Introducción a Ciencias de la Computación
2020-2
Práctica 8: Excepciones, Entrada y Salida

Pedro Ulises Cervantes González
confundeme@ciencias.unam.mx

Emmanuel Cruz Hernández
emmanuel_cruzh@ciencias.unam.mx

Adriana Sánchez del Moral
adrisanchez@ciencias.unam.mx

Víctor Zamora Gutiérrez
agua@ciencias.unam.mx

Fecha límite de entrega: 9 de Junio de 2020.
Hora límite de entrega: 23:59.

1. Objetivo

Manejar excepciones creadas por Java y creadas desde 0 para poder atrapar los fallos que pueden ocurrir durante la ejecución de un programa. Con esto, aprender a *cachar* los eventos inesperados para que un programa no termine abruptamente y pueda continuar con su ejecución, dando al programa una alternativa para poder recuperarse.

Además de saber cómo guardar archivos con datos de un programa, así como leer datos ya existentes para darle un estado a un programa al momento de ejecutar.

2. Actividad

2.1. Actividad 1 *(1 punto)*

Crea una clase llamada *InvalidCardException* que extienda la clase *Exception* e implementa el método constructor de esta clase.

2.2. Actividad 2 *(0.25 puntos)*

Crea una clase llamada *Carta* que representa una carta de juego con las siguientes características:

- Debe tener un atributo que represente el símbolo o la figura que guarda la carta.
- Debe tener un atributo que permita saber si la figura de la carta está visible o no.
- Debe tener un método que permita voltear la carta, esto es, cambiar el estado de visibilidad de la figura de una carta.
- Sobrescribe el método *toString()*. La representación de este objeto es la figura de la carta.

NOTA: Puedes agregar todos los atributos y métodos que consideres necesarios para esta clase.

2.3. Actividad 3

Crea una clase llamada *Memorama* que tenga un atributo que sea un arreglo bidimensional cuadrado de tamaño al menos 4x4, que alamacene objetos de tipo *Carta*.

2.4. Actividad 4 *(1 punto)*

Dentro de la clase *Memorama* implementa un método que permita verificar si dos cartas tienen el mismo símbolo. Si tienen el mismo símbolo regresa *true* y *false* en caso de no ser las mismas figuras. Este método se llamará *volteaCartas(..)* y puede recibir todos los parámetros que consideres necesarios.

2.5. Actividad 5 *(2.5 puntos)*

En la clase *Memorama* implementa un método llamado *juegaPartida* que permita realizar una partida. Una partida es mostrar la figura que contienen dos cartas seleccionadas, presionar una tecla para continuar y verificar si las cartas seleccionadas contienen la misma figura.

- Si tienen la misma figura, entonces la figura de ambas cartas queda visible durante el resto del juego.
- Si no tienen la misma figura, entonces las cartas vuelven a ocultar la figura.

Este método lanza la excepción *InvalidCardException* en los casos siguientes:

- La posición de la primera carta a voltear es igual a la posición de la segunda carta a voltear. Esto es querer voltear exactamente la misma carta.
- Intentar voltear una carta que ya está volteada, es decir, que la figura que almacena ya es visible. Sólo se pueden voltear cartas que tienen su figura oculta.

2.6. Actividad 6 *(0.25 puntos)*

Cuando se lleva a cabo una partida, el usuario debe tener las siguientes opciones:

- Jugar nueva partida: poner todas las cartas con su figura oculta.
- Jugar partida actual: mandar a llamar el método *juegaPartida*.
- Guarda partida: guardar en un archivo el estado actual del juego, esto implica guardar en un archivo la representación de las cartas que ya han sido volteadas y las que no.
- Cargar partida: leer de un archivo alguna partida que haya sido guardada, leyendo toda la información contenida en el archivo para recuperar el estado de alguna partida.

2.7. Actividad 7 *(2 puntos)*

Crea un método que permita guardar el estado actual de una partida, escribiendo en un archivo toda la información que consideres necesaria guardar.

2.8. Actividad 8 *(2 puntos)*

Crea un método que permita leer de un archivo la información de una partida guardada, logrando que el programa regrese al estado actual de la partida guardada.

2.9. Actividad 9 *(1 punto)*

La información de las cartas debe leerse de un archivo. Así que al crear una nueva carta, la información se debe obtener al leer la línea de un archivo.

3. Nota importante

Debes crea un menú que permita interactuar con el usuario, de tal forma que pueda elegir las operaciones a realizar, mostrando las entradas y las salidas de cada uno de los métodos.

Esta práctica se puede entregar en parejas. El formato de entrega es el siguiente:

- Apellido1Nombre1Apellido2Nombre2_09
  - src
    - InvalidCardException.java
    - Memorama.java

- Readme.txt

En caso de cambiar pareja con quien realizaron la práctica 7, uno de los integrantes debe enviar un correo a `emmanuel_cruzh@ciencias.unam.mx` a más tardar un día antes de la fecha de entrega de la práctica, mencionando el nombre de los nuevos integrantes.

El archivo *Readme.txt* debe contener una breve descripción de qué hizo cada integrante y cuál fue la forma en que se organizaron para realizar la práctica. **Ambos integrantes deben contribuir en la implementación de la práctica**.

Sólo una persona debe subir y entregar la práctica en su versión final a Classroom, el otro integrante sólo debe marcar la práctica como entregada.

4. Materiales para consultar

1. Concepto de excepciones: `https://youtu.be/fDmuSDRSDLQ`
2. Jerarquía de excepciones: `https://youtu.be/5pMdEGfC2V8`
3. Try... catch... finally: `https://youtu.be/mCmu7Ps55Dc`
4. Escritura de ficheros: `https://youtu.be/E0H4OzW2_1Y`
5. Lectura de ficheros: `https://youtu.be/etQN4EfYN7k`

5. Reglas Importantes

- **El programa debe ser robusto. En caso de fallar y terminar abruptamente la ejecución, se restará 1 punto de la calificación obtenida por cada falla**.
- Cumple con los lineamientos de entrega.
- Todos los archivos deberán contener nombre y número de cuenta.
- Tu código debe estar comentado. Esto abarca clases, atributos, métodos y comentarios extra.
- Para cada clase solicitada, crea un nuevo archivo.
- Utiliza correctamente las convenciones para nombrar variables, constantes, clases y métodos.
- Sólo se permite el uso de la biblioteca Scanner.
- **SÍ** se reciben prácticas con retraso. Por cada día se restará 1 punto.
- En caso de no cumplirse alguna de las reglas especificadas, se restará 0.5 puntos en tu calificación obtenida.