**Data Structure**

Data Structure has two type:

* Non-Linear: graph, tree (), jungle
* Linear: List, Array, ArrayList, Stack, Queue, Deque

Big O() shows the worst case in data structure

1. Linear:

* List: là một data structure liên tiếp, nếu muốn access member bất kì (VD: index 3), nó phải access one-by-one in order which start from index 0, index 1, index 2 trước. All members of List can have **different** data types from each other
* Array: là một data structure cho phép random access members, nó có thể access thẳng vào index 3 mà không cần access vào index 0, 1, 2. All members of Array must have the **same** data type
* The best case scenario in query is the number you want to find at index 0 => o(1). The worst case in query is the value has been found at the last index which is Big O(n) (Big O means the worst scenario). The average case is in the middle => o(n/2)

1. Exercise:

* Ex1: Assuming an array is given to you which has ‘m’ dimension. Write a code to find a specific value in this array

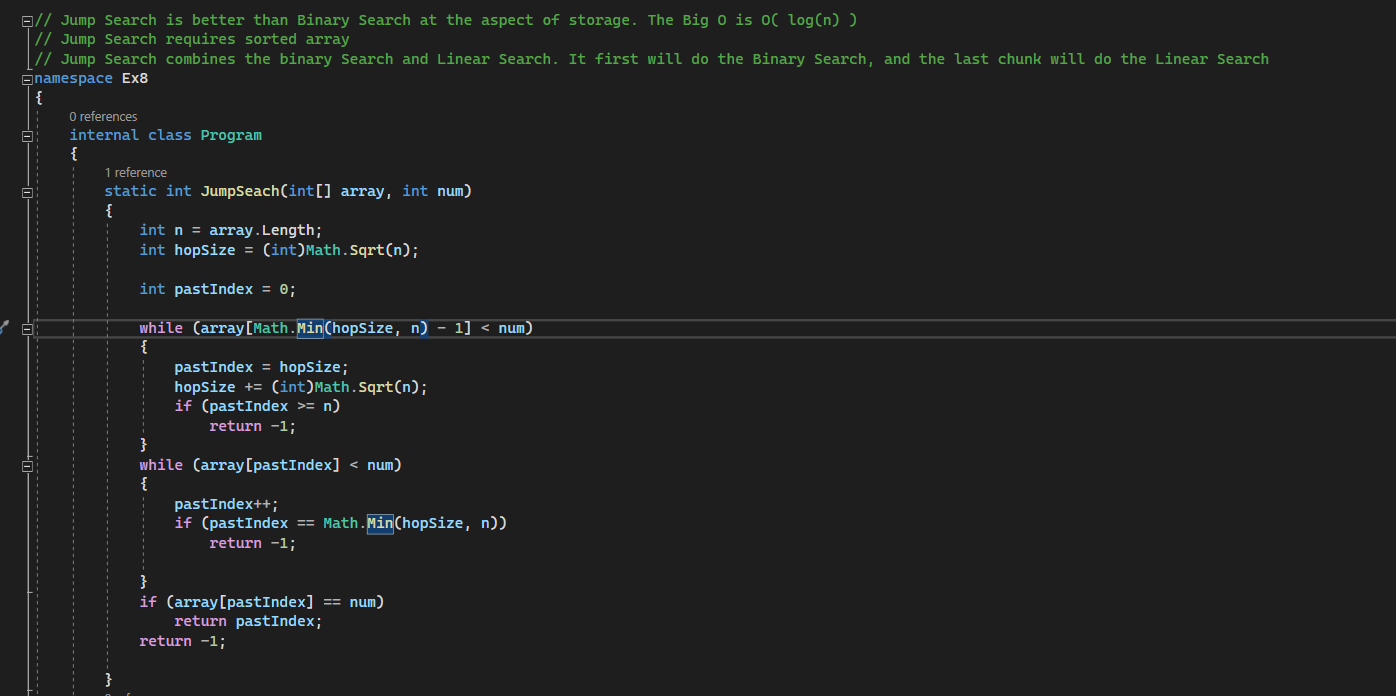
Text

Description automatically generated

Text

Description automatically generated

* Ex2: Write a code to find the maximum value in a array and analyze Big O
* Ex3: Jump Search to search a number in array



Text

Description automatically generated

1. Stack:

* A stack is a linear data structure that the element which gets in first will be the last element fetch out (LIFO: Last I)
* Recursion is a method call itself
* Binary Search: has two ways to implement which are loop based and recursion-based
* Recursion requires a stack
* Exercise of Recursion:
  + Factorial Number:

Text

Description automatically generated

Text

Description automatically generated

* + Fibonacci Serial number:

