GeeksforGeeks

Welcome All

@

Mastering DSA Basic to Advance

(Course for Product-Based Companies)

(Arrays + Recursion + Backtracking)

Presented By: Puneet Kansal

Arrays

Introduction to Array

- **Definitions:**

  An array is a collection of similar items stored at contiguous memory locations.

- **1-D Array:**

  I. **Accessing element in 1-D Array:**

  *Base + (index * element size)*

Why array indexing starts with Zero?

*Note, vectors and Arraylist also store data in Contiguous memory locations. Although list in python store data in non-contiguous memory locations (using Array of Pointers).

Types of Arrays

Fixed/Variable sized array

- Example (C/C++):
- int a[100] //Fixed length
- int a[n] //variable length
- int *a = new int[n] //variable length
- int a[ ] = {1,2,3,4} //Fixed length

- Example (Java):
- int [] a = new int[100] //Fixed length
- int [] a = new int[n] //variable length
- int a[ ] = {1,2,3,4} //Fixed length

Dynamic sized array

- C++ : Vector
- Java: Arraylist
- Python: list

Note: In variable length array, size of array can be taken from user at run time while in dynamic sized array, size of array is automatically increase and decreased at run time.

C99, C++ 14 onward and Java support variable length array.

Largest Contiguous “pair” Sum

Given an integer array **arr[]** of size **N**, the task is to find **contiguous pair** {a, b} such that sum of both elements in the pair is maximum. Print the maximum sum.

Example1:

Input: *N= 4**,** arr[] = {1, 2, 3, 4}*

Output: *7*

Example2:

Input: *N=5**,** arr[] = {11, -5, 9, -4, 2}*

Output: *6*

Max Sum Subarray of size K

Given an **array** of integers of size **N**, and given a number **K**. Return the maximum sum of a subarray of size K.

NOTE: A subarray is a “contiguous” part of any given array.

Example1

Input: N = 5, K = 3 Arr = [1,3,5,2,4]

Output: 11

Example2

Input: N = 8, K = 4 Arr = [1,2,3,4,3,2,1,0]

Output: 12

Expected Time Complexity: O(N)

Constraints: $1<=N<=10^6$, $1<=K<=N$

https://practice.geeksforgeeks.org/problems/max-sum-subarray-of-size-k5313/1

longest sum contiguous subarray
(Array-Problem4)

amazon

- Given an array arr[] of size N. The task is to find the sum of the contiguous subarray within a arr[] with the largest sum.

Example1:

- ***Input:*** *arr[] = {2, -3, 4, 5}*
- ***Output:*** *9*

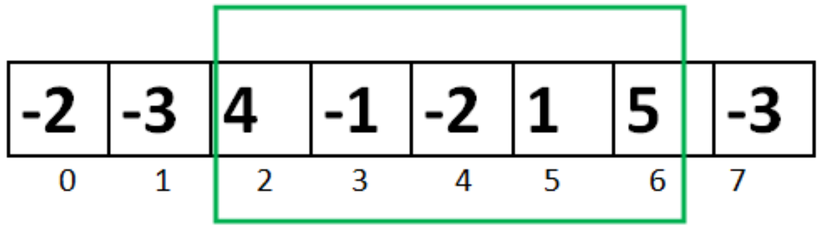
Example2:

- ***Input:*** *arr[] = {6, -5, 9, 1, 2}*
- ***Output:*** *13*

Example3:

- ***Input:*** *arr[] = {1, 2, -3, 2, 3, -4, 2, 4, -5, 2}*
- ***Output:*** *7*

Sample Example



4 + (-1) + (-2) + 1 + 5 = 7

Maximum Contiguous Array Sum is 7

How we can solve it in O(n) time?

https://practice.geeksforgeeks.org/problems/kadanes-algorithm-1587115620/1

Solution

Largest Sum Contiguous Subarray

1	2	-3	2	3	-4	2	4	-5	2

- **Pseudocode:**
- *Initialize:*
  maxsum = a[0]
  curr_sum = 0
- *Loop for (i=0 to n-1)*

  (a) curr_sum = curr_sum + a[i]

  (b) if(maxsum < curr_sum)
  maxsum = curr_sum

  (c) if(curr_sum <= 0)
  curr_sum = 0

return maxsum

Kadane's Algorithm

Intuition:

1. Neglect previous terms if there sum is less then or equal to Zero.
2. Keep record of Max till now.

Prefix sum for a given range
(Array-Problem5)

- Given an array of N integers and Q queries, each query having a range from index L to R. Find the prefix-sum for the range L to R.
- Note: Assume 0 based indexing.

