1) **Como o algoritmo Scanline identifica os pares de interseções em cada linha horizontal do polígono?**

Explique o processo de detecção e ordenação das interseções com as arestas do polígono para realizar o preenchimento.

Resposta: O algoritmo passa linha por linha (as chamadas "scanlines") na vertical da imagem, verificando onde essa linha cruza com as arestas do polígono. Para cada aresta, ele calcula o ponto exato onde a scanline corta ela, usando uma fórmula que depende das coordenadas dos dois pontos da aresta. Depois de encontrar todos os pontos de interseção naquela linha, o algoritmo coloca esses valores em uma lista e ordena do menor pro maior no eixo X. Aí ele começa a preencher os pixels de dois em dois, ou seja, entre o primeiro e o segundo ponto, entre o terceiro e o quarto, e assim por diante.

2) **Por que é necessário ignorar arestas horizontais durante a construção da tabela de arestas no algoritmo Scanline?**

Reflita sobre como essas arestas poderiam afetar negativamente o cálculo das interseções e o preenchimento correto.

Resposta: Porque as arestas horizontais não ajudam a definir onde começa ou termina o preenchimento numa linha específica. Elas estão na mesma altura da scanline, então se fossem incluídas, poderiam gerar interseções duplicadas ou confusas. Isso atrapalharia o emparelhamento dos pontos de entrada e saída do polígono, podendo preencher áreas erradas ou fazer com que o polígono fique com "buracos" ou linhas extras. Ignorando essas arestas, o algoritmo evita esses erros e mantém o preenchimento correto.

3) Quais seriam as diferenças principais se este algoritmo fosse aplicado a polígonos não convexos?

Analise as limitações do algoritmo Scanline apresentado neste código e o que seria necessário modificar para lidar com casos mais complexos.

Resposta: Em polígonos convexos, as interseções em cada scanline sempre aparecem em pares bem definidos, então é fácil saber onde começa e onde termina o preenchimento. Já nos polígonos não convexos, isso pode se complicar, porque uma mesma linha pode cortar o polígono várias vezes em zigue-zague. Nesse caso, o algoritmo teria que tomar mais cuidado com a ordem e com os pares de interseções. Seria necessário melhorar a lógica de emparelhamento ou usar alguma estrutura mais inteligente para lidar com esses múltiplos cruzamentos e garantir que ele preencha só o que está “dentro” do polígono, mesmo que ele tenha reentrâncias.

4) De que maneira o algoritmo implementado evita erros visuais causados por interseções múltiplas ou coincidentes?

Descreva o papel da ordenação e do emparelhamento das interseções no resultado final da renderização.

Resposta: Ele resolve isso principalmente ordenando as interseções da esquerda para a direita em cada scanline. Isso garante que ele sempre vai preencher os pixels entre pares corretos. Além disso, o algoritmo normalmente verifica se duas arestas compartilham o mesmo vértice para evitar contar interseções repetidas ou muito próximas, o que causaria duplicidade no preenchimento. Com a ordenação e o emparelhamento bem feitos, o algoritmo consegue evitar que partes erradas sejam pintadas ou que fiquem falhas na imagem.