

CS263: Design and Analysis of Algorithm Laboratory

Name : Hritik Kumar
Roll No. : 202051088

Task – 1

○ Linear Search- Algorithm

```
void LinearSearch(int arr[], int n, int k)
{
    int flag = 0;
    for(int i=0; i<n; i++){
        if(arr[i] == k){
            cout<<"Found!!!"<<endl;
            flag = 1;
            break;
        }
    }

    if(flag == 0)
        cout<<"Not Found!!!"<<endl;
}
```

Time complexity: -

Initialization of **x** in the loop----- 1 time

Comparison of **x** in the loop ----- n + 1 times

Increment of **x** in the loop ----- n times

Conditional comparison ----- n times

Return if condition is true ----- n times

Final return ----- 1 time

Total steps $4n + 3$ so **Time complexity = $O(n)$**

○ Binary Search- Algorithm:-

```
void BinarySearch(int arr[], int l, int r, int x)
{
    int flag = 0;
    if (r >= l) {
        int mid = l + (r - l) / 2;
        if (arr[mid] == x){
            flag = 1;
            cout<<"Found!!!"<<endl;
            return;
        }

        if (arr[mid] > x)
            return BinarySearch(arr, l, mid - 1, x);

        return BinarySearch(arr, mid + 1, r, x);
    }

    if(flag ==0)
        cout<<"Not Found!!!"<<endl;
}
```

Time complexity: -

Initialisation of left ----- 1 time
Initialisation of right ----- 1 time
Comparison in loop ----- $\log(n) + 1$ times
Initialisation of mid ----- $\log(n)$ times
Conditional statement 1 comparison ----- $\log(n)$ times
Conditional statement 2 comparison ----- $\log(n)$ times
Conditional statement 3 comparison ----- $\log(n)$ times
Final return ----- 1 time
Total steps $5\log(n) + 4$ so **Time complexity = $O(\log(n))$**

○ Bubble sort- Algorithm

```
void BubbleSort(int arr[], int n)
{
    for(int i=1; i<n; i++){
        for(int j=0; j<n-i; j++){
            if(arr[j]>arr[j+1]){
                arr[j] = arr[j]^arr[j+1];
                arr[j+1] = arr[j]^arr[j+1];
                arr[j] = arr[j]^arr[j+1];
            }
        }
    }
}
```

Time complexity: -

Initialisation of i in the for loop ----- 1 time

Comparison of i in the for loop ----- n times

Increment of i in the for loop ----- n-1 times

Initialization of j in the for loop -----n time

Comparison of j in the for loop ----- n^2 times

Increment of j in the for loop ----- $n^2 - n$ times

Comparisons of arr[j] and arr[j+1] ----- n^2 times

Swapping arr[j] and arr[j+1] ----- $3*n^2$ times

Final return ----- 1 time

Total steps $6n^2 + 2n + 1$ so **Time complexity = $O(n^2)$**

○ Selection sort- Algorithm

```
void SelectionSort(int arr[], int n)
{
    for(int i=0; i<n-1; i++){
        int mini = i;
        for(int j=i+1; j<n; j++){
            if(arr[mini]>arr[j]){
                mini = j;
            }
        }
        if(mini == i)
            continue;
        arr[i] = arr[i]^arr[mini];
        arr[mini] = arr[i]^arr[mini];
        arr[i] = arr[i]^arr[mini];
    }
}
```

Time complexity: -

Initialization of i in the for loop ----- 1 time

Comparison of i in the for loop ----- n + 1 times

Increment of i in the for loop ----- n times

Initialization of mini ----- n times

Initialization of j in the for loop ----- n time

Comparison of j in the for loop ----- n^2 times

Increment of j in the for loop ----- $n^2 - n$ times

Comparisons of arr[j] and arr[min] ----- n^2 times

Assignment of min = j ----- n^2 times

Swapping arr[i] and arr[min] ----- 3n times

Final return ----- 1 time

Total steps $4n^2 + 6n + 3$ so **Time complexity = $O(n^2)$**

○ Insertion sort- **Algorithm**

```
void InsertionSort(int arr[], int n)
{
    int temp;
    int j;
    for(int i=1; i<n; i++){
        temp = arr[i];
        for(j= i-1; j>=0 && arr[j]>temp; j--){
            arr[j+1] = arr[j];
        }
        arr[j+1] = temp;
    }
}
```

