1 a) Sim, é possível concluir que o perceptron de camada simples pode funcionar com esses dados, pois parece que os dados são linearmente separáveis pela linha verde. Ou seja, há uma linha que divide as maçãs das laranjas, o que é a condição necessária para o funcionamento de um perceptron de camada simples. Se essa separação linear não fosse possível, o perceptron de camada simples não seria adequado, e técnicas mais complexas, como redes neurais multicamadas, seriam necessárias.

b) Excel   
  
c) código no git,  
  
Explicação do Código:

Estrutura do Perceptron:

Ele começa com pesos aleatórios (entre -1 e 1) e um bias.

A função de ativação (degrau) classifica a saída como 1 (maçã) ou 0 (laranja).

A função de treino ajusta os pesos e bias a cada iteração com base no erro.

Dados de Treinamento:

O código usa os dados de peso e pH das frutas como entrada.

Os rótulos 1 (maçã) e 0 (laranja) são usados como saídas esperadas.

Treinamento:

O perceptron é treinado por 10 iterações, e os pesos e bias são ajustados em cada uma delas. Os pesos resultantes de cada iteração são exibidos no console.

Testando Novos Casos:

Após o treinamento, o código gera cinco novos casos com diferentes valores de peso e pH para testar o modelo e prever se cada caso é uma maçã ou laranja.   
  
d)  
  
Em problemas mais complexos, com maior variação de características, precisamos de muitos treinamentos para que a RNA possa:

1. **Aprender a diversidade dos dados**: Mais exemplos ajudam a RNA a identificar padrões e melhorar a generalização.
2. **Evitar overfitting**: Treinamentos prolongados ajudam a RNA a não se ajustar demais aos dados de treinamento, melhorando sua performance em novos dados.
3. **Ajustar redes complexas**: Modelos mais complexos (com mais neurônios e camadas) precisam de mais iterações para encontrar os melhores pesos e bias.