Introducción a Ciencias de la Computación I
2014-1
Práctica 7 - Herencia

Profesor: José de Jesus Galaviz Casas
Ayud. lab.: Roberto Monroy Argumedo

Fecha de entrega: 12 de octubre del 2013

1. Objetivos

Poner en práctica los conceptos de herencia y sobrecarga de métodos. Conocer y hacer uso de las clases `LinkedList` para listas ligadas y `Scanner` para la lectura de archivos de texto con formato, ambas clases forman parte del API de java.

2. Desarrollo

En la práctica 3 se escribieron clases para abstraer algunos elementos del juego de tarjetas coleccionables Magic. Retomando la idea, observamos que las clases tenian elementos en común, por lo que utlizaremos el concepto de herencia, visto en clase, para mejorar la abstracción.

3. Actividades

- (1 pto.) Escribe una clase `Tarjeta` la cual abstraerá los campos que tengan en común las tarjetas. Escribe un constructor para la clase `Tarjeta` con los parámetros necesarios para inicializar todos los atributos.
- (1 pto.) Utilizando el concepto de sobrecarga, escribe los métos `equals(Object obj)` y `toString()` para la clase `Tarjeta`, estos métodos se heredan de la clase `Object`.

- (3 ptos.) Haciendo uso del concepto de herencia, escribe las clases `Tierra`, `Hechizo` y `Criatura` las cuales serán subclases de `Tarjeta`. Sobrecarga el método `toString()`, heredado de `Tarjeta`, y agrega los nuevos atributos de cada tipo de tarjeta.
- Las clases `Tarjeta`, `Tierra`, `Hechizo` y `Criatura` se deben encontrar dentro del paquete `tarjetas`.
- Crea una clase `Main` fuera del paquete, la clase tendrá dos métodos estáticos:
  - (2.5 ptos.) `LinkedList<Tarjeta> cargaMazo(String nombreArchivo)`

    El método cargará desde un archivo de texto plano un número no determinado de tarjetas y devolverá en una lista ligada las tarjetas.

    El nombre del archivo estrará indicado en el parámetro `nombreArchivo`.

    El archivo debe de ser leido haciendo uso de las clases `Scanner` del paquete `java.util`.

    En cada linea del archivo se encontrará la información de una tarjeta, cada campo separado por un punto y coma, dependiendo del tipo de tarjeta, se podrán encontrar de la siguiente forma:
    - Tierra:
      T; nombre; edicion; color; numero tarjeta; costo generico; costo color;
    - Hechizo:
      H; nombre; edicion; color; numero tarjeta; costo generico; costo color; efecto; leyenda;
    - Criatura:
      C; nombre; edicion; color; numero tarjeta; costo generico; costo color; ataque; defensa; efecto; leyenda; tipo

    Ejemplo:

    H; Neblina; Septima; Green; 245; 0; 1; Preven todo el daño de combate que se fuera a hacer este turno;;

- (2.5 ptos.) `void main(String[] args)`

  El método main, el cual usará el método `cargaMazo` para cargar un mazo de tarjetas y posteriormente imprimirá en pantalla cada elemento de la lista ligada usando el método `toString()`, el nombre del archivo se indicará por la linea de comando.

4. Observaciones

Puedes consultar la documentación de las clases Scanner y LinkedList en linea, donde encontrarás más métodos que los que vimos en clase que te pueden ser útiles.

5. Entrega

Al finalizar deberás enviar por correo electrónico con asunto [Practica07] el código fuente generado en un archivo tar con la siguiente estructura:

```
loginAda
└── practica07
    ├── src
    │   ├── tarjetas
    │   │   ├── Criatura.java
    │   │   ├── Hechizo.java
    │   │   ├── Tarjeta.java
    │   │   └── Tierra.java
    │   └── Main.java
    └── README
```

En donde loginAda lo sustituirás por tu login y README es un archivo de texto plano en donde se encontrará tu nombre completo comenzando por tus apellidos.

Recuerda que debes respetar las convenciones de código y que éste debe de estar completamente documentado.