Time complexity: -

Initialization of i in the for loop ----- 1 time

Comparison of i in the for loop ----- n times

Increment of i in the for loop ----- n-1 times

Initialization of temp ----- n times

Initialization of h ----- n times

Comparisons in the while loop ----- $n^2 + n$ times

Assignment of $arr[h+1] = arr[h]$ ----- n^2 times

Decrement of h ----- n^2 times

Assignment of $arr[h+1] = temp$ ----- n times

Final return ----- 1 time

Total steps $3n^2 + 6n + 1$ so **Time complexity = $O(n^2)$**

	Best Case	Average Case	Worst Case
Linear Search	$O(1)$	$O(n)$	$O(n)$
Binary Search	$O(1)$	$O(\log(n))$	$O(\log(n))$
Bubble Sort	$O(n)$	$O(n^2)$	$O(n^2)$
Selection Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$
Insertion Sort	$O(n)$	$O(n^2)$	$O(n^2)$

Task -2

Linear Search:-

The screenshot shows the Visual Studio Code interface with the following details:

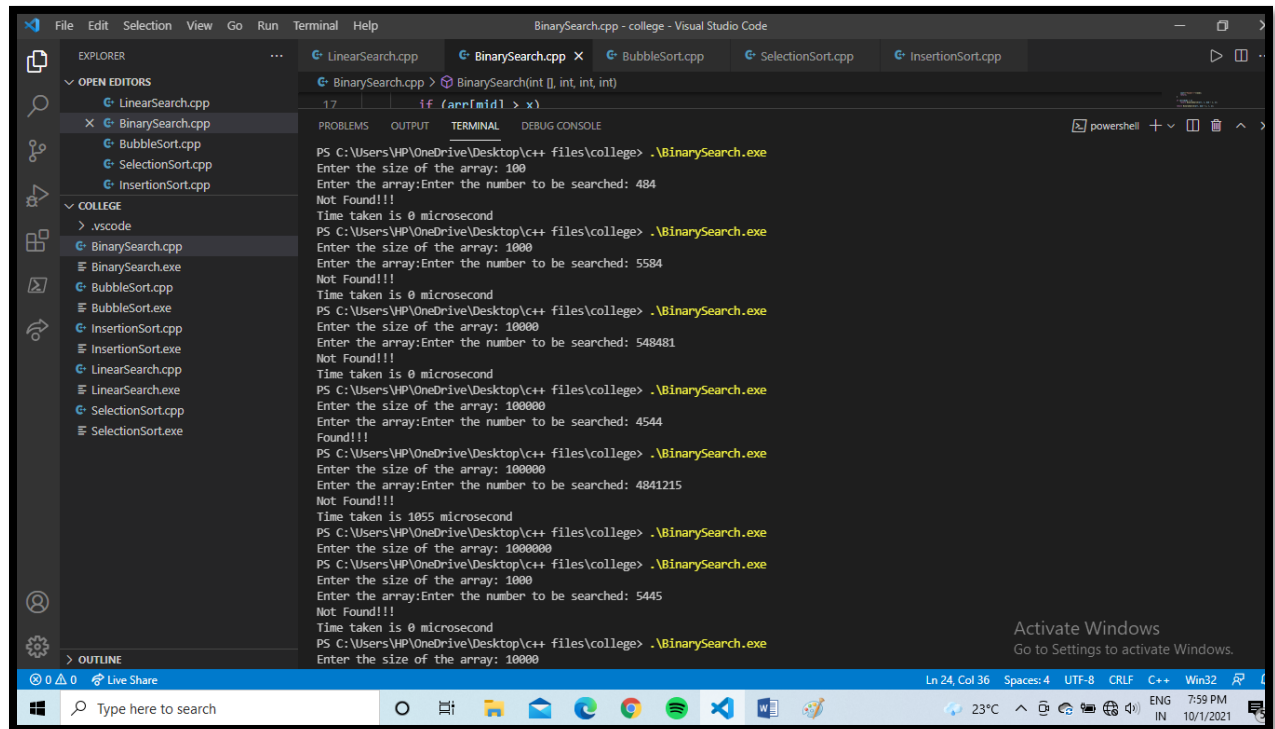
- Explorer:** A folder named 'COLLEGE' containing files: `.vscode`, `BinarySearch.cpp`, `BubbleSort.cpp`, `SelectionSort.cpp`, `InsertionSort.cpp`, `LinearSearch.cpp`, `BinarySearch.exe`, `BubbleSort.exe`, `InsertionSort.exe`, `LinearSearch.exe`, `SelectionSort.exe`, and `SelectionSort.cpp`.
- Editor:** The file `LinearSearch.cpp` is open, showing a function signature `LinearSearch(int l, int, int)` and a line number 14.
- Terminal:** The terminal output shows the execution of `LinearSearch.exe` with the following commands and results:


```

PS C:\Users\VIP\OneDrive\Desktop\c++ files\college> .\LinearSearch.exe
Enter the size of the array: 100
Enter the array:Enter the number to be searched: 256
Not Found!!!
PS C:\Users\VIP\OneDrive\Desktop\c++ files\college> .\LinearSearch.exe
Enter the size of the array: 1000
Enter the array:Enter the number to be searched: 2569
Not Found!!!
PS C:\Users\VIP\OneDrive\Desktop\c++ files\college> .\LinearSearch.exe
Enter the size of the array: 10000
Enter the array:Enter the number to be searched: 5165
Not Found!!!
PS C:\Users\VIP\OneDrive\Desktop\c++ files\college> .\LinearSearch.exe
Enter the array:Enter the number to be searched: 154
Found!!!
Time taken is 996 microsecond
PS C:\Users\VIP\OneDrive\Desktop\c++ files\college> 1000000
1000000
PS C:\Users\VIP\OneDrive\Desktop\c++ files\college> .\LinearSearch.exe
Enter the size of the array: 1000000
PS C:\Users\VIP\OneDrive\Desktop\c++ files\college> .\LinearSearch.exe
Enter the size of the array: 1000000
PS C:\Users\VIP\OneDrive\Desktop\c++ files\college>

