Herramientas Informáticas para Juegos de Azar

Curso 2021/2022
Facultad de Informática
Universidad Complutense de Madrid

Práctica

El valor de una mano

Profesor de teoría: Manuel Núñez
Profesor de prácticas: Alberto Núñez

1.- Objetivo

Esta práctica consiste en calcular **el valor de una mano** en el juego NLHE.

2.- Descripción

Para desarrollar la práctica se hará uso del IDE *Netbeans* o *Eclipse*. La práctica se desarrollará enteramente en **Java**.

Esta práctica está formada por 3 apartados obligatorios y 2 opcionales, de tal forma que el 70% de la calificación corresponde a la parte obligatoria y el 30% restante a la parte opcional. Para poder realizar la parte opcional **es necesario** haber realizado la parte obligatoria. No se tendrán en cuenta las partes opcionales realizadas cuando la parte obligatoria no esté completamente implementada.

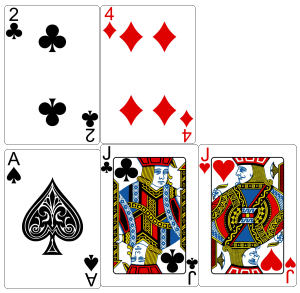
Aunque el objetivo de cada apartado es siempre el mismo, calcular el valor de una mano, las condiciones de cada uno variarán incrementando la complejidad de su cálculo.

El diseño e implementación de cada apartado es libre, siendo la única limitación impuesta los parámetros de entrada de la aplicación.

Cada carta se procesará inicialmente con un *String* de, exactamente, dos caracteres:

- El primer carácter indicará el valor de la carta (de mayor a menor): **A**, **K**, **Q**, **J**, **T**, **9**, **8**, **7**, **6**, **5**, **4**, **3** y **2**.
- El Segundo carácter representa el palo de dicha carta, siendo 4 las posibles opciones: **h** (corazones), **d** (diamantes), **c** (tréboles) y **s** (picas).

Dos ejemplos de la representación de cartas con la notación descrita:



"2c4d"

"AsJcJh"

Aunque las cartas se "lean" con un *String*, se permite el uso de estructuras de datos para poder procesarlas antes de emitir un resultado. El diseño y uso de estas estructuras es libre y queda a elección del alumno.

2.1 Indicar la mejor jugada dadas 5 cartas (obligatorio)

En este caso se pretende calcular la mejor jugada dadas 5 cartas. Además, se debe indicar los posibles ***draws***. Un *draw* es una mano que no está completa pero se puede completar con cartas adicionales. Por ejemplo, 4 cartas del mismo palo o 4 cartas tales que con una mas pueden formar escalera. En el segundo caso distinguiremos entre 4 cartas seguidas (open-ended) y 4 cartas con un hueco (*gutshot*).

Ejemplo para **AhAcQhJhTh**

- Pareja de ases.
- Gutshot.
- Flush draw.

La entrada para este apartado será un fichero de **texto plano** donde cada línea contiene exactamente 10 caracteres sin espacios, 2 por cada carta. La salida de este apartado será otro fichero de **texto plano** donde, para cada una de las manos procesadas, se indica tanto la mejor mano como los posibles *draws*.

Ejemplo de fichero de entrada "entrada.txt":

```
AhAcQhJhTh
```

Ejecución:

```
$> java –jar nombreProyecto.jar 1 entrada.txt salida.txt
```

Siendo el primer parámetro (1) el número del apartado que se desea probar, el segundo parámetro (entrada.txt) el fichero de entrada que contiene las manos y el tercer parámetro (salida.txt) el fichero de salida con los resultados.

Ejemplo de fichero de salida "salida.txt" para la ejecución anterior:

```
AhAcQhJhTh
 - Best hand: Pair of Aces (AhAc)
 - Draw: Straight Gutshot
 - Draw: Flush
```

El fichero de entrada podrá contener **varias manos**, siempre estando cada mano en una línea diferente del fichero. Para una mejor lectura del fichero de salida, se deberá dejar una línea en blanco después de los resultados de cada mano.