Example1:

Input: a[ ] = {1, 2, 3, 4} Q = 2

L1 = 0, R1 = 3

L2 = 2, R2 = 3

Output: 10 7

Example2:

Input: a[ ] = {2, 8, 3, 9, 6, 5, 4} Q = 2

L1 = 0, R1 = 3

L2 = 2, R2 = 5

Output: 22 23

Solution 1: O(R-L+1)//for one query

Solution 2: (Prefix sum approach)

```
#include <iostream>
using namespace std;
int main() {
    int arr[] = {2, 8, 3, 9, 6, 5, 4};
    int n = sizeof(arr) / sizeof(arr[0]);
    int l, r;
    cout << "Enter the value of l: ";
    cin >> l;
    cout << "Enter the value of r: ";
    cin >> r;
    calculatePrefixSum(arr, n, l, r);
    return 0;
}
```

Solution

Prefix sum for a given range

```
// Function to calculate prefix sum and result
void calculatePrefixSum(int arr[], int n, int l, int r)
{
        int ps[n];
        ps[0] = arr[0];
        for (int i = 1; i < n; i++)
                ps[i] = ps[i - 1] + arr[i];

        if (l == 0)
                cout << ps[r];
        else
                cout << ps[r] - ps[l - 1];
        }
```

Time Complexity: *O(1)*

//Time complexity for one query after having prefix sum

Two Sum II – Input Array is Sorted

(Array-Problem2)

Given a 1-indexed array of integers numbers that is already sorted in non-decreasing order, find two numbers such that they add up to a specific target number. Let these two numbers be numbers[index1] and numbers[index2] where 1 <= index1 < index2 <= numbers.length.

Return the indices of the two numbers, index1 and index2, added by one as an integer array [index1, index2] of length 2.

Your solution must use only constant extra space.

Example 1:

Input: sum = 17, arr[] = {2, 5, 8, 12, 30}

Output: [2,4], Explanation: The pair is 5, 12

Constraints: 2 <= N

Expected Time Complexity: O(N).

https://leetcode.com/problems/two-sum-ii-input-array-is-sorted/description/

*in non-decreasing order, every next element is greater than or equal to , but not less than the previous one.

“Pair” with given sum in a sorted array

(Array-Problem6)

Given a sorted array and a sum, find if there is any pair of elements with given sum?

Example 1:

Input: sum = 17, arr[] = {2, 5, 8, 12, 30}

Output: True

Explanation: The pair is 5, 12

Constraints: 2 <= N

Expected Time Complexity: O(N).

https://practice.geeksforgeeks.org/problems/pair-with-given-sum-in-a-sorted-array4940/1

Count pair sum
(Array-Problem7)

amazon

Given two sorted arrays(arr1[] and arr2[]) of size M and N of distinct elements. Given a value Sum. The problem is to count all pairs from both arrays whose sum is equal to Sum.
Note: The pair has an element from each array.

Example 1:
Input:
M=4, N=4 , Sum = 10
arr1[] = {1, 3, 5, 7}
arr2[] = {2, 3, 5, 8}
Output: 2
Explanation: The pairs are: (5, 5) and (7, 3).

https://practice.geeksforgeeks.org/problems/count-pair-sum5956/1?page=1&difficulty[]=-1&category[]=Sorting&sortBy=submissions

Container With Most Water
(Array-Problem3)

Apple

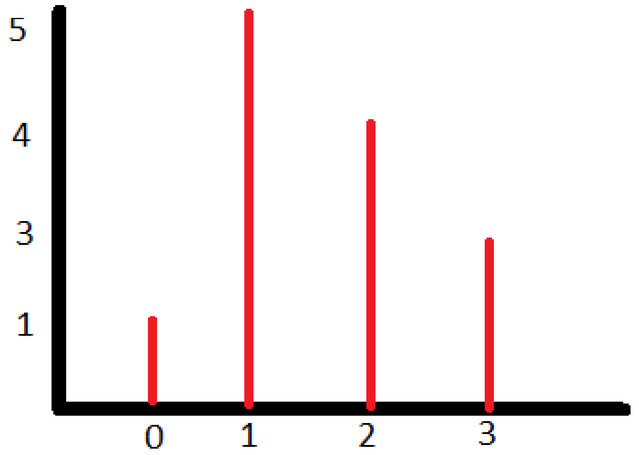
You are given an integer array height of length n. There are n vertical lines drawn such that the two endpoints of the ith line are (i, 0) and (i, height[i]).

