Actividad de Evaluación Continua 2

Práctica 2 de programación con C++

Alumno: Alexander Sebastian Kalis
Profesor: Javier Llorente Ayuso

4 de enero de 2026

Índice

1. Introducción y Metodología

En este proyecto se ha desarrollado un simulador del juego "Poker con dados"para dos jugadores. El desarrollo se ha basado en la metodología de programación estructurada y, especialmente, en la descomposición modular. El objetivo principal es aplicar el uso de funciones, estructuras de datos (como el tipo `struct` y vectores), tipos enumerados y la gestión de múltiples archivos de código para crear una aplicación robusta y escalable.

2. Diseño del Algoritmo y Estructura

El programa se ha dividido en tres módulos lógicos, cada uno con su archivo de cabecera (`.h`) y su archivo de implementación (`.cpp`), lo que permite separar las responsabilidades de cada parte del sistema.

2.1. Módulo Dado

Se encarga de simular el comportamiento de un dado de poker de 6 caras.

- **resultadoDado**: Es un tipo enumerado que define los valores posibles: As, King, Queen, Jota, Diez y Nueve.
- **tirarDado()**: Utiliza la función `rand()` para generar un número entre 1 y 6, devolviendo su equivalente en el tipo enumerado.
- **mostrarDado()**: Traduce el valor interno del enumerado a texto legible para el usuario.

2.2. Módulo Jugador

Gestiona la información de los participantes.

- **Estructura Jugador**: Almacena el nombre del usuario, sus puntos en la ronda actual y el histórico de victorias en la partida.
- **Funciones**: Incluye la solicitud de datos por teclado y la actualización del contador de victorias cuando un jugador gana una ronda.

2.3. Módulo Tirada

Es el núcleo lógico que procesa los lanzamientos.

- **Tipo Tirada**: Definido mediante un `typedef` como un vector de 5 elementos de tipo `resultadoDado`.
- **Cálculo de Puntos**: Implementa un algoritmo de comparación cruzada para contar parejas. Según el número de coincidencias encontradas, se asignan puntos siguiendo la jerarquía oficial (desde Pareja hasta Repoker).

3. Código Fuente

3.1. Módulo Dado

```
#ifndef DADO_H
#define DADO_H

enum resultadoDado { as = 1, king, queen, jota, diez, nueve, otro };

resultadoDado tirarDado();
void mostrarDado(resultadoDado carta);

#endif
```

Listing 1: dado.h

```cpp
#include "dado.h"
#include <iostream>
#include <cstdlib>

using namespace std;

resultadoDado tirarDado() {
    int numeroDado = 1 + rand() % 6;
    return (resultadoDado)numeroDado;
}

void mostrarDado(resultadoDado carta) {
    switch (carta) {
        case as:    cout << "As";    break;
        case king:  cout << "King";  break;
        case queen: cout << "Queen"; break;
        case jota:  cout << "Jota";  break;
        case diez:  cout << "Diez";  break;
        case nueve: cout << "Nueve"; break;
        default:    cout << "Error"; break;
    }
}
```

Listing 2: dado.cpp

3.2. Módulo Jugador

```cpp
#ifndef JUGADOR_H
#define JUGADOR_H
#include <string>

struct Jugador {
    std::string nombre_usuario;
    int puntos_parciales;
    int tiradas_ganadas;
};

void solicitarJugador(Jugador& jugador);
void actualizarPartidasGanadas(Jugador& jugador);

#endif
```

