Trabajo Práctico Especial
Programación Orientada a Objetos
Diciembre 2019

Objetivo

El objetivo del presente trabajo es implementar nuevas funcionalidades en un juego desarrollado en lenguaje Java, con interfaz visual, aplicando los conceptos sobre diseño orientado a objetos adquiridos en la materia.

Descripción funcional

La aplicación es una versión académica y muy reducida del juego "Candy Crush". El juego consiste en una grilla orientada verticalmente, donde caen y se apilan caramelos de diferentes colores. La mecánica del juego consiste en elegir un par de caramelos adyacentes para intercambiar sus posiciones. Este intercambio es válido solo si al realizarlo se forma alguna figura válida con caramelos de un mismo color.

Las figuras son líneas horizontales o verticales de 3, 4 o 5 caramelos, o bien una T (de tres caramelos en línea y dos perpendiculares en el medio) en cualquier orientación, o bien una L en cualquier orientación.

Si el intercambio es válido, porque debido al mismo se forman una o dos figuras, los caramelos que las componen "explotan" y desaparecen. Y en su lugar caen los caramelos que estaban por encima de estos (esto es un proceso recursivo).

Ciertas figuras al explotar dejan en su lugar caramelos especiales, que cuando explotan generan explosiones en cascada de distinto tipo.

Cada caramelo que explota le otorga puntaje al usuario.

Desarrollo del trabajo

En Campus ITBA, en la carpeta Final del material didáctico de la materia, se encuentra el proyecto Java con los fuentes de esta aplicación.

Como primer paso el grupo debe analizar todos los fuentes para comprender el diseño y las clases que lo componen y entender cada detalle de funcionamiento.

Como segundo paso, el grupo deberá agregar nuevas funcionalidades a la implementación provista. Cada nueva funcionalidad estará implementada en un nivel de juego. El proyecto original contiene la clase `Level1` que representa un nivel de juego con características propias. Se deben agregar nuevas clases `Level2`, `Level3`, etc donde cada nivel permita utilizar las nuevas funcionalidades. Y respecto al *front-end* deberán agregar al menú de la aplicación una forma de elegir el nivel a ejecutar.

En todos los casos, se espera que la implementación de la nueva funcionalidad demuestre las ventajas del paradigma.

En caso de rendir en la primera fecha de final y formando un grupo de tres o menos alumnos, deberán implementar dos de las siguientes funcionalidades.

En caso de rendir en la primera fecha de final y formando grupo de cuatro alumnos, deberán implementar cuatro de las siguientes funcionalidades.

En caso de rendir en la segunda fecha de final deberán formar grupos de tres o menos alumnos e implementar cuatro de las siguientes funcionalidades. No se aceptan grupos de cuatro alumnos para la segunda fecha.

1. GoldenBoard: Convertir el tablero en oro

El objetivo de este nivel es que todo el tablero se convierta en oro. Para ello es necesario hacer intercambios. Intercambios horizontales válidos transforman la fila en cuestión en dorado. Intercambios verticales válidos transforman la columna en cuestión en dorado. Si una fila o columna ya es dorada y se realizan intercambios válidos sobre la misma, no hace nada. La cantidad de casilleros del tablero que restan convertirse en dorado deberá mostrarse en todo momento en el panel inferior, junto con el puntaje que ya se muestra en la implementación original.

Respecto al *front-end*, para lograr el efecto dorado en una posición del tablero que se muestra en la captura, se propone la siguiente modificación en el método `setImage` de la clase `BoardPanel`, que consiste en agregar un efecto de luz a la imagen existente:

```
public void setImage(int row, int column, Image image) {
        ...
        Light.Distant spotLight = new Light.Distant();
        spotLight.setColor(Color.YELLOW);
        spotLight.setElevation(100);
        Lighting lighting = new Lighting(spotLight);
        cells[row][column].setEffect(lighting);
        ...
}
```

2. Wall Blast

Se define en el tablero en una posición definida una pared compuesta por varias celdas contiguas. Cada celda de la pared sólo puede romperse por explosiones de *candys* especiales. Si la explosión de un candy especial pasa por una celda de la pared, esta celda se rompe. Por ejemplo la explosión de un caramelo especial horizontal eliminará todas las celdas de la pared que estén en la fila de la explosión. El total de celdas de la pared que restan romper deberá mostrarse en todo momento en el panel inferior, junto con el puntaje que ya se muestra en la implementación original. Cuando se rompen todas las partes de la pared del tablero, el nivel se da por ganado.

