Entrega: carpeta comprimida con nombre y apellidos que contenga este archivo, los pseudocódigos (en formato psc) y los ficheros fuente de las clases.

1. Realiza el pseudocódigo y la codificación en lenguaje de programación Java de un algoritmo que solicite por teclado números enteros hasta que se introduzca un número no positivo (0 o negativo) y por cada número positivo introducido muestre por pantalla la suma de sus dígitos. Al finalizar mostrará la cantidad de números positivos introducidos.

Ejemplo:

Entrada: 16345 Salida: SUMA DE DÍGITOS: 19
Entrada: 1357 Salida: SUMA DE DÍGITOS: 16

Entrada: 0 Salida: Números positivos introducidos: 2

2. Escribe un programa en Java que solicite un mensaje por teclado y muestre por pantalla los **hashtags** incluidos en el mismo. Un hashtag es una palabra precedida del carácter #.

## Ejemplo:

Entrada: Entra en erupción el volcán #Merapi en #Java #Indonesia Salida: #Merapi

#Java #Indonesia

3. Implementa en lenguaje de programación Java la clase **NumeroComplejo** presentada en el siguiente diagrama de clases UML.

Se define un número complejo z como un par ordenado de números reales: z = (a,b).

También se puede representar de forma binómica: z = a + bi

a es la parte real, b es la parte imaginaria

Operaciones con números complejos:

$$z1=(a, b)$$
  $z2=(c, d)$ 

## Suma:

$$z1 + z2 = (a, b) + (c, d) = (a+c, b+d)$$

## Resta:

$$z1 - z2 = (a, b) - (c, d) = (a-c, b-d)$$

## NumeroComplejo

-parteReal: float -parteImaginaria: float

+NumeroComplejo()

+NumeroComplejo(parteReal: float, parteImaginaria: float)

+getParteReal(): float +getParteImaginaria(): float

+setParteReal(parteReal: float): void

+setPartelmaginaria(partelmaginaria: float): void +sumar(z: NumeroComplejo): NumeroComplejo +restar(z: NumeroComplejo): NumeroComplejo

+multiplicar(z: NumeroComplejo): NumeroComplejo

+conjugado(): NumeroComplejo

+toString(): String

<u>+sumar(z1: NumeroComplejo, z2: numeroComplejo): NumeroComplejo</u> +printFormaBinomica(): String

**Producto**: z1 \* z2 = (a, b) \* (c, d) = (a\*c-b\*d, a\*d+b\*c) **Conjugado**:  $\overline{z1} = (a, -b)$ 

El método toString() devuelve el número complejo en el siguiente formato: (parteReal, partelmaginaria)

El método estático **sumar** recibe dos números complejos y devuelve la suma de los dos números complejos recibidos.

El método printFormaBinomica() devuelve el número complejo en el formato: parteReal + parteImaginaria i

Número complejo 1: z1 = (1, 5) Número complejo 2: z2 = (2, -3)

z1.sumar(z2) => (3, 2) z1.restar(z2) => (-1, 8) z1.multiplicar(z2) => (17, 7) z1.conjugado() => (1, -5) z1.toString() => (1.0, 5.0) z2.toString() => (2.0, -3.0)

z1.printFormaBinomica() =>  $\frac{1.0 + 5.0i}{2.0 - 3.0i}$  z2.printFormaBinomica() =>  $\frac{2.0 - 3.0i}{2.0 - 3.0i}$ 

4. Realiza el pseudocódigo y la codificación en lenguaje de codificación Java de un algoritmo que llene un vector (array unidimensional) de enteros con 10 números aleatorios comprendidos entre 0 y 100. A continuación deberá calcular el rango y lo mostrará por pantalla. Para calcular el rango procederemos de la siguiente forma: buscaremos el valor más alto y el más bajo, restaremos al valor más alto el valor más bajo y el resultado es el rango.

Ejemplo: 33 20 19 28 2 34 46 27 60 95

<u>Valor más alto</u>: 95 <u>Valor más bajo</u>: 2 <u>Rango</u>: 95 - 2 = 93