GRADO

PROGRAMACIÓN Y ESTRUCTURAS DE DATOS AVANZADAS

ENUNCIADO PRÁCTICA 2

Curso 2023-2024

GRADO EN INGENIERÍA EN INFORMÁTICA Y EN TECNOLOGÍAS DE LA INFORMACIÓN

1.- ENUNCIADO DE LA PRÁCTICA

Un robot se mueve en un edificio en busca de un tornillo. Se trata de diseñar un algoritmo que le ayude a encontrar el tornillo y a salir después del edificio. El edificio debe representarse como una matriz de entrada a la función, cuyas casillas contienen uno de los siguientes tres valores: L para "paso libre", E para "paso estrecho" (no cabe el robot) y T para "tornillo". El robot sale de la casilla (1,1) y debe encontrar la casilla ocupada por el tornillo. En cada punto, el robot puede tomar la dirección Norte, Sur, Este u Oeste siempre que no sea un paso demasiado estrecho. El algoritmo debe devolver la secuencia de casillas que componen el camino de regreso desde la casilla ocupada por el tornillo hasta la casilla (1,1). Supondremos que la distancia entre casillas adyacentes es siempre 1.

No se pide el camino más corto sino el primero encontrado por lo que se utilizará el **esquema de vuelta atrás,** de forma que la búsqueda se detenga en la primera solución y devuelva la secuencia de casillas desde el tornillo hasta la salida. El enunciado del problema, la descripción de la solución, junto con un ejemplo explicativo se puede encontrar en el apartado 6.6 del texto base de la asignatura.

2.- REALIZACIÓN DE LA PRÁCTICA

2.1.- Diseño del algoritmo

La práctica constará de una memoria y de un programa en java original que resuelva el problema aplicando el esquema de vuelta atrás.

2.2.- Argumentos y parámetros

La práctica se invoca usando la siguiente sintaxis:

```
java robot [-t][-h] [fichero entrada] [fichero salida]
```

o

```
java –jar robot.jar [-t][-h] [fichero entrada] [fichero salida]
```

Los argumentos son los siguientes:

- **-t**: traza cada paso de manera que se describa la aplicación del algoritmo utilizado.
- **-h**: muestra una ayuda y la sintaxis del comando.
- **fichero_entrada:** es el nombre del fichero del que se leen los datos, en este caso, la matriz que representa la forma del edificio por el que se mueve el robot.
- **fichero_salida:** es el nombre del fichero que se creará para almacenar la salida. Si el fichero ya existe, el comando dará un error. Si falta este argumento, el programa muestra el resultado por pantalla.

Por ejemplo:

```
$ java robot -h <ENTER>
```

```
SINTAXIS: robot [-t][-h] [fichero entrada]
-t                   Traza el algoritmo
-h                  Muestra esta ayuda
[fichero entrada]    Nombre del fichero de entrada
[fichero salida]     Nombre del fichero de salida
```

2.3- Datos de entrada

El fichero de entrada consta de una primera fila que indica el número de filas de la matriz que representa los pasos por el edificio por el que se mueve el robot, una segunda fila que muestra el número de columnas de dicha matriz, y, a continuación, la propia matriz (L: paso libre, E: paso estrecho y T: tornillo). Los valores se escriben separados por blancos, como en el ejemplo:

```
5
6
L L E L L E
L E L L L E
L L L E L L
L L E L L L
L L L L T L
```

En caso de que el fichero de entrada no exista, se leerán los datos por la entrada estándar.

2.4- Datos de salida

La salida constará de una secuencia de coordenadas que van desde la posición del tornillo a la de entrada al edificio.

```
( 5 , 5 )
( 4 , 5 )
( 3 , 5 )
( 2 , 5 )
( 1 , 5 )
( 1 , 4 )
( 2 , 4 )
( 2 , 3 )
( 3 , 3 )
( 3 , 2 )
( 4 , 2 )
( 5 , 2 )
( 5 , 1 )
( 4 , 1 )
( 3 , 1 )
( 2 , 1 )
( 1 , 1 )
```

En caso de que no sea posible llegar al tornillo desde la casilla de partida, el programa debe indicarlo:

No se encontró un camino hasta el tornillo.

En caso de que el fichero de salida no se indique en la llamada al programa, se escribirá el resultado por la salida estándar.

2.5.- Implementación del algoritmo

El programa se desarrollará en Java siguiendo un diseño orientado a objetos. Los detalles del entorno recomendado se encuentran en la guía de la asignatura. Se valorará el diseño OO y la eficiencia del desarrollo.

3.- CUESTIONES TEÓRICAS DE LA PRÁCTICA

1) Indica y razona sobre el coste temporal y espacial del algoritmo.
2) Explica qué otros esquemas pueden resolver el problema y razona sobre su idoneidad.

4.- ENTREGA DE LA PRÁCTICA

4.1 Material que hay que entregar al Tutor

Se confeccionará una memoria, en PDF con el siguiente índice:

1. Portada de la memoria con nombre, apellidos, dni y dirección de correo.
2. Respuesta a las cuestiones teóricas planteadas en este enunciado.
3. Un ejemplo de ejecución para distintos tamaños del problema.
4. Un listado del código fuente completo.

4.2 Normativa de las prácticas en relación con el Centro Asociado:

1. La asistencia a las sesiones de prácticas es obligatoria.
2. El calendario y procedimiento para asistir a las sesiones de prácticas **está publicado en su Centro Asociado o bien aparece en el foro correspondiente a su centro en el curso virtual.**
3. El plazo de entrega de la documentación y de la práctica **lo establece el Tutor de prácticas de cada Centro Asociado o Campus**.
4. El Tutor califica la práctica, informa al alumno y en su caso la revisa de acuerdo con los horarios y procedimiento que establezca el Centro Asociado.
5. Todos los alumnos deberán registrarse a través del Curso Virtual en el grupo del Tutor/a con el que hayan asistido a las sesiones presenciales obligatorias a fin de que su práctica pueda ser calificada.
6. La práctica se debe aprobar en la misma o anterior convocatoria para que se pueda calificar la asignatura. En caso contrario la calificación será de suspenso.
7. La práctica se entregará tanto en el entorno virtual como al Tutor. La falta de cualquiera de ellas será motivo suficiente para quedar excluida de la convocatoria.

El alumno debe asegurarse de que no se da ninguna de las siguientes circunstancias, ya que implican automáticamente una calificación de suspenso:

- **Código:** el código no compila, no está desarrollado en Java, no se corresponde con el pseudocódigo recogido en la documentación, no es original, está copiado de la red, academia, compañero, etc., o no sigue un diseño OO encapsulado o modular.
- **Ejecutable:** el ejecutable no termina, se queda sin memoria con ejemplares pequeños o aborta sin justificación. El ejecutable no lee los ficheros previstos en el formato adecuado. No trata los argumentos o no se ajusta a las especificaciones.
- **Documentación:** No se presenta en el soporte indicado por el tutor o está incompleta.
- **Soporte:** No se puede leer, o contiene un virus de cualquier tipo. A este respecto, las prácticas en las que se detecte cualquier tipo de virus estarán suspensas.

Los **alumnos estudiando en el EXTRANJERO** se deberán poner en contacto con el profesor tutor que se indicará en los foros.