**This Note has been prepared by Partow Moradi, it is free to use. This note has been sorted from multiple books and multiple videos. wish it helps you in the way of programming.**

**If you have any question, you can reach me by My email address:** [**perry.moradi@yahoo.com**](perry.moradi@yahoo.com)

**-------------------------------------------------------------------------------------------------------------------------------**

What is data structure?

Data Structure is a different way of organizing data on your computer, that can be used effectively.

What is an Algorithm?

Set of instruction to perform a Task. Algo in computer science: set of rules for a computer program to accomplish a Task.

Why are Data Structures and algo are important?

Because it serves these steps:

1. Input
2. Processing
3. Output

Types of Data Structure: built in data structure.

1. Primitive
2. Integer
3. Float
4. Character
5. Boolean
6. Non-Primitive: User defined data structures
7. Linear: Items are arranged in memory in linear sequential manner, they can be either static or dynamic.
8. Static: Size is fixed
9. Array
10. Dynamic: is not fixed
11. Linked list
12. Stack
13. Queue
14. Non- linear:
15. Tree
16. Graph

Types of Algorithms

1. Simple recursive algorithms

An algorithm that calls itself recursively.

1. Divide and conquer algorithms.
2. Divide the problem into smaller subproblems of the same type and solve these subproblem recursively. And combine the solution to the original problem. It contains at least two recursive calls.

Example: Quick Sort and merge sort.

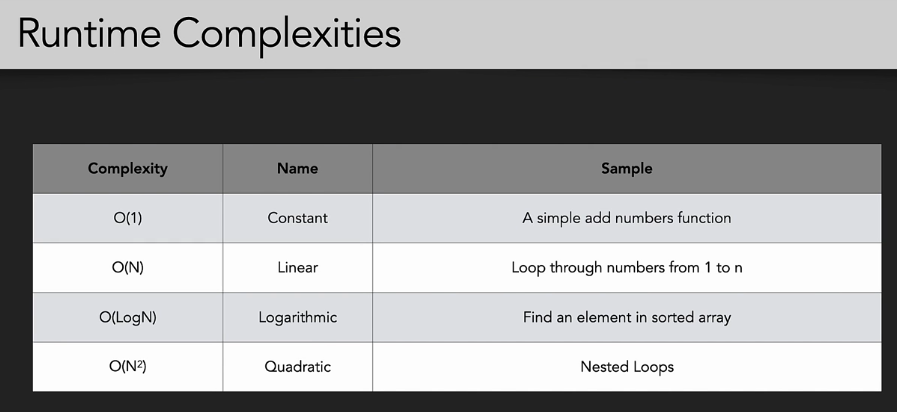
1. Dynamic programming algorithm
2. They worked based on memorization.
3. To find the best solution.
4. Greedy programming algorithm
5. We take the best we can without worrying about future consequences.
6. We hope that by choosing a local optimum solution at each step, we will end up at a global optimum solution.
7. Brute force algorithm
8. Use a random number at least once during the computation to decide.
9. Randomized algorithm
10. Use a random number at least once during the computation to decide.

What is Big O?

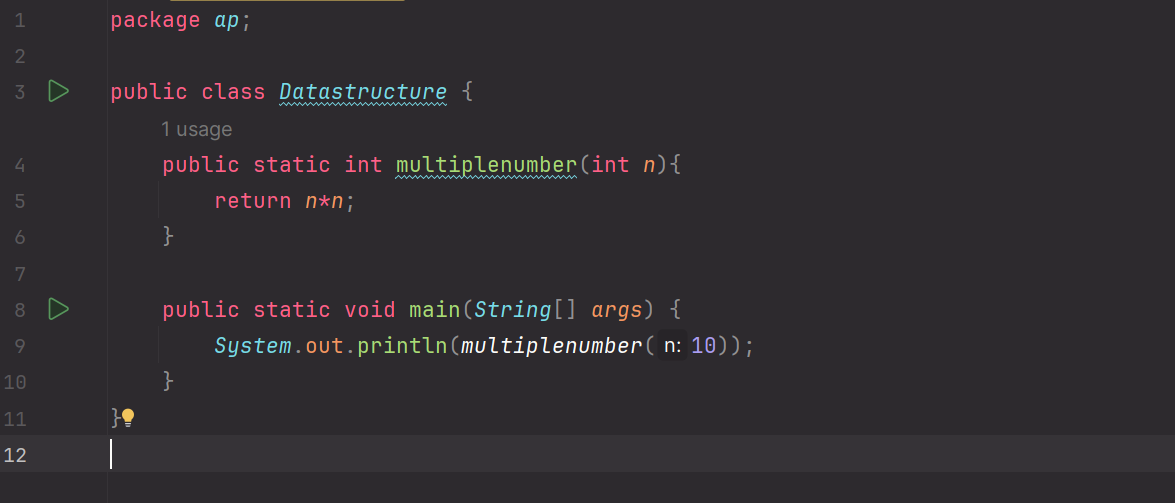
Big O is the language and metric we use to describe the efficiency of algorithms. For example, if you want to send a file to your friend and the size is small you send it easily but what if the size is big? So, if you want to give it to your friend physically id does not matter how big the file is, in computer science this is called time complexity. It is a way of showing how the runtime of the function increases as the size of the input increases.

