PROFESSORA: Renata Muniz do Nascimento	
CURSO: Ciência da Computação	
DISCIPLINA: Cálculo 1	
TURMA: Matutino	DATA: 22/04/2025
ALUNOS: Analice Coimbra Carneiro, Harry Zhu, João Pedro Da Silva, Rafaela Florêncio Morais	

TÍTULO DA ATIVIDADE: Modelagem Matemática e Funções Aplicadas ao Jogo de Terror

1. Definição Matemática do Jogo

O nosso jogo é uma experiência de terror em primeira pessoa, onde o jogador deve sobreviver das 20:00 até as 6:00 da manhã, monitorando o ambiente com sensores de movimento e câmeras. O jogador deve evitar ser capturado por um monstro que ronda aleatoriamente pela casa. O timer do jogo avança em ciclos de 5 minutos e, para sobreviver, o jogador precisa tomar decisões estratégicas sobre onde e quando posicionar seus equipamentos.

Mecânica do Jogo (Resumo):

- O timer avança em intervalos de 5 minutos (ex: 20:00 → 20:05 → 20:10 → ...).
- Cada hora do jogo equivale a **2 minutos e 30 segundos** da vida real.
- O jogador pode pegar objetos (como sensores e câmeras) apenas se estiver a até **2 unidades de distância.**
- O monstro se move aleatoriamente e pode ser detectado com sensores/câmeras baseados em **GameObjects com "Trigger".**

2. Conjuntos Numéricos e Operações Algébricas Utilizados

Conjuntos Numéricos:

- **- ℕ (naturais):** contagem de sensores colocados, número de movimentos do monstro, ciclos de tempo.
- **- ℤ (inteiros):** variações no tempo ou deslocamento espacial (positivos e negativos).
- **- ℝ (reais):** cálculo de distâncias entre jogador e objetos, tempo real decorrido.

Operações Utilizadas:

- **- Soma/Subtração:** movimentação do monstro ou jogador no espaço.
- **- Multiplicação/Divisão:** conversão de tempo do jogo para tempo real.
- **- Radiciação:** distância entre dois pontos no espaço.
- **- Potenciação:** distância entre dois pontos no espaço

3. Aplicação de Equações e Inequações

a) Atualização do Horário

No jogo, **cada hora do jogo equivale a 2 minutos e 30 segundos da vida real** — ou seja, **1 hora do jogo = 150 segundos reais**.

Como o timer do jogo avança de **5 em 5 minutos(do jogo)**, precisamos calcular **quantos segundos reais** passam a cada avanço de 5 minutos do jogo:

$$\frac{150\ segundos}{60\ segundos} \cdot 5\ minutos = 12,5\ segundos\ reais$$

Ao dividir os **1 hora do jogo**(150 segundos reais) por 60 segundos, descobrimos que a cada 2,5 segundos reais se passa 1 minuto no jogo. Multiplicando este resultado por 5 minutos do jogo(taxa de atualização do timer) resulta em 12,5 segundos reais.

Conclusão: O *timer do jogo* avança **5 minutos a cada 12,5 segundos reais.**

Fórmula para calcular o horário do jogo

O cálculo utilizado para converter o tempo passado na vida real para o tempo no jogo é feito a partir dessa fórmula:

$$tj = 20 \cdot 60 + 5 \cdot \left(\frac{tr}{12,5}\right)$$

Onde:

- *tr*: tempo real passados em segundos desde o início da partida.
- *tj*: tempo passado no jogo em minutos.
- 20: horário do início do timer.(“20:00”)
- 60: multiplica-se para transformar em minutos.
- 5: taxa de atualização do timer.

EX: Caso o tempo jogado seja de 450 segundos

$$tj = 20 \cdot 60 + 5 \cdot \left(\frac{450}{12,5}\right) \rightarrow tj = 20 \cdot 60 + 5 \cdot 36 \rightarrow tj = 1200 + 180 \rightarrow tj = 1380 \text{ minutos}$$

No código, o script divide esse tempo por **60**, para converter de volta em horas, então pega o **resto da divisão** desse número por 24, para garantir que o valor fique no intervalo de **0 a 23** (relógio de 24h).

Pois, caso o ***tr*** vá além de 1440 minutos (24 horas). Sem o **resto por 24**, o relógio poderia mostrar **"25:00", "26:00"**, etc. Com **resto por 24**, ele "reinicia" a contagem após 23h, voltando para 0h, como um relógio real.

EX: caso ***tj*** seja 1500 minutos

$1500 \div 60 = 25 \rightarrow resto\ de\ 25 \div 24 = 1$ O timer vai mostrar “01:00”

b) Verificação de Alcance para Coleta de Objetos

Para que o jogador possa pegar um objeto, a **distância d entre ele e o objeto** precisa ser:

$$d \leq 2$$

A distância é calculada utilizando a fórmula da **distância entre dois pontos no espaço**:

$$d = \sqrt{(Xp - Xo)^2 + (Yp - Yo)^2 + (Zp - Zo)^2}$$

Onde(Xp, Yp, Zp) é a posição do jogador e (Xo, Yo, Zo) é a posição do objeto.

Vale ressaltar que o comando *Physics.Raycast* realiza essas operações automaticamente.

c) Detecção por Câmeras e Sensores

A detecção acontece se o monstro **entra no raio de um "Trigger"**. Considerando um raio Rd de detecção:

$$d\,monstro \leq Rd$$

Essa mesma fórmula de distância é usada, aplicada entre a posição do monstro e do sensor/câmera.

d) Duração Total da Partida

A partida vai das **20:00 até 6:00**, ou seja, 10 horas no jogo = **600 minutos do jogo**. Se **1 hora de jogo** leva **150 segundos reais**, temos:

$$10\ horas\ no\ jogo \cdot 150\ segundos\ reais = 1500\ segundos\ reais = 25\ minutos\ reais$$

Então, **uma partida completa dura exatamente 25 minutos da vida real**.

e) Sistema de Sanidade do jogador

No jogo, o que vai definir se o jogador sobrevive ou não, é o sistema de sanidade. A sanidade do personagem é representada por um número inteiro que varia ao longo do tempo, de acordo com certos fatores (monstro, janela, barulhos e luz).

Cada fator apresenta um valor que irá diminuir na sanidade do jogador:

EVENTOS	VALOR
Monstro na janela	20
Janela vazia	5
Barulhos	6
Falta de luz	15

A mudança de sanidade é calculada com uma **equação simples de soma algébrica**:

$$Sf=Si+\Delta S$$

- Sf = sanidade final
- Si = sanidade inicial
- ΔS = variação da sanidade (positiva ou negativa)

Por exemplo, se a sanidade inicial for 50 e o personagem ver o monstro na janela e estiver sem luz:

$$\Delta S=-20+(-15)=-35$$

$$Sf=50+(-35)=15$$

- A sanidade nunca pode passar de 100 nem cair abaixo de 0. Isso é feito com uma **inequação dupla**, que representa um **intervalo fechado**:

$$0\leq S\leq 100$$

Essa condição garante que mesmo que os fatores causem uma grande perda ou ganho, o valor sempre será mantido dentro desse intervalo.