Fundamentos de Programación

Práctica Final 1 – 2020-2

Este documento describe la práctica final de la asignatura Fundamentos de Programación, del semestre 2020-2

Objetivo General

El objetivo de este trabajo es poner en práctica los conocimientos adquiridos en el curso, en particular los conceptos de matrices, objetos, y herencia.

Objetivos Específicos

En la práctica se deben poner en evidencia las siguientes habilidades:

a. Capacidad de esquematizar en un *diagrama de clases* las entidades que conforman la solución del problema.
b. Capacidad de *implementar* un programa en Java que funcione correctamente y que sea acorde al diagrama de clases elaborado.
c. Capacidad para generar la *documentación* del código utilizando JavaDOC

Tema para el trabajo

En la práctica vamos a implementar un juego tipo PacMan, pero con interfaz tipo texto. El objetivo es que el jugador lleve al PacMan desde el punto de partida hasta la meta. Como es un PacMan paisa, en algunas versiones en el laberinto habrá arepitas, cada arepita le dará un punto de vida adicional. El problema es que algunas de las arepitas son explosivas, y si se las come, perderá 5 puntos de vida. Las arepitas alimenticias y las explosivas tienen la misma apariencia, de modo que no es posible saber de qué tipo de arepita se trata antes de comérsela. Adicionalmente, cada 10 turnos el jugador perderá automáticamente 1 punto de vida. Si el jugador logra llevar el PacMan a la meta, ganará el juego y pasará al siguiente nivel. Si los puntos de vida llegan a 0 antes de que el PacMan llegue a la meta, se perderá el juego.

En la figura 1 se presenta un posible laberinto vacío.

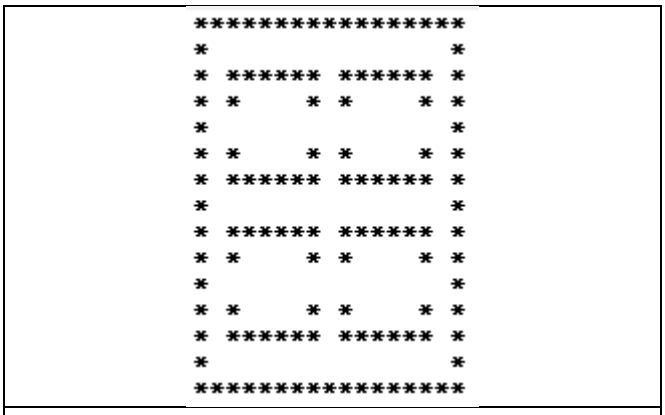


Fig. 1. Laberinto vacío de 15 filas x 17 columnas

El PacMan se representa con el carácter '^', la meta con el carácter 'O' (o mayúscula), ¿los fantasmas con la letra 'w? y las arepitas con el carácter '.' (punto). El PacMan se mueve con las siguientes teclas: 'a' (izquierda), 'd' (derecha), 'w' (arriba), 's' (abajo). Como estamos utilizando entrada por la consola, sin utilizar la interfaz gráfica de Java, es necesario que el usuario presione la telca <enter> luego de cada letra de movimiento. Esto es, si se va a mover el PacMan a la izquierda, se digita 'a' y luego <enter>.

Se van a implementar las siguientes versiones:

En la **primera entrega** de la práctica, se deben programar las versiones 1 y 2.

- **Versión 1**: El jugador mueve el PacMan usando el teclado. No hay arepitas. No hay enemigos. El jugador simplemente debe llevar el PacMan hasta la meta.
- **Versión 2**: El jugador mueve el PacMan usando el teclado. Hay arepitas, unas buenas y otras explosivas. No hay enemigos. El jugador debe llevar el PacMan hasta la meta antes de que se le acaben los puntos de vida.

En la **segunda entrega**, se deben programar las versiones 3 y 4.

- **Versión 3**: El jugador mueve el PacMan. Hay arepitas buenas y explosivas. Hay fantasmas y son controlados por el programa ("*non-playable characters*"). Si el fantasma tiene al PacMan en su línea de visión, lo persigue; de lo contrario se mueve de manera aleatoria. Los fantasmas se mueven 2 casillas por turno, el jugador 1 casilla por turno.
- **Versión 4**: El programa mueve el PacMan. No hay arepitas. No hay fantasmas. El programa debe ser capaz de mover el PacMan a la meta. Este algoritmo es un poco complejo, pero muy interesante. Buscar un algoritmo que encuentre la ruta, y que sea una ruta corta (estilo "Waze"), será un extra-crédito (no obligatorio).

