README PARCIAL

Nombre del proyecto:

Batalla de Puchamon – C++

Integrantes:

- Luis Miguel López Acero
- Thomas Leandro Bacca Beltrán

¿En qué consiste nuestro proyecto?

Este programa es un juego de batalla por turnos basado en Pokémon. Cada jugador elige un Pokémon, que tiene cuatro ataques diferentes, y lucha contra otro hasta que uno quede sin vida. Se hizo en C++, usando clases y objetos para organizar la información de los Pokémon y sus ataques.

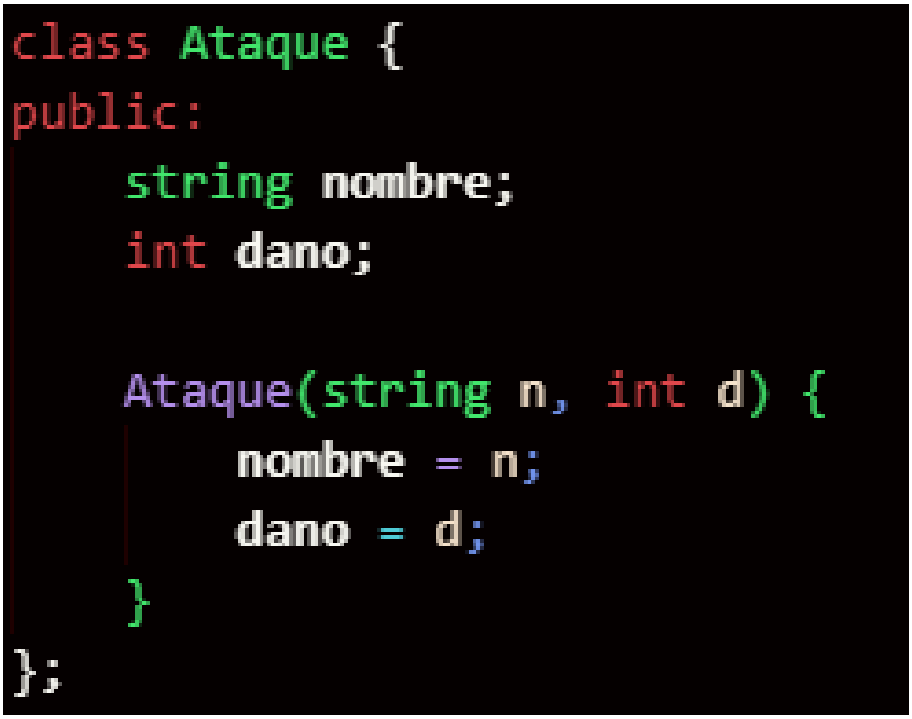
¿Por qué se consideró necesario?

La idea era practicar programación orientada a objetos (POO) y aprender a manejar datos dentro de un videojuego sencillo. También nos ayudó a mejorar la lógica de programación y a estructurar mejor el código.

¿Cómo lo hicimos?

Empezamos dividiendo el problema en partes mas pequeñas para eso lo primero que haremos es incluir 2 librerías, iostream y sitring mas adelante explicaremos porque las usamos.

Después crearemos 2 clases globales la clase ataque y la clase Puchamon.



```
class Ataque {
public:
    string nombre;
    int dano;

    Ataque(string n, int d) {
        nombre = n;
        dano = d;
    }
};
```

```
class Puchamon {
public:
    string nombre;
    int vida;
    Ataque ataque1;
    Ataque ataque2;
    Ataque ataque3;
    Ataque ataque4;

    Puchamon(string n, int v, Ataque a1, Ataque a2, Ataque a3, Ataque a4)
        : nombre(n), vida(v), ataque1(a1), ataque2(a2), ataque3(a3), ataque4(a4) {}

    void atacar(Puchamon &enemigo, Ataque ataque) {
        cout << nombre << " usa " << ataque.nombre << endl;
        enemigo.vida = enemigo.vida - ataque.dano;
        if (enemigo.vida < 0) enemigo.vida = 0;
        cout << enemigo.nombre << " le quedan " << enemigo.vida << " puntos de vida." << endl << endl;
    }

    void mostrarAtaques() {
        cout << "1. " << ataque1.nombre << " (" << ataque1.dano << " daño)" << endl;
        cout << "2. " << ataque2.nombre << " (" << ataque2.dano << " daño)" << endl;
        cout << "3. " << ataque3.nombre << " (" << ataque3.dano << " daño)" << endl;
        cout << "4. " << ataque4.nombre << " (" << ataque4.dano << " daño)" << endl;
    }

    Ataque elegirAtaque(int opcion) {
        if (opcion == 1) return ataque1;
        else if (opcion == 2) return ataque2;
        else if (opcion == 3) return ataque3;
        else return ataque4;
    }
};
```

¿Qué hacen estas clases?

1. **Atributos (nombre, daño)**

El nombre es un sitring que almacena el nombre de los ataques que asignamos a los Pokémon ejemplo impactrueno, Daño lo declaramos como int que guardara el daño que el ataque inflige al oponente

2. **Constructor:**

(Ataque (string n, int d))

Lo usamos para crear un ataque con un nombre y una cantidad de daño que se asigna a los valores **N** y **D** a los atributos de “nombre” y “daño”

Aclaramos para la ayuda de los constructores me guie de este video (me salió gracias a dios y me evito quizás una gran parte del trabajo)

https://youtu.be/IAPL8cHLnuY?si=5sx04iKbHtb_3B2o

¿Por qué lo usamos?

Me facilito la inicialización de objetos, sin el constructor tendría que haber asignado los valores de cada atributo después de crear el objeto, esto también le da un plus al código ya que se inicializa automáticamente y simplifica el código como ya lo dije.

Es importante aclarar que un constructor es un concepto fundamental en **POO**

Seguido de esto la clase de los Pokémon realizamos (Leandro colaboro con el desarrollo de este código):

1. **Atributos:**

   nombre; // Guarda el nombre de Puchamon

   Ataque ataque1… etc. //guarda los cuatro ataques disponibles

   Estos atributos guardan los valores básicos de cada Puchamon, asignándoles un nombre, vida y un conjunto de ataques.

2. **Constructor (inicializa los atributos)**

Pokémon (string n, int v, Ataque a1, Ataque a2, Ataque a3, Ataque a4) : nombre(n), vida(v), ataque1(a1), ataque2(a2), ataque3(a3), ataque4(a4) {}

Como bien explique y argumente en la primera clase nos permite crear un Pokémon **con nombre, vida y ataques predefinidos** en una sola línea Sin el tendríamos que asignar los valores uno por uno, lo que haría el código más largo y propenso a errores.

3. **Método: atacar ()**

(Reduce la vida del enemigo cuando el Pokémon ataca)

Usamos void porque el método no necesita devolver un valor, si no que solo ejecuta la acción de atacar, Tome al enemigo por referencia (Pokémon &enemigo), lo que permite modificar directamente su vida sin hacer copias innecesarias, tome un Ataque como parámetro, que es el ataque elegido para usar en ese turno, mostré el ataque en pantalla **(cout << nombre << " usa " << ataque.nombre << "!";)** , reducimos la vida del enemigo.
Mostramos cuánta vida le queda al enemigo después del ataque

4. **Método: mostrarAtaques ()**

Permite que nuestros jugadores puedan ver qué ataques tiene su Pokémon y decidir cuál usar en la batalla

Método elegirAtaque () selecciona uno de los ataques según los datos de entrada

Permite que el jugador **elija un ataque** según el número ingresado (1-4).

OBSERVACION:

Si el jugador ingresa un numero incorrecto, por defecto se elige el ataque 4

No supimos arreglar este detalle (le pregunte a chat, pero me confundió más entonces aclaramos que debe marca si o si solo de 1-4)

Se vería de esta manera en la terminal

¿Por qué incluimos la librería string?

El uso de string en C++ nos permitió trabajar con texto sin preocuparnos por la gestión manual de memoria Diferente a lo que ocurre con char[] ya que requiere definir un tamaño.

Con la librería de string nos dimos cuenta que podemos unir palabras o frases de manera intuitiva con el operador + diferente de char[] incluyendo que necesitamos usar strcat(), lo que complica la manipulación de cadenas.

Tambien las cadenas de string pueden imprimirse directamente con “cout”, mientras que char[] nos pide recorrer todo el array o usar funciones específicas como printf()

cout << "Ataque: " << nombre << endl; // **Con string (simple)**
printf("Ataque: %s\n", nombre); // **Con char[] (menos intuitivo)**

ELEGIR POKEMON

```
Puchamon elegirPuchamon(int jugador) {
    cout << "Jugador " << jugador << ", elige tu Puchamon:" << endl;
    cout << "1. Pikachu\n2. Charmander\n3. Bulbasaur\n4. Squirtle" << endl;
    int eleccion;
    cin >> eleccion;

    if (eleccion == 1) {
        return Puchamon("Pikachu", 100,
            Ataque("Impactrueno", 20),
            Ataque("Placaje", 10),
            Ataque("Ataque Rapido", 15),
            Ataque("Rayo", 30));
    } else if (eleccion == 2) {
        return Puchamon("Charmander", 100,
            Ataque("Lanzallamas", 25),
            Ataque("Aranazo", 15),
            Ataque("Bola de Fuego", 15),
            Ataque("Placaje", 10));
    } else if (eleccion == 3) {
        return Puchamon("Bulbasaur", 100,
            Ataque("Latigazo", 20),
            Ataque("Hoja Afilada", 25),
            Ataque("Placaje", 10),
            Ataque("Drenadoras", 15));
    } else {
        return Puchamon("Squirtle", 100,
            Ataque("Pistola Agua", 20),
            Ataque("Burbuja", 15),
            Ataque("Placaje", 10),
            Ataque("Hidro Pulso", 25));
    }
}
```

Aquí definimos la función y el objeto que este recibe, luego en el terminal aparece un menú para que el jugador escoga el Puchamon que usara por ejemplo si el jugador elige 1:

```
if (eleccion == 1) {
    return Puchamon("Pikachu", 100,
        Ataque ("Impactrueno", 20),
        Ataque ("Placaje", 10),
        Ataque ("Ataque Rapido", 15),
        Ataque ("Rayo", 30));
```

Se crea un objeto Puchamon con nombre "Pikachu", 100 puntos de vida, y cuatro ataques distintos, cada uno con un nombre y un valor de daño.

Los otros bloques else if funcionan igual para "Charmander", "Bulbasaur", y "Squirtle", con ataques distintos en cada caso.

Finalmente, si el jugador no elige un número del 1 al 3, el else hace que por defecto se le asigne "Squirtle".

```
int main() {
    cout << "Batalla Puchamon" << endl;

    Puchamon jugador1 = elegirPuchamon(1);
    Puchamon jugador2 = elegirPuchamon(2);

    while (jugador1.vida > 0 && jugador2.vida > 0) {
        int opcion;

        cout << jugador1.nombre << ", escoge ataque:" << endl;
        jugador1.mostrarAtaques();
        cin >> opcion;
        jugador1.atacar(jugador2, jugador1.elegirAtaque(opcion));

        if (jugador2.vida == 0) break;

        cout << jugador2.nombre << ", escoge ataque:" << endl;
        jugador2.mostrarAtaques();
        cin >> opcion;
        jugador2.atacar(jugador1, jugador2.elegirAtaque(opcion));
    }

    if (jugador1.vida == 0 && jugador2.vida == 0) {
        cout << "Empate" << endl;
    } else if (jugador1.vida == 0) {
        cout << jugador2.nombre << " gana" << endl;
    } else {
        cout << jugador1.nombre << " gana" << endl;
    }

    return 0;
}
```

Para la función main, primero se imprime el titulo del juego, luego a cada jugador se le da la opción. A continua se creará un bucle while hasta que alguno de los 2 Puchamon tengan su vida en 0. A continuación se muestra lo que hace cada jugador por turnos.

1. Se muestran los ataques del jugador
2. El jugador elige un ataque (cin >> opcion).
3. Se ejecuta ese ataque sobre jugador.

Luego se verifica si el jugador contrario perdió, de ser así la terminal muestra un Ganaste.

Para ayuda de nuestro proyecto de Pokémon en C++ nos guiamos de los siguientes videos como ayuda visual y corrección de errores NO COPIAMOS NADA DE LOS VIDEOS.

- Curso de C con Pokémon:
  https://www.youtube.com/watch?v=d_BF2z_DqFE
  https://www.youtube.com/watch?v=4tixT2DGG6U
- Sistema de turnos Playlist:
  https://www.youtube.com/watch?v=VPh5qoW8dx0&list=PLl9ZbfOa8iGm0KDK6gWReFE50Mc4PWtmA&index=6
  https://www.youtube.com/watch?v=_cvcYQke4HA&list=PLl9ZbfOa8iGm0KDK6gWReFE50Mc4PWtmA&index=5
  https://www.youtube.com/watch?v=mrFuRrHQHUU&list=PLl9ZbfOa8iGm0KDK6gWReFE50Mc4PWtmA&index=4
  https://www.youtube.com/watch?v=5E5UEK_YziM&list=PLl9ZbfOa8iGm0KDK6gWReFE50Mc4PWtmA&index=3
  https://www.youtube.com/watch?v=wUPH2Clpnd4&list=PLl9ZbfOa8iGm0KDK6gWReFE50Mc4PWtmA&index=2
  https://www.youtube.com/watch?v=L8qcE-7xVxk&list=PLl9ZbfOa8iGm0KDK6gWReFE50Mc4PWtmA&index=1
- Pokémon Black Consolé
  https://www.youtube.com/watch?v=Fov9lJVKmPc&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=9
  https://www.youtube.com/watch?v=99oHHMltok4&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=8
  https://www.youtube.com/watch?v=Spv5zGZTwgM&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=7
  https://www.youtube.com/watch?v=dVnKirQg5TE&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=6
  https://www.youtube.com/watch?v=dTAJiywpw_k&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=5

https://www.youtube.com/watch?v=L31XYxnugKE&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=4
https://www.youtube.com/watch?v=UX56AfTwsyU&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=3
https://www.youtube.com/watch?v=qnCBDW9dQbk&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=2
https://www.youtube.com/watch?v=Y2-VY7gdIXc&list=PLl9ZbfOa8iGnsr9lpYJ17_qxIZHdCwFk3&index=1