Práctica 2: Distancia (continuación)

1. Objetivos

- Practicar la programación modular
- Practicar el uso de la sentencia if-else
- Practicar el manejo de Strings
- Aprender a ejecutar un programa java desde la línea de comandos pasando parámetros a la ejecución.

2. Introducción

En la práctica 1 hemos resuelto el problema de calcular la distancia entre dos puntos, dadas sus coordenadas. Para ello hemos implementado métodos de utilidad en la clase Coordenadas del proyecto útiles, y los hemos utilizado en la clase Distancia del proyecto practica1Distancia.

El interfaz con el usuario, sin embargo, no es muy amigable. Éste tiene que introducir gran cantidad de datos (14 en total).

En esta práctica vamos a modificar el programa principal de la práctica 1 para que el usuario introduzca las latitudes y longitudes de los puntos usando un texto como el siguiente

20º30'50.7''N

Esta es la forma habitual de expresar coordenadas.

Por otra parte, existe la posibilidad de ejecutar un programa Java desde el Shell del Sistema Operativo, en vez de hacerlo desde el entorno de desarrollo Eclipse (como lo hemos hecho hasta ahora). Aprenderemos cómo hacerlo.

Para no perder el trabajo hecho en la practica 1, haremos un **duplicado** de proXXX.practica1Distancia al que llamaremos **proXXX.practica2Distancia**. En esta práctica trabajaremos sobre éste último proyecto y sobre útiles. Haremos modificaciones en ambos proyectos. Para hacer un duplicado basta con copiar y pegar el proyecto.

3. Mejorar el interfaz de usuario.

Las coordenadas sexagesimales suelen expresarse mediante una cadena de la forma:

GGºMM'SS''D

En dicha cadena (que no contiene espacios en blanco):

- GG son los grados (número entero sin signo)
- MM son los minutos (número entero sin signo)
- SS son los segundos (número real sin signo)
- D es la dirección, un solo carácter, que puede ser:
    - N: Norte
    - S: Sur
    - E: Este
    - O: Oeste

Por ejemplo: 20º30’50.7’’N

Vamos a hacer que el usuario introduzca 6 datos (en lugar de 14):

- Longitud, latitud y altitud del primer punto
- Longitud, latitud y altitud del segundo punto.

Para ello, sigue los siguientes pasos:

1. En la clase Coordenadas (proyecto proXXX.utiles), implementa un método

   public static float sexagesimalToDecimal(String coordenada)

   que dada una coordenada sexagesimal expresada de la forma descrita anteriormente devuelva los grados decimales correspondientes a dicha coordenada. El método supondrá que la coordenada ha sido escrita correctamente.

    - Habrá que separar los dígitos numéricos que contiene de los separadores (º, ‘ y ‘’) y convertirlos a número. Para convertir texto a número sigue el apartado 4.2. de este enunciado.
    - Habrá que tener en cuenta la dirección (N,S,E u O) de la coordenada: Las direcciones N y E producen grados positivos, mientras que las direcciones S y O producen grados negativos.
    - En la clase Coordenadas, implementa un método
2. Modifica el método main de la clase Distancia (proyecto proXXX.practica2Distancia):
    - Debe solicitar las coordenadas pidiendo sólo 6 puntos (en lugar de 14).
    - Convertirlas a coordenadas decimales utilizando llamadas al método sexagesimalToDecimal.
    - Realizar los cálculos de las distancias y mostrar los resultados.

4. Ejecutar el programa desde el Shell del S.O.

Una vez finalizado el apartado anterior, vamos a modificar el programa para que se pueda ejecutar desde el Shell del Sistema Operativo **sin más intervención del usuario**. El usuario, ejecutaría desde el Shell algo similar a lo siguiente:

java Distancia 10º20'53.2"N 11º12'23"E 0.25 11º20'53.2"N 21º12'23"O 0.38

es decir, se llama al programa y a continuación se indican los datos con los que se debe ejecutar (latitud, longitud y altitud del primer punto, latitud, longitud y altitud del segundo).

En este caso, el programa se ejecutará sin intervención del usuario, es decir, mostrará las distancias sin que el usuario tenga que introducir datos durante la ejecución del programa.

