

BUBBLE SORTING USING ARRAYS

Bubble sorting sorts the numbers of an array by repeatedly swapping the adjacent elements until an order(ascending or descending) is attained in their arrangement

Mechanism of bubble sorting:

i = 0	j	0	1	2	3	4	5	6	7
	0	5	3	1	9	8	2	4	7
	1	3	5	1	9	8	2	4	7
	2	3	1	5	9	8	2	4	7
	3	3	1	5	9	8	2	4	7
	4	3	1	5	8	9	2	4	7
	5	3	1	5	8	2	9	4	7
	6	3	1	5	8	2	4	9	7
i=1	0	3	1	5	8	2	4	7	9
	1	1	3	5	8	2	4	7	
	2	1	3	5	8	2	4	7	
	3	1	3	5	8	2	4	7	
	4	1	3	5	2	8	4	7	
	5	1	3	5	2	4	8	7	
i=2	0	1	3	5	2	4	7	8	
	1	1	3	5	2	4	7		
	2	1	3	5	2	4	7		
	3	1	3	2	5	4	7		
	4	1	3	2	4	5	7		
i=3	0	1	3	2	4	5	7		
	1	1	3	2	4	5			
	2	1	2	3	4	5			
	3	1	2	3	4	5			
i=4	0	1	2	3	4	5			
	1	1	2	3	4				
	2	1	2	3	4				
i=5	0	1	2	3	4				
	1	1	2	3					
i=6	0	1	2	3					
	1	1	2						

BUBBLE SORTING

CODE

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/*****
Author : G V S S SRI LASYA
Purpose : Bubble sorting using array
*****/
namespace Day9Presentation
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //arranging user inputs in ascending order using bubble sorting and array

            int intermediate = 0, size = 0;
            bool swap;

            //taking array size as input from user
            Console.WriteLine("\nEnter number of values to be sorted : ");
            size = Convert.ToInt32(Console.ReadLine());

            int[] numbers = new int[size];

            //taking in values of array elements as inputs
            for (int i = 0; i < size; i++)
            {
                Console.WriteLine($"Enter number {i + 1} : ");
                numbers[i] = Convert.ToInt32(Console.ReadLine());
            }

            //bubble sorting the numbers stored in array
            for (int i = 0; i < numbers.Length - 1; i++)
            {
                swap = false;
                for (int j = 0; j < numbers.Length - 1 - i; j++)
                {
                    if (numbers[j] > numbers[j + 1])
                    {
                        intermediate = numbers[j];
                        numbers[j] = numbers[j + 1];
                        numbers[j + 1] = intermediate;
                        swap = true;
                    }
                }

                if (swap == false)
                    break;
            }

            //printing out the sorted numbers
            Console.WriteLine("\n\nNumbers after bubble sorting : ");
            for (int i = 0; i < numbers.Length; i++)
                Console.Write($"{numbers[i]}");

            Console.ReadLine();
        }
    }
}
```

```
}
```

OUTPUT

```
Enter number of values to be sorted : 5
```

```
Enter number 1 : 3
```

```
Enter number 2 : 52
```

```
Enter number 3 : 12
```

```
Enter number 4 : 19
```

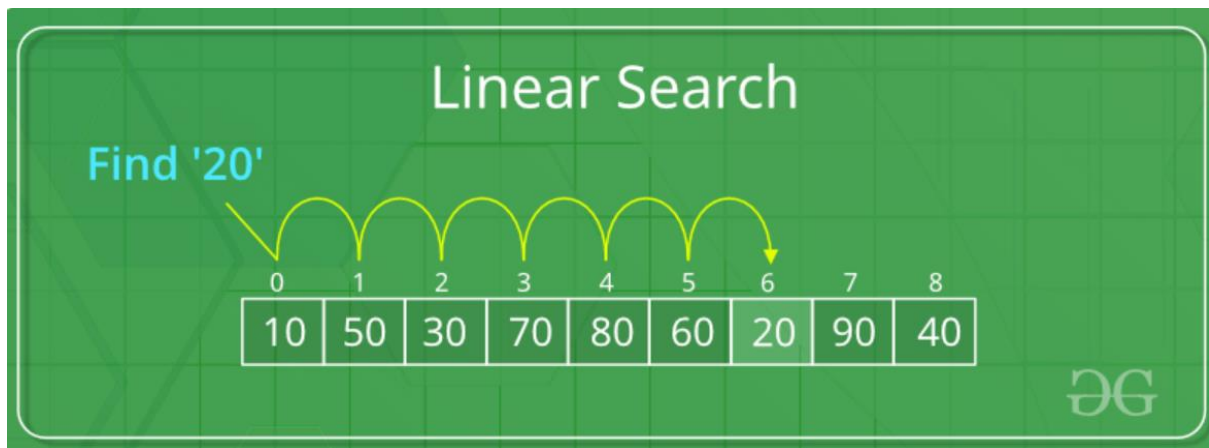
```
Enter number 5 : 46
```

```
Numbers after bubble sorting : 3      12      19      46      52
```

LINEAR SEARCH USING ARRAYS

It lets us search for a number from a list of numbers of an array by comparing the given number with numbers of the array one by one till either the target number is found or all the numbers of the array are checked for match and failed

Mechanism of linear search of arrays:



LINEAR SEARCH

CODE

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/*****
Array : G V S S SRI LASYA
Purpose : Linear Search of an array
*****/

namespace Day9ppt2
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int value,i;

            //Create an array and initialize it
            int[] numbers = new int[] { 3, 1, 9, 8, 7, 12, 56, 23, 89 };

            //Read the target value to search
            Console.Write("\nEnter a number to search: ");
            value = Convert.ToInt32(Console.ReadLine());

            for ( i = 0; i < numbers.Length; i++)
            {
                if (value == numbers[i])
                {
                    Console.Write($"Number is found at position {i + 1}");
                    break;
                }
            }

            if (i == numbers.Length)
                Console.Write("\nValue not found");

            Console.ReadLine();
        }
    }
}
```