Respecto al *front-end*, para lograr el efecto de pared en una posición del tablero que se muestra en la captura, se propone una modificación similar a la propuesta para la funcionalidad **1. GoldenBoard: Convertir el tablero en oro** usando el color `Color.SANDYBROWN`

3. Time Bomb: Caramelos con contador

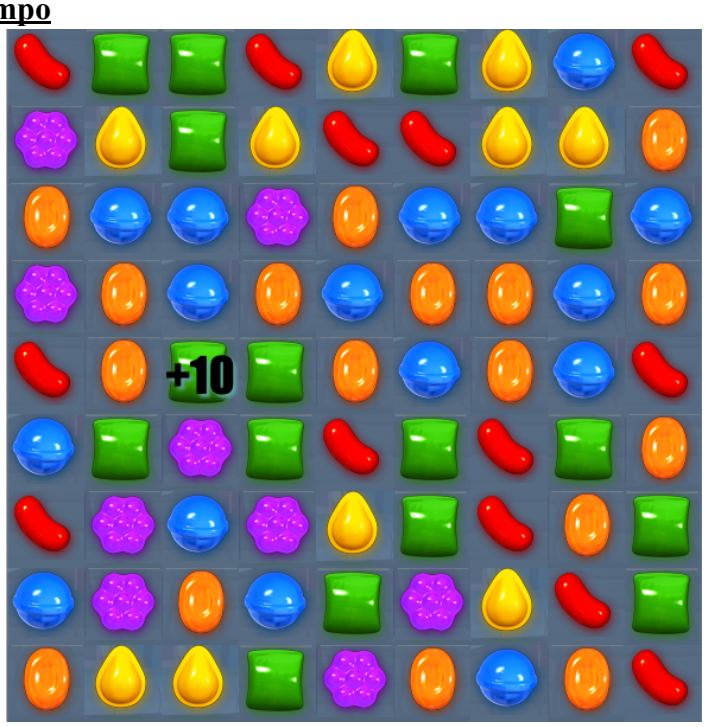
En este nivel algunos caramelos tendrán la particularidad de ser bombas de tiempo, es decir, estarán acompañados de un contador. Ante cada intercambio válido que se realice en cualquier parte del tablero, el contador de todos los caramelos bombas de tiempo que estén presentes en el tablero se restan en uno. El objetivo del nivel es eliminar los caramelos bomba de tiempo antes de que alguno de ellos llegue a cero, con cualquier intercambio válido que lo involucre. Si algún caramelo bomba de tiempo del tablero llega a cero, el nivel se da por perdido. La cantidad restante de movimientos (el mínimo de los contadores de todas las bombas de tiempo del tablero) deberá mostrarse en todo momento en el panel inferior, junto con el puntaje que ya se muestra en la implementación original. Deberán establecer una cantidad máxima fija de caramelos bomba que tendrá el nivel, los cuales deberán ir apareciendo a medida que se eliminar caramelos en el nivel.

Respecto al *front-end*, para superponer un texto sobre la imagen del caramelo en cierta posición del tablero como se muestra en la captura, se propone la siguiente modificación en la clase `BoardPanel`, que consiste en reemplazar la matriz de instancias de `ImageView` por una matriz de instancias de `javafx.scene.layout.StackPane` (un *layout* que tendrá dos componentes superpuestos: la imagen del caramelo -instancia de `ImageView`- y por encima el texto con el contador -instancia de `javafx.scene.text.Text`).

```
public void setImage(int row, int column, Image image) {
        cells[row][column].getChildren().add(new ImageView(image));
        ...
        DropShadow dropShadow = new DropShadow();
        dropShadow.setRadius(3.0);
        dropShadow.setOffsetX(3.0);
        dropShadow.setOffsetY(3.0);
        dropShadow.setColor(Color.ORANGERED);

        Text text = new Text("10");
        text.setFont(Font.font("Impact", FontWeight.BOLD, 40));
        text.setFill(Color.BLACK);
        text.setEffect(dropShadow);
        cells[row][column].getChildren().add(text);
        ...
}
```