```
- Status Bar:** Shows 'Ln 15, Col 6', 'Spaces: 4', 'UTF-8', 'CRLF', 'C++', 'Win32', and a system tray with temperature (23°C), time (7:04 PM), and date (10/1/2021).

Binary Search:-



```
File Edit Selection View Go Run Terminal Help
BinarySearch.cpp - college - Visual Studio Code

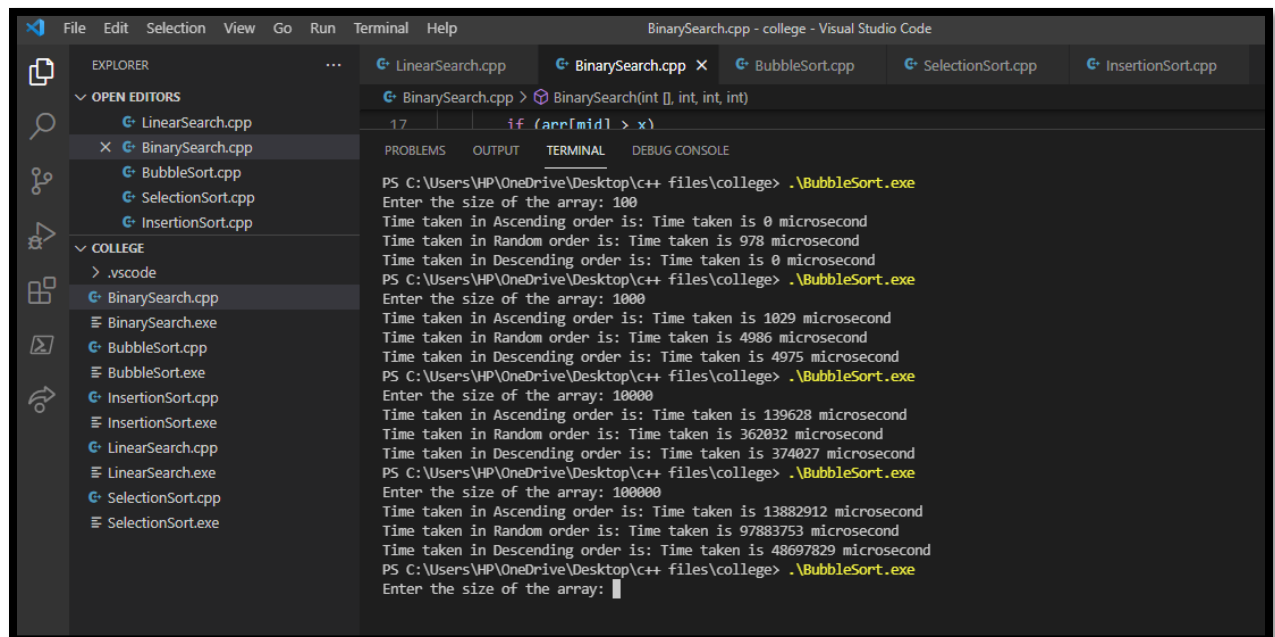
EXPLORER
  OPEN EDITORS
    LinearSearch.cpp
    BinarySearch.cpp
    BubbleSort.cpp
    SelectionSort.cpp
    InsertionSort.cpp
  COLLEGE
    .vscode
    BinarySearch.cpp
    BinarySearch.exe
    BubbleSort.cpp
    BubbleSort.exe
    InsertionSort.cpp
    InsertionSort.exe
    LinearSearch.cpp
    LinearSearch.exe
    SelectionSort.cpp
    SelectionSort.exe

  OUTLINE

  Live Share

  PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BinarySearch.exe
  Enter the size of the array: 100
  Enter the array: Enter the number to be searched: 484
  Not Found!!!
  Time taken is 0 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BinarySearch.exe
  Enter the size of the array: 1000
  Enter the array: Enter the number to be searched: 5584
  Not Found!!!
  Time taken is 0 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BinarySearch.exe
  Enter the size of the array: 10000
  Enter the array: Enter the number to be searched: 548481
  Not Found!!!
  Time taken is 0 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BinarySearch.exe
  Enter the size of the array: 100000
  Enter the array: Enter the number to be searched: 4544
  Found!!!
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BinarySearch.exe
  Enter the size of the array: 100000
  Enter the array: Enter the number to be searched: 4841215
  Not Found!!!
  Time taken is 1055 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BinarySearch.exe
  Enter the size of the array: 1000000
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BinarySearch.exe
  Enter the size of the array: 1000
  Enter the array: Enter the number to be searched: 5445
  Not Found!!!
  Time taken is 0 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BinarySearch.exe
  Enter the size of the array: 100000
```