There are three signs to show this complexity:

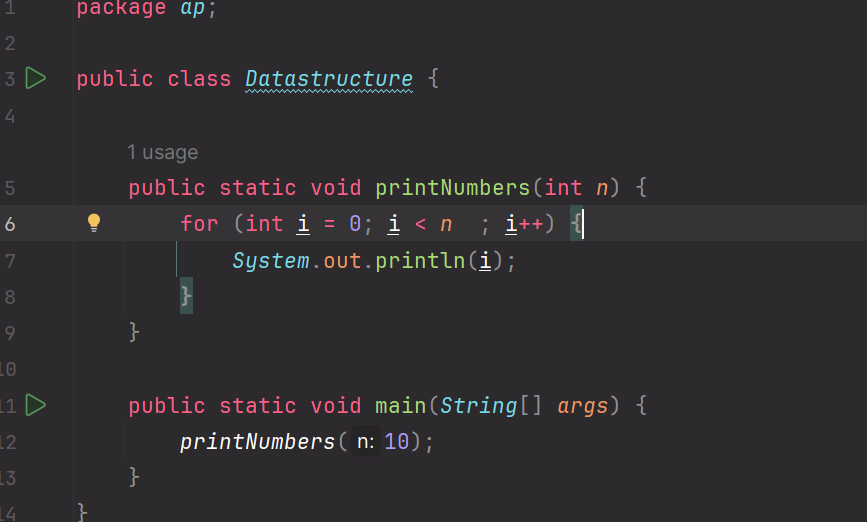
1. Omega (best case): it is going to be at least more than the best case.
2. Tita (average case): bounds of the worst and the best case.
3. O Micron (worst case): is going to be less or equal to the worst case.



O(1) or order of one or constant time complexity: for any given input the execution time won’t change. It will remain constant. For example here does not matter what n is and the complexity is O (1):



O(n) or linear time complexity, which grows to the size of input data.

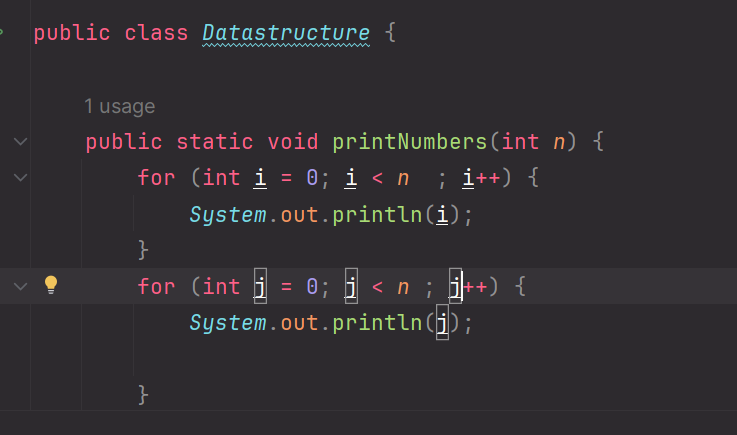


This is going to be O(n) , the search continues how big the n is.

Why we need to drop constant in O(n)?

For example, for the constant O(2n) why we say O(n)?