4. Límite de Tiempo

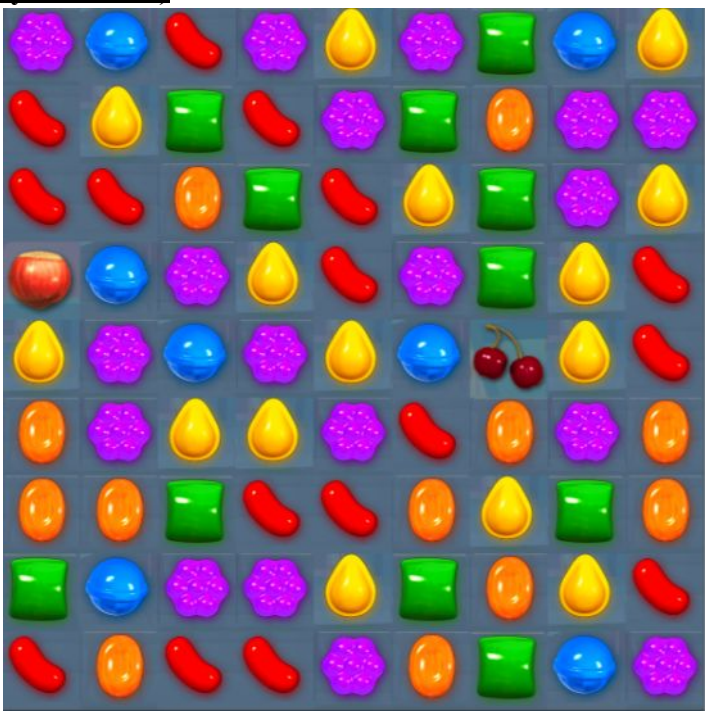


Este nivel tendrá un límite de tiempo, es decir arranca con un tiempo inicial y ante cada segundo que pase el tiempo restante del nivel disminuye hasta llegar a cero. El tiempo restante del nivel deberá mostrarse en todo momento en el panel inferior de puntaje, junto con el puntaje que ya se muestra en la implementación original. En este nivel algunos caramelos tendrán la particularidad de ser proveedores de tiempo, es decir, estarán acompañados de un número que representa una cantidad de segundos. Ante la eliminación de un caramelo proveedor de tiempo a partir de un intercambio válido que lo involucre, el tiempo que tenía este caramelo es adicionado al tiempo restante del nivel. El objetivo del nivel es eliminar los caramelos proveedores de tiempo para que el tiempo restante del nivel no llegue a cero. Cuando el tiempo del nivel se acaba, el nivel se da por terminado. Deberán establecer una cantidad máxima fija de caramelos proveedores de tiempo que tendrá el nivel, los cuales deberán ir apareciendo a medida que se eliminar caramelos en el nivel.

Respecto al *front-end*, para superponer un texto sobre la imagen del caramelo en cierta posición del tablero se propone una modificación similar a la propuesta para la funcionalidad **3. Time Bomb: Caramelos con Contador**. Y para actualizar el tiempo restante en el panel inferior pueden utilizar el siguiente fragmento de código que utiliza la clase `java.util.Timer` para actualizar el texto del `scorePanel` cada 1 segundo:

```
Timer timer = new Timer();
timer.scheduleAtFixedRate(new TimerTask() {
  @Override
  public void run() {
     Platform.runLater(new TimerTask() {
        @Override
        public void run() {
           scorePanel.updateScore(...);
        }
     });
  }
}, 0, 1000);
```

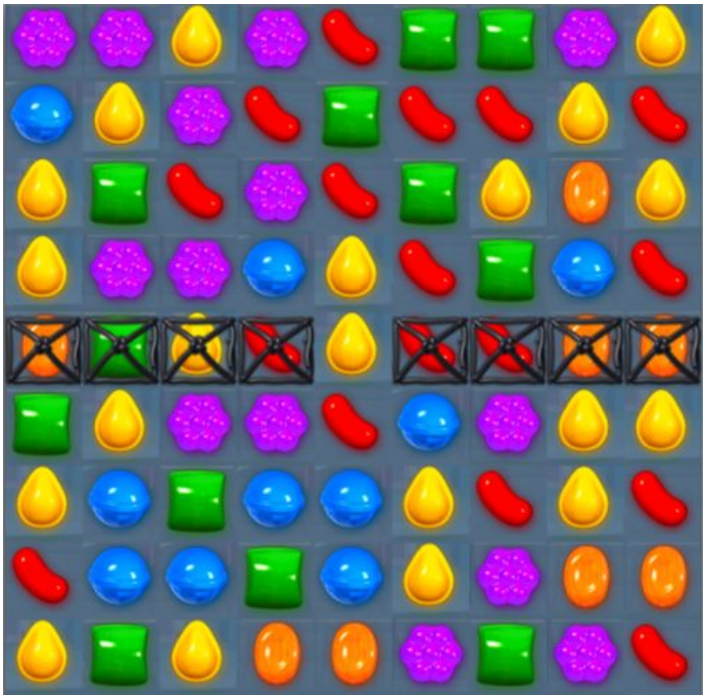
5. Fruta (Cherry y Hazelnut)



En el juego original, uno tiene que llevar todas las "frutas" a la parte inferior del tablero para poder ganar. Estas frutas no pueden generar combos, ningún *candy* especial las borra y solo desaparecen cuando llegan a la última fila del tablero. El objetivo del nivel es llevar a la última fila del tablero una cierta cantidad de frutas en menos de una cierta cantidad de movimientos. El número de frutas restantes y el número de movimientos restantes deberán mostrarse en todo momento en el panel inferior, junto con el puntaje que ya se muestra en la implementación original. Si la cantidad de movimientos restantes del nivel llega a cero y todavía quedan frutas por eliminar, el nivel se da por perdido.