Para ser más exactos, lo que pretendemos es lo siguiente:

- Si el usuario ejecuta el programa indicando los 6 datos necesarios, se mostrarán los resultados sin más intervención del usuario.
- Si el usuario ejecuta el programa sin indicar ningún dato, el programa solicitará los 6 datos al usuario (como venía haciendo hasta ahora).
- Si el usuario ejecuta el programa con un número incorrecto de datos, el programa mostrará un mensaje de error, no solicitará ningún dato ni mostrará ningún resultado. El mensaje de error será similar al siguiente:

  Los datos proporcionados al programa son incorrectos.

  Uso: Distancia lat1 long1 alt1 lat2 long2 alt2

4.1. ¿Cómo sabemos con cuantos datos se ha ejecutado un programa y cómo accedemos a dichos datos?

Al estudiar la estructura de un programa se ha visto que ésta era de la forma:

```
class NombrePrograma {
        public static void main (String args[]) {
                ...
        }
}
```

El método main tiene un argumento o parámtro entre paréntesis: String args[], que es un grupo de variables enumeradas (un array en términos de Java) de tipo String.

Cuando el usuario ejecuta el programa NombrePrograma, puede hacerlo utilizando argumentos que se separan por espacios en blanco, como se hace en el siguiente ejemplo en el que se utilizan tres argumentos: "ejem1", "37" y "ejem3"

```
java –jar NombrePrograma ejem1 37 ejem3
```

Cuando el programa NombrePrograma se ejecuta puede usar los argumentos con los que se le ha ejecutado mediante las variables: args[0], cadena de caracteres (String) de valor ”ejem1”; args[1], cadena de caracteres (String) de valor ”37”y args[2], cadena de caracteres (String) de valor ”ejem3”. Los argumentos:

- pueden ser tantos como se desee,
- se separan entre sí por espacios en blanco, y
- son todos de tipo String
- el número de argumentos viene dado por la expresión args.length

4.2. Transformación de Strings a números

En ocasiones necesitamos convertir un String que sabemos que tiene “aspecto” de número en su correspondiente valor numérico.

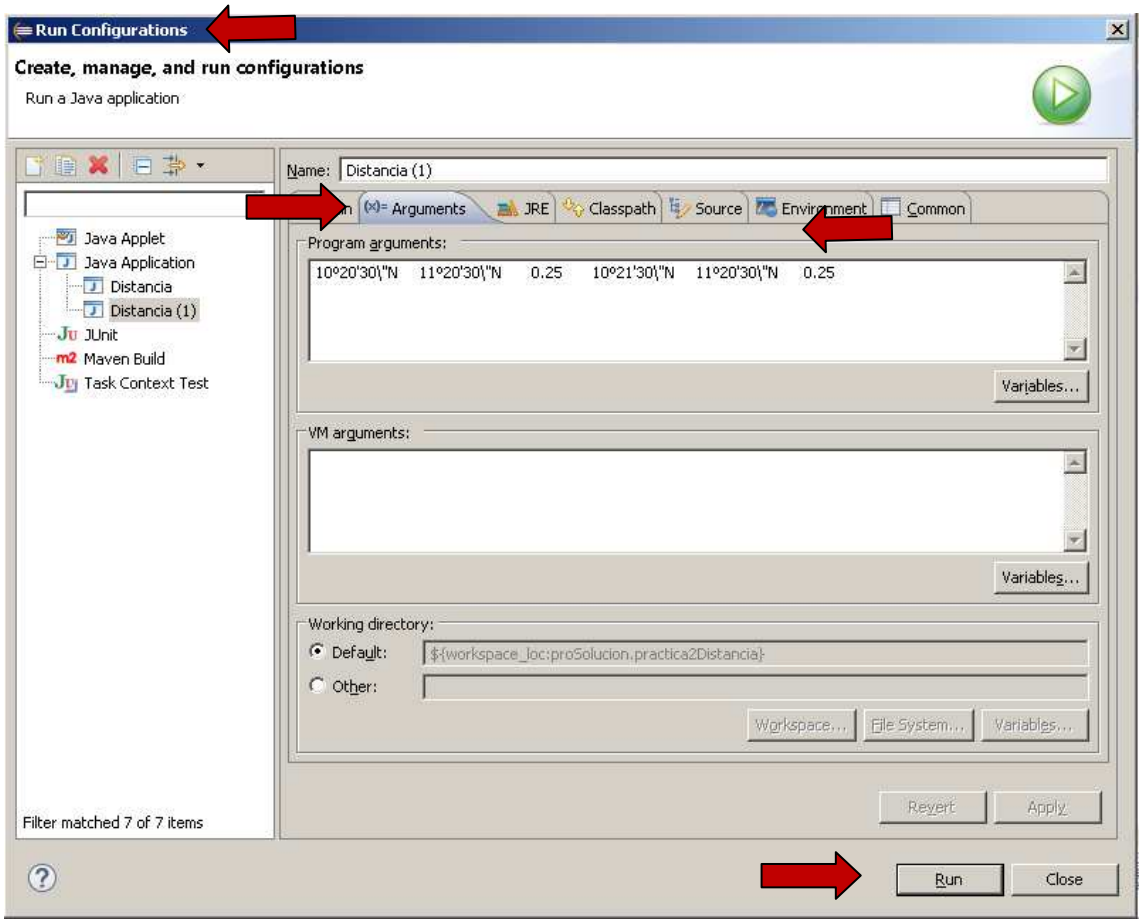
Para hacer esto posible, el lenguaje Java hace uso de unas clases especiales, denominadas clases envoltorio (wrappers en inglés), que tienen operaciones específicas para realizar dicho tipo de conversiones o transformaciones. Las clases Integer y Double, (con mayúsculas) son, por ejemplo, clases envoltorio de los tipos int y double, respectivamente. Otras clases envoltorio son Byte, Short,Long,Float y Boolean.