Find two lines that together with the x-axis form a container, such that the container contains the most water.

Return the maximum amount of water a container can store.

Example 1: N = 4, a[] = {1,5,4,3}
Output: 6
Explanation: (3-1) * min(5,3)
total area = 2 * 3 = 6.

Expected Time Complexity: O(N).



https://leetcode.com/problems/container-with-most-water/description/

Find triplets with zero sum

(Array-Problem9)

Flipkart

Given an array arr[] of n integers. Check whether it contains a triplet that sums up to zero.

Note: Return true, if there is at least one triplet following the condition else return false.

- **Example:**

  Input: *n = 5, arr[] = {0, -1, 2, -3, 1}*

  Output: *true*

  Explanation: *0, -1 and 1 forms a triplet with sum equal to 0.*

Constraints:

$1 <= n <= 10^4$

$-10^6 <= A_i <= 10^6$

Expected Time Complexity: $O(n^2)$

Expected Auxiliary Space: $O(1)$

https://practice.geeksforgeeks.org/problems/find-triplets-with-zero-sum/1?page=1&difficulty[]=-1&category[]=Searching&category[]=Binary%20Search&sortBy=submissions

Trapping Rain Water

(Array-Problem10)

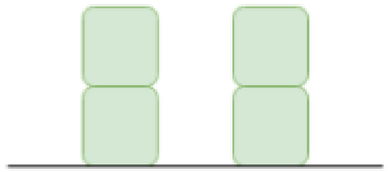
Given n non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it is able to trap after raining.

Example 1:

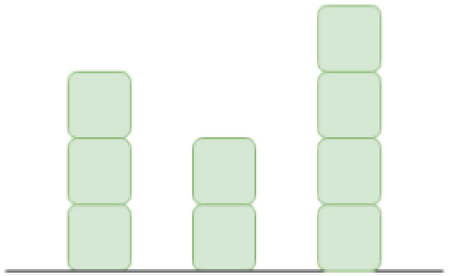
Input: arr[] = {2, 0, 2}

Output: 2

Explanation:



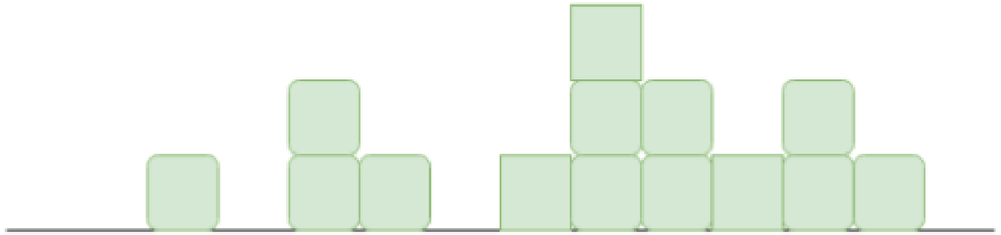
Example 2:



Output: 7

Water stored = 0 + 3 + 1 + 3 + 0 = 7

Example 3:



Output: 6

https://practice.geeksforgeeks.org/problems/trapping-rain-water-1587115621/1

Recursion

Introduction to Recursion

- ***Definition:***

  The process in which a function calls itself directly or indirectly is called recursion.

- ***Why use recursion:***
  - *To solve big problem in the form of small problem.*
  - *Easier to understand by programmer although difficult to understand by CPU.*

- ***Example :***

```
#Factorial using recursion
def fact(n):
    if n==1:
        return 1
    else:
        return n*fact(n-1)
```

Recursion Problem-5

What is the function of given code?

```
        //Multiple recursive calls

int foo(int n)
{
    if (n <= 1)
        return n;
    return foo(n - 1) + foo(n - 2);
}
```

Output: n^{th} Fibonacci Number

0	1	2	3	4	5	6	7	8	9
0	1	1	2	3	5	8	13	21	34

What is tail recursion?

