Tema 4: EL TIPO DE DATOS CARÁCTER y EL TIPO DE DATOS REAL

4.1.- El tipo de datos carácter (char)

4.1.1.- Dominio de valores:

Es el conjunto de caracteres: letras, cifras, signos de puntuación, etc.

{A,B,C,...,Z,a,b,...,z,á,..,Á,...Ú,ç,Ç,..ñ,Ñ,0,1,...,9,=,+,?,*,#,...}

4.1.2.- Representación interna

Para representar cada carácter, basta con codificarlo con un determinado código binario. Lo importante es que esa codificación sea conocida por los distintos ordenadores, lenguajes de programación, sistemas operativos, aplicaciones, etc. y así puedan entenderse, esto es, hay que definir un estándar de codificación que sea aceptado y conocido por todos. Habitualmente, en estos casos, lo que suele ocurrir es que no hay un único estándar. Uno de los más aceptados es el llamado ASCII que asigna a cada carácter un número binario de 7 bits (es decir, un número decimal entre 0 y 127). Mostramos en la siguiente tabla algunos de los caracteres llamados "imprimibles" (los que pueden visualizarse en pantalla o imprimirse). El resto son caracteres de control y otros especiales.

Carácter	Número binario (7 bits)	Número decimal
Espacio	0100000	32
!	0100001	33
.....	.....	.....
0	0110000	48
1	0110001	49
.....		
9	0111001	57
:	0111010	58
.....	.....	.....
A	1000001	65
B	1000010	66
.....	.....	.....
Z	1011010	90
[	1011011	91
.....	.....	.....
a	1100001	97
b	1100010	98
.....	.....	.....
z	1111010	122

Debido a que con tan solo 7 bits no es posible codificar todos los caracteres de todos los alfabetos del mundo, comenzaron a surgir diferentes extensiones al estándar ASCII para poder trabajar con distintos alfabetos.

En 1991 se definió un nuevo estándar llamado UNICODE el cual asigna un código a más de 50000 símbolos de todos los alfabetos europeos, chinos, japoneses, coreanos y otros más. Los símbolos del 0 al 127 del estándar UNICODE coinciden con los del estándar ASCII.

4.1.3.- Representación externa y definición en Java

En Java el tipo `char` es un carácter de 16 bits, que codifica los caracteres siguiendo el estándar UNICODE.
Las constantes de tipo carácter en Java aparecen entre comillas simples.
Ejemplos: 'A' 'a' ';'
'\n' (que es el carácter de fin de línea)
'\t' (que es el carácter tabulador)
'\u0108' (que es el carácter UNICODE 0108 en base hexadecimal, que corresponde al 264 en base decimal)

4.1.4.- Operaciones

De comparación

(<, <=, >, >=, == (igual), != (distinto)) cuyos resultados son valores booleanos o lógicos.
El resultado de una comparación de dos caracteres es el resultado de la comparación de los números ASCII correspondientes a cada carácter.

Ejemplo: 'A'<='B' → verdadero ya que 65 <= 66 es verdadero
'C'>'F' → falso ya que 67>70 es falso

```
char c;
char d;
c='F';
d='G';
if (c!=d) System.out.println("Son caracteres distintos");
else System.out.println("Son caracteres iguales");
if (c<d) System.out.println("c anterior a d");
else System.out.println("d anterior a c");
```

Es importante señalar que el orden ASCII respeta el orden alfabético pero con algunas excepciones:
Cualquier letra minúscula es mayor que cualquier mayúscula. Ej: 'Z' > 'b' es falso.
La ñ y la Ñ son caracteres especiales y no se encuentran entre la n y la o (N y la O).

4.2.- El tipo de datos cadena de caracteres (String)

Aunque ciertamente no se trate de un tipo de datos de manera estricta, sino más bien de una clase, indicamos brevemente aquí cómo definir variables de tipo String, cómo son las constantes y cómo se puede realizar entrada/salida de Strings.

Definición de variables de tipo String:

```
String nombre;
```

Constantes de tipo String

Se ponen entre comillas. Por ejemplo: `"Hola"`

Entrada/Salida de Strings

Entrada: Con la instrucción `readLine` se puede leer un String y asignarlo a una variable de tipo String. Para ello se necesita definir un objeto de la clase `IODialog.`

Salida: Se consigue con las instrucciones `print y println,` que ya conocíamos. Se necesita definir un objeto de la clase `IODialog,` o también se puede realizar directamente usando `System.out`