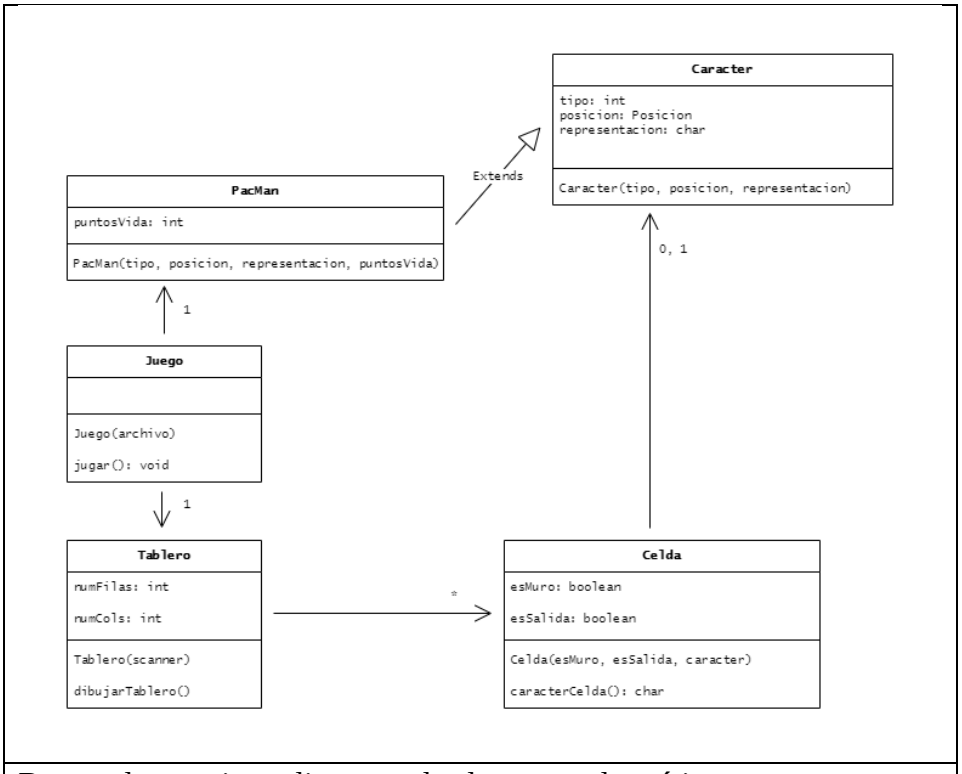
En cada turno de las versiones 1 a 3, se llevan a cabo las siguientes acciones:

- Se lee del teclado la letra que representa la dirección en que el jugador desea mover el PacMan. Si se trata de una celda válida, se mueve el PacMan a la nueva celda, si no es válida, el usuario perdió su jugada.
    - Si la celda es la meta, el jugador gana el juego
    - Si en el nivel hay fantasmas y la celda está ocupada por un fantasma, el jugador pierde el juego

- Si hay arepitas, se actualizan los puntos de vida del jugador dependiendo si es una arepita buena o una arepita explosiva.
- Si el turno es múltiplo de 10, se restan los puntos de vida que se haya definido.
- Si hay fantasmas, se hace lo siguiente por cada fantasma:
    - Si hay línea de visión directa hacia el PacMan, mover el fantasma 2 celdas en la dirección donde está el PacMan. De lo contrario, el fantasma se mueve de manera aleatoria.
    - Si el fantasma alcanza al usuario, el jugador pierde el juego
- Luego de hacer las actualizaciones anteriores, se debe dibujar el nuevo tablero y mostrarle al usuario cuántos puntos de vida le quedan.