- *A recursive function is tail recursive when a recursive call is the last thing executed by the function.*
- The tail recursive functions considered better than non tail recursive functions as tail-recursion can be optimized by the compiler.

Example:

```
void print(int n)
{
    if (n < 0)  return;
    cout << " " << n;

    // The last executed
statement is recursive call
    print(n-1);
}
```

```
void print(int n)
{
start:
    if (n < 0)
       return;
    cout << " " << n;

    // Update parameters of recursive call
    // and replace recursive call with goto
    n = n-1
    goto start;
}
```

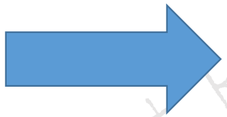
- *No extra stack space is required the tail recursive programs.*

Tail recursion
Continue…

*Can a **non-tail recursive** function be written as **tail-recursive** to optimize it?*

What is the output?

```
void fun(int n)
{
    if(n == 0)
        return;

    fun(n - 1);
    cout<<n<<" ";
}

int main() {  fun(3);  }
```



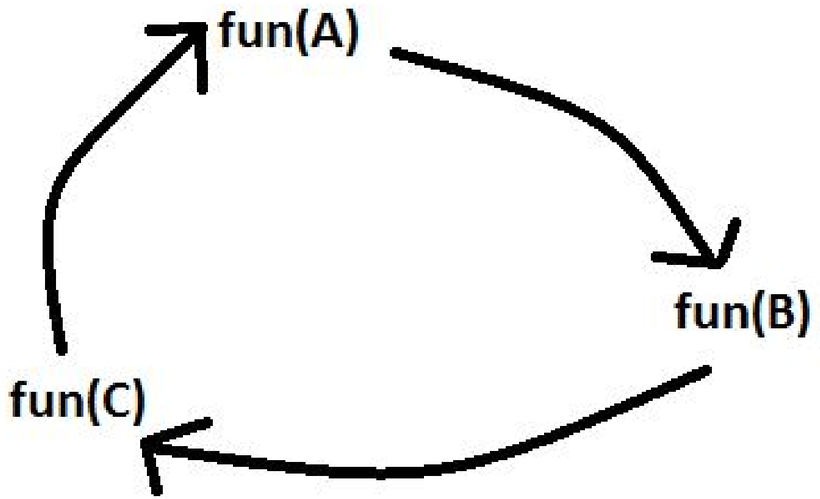
```
void fun(int n, int k)
{
    if(n == 0)
        return;

    cout<<k<<" ";
    fun(n - 1, k + 1);
}

int main() { fun(3, 1); }
```

***Not every non-tail recursive function be written as tail-recursive.**

** GCC, Clang, Lisp, Hashkell, Scala supports tail recursion optimization. C, C++, Java and Python don't support tail recursion optimization. Although, this also depends on compiler to compiler such as "GNU Compiler Collection (GCC)" includes C, C++, Objective-C, Fortran, Java, Ada, and Go languages in its package and support tail recursion optimization.

Indirect Recursion

- *if the first function calls the second function, the second function calls the third function, and again the third function calls back the first function.*



Pow(x,n)

facebook

Implement pow(x, n), which calculates x raised to the power n (i.e., x^n).

