Annex I.- Character i String

Quan treballem amb caràcters es fan necessaries algunes funcions de comprobació i traslació. Les classes **Character** i **String** disposen d'una serie de propietats i mètodes que ens poden facilitar molt el treball amb caràcters o cadenes de caràcters.

Character

Podem representar un caràcter de dues formas diferents:

- char c;
- Character c;

La primera instrucció declara una variable del tipus primitiu char, i per tant no té propietats ni mètodes asociats.
La segona instrucció declara una variable de tipus Character, que és una classe i per tant té propietats i mètodes que podem utilitzar.

La majoria de mètodes de la classe Character són estàtics i per tant no és necessari crear un objecte d'aquests tipus. Podem executar els seus mètodes posant el nom de la classe "punt" i el nom del mètode. Per example: Character.isDigit('0');

Mètodes estàtics més usats

- boolean Character.**isLowerCase**(char c). Comprova si el caràcter està en minúscules.
- boolean Character.**isUpperCase**(char c). Comprova si el caràcter està en majúscules.
- boolean Character.**isDigit**(char c). Comprova si el caràcter és un dígit.
- boolean Character.**isLetter**(char c). Comprova si el caràcter és una lletra.
- boolean Character.**isSpaceChar**(char c). Comprova si el caràcter és un espai en blanc.
- char Character.**toLowerCase**(char c). Converteix el caràcter a minúscules.
- char Character.**toUpperCase**(char c). Converteix el caràcter a majúsucules.
- Character Character.**valueOf**(char c). Retorna una instància de Character representant el caràcter especificat.
- int Character.**getNumericValue**(char c). Retorna el valor numèric del caràcter pasat.
- String Character.**toString**(char c). Converteix el caràcter a un String.

Podem crear un objecte de tipus Character de la següent forma:
Character c = new Character('G');

String

La classe String permet representar cadenes de caràcters, per tant és ideal per enmagatzemar paraules, frases o textos.
Podem representar una cadena de caràcters de les següents formes:

- String s = “Hola”;
- String s = new String(“Hola”);

Encara que sembla el mateix, la forma de crear-se l'objecte no és igual. La segona forma sempre crea un nou objecte en el heap, una zona de memòria especial per a les variables dinàmiques, mentre que la primera forma pot crear o no un nou objecte (si no el crea el reutilitza del String Pool, una memòria caché dissenyada per a reciclar Strings).

Mètodes més usats

- char **charAt**(int i). Retorna el caràcter que ocupa la posició i.
- int **compareTo**(String anotherString). Compara lexicogràficament (alfabéticament) l'String actual amb l'String passat com a paràmetre. Sol utilitzar-se per a ordenar cadenes de caràcters.
  - Retorna < 0, si la cadena que invoca al mètode es menor lexicogràficament que la cadena rebuda com a paràmetre.
  - Reorna == 0 si les dos cadenes són iguals lexicogràficament.
  - Retorna > 0, si la cadena que invoca al mètode és major lexicogràficament que la cadena rebuda com a paràmetre.
- String **concat**(String s). Concatena la cadena actual amb la cadena rebuda com a paràmetre.
- boolean **contains**(CharSequence s). Comprova si la cadena actual conté la cadena rebuda com a paràmetre.
- boolean **equals**(Object o). Comprova si el valor de l'String és igual al del objecte o.
- booean **equalsIgnoreCase**(String s). Comprova si el valor de l'String és igual, sense tindre en compte les majúscules, a l'String rebut com a paràmetre.
- byte[] **getBytes**(Charset charset). Codifica l'String en una seqüència de bytes utilitzant el sistema de codificació de caràctes rebut com a paràmetre.
- int **hashCode**(). Retorna un hash code per a l'String.
- int **indexOf**(…). Múltiples variants. Retorna l'índex de la primera ocurrència trobada del caràcter/String en el valor rebut com a paràmetre. Si no es troba cap coincidència retorna -1.
- int **length**(). Retorna la longitud de l'String.
- String **replace**(char oldChar, char newChar). Canvia totes les ocurrències del caràcter oldChar pel caràcter newChar.
- String **replaceFirst**(String regex, String replacement). Canvia la primera ocurrència que coincidisca amb l'expressió regular regex pel substring replacement.
- String **replace**(CharSequence target, CharSequence replacement). Canvia cada substring que coincidisca amb el target per la seqüència de caracters indicada per replacement.
- String[] **split**(String regex). Parteix l'String en varies parts segons les regles indicades en la expressió regular regex.
- String **substring**(int beginIndex). Retorna un String que comença desde el caràcter número beginIndex fins el final.
- String **substring**(int beginIndex, int endIndex). Retorna un String que comença desde el caràcter

número beginIndex fins el caràcter número endIndex.

- char[] **toCharArray**(). Converteix l'String en un array de caràctes.
- String **toLowerCase**(). Converteix l'String a minúscules.
- String **toUpperCase**(). Coverteix l'String a majúscules.
- String **trim**(). Retorna el mateix String però sense els espais en blanc de l'inici i del final.
- String String.**ValueOf**(…). Múltiples opcions. Retorna com a String el tipus de dades passat com a paràmetre.