

PRÁCTICA Nº 2

Cátedra Programación II

Agosto 2023

1. Bucles For

EJERCICIO 1. Escriba un ciclo definido para imprimir por pantalla todos los números entre 10 y 20.

EJERCICIO 2. Escriba un programa que imprima por pantalla todas las fichas del dominó, una por línea y sin repetir.

EJERCICIO 3. Modifique el programa anterior para que pueda generar fichas de un juego que puede tener números de 0 a n .

EJERCICIO 4. Escriba una función que tome una cantidad m de valores que serán ingresados por el usuario y, a medida que se ingresa cada número, muestre el factorial del mismo. El valor de m es ingresado inicialmente por el usuario.

EJERCICIO 5. Usando la función dada como ejemplo en la presentación de *La Receta en Python* para convertir una temperatura de Fahrenheit a Celsius, genere una tabla de conversión de temperaturas, desde $0^{\circ}F$ hasta $120^{\circ}F$, de 10 en 10.

EJERCICIO 6. Escriba una función que reciba un número n como parámetro e imprima los primeros n números triangulares, junto con sus respectivos índices. El número triangular n se obtiene mediante la suma de los números naturales desde 1 hasta n . Es decir, si se piden los primeros 5 números triangulares, el programa debería imprimir:

```
1 1 - 1
2 2 - 3
3 3 - 6
4 4 - 10
5 5 - 15
```

Nota: hacerlo usando y sin usar la ecuación:

$$\sum_{i=1}^n i = \frac{n \times (n + 1)}{2}.$$

¿Qué versión realiza más operaciones?

2. Bucles While

EJERCICIO 7. Escriba una función que le pida al usuario que ingrese un número positivo. Si el usuario ingresa cualquier cosa que no sea lo pedido se le debe informar de su error mediante un mensaje y volver a pedirle el número, repitiendo este proceso hasta que ingrese lo pedido.

EJERCICIO 8. Escriba un programa que permita al usuario ingresar un conjunto de notas, preguntando a cada paso si desea ingresar más notas, e imprima el promedio correspondiente al finalizar la toma de datos.

EJERCICIO 9. Escriba un programa que pida dos números enteros. El programa pedirá de nuevo el segundo número mientras no sea mayor que el primero. El programa terminará escribiendo los dos números.

EJERCICIO 10. Escriba una función que reciba dos números como parámetros y devuelva cuántos múltiplos del primero hay que sean menores que el segundo.

- a) Implementarla utilizando un ciclo `for`, desde el primer número hasta el segundo.
- b) Implementarla utilizando un ciclo `while`, que multiplique el primer número hasta que sea mayor que el segundo.
- c) Comparar ambas implementaciones: ¿Cuál es más clara? ¿Cuál realiza menos operaciones?

EJERCICIO 11. Manejo de contraseñas

- a) Escriba un programa que contenga una contraseña inventada. El programa debe preguntarle al usuario la contraseña y no permitirle continuar hasta que la haya ingresado correctamente.
- b) Modifique el programa anterior para que solamente permita como máximo una cantidad fija de intentos.
- c) Modifique el programa anterior para que sea una función que devuelva si el usuario ingresó la contraseña correctamente o no, mediante un valor booleano (`True` o `False`).

EJERCICIO 12. Escriba una función que reciba un número natural e imprima todos los números primos que hay menores o iguales que ese número. Para esto se pide que:


- a) Defina una función `es_primo` que toma un número natural y verifica si es un número primo.
- b) Resuelva el problema usando la función definida en el punto anterior.

EJERCICIO 13. Potencias de dos

- a) Escriba una función `es_potencia_de_dos` que reciba como parámetro un número natural y devuelva `True` si el número es una potencia de 2 y `False` en caso contrario.
- b) Escriba una función que, dados dos números naturales pasados como parámetros, devuelva la suma de todas las potencias de 2 que hay en el rango formado por esos números (0 si no hay ninguna potencia de 2 entre los dos). Utilice la función `es_potencia_de_dos` descrita en el punto anterior.


3. Prueba tu suerte

Random, como su traducción lo indica, es un módulo en Python que nos permite ingresar “azar” o “suerte” a ciertos programas. Los módulos en Python nos permiten incluir definiciones que podemos usar en nuestro programa. En este caso, el módulo `random` es una librería de Python que contiene funciones para generar aleatorios. Para acceder a él se debe cargar al programa con la instrucción `import`.




```
1 import random
```

Ahora bien, esta definición, lo único que hace es ingresar el nombre del módulo `random`. Para luego, usar las funciones, se debe hacer:



```
1 import random
2 a = random.random() # acá se estaría invocando a la función random
```

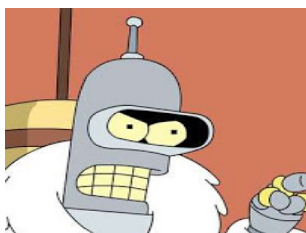
Para evitar esto, lo que vamos a hacer es:



```
1 from random import * # importa todas las funciones del módulo
2 a = random()          # luego usamos directamente la función
```

En el módulo `random` vamos a encontrar las siguientes funciones:

<code>random()</code>	Genera un número aleatorio entre 0 y 1
<code>randint(x,y)</code>	Genera un número entero aleatorio entre x e y, incluyendo ambos
<code>randrange(x)</code>	Genera un número entero aleatorio entre 0 y x-1



"Haré mis propios programas con juegos de azar..."

- 1) Simule lanzamientos de un dado. Muestre el resultado en cada intento y finalice cuando salga el número 6. También añada cuantas veces se lanzó el dado.
- 2) Simule n lanzamientos de dos dados, donde n es un valor que se debe pedir que se ingrese por teclado. Muestre cuántas veces los dados tuvieron el mismo resultado.
- 3) Simule n lanzamientos de un dado en un juego con las siguientes reglas: si sale 6 gana 4 pesos; si sale 3 gana 1 dólar; si sale 1 sigue jugando y si sale 2, 4 o 5 pierde 2 pesos. Muestre los valores que salen y el resultado final del juego.