Explicação do Código

1. **Bucket Sort**:
    - **Função `bucket_sort`**: O algoritmo distribui as notas em 10 baldes, com base na parte inteira da divisão da nota por 10. Após isso, ele ordena cada balde e junta todos os baldes ordenados em uma lista única.
    - **Como funciona**:
        - Os baldes são criados para cada intervalo de 10 pontos (0-9, 10-19, ..., 90-100).
        - As notas são distribuídas nos baldes apropriados e, em seguida, cada balde é ordenado. Finalmente, todos os baldes são concatenados para formar a lista ordenada.
2. **Interpolation Search**:
    - **Função `interpolation_search`**: Realiza a busca pela fórmula de interpolação, que calcula uma posição provável de um valor com base na distribuição dos elementos. Isso pode ser mais eficiente que a busca binária em alguns casos, quando os dados são uniformemente distribuídos.
    - **Como funciona**:
        - O algoritmo calcula uma posição provável para o alvo com base na fórmula de interpolação.
        - Se o valor na posição calculada é o alvo, retorna a posição; se o valor na posição é menor, a busca continua na parte superior da lista; caso contrário, ela continua na parte inferior da lista.
3. **Teste com Notas de Alunos**:
    - **Lista de notas**: Uma lista de notas é fornecida.
    - **Ordenação das notas**: As notas são ordenadas usando **Bucket Sort**.
    - **Busca da nota**: O programa procura por uma **nota específica** na lista ordenada usando **Interpolation Search**.

Considerações

- **Bucket Sort** é eficiente para listas com distribuições uniformes, como notas em uma turma de alunos (onde as notas variam entre 0 e 100).
- **Interpolation Search** funciona bem quando os dados estão ordenados e têm uma distribuição uniforme. Se as notas forem muito desiguais ou se houver grandes lacunas, o desempenho pode ser inferior ao de outros algoritmos de busca.
    -