Example:
Input: x = 2, n = 10
Output: 1024

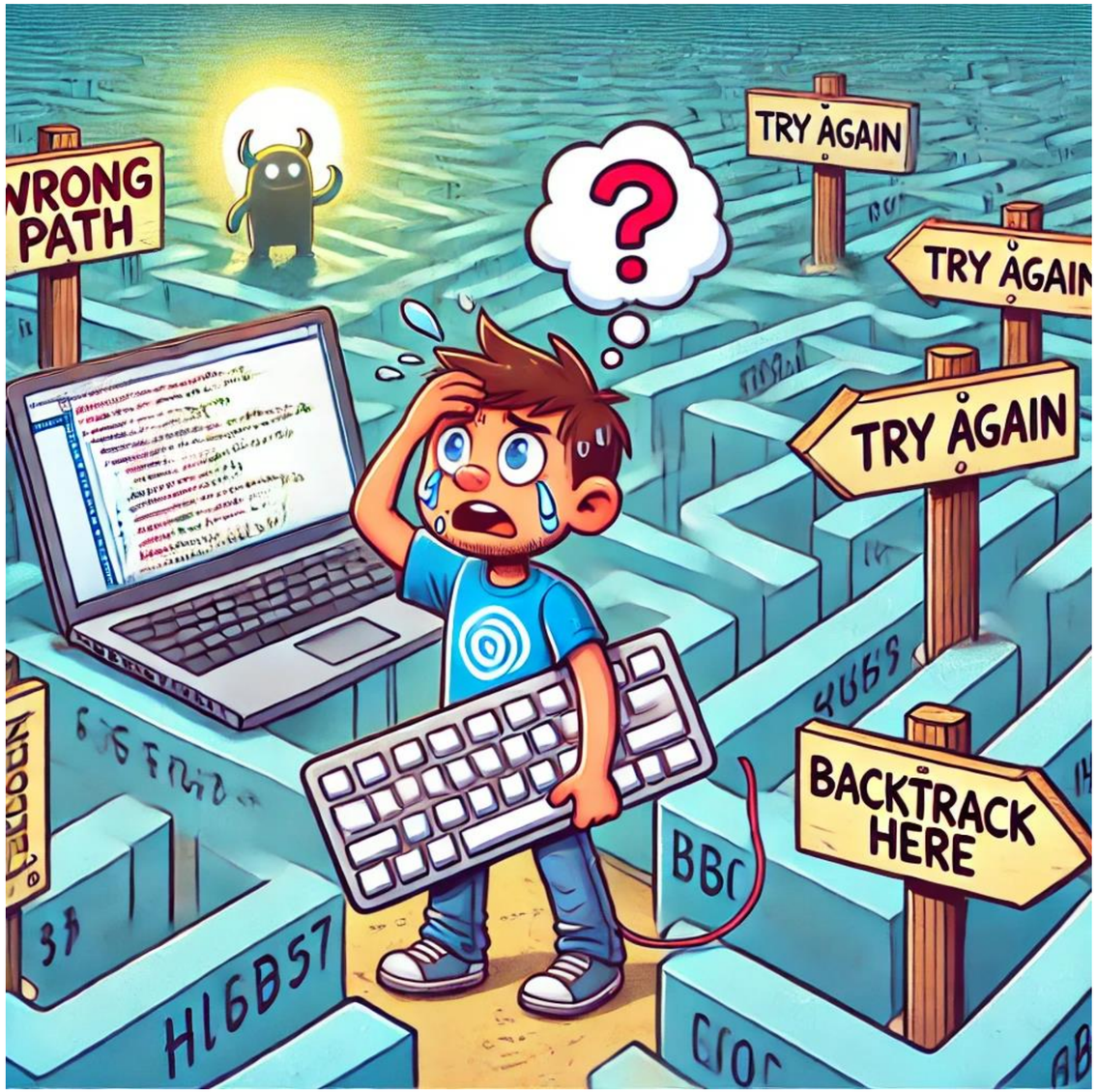
Example 2:
Input: x = 2.1, n = 3
Output: 9.261

Example 3:
Input: x = 2, n = -2
Output: 0.25
Explanation: $2^{-2} = 1/2^2 = 1/4 = 0.25$

https://leetcode.com/problems/powx-n/description/

Expected Time Complexity: O(logn)