Text

Description automatically generated

A white board with writing on it

Description automatically generated

Big O(n) of Fibonacci

A picture containing schematic

Description automatically generated

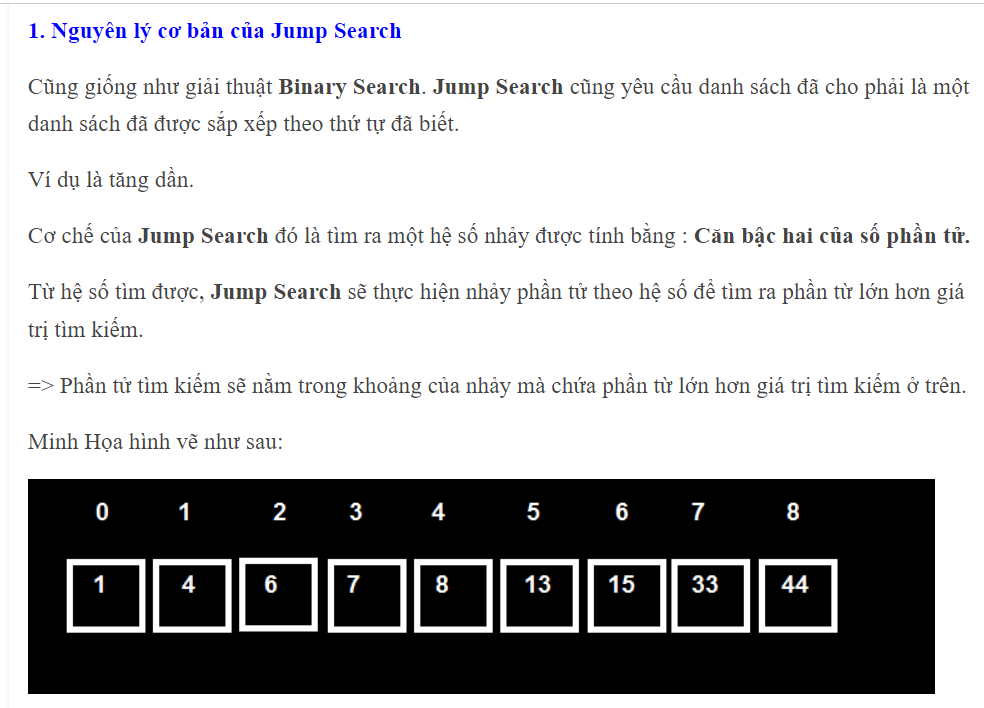
1. Search Algorithm:

* Linear search: O(n): Search
* Binary search: O(log n) (log base is 2)

Text

Description automatically generated

* Jump search: O(log n) (log base is 2) it will divide into multiple chunk then compare like the way of Binary search, the last chunk will use the the Linear Search. Jump Searrch requires sorted array. Jump Search is better than Binary Search at the aspect of storage
  + Nguyên lý cơ bản và ví dụ



Graphical user interface, text, application

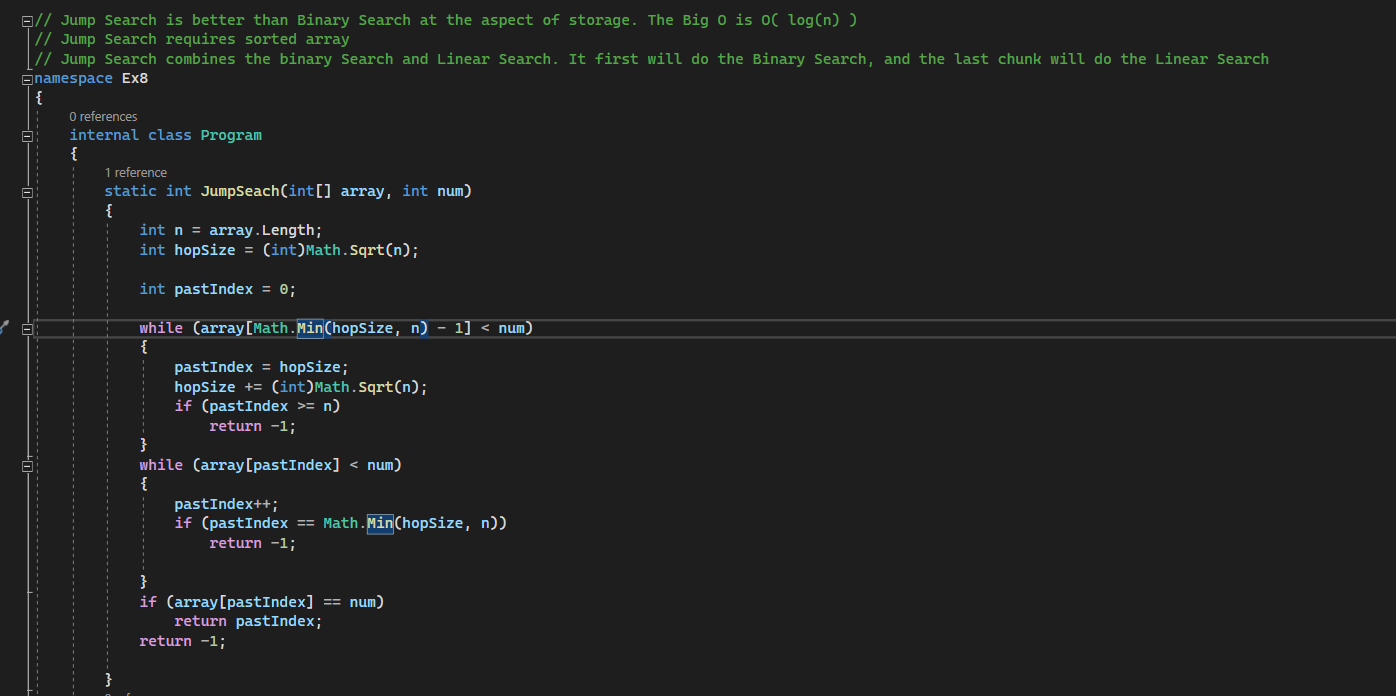
Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, text, application

