1.- Realiza un programa en java que invoque los métodos descritos a continuación. El programa debe pedir un número usando el método dameNumero, y para ese número mostrará la serie ascendente e imprimirá la suma de sus términos. A continuación, pedirá un segundo número, mostrando la serie descendente que le corresponde e imprimirá la suma de sus términos.

- Método **dameNumero,** recibe como parámetro un objeto Scanner y una cadena. Desde dentro de este método se pide al usuario un número mayor que 4 e impar (usando la cadena recibida para escribir el mensaje). El método comprobará que el número sea mayor que 4 e impar, y si no lo cumple, seguirá pidiéndolo). El método devuelve el número introducido por el usuario al programa principal**.**
- Método **mostrarSerieAscendente,** recibe como parámetro un entero, con este número escribirá, dentro del método, una secuencia desde el número 1 al número introducido, repitiéndose el número tantas veces como indica su valor. Devolverá la suma de sus términos.
- Método **mostrarSerieDescendente,** recibe como parámetro un entero, con este número escribirá, dentro del método, una secuencia desde el número n al 1, repitiéndose el número tantas veces como indica su valor. Devolverá la suma de sus términos.

***Ejemplo*:**

Dame un número mayor que 4 e impar: 3
Dame un número mayor que 4 e impar: 6
Dame un número mayor que 4 e impar: 5

SERIE ASCENDENTE

122333444455555
Sus valores suman: 55

Dame un número mayor que 4 e impar: 7

SERIE DESCENDENTE

7777777666666555554444333221
Sus valores suman: 140

2.- Se pide desarrollar los **métodos** que se describen a continuación.

- **dameFrase**: recibe como parámetro un objeto Scanner y una cadena. La cadena representa el mensaje en el que se pide al usuario una cadena que sea distinta de vacía (el método lo comprobará y si no lo cumple sigue pidiéndola). El método devuelve la cadena introducida por el usuario.
- **posPrimeraVocal**: recibe una cadena (String) y devuelve la posición de la primera vocal, la posición empieza en 1 cuando se la mostramos al usuario**.**
- **posUltimaVocal**: recibe una cadena (String) y devuelve la posición de la última vocal, la posición empieza en 1 cuando se la mostramos al usuario.
- **esVocal**: recibe un carácter y devuelve true si es una vocal y false si no lo es. Se usará en los dos métodos anteriores. Tendrás en cuenta solo las vocales sin acentuar, tanto mayúsculas como minúsculas.
- **darVuelta**: recibe una cadena (String) y devuelve otra cadena de manera que en el programa principal tengamos una copia de la cadena original, pero dada la vuelta (**nota**: no se puede usar ningún método de Java que realice esta operación directamente).

Realizarás un programa en java que pruebe todos métodos secuencialmente

Ejemplo:

Entrada:

Dame una frase que no sea la cadena vacía

"El que fue a Sevilla perdio su silla"

Salida:

La primera vocal en la cadena "El que fue a sevilla perdio su silla" está en la posicion 1

La última vocal en la cadena "El que fue a sevilla perdio su silla" está en la posicion 36

La cadena dada la vuelta es "allis us oidrep allives a euf euq lE"

3.- Dos números son **gemelos** si son primos (el número uno no se considera primo) y, además se diferencian en dos unidades.

Ejemplos de parejas de números gemelos:

3 y 5
5 y 7
17 y 19
29 y 31
101 y 103

Se pide:

- Implementa una función que siga la siguiente cabecera para determinar si un número es primo:

  public static boolean esPrimo(**int** num);
- Implementa una función que siga la siguiente cabecera para determinar si 2 números son gemelos, haciendo uso de la función anterior:

  public static boolean sonGemelos(**int** num1, **int** num2);
- En el programa principal se pedirán parejas de números que pasaremos como parámetros al método **sonGemelos**. El usuario indicará de alguna manera que no desea introducir ninguno más (se deja a tu elección la condición para parar de pedir números).

4.- Escribe una **clase** llamada **MiDoble** que tenga como atributo un número decimal en punto flotante (**double**), un constructor por defecto y otro que reciba el valor del número. Completa con los gettter, setter y toString() necesarios.

Implementa también dos métodos en la clase para obtener la parte entera y la parte decimal del número en punto flotante (**double**) que representa la clase. La definición de esos métodos es como sigue:

- **int** getParteEntera**()**
- **int** getParteDecimal**()**

Puedes desarrollar los métodos de la forma que creas más conveniente.

El programa principal pedirá al usuario que introduzca un número por teclado y posteriormente que elija qué operación de las dos disponibles (obtener la parte entera o la parte decimal del número haciendo uso de los métodos de la clase) desea realizar sobre el número introducido. Se seguirá realizando la operación pedida mientras el número sea mayor que cero.
Además, cuando se ejecute la sentencia system.out.println() mandando un objeto de tipo MiDoble, se imprimirá el valor del número de la siguiente forma (48,25 será el valor del atributo de la clase) .

MiDoble -> numero=48.25

Justo antes de acabar el programa, se mostrará el número de objetos de tipo MiDoble que se ha creado, sin realizar la cuenta en el programa principal (se lleva la cuenta en la propia clase).