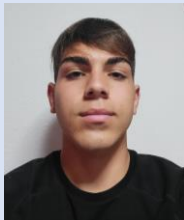
	M5 – OPTIMIZACIÓ DE PROGRAMARI		
	UF2 – OPTIMIZACION DE PROGRAMARI		
Pr4-Refactorizacion-de-codigo			
Apellidos: Ortiz Roque	Nombre: Marc	Fecha:	13/01/2023

## Contenido

Optimización de código.....	2
MAIN .....	2
ENEMIESATTACK() .....	2
CHECKHEROSTATUS() .....	3
Parámetros por referencia .....	4

## Optimización de código

### MAIN()

La primera tarea para hacer era refactorizar el MAIN, añadiéndole la siguiente función y mejorando otras

- checkHeroStatus

Además de añadir comentarios para hacer mucho mas entendedor el texto

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

    while ((enemyIsAlive || enemy2IsAlive) && heroIsAlive) {
        chooseEnemy();
        chooseAtaque();
        heroIsAlive = enemiesAttack();
    }
}
```

Marc Ortiz

Ilustración 1 Main antes

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

    while ((enemy1IsAlive || enemy2IsAlive) && heroIsAlive) {
        //ELEGIMOS AL ENEMIGO QUE VAMOS A ATACAR
        chooseEnemy();
        //ESCOGEMOS EL ATAQUE QUE QUEREMOS REALIZAR
        chooseAtaque();
        //ATACAN LOS ENEMIGOS Y COMPROBAMOS SI ESTAN VIVOS
        enemiesAttack();
        //COMPROVAMOS SI EL HEROE ESTA VIVO
        heroIsAlive = checkherostatus();
    }
}
```

Marc Ortiz

Ilustración 2 Main después

### ENEMIESATTACK()

La función enemiesAttack() era muy difícil de leer y poco optimizada, pasamos a crear dos nuevas funciones enemiesAttack()- i checkHeroStatus()

```
bool enemiesAttack() {
    enemyDamage = rand() % 999 + 30;
    enemy2Damage = rand() % 999 + 30;

    if (enemyIsAlive && enemy2IsAlive) {
        dano = enemyDamage + enemy2Damage;
        heroHP = heroHP - dano;
        if (heroHP <= 0) {
            heroHP = 0;
        }
        cout << "\nLos enemigos te han hecho " << dano << " te quedan " << heroHP << " puntos de vida\n\n";
    }
    else if (enemyIsAlive && enemy2IsAlive == false) {
        heroHP = heroHP - enemyDamage;
        if (heroHP <= 0) {
            heroHP = 0;
        }
        cout << "\nEl " << enemyname << " te ha hecho " << enemyDamage << " te quedan " << heroHP << " puntos de vida\n\n";
    }
    else if (enemyIsAlive == false && enemy2IsAlive) {
        heroHP = heroHP - enemy2Damage;
        if (heroHP <= 0) {
            heroHP = 0;
        }
        cout << "\nEl " << enemyname2 << " te ha hecho " << enemy2Damage << " te quedan " << heroHP << " puntos de vida\n\n";
    }
    else if (enemyIsAlive == false && enemy2IsAlive == false) {
        cout << "Has Ganado el combate\n";
    }

    if (heroHP <= 0) {
        return false;
        cout << "El heroe ha sido derrotado\n ";
    }
    else
    {
        return true;
    }
}
```

Marc Ortiz

Ilustración 1 enemiesAttack() antes

Después de la refactorización nos quedaría tal que así:

La función `enemiesAttack()` pasa a ser `void`

```
void enemiesAttack() {
    enemyDamage = rand() % 999 + 30;
    enemy2Damage = rand() % 999 + 30;

    if (enemy1IsAlive && enemy2IsAlive) {
        dano = enemyDamage + enemy2Damage;
        heroHP = heroHP - dano;
        if (heroHP <= 0) {
            heroHP = 0;
        }
        cout << "\nLos enemigos te han hecho " << dano << " te quedan " << heroHP << " puntos de vida\n\n";
    }
    else if (enemy1IsAlive && enemy2IsAlive == false) {
        heroHP = heroHP - enemyDamage;
        if (heroHP <= 0) {
            heroHP = 0;
        }
        cout << "\nEl " << enemyname << " te ha hecho " << enemyDamage << " te quedan " << heroHP << " puntos de vida\n\n";
    }
    else if (enemy1IsAlive == false && enemy2IsAlive) {
        heroHP = heroHP - enemy2Damage;
        if (heroHP <= 0) {
            heroHP = 0;
        }
        cout << "\nEl " << enemyname2 << " te ha hecho " << enemy2Damage << " te quedan " << heroHP << " puntos de vida\n\n";
    }
    else if (enemy1IsAlive == false && enemy2IsAlive == false) {
        cout << "Has Ganado el combate\n";
    }
}
```

Marc Ortiz

*Ilustración 2 enemiesAttack() después*

## CHECKHEROSTATUS()

Y creamos la segunda función donde solamente comprobamos si el enemigo ha sido derrotado o no por ello es una función booleana, que nos devolverá un `true` o un `false` si se cumple la condición

```
bool checkHeroStatus() {
    if (heroHP <= 0) {
        cout << "El heroe ha sido derrotado\n ";
        return false;
    }
    else
    {
        return true;
    }
}
```

