Ejercicios integradores para revisar en la reunión sincrónica 7

1. Escribir una función que calcule el máximo común divisor entre dos números.
2. Escribir una función que calcule el mínimo común múltiplo entre dos números
3. Escribir un programa que reciba una cadena de caracteres y devuelva un diccionario con cada palabra que contiene y la cantidad de veces que aparece (frecuencia).
4. Escribir una función que reciba una cadena de caracteres y devuelva un diccionario con cada palabra que contiene y la cantidad de veces que aparece (frecuencia). Escribir otra función que reciba el diccionario generado con la función anterior y devuelva una tupla con la palabra más repetida y su frecuencia.
5. Sabiendo que ValueError es la excepción que se lanza cuando no podemos convertir una cadena de texto en su valor numérico, escriba una función get_int() que lea un valor entero del usuario y lo devuelva, iterando mientras el valor no sea correcto. Intente resolver el ejercicio tanto de manera iterativa como recursiva.
6. Crear una clase llamada Persona. Sus atributos son: nombre, edad y DNI. Construya 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.
   - es_mayor_de_edad(): Devuelve un valor lógico indicando si es mayor de edad.
7. Crea una clase llamada Cuenta que tendrá los siguientes atributos: titular (que es una persona) y cantidad (puede tener decimales). El titular será obligatorio y la cantidad es opcional. Crear los siguientes métodos para la clase:
   - Un constructor, donde los datos pueden estar vacíos.
   - Los setters y getters para cada uno de los atributos. El atributo no se puede modificar directamente, sólo ingresando o retirando dinero.
   - mostrar(): Muestra los datos de la cuenta.
   - ingresar(cantidad): se ingresa una cantidad a la cuenta, si la cantidad introducida es negativa, no se hará nada.
   - retirar(cantidad): se retira una cantidad a la cuenta. La cuenta puede estar en números rojos.
8. Vamos a definir ahora una “Cuenta Joven”, para ello vamos a crear una nueva clase CuantaJoven que deriva de la clase creada en el punto 7. Cuando se crea esta nueva clase, además del titular y la cantidad se debe guardar una bonificación que estará expresada en tanto por ciento. Crear los siguientes métodos para la clase:
   - Un constructor.
   - Los setters y getters para el nuevo atributo.
   - En esta ocasión los titulares de este tipo de cuenta tienen que ser mayor de edad, por lo tanto hay que crear un método es_titular_valido() que devuelve verdadero si el titular es mayor de edad pero menor de 25 años y falso en caso contrario.
   - Además, la retirada de dinero sólo se podrá hacer si el titular es válido.
   - El método mostrar() debe devolver el mensaje de “Cuenta Joven” y la bonificación de la cuenta.