

2021-I_mcpp_taller_2_Valentina_Cuenca

February 13, 2021

1 Taller 2

Métodos Computacionales para Políticas Públicas - UROSARIO

Entrega: viernes 19-feb-2021 11:59 PM

Valentina Cuenca Norma.cuenca@urosario.edu.co

1.1 Instrucciones:

- Guarde una copia de este *Jupyter Notebook* en su computador, idealmente en una carpeta destinada al material del curso.
- Modifique el nombre del archivo del *notebook*, agregando al final un guión inferior y su nombre y apellido, separados estos últimos por otro guión inferior. Por ejemplo, mi *notebook* se llamaría: mcpp_taller2_santiago_matallana
- Marque el *notebook* con su nombre y e-mail en el bloque verde arriba. Reemplace el texto “[Su nombre acá]” con su nombre y apellido. Similar para su e-mail.
- Desarrolle la totalidad del taller sobre este *notebook*, insertando las celdas que sea necesario debajo de cada pregunta. Haga buen uso de las celdas para código y de las celdas tipo *markdown* según el caso.
- Recuerde salvar periódicamente sus avances.
- Cuando termine el taller:
 1. Descárguelo en PDF.
 2. Suba los dos archivos (.pdf y .ipynb) a su repositorio en GitHub antes de la fecha y hora límites.

(El valor de cada ejercicio está en corchetes [] después del número de ejercicio.)

1.2 1.

[Pensar como un computador] Considere el siguiente código: `if x > 2: if y > 2: z = x + y print("z es", z) else: print(x)` ¿Cuál es el resultado si

- a) $x = 2, y = 5$? , En este caso el resultado es “2”
- b) $x = 3, y = 1$?, en este caso no hace nada
- c) $x = 1, y = 1$?, resultado “ 1”
- d) $x = 4, y = 3$?, resultado es “ z es 7”

1.2.1 Justificación

a)

```
[1]: x= 2  
     y= 5  
     x > 2
```

[1]: False

Al ser falso que x es mayor a 2, entonces pasa a else, donde imprime el valor de 2.

b)

```
[6]: x = 3  
     y = 1  
     x > 2
```

[6]: True

```
[5]: y > 2
```

[5]: False

Es verdadero que x es mayor a 2, entonces entra a ese bloque, pero es falso que y es mayor a 2, por lo que no entra en el siguiente bloque, y como no tiene instrucciones de hacer más, no realiza ninguna acción.

c)

```
[8]: x = 1  
     y = 1  
     x > 2
```

[8]: False

Al ser falso que x es mayor a 2, entonces pasa a else, donde imprime el valor de 2.

d)

```
[9]: x = 4  
     y = 3  
     x > 2
```

[9]: True

```
[10]: y > 2
```

[10]: True

```
[11]: z = x + y  
      z
```

[11]: 7

Es verdadero que x es mayor a 2, entonces entra en ese bloque, luego comprueba si y es mayor a 7, como es verdadero también entra en ese bloque, define z como la suma de x y y, dando como resultado 7, y luego imprime " z es 7"

1.3 2.

[Pensar como un computador] ¿Cuál es el resultado del siguiente código y cuántas veces se recorre el loop? `i = 0 while i < 10: i = i + 1 if i print(i)` El loop se hace 10 veces, hasta que i es igual a 9, porque cuando se convierte en 10 ya no puede entrar al while.

1.3.1 Resultado

2
4
6
8
10

1.3.2 Justificación

```
[39]: i = 0  
      i < 10
```

[39]: True

```
[40]: i = i + 1  
      i % 2 == 0
```

[40]: False

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, pero al no ser par no se imprime el valor

```
[33]: i < 10
```

[33]: True

```
[41]: i = i + 1  
      i % 2 == 0
```

[41]: True

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, al ser par se imprime el valor, que es 2.

```
[35]: i < 10
```

```
[35]: True
```

```
[42]: i = i + 1  
      i % 2 == 0
```

```
[42]: False
```

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, pero al no ser par no se imprime el valor

```
[37]: i < 10
```

```
[37]: True
```

```
[43]: i = i + 1  
      i % 2 == 0
```

```
[43]: True
```

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, al ser par se imprime el valor, que es 4.

```
[44]: i < 10
```

```
[44]: True
```

```
[45]: i = i + 1  
      i % 2 == 0
```

```
[45]: False
```

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, pero al no ser par no se imprime el valor

```
[47]: i < 10
```

```
[47]: True
```

```
[48]: i = i + 1  
      i % 2 == 0
```

```
[48]: True
```

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, al ser par se imprime el valor, que es 6.

```
[49]: i < 10
```

[49]: True

```
[50]: i = i + 1  
      i % 2 == 0
```

[50]: False

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, pero al no ser par no se imprime el valor

```
[51]: i < 10
```

[51]: True

```
[52]: i = i + 1  
      i % 2 == 0
```

[52]: True

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, al ser par se imprime el valor que es 8.

```
[53]: i < 10
```

[53]: True

```
[54]: i = i + 1  
      i % 2 == 0
```

[54]: False

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, pero al no ser par no se imprime el valor

```
[55]: i < 10
```

[55]: True

```
[56]: i = i + 1  
      i % 2 == 0
```

[56]: True

Como si es menor a 10, entra en el while, se le suma un valor, y se valora si es par, al ser par se imprime el valor de 10.

```
[57]: i < 10
```

[57]: False

Al no ser menor a 10, ya que en este punto $i = 10$, esto hace que no entre en el while y se termine el ciclo. Si contamos el número de veces que entró en el loop, nos damos cuenta que fueron 10.

