06

Bomba

Transcrição

[00:00] Nossa recursão funciona, mas não tem nada além do que um for não poderia ter. Era só validar se tinha um muro ali e parava o for. A gente aprendeu o break, por exemplo. Só que ainda é meio mentirosa, ela vai quatro para a direita e acabou. Fazer quatro para cima também é fácil, para direita, esquerda. Agora, como uma bomba se comportaria de verdade em um jogo do gênero?

[00:32] Vamos criar outro mapa. Repare como ele funciona. Tenho meu herói, a bomba, fantasmas. Uma bomba de verdade não andaria só para baixo, mas para todas as direções. Ela anda para direita e para a esquerda ao mesmo tempo. Quero conseguir alcançar todas as casas que são alcançáveis a quatro de distância. Para todos os lados, ao mesmo tempo.

[01:47] Uma abordagem inicial seria fazer um for. Fazermos um for para linha de menos quatro a mais quatro, um para a coluna, depois limpamos as linhas. Se eu fizer esse for, vou pegar a posição em que estou e vou andar menos quatro para a esquerda e menos quatro para cima. De lá, até quatro para baixo e quatro para direita vai limpar tudo. Mesmo que eu validasse as bordas, eu ainda limparia tudo que é quatro para direita, esquerda. Não faz sentido nenhum. Não é o que quero.

[03:00] A bomba anda, segue o caminho dos corredores, onde consegue chegar. Quero saber todos os pontos do meu mapa em que consigo andar com quatro de distância. Não quero sair explodindo todos os lados. O for não me deixa fazer isso dessa maneira. Não tem nada a ver com o que quero fazer.

[03:33] Vamos para a recursão. Na recursão, dizemos para ele executar a remoção para a direita, por exemplo. Ou para a esquerda. Ou para cima e para baixo. Preciso desses métodos também no meu herói.

[04:30] Estamos falando que na hora de fazer a recursão, devemos ver se queremos andar mesmo. Se você chegou na base, já andou o que tinha que andar de quatro. Senão, tenta andar um para a direita, para a esquerda, para cima, para baixo. Todos ao mesmo tempo, entre aspas. Porque o programa vai executar um por vez. Primeiro ele vai chamar o executa remoção para a direita, com a quantidade quatro. Aí ele chama a posição da direita com três e executa tudo isso de novo com três. Mas já na posição da direita, porque se ele andou um para a direita, a partir daqui ele consegue andar três em qualquer direção. A partir do próximo, ele consegue andar dois em qualquer direção. E depois um. E então não consegue fazer nada.

[05:30] Para toda posição que ele consegue chegar, tenta andar para todos os lados o mesmo número de passos. Se ele chegar numa posição qualquer, tenta andar a partir dessa posição a quantidade de passos que você quer dar.

[05:54] Agora sim, limpei tudo que tinha quatro de distância. O último fantasma sobrou e andou para a esquerda. Se eu rodar de novo, é capaz de andar para cima. Mas ele foi para a esquerda. A sacada foi que tentamos tudo que tinha quatro de distância.

[06:25] O for não faria isso. Pegamos nossa recursão, que diz que a partir de tal posição, ele deve andar quatro para a direita. Nós mudamos para qualquer lado. Ela não passa por muros porque bloqueamos. E é óbvio, ele só vai andar quatro no máximo para qualquer combinação de quatro. Qualquer combinação de quatro ele vai executar.

[07:03] Repare que com a recursão é muito fácil buscar todas as posições que estão a x de distância. Quais são todas as posições do mapa que com quatro passos consigo chegar? Faço uma função, recebo o número quatro, o quanto quero

andar, anda para todos os lados, passando o número três, vê se a posição é válida, depois verifico o número dois, o um, e paro no zero. Conseguimos passar por todas as posições válidas do mapa que alcançamos com quatro de distância.

[07:45] Com a recursão ao invés de chamar uma única vez, chamamos quatro, uma para cada lado. Com isso, seguimos os caminhos válidos, porque bloqueamos os inválidos para todos os lados.