

Auxiliar 5

Strings y Archivos

Los String



Un String es una “cadena” de caracteres. En JAVA se representa por un objeto que no se puede modificar.

Se declara de la siguiente forma:

```
String s="c0c1c2...cn"
```

Donde cada “ci”, con i de 0 a n, es un caracter.

Sobre los String se pueden realizar varias operaciones, en forma de métodos, que tienen como resultado otro String o algún parámetro.

Métodos



Los métodos más ocupados son los siguientes.

```
s.length();      -> retorna un int con el largo del String
s.equals(r);     -> retorna true si s es igual a r
s.compareTo(r);  -> retorna un int mayor que 0 si s>r
s.charAt(i);     -> retorna el char en el indice i
s.indexOf(r);    -> retorna el indice del string r en s
s.concat(r);     -> retorna un String s concatenado con r
s+r;            -> IDEM
s.substring(i,j); -> retorna la subcadena desde i hasta j
```

Es importante notar que ningún método actúa sobre el String mismo, si no que retorna un String u otro valor.

Toques



Vamos a realizar una solución no autorizada de la tarea.

```
static public void main(String args[]){
    String a=U.readLine("Entero de 4 digitos");
    String b=U.readLine("Entero de 4 digitos");
    String c="0123456789",r="";
    for(int i=0;i<c.length();++i)
        if(a.indexOf(c.charAt(i))>=0 && b.indexOf(c.charAt(i))>=0)
            r+=c.charAt(i);
    U.println("Tienen "+r.length()+" toques, "+r);
}
```

Claramente, es mucho más sencillo que con enteros.

Problema



El siguiente ejercicio se trata de encontrar las palabras capicúa. Las palabras capicúa, son aquellas que son iguales con sus caracteres en el orden inverso. Ejemplo: oso, ana, anitalavalatina.

Para ello escribirán un método que “de vuelta” un String. El programa será de la siguiente forma.

```
static public void main(String args[]){  
    String a=U.readLine("Introduzca palabra: ");  
    String b=volcar(a);  
    if(a.equals(b))  
        U.println("es capicua");  
    else  
        U.println("no es capicua");  
}
```

Solución

```
static public String volcar(String x){  
    String r="";  
    for(int i=x.length()-1;i>=0;--i)  
        r+=x.charAt(i);  
    return r;  
}
```

Esta función puede resultarnos muy útil más adelante.

En efecto, en el siguiente problema la utilizaremos.

Archivos



Es claro que hasta el momento en JAVA, lo que el programa realiza se “muere” al terminar la ejecución.

Por supuesto hay veces en que necesitaremos que la información perdure en el tiempo.

Para este proposito, JAVA dispone de clases que permite manejar archivos de texto plano (.txt), leyendolos y escribiendo en ellos.

Declararemos una variable que representa un archivo de lectura en JAVA.

```
BufferedReader br=new BufferedReader(new FileReader("archivo.txt"));
```

Para los archivos de exritura, la sintaxis es:

```
PrintWriter pw=new PrintWriter(new FileWriter("archivo.txt"));
```

Archivos



Las funciones que usaremos con archivos son las siguientes.

```
br.readLine();    -> lee una linea del archivo (hasta enter)
br.close();       -> cierra el archivo, no se puede ocupar más

pw.print("String"); ->Escribe "String" en un archivo
pw.close();       -> cierra el archivo, no se puede ocupar más
```

Cuando leemos un archivo con `readLine()`, recibiremos Strings que corresponderán a las lineas consecutivas de este. Es decir, la primera vez que lo llamamos leemos la primera, la segunda vez la segunda, etc.

La última línea corresponde a un simbolo en JAVA llamado **null**

Problema



Escriba en pantalla todas las líneas desde un archivo llamado alumnos.

Deben fijarse en que en algún momento termine el `while` que necesitarán, pues no saben el número de líneas del archivo.

Deben escribir el número de la línea al lado izquierdo de cada una.

Para utilizar `BufferedReader` y `PrintWriter` debe importar `java.io.*`

Solución

```
import java.io.*;
class Auxiliar{
    static public void main(String args[])throws IOException{
        BufferedReader br=new BufferedReader(new
FileReader("alumnos.txt");
        String linea=br.readLine();
        int i=0;
        while(linea!=null)
            U.println((i++)+" "+linea);

        br.close();
    }
}
```

Ejercicio



Escriba en un archivo .java un método con el siguiente encabezado.

```
static public void volcador(String in,String out)throws IOException
```

El String in es el nombre de un archivo de entrada, y el String out es el nombre de un archivo que es el resultado de volcar todas las líneas de in.

FIN

Profesor: Andrés Muñoz

Auxiliares: Oscar Alvarez
Pedro Valencia

presentación realizada con *OpenOffice.org Impress*