Fundamentos de la Programación

Presentación de la Práctica 1 (Versión 2)

(Basado en la práctica de Mercedes Gómez, Luis Hernández, Ramón González y Federico Peinado)

Índice

1. Introducción

- ✓ En esta práctica seguimos trabajando con el juego **las siete y media**.
- ✓ La versión añade a la práctica la utilización del bucle **for** y de los **arrays** de tipos simples.

2. Funcionalidad de la Aplicación (Versión 2)

- ✓ Cada versión avanza en objetivos y temario del curso.
- ✓ El mazo de 40 cartas sigue estando en un **fichero de texto**.
- ✓ En esta segunda versión, el **menú** permite seleccionar un tercer modo de juego: el **modo C**. Las diferencias de este modo de juego con los anteriores son:
  - Ninguno de los jugadores tendrá limitación en el **número de cartas** que puede coger (mas allá de las que existan en el mazo en cada momento).
  - La máquina se plantará si su puntuación supera la del humano o, en caso de empate, si la **probabilidad de pasarse** supera el 50%.

- ✓ Para que la máquina pueda estimar como de probable es que se pase, es necesario llevar un recuento de **cuantas cartas de cada tipo** quedan en el mazo.
- ✓ Supongamos una partida en la que el humano ha robado tres cartas: un 3, un 2 y una figura. Su puntuación sería 5.5 y el recuento de cartas que quedan en el mazo sería el siguiente:

figura	uno	dos	tres	cuatro	cinco	seis	siete
11	4	3	3	4	4	4	4

- ✓ Ahora suponemos que en el turno de la máquina ésta llega a un estado en el que ha robado: un 3, un 1 y tres figuras. Su puntuación también sería 5.5 por lo que estarían empatados.
- ✓ Podemos conocer la probabilidad que tendrá la máquina de pasarse si robara una carta más. Es decir, la probabilidad de robar una carta >= 3.

- Esta probabilidad se calcula sabiendo que quedan 32 cartas en el mazo y que estas son:

figura	uno	dos	tres	cuatro	cinco	seis	siete
8	3	3	2	4	4	4	4

- **probabilidad (sacar >=3)** = (2 treses + 4 cuatros + 4 cincos + 4 seises + 4 sietes) / 32 = 0.56.
- En este ejemplo, la máquina decidiría no arriesgarse y que fuese el azar quien decidiese quien gana.
- Supongamos ahora que el humano roba un 4 y se planta. Y que la máquina también roba un 4. En ese momento, las cartas que quedarían en el mazo serían:

figura	uno	dos	tres	cuatro	cinco	seis	siete
12	4	4	4	2	4	4	4

- ✓ Y la probabilidad de que la máquina se pase si roba una carta más sería:
- ✓ **probabilidad (sacar >=4)** = (2 cuatros + 4 cincos + 4 seises + 4 sietes) / 38 = 0.37.
- ✓ En este ejemplo, la máquina asumiría el riesgo y robaría, al menos, otra carta más.

3. Detalles de Implementación (Versión 2)

- ✓ A continuación se dan algunas indicaciones para implementar la segunda versión.
- ✓ Los **archivos de texto** con 40 cartas contienen 40 números. Cada número aparece en una línea. No hay información acerca de los palos.
- ✓ Incluye un tipo array *tCartasPorAparecer* cuyas variables permitan llevar un recuento de cuantas cartas de cada tipo quedan en el mazo.

✓ Incorpora, además, los siguientes subprogramas:

- `float modoChumano (ifstream &file, tCartasPorAparecer cartas) // Permite realizar el turno del jugador humano en el modo C. Recibe el archivo con el mazo y una variable cartas que indica cuántas cartas de cada tipo quedan, y devuelve los puntos obtenidos y actualiza cartas de acuerdo con las cartas que haya robado el humano.`
- `float modoCmaquina (ifstream &file, tCartasPorAparecer cartas) // Permite realizar el turno del jugador máquina en el modo C. Recibe el archivo con el mazo, una variable cartas que indica cuántas cartas de cada tipo quedan y devuelve los puntos obtenidos y actualiza cartas de acuerdo con las cartas que haya robado la máquina.`
- `bool esProbablePasarse (float puntosMaquina, const tCartasPorAparecer cartas) // Determina si la probabilidad que tiene la máquina de pasarse si robara una carta más es mayor que 0.5. Recibe la puntuación actual de la máquina y una variable cartas que indica cuántas cartas de cada tipo quedan, y devuelve true si la probabilidad de pasarse si roba una carta más supera 0.5 y false en caso contrario.`