Marc Ortiz

*Ilustración 3 checkHeroStatus() creada*

## Parámetros por referencia

También añadimos dos funciones nuevas por referencia para poder optimizar el código estas serán

- attackChose() → Donde aplicara el ataque del héroe
- checkStatus() → Comprobara si el enemigo esta derrotado

```
void attackChose(int& enemyHP) {
    if (ataque == 1) {
        enemyHP = enemyHP - espada;
    }
    else if (ataque == 2) {
        enemyHP = enemyHP - golpe;
    }
    else if (ataque == 3) {
        if (limitado != 0) {
            enemyHP = enemyHP - magia;
            limitado -= 1;
        }
        else {
            cout << "no has podido atacar con magia no te quedan usos\n";
        }
    }
    else {
        "Has fallado, no existe ese ataque";
    }
}
```

Marc Ortiz

```
bool checkStatus(bool& enemyIsAlive, int& enemyHP, string& enemynamex) {
    if (enemyHP <= 0) {
        enemyHP = 0;
        enemyIsAlive = false;
        cout << "Has derrotado al " << enemynamex << "\n";
    }
    else {
        return true;
    }
}
```

Marc Ortiz

Ilustración 4 checkStatus() creada

Ilustración 5 attackChose() creada

Además, con estas dos funciones conseguimos optimizar mas la función chooseAtaque()

```
void chooseAtaque() {
    cout << "Que ataque quieres realizar:\n" << "Espada[1]\n" << "Golpe[2]\n" << "Magia[3] Quedan " << limitado << " ataques restantes\n";
    cin >> ataque;

    if (enemyIsAlive) {
        if (escoger == 1) {
            if (ataque == 1) {
                enemyHP = enemyHP - espada;
            }
            else if (ataque == 2) {
                enemyHP = enemyHP - golpe;
            }
            else if (ataque == 3) {
                if (limitado != 0) {
                    enemyHP = enemyHP - magia;
                    limitado -= 1;
                }
                else {
                    cout << "no has podido atacar con magia no te quedan usos\n";
                }
            }
        }
        else {
            "Has fallado, no existe ese ataque";
        }
        if (enemyHP <= 0) {
            enemyHP = 0;
            enemyIsAlive = false;
            cout << "Has derrotado al enemigo 1\n";
        }
        cout << "\nAl " << enemynamex << " le quedan " << enemyHP << " puntos de vida\n";
        cout << "\nAl " << enemynamex << " le quedan " << enemyHP << " puntos de vida\n";
    }
    else {
        cout << "\nEl " << enemynamex << " esta muerto no puedes seguir atacandole, has perdido el turno\n";
    }

    if (enemy2IsAlive) {
        if (escoger == 2) {
            if (ataque == 1) {
                enemy2HP = enemy2HP - espada;
            }
            else if (ataque == 2) {
                enemy2HP = enemy2HP - golpe;
            }
            else if (ataque == 3) {
                if (limitado != 0) {
                    enemy2HP = enemy2HP - magia;
                    limitado -= 1;
                }
                else {
                    cout << "no has podido atacar con magia, no te quedan usos\n";
                }
            }
        }
        else {
            "Has fallado, no existe ese ataque";
        }
        if (enemy2HP <= 0) {
            enemy2HP = 0;
            enemy2IsAlive = false;
            cout << "Has derrotado al enemigo 2\n";
        }
        cout << "\nAl " << enemynamex << " le quedan " << enemyHP << " puntos de vida\n";
        cout << "\nAl " << enemynamex2 << " le quedan " << enemy2HP << " puntos de vida\n";
    }
    else {
        cout << "\nEl " << enemynamex2 << " esta muerto no puedes seguir atacandole, has perdido el turno\n";
    }
}
```

Marc Ortiz

## Resultado final después de optimizar la función

```
void chooseAtaque() {
    //ESCOGEMOS EL ATAQUE
    cout << "Que ataque quieres realizar:\n" << "Espada[1]\n" << "Golpe[2]\n" << "Magia[3] Quedan " << limitado << " ataques restantes\n";
    cin >> ataque;

    //ATACAMOS AL ENEMIGO 1 Y COMPROBAMOS SI ESTA VIVO O MUERTO
    if (enemy1IsAlive) {
        if (escoger == 1) {
            attackChose(enemy1HP);
            enemy1IsAlive = checkStatus(enemy1IsAlive, enemy1HP, enemyname);
        }
    }
    else {
        cout << "\nEl " << enemyname << " esta muerto no puedes seguir atacandole, has perdido el turno\n";
    }

    //ATACAMOS AL ENEMIGO 2 Y COMPROBAMOS SI ESTA VIVO O MUERTO
    if (enemy2IsAlive) {
        if (escoger == 2) {
            attackChose(enemy2HP);
            enemy2IsAlive = checkStatus(enemy2IsAlive, enemy2HP, enemyname2);
        }
    }
    else {
        cout << "\nEl " << enemyname2 << " esta muerto no puedes seguir atacandole, has perdido el turno\n";
    }

    cout << "\nAl " << enemyname << " le quedan " << enemy1HP << " puntos de vida\n";
    cout << "\nAl " << enemyname2 << " le quedan " << enemy2HP << " puntos de vida\n";
}
```

Marc Ortiz