Laboratorio A.E.D. Ejercicio Individual 4

Guillermo Román
guillermo.roman@upm.es
Lars-Åke Fredlund
lfredlund@fi.upm.es
Manuel Carro
mcarro@fi.upm.es
Marina Álvarez
marina.alvarez@upm.es
Julio García
juliomanuel.garcia@upm.es
Tonghong Li
tonghong@fi.upm.es

Normas.

- Fechas de entrega y la penalización aplicada a la puntuación obtenida sobre 10:

Hasta el Lunes 23 de octubre, 23:59 horas	0 %
Hasta el Martes 24 de octubre, 23:59 horas	20 %
Hasta el Miércoles 25 de octubre, 23:59 horas	40 %
Hasta el Jueves 26 de octubre, 23:59 horas	60 %
Después la puntuación máxima será 0	

- Se comprobará plagio y se actuará sobre los detectados
- Usad las horas de tutoría para preguntar sobre programación – son oportunidades excelentes para aprender

Entrega

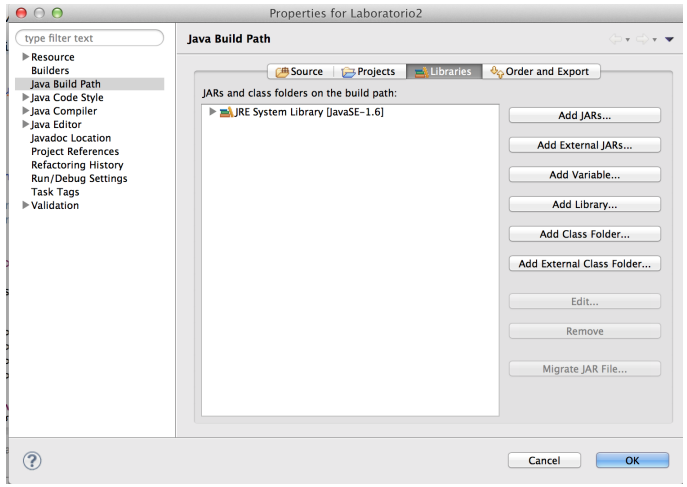
- Todos los ejercicios de laboratorio se deben entregar a través de la web `http://lml.ls.fi.upm.es/~entrega`.
- El fichero que hay que subir es `Exploradora.java`.

Configuración previa

- Arrancad Eclipse
- Si trabajáis en portátil, podéis utilizar cualquier versión relativamente reciente de Eclipse. Debería valer cualquier versión a partir de la versión 3.7. Es suficiente con que instaléis la *Eclipse IDE for Java Developers*
- Cambiad a "Java Perspective".
- Cread un proyecto Java llamado `aed`:
  - Seleccionad separación de directorios de fuentes y binarios
- Cread un *package* `aed.laberinto` en el proyecto `aed`, dentro de `src`
- Aula Virtual $\rightarrow$ AED $\rightarrow$ Laboratorios y Entregas Individuales $\rightarrow$ Individual 4 $\rightarrow$ `Individual4.zip`; descomprimidlo
- Contenido de `Individual4.zip`:
  - `Exploradora.java`, `Lugar.java`, `Punto.java`, `PuntoCardinal.java`, `RunOneTest.java`,`TesterInd4.java`

Configuración previa al desarrollo del ejercicio.

- Importad al paquete `aed.laberinto` los fuentes que habéis descargado (`Exploradora.java`, `Lugar.java`, `Punto.java`, `PuntoCardinal.java`, `RunOneTest.java`,`TesterInd4.java`)
- Añadid al proyecto `aed` la librería `aedlib.jar` que tenéis en Moodle (en Laboratorios y Entregas Individuales). Para ello:
- Project → Properties → Java Build Path. Se abrirá una ventana como esta:



- Usad la opción “Add External JARs...”.
- Intentad ejecutar `TesterInd4.java`

Documentación de la librería aedlib.jar

- La documentación de la API de la librería aedlib.jar esta disponible en `http://lml.ls.fi.upm.es/~entrega/aed/docs/aedlib/`
- También se puede añadir la documentación de la librería a Eclipse (*no es obligatorio*): en el "Package Explorer": "Referenced Libraries" $\rightarrow$ `aedlib.jar` y elige la opción "Properties". Se abre una ventana donde se puede elegir "Javadoc Location" y ahí se pone como "javadoc location path:" `http://lml.ls.fi.upm.es/~entrega/aed/docs/aedlib/` y presionar el botón "Apply and Close"

Tarea: Explorar un laberinto usando una pila (LIFO)

- El objetivo de este laboratorio es desarrollar un método `explora`, dentro la clase `Exploradora`, que es capaz de sistemáticamente explorar un laberinto, buscando un "tesoro".
- El laberinto esta compuesto por "lugares", que están implementados como objetos de la clase `Lugar`
- En un lugar:
  - puede haber un tesoro (un objeto)
  - el "suelo" puede estar marcado con tiza (para detectar que un lugar ya ha sido visitado)
  - puede haber caminos hacia otros lugares.

