Laboratorio 2: Arreglos y Excepciones

# Análisis del Sistema (30 puntos)

## Requisitos funcionales del sistema (5 pts)

Escribir los puntos exactos que debe cumplir tu sistema.

- El programa debe ser capaz de generar parejas de datos en un tablero que pueda mostrar.

- El programa debe ser capaz de identificar a dos jugadores y separarlos por turnos.

- El programa debe ser capaz de analizar códigos ASCII para generar emoticones.

## Clases necesarias y su propósito (5 pts)

|  |  |
| --- | --- |
| Clase | Propósito |
| Main | Instancia constructora del juego. |
| Controlador | Mantendrá el orden lógico del programa. |
| Consola | Se encargará de mostrar y responder al usuario. |
| Carta | Mantendrá la información de cada carta: símbolo, mostrado, emparejado. |
| Tablero | Organizará las cartas en forma cuadrada. |
| Jugador | Mantiene la información de los jugadores. Nombre, puntos y juegos ganados. |

* 1. **Atributos de cada clase (10 pts)**

Main:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| memoria | Controlador | publico | Inicio del juego |

Controlador:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| jugador1  jugador2  tablerito  consolita | Jugador  Jugador  Tablero  Consola | Private  Private  Private  Private final | Guardar información del jugador 1  Guardar información del jugador 2  Tablero del juego |

Consola:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Sc  Imp | Scaner  System.out | Private final  Private final | Variable para realizar lectura en terminal.  Variable para facilitar impresión en terminal. |

Carta:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Símbolo  Revelado  Emparejado | String  Boolean  Boolean | Private final  Private  Private | Emoticono que representa la tarjeta.  Si se ha seleccionado en el turno.  Si ya se determinó como pareja encontrada. |

Jugador:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Nombre  Puntos  Victorias | String  Int  Int | Private final  Private  Private | Nombre del jugador asignado  Puntos conseguidos en la ronda  Rondas ganadas. |

Tablero:

|  |  |  |  |
| --- | --- | --- | --- |
| Atributo | Tipo de dato | Visibilidad | Propósito |
| Dimensiones  casillasTarjetas  ParejasTotales  ParjeasListas | Int  Carta[][]  Int  Int | Private  Private  Private  Private | Obtener largo y ancho del tablero.  Matriz del tablero. Con datos de dimensiones.  Total de parejas generadas.  Parejas que ya se encontraron. |

## Métodos de cada clase (10 pts)

Controlador:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atributo | Método | Parámetros : Tipo de dato | Visibilidad | Propósito |
| Void  Void  Void | Controlador  Turno  Jugar | (Void)  (Void)  (Void) | Public  Public  Public | Constructor  Se active cada turno y mantiene el control del juego  Se utiliza para iniciar un nuevo juego. |

Consola:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atributo | Método | Parámetros : Tipo de dato | Visibilidad | Propósito |
| Void | Consola | (void) | Public | Constructor. Se establecen los atributos.  El resto de métodos se encargan de la interacción con el usuario en la terminal. |

Jugador:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atributo | Método | Parámetros : Tipo de dato | Visibilidad | Propósito |
| String  Void  Int  Void  Int  Void | getNombre  setNombre  getPuntos  setPuntos  getVictorias  setVictorias | (void)  (String nombre)  (void)  (int modificador)  (void)  (int modificador) | Public  Public  Public  Public  Public  Public | Modificar nombre  Obtener nombre  Modificar puntos  Obtener puntos  Modificar victorias  Obtener victorias |

Carta:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atributo | Método | Parámetros : Tipo de dato | Visibilidad | Propósito |
| Boolean  Void  Boolean  Void  Char  Void | getSeleccionada  Seleccionar  getEmparejada  Emparejar  getSimbolo  quitarSeleccion | (Void)  (Void)  (Void)  (Void)  (Void)  (Void) | Public  Public  Public  Public  Public  Public | Revisar si ya fue seleccionada.  Seleccionar carta  Revisar si ya está emparejada.  Emparejar carta  Obtener símbolo de la carta.  Quita selección de carta. |

Tablero:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Atributo | Método | Parámetros : Tipo de dato | Visibilidad | Propósito |
| Int  Tarjeta  Int  Int  Void | getDimensiones  getTarjeta  getParejasTotales  getParjeasListas  setParjeasListas | (Void)  (int x, int y)  (Void)  (Void)  (int modificador) | Public  Public  Public  Public  Public | Obtener tamaño del tablero.  Obtener tarjeta específica.  Obtener parejas totales de tarjetas.  Obtener parejas encontradas.  Modificar parejas encontradas. |

# 

1. **Diseño: Diagrama de Clases (30 puntos)**
   * Asegúrate de mostrar atributos y métodos con visibilidad (+, -).
   * Indica relaciones entre clases (asociación, agregación, etc.).
   * Incluye el driver program (Main).

Diagrama de clases aquí o adjunto en un archivo aparte.

# Programa (40 puntos)

En cada archivo `.java`, asegurarse de incluir:

* + Las clases necesarias.
  + Uso adecuado de objetos.

Menú que debe implementar el driver program:

* + 1. Nuevo comprador
    2. Nueva solicitud de boletos
    3. …
    4. …
    5. Salir

GitHub: colocar aquí la URL: <https://github.com/Hola2212/Lab2_Arreglos_y_Excepciones.git>

# Checklist antes de entregar

⬛✓⬛ Está claro el análisis?

⬛✓⬛ El diagrama tiene los elementos UML correctamente?

⬛✓ Subiste tu código a GitHub con todo lo necesario?