Bubble Sort:-



```
File Edit Selection View Go Run Terminal Help
BinarySearch.cpp - college - Visual Studio Code

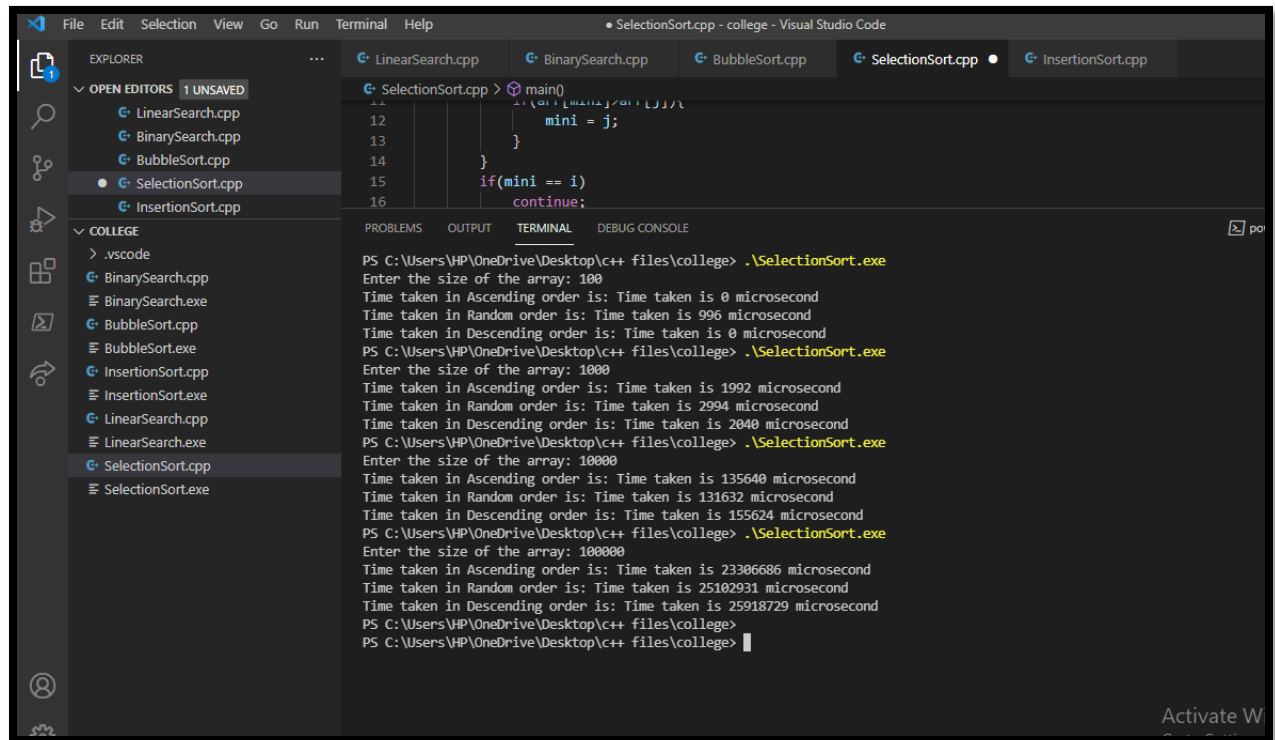
EXPLORER
  OPEN EDITORS
    LinearSearch.cpp
    BinarySearch.cpp
    BubbleSort.cpp
    SelectionSort.cpp
    InsertionSort.cpp
  COLLEGE
    .vscode
    BinarySearch.cpp
    BinarySearch.exe
    BubbleSort.cpp
    BubbleSort.exe
    InsertionSort.cpp
    InsertionSort.exe
    LinearSearch.cpp
    LinearSearch.exe
    SelectionSort.cpp
    SelectionSort.exe

  OUTLINE

  Live Share

  PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BubbleSort.exe
  Enter the size of the array: 100
  Time taken in Ascending order is: Time taken is 0 microsecond
  Time taken in Random order is: Time taken is 978 microsecond
  Time taken in Descending order is: Time taken is 0 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BubbleSort.exe
  Enter the size of the array: 1000
  Time taken in Ascending order is: Time taken is 1029 microsecond
  Time taken in Random order is: Time taken is 4986 microsecond
  Time taken in Descending order is: Time taken is 4975 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BubbleSort.exe
  Enter the size of the array: 10000
  Time taken in Ascending order is: Time taken is 139628 microsecond
  Time taken in Random order is: Time taken is 362032 microsecond
  Time taken in Descending order is: Time taken is 374027 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BubbleSort.exe
  Enter the size of the array: 100000
  Time taken in Ascending order is: Time taken is 13882912 microsecond
  Time taken in Random order is: Time taken is 97883753 microsecond
  Time taken in Descending order is: Time taken is 48697829 microsecond
  PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\BubbleSort.exe
  Enter the size of the array: 
```


