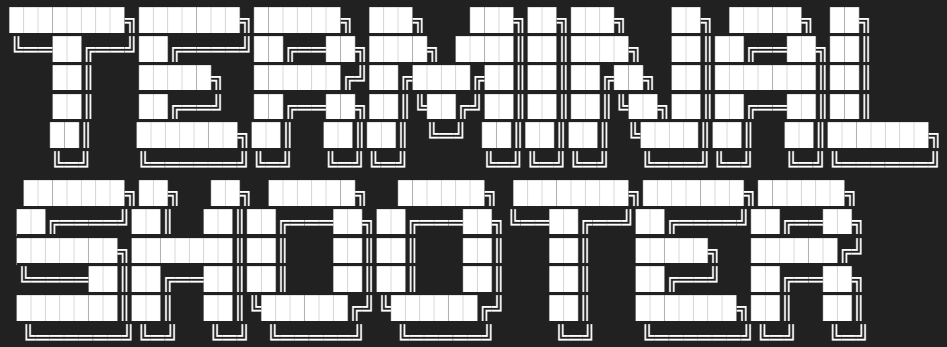
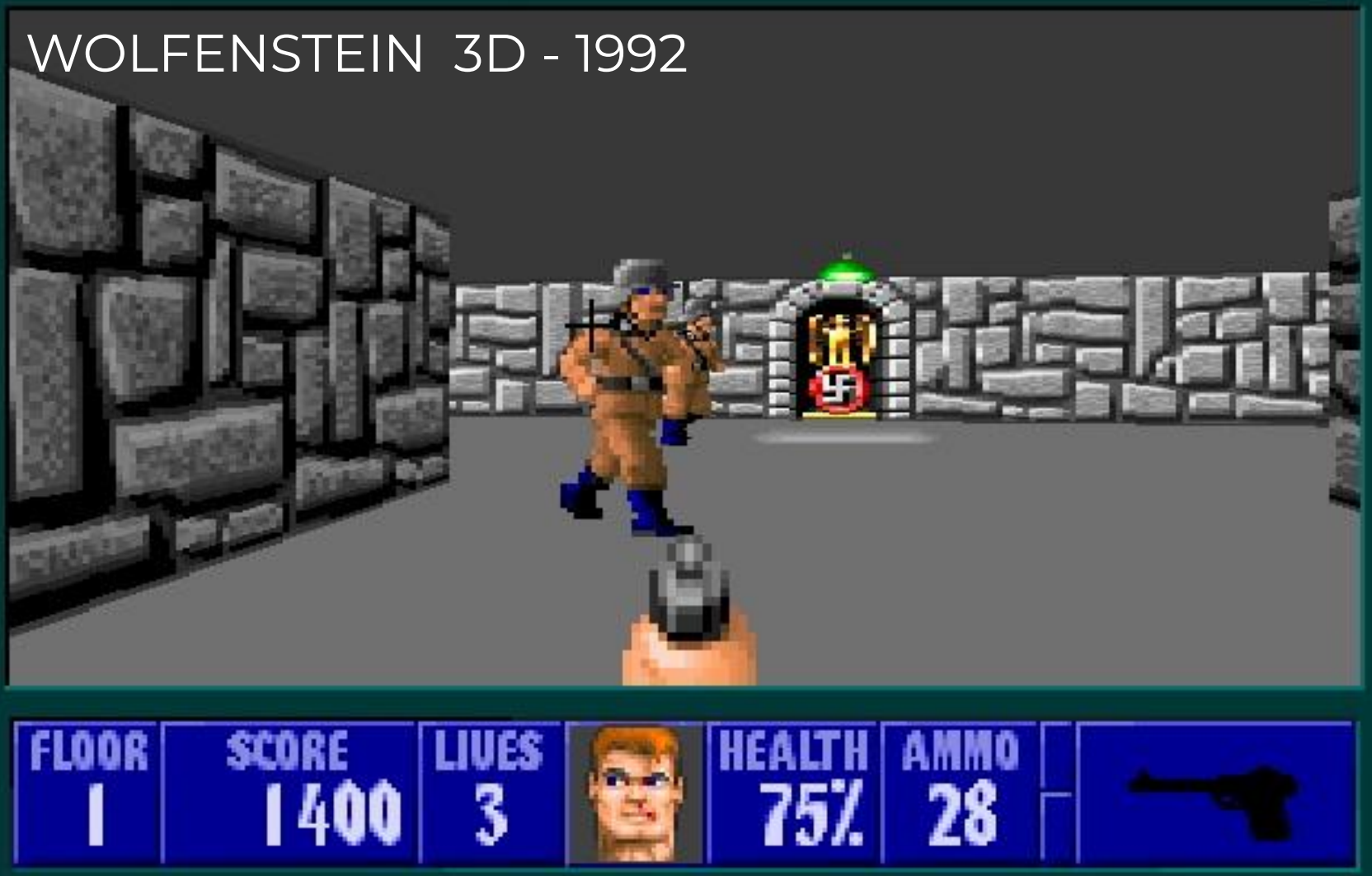