Respecto al *front-end*, en el directorio `resources/` del proyecto están presentes las imágenes de la cereza y la avellana.

6. Jaula



Esta celda bloquea el caramelo que contiene e impide que el mismo se mueva. Un movimiento involucrando una jaula es considerado inválido. Para "liberarla" hay que combinar 3 o más caramelos del mismo color, donde uno sea el contenido por esta celda. A partir de ese momento la celda se comporta como una celda común. El número de jaulas restantes por liberar deberá mostrarse en todo momento en el panel inferior, junto con el puntaje que ya se muestra en la implementación original. Cuando se liberan todas las jaulas del tablero, el nivel se da por ganado. Si se realizan explosiones debajo de una jaula, como ésta impide la caída de caramelos, aparecerán celdas libres (sin caramelos) debajo de la misma.

Respecto al *front-end*, en el directorio `resources/` del proyecto está presente la imagen de la jaula y la misma se puede superponer a la imagen del caramelo.

Grupos

Los alumnos deben informar la conformación del grupo y la fecha de final elegida en el Foro de Discusión **"TPE Final"** de Campus ITBA antes del **04/12/2019 23:59 ART.**

Cada grupo deberá realizar la entrega mediante la actividad correspondiente en Campus ITBA:

- **En caso de rendir en la primera fecha de final (10/12/2019 14:00 ART) deberán entregar antes del 07/12/2019 23:59 ART**
- **En caso de rendir en la segunda fecha de final (20/12/2019 14:00 ART) deberán entregar antes del 17/12/2019 23:59 ART**

En ambos casos el grupo completo deberá presentarse el día y hora del final donde se le comunicará el resultado del mismo y rendir un coloquio, el cual podrá modificar la nota individual de los integrantes del grupo.

Para que el trabajo sea aceptado todos los integrantes del grupo deben estar inscriptos en fecha de final que acordaron en el armado del grupo.

En caso de inscribirse y no entregar en fecha, se calificará como ausente. Si algún miembro del equipo no se presenta en el horario del final fijado por Secretaría Académica el alumno será calificado como ausente, y deberá conformar otro grupo en otra fecha de final, no afectando la evaluación de los alumnos que se presenten.

Material a entregar

Se deberá entregar el proyecto con las modificaciones y además un informe que contenga:

- Enumeración y breve descripción de las funcionalidades agregadas.
- Explicación de todas las modificaciones realizadas al proyecto original
- Problemas encontrados durante el desarrollo
- Cambios hechos en la implementación provista por la Cátedra, justificando si fue para corregir un error de funcionamiento, mejorar la eficiencia o el estilo

Criterios de evaluación y calificación

No se aceptará el uso de bibliotecas de terceros, a excepción de la biblioteca estándar de Java. El código debe ser íntegramente de autoría propia. El uso de bibliotecas no autorizadas implicará la desaprobación.

El programa no debe abortar por ningún motivo y ante cualquier error se debe mostrar un mensaje adecuado.

Para la evaluación y calificación del trabajo especial se considerarán:

- el correcto funcionamiento del programa (recordar el uso de métodos de prueba de software)
- la modularización realizada
- el correcto diseño de las clases
- la separación del *front-end* y el *back-end*
- la claridad del código
- el cumplimiento de las reglas de estilo de programación dadas en clase

Las pruebas serán realizadas con la versión 8 de Java y JavaFX. Verificar que la versión de Java que utilizan para el desarrollo coincide con la pedida.

Dudas sobre el TPE

Las mismas deben volcarse al **Foro de Discusión "TPE Final" del Campus ITBA**.

Entorno de Desarrollo

Para poder ejecutar el código brindado es necesario que seteen en el IDE de su elección el *path* donde se encuentran las imágenes que utiliza el proyecto.

Por ejemplo, para el caso de IntelliJ IDEA, es necesario que en Project Structure, definan como carpeta de recursos al directorio `resources/`, como se indica en la captura a continuación: