Tel-102 Paralelo 2
2º Semestre 2022
Control 3
27 de Octubre de 2022

Instrucciones

- Lea con detenimiento y desarrolle **individualmente** cada una de las actividades a realizar durante la experiencia.
- Modifique el archivo con extensión **.cpp** proporcionado con lo que ud. ha desarrollado. El nombre del archivo debe tener el siguiente formato: **TEL102_C3_Nombre_Apellido.cpp** (Ej. TEL102_C3_Patricio_Olivares.cpp), sin incluir tíldes.
- Enviar el archivo a través de la página de aula del ramo, sección "Control 3" hasta las 18:30:00 del día de **hoy**, Jueves 27/10/2020, Hora continental de Chile (UTC-3).
- No debe entregar el archivo.h entregado, ya que no puede ser modificado.
- Cada minuto de atraso tendrá un descuento siguiendo la serie de Fibonacci.
- Trate de utilizar herramientas conocidas o aprendidas en clases. **No copie literalmente de recursos online**.
- Comente adecuadamente el programa, describiendo lo que hace.
- Sea riguroso con las instrucciones de desarrollo.
- ¡Éxito!

Kombate Mortal XI

La famosa empresa *TeleRealm Studios* está desarrollando la nueva versión del aclamado juego **Kombate Mortal**. En búsqueda de nuevos desarrolladores, se han abierto nuevos puestos de trabajo para gente que quiera entrar en esta prominente industria. Como gran alumno de **TEL102**, ud. cree que es una gran oportunidad para probar sus capacidades. En la prueba del puesto de trabajo, *TeleRealm Studios* quiere ver si es capaz de poder desarrollar un prototipo básico de lo que sería el nuevo juego **Kombate Mortal XI**. El juego es un clásico de las peleas de arcade, donde dos jugadores seleccionan a un personaje y pelean entre ellos; ejecutando ataques, combinaciones y poderes; de tal forma de vencer al oponente.
La persona a cargo de la selección, el mismismo *Eduardo Boon*, le entrega un header base llamado `kombate_mortal.h` el cual ud. **no puede modificar**.

kombate_mortal.h

```
#include<iostream>
#include<cstring>

using namespace std;

class kombatant{
        public:
                //constructor
                kombatant();

                //metodos de entrada
                void setHP(float hp);
                void setPower(float power);
                void setDefense(float defense);

                //metodos de salida
                float getHP();
                int getBlock();
                int getGauge();
                float attack();
                float special();
                void fatality();

        private:
                float hp;
                float power;
                float defense;
                int gauge;

};
```

En este archivo se define la clase `kombatant`, la cual modela a un personaje elegible para una pelea. Cada personaje posee los siguientes atributos:

- `hp`: Atributo que caracteriza la vida o *stamina* del personaje. Ésta se ve reducida cada vez que recibe un ataque del rival. Siempre tiene el mismo valor inicial para cada personaje del juego: cien (100).
- `power`: Atributo que caracteriza la fuerza del personaje. Éste siempre oscilará entre los valores veinte (20) y treinta (30). Se utiliza para calcular varios elementos del personaje, como el daño que inflingen los ataques o la capacidad de bloquear un golpe rival.
- `defense`: Atributo que caracteriza la defensa del personaje. Éste siempre oscilará entre los valores uno (1) y diez (10). Al igual que *power*, se utiliza para calcular varios elementos del personaje.
- `gauge`: Atributo que caracteriza la capacidad del personaje. Este parte siempre en valor cero (0), y va aumentando acorde a que los ataques del personaje sean recibidos por el rival.

Para poder efectuar la simulación de una pelea, esta clase posee un conjunto de métodos para poder trabajar con los atributos antes explicados. Estos métodos son:

- `setHP(float hp)`: Método que permite modificar el atributo `hp` del personaje a partir de su parámetro.
- `setPower(float power)`: Método que permite modificar el atributo `power` del personaje a partir de su paramétro.
- `setDefense(float defense)`: Método que permite modificar el atributo `defense` del personaje a partir de su parámetro.

- `getHP()`: Método que permite obtener el valor actual del atributo `hp` del personaje.
- `getBlock()`: Método que permite obtener el ratio de bloqueo del personaje. Este se calcula como el valor truncado de la división entre el atributo `power` y el atributo `defense`: $\left\lfloor \frac{\texttt{power}}{\texttt{defense}} \right\rfloor$.
- `getGauge()`: Método que permite obtener el valor del atributo `gauge` del personaje.
- `attack()`: Método que realiza un ataque normal. Este ataque se calcula como $\texttt{power} \times \frac{\texttt{defense}}{100}$. Este será el ataque utilizado por el personaje s el atributo `gauge` está entre los valores 50 y 100.
- `special()`: Método que realiza un ataque especial. Este ataque se calcula como $\texttt{power} \times \frac{\texttt{defense}}{\texttt{hp}}$. Este será el ataque a usar por el personaje si el atributo `gauge` está entre los valores 0 y 49.
- `fatality()`: Método que realiza un movimiento final de pelea, llamado *Fatality*, el cual no modifica ningún atributo: tiene un efecto netamente decorativo. Este movimiento se efectuará solo cuando el personaje haya derrotado a su rival, y tenga cien (100) como valor de su atributo `gauge`.

Considere que en una pelea, dos personajes de la clase `kombatant` se enfrentarán y deben cumplir lo siguiente:

- Los personajes comenzarán y continuarán con la pelea, hasta que alguno de ellos quede con el atributo `hp` con valor 0 o inferior. Debe considerar el caso de que ambos queden con valor negativo.
- Los ataques se realizan simultaneamente. Ninguno tiene prioridad sobre el otro.
- Los personajes pueden automáticamente bloquear ataques según su ratio de bloqueo. Por ejemplo, si el ratio de bloqueo da valor tres (3), cada tres ataques, el tercero no será exitoso y se considera bloqueado. Vea el ejemplo de ejecución para mayor claridad.
- Un ataque de tipo normal, solamente si es recibido por rival, aumenta el atributo `gauge` en 5 unidades.
- Un ataque de tipo especial, solamente si es recibido por el rival, aumenta el atributo `gauge` en 10 unidades.

Además, Eduardo Boon le entrega un archivo llamado `kombate_mortal.cpp`, que es el archivo base donde ud. debe programar. Por lo tanto, **no pueden implementar ningun método o función en en el archivo header**.

kombate_mortal.cpp

```
//incluya el archivo que contienen la clase y programe aquí constructures y métodos.

void kombate_mortal(kombatant &p1, kombatant &p2){

        //programe aquí la función kombate_mortal

}

int main(){
        kombatant kung_leo;
        kombatant luis_kang;

        kung_leo.setPower(25);
        kung_leo.setDefense(7);

        luis_kang.setPower(23);
        luis_kang.setDefense(9);

        cout << "Final Round! FIGHT!" << endl;
        kombate_mortal(kung_leo, luis_kang);

        return 0;
}
```

En su entrevista de conocimientos se le solicita lo siguiente:

1. (25pts) Implemente el constructor y los métodos de entrada (set) de los atributos de la clase `kombatant`.

2. (35pts) Implemente los metodos de salida (get y ataques) de la clase `kombatant`. Tome en consideracion todos los detalles explicados anteriormente.

3. (40pts) Implemente la función `void kombate_mortal(kombatant &p1, kombatant &p2)`. Esta función simula la pelea entre dos `kombatant p1` y `p2`, imprimiendo en cada turno de la pelea, el tipo de ataque realizado por los personajes, si el ataque fue bloqueado, y cuanto es el valor de los atributos `hp` de estos. Considere bien todos los detalles explicitados en el enunciado. A continuación se presenta un ejemplo de ejecución del programa, para que lo considere de guía.

Salida (consola)

```
[elprofe@tel102 control3]$ ./MK11
Final Round! FIGHT!
Kung Leo Ataca Especial
Luis Kang Ataca Especial
Kung Leo HP: 97.8931. Luis Kang HP: 98.25

Kung Leo Ataca Especial
Luis Kang Ataca Especial
Kung Leo HP: 95.7472. Luis Kang HP: 96.4623

Luis Kang bloquea!
Luis Kang Ataca Especial
Kung Leo HP: 93.6013. Luis Kang HP: 96.4623

Kung Leo Ataca Especial
Kung Lao bloquea!
Kung Leo HP: 93.6013. Luis Kang HP: 94.5927
...
...

...
...
Luis Kang bloquea!
Luis Kang Ataca Normal
Kung Leo HP: 0.126411. Luis Kang HP: 26.0098

Kung Leo Ataca Normal
Kung Lao bloquea!
Kung Leo HP: 0.126411. Luis Kang HP: 24.2598

Kung Leo Ataca Normal
Luis Kang Ataca Normal
Kung Leo HP: -1.94359. Luis Kang HP: 22.5098

FINISH HIM!!!!
Uppercut, lo eleva y el rival cae al fondo del abismo con clavos gigantes
FATALITY!!!!
Luis Kang Wins
```

Nota: Los símbolos ... indican que existen más mensajes en la salida del programa, vale decir, solo se muestra una fracción inicial y final de la ejecución.