Um jogo FPS desenvolvido em C/C++ apenas com bibliotecas standard.

Augusto L. Matos
Daniel Lombardi
Pedro F. Baleeiro

Inspirações

WOLFENSTEIN 3D - 1992



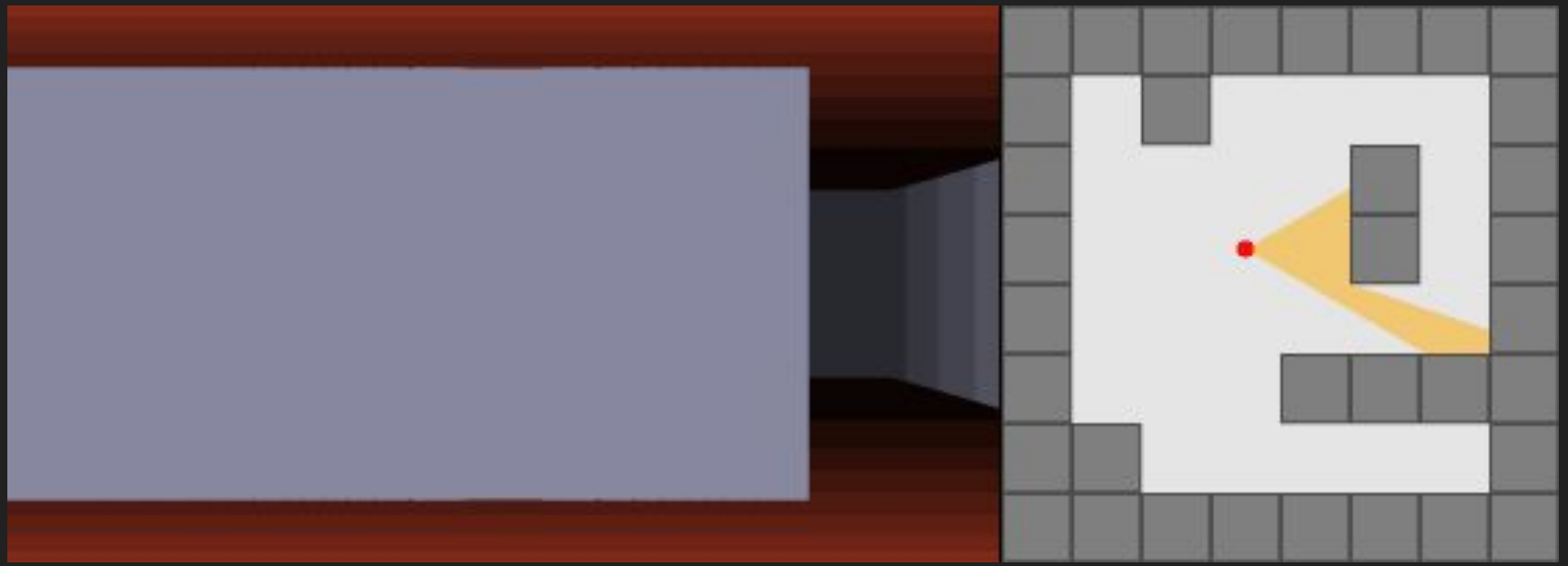

DOOM - 1993
0
100
100

DOOM - 1993


8
AMMO
58%
HEALTH
2 3 4
5 6 7
ARMS
79%
ARMOR
BULL 110 / 400
SHEL 8 / 100
RCKT 1 / 100
CELL 260 / 600


QUAKE - 1996

Renderização por RayCasting



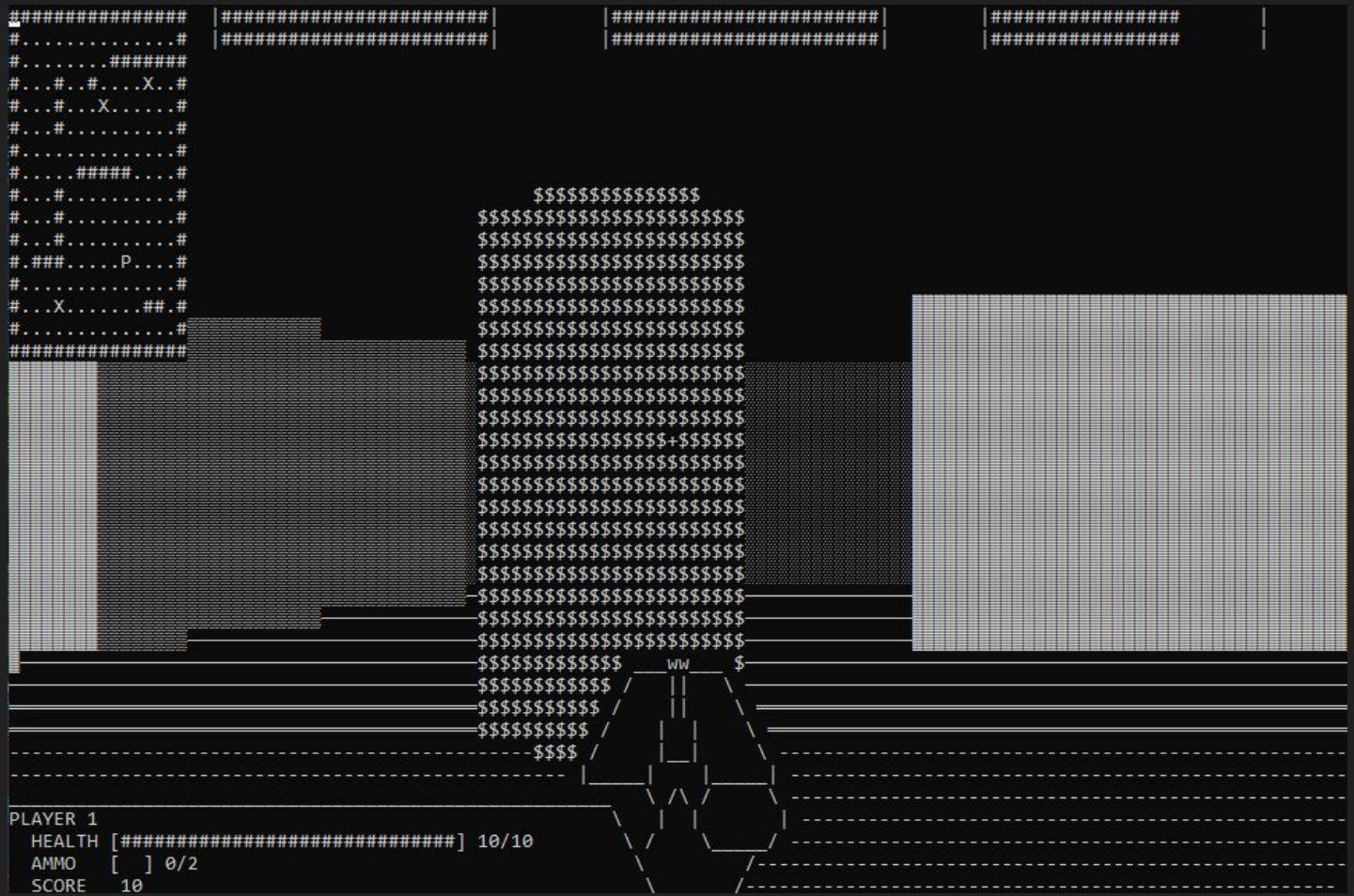
```
while(!bHitWall && !bHitMob &&fDistanceToWall < this->fDepth) {
            fDistanceToWall += F_RAY_STEP_SIZE;

            int nTestX = (int)(player.pos.x + fEyeX * fDistanceToWall);
            int nTestY = (int)(player.pos.y + fEyeY * fDistanceToWall);

            // Ray out of bounds
            if (nTestX < 0 || nTestX >= this->map.nMapWidth || nTestY < 0 ||
                nTestY >= this->map.nMapHeight) {
                fDistanceToWall = this->fDepth;
            } else {
                // Ray inbounds
                if (this->map.smap[nTestY * this->map.nMapWidth + nTestX] == '#')
                    bHitWall = true;
                else if (this->map.smobs[nTestY * this->map.nMapWidth + nTestX] == 'X')
                    bHitMob = true;
            }
        }
```