1.4 3.

[Pensar como un computador] ¿Cuál es el resultado del siguiente código y cuántas veces se recorre el loop? $i = 0$ while $i > 10$: $i = i + 1$ if i print(i) En este código no alcanzamos a entrar al while, ya que i es igual a 0 desde un principio, y este no es mayor que 10, por lo tanto el loop no se recorre ni una sola vez.

1.4.1 Justificación

```
[58]: i = 0
      i > 10
```

[58]: False

Como no cumple con esta condición, nunca entra en el bloque.

1.5 4.

Escriba un programa que pida al usuario ingresar un número entero, y que imprima “par” si el número es par e “impar” si el número es impar. Agregue a su programa un código que genere una advertencia en caso de que el usuario ingrese algo diferente a un número entero: “Error. El usuario debe ingresar un número entero.” (Investigue por su cuenta cómo lograr dicha validación y la generación del mensaje.)

```
[91]: while True:
      try:
          numero = int(input("Escribe un número entero: "))
      except ValueError:
          print("Error. El usuario debe ingresar un número entero")
          continue
      break
  if numero%2 == 0:
      print("Es par")
  else:
      print("Es impar")
```

Escribe un número entero: 2.0

Error. El usuario debe ingresar un número entero

Escribe un número entero: 2.8
Error. El usuario debe ingresar un número entero
Escribe un número entero: 4
Es par

1.6 5.

Escriba un for loop que imprima todos los múltiplos de 3 desde 40 hasta 0 en orden decreciente. Esto es, 39, 36, 33,..., 3, 0.

```
[67]: n= range(40,-1,-1)
      for i in n:
          if i % 3 == 0:
              print(i)
```

39
36
33
30
27
24
21
18
15
12
9
6
3
0

1.7 6.

Escriba un loop que imprima todos los números entre 6 y 30 que no son divisibles por 2, 3 o 5.

```
[68]: n=range(6,31,1)
      for i in n:
          if i%2 != 0:
              if i%3 != 0:
                  if i%5 != 0:
                      print(i)
```

7
11
13
17
19
23
29

1.8 7.

Escriba un programa llamado “Adivine ni número”. El computador generará aleatoriamente un entero entre 1 y 100. El usuario digita un número y el computador responde “Menor” si el número aleatorio es menor que el escogido por el usuario, “Mayor” si el número aleatorio es mayor, y “¡Correcto!” si el usuario adivina el número. El jugador puede continuar ingresando números hasta que adivine correctamente.

Ejemplo: - El número aleatorio es 79. - El computador muestra el texto “Adivine el número entre 1 y 100:” y espera a que el usuario lo digite. - El usuario digita el número que está abajo en *itálicas*. - El computador devuelve uno de tres textos, según el caso: “Mayor”, “Menor”, o “¡Correcto!”.

Adivine el número entre 1 y 100: *40* Mayor Adivine el número entre 1 y 100: *70* Mayor Adivine el número entre 1 y 100: *80* Menor Adivine el número entre 1 y 100: *77* Mayor Adivine el número entre 1 y 100: *79* ¡Correcto!

¿Cómo generar números aleatorios en Python?

- Al comienzo de su programa escriba: `import random`
- Para generar un número aleatorio entre 1 y 100 escriba: `random.randint(1, 100)`

Pistas:

- Piense en qué estructuras de control le sirven para resolver el problema.
- ¿Cómo determina si el número es mayor, menor o correcto?
- ¿Cómo le da turnos adicionales al usuario para adivinar, dependiendo de si en el turno anterior adivinó o no?

1.8.1 Solución

```
[90]: import random
numero=random.randint(1,100)
x=int(input("Adivine el número entre 1 y 100: "))
while x != numero :
    if x > numero :
        print("Mayor")
    if x < numero :
        print("Menor")
    x=int(input("Adivine el número entre 1 y 100: "))
print("Correcto")
```

```
Adivine el número entre 1 y 100: 4
Menor
Adivine el número entre 1 y 100: 80
Menor
Adivine el número entre 1 y 100: 90
Menor
Adivine el número entre 1 y 100: 98
Mayor
Adivine el número entre 1 y 100: 95
Menor
Adivine el número entre 1 y 100: 96
```


Menor

Adivine el número entre 1 y 100: 97

Correcto