Selection Sort:-



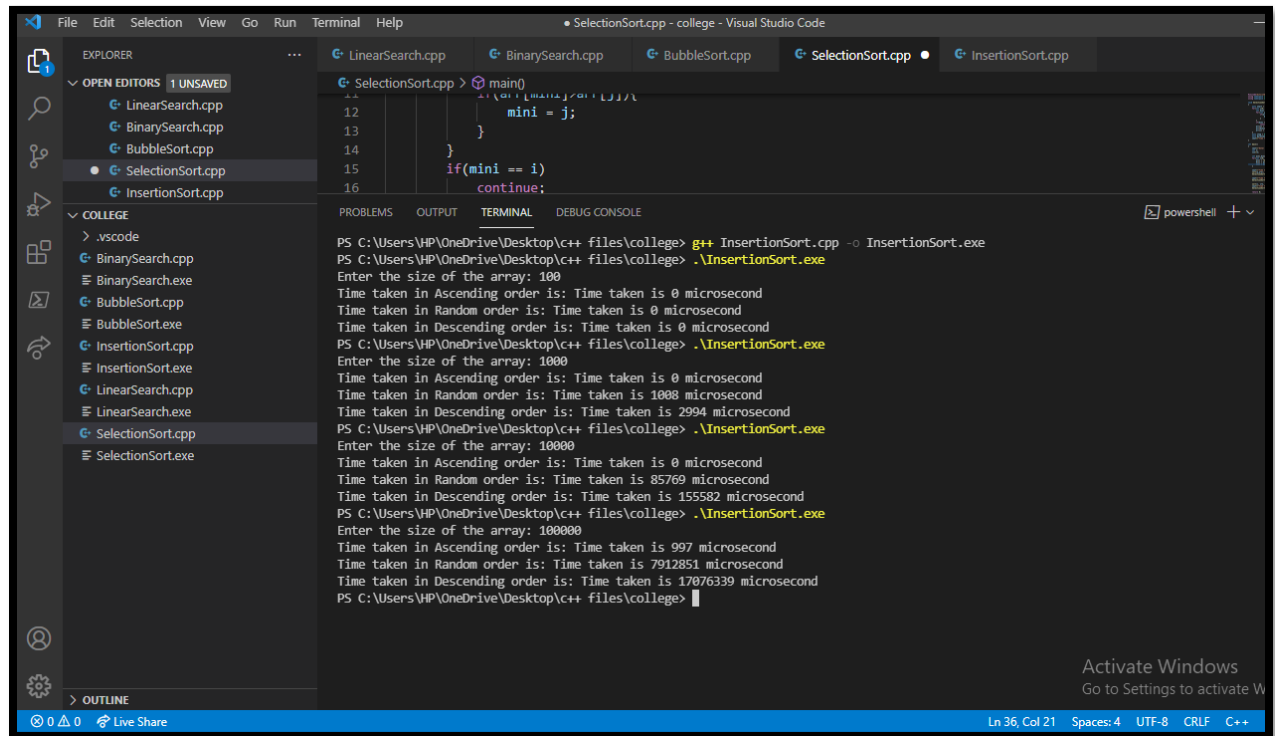
The screenshot displays the Visual Studio Code interface with the 'SelectionSort.cpp' file open. The code implements the Selection Sort algorithm. The Explorer sidebar shows the project structure with files like 'LinearSearch.cpp', 'BinarySearch.cpp', 'BubbleSort.cpp', 'SelectionSort.cpp', and 'InsertionSort.cpp'. The Terminal panel shows the execution of the program for different array sizes (100, 1000, 10000, 100000) and the time taken for ascending, random, and descending order sorting.

```
SelectionSort.cpp > main()
...
12     mini = j;
13     }
14 }
15 if(mini == i)
16     continue;
```