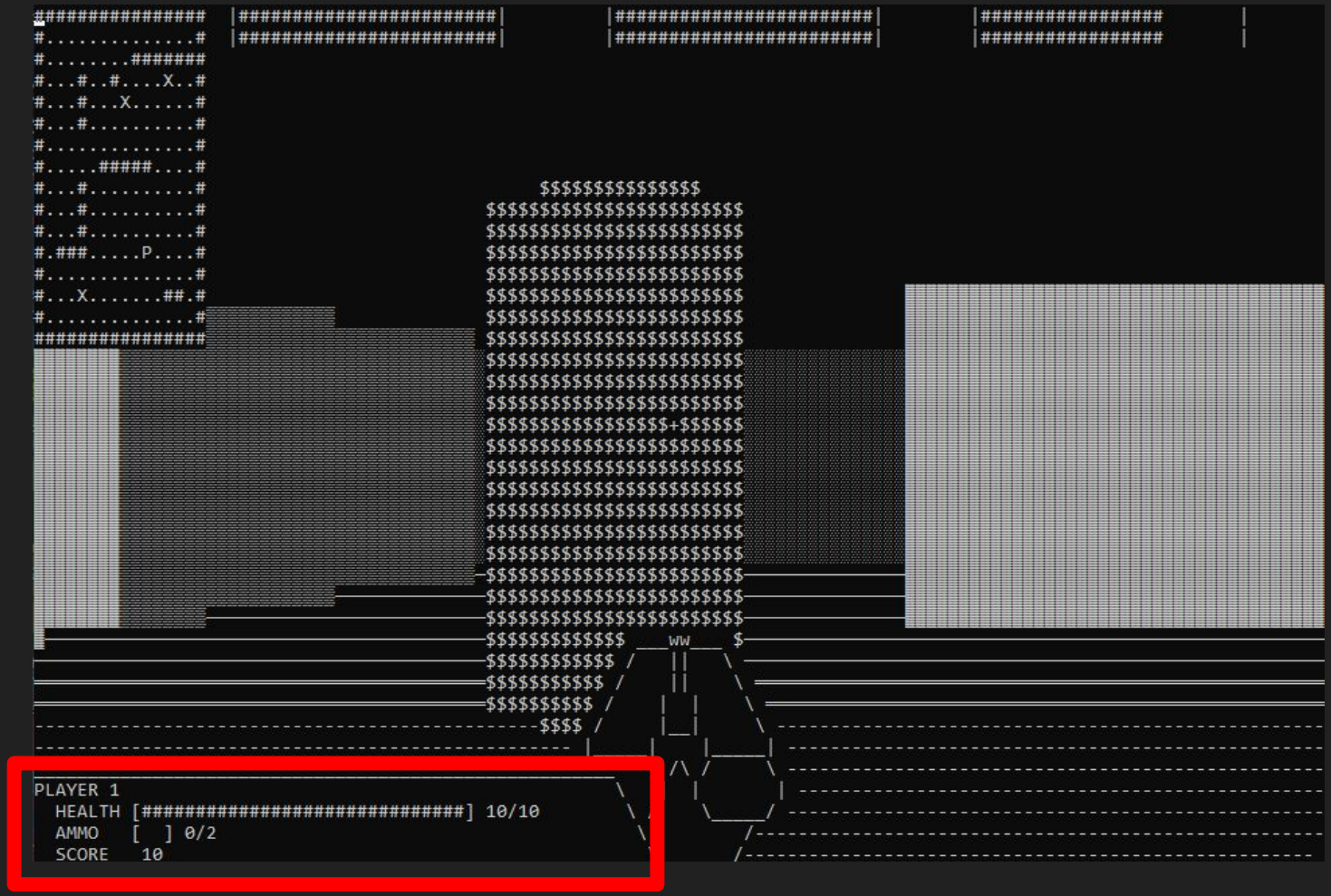
Estrutura de Dados e TAD