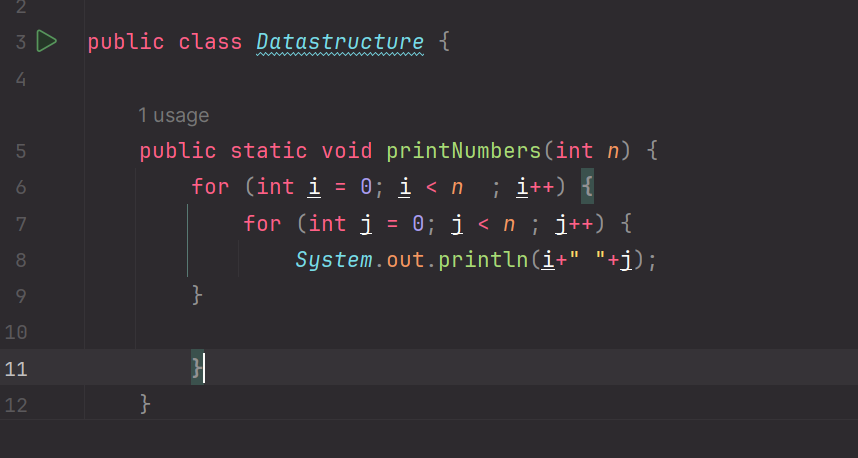
Look at this code:



Each has O(n) + O(n) = O(2n) and it does not matter what the constant is we say the complexity is O(n). why?

Because O(n) only defines the rate of complexity, and each computer has different architecture we ignore the constant.

O(n^2):



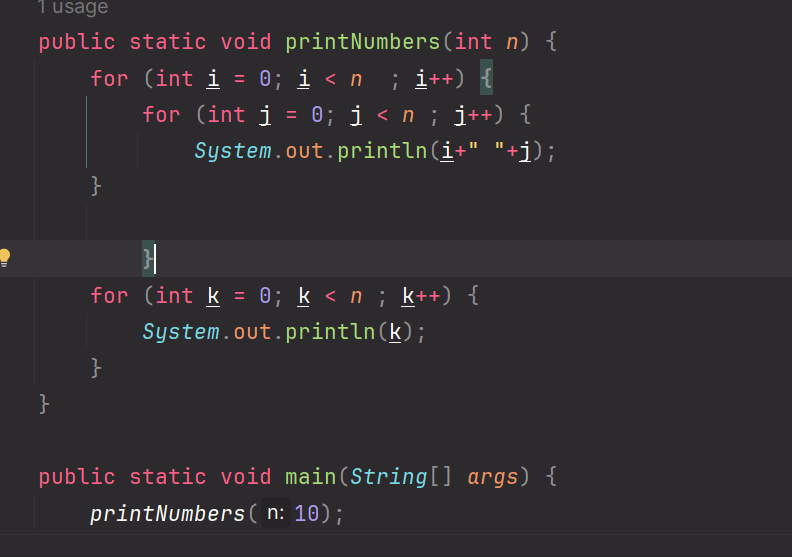
Here the number of items that we have passed is n and the number of operations is n\*n.

Dropping non dominant term:

Because the number is very small and there is no effect.

O(n^2+n) for example if n= 10

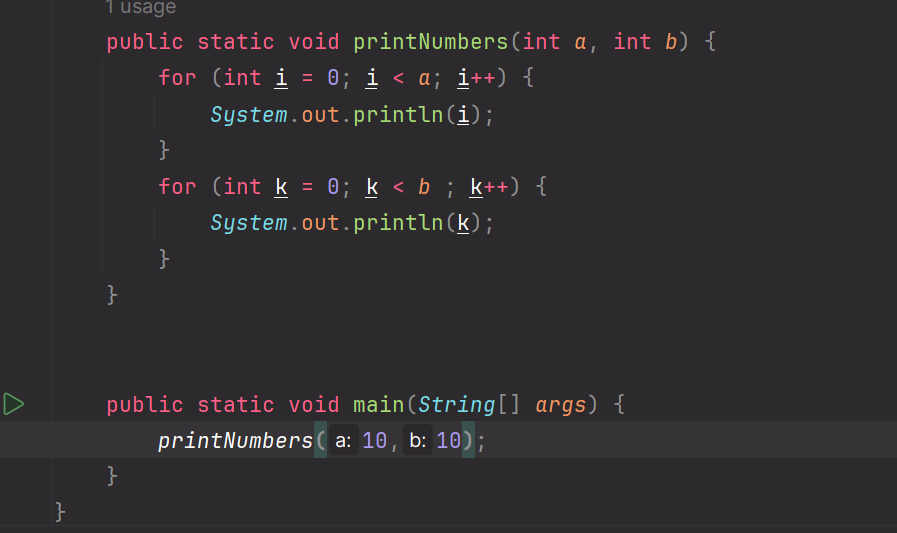
O(100000+100), 100 is a really small number. For simplify we say O(n)



Log(n): it is efficient from O(n).

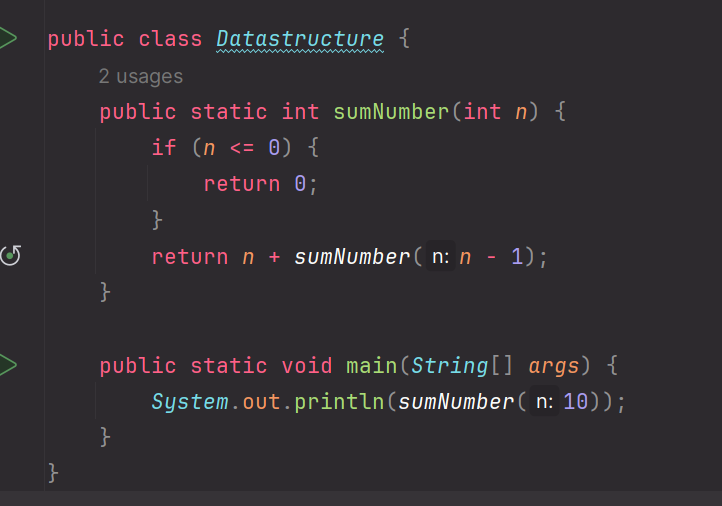
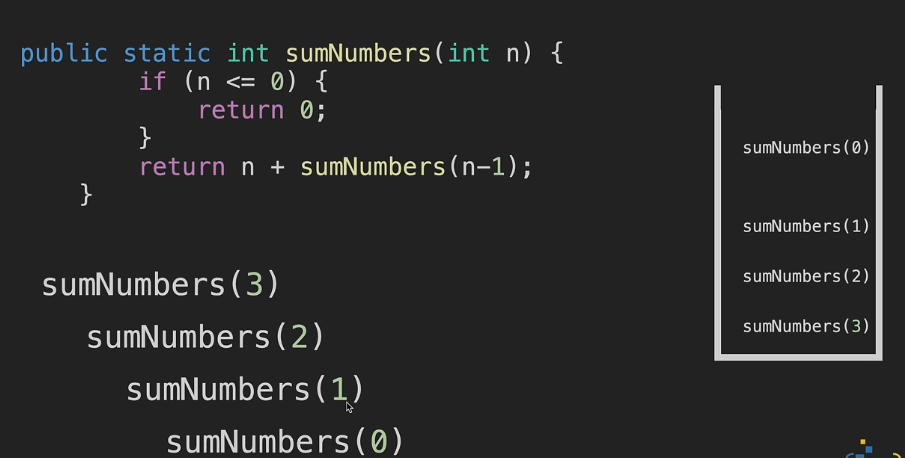
Different Terms for input:

Here the complexity is O(A\*B)



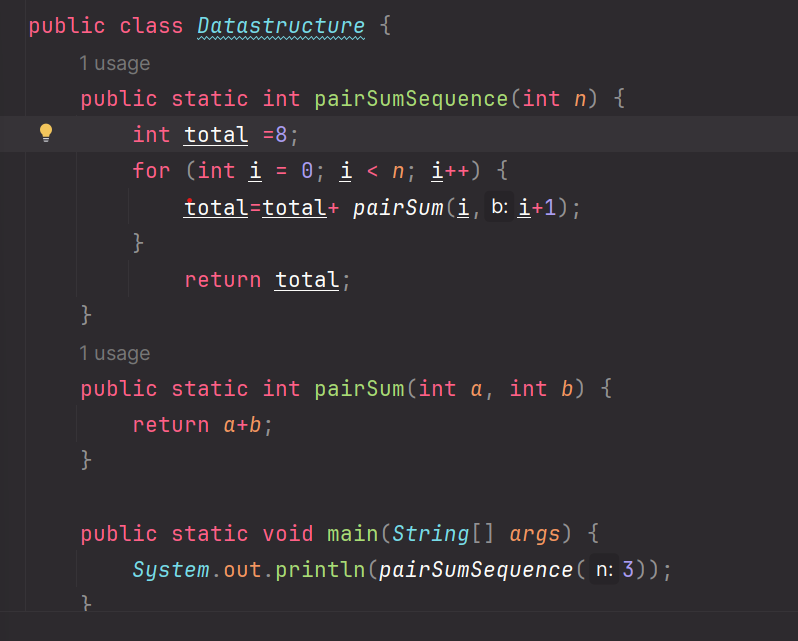
Space Complexity: how much memory we need for each algo:

Here this code works like this:



There is a website by waterloo university(Java visualizer) for executing java code and you can see the processing of code step by step.

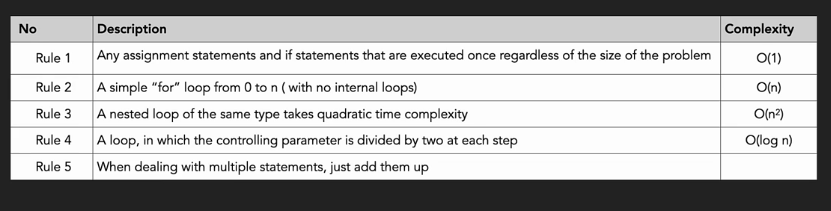
And space complexity to this code is O(n) space complexity.

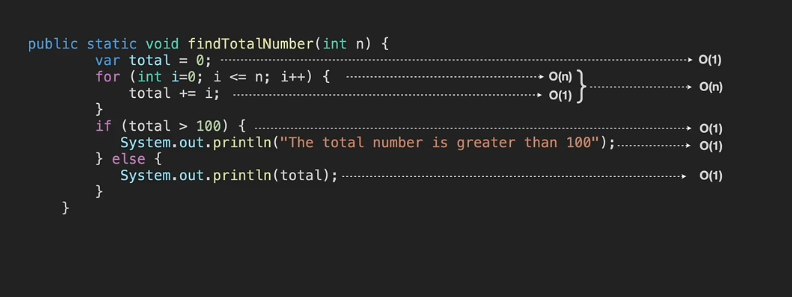


This code it is always adding and removing in the memory it is not keeping the numbers in the memory, here the space complexity is O(1).

How to measure the code using Big O?

There are 5 Rules:





Array:

All elements locate next to each other, and the size is fixed and we cannot change the size, array can store data of specific type. Each element has a uique address. And indexes start with 0, and the size of the array is predefined and not changeable.

In commuter science an array is a data structure considering of a collection of elements each identified by at least one array index or key. an array is stored such that the position of each element can be computed from its index by a mathematical formula.

Types of Arrays:

1. One Dimensional array
2. Linear array: with a bunch of values having been declared with a single index. Has a single row and multiple columns. For accessing the element over the array first write the array name then its index: a(0)

For example, it returns 4.

1. Two Dimensional Arra

There are multiple rows and columns. It consists of 1D arrays.

We have index for columns and index for rows. When we want to access an element, we need to put array name then index of row the index of column. a (2)(3)

1. 3D Array:

It is a multiple 2D arrays, and we have a depth here like a cube and each face of the cube consists of 2D arrays. Name, depth index, row index and column index A(0)(2)(3)

Arrays in memory:

In 1D array it gets any collection of RAM, no difference where it locates but not separately it needs to allocate the whole array which elements are next to each other.

In 2D array it represents in the memory as 1D array, sequentially next to each other. All will locate in one row.

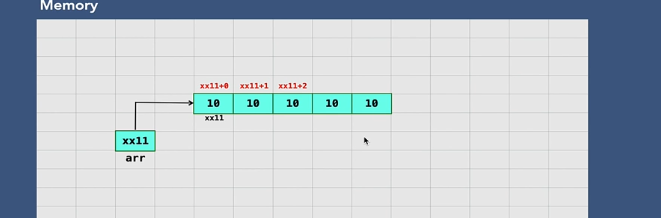
In 3D array, again it stores in memory next to each other.

In Java there are 3 steps to create an array:

1. Declare- create a reference to array.
2. Instantiation of an array
3. Initialization – assigns value to cells in array.

What happens to the memory?