OUTPUT

```
Enter a number to search: 12
Number is found at position 6
```


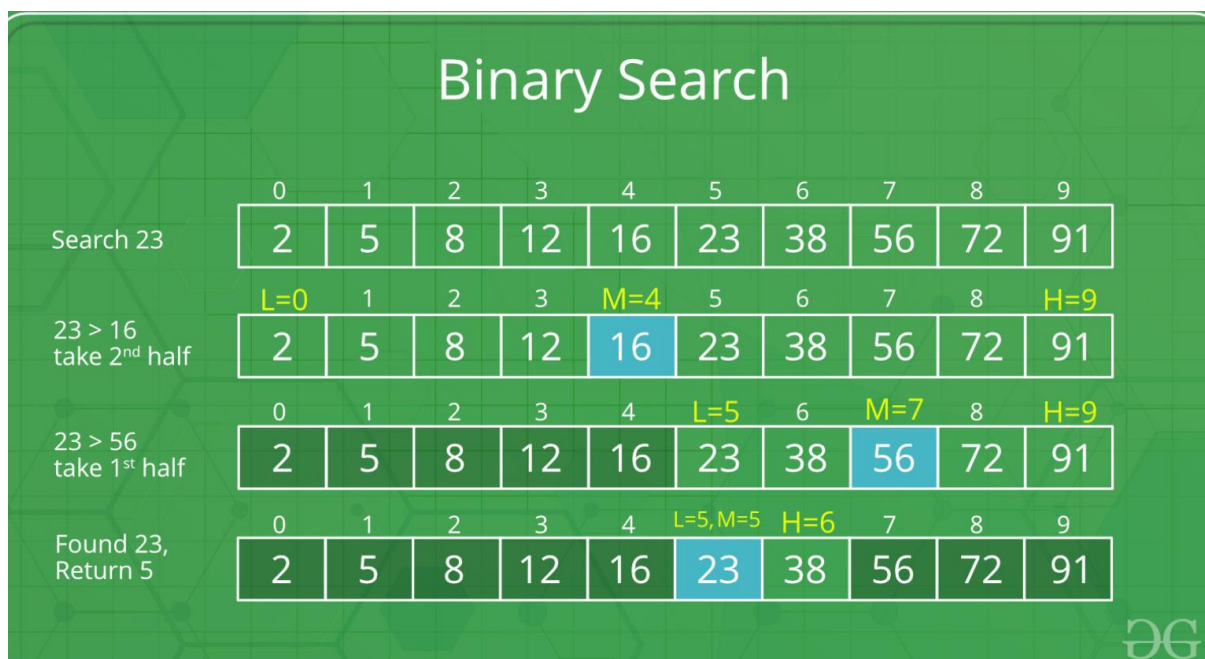
BINARY SEARCH USING ARRAYS

Binary search works on a sorted array. The value is compared with the middle element of the array. If equality is not found, then if the value is greater than middle element, the first half of array is eliminated and only the second half is searched. If the value is smaller than middle element, the second half of array is eliminated and only the first half is searched for the value. Thus reduces time and increases efficiency of search. If equality is found then the middle element index is printed.

Mechanism of binary search of arrays:

Binary Search

	0	1	2	3	4	5	6	7	8	9
Search 23	2	5	8	12	16	23	38	56	72	91
	L=0	1	2	3	M=4	5	6	7	8	H=9
23 > 16 take 2 nd half	2	5	8	12	16	23	38	56	72	91
	0	1	2	3	4	L=5	6	M=7	8	H=9
23 > 56 take 1 st half	2	5	8	12	16	23	38	56	72	91
	0	1	2	3	4	L=5, M=5	H=6	7	8	9
Found 23, Return 5	2	5	8	12	16	23	38	56	72	91



BINARY SEARCH

CODE

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

/*****
Author : G V S S SRI LASYA
Purpose : Binary search on arrays
*****/

namespace Day9ppt3
{
    internal class Program
    {
        public static int BinarySearch(int[] numbers, int input)
        {
            int startIndex=0, endIndex=numbers.Length-1;

            while(startIndex <= endIndex)
            {
                int middleIndex = startIndex + ((endIndex-startIndex)/2);

                if (numbers[middleIndex] == input)
                    return middleIndex;
                else if (numbers[middleIndex] < input)
                    startIndex = middleIndex + 1;
                else
                    endIndex = middleIndex - 1;
            }

            return -1;
        }

        static void Main(string[] args)
        {
            int size, input, result;

            Console.Write("\nEnter the size of array : ");
            size = Convert.ToInt32(Console.ReadLine());

            int[] numbers = new int[size];

            for(int i =0; i<size; i++)
            {
                Console.Write($"Enter number{i+1} : ");
                numbers[i] = Convert.ToInt32(Console.ReadLine());
            }

            Console.Write("\nEnter the number to be searched for in array : ");
            input = Convert.ToInt32(Console.ReadLine());

            result = BinarySearch(numbers, input);

            if (result == -1)
                Console.Write("\nValue not found");
            else
                Console.Write($"Value found at {result} index");

            Console.ReadLine();
        }
    }
}
```

OUTPUT

```
Enter the size of array : 8
Enter number1 : 1
Enter number2 : 2
Enter number3 : 3
Enter number4 : 4
Enter number5 : 5
Enter number6 : 6
Enter number7 : 7
Enter number8 : 8

Enter the number to be searched for in array : 5

Value found at 4 index
```