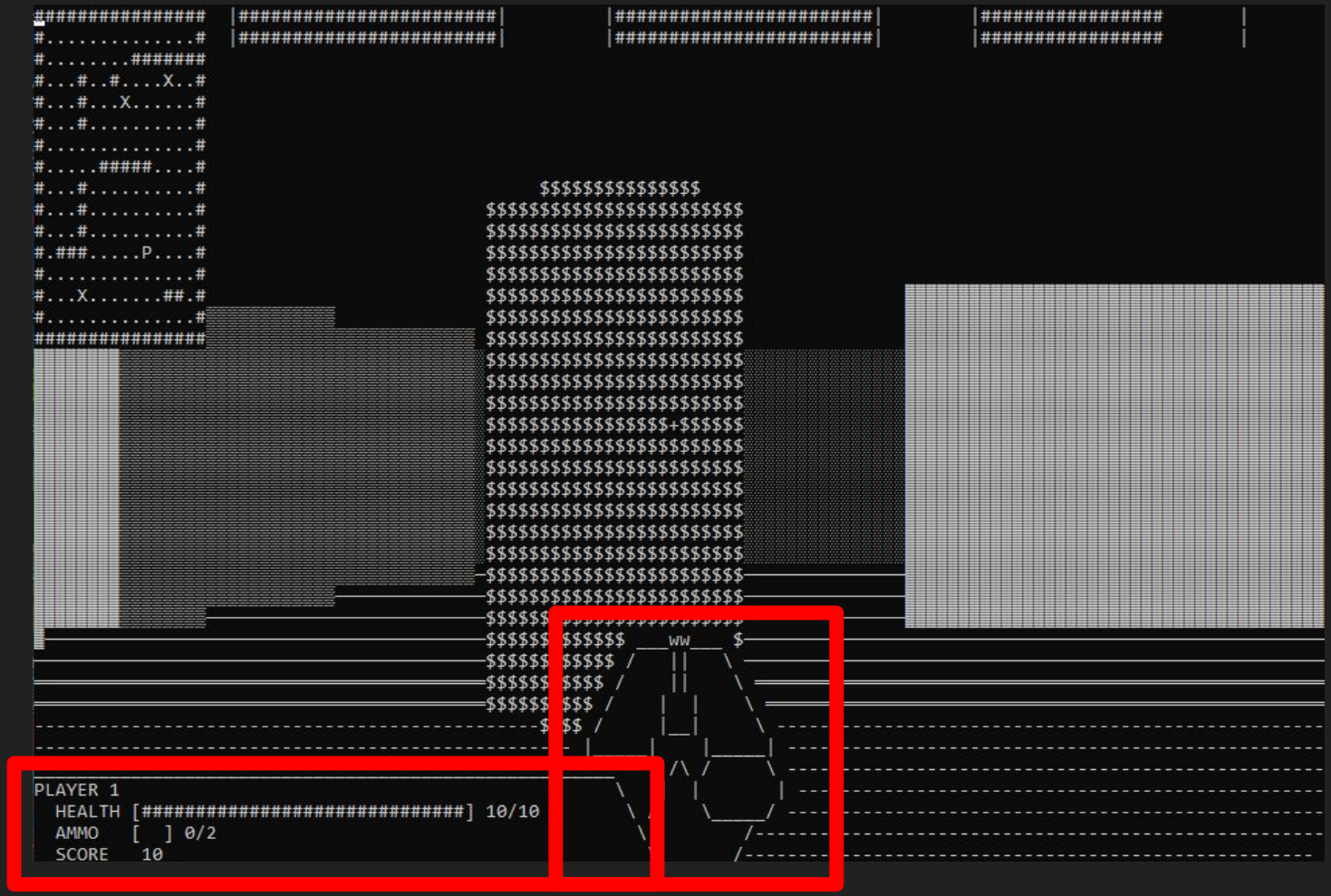
Estruturas de Dados (Lista Cadastral) no Renderizador