2.2 Indicar la mejor jugada teniendo 2 cartas propias y 3, 4 ó 5 cartas comunes. (Obligatorio)

En este apartado se deberá calcular la mejor mano partiendo de las 2 cartas del jugador, y las 3 (flop), 4 (turn) ó 5 (river) cartas comunes existentes en la mesa. **Cuando haya 5 cartas comunes en la mesa, no se calcularán los draws.**

En este caso, el fichero de entrada tendrá una estructura similar a la del apartado anterior. La sintaxis de cada línea del fichero de entrada será:

```
Carta_1Carta_2;n;CartasComunes
```

donde:

- `Carta_1` es la primera **carta** del jugador.
- `Carta_2` es la segunda **carta** del jugador.
- `n` es el **número** de cartas comunes en la mesa (board), siendo $3 \leq n \leq 5$.
- `CartasComunes` serán las n **cartas** de la mesa, habiendo 2n caracteres para representarlas, sin espacios.
- Los caracteres "`;`" se utilizan únicamente como separadores para facilitar la lectura de los datos.

Un ejemplo de fichero de entrada "entrada2.txt" para este apartado podría ser:

```
AhAc;3;QhJhTh
AhAc;5;As2s3h4dJd
```

Ejecución:

```
$> java –jar nombreProyecto.jar 2 entrada2.txt salida2.txt
```

Siendo el primer parámetro (2) el número del apartado que se desea probar, el segundo parámetro (entrada2.txt) el fichero de entrada que contiene las manos y el tercer parámetro (salida2.txt) el fichero de salida con los resultados.

Ejemplo de fichero de salida "salida2.txt" para la ejecución anterior:

```
AhAc;3;QhJhTh
 - Best hand: Pair of Aces with AhAcQhJhTh
 - Draw: Straight Gutshot
 - Draw: Flush

AhAc;5;As2s3h4dJd
 - Best hand: Three of a kind (Aces) with AhAcAs4dJd
```

El fichero de entrada podrá contener **varias manos**, siempre estando cada mano en una línea diferente del fichero. Para una mejor lectura del fichero de salida, se deberá dejar una línea en blanco después de los resultados de cada mano.

2.3 Dados **n jugadores** (con 2 ≤ n ≤ 9) con 2 cartas cada uno y 5 cartas comunes, **ordena** los jugadores de mejor a peor mano. (Obligatorio)

En este apartado se deberá tener en cuenta a los demás jugadores de la mesa. Cada jugador tendrá 2 cartas propias y en la mesa siempre habrá 5 cartas comunes. El número de jugadores podrá variar de 2 a 9. Se pide **ordenar a los jugadores** teniendo en cuenta su jugada, es decir, de mejor a peor.

La sintaxis de cada línea del fichero de entrada será:

`N;J1CartasJug1;J2CartasJug2;…;JNCartasJugN;CartasComunes`

Donde:

- `N` es el número de jugadores presentes en la mano, de forma que 2 ≤ N ≤9.
- `J1, J2, …, JN son los identificadores para cada jugador.`
- `CartasJug1` son las **dos** cartas del jugador 1.
- `CartasJug2` son las **dos** cartas del jugador 2
- `CartasJugN` son las **dos** cartas del jugador N
- `CartasComunes` serán las **5 cartas** de la mesa, habiendo 10 caracteres para representarlas, sin espacios.
- Los caracteres ";" se utilizan únicamente como separadores para facilitar la lectura de los datos.

Un ejemplo de fichero de entrada "entrada3.txt" para este apartado podría ser:

```
4;J1AhAc;J2JsJh;J37c8c;J44sKc;5dKs6cTh9h
```

Ejecución:

```
$> java –jar nombreProyecto.jar 3 entrada3.txt salida3.txt
```

Siendo el primer parámetro (3) el número del apartado que se desea probar, el segundo parámetro (entrada3.txt) el fichero de entrada que contiene las manos y el tercer parámetro (salida3.txt) el fichero de salida con los resultados.

Ejemplo de fichero de salida "salida3.txt" para la ejecución anterior:

```
4;J1AhAc;J2JsJh;J37c8c;J44sKc;5dKs6cTh9h
J3: 6c7c8c9hTh (Straight)
J1: AhAcKsTh9h (Pair of Aces)
J4: 4sKcKsTh9h (Pair of Kings)
J2: JsJhKsTh9h (Pair of Jacks)
```

El fichero de entrada podrá contener **varias manos**, siempre estando cada mano en una línea diferente del fichero. Para una mejor lectura del fichero de salida, se deberá dejar una línea en blanco después de los resultados de cada mano.