Listing 3: jugador.h

3.3. Módulo Tirada

```cpp
#include "tirada.h"
#include <iostream>

using namespace std;

int realizarTirada(Tirada& tirada) {
    cout << "Resultado de la tirada:" << endl;
    for (int i = 0; i < 5; i++) {
        tirada[i] = tirarDado();
        cout << "- ";
        mostrarDado(tirada[i]);
        cout << endl;
    }

    int contadorParejas = 0;
    for (int i = 0; i < 4; i++) {
        for (int j = i + 1; j < 5; j++) {
            if (tirada[i] == tirada[j]) {
                contadorParejas++;
            }
        }
```

```
    }

    int puntos = 0;
    if (contadorParejas == 1) puntos = 1;
    else if (contadorParejas == 2) puntos = 2;
    else if (contadorParejas == 3) puntos = 3;
    else if (contadorParejas == 4) puntos = 4;
    else if (contadorParejas == 6) puntos = 6;
    else if (contadorParejas == 10) puntos = 10;

    return puntos;
}
```

Listing 4: tirada.cpp

3.4. Módulo Principal

```
#include <iostream>
#include <ctime>
#include <cstdlib>
#include "jugador.h"
#include "tirada.h"

using namespace std;

int main() {
    srand(time(NULL));

    Jugador listaJugadores[2];
    Tirada tiradaActual;
    char seleccion = 's';

    for(int i = 0; i < 2; i++) {
        cout << "JUGADOR " << i + 1 << ":" << endl;
        solicitarJugador(listaJugadores[i]);
    }

    while (seleccion == 's' && listaJugadores[0].tiradas_ganadas < 5 && listaJugadores
    [1].tiradas_ganadas < 5) {
        for (int i = 0; i < 2; i++) {
            cout << "\nTURNO DE: " << listaJugadores[i].nombre_usuario << endl;
            listaJugadores[i].puntos_parciales = realizarTirada(tiradaActual);
            cout << "Puntos obtenidos: " << listaJugadores[i].puntos_parciales << endl;
        }

        if (listaJugadores[0].puntos_parciales > listaJugadores[1].puntos_parciales) {
            cout << "\nGana la ronda: " << listaJugadores[0].nombre_usuario << endl;
            actualizarPartidasGanadas(listaJugadores[0]);
        } else if (listaJugadores[1].puntos_parciales > listaJugadores[0].
    puntos_parciales) {
            cout << "\nGana la ronda: " << listaJugadores[1].nombre_usuario << endl;
            actualizarPartidasGanadas(listaJugadores[1]);
        } else {
            cout << "\nEmpate en esta ronda." << endl;
        }

        cout << "MARCADOR ACTUAL: " << listaJugadores[0].nombre_usuario << " [" <<
    listaJugadores[0].tiradas_ganadas << "] - ";
        cout << listaJugadores[1].nombre_usuario << " [" << listaJugadores[1].
    tiradas_ganadas << "]" << endl;

        if (listaJugadores[0].tiradas_ganadas < 5 && listaJugadores[1].tiradas_ganadas <
    5) {
            cout << "\n Desea continuar (s/n)?: ";
            cin >> seleccion;
        }
    }

    if (listaJugadores[0].tiradas_ganadas == 5)
```

```
48          cout << "\nEL GANADOR DE LA PARTIDA ES " << listaJugadores[0].nombre_usuario <<
        endl;
49      else if (listaJugadores[1].tiradas_ganadas == 5)
50          cout << "\nEL GANADOR DE LA PARTIDA ES " << listaJugadores[1].nombre_usuario <<
        endl;
51
52      return 0;
53  }
```

Listing 5: main.cpp

4. Resultados y Pruebas

Se han realizado diversas ejecuciones para comprobar que la lógica de puntuación es correcta. El programa identifica adecuadamente desde parejas simples hasta combinaciones complejas como el Full o el Poker.

[Pegar aquí la captura de pantalla de la terminal con una partida ganada]

Figura 1: Interfaz del juego y ejemplo de tirada con puntuación.

5. Conclusión

La realización de esta práctica ha permitido consolidar los conocimientos sobre la programación modular en C++. La separación del código en archivos independientes facilita enormemente la lectura y el mantenimiento del software, permitiendo que cada módulo se encargue de una tarea específica del juego sin interferir en las demás.