Ejercicio 1

Escribir una clase en python con 2 métodos: **get_string** y **print_string**. get_string acepta una cadena ingresada por el usuario y print_string imprime la cadena en mayúsculas.

Ejercicio 2

Escribir una clase en python llamada **rectangulo** que contenga una base y una altura, y que contenga un método que devuelva el área del rectángulo.

Ejercicio 3

Escribir una clase en python llamada **circulo** que contenga un radio, con un método que devuelva el área y otro que devuelva el perímetro del circulo.

Ejercicio 4

Vamos a crear una clase llamada Persona. Sus atributos son: **nombre**, **edad** y **DNI**. Construye los siguientes métodos para la clase:

- Un constructor, donde los datos pueden estar vacíos.
- Los setters y getters para cada uno de los atributos. Hay que validar las entradas de datos.
- mostrar(): Muestra los datos de la persona.
- esMayorDeEdad(): Devuelve un valor lógico indicando si es mayor de edad.

Ejercicio 5

Desarrollar un programa que cargue los datos de un triángulo. Implementar una clase con los métodos para inicializar los atributos, imprimir el valor del lado con un tamaño mayor y el tipo de triángulo que es (equilátero, isósceles o escaleno).

Ejercicio 6

Escribe una clase de Python, y define dos métodos que devuelvan el área del cuadrado y el perímetro.

Ejercicio 7

Enunciado: clase CuentaBancaria

Se requiere un programa que modele una cuenta bancaria que posee los siguientes atributos:

- Nombres del titular.
- Apellidos del titular.
- Número de la cuenta bancaria.
- Tipo de cuenta: puede ser una cuenta de ahorros o una cuenta corriente.
- Saldo de la cuenta.

Se debe definir un constructor que inicialice los atributos de la clase. Cuando se crea una cuenta bancaria, su saldo inicial tiene un valor de cero.

En una determinada cuenta bancaria se puede:

- Imprimir por pantalla los valores de los atributos de una cuenta bancaria.
- Consultar el saldo de una cuenta bancaria.
- Consignar un determinado valor en la cuenta bancaria, actualizando el saldo correspondiente.
- Retirar un determinado valor de la cuenta bancaria, actualizando el saldo correspondiente. Es necesario tener en cuenta que no se puede realizar el retiro si el valor solicitado supera el saldo actual de la cuenta.

Ejercicio 8

1 - Crear una clase Cálculos con un constructor por defecto (sin parámetros) que permita realizar varios cálculos sobre números enteros.

2 - Crear un método llamado factorial() que permita calcular el factorial de un entero.

3 - Crear un método llamado suma() que permita calcular la suma de los primeros n enteros 1 + 2 + 3 + .. + n.

4 - Crea un método llamado testPrimo() en la clase Cálculo para comprobar si un número es primo. El resultado será True o False.

5 - Crear un método llamado testPrimos() que permita comprobar si dos números son primos entre sí.

6 - Cree un método tablaMult() que cree y muestre la tabla de multiplicación de un número entero dado. A continuación, cree un método allTablesMult() para mostrar todas las tablas de multiplicación de enteros 1, 2, 3, ..., 9.

7 - Crear un método listDiv() que obtenga todos los divisores de un entero dado en una nueva lista llamada Ldiv.

Ejercicio 9

Sigue los pasos:

- Crea una clase, Triangulo. Su método __init__() debe tomar como argumentos self, angle1, angle2 y angle3. Asegúrate de establecerlos apropiadamente en el cuerpo del método __init__().

- Crea una variable llamada numero_de_lados y ponla igual a 3.

- Crea un método llamado comprobar_angulos. La suma de los tres ángulos de un triángulo debe devolver True si la suma de ángulo1, ángulo2 y ángulo3 es igual a 180, y False en caso contrario.

- Crea una variable llamada mi_triangulo y hazla igual a una nueva instancia de tu clase Triangulo. Pásale tres ángulos que sumen 180 (por ejemplo, 90, 30, 60).
- Imprime mi_triangulo.numero_de_lados e imprime mi_triangulo.comprobar_angulos().

Ejercicio 10

Define una clase llamada Songs, que mostrará la letra de una canción. Su método __init__() debe tener dos argumentos: self y lyrics. lyrics es una lista. Dentro de tu clase crea un método llamado sing_me_a_song que imprima cada elemento de la letra en su propia línea. Define una variable:

```
happy_bday = Song(["Que Dios te bendiga, ",

        "Que te acompañe el sol,",

        "¡Feliz cumpleaños a ti!"])
```

Llama al método sing_me_a_song sobre esta variable.