**📘 Aplicação de Derivadas no Jogo**

**🎯 Objetivo**

Demonstrar como as **derivadas** são aplicadas no desenvolvimento do jogo, especialmente para modelar **mudanças dinâmicas** como:

* Velocidade
* Aceleração
* Dificuldade
* Variação de pontuação

**📐 1. Interpretação Geométrica e Aplicação das Derivadas**

**🧠 O que é derivada?**

A **derivada** de uma função representa a **taxa de variação** de uma grandeza em relação a outra. Geometricamente, ela corresponde à **inclinação da reta tangente** a uma curva em determinado ponto.

**🚀 2. Aplicações Práticas das Derivadas no Jogo**

**🟢 A) Aceleração do Pulo – Derivada da Velocidade**

**Código:**

csharp

CopiarEditar

if (ForcaY > -9.81f)

{

ForcaY += -9.81f \* Time.deltaTime;

}

**Interpretação:**

* Aqui, ForcaY representa a **velocidade vertical** do personagem.
* -9.81f é a **aceleração da gravidade**.
* A derivada da posição é a **velocidade**, e a derivada da velocidade é a **aceleração**.

✅ **Aplicação da derivada**:  
A aceleração (constante) é derivada da velocidade e é aplicada em tempo real. Isso afeta a jogabilidade, pois simula fisicamente o movimento de pulo e queda do personagem.

**🟢 B) Velocidade de Movimento – Derivada da Posição**

**Código:**

controller.Move(movimento \* Time.deltaTime \* 15);

**Interpretação:**

* O vetor movimento representa uma direção.
* Multiplicar por Time.deltaTime é o mesmo que integrar uma velocidade.
* A velocidade é a derivada da **posição** em relação ao tempo.

✅ **Aplicação da derivada**:  
A posição do personagem muda com o tempo, e essa mudança é proporcional à velocidade. Assim, o jogador sente que o personagem **acelera ou desacelera** conforme pressiona os controles, o que é um reflexo direto da **derivada da posição**.

**🟢 C) Crescimento da Dificuldade com Pontuação – Derivada do Score**

**Código:**

public void AddPoint()

{

score++;

if (score >= maxScore)

{

SceneManager.LoadScene("Beefed Up! Cena Final");

}

}

**Expansão possível:**

Você poderia, por exemplo, aumentar a velocidade dos inimigos à medida que a pontuação aumenta:

agente.speed = 3 + score \* 0.5f;

**Interpretação:**

* A derivada do **score** em relação ao tempo (ds/dt) pode ser usada para ajustar dinamicamente a dificuldade.
* Se o jogador coleta moedas rapidamente, a taxa de variação do score é alta → **jogo fica mais difícil mais rápido**.

✅ **Aplicação da derivada**:  
A taxa de variação da pontuação serve como base para **escalar a dificuldade**, afetando inimigos, obstáculos ou até o tempo de exibição de legendas.

**📈 3. Interpretação Geométrica e Impacto na Jogabilidade**

**📉 A reta tangente**

* Quando analisamos a **trajetória do pulo**, por exemplo, a **reta tangente** em qualquer ponto da parábola representa a **velocidade instantânea vertical**.
* O ponto mais alto do pulo é onde a **derivada da velocidade é zero** (velocidade vertical nula), e o personagem começa a cair.

**🧠 Como isso afeta o gameplay?**

* O tempo que o jogador fica no ar depende dessa curva.
* Saber que a **mudança da posição depende da velocidade**, e esta da aceleração (gravidade), é essencial para calibrar corretamente os **valores de pulo e queda** no jogo.
* Exemplo: um salto muito forte (derivada muito alta) ou uma gravidade muito fraca (derivada pequena) pode desequilibrar o jogo.

**✅ Resumo Final – Conteúdo Esperado**

| **Conceito Matemático** | **Aplicação no Código** | **Impacto no Jogo** |
| --- | --- | --- |
| Derivada da posição | controller.Move(...) | Controla a **velocidade de movimento** |
| Derivada da velocidade | ForcaY += -9.81f \* Time.deltaTime | Simula **gravidade e aceleração de queda** |
| Taxa de variação da pontuação | score++ + lógica adaptativa | Permite **ajustar dificuldade dinamicamente** |
| Interpretação geométrica | Curva de pulo / tangente à trajetória | Ajuda a **balancear tempo no ar e sensação física** |