Description automatically generated



Text

Description automatically generated

1. Sort Algorithm:

* Insertion sort: O(n^2): Insertion sort will compare index 0 and index 1 first then if value of index 0 > value of index 1, it will swap both of them. Then compare the value of index 1 with value of index 2, if value of index 1 > value of index 2, it will swap both of them. Then check again if value of index 1 now (which is the value of index 2) is it higher than value of index 0 if yes, swap them too. Then keep continue
  + The code illustrates below

Text

Description automatically generated

* + Nguyên lý cơ bản và ví dụ:

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

* Selection sort: O(n^2) sSelection sort will take the value of index 0 as a minimum value then it will compare with the rest element of array which starts from index 1 (I will call the rest element of array is Rest\_array), and the minimum value of the index in the Rest\_Array which is smaller than the value of index 0 will swap with the value of index 0. Then it will start with the index 1 as a minimum value, and compare it with the rest element of array which starts from index 2, and the minimum value of the index in the Rest\_Array which is smaller than the value of index 1 will swap with the value of index 1. Then keep continue.
  + The code illustrates below

Text

Description automatically generated

* Quick sort: Big O(n \* log(n) base 2)

Text, letter

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

* Merge Sort: Big O(n\*log n base 2):

<https://viettuts.vn/cau-truc-du-lieu-va-giai-thuat/giai-thuat-sap-xep-tron-merge-sort>

MergeSort is a method used to sort an array. MergeSort first will divide the array into two half, and each half of two half will divide into another two half, and continue until the subarray contains only 1 element. Then it will compare each two elements, and merge them in ascending order. Now the two element becomes 1 sub array (subarray1) which has 2 elements, and it will compare with sub array (subarray2) that has two element and next to that subarray1, after sort the value merge them again and we have another sub array that contains 4 elements, and continue compare with another sub array that has 4 elements and next to that sub array

Text, letter

Description automatically generated

Diagram

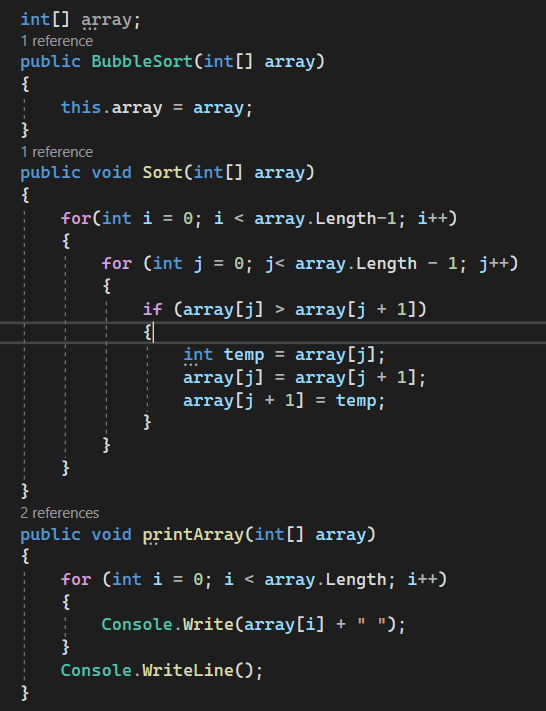
Description automatically generated

* Tim Sort: Big O(n\*log n base 2) Combine Merge and Insertion, but there is no swap between the element just which one is minimum will attach the left side and which one is maximum will attach the right side. TimSort is

Text

Description automatically generated

* Bubble Sort: Big O(n^2) Every Step takes two items index 0 and index 1. If they are in order, don’t do anything otherwise swap. They will take the next two items which are index 1 and index2, and do the same. However, it won’t compare value index 0 and value index 1 (is knows as value of index 2 before if the value of index 1 swap with value of index 2)



* Heap Sort: Big O (): Heap always relates to a graph data structure, and a graph has multiple vertexes and the connections (edges) between them. An array will be heapified into a small graph to sort it. Vertex giống như là nốt trong binary tree

1. The best case to worst case

* O(1) < O(logn)< O(n) < O(n\*logn) < O(n^2) < O(n^3)

1. Question:

* Members of List in C# need the same data type