Práctica 4: Algoritmos devoradores

Importante: todos los alumnos involucrados en casos de plagio o copia recibirán una calificación de cero (0) en la práctica, obteniendo por tanto una calificación de SUSPENSO en la primera convocatoria ordinaria del curso. Se hace uso de herramientas automáticas para la detección de copias.

1 Elección de torretas

En el campo de batalla académico de la Universidad de los Sabios, la colocación estratégica de las torretas se ha convertido en la piedra angular para resistir la invasión goblin. Los estrategas defensores se enfrentan a la crucial tarea de diseñar un escudo impenetrable mediante la disposición táctica de sus torres en el terreno de juego.

Cada celda no transitable del tablero representa un bastión potencial para erigir una torreta, y la decisión de cómo distribuir estas defensas influye directamente en la eficacia de la resistencia. ¿Se concentrarán los defensores en puntos críticos, creando una fortaleza en lugares estratégicos, o extenderán su defensa para abarcar una mayor área del terreno? El desafío radica en anticipar los movimientos de los goblins y bloquear sus caminos de manera eficiente. La elección de la ubicación de las torretas determinará el destino de la batalla, ya que un diseño hábil puede desviar a los invasores y maximizar el impacto de los ataques defensivos.

En este duelo de ingenio y astucia, la colocación estratégica de torretas no solo es una medida de seguridad, sino una obra maestra táctica que busca preservar el conocimiento de la universidad. ¿Podrán los defensores diseñar un escudo lo suficientemente fuerte como para resistir la embestida goblin y proteger el preciado legado académico? ¡La batalla por la Universidad de los Sabios comienza en la elección sabia de la ubicación de las torretas!

2 Desarrollo de la práctica

El juego sigue las mismas reglas que se describieron en los guiones de las prácticas anteriores. Como jugadores, nuestro objetivo es determinar en qué ubicación colocamos las torretas disponibles. Pero no todas las ubicaciones posibles son correctas. Para poder colocar una torreta en una ubicación del mapa han de cumplirse las siguientes condiciones:

- El centro de las torretas coincidirá con el centro de las celdas del terreno.
- Tiene que quedar ubicada completamente dentro de los límites del terreno de juego. No puede salirse de sus límites.
- No puede colisionar con ningún obstáculo ni con ninguna otra torreta.
- No pueden colocarse sobre los caminos. Para garantizar que los enemigos pueden hacer uso de las celdas transitables habrá obligatoriamente que dejar espacio suficiente hasta el centro de las celdas transitables. Deberá dejarse al menos una cantidad de espacio de al menos tantas unidades como radio tenga el mayor de los enemigos posibles (10 unidades por defecto).

Tenga en cuenta que todas las entidades del juego emplean círculos para representar el espacio que ocupan en el terreno de juego (ver Fig. 1).

Para definir la estrategia de colocación, debe modificarse el método `placeTowers` de la clase `TowerPlacer`, situada dentro del paquete `net.agsh.towerdefense.strats`. El método recibe una lista de objetos de tipo `Tower` y un objeto de tipo `Map` con la información sobre el terreno de juego. El método debe devolver una lista con las torretas, a las cuales se les ha establecido una posición mediante el método `setPosition`.

Para compilar la práctica basta con ejecutar el archivo `compile.bat` ejecutar la siguiente orden en la consola (siempre desde el directorio raíz de la práctica):

```
javac -cp "towerdefense.jar" net\agsh\towerdefense\strats\*.java
```

Y para ejecutarlo basta con usar el comando `run.bat` o ejecutar la siguiente orden en la consola de comandos (siempre desde el directorio raíz de la práctica):

```
java -cp ".;towerdefense.jar" Main
```

Se imprimirán a continuación una serie de mensajes con información sobre la partida. Opcionalmente, es posible visualizar la partida de forma gráfica. Basta con añadir el parámetro `GUI=1` al final de la instrucción anterior o hacer uso del archivo `run-gui.bat`. En ambos casos se mostrará una ventana similar a la mostrada en la Fig. 2.

3 Ejercicios

Para completar la práctica se proponen una serie de tareas, que deberán llevar a cabo cada alumno **individualmente**. El resultado de estas tareas dará lugar al desarrollo de una aplicación en Java y de una pequeña memoria a incluir al comienzo del código fuente en forma de comentario. El resultado de la práctica deberá entregarse a través del campus virtual antes de la fecha límite, a través de la tarea creada al efecto y en el formato solicitado. No se corregirán productos enviados por otros medios o que se envíen en un formato distinto al solicitado.