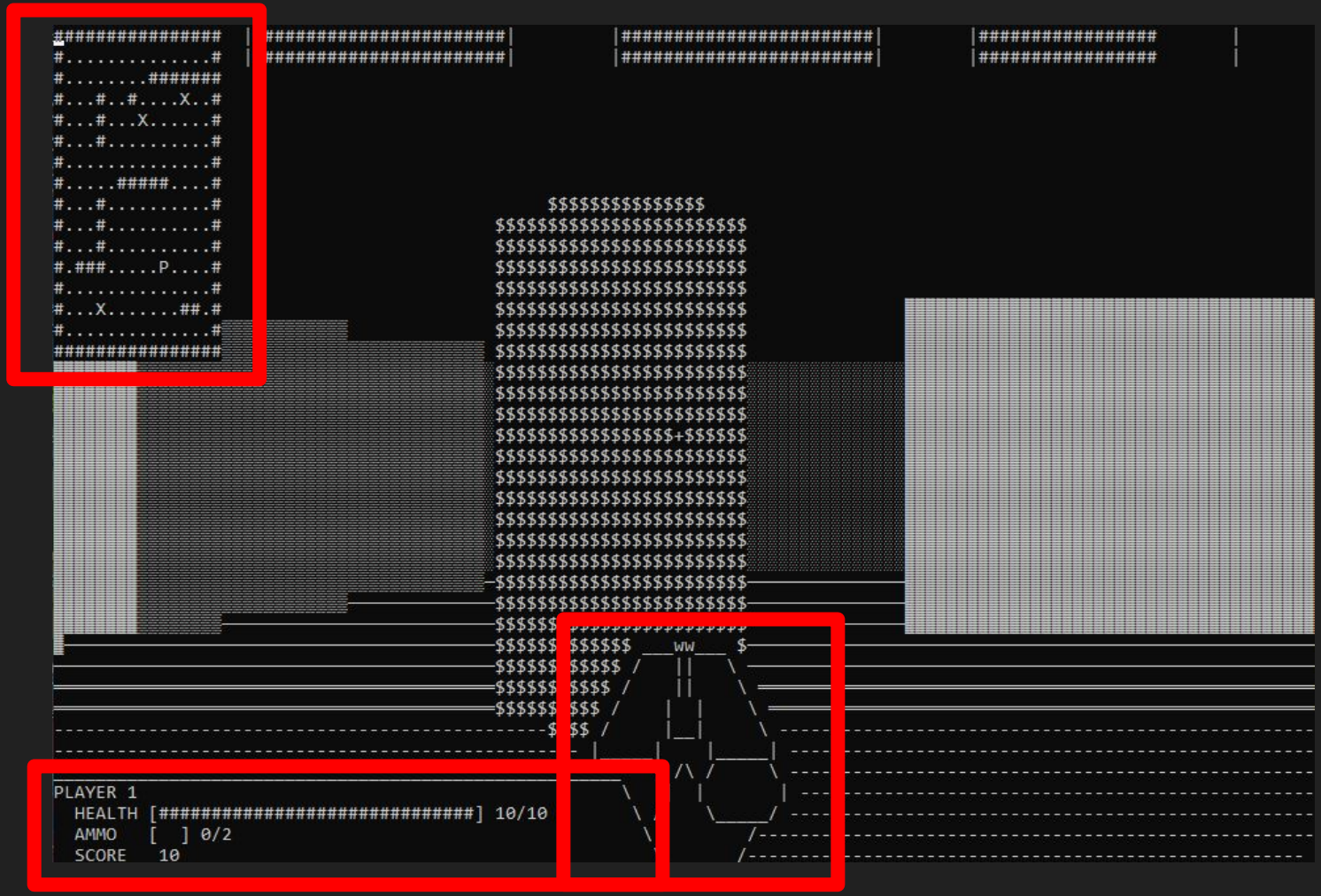
Mapeamento de valores encontrados no mapa e outras estruturas, ex:

- HUD
- Sprite da arma
- Mini-mapa
- Tela de score final
- Posição dos inimigos no mapa