2.4 Indicar la mejor mano para la modalidad de **Omaha** (Opcional)

En este apartado se calculará la mejor jugada para la modalidad Omaha. Cada jugador recibe 4 cartas. Las cinco cartas comunes se conforman como en NLHE (flop, turn y river). Una mano de Omaha se forma con **exactamente 2 cartas propias** y **exactamente 3 cartas comunes**. Dadas **4 cartas propias** y **3/4/5 cartas comunes**, indica la **mejor jugada** y los posibles **draws**.

La sintaxis de cada línea del fichero de entrada será:

```
Carta_1Carta_2Carta_3Carta_4;n;CartasComunes
```

Donde:

- `Carta_1` es la primera **carta** del jugador
- `Carta_2` es la segunda **carta** del jugador
- `Carta_3` es la tercera **carta** del jugador
- `Carta_4` es la cuarta **carta** del jugador
- `n` es el **número** de cartas comunes en la mesa (board), siendo $3 \leq n \leq 5$
- `CartasComunes` serán las n **cartas** de la mesa, habiendo 2n caracteres para representarlas, sin espacios.
- Los caracteres "`;`" se utilizan únicamente como separadores para facilitar la lectura de los datos.

Un ejemplo de fichero de entrada "entrada4.txt" para este apartado podría ser:

```
AhAc8s5h;4;2h2d2c2s
```

Ejecución:

```
$> java –jar nombreProyecto.jar 4 entrada4.txt salida4.txt
```

Siendo el primer parámetro (4) el número del apartado que se desea probar, el segundo parámetro (entrada4.txt) el fichero de entrada que contiene las manos y el tercer parámetro (salida4.txt) el fichero de salida con los resultados.

Ejemplo de fichero de salida "salida4.txt" para la ejecución anterior:

```
AhAc8s5h;4;2h2d2c2s
 - Best hand: 2's full of Aces with 2h2d2cAhAc
```

El fichero de entrada podrá contener **varias manos**, siempre estando cada mano en una línea diferente del fichero. Para una mejor lectura del fichero de salida, se deberá dejar una línea en blanco después de los resultados de cada mano.

2.5 Desarrollar una interfaz gráfica (Opcional)

Desarrollar una GUI (Graphical User Interface) para los apartados anteriores.

La GUI a desarrollar deberá contemplar, al menos, las siguientes funcionalidades:

- Carga del fichero de entrada.
- Selección del apartado a probar.
- Mostrar por pantalla tanto el Board como las cartas de cada jugador.
- Mostrar en un cuadro de texto (no por consola) el resultado de cada mano.

El diseño y la implementación de esta GUI es libre, siendo los únicos requisitos establecidos que cumpla la funcionalidad anteriormente descrita y lleve a cabo los apartados anteriores.

Los gráficos para representar las cartas se pueden descargar del siguiente link:
https://code.google.com/p/vector-playing-cards/

3.- Fecha de entrega

La práctica deberá entregarse antes del día **6 de octubre de 2021.** En la clase de laboratorio del día **7 de octubre será evaluada**. Para ello, deberá asistir el grupo completo. No se permitirá la entrega de prácticas fuera del plazo establecido.

4.- Modo de entrega

La práctica deberá entregarse a través del Campus Virtual, mediante el entregador habilitado para esta práctica.

No se tendrán en cuenta aquellas prácticas enviadas por otros medios que no sean el entregador habilitado, como por ejemplo, enviar la práctica como fichero adjunto en un e-mail.

Una vez finalizada la práctica, ésta se entrega mediante **un único fichero** comprimido con extensión ".zip". Dicho fichero deberá contener:

a) El fichero **alumnos.txt**. El **nombre y apellidos** de cada miembro del grupo que haya desarrollado esta práctica deberá estar presente en este fichero.

b) Todos los ficheros que formen el proyecto (código fuente, directorios, ficheros de configuración, etc.) así como librerías externas o imágenes utilizadas.

NOTA: Se recomienda comprobar que, una vez generado el fichero comprimido ".zip", el proyecto de la práctica puede abrirse correctamente. Esto se comprueba descomprimiendo el fichero y abriendo el proyecto correspondiente con *NetBeans*. En algunos casos, ficheros de configuración ocultos no se incluyen en el fichero comprimido, de forma que el proyecto no puede abrirse.