# Algorithm Design & Analysis Lab Assignment 01

**Submitted By:**

Name: Sharif Md. Mehedi Hasan

ID: 221-35-879

Sec: D

Course Code: SE215

**Submitted To:**

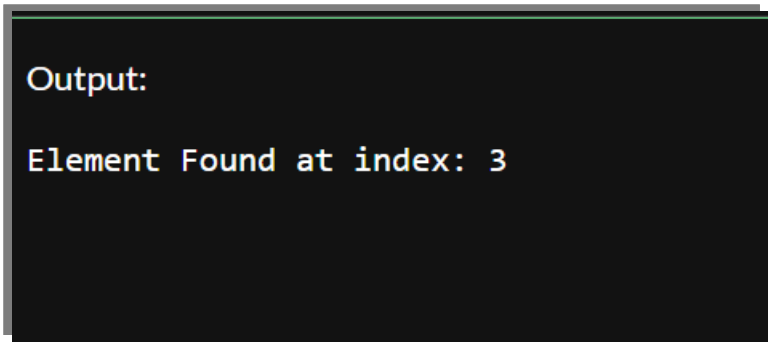
Md. Masum Billah

Lecturer in SWE

Daffodil International University

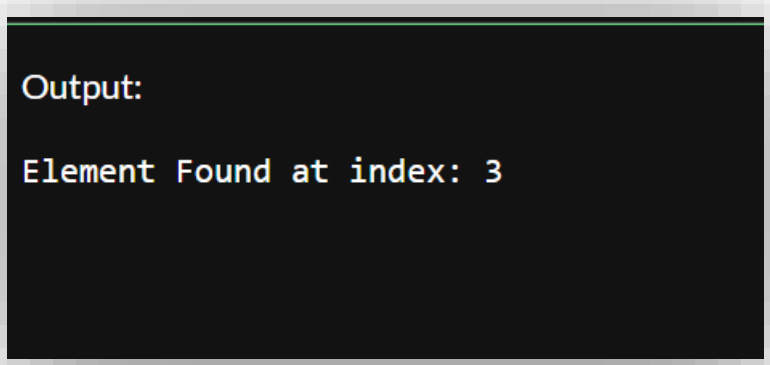

## Linear Search:

```
#include <iostream>
using namespace std;
int search(int array[], int n, int key)
{
	// Search Sequentially Every In Index
	for (int i = 0; i < n; i++)
		if (array[i] == key)
			return i;
	return -1;
}
int main()
{
	int array[] = { 2, 4, 0, 1, 9 };
	int key = 1;
	int n = sizeof(array) / sizeof(array[0]);

	int result = search(array, n, key);

	(result == -1) ? cout << "Element Not Found": cout << "Element Found at index: " << result;
}
```



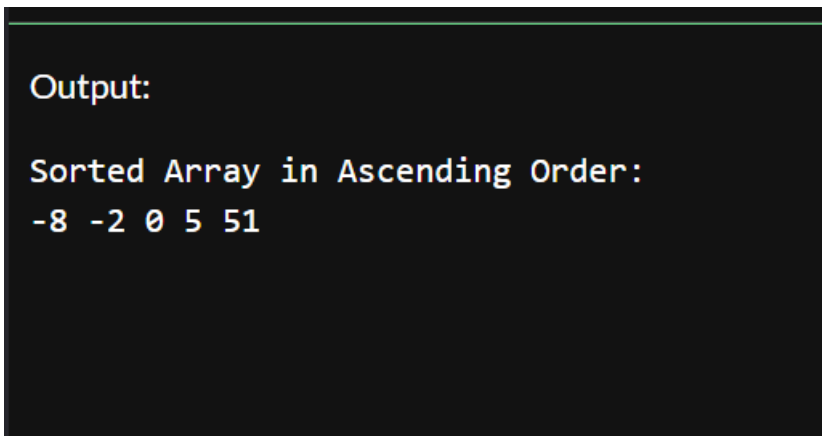

## Binary Search Recursive Method:

```
#include <bits/stdc++.h>
using namespace std;
int binarySearch(int arr[], int l, int r, int x)
{
	if (r >= l)
	{
		int mid = l + (r - l) / 2;
		if (arr[mid] == x)
			return mid;
		if (arr[mid] > x)
			return binarySearch(arr, l, mid - 1, x); //If Recursive Call
		return binarySearch(arr, mid + 1, r, x); //Else Part Recursive Call

	}
	return -1;
}
int main()
{
	int arr[] = { 2, 3, 4, 10, 40 };
	int x = 10;
	int n = sizeof(arr) / sizeof(arr[0]);
	int result = binarySearch(arr, 0, n - 1, x);
	(result == -1) ?
	cout << "Element Not Found": cout << "Element Found at Index " << result;
}
```



