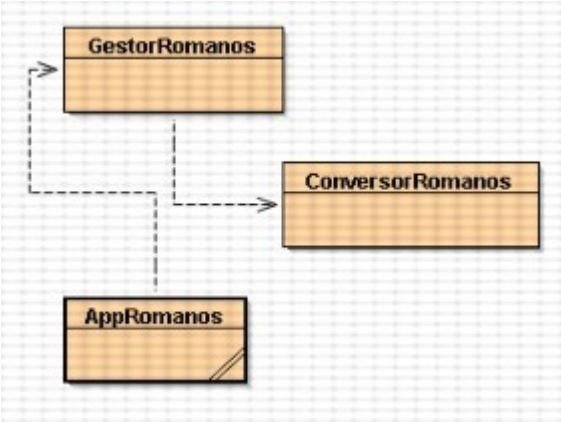
Ejercicios adicionales UT6 (V) –
Colecciones de tamaño flexible: HashSet, HashMap.

Ejercicio 10.

a) Queremos diseñar una aplicación que genere una lista de números arábigos a partir de sus correspondientes números romanos. La lista además estará en orden ascendente según el valor del número arábigo.

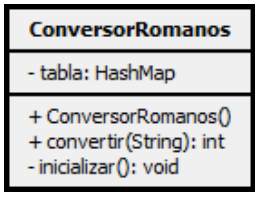
El proyecto constará de las clases indicadas en el diagrama:



La clase ConversorRomanos convierte un nº romano (que suponemos correcto) en un nº arábigo.

La clase GestorRomanos mantiene la lista de números arábigos y sus números romanos asociados.

La clase AppRomanos es la clase que contiene el main() y la que arranca la aplicación.



Un objeto de esta clase convierte un nº romano en un nº arábigo.

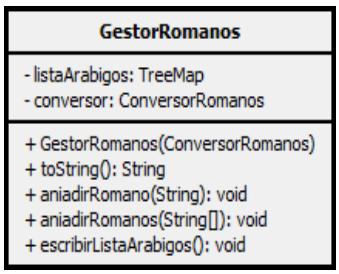
Consideraremos la notación romana antigua en la que puede haber hasta 4 símbolos iguales. Por ejemplo, el 4 es el nº romano IIII y no el IV, el 9 es VIIII y no el IX.

Para hacer la conversión la clase utiliza un objeto HashMap en el que se establece la asociación:

1000	500	100	50	10	5	1	***valores***
M	D	C	L	X	V	I	***claves***

Las claves en el map **no** son de tipo String.

El constructor crea el *hash* adecuadamente y llama al método privado inicializar() para asignarle sus valores iniciales.



La clase GestorRomanos representa a través de un *map* la serie de números arábigos estableciendo una asociación entre la clave (el nº arábigo) y su nº romano asociado.

Puesto que la serie de números arábigos hay que mostrarla al final ordenada (por el valor arábigo) en lugar de utilizar una clase HashMap utilizaremos una clase TreeMap que permite que las claves (en nuestro caso números arábigos) se guarden en orden.

Observa que el constructor recibe como parámetro un objeto `ConversorRomanos`.

public void añadirRomano(String romano) – añade al *map* el arábigo correspondiente al nº romano pasado como parámetro.

public void añadirRomanos(String[] romanos) – añade todos los romanos indicados en el array al *map*

public void escribirListaArabigos() –muestra el *map* tal como indica la figura

Ejemplo

Si se hace: añadirRomanos(new String[]({"D","XII","III","MDII","CII"});

la salida mostrada será:

La clase `AppRomanos` contiene el main(). En esta clase:

- ✗ comprueba que el nº de argumentos no es 0, si es así muestra el mensaje correspondiente y termina el programa
  Error, Sintaxis: java AppRomanos <romano1> <romano2> <romano3> ….
- ✗ pasa los strings que representan números romanos al gestor que previamente has creado y muestra la lista gráficamente.

b)

Desde BlueJ:

- Añade las clases `GestorRomanos` y `ConversorRomanos` al paquete `pkgromanos`
- La clase `AppRomanos` se queda en el paquete por defecto
- Haz los cambios necesarios y verifica que todo funciona bien
- Crea un *jar* (*romanos.jar)* para distribuir el ejecutable y colócalo en una carpeta diferente a la del proyecto
- Comprueba que lo puedes ejecutar haciendo una llamada al *jar* desde la línea de comandos del DOS (ten en cuenta que la clase que contiene el main() recibe argumentos). Anota cómo has hecho esta llamada.

c)

- Elimina el fichero r*omanos.jar* anterior
- Sal a línea de comandos y:
  - ✗ sitúate en el directorio base de tu proyecto (la carpeta raíz del proyecto)
  - ✗ ejecuta la aplicación y anota el comando que has realizado
  - ✗ crea ahora el fichero *romanos.jar* ejecutable y anota el comando que has realizado
  - ✗ ejecuta el fichero *romanos.jar* y comprueba que todo funciona bien. Anota el comando realizado.