de elementos indexados;

• range(n)- crea una secuencia de números de 0 a n-1.

• print(f"E = {E}\n")- imprime la salida con formato;

• {E} es el valor de la variable a mostrar y

• \n es un salto de renglón.

• sum(x)- suma los elementos de un conjunto

• a%b- es el operador módulo que da como resultado el residuo del cociente de a y b.

• numpy1- módulo de Python que permite realizar cálculos científicos;

• np.prod- calcula el producto de los elementos de un conjunto;

• len(E)- función que entrega el número de elementos de un objeto;

• {p:.4f}- en la impresión con formato muestra la salida con cuatro cifras decimales.

• random- módulo de Python que permite generar números pseudoaleatorios y realizar acciones aleatorias, como elegir un valor aleatorio de una lista.

• rd.choice()- selecciona aleatoriamente un elemento de una secuencia especificada.

• permutations()- forma los ordenamientos posibles, sin elementos repetidos.

• urna = {}- crea una lista vacía para almacenar los elementos de la urna.

• el primer ciclo for- llena la urna con las bolas de distintos colores.

• N=100000- establece el número de simulaciones a realizar.

• N_E=0-iniciaelcontadordeocurrencias del evento E.

• segundo ciclo for- cuenta el número de ocurrencias de E.

• lst.append(urna[rd.randint(0, 18)])- realiza la selección aleatoria de las bolas de la urna.

• collections- es un módulo que proporciona estructuras alternativas de agrupamiento.

• diccionario- es una estructura de agrupamiento que permite expresar de forma concisa los datos. Los elementos del diccionario se presentan en pares. Por ejemplo, E = {'a' : 3,'b' : 2} donde el primer elemento 'a' es la clave y el segundo elemento 3 es el valor.

• defaultdict- es un contenedor del módulo colecciones. A diferencia de los diccionarios defaultdict no genera un mensaje de error si la clave no está presente

• primer ciclo for- crea la permutación de E; la usa como clave para ponderado e incrementa el respectivo valor cada vez que ocurre una permutación específica.

• segundo ciclo for- cuenta las ocurrencias de las per mutaciones.

• items()- se usa para devolver una lista con todas las claves del diccionario con valores.

• count()- devuelve el número de elementos de un valor especificado.

ejercicio

1. Escriba un programa que calcule la probabilidad de que el producto de los puntos de tres lanzamientos de los dados sea menor que 50.

2. Escriba un código para determinar en qué cuadrante se encuentra un punto ingresado por teclado.

3. Determine el número de formas en las que se puede vestir un joven si dispone de dos pares de zapatos, dos pantalones y tres camisas.