Backtracking

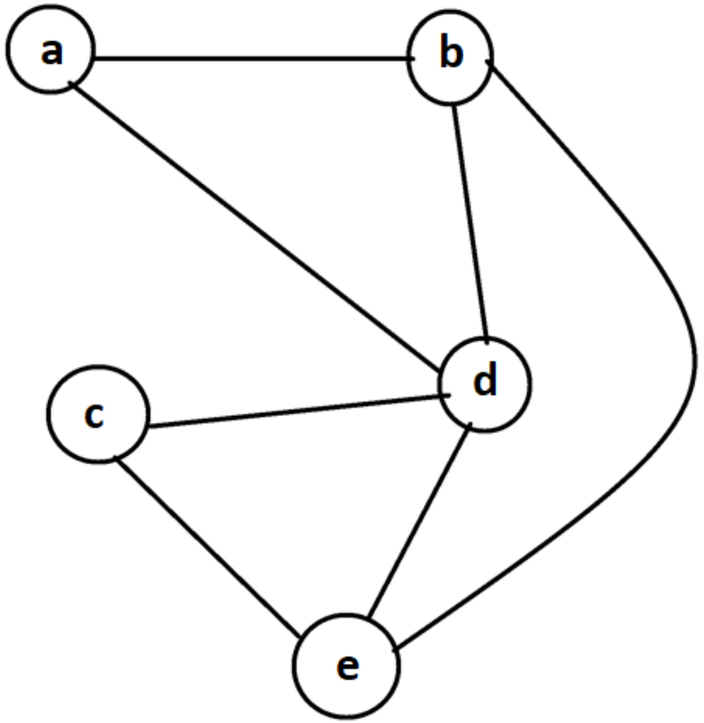


Introduction to Backtracking

- Backtracking is like trying different paths, and when you hit a dead end, you backtrack to the last choice and try a different route.

- It is a Brute force approach which *check all possible conditions* to find the solution.

With how many minimum colors we can color this graph, such a way that no two adjacent vertices have same color?



Backtracking Problem-1

Print all possible combinations of the numbers 1, 2, 3.

Output:

1 1 1	2 1 1	3 1 1
1 1 2	2 1 2	3 1 2
1 1 3	2 1 3	3 1 3
1 2 1	2 2 1	3 2 1
1 2 2	2 2 2	3 2 2
1 2 3	2 2 3	3 2 3
1 3 1	2 3 1	3 3 1
1 3 2	2 3 2	3 3 2
1 3 3	2 3 3	3 3 3

Backtracking Problem-2

Print all possible ways to visit 3 cities, considering each city can only be visited once.

Input:

cities = {"Ambala", "Pathankot", "jaisalmer"};

Output:

Ambala	Pathankot	jaisalmer
Ambala	jaisalmer	Pathankot
Pathankot	Ambala	jaisalmer
Pathankot	jaisalmer	Ambala
jaisalmer	Ambala	Pathankot
jaisalmer	Pathankot	Ambala

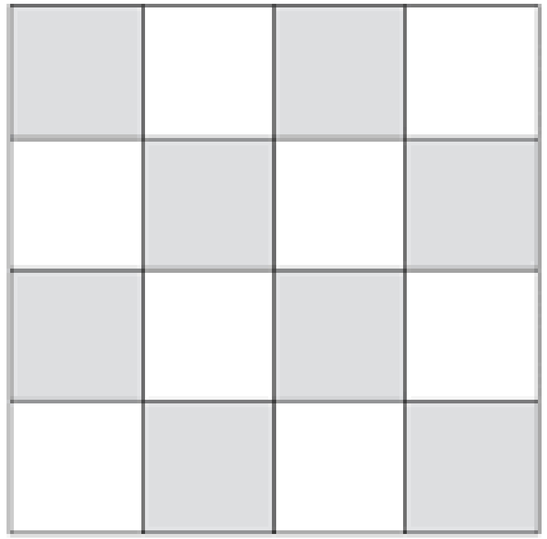
N-Queen Problem

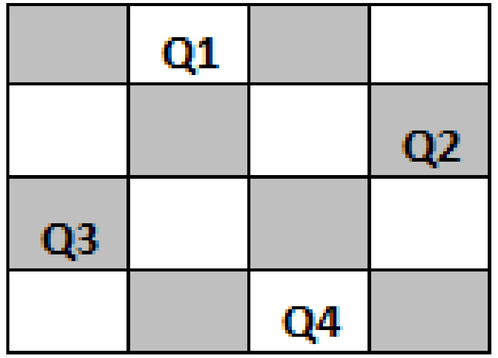
D E Shaw & Co

Problem statement:

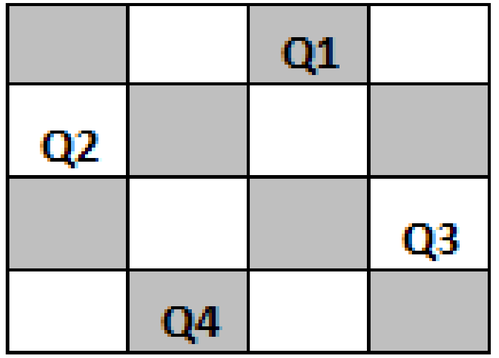
The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other.

Can you arrange 4 queens on a 4x4 chess board such that they don't attach each other?





Solution 1



Solution 2

https://www.geeksforgeeks.org/problems/n-queen-problem0315/1


THANK
YOU!