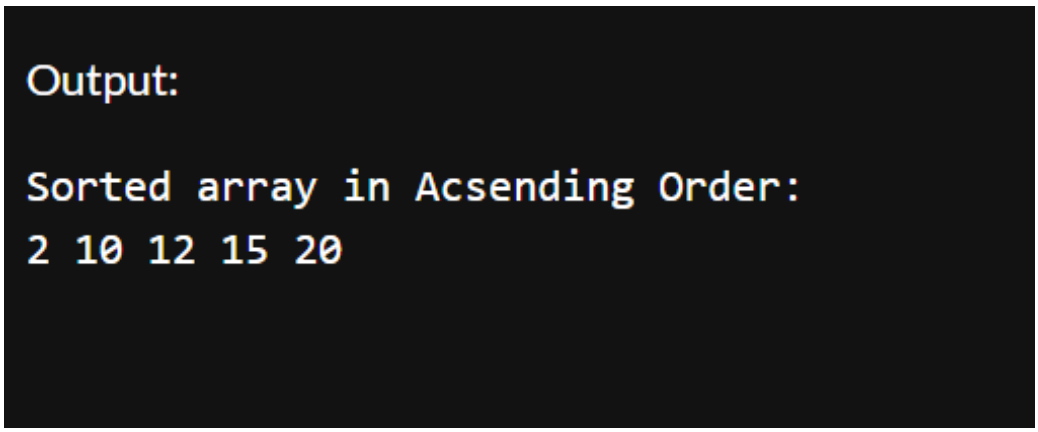
## Bubble Sort Optimized:

```
#include <iostream>
using namespace std;
void bubbleSort(int array[], int size)
{
	for (int j = 0; j < (size - 1); ++j)
	{
		int flag = 0;
		for (int i = 0; i < (size - j - 1); ++i)
		{
			if (array[i] > array[i + 1])
			{
				int temp = array[i];
				array[i] = array[i + 1];
				array[i + 1] = temp;
				flag = 1;
			}
		}

		if (flag == 0)
			break;
	}
}

int main()
{
	int data[] = { -2, 5, 0, 51, -8 };
	int size = sizeof(data) / sizeof(data[0]);
	bubbleSort(data, size);
	cout << "Sorted Array in Ascending Order:\n";
	for (int i = 0; i < size; i++) cout << data[i] << " ";
}
```



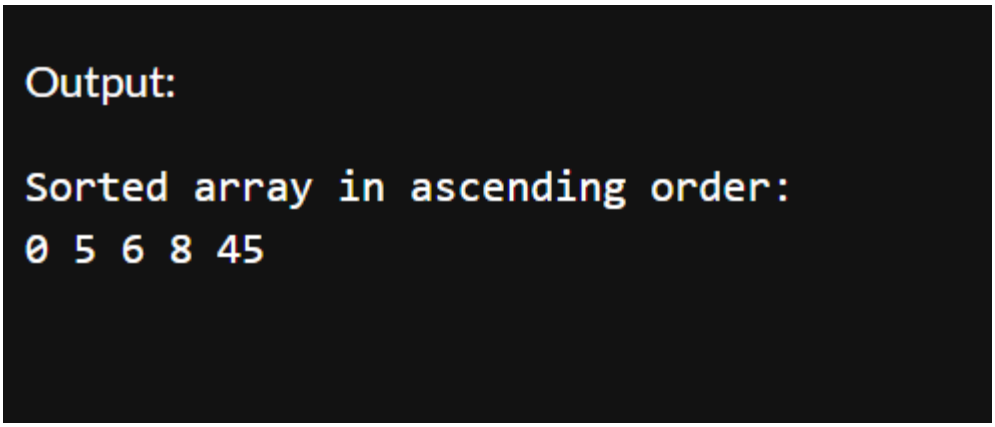
## Selection Sort:

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;

int main()
{
	int data[] = { 20, 12, 10, 15, 2 };
	int size = sizeof(data) / sizeof(data[0]);
	for (int i = 0; i < size - 1; i++)
	{
		int min = i;
		for (int j = i + 1; j < size; j++)
		{
			if (data[j] < data[min])
				min = j;
		}

		swap(data[min], data[i]);
	}

	cout << "Sorted array in Acsending Order:\n";
	for (int i = 0; i < size; i++) cout << data[i] << " ";
}
```



## Insertion Sort:

```
#include <iostream>
using namespace std;
void insertionSort(int array[], int size)
{
	for (int i = 1; i < size; i++)
	{
		int key = array[i];
		int j = i - 1;
		while (key < array[j] && j >= 0)
		{
			array[j + 1] = array[j];
			--j;
		}

		array[j + 1] = key;
	}
}

int main()
{
	int data[] = { 8, 5, 6, 45, 0 };
	int size = sizeof(data) / sizeof(data[0]);
	insertionSort(data, size);
	cout << "Sorted array in ascending order:\n";
	for (int i = 0; i < size; i++) cout << data[i] << " ";
}
```



## Queue:

```
#include <iostream>
using namespace std;

#define size 5
int Queue[size], front = -1, rear = -1;
void push(int val)
{
	if (rear == size - 1) cout << "Queue Overflow\n";
	else
	{
		if (front == -1) front = 0;
		Queue[++rear] = val;
	}
}

void pop()
{
	if (front == -1 || front > rear) cout << "Queue Is Underflow\n";
	else front++;
}

void display()
{
	if (front == -1)
	{
		cout << "Queue Is Empty\n";
		return;
	}

	cout << "Queue Elements Are : ";
	for (int it = front; it <= rear; it++)
		cout << Queue[it] << " ";
	cout << endl;
}

int main()
{
	display();
	pop();
	push(2);
	push(88);
	pop();
	display();
}
```

Output:

```
Queue Is Empty
Queue Is Underflow
Queue Elements Are : 88
```

## Stack:

```
#include <bits/stdc++.h>
using namespace std;

#define size 5
int top = -1, Stack[size];
void push(int x)
{
	if (top == size - 1) cout << "Stack Overflow\n";
	else Stack[++top] = x;
}

void pop()
{
	if (top == -1) cout << "Stack Underflow\n";
	else top--;
}

void display()
{
	int it = top + 1;
	if (!it)
	{
		cout << "Stack Empty\n";
		return;
	}

	cout << "Stack Elements: ";
	while (it-- > 0) cout << Stack[it] << " ";
	cout << endl;
}

int main()
{
	display();
	pop();
	push(62);
	push(94);
	push(53);
	push(19);
	push(24);
	display();
	pop();
	display();
}
```

Output:

```
Stack Is Empty
Stack Underflow
Stack Elements Are : 24 19 53 94 62
Stack Elements Are : 19 53 94 62
```