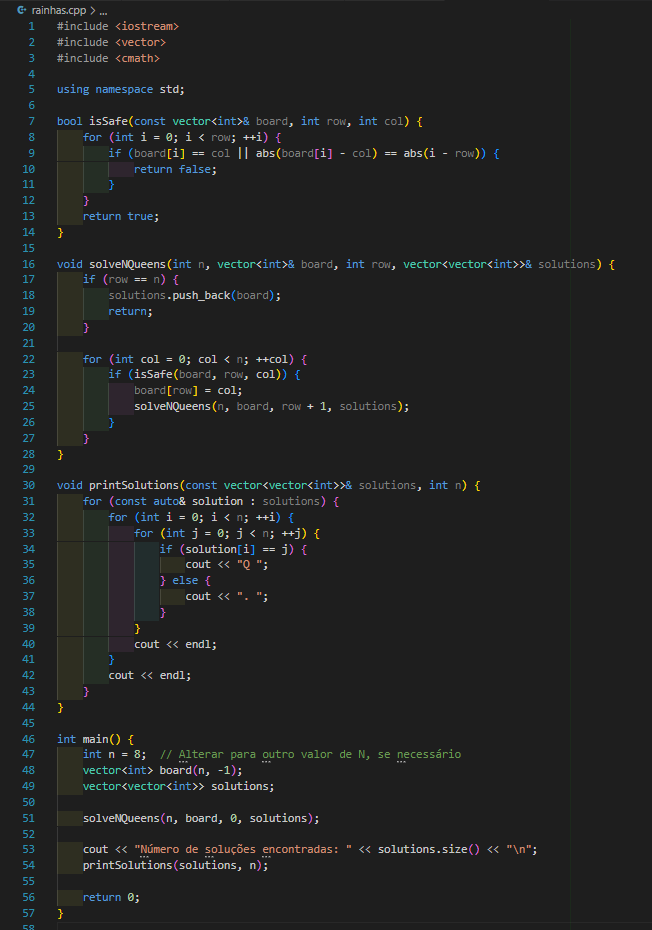
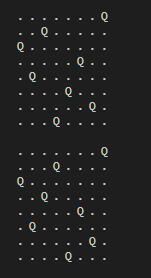
Conjunto 04 – O problema das N-Rainhas consiste em colocar 𝑁 rainhas em um tabuleiro 𝑁 × 𝑁 forma que nenhuma rainha ataque outra.

**1. Problema das N-Rainhas**

**Casos de Teste**

1. Entrada *N* = 4:
   1. Saída Esperada:
2. Entrada *N* = 8:
   1. Saída Esperada:
3. Entrada *N* = 16:
   1. Saída Esperada:

**Análise de Complexidade**

Problema das N-Rainhas:

* Tempo: devido ao número de permutações possíveis de *N.*
* Espaço: para armazenar o tabuleiro e as chamadas recursivas.

**Operação Básica**

1. **Verificação de segurança (Função isSafe)**

* A operação básica dessa função é verificar se uma rainha pode ser colocada em uma posição (linha, coluna) sem conflito com outras rainhas. Para isso, o algoritmo percorre todas as linhas anteriores e compara as colunas e diagonais. A complexidade dessa operação é O(r), onde r é o número de linhas já preenchidas.

1. **Colocação da Rainha (Função solveNQueens)**

* A operação básica é a tentativa de colocar uma rainha em uma posição válida na linha atual. O algoritmo tenta colocar a rainha em todas as colunas de uma linha e recursivamente passa para a próxima linha, caso a posição seja válida. A operação recursiva ocorre em n linhas e n colunas, com uma complexidade total de O(n!) no pior caso.

1. **Backtracking (Função solveNQueens)**

* O algoritmo retorna (backtrack) para a linha anterior se uma solução inválida for encontrada. Essa operação envolve desfazer a colocação da rainha na linha atual e tentar a próxima posição. O backtracking é realizado durante a exploração do espaço de soluções.

1. **Impressão das Soluções (Função printSolutions)**

* A operação básica dessa função é percorrer todas as soluções encontradas e imprimir o tabuleiro. Para cada solução, a função imprime cada linha do tabuleiro e, dentro de cada linha, imprime se a célula contém uma rainha ("Q") ou não ("."), o que envolve O(n2) de operações por solução.

**Expressão de Custo Completa**

Para se obter a expressão de custo completa, é necessário combinar as complexidades de todas as operações:

* Complexidade de isSafe: Para cada tentativa de colocar uma rainha, a função isSafe executa em O(r), onde r é o número de linhas preenchidas até o momento.
* Complexidade de solveNQueens (chamadas recursivas): No pior caso, o número de chamadas recursivas é O(n!), pois o algoritmo explora todas as permutações possíveis de colunas nas n linhas.
* Complexidade total de backtracking: O algoritmo realiza O(n!) tentativas de solução (pois são exploradas todas as possíveis disposições das n rainhas), e em cada tentativa, a função isSafe é chamada para cada linha, o que tem um custo de O(r).
* Complexidade de impressão de soluções: Se houver s soluções, o custo de impressão será O(s⋅n2).

**Complexidade Total**

* A complexidade total do algoritmo de resolução do problema das N Rainhas é dada pela combinação da complexidade de backtracking e a complexidade de impressão das soluções:

O (n!) ⋅ O (n) + O (s⋅n2)

Onde:

* O(n!) é o custo de explorar todas as permutações possíveis para colocar as n rainhas.
* O(n) é o custo de verificar a segurança para cada tentativa de colocação de rainha.
* O(s⋅n2) é o custo de imprimir as soluções, onde s é o número de soluções encontradas.

Casos Especiais:

* Para n grande, a complexidade de O(n!) domina, tornando o algoritmo exponencial em relação a n.
* O número de soluções s pode variar dependendo do valor de n, mas em muitos casos o número de soluções cresce exponencialmente.

**Classe de Problemas**

O Problema das N Rainhas se encaixa na classe de problemas NP-completos quando se busca encontrar uma solução ótima (ou uma solução para o número máximo de soluções possíveis).

**Classe NP**

* O problema das N Rainhas se encontra na classe NP porque, dado um possível posicionamento das rainhas, pode-se verificar em tempo polinomial (ou seja, O(n)O(n)O(n)) se a configuração é válida, ou seja, se nenhuma rainha está atacando outra.
* A solução pode ser verificada rapidamente, mas encontrar a solução (sem usar uma abordagem de força bruta) é um problema difícil.