La clase Lugar

```
public class Lugar {
  public boolean tieneTesoro()       // Devuelve true si el lugar tiene un tesoro

  public Object getTesoro()          // Devuelve el tesoro (un objeto) o null

  public Iterable<Lugar> caminos()   // Devuelve los lugares vecinos que conecta
                                     // con el lugar (del objeto)

  public void marcaSueloConTiza()    // Permite marcar el ''suelo'' en el lugar
                                        con ''tiza''

  public boolean sueloMarcadoConTiza()   // Esta marcado el suelo con tiza?

  public String toString()               // Para imprimir el lugar

  public void printLaberinto()           // Imprime todo el laberinto
}
```

Completar la clase `Exploradora`

Solo es necesario implementar el método `explora` de la clase `Exploradora.java`:

```
// Explora el laberinto, empezando en el lugar inicial,
// y devolviendo ``el tesoro'' (un Object), o null
// si no es posible encontrar ningun tesoro.
public static Object explora(Lugar lugarInicial) {
    LIFO<Lugar> faltaPorExplorar = new LIFOList<Lugar>();

    // Modificar el resto de este metodo
    ...
}
```

- Es **obligatorio** usar el atributo

```
private LIFO<Lugar> faltaPorExplorar;
```

para guardar lugares nuevos de explorar (devuelto por llamadas al método `caminos` de la clase `Lugar`)

Reglas de la Implementación

- Modificad solo `Exploradora.java`.
- NO esta permitido hacer *casting* ni usar **instanceof**
- Está permitido añadir métodos auxiliares
- NO está permitido añadir nuevos atributos
- NO se debe modificar el contenido de las estructuras de datos recibidas como parámetros en los métodos

Consejos:

- Imprime un lugar usando por ejemplo:

```
// Asumimos que lugar es un variable de tipo Lugar
System.out.println("Estoy␣en␣el␣lugar␣"+lugar);
        ==> Estoy en el lugar {x=0,y=1}, esta marcado con tiza
```

- Se puede imprimir el laberinto entero llamando:

```
// Asumimos que lugar es un variable de tipo Lugar
lugar.printLaberinto();
```

- Sale una representación textual del laberinto:

```
  +-----+-----+-----+
2 |     |           |
  +-----+-----+     +
1 | .       x |  $  |
  +     +     +     +
0 | .   |           |
  +-----+-----+-----+
      0     1     2
```

El símbolo "$" marca el tesoro en el lugar (2,1), "x" es el lugar actual en el laberinto (1,1), y "." adorna los lugares marcado con tiza.

Consejos: ejecutar solo una prueba

- Se puede ejecutar solo una prueba fácilmente cambiando la clase `RunOneTest.java`, y ejecutando la clase
- Por defecto la clase contiene

```
package aed.laberinto;

public class RunOneTest {
  public static void main(String args[]) {
    TesterInd4.test_5();
  }
}
```

  para poder correr el "test_5".
- Otro alternativo para solo correr un test es especificar el numero de test como parámetro para el programa `TesterInd4` dentro Eclipse: "Run As" $\Rightarrow$ "Run Configurations" $\Rightarrow$ "TesterInd4" $\Rightarrow$ "Arguments".

Algoritmo para Explorar el Laberinto

- Un posible algoritmo para explorar el laberinto es:
  - En el método `explora`:
    1. poner en la pila el lugar inicial
    2. Si la pila esta vacía hemos terminado y se devuelve `null` para indicar que no se ha encontrado el tesoro
    3. Si la pila no esta vacía, extraemos el primer `lugar`
    4. Si el suelo de este `lugar` esta marcado con tiza, regresamos a paso 2
    5. Si el `lugar` contiene el tesoro, devolvemos el objeto encontrado y terminamos
    6. Marcamos el suelo con tiza
    7. Recuperamos los lugares directamente alcanzables desde `lugar` llamando al método `caminos` de la clase `Lugar` y ponemos todos estos lugares en la pila
  - Volver al paso 2

Notas

- El proyecto debe compilar sin errores y debe cumplirse la especificación de los métodos a completar
- Debe ejecutar `TesterInd4.java` correctamente y sin mensajes de error
  - Nota: una ejecución sin mensajes de error no significa que el método sea correcto (es decir, que funcione bien para cada posible entrada)
- Todos los ejercicios se comprueban manualmente antes de dar la nota final