Diagrama de Clases

La siguiente figura representa una primera versión de un diagrama de clases (simplificado) para la práctica:



Boceto de un primer diagrama de clases para la prática

La clase encargada de crear todo lo demás es la clase Juego. Notar que la clase Juego tiene un Pacman y un Tablero. Notar que el PacMan extiende la clase Carácter. En la primera versión el único Carácter será el PacMan, pero en versiones posteriores habrá Fantasmas, que también extenderán la clase Carácter. Para cada carácter se guarda su tipo (PacMan o Fantasma), su

posición (en fila y columna) y su representación (‘^’ para el PacMan, ‘w’ para los fantasmas). Además de los atributos que hereda el PacMan del carácter, se almacena también los puntos de vida que tiene. El Tablero tiene un número de filas y columnas determinado y tiene una matriz de referencias a Celdas. Cada Celda tiene una variable boooleana que indica si es muro (si no es muro es un espacio libre), y si es la salida. Adicionalmente, puede tener (o no) una referencia a un carácter, si está ocupada por uno.

El método jugar(), dentro de la clase Juego, es el centro del juego. Allí se le pide la entrada al usuario, se actualizan las posiciones, se calcula si se ganó o se perdió el juego, etc. Al final de cada turno, se invoca el método dibujarTablero(), del objeto tablero, para que el usuario pueda ver qué ocurrió durante el turno.

Ejemplo Versión 1

El siguiente video muestra la interacción de un usuario con el juego:

Ejemplo de una interacción del jugador con la versión 1 del juego

Anotaciones importantes

El laberinto se lee desde un archivo tipo texto. En la primera línea se lee el número de filas (nFilas) y el número de columnas (nCols) que tiene el laberinto. Luego viene una matriz donde cada carácter es ‘*’ si corresponde a una pared del laberinto o un espacio ‘ ‘ si es un espacio libre en el laberinto.
En las siguientes líneas se especifica lo siguiente:

- Si hay una ‘P’ en la primera columna, los dos siguientes enteros son la posición inicial (por fila y columna) del PacMan.
- Si hay una ‘F’ en la primera columna, los dos siguientes enteros son la posición inicial del Fantasma. Puede haber 0, 1 o varios fantasmas (solo en la versión 3 del juego).

Qué se debe entregar

Se debe entregar lo siguiente con respecto al proyecto:

- Código fuente.
- Documentación del código fuente utilizando JavaDoc. Se debe documentar cada clase, cada método (con sus parámetros y valores de retorno) y cada atributo público.
- Diagrama de clases del proyecto. Para construirlo se debe utilizar una herramienta como ArgoUML, StarUML, Draw.io o similares.

- Un manual de usuario de 2 o 3 páginas (en archivo pdf) que describa la interacción con el usuario a través de los menús tipo texto.
- Un reporte de algunas estrategias que se hayan implementado y las pruebas correspondientes para algunos intervalos de tiempo determinados. Discusión de los resultados.

Fechas de Entrega

La entrega 1: por definir
La entrega 2: por definir

Importante:

Cada estudiante debe dar cuenta completa de *todos* y *cada uno* de los productos que entrega. No se debe utilizar código creado por otras personas, ni código disponible en Internet, con excepción de las librerías de Java y el código que entrega el profesor de la asignatura. Durante la sustentación el profesor hablará con cada estudiante y solicitará que se realicen cambios al código. Si el estudiante no puede realizar estas modificaciones, se entenderá que *no hizo la aplicación* y se considerará como fraude de acuerdo con el reglamento estudiantil. Muy importante mirar los artículos pertinentes del Reglamento Académico en las siguientes páginas:

http://www.eafit.edu.co/institucional/reglamentos/Documents/pregrado/regimen-disciplinario/cap1.pdf

http://www.eafit.edu.co/institucional/reglamentos/Documents/pregrado/regimen-disciplinario/cap4.pdf

Código de Ética:

- Usted puede conversar con sus compañeros acerca de los enfoques que cada uno está utilizando para la práctica, pero NO se debe mirar el código de sus compañeros y mucho menos usarlo como parte de su práctica.
- Se puede utilizar, como punto de partida, el código que se entrega en el curso.
- No debe aceptar que otra persona (compañero, tío, amigo, novia, primo hermano del mocho) "le ayude" escribiendo parte del código de su práctica. Al hacerlo estaría perdiendo la mejor oportunidad para aprender que usted tiene en el curso.

Esperamos que disfruten mucho haciendo su práctica. Las cosas que uno disfruta haciendo resultan mejor.

La recomendación más importante:
Es mejor entregar una práctica con una funcionalidad sencilla, pero hecha completamente por Usted, que una práctica muy completa hecha con ayuda indebida por parte de otra persona.

Realizar la práctica, de manera individual, es la mejor experiencia de aprendizaje de la asignatura.