Para obtener un entero a partir de un String *s* utilizaríamos *Integer.parseInt(s).* Para obtener un double, *Double.parseDouble(s*), etc. Para que la operación correspondiente se pueda llevar a cabo será necesario que el String s tenga el aspecto de un int (P. ej “27”) o de un double (P.ej. “30.56”), etc. De lo contrario, la conversión producirá una Excepción (error).

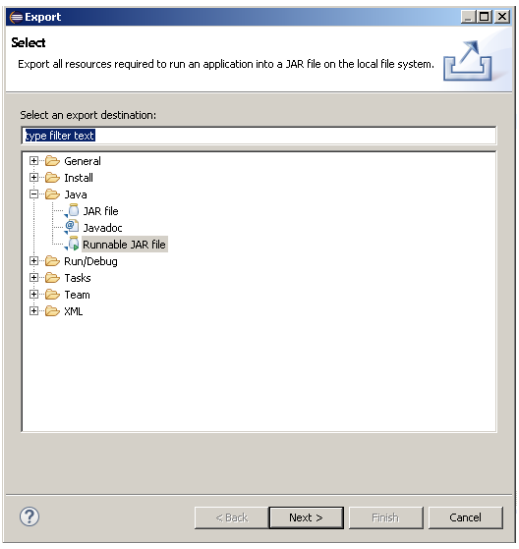
4.3. Ejecutar un programa con argumentos en Eclipse

Tenemos dos posibilidades para ejecutar nuestro programa con argumentos:

