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

Ejercicio 5.

Completa la clase UtilsString . Solo contiene utilidades, métodos estáticos, para trabajar con cadenas de caracteres.

- public static ..... getConjuntoCaracteres(String cadena) - Dada una cadena obtiene y devuelve su conjunto de caracteres. Importa el orden en el conjunto
- public static ..... getConjuntoDigitos(String cadena) - Dada una cadena obtiene su conjunto de caracteres numéricos. No importa el orden en el conjunto. Usa algún método de la clase Character para comprobar si un carácter es numérico o no
- public static ..... getConjuntoMayusculas(String cadena) - Dada una cadena obtiene su conjunto de letras mayúsculas en el orden en el que aparecen en la cadena (si la cadena es "*PruEbA*" devuelve el conjunto *[PEA]*)

Completa ahora la clase ListaNombres.

La colección que almacena la clase guarda una serie de nombres en todo momento ordenados. No utilizaremos el método sort() de la clase Collections sino que en el método añadirNombre() lo que haremos será obtener la posición en la que hay que añadir el nuevo nombre a la lista de forma que quede en orden. Esta posición la obtiene el método privado buscarPosicion(). Una vez obtenida la posición añadiremos el nombre .

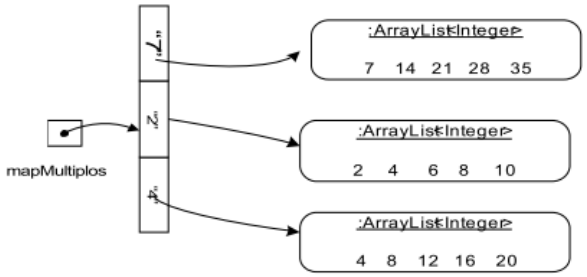
- public ..... obtenerConjuntoDeCaracteres(int pos) - Obtiene el conjunto de caracteres del nombre de la posición indicada. Lanza la excepción IllegalArgumentException si el argumento recibido es incorrecto
- public .....obtenerConjuntoDeDigitos(int pos) - Obtiene el conjunto de dígitos numéricos del nombre de la posición indicada. Lanza la excepción IllegalArgumentException si el argumento recibido es incorrecto
- public ..... obtenerConjuntoMayusculas(int pos) - Obtiene el conjunto de mayúsculas del nombre de la posición indicad. Lanza la excepción IllegalArgumentException si el argumento recibido es incorrecto
- public String toString() - Obtiene una representación textual de la lista tal como indica la figura (nombre con cada uno de sus conjuntos asociados)

```
Nombres y sus conjunto asociados (Caracteres  Dígitos  Mayúsculas)
LUis   [L, U, i, s]   []   [L, U]
Luisa   [L, a, i, s, u]   []   [L]
PRUEba3   [3, E, P, R, U, a, b]   [3]   [P, R, U, E]
ana   [a, n]   []   []
eJEMPLO_14   [1, 4, E, J, L, M, O, P, _, e]   [1, 4]   [J, E, M, P, L, O]
ejemplo1   [1, e, j, l, m, o, p]   [1]   []
pepe   [e, p]   []   []
prueba2.2   [., 2, a, b, e, p, r, u]   [2]   []
```

Para probar la lista puedes llamar al método leerDeFichero() que incluye la clase ListaNombres.

Ejercicio 6 .

La clase SerieMultiplos guarda en un HashMap la relación de múltiplos correspondientes a un determinado nº. Las claves en el map serán String y los valores ArrayList de tipo Integer.



Define en la clase un atributo *mapMultiplos* de estas características.

Los métodos que se pueden invocar sobre un objeto de esta clase serán, además del constructor:

- public void añadirEntrada(int num) – añade una nueva entrada al *map*. Recuerda que las claves en este map son String (si el parámetro *num* = 5 , la clave será "5"). La colección ArrayList asociada (el valor correspondiente a la clave) se obtiene con ayuda del método generarMultiplos().
- private ArrayList<Integer> generarMultiplos(int num) – dada un nº genera una colección de 10 múltiplos
- public ArrayList<Integer> obtenerMultiplosDe(int num) – devuelve los múltiplos del número pasado como parámetro
- public void escribirMap() - visualiza el *map*, cada clave junto con sus múltiplos asociados. Los múltiplos se muestran llamando al método escribirValor(). Usa el conjunto de claves.
- private void escribirValor(ArrayList<Integer> lista) - muestra la colección recibida como parámetro
- public int borrarMultiplo(int multi) - borrar del *map* el múltiplo indicado y devuelve el total de múltiplos borrados. Con Map.Entry y un iterador.

Ejercicio 7.

Abre el proyecto que se te proporciona y completa la clase FrecuenciaNumeros. Esta clase define un atributo *frecuencias* implementado como un HashMap en el que las claves son de tipo String y los valores asociados objetos Contador.
La clase registra las apariciones de una serie de números leídos desde un fichero de texto.

Complétala de la siguiente forma:

- define el atributo *frecuencias*
- completa el constructor
- public void añadirNumero(int numero) – contabiliza un nuevo nº en el map
- public void listarFrecuencias() - Lista cada número aparecido y su frecuencia. Utiliza el conjunto de entradas (Map.Entry)
- public void listarNumeros() - Muestra cuántos números diferentes han aparecido y la relación de todos ellos. Utiliza la "*vista*" sobre las claves (el conjunto de claves) y un for mejorado

```
Nº 34   Frecuencia 3
Nº 78   Frecuencia 1
Nº 1    Frecuencia 2
Nº 2    Frecuencia 1
Nº 3    Frecuencia 5
Nº 5    Frecuencia 2
Nº 6    Frecuencia 2
Nº 9    Frecuencia 2
Nº 20   Frecuencia 1
Nº 54   Frecuencia 1
Nº 21   Frecuencia 2
```

```
Números aparecidos 11
34      78      1       2       3       5       6       9       20      54      21
```