start by alura

Aula 7

Comandos em JavaScript

▶ Unidade

Lógica de programação: criando arte interativa com p5.js

Questão 1 – Cálculo de distância com dist()

No código abaixo, o objetivo é calcular a distância entre o ponto em que o cursor do mouse se encontra e um ponto oculto na tela usando a função **`dist()`** do p5.js. Qual seria uma maneira alternativa de calcular essa distância?

```
let distanciaX = mouseX - x;
let distanciaY = mouseY - y;
let distancia = dist(mouseX, mouseY, x, y);
```

Assinale a alternativa que contém uma nova versão correta dessa função:

a) `distancia = mouseX + mouseY + x + y;`

b) `distancia = sqrt(distanciaX * distanciaX + distanciaY * distanciaY);`

c) `distancia = pow(mouseX - x, 2) + pow(mouseY - y, 2);`

d) `distancia = (mouseX - x) / (mouseY - y);`

Alternativa A, incorreta. A soma dos valores não reflete a fórmula correta para cálculo de distância.

Alternativa B, correta. Essa é a implementação manual do Teorema de Pitágoras para calcular a distância entre dois pontos.

Alternativa C, incorreta. Esse código apenas soma os quadrados das diferenças, mas não tira a raiz quadrada necessária para obter a distância.

Alternativa D, incorreta. A divisão entre as diferenças de coordenadas não corresponde à fórmula de distância.

Questão 2 – Função dist()

Durante a aula, você aprendeu sobre o uso da função **`dist()`** para calcular a distância entre dois pontos.

Avalie as afirmativas a seguir e aponte quais são verdadeiras (V) e quais são falsas (F).

( ) A função **`dist(x1, y1, x2, y2)`** calcula a distância entre dois pontos **`(x1, y1)`** e **`(x2, y2)`**.

( ) O código **`sqrt((x2 - x1) * (x2 - x1) + (y2 - y1) * (y2 - y1))`** é equivalente à função **`dist()`**.

( ) A função **`dist()`** retorna a distância em pixels entre dois pontos.

( ) A função **`dist()`** não pode ser substituída por cálculos manuais, já que não há outro método equivalente para calcular distâncias.

Escreva a sequência correta de letras nas linhas a seguir:

Sequência correta: V | V | V | F

Comentário: este exercício é útil para revisar conceitos básicos sobre o cálculo de distâncias em p5.js e verificar a compreensão da função *dist()*.

Questão 3 – Comportamento do diâmetro do círculo

Analise o código a seguir:

```
let x;
let y
function setup() {
  createCanvas(400, 400);
  x = random(400);
  y = random(400);
}

function draw() {
  background(220);
  distancia = dist(mouseX,mouseY,x,y);
  circle(mouseX, mouseY, distancia);
}
```

Em seguida, associe as afirmações abaixo ao código:

Afirmação 1. O diâmetro do círculo depende da distância entre o cursor do mouse e um ponto oculto na tela.

Afirmação 2. Quanto menor a distância entre o cursor do mouse e o ponto oculto, menor será o diâmetro do círculo.

Por fim, assinale a alternativa correta:

a) Ambas as afirmações são verdadeiras.

b) Ambas as afirmações são falsas.

c) A primeira é verdadeira, mas a segunda não é uma justificativa correta para ela.

d) A primeira é falsa, e a segunda é uma justificativa correta para ela.

Alternativa A, correta. Ambas as afirmações são verdadeiras, pois o diâmetro realmente depende da distância, e quanto menor a distância, menor o diâmetro.

Alternativa B, incorreta. Para esta opção ser verdadeira, as duas afirmações precisam ser falsas. Veja se o diâmetro do círculo realmente não depende da distância.

Alternativa C, incorreta. Será que o diâmetro realmente diminui conforme a distância diminui? Para constatar isso, analise o comando *circle(mouseX, mouseY, distancia)*.

Alternativa D, incorreta. Para esta alternativa ser válida, a primeira afirmação teria que ser falsa, o que significa que o diâmetro do círculo não depende da distância. Analise o comando *circle(mouseX,mouseY, distancia)* no código para verificar o resultado esperado desse comando.