PLAYER 1
HEALTH [##############################] 10/10
AMMO [ ] 0/2
SCORE 10


PLAYER 1
HEALTH [##############################] 10/10
AMMO [ ] 0/2
SCORE 10


PLAYER 1
HEALTH [##############################] 10/10
AMMO [ ] 0/2
SCORE 10

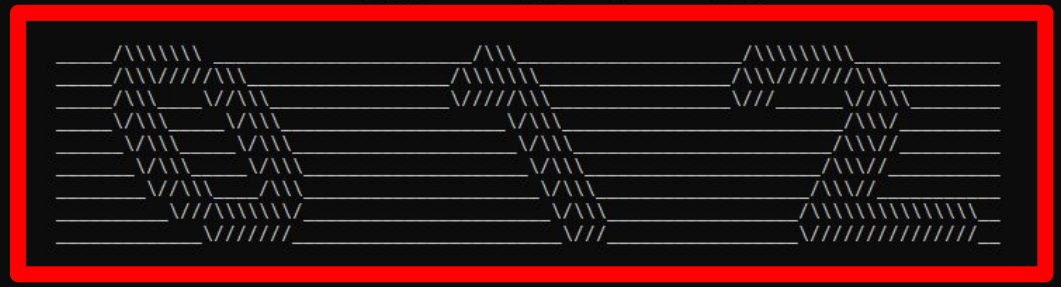

PLAYER 1
HEALTH [##############################] 10/10
AMMO [ ] 0/2
SCORE 10

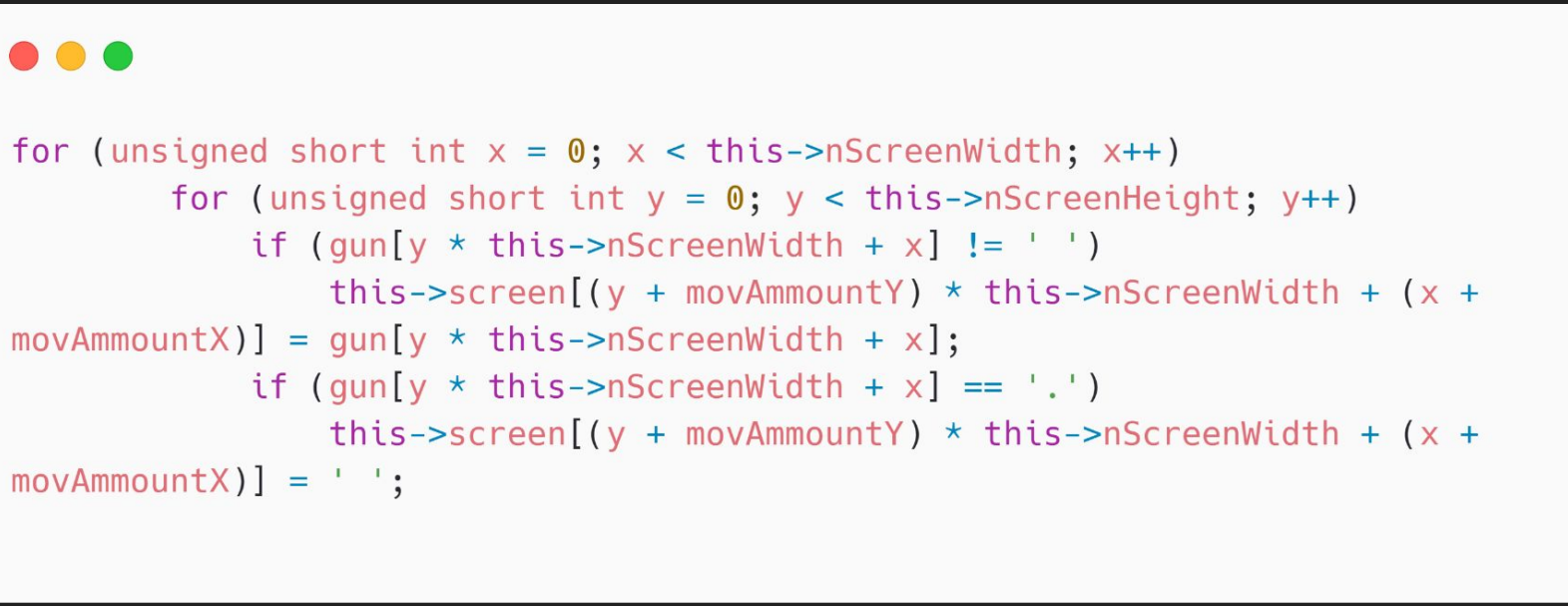
YOU DIED

SCORE:

YOU DIED

SCORE:





```
for (unsigned short int x = 0; x < this->nScreenWidth; x++)
        for (unsigned short int y = 0; y < this->nScreenHeight; y++)
            if (gun[y * this->nScreenWidth + x] != ' ')
                this->screen[(y + movAmmountY) * this->nScreenWidth + (x +
movAmmountX)] = gun[y * this->nScreenWidth + x];
            if (gun[y * this->nScreenWidth + x] == '.')
                this->screen[(y + movAmmountY) * this->nScreenWidth + (x +
movAmmountX)] = ' ';
```

```
for (size_t x = 0; x < larguraDaTela; x++)
    for (size_t y = 0; y < alturaDaTela; y++)
        if (overlayArma[y * larguraDaTela + x] != ' ')
            tela[y * larguraDaTela + x] = overlayArma[y * larguraDaTela + x];
        else if (overlayArma[y * larguraDaTela + x] == '.')
            tela[y * larguraDaTela + x] = ' ';
```

Estrutura de Dados e TAD do tipo Fila

Implementação da fila de ondas de inimigos, por meio da classe “Queue” e sua classe filha “WaveQueue”.

Características de cada onda de inimigos:

- Quantidade de mobs (1, 2 ou 3)
- Vida de cada mob (inversamente proporcional a quantidade de mobs, em média)
- Dano de cada mob (também inversamente proporcional a quantidade de mobs, em média)

```
template <typename T>
    class Queue {
        static const unsigned short int _ALLOC_BLOCK_SIZE = 5;
    private:
        T* _items;
        int _nEntriesAmmount;
        int _nCurrentSize;

        void _shiftBack();
        bool _allocNewBlock();
        bool _full();

    public:
        Queue();
        ~Queue();

        bool push(T element);
        bool pop(T& element);
        bool empty();
    };
```

```
class WaveQueue : public utils::Queue<MobsWave> {
private:
    MobsWave enemyWaves[5];
    int dificulty;
    Map map;

public:
    WaveQueue(Map map);
    bool pop(MobsWave& wave);
    int getDificulty();
};
```

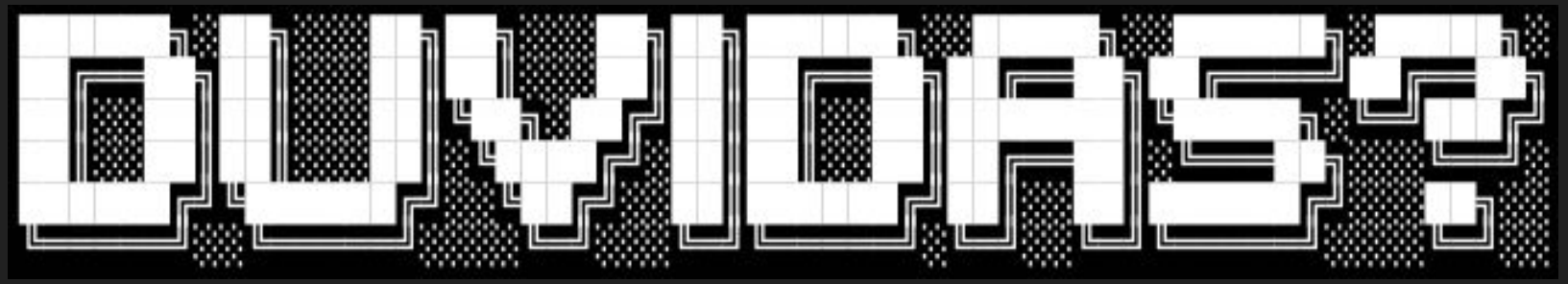
Extras

- Classes de Logger
- Arte ASCII
- Música atmosférica do jogo
- Queríamos fazer mais mecânicas, porém não tivemos tempo =(
- Agradecimento ao javidx9 pela ideia do renderizador.

Como Jogar

- Teclas WASD para movimentação do jogador
- Setas esquerda e direita para olhar em volta
- Espaço para atirar
- R para recarregar

Agradecimentos



Sintam-se livres para jogar e divulgar o nosso jogo à vontade!

Lembrem de ler o Readme!

https://drive.google.com/file/d/1ZDtWo98O4yaQfBbYKrd0kGaxFVAVJ6um/view?usp=sharing

https://github.com/LombardiDaniel/terminal-shooter