Terminal Output:

```
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\SelectionSort.exe
Enter the size of the array: 100
Time taken in Ascending order is: Time taken is 0 microsecond
Time taken in Random order is: Time taken is 996 microsecond
Time taken in Descending order is: Time taken is 0 microsecond
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\SelectionSort.exe
Enter the size of the array: 1000
Time taken in Ascending order is: Time taken is 1992 microsecond
Time taken in Random order is: Time taken is 2994 microsecond
Time taken in Descending order is: Time taken is 2040 microsecond
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\SelectionSort.exe
Enter the size of the array: 10000
Time taken in Ascending order is: Time taken is 135640 microsecond
Time taken in Random order is: Time taken is 131632 microsecond
Time taken in Descending order is: Time taken is 155624 microsecond
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\SelectionSort.exe
Enter the size of the array: 100000
Time taken in Ascending order is: Time taken is 23306686 microsecond
Time taken in Random order is: Time taken is 25102931 microsecond
Time taken in Descending order is: Time taken is 25918729 microsecond
PS C:\Users\HP\OneDrive\Desktop\c++ files\college>
```

Insertion Sort:-



The screenshot displays the Visual Studio Code interface with the following components:

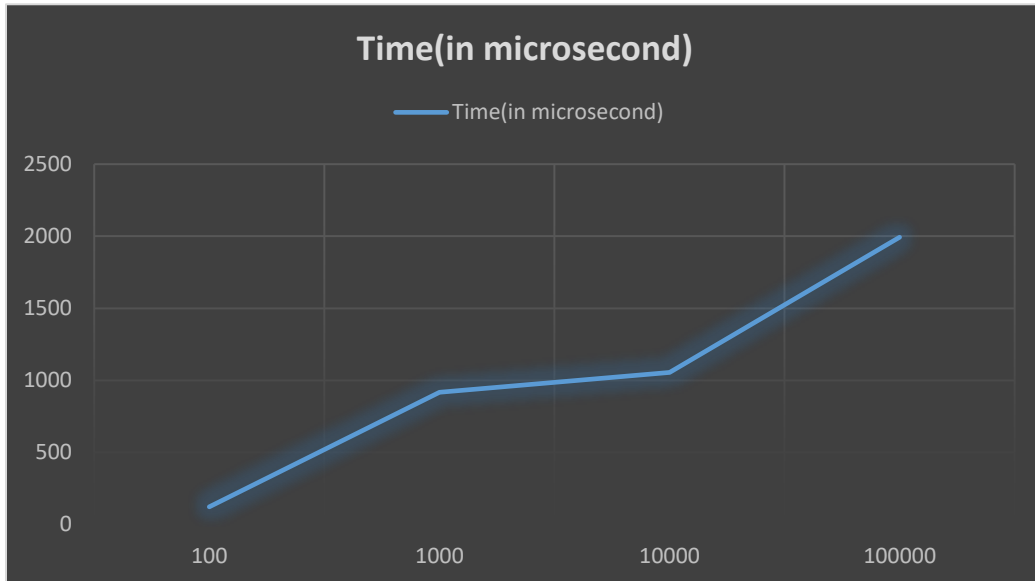
- EXPLORER:** Shows the project structure with files like `LinearSearch.cpp`, `BinarySearch.cpp`, `BubbleSort.cpp`, `SelectionSort.cpp`, and `InsertionSort.cpp`.
- EDITOR:** Displays the `SelectionSort.cpp` file with the following code:

```
12     }  
13     }  
14     }  
15     if(mini == i)  
16         continue;
```
- TERMINAL:** Shows the execution of the `InsertionSort.exe` program. The output includes the size of the array and the time taken for different input sizes and orders.