El objetivo de la práctica consiste en desarrollar un algoritmo, **basado en el enfoque de los Algoritmos Devoradores**, que sirva para determinar la mejor ubicación para las torretas.

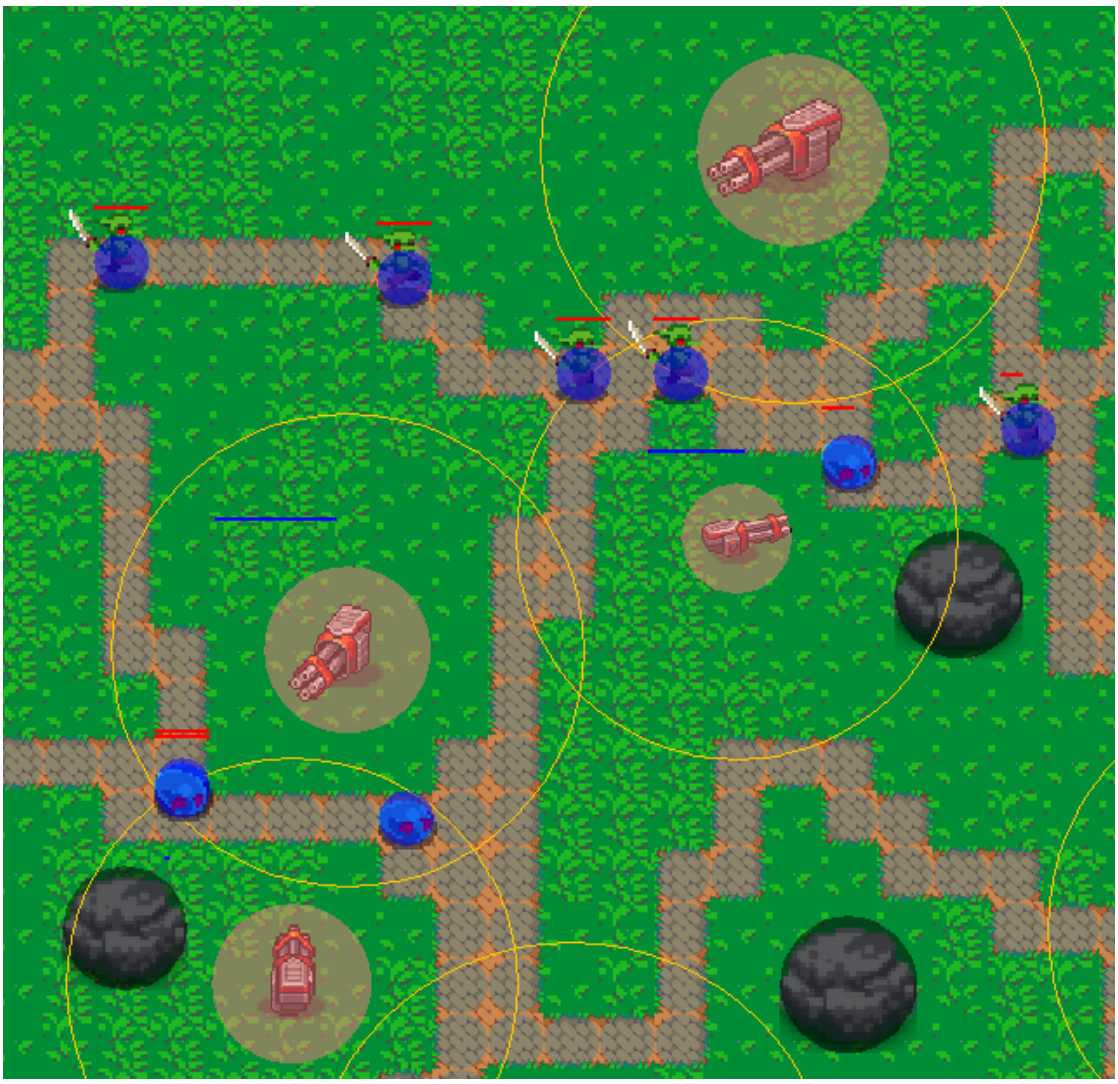

Figure 1: Representación de entidades como círculos.