System never allocates a memory for an array, it creates a reference variable, next step is we are instantiating an array, so then the compiler allocated space in the memory. After that we initialization an array.

How can we insert a new value to the array?



How can we access an element in array list?

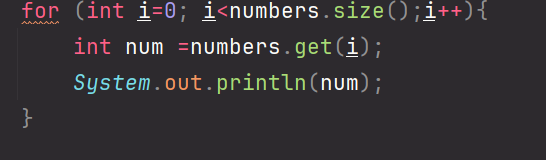
We use get method we return the value of an element.

The time complexity of it is O(1) and space complexity is O(n) too.

What is Traversal of ArrayList?

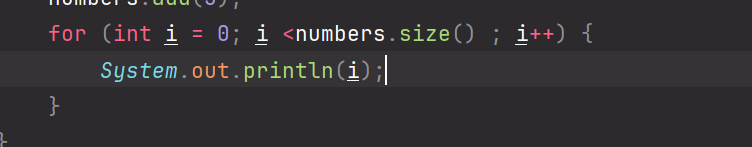
It involves visiting each element in the list, typically to perform an operation on each element, in total it means we are starting first element and visiting each element one by one until we reach the last element. There are several ways to traverse an arraylist in Java, including using a for loop, a for each loop and iterator.

For Loop: time complexity :O(n) and space is O(1)



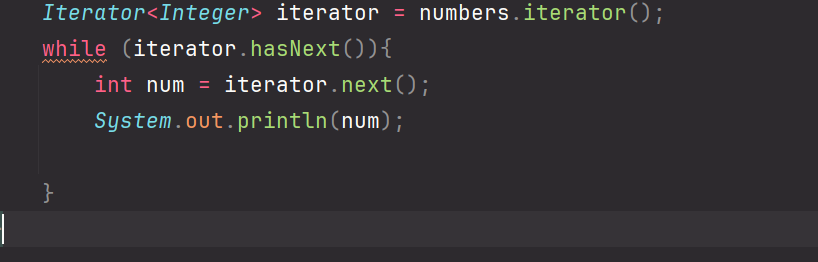
For each loop : traversing a collection in an array list:

time complexity :O(n) and space is O(1)



An Iterator: it uses next to check the next element in the list,

time complexity :O(n) and space is O (1)



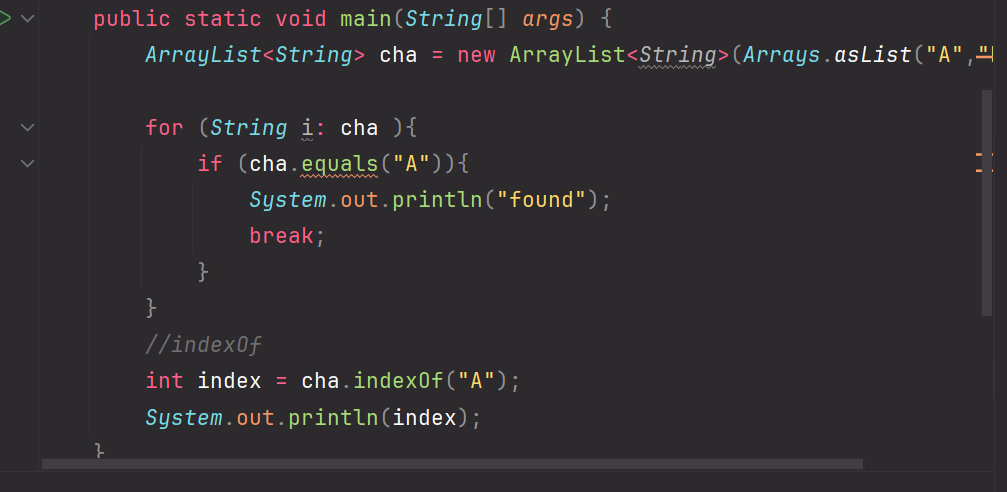
Search an element in ArrayList?

Checking it exists or not?

It checks each element one by on, there are several ways to search for an element

In an array list, such as using a for loop, a for each loop , index of method:

time complexity :O(n) and space is O (1)



How can we delete an element in ArrayList;

One way is removing with the parameter of index.

The second one is also remove with the parameter of object.

time complexity :O(n) and space is O (1)



Time and Space Complexity:

