Fundamentos de la Programación

Ayuda de la Práctica 1 (Versión 2)

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

Índice

1. Introducción

- ✓ Con esta presentación se pretende ayudar en la implementación de la versión 2 de la práctica 1.
- ✓ Necesario: un mínimo de práctica con **bucles for** y **arrays**.
- ✓ ¡Practica con los ejercicios del tema 3 si no lo has hecho ya!
- ✓ Esta ayuda es opcional. No es obligatorio seguirla para implementar la solución de la versión 2.

2. Planificación

- ✓ Para realizar la versión 2 en el tiempo estimado, aconsejamos cumplir la siguiente planificación:
  - 7/11: Proyecto, Modificaciones y Modo C Humano
  - 14/11: Modo C Maquina y Pruebas

3. Proyecto y Modificaciones

- ✓ Crea un **nuevo proyecto** en Visual Studio para la versión 2.
- ✓ Copia el archivo *main.cpp* de la versión 1 en el directorio del nuevo proyecto de Visual Studio.
- ✓ Añade tu archivo *main.cpp* al proyecto desde el explorador de la solución (en la carpeta de fuentes).
- ✓ Pon un comentario en el archivo *main.cpp* indicando que se trata de la solución de la versión 2 de la práctica 1.
- ✓ No te olvides de añadir el comentario con el nombre y apellidos de los integrantes del grupo de laboratorios.

- Modifica el programa *main.cpp* añadiendo los prototipos de las nuevas funciones

```
bool quedanCartas(const tCartasPorAparecer cartas);
float modoChumano(ifstream &file, tCartasPorAparecer cartas);
bool esProbablePasarse(float machineScore, const tCartasPorAparecer cartas);
float modoCmaquina(ifstream &file, tCartasPorAparecer cartas, float puntosHumano);
```

- Prepara las funciones (sin código) después de la función *main*.
- En esta versión, no es necesario modificar la función *main()* porque la lógica principal del juego sigue siendo la misma.
- Modifica la función *menu()* incluyendo una opción 3 (Play Modo C).

- ✓ Define el tipo array *tCartasPorAparecer* con tamaño 10. Define dicho tamaño con una constante.
- ✓ La constante y la definición del tipo array deben aparecer antes de los prototipos de las funciones.
- ✓ Modificaciones en la función *juego()*:
  - Declara una variable del tipo *tCartasPorAparecer* e inicializalá dando 4 cartas a cada una de sus posiciones. La posición 0 del array es para la carta 1. Y la última para la carta 12.
  - Añade la llamada al modo de juego C donde corresponda (cuando se ejecuta el modo A o el modo B debe poderse ejecutar también el modo C dependiendo de la opción).

4. Implementación de la Función QuedanCartas

- ✓ Esta función cuenta el número total de cartas y devuelve true si ese número es positivo

```
Declarar e inicializar total = 0
Desde la posición i = 0 hasta la última del array cartas
   Sumar al total el elemento cartas[i]
Devolver (total > 0)
```

5. Implementación de la Función ModoCHumano

```
Inicializar variables
Mientras quedanCartas(cartas), no se haya pasado y no haya stop
    Leer una carta del fichero
    Si el valor de la carta > 7
       Sumar 0.5 a la puntuación
       Actualizar el array de cartas quitando la carta
    Sino
       Sumar la carta a la puntuación
       Actualizar el array de cartas quitando la carta
    Mostrar la carta y la puntuación actualizada
    Actualizar contador de cartas
    Si (puntuación > 7.5)
          se ha pasado
    Sino
          Si quedanCartas(cartas)
              Preguntar si desea plantarse
              Actualizar stop según la respuesta
Devolver la puntuación
```

6. Implementación de la Función EsProbablePasarse

- ✓ Esta función estima la probabilidad de pasarse devolviendo true si la estimación es mayor que 0.5

```
cartasMalas = 0, total = 0
diferencia = 7.5 - machineScore
Desde la posición i = 0 hasta la última del array cartas
   Actualizamos total sumando cartas[i]
   Si (i < 7) y (diferencia < i + 1)
      Actualizamos cartasMalas sumando cartas[i]
   Si (i >= 7) y (diferencia < 0.5)
      Actualizamos cartasMalas sumando cartas[i]
estimacion = (double)cartasMalas / total
Devolver (estimación > 0.5)
```

7. Implementación de la Función ModoCMaquina

```
Inicializar variables
Mientras quedanCartas(cartas), no se haya pasado y no haya stop
    Leer una carta del fichero
    Si el valor de la carta > 7
       Sumar 0.5 a la puntuación
       Actualizar el array de cartas quitando la carta
    Sino
       Sumar la carta a la puntuación
       Actualizar el array de cartas quitando la carta
    Mostrar la carta y la puntuación actualizada
    Actualizar contador de cartas
    Si (puntuación > 7.5)
          se ha pasado
    Sino
          Si quedanCartas(cartas)
              stop = (puntuación == 7.5) o (puntuación > puntuación
                 del humano) o ((puntuación == puntuación del humano)
                 y esProbablePasarse(puntuacion, cartas))
Devolver la puntuación
```