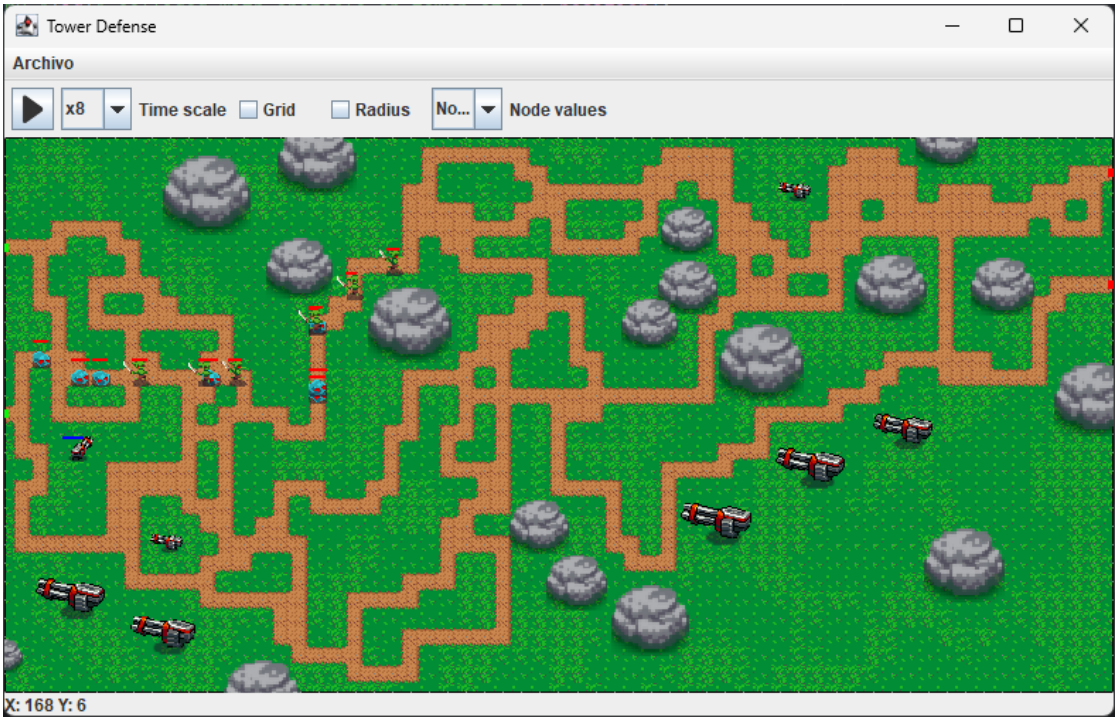

Figure 2: Interfaz gráfica del juego

En el material de prácticas podrá encontrar un código de ejemplo que coloca aleatoriamente las torretas disponibles. Este código, aunque funcional, es muy poco eficaz. Deberá sustituir ese código de ejemplo por uno mejor. Para ello, es conveniente tener en cuenta la forma del terreno de juego. Siempre será mejor colocar las torretas cerca de las celdas transitables en lugar de colocarlas alejadas de éstas. Concretamente, las tareas a llevar a cabo son:

1. Descargar el archivo comprimido del campus con los códigos de ayuda y estudiar su contenido.

2. Ejecutar el archivo "compile.bat" desde una consola de comandos para compilar el código de ejemplo.

3. Ejecutar el archivo "run.bat" para ejecutar el código de ejemplo.

4. Ejecutar el archivo "run-gui.bat" para mostrar la interfaz gráfica.

5. Modificar el método `collide` de la clase `TowerPlacer` para que devuelva un valor cierto en caso de que dos entidades dadas se solapen.

6. Modificar el método `getNodeValue` para que devuelva el valor asignado a una celda del terreno de juego. Ese valor debe ser numérico y puede depender de las

características del terreno de juego. Pueden añadirse todos los parámetros que se estimen oportunos.

7. Añadir un comentario al comienzo del fichero `TowerPlacer.java` con la formula o procedimiento empleado para determinar el valor de una celda (`MapNode`) del terreno.

8. Modificar el método `placeTowers` para que asigne la posición a las torretas. Estas posiciones deberán coincidir con la posición de algunas de las celdas del terreno y ser válidas, de acuerdo a las condiciones explicadas en la sección anterior. Deberá emplearse para este fin un algoritmo **basado en el enfoque de los Algoritmos Devoradores**. Deberá hacer uso de los métodos modificados anteriormente.

4 Entrega

Todo el código desarrollado en esta práctica deberá estar incluido en el fichero "TowerPlacer.java". Será éste el único archivo que se envíe a través de la tarea habilitada al efecto en el campus virtual. El fichero debe enviarse sin comprimir y listo para ser compilado en los ordenadores de los laboratorios. **Si el fichero no compila no se evaluará la práctica.**

5 Evaluación

Todos los programas enviados por los alumnos serán ejecutados en uno de los ordenadores del laboratorio y se anotará la puntuación obtenida por cada alumno. La calificación de la práctica se otorgará en función de la puntuación obtenida.