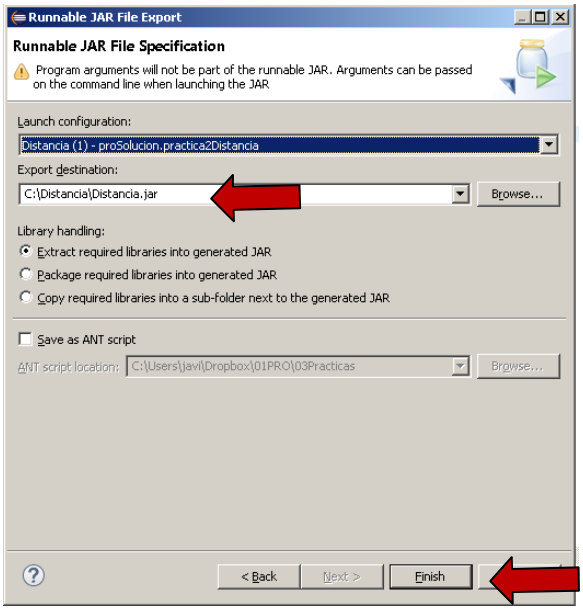
- Usando Eclipse, sin necesidad de abrir la ventana de comandos. Para ello,
    - Hacer clic con el botón derecho en el proyecto Practica2Distancia.
    - Selecciona Run As – Run configurations.
    - En la ventana que aparece selecciona la pestaña “Arguments”.
    - En el cuadro de texto “Program Arguments” escribe, separados por espacios en blanco, los parámetros que quieres pasar al programa.
    - Pulsa “Run”



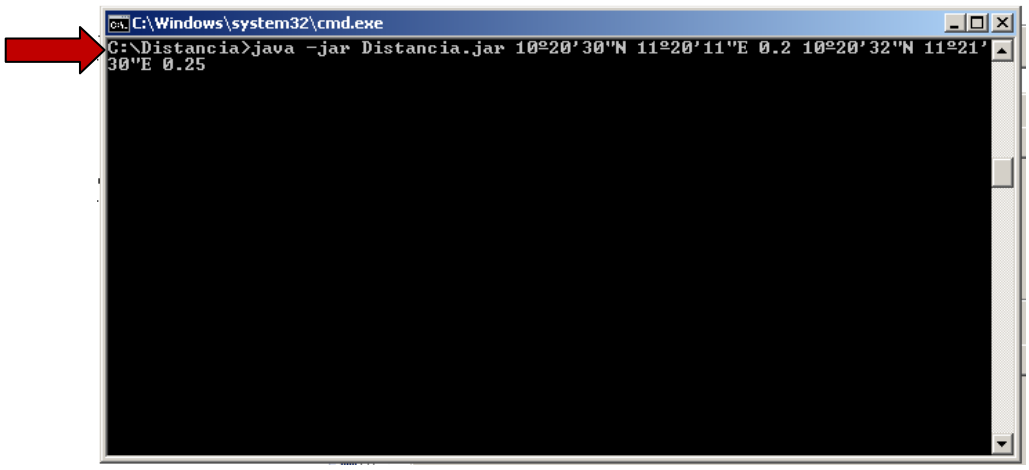
- En la línea de comandos del sistema operativo:
    - Exportar el proyecto a un fichero .jar ejecutable (runnable jar file):
        - Haz clic con el botón derecho sobre el proyecto. Menú “Export”
        - En la ventana, localiza la categoría Runnable JAR file



- En la ventana que aparece, indica una carpeta de destino para el archivo .jar y pulsa “Finish”:



- En el Shell del Sistema Operativo sitúate en la carpeta que contiene el archivo .jar y ejecútalo de la siguiente forma:
  - java –jar Distancia.jar parametros



5. Evaluación de la práctica

Se valorará:

- La corrección de los métodos propuestos.
- Que los proyectos se nombren tal y como indica el enunciado.
- Que las identificadores (de métodos, variables y clases) sean adecuados: nombres significativos y siguiendo el criterio de mayúsculas y minúsculas recomendado por Java.

- La correcta identación del código y su legibilidad.
- La definición de constantes cuando se dé el caso.
- La elección del tipo de las variables.

6. Entrega de la práctica

- Realiza una captura de pantalla mientras ejecutas el programa desde la línea de comandos en la que se vean el comando utilizado y el resultado de la ejecución.
- Localiza la carpeta de proyecto en tu workspace y comprímela en un archivo y renómbralo como XXXpractica2 (y la extensión correspondiente), donde XXX será tu apellido y nombre. En el archivo comprimido incluye también la captura de pantalla indicada anteriormente
- Súbelo al aula virtual: Junto a la carpeta de la práctica encontrarás un apartado que te permitirá subir el archivo comprimido.
- Plazo entrega: Hasta el martes 17 de noviembre a las 23:55.