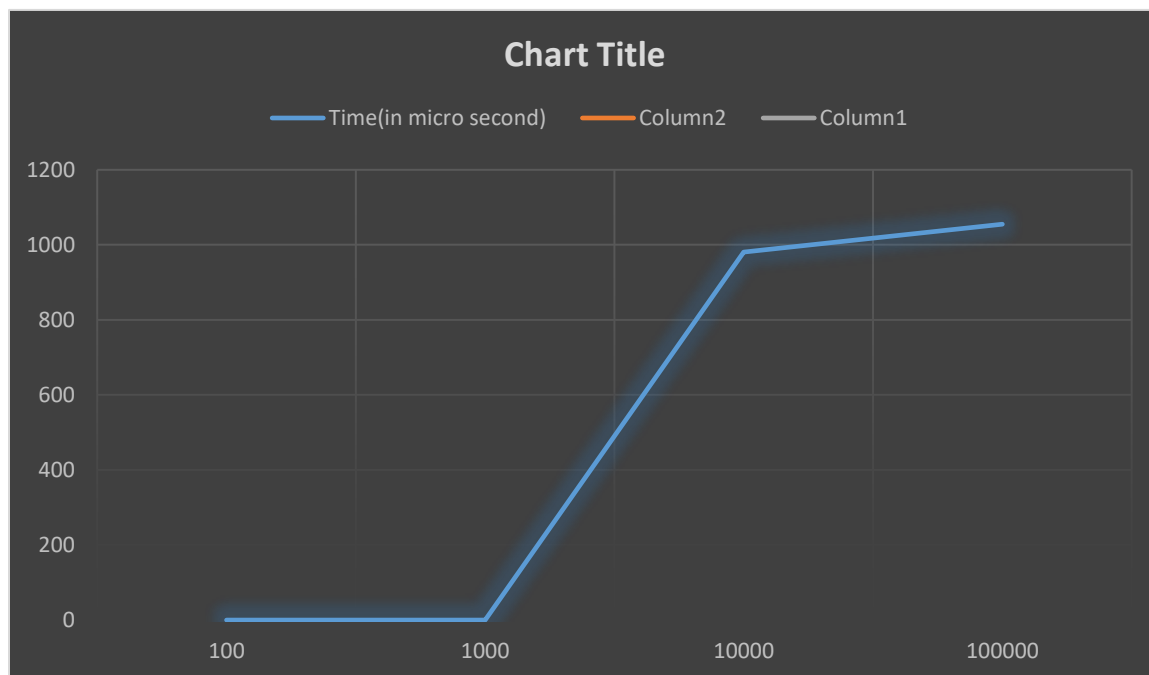
```
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> g++ InsertionSort.cpp -o InsertionSort.exe  
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\InsertionSort.exe  
Enter the size of the array: 100  
Time taken in Ascending order is: Time taken is 0 microsecond  
Time taken in Random order is: Time taken is 0 microsecond  
Time taken in Descending order is: Time taken is 0 microsecond  
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\InsertionSort.exe  
Enter the size of the array: 1000  
Time taken in Ascending order is: Time taken is 0 microsecond  
Time taken in Random order is: Time taken is 1008 microsecond  
Time taken in Descending order is: Time taken is 2994 microsecond  
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\InsertionSort.exe  
Enter the size of the array: 10000  
Time taken in Ascending order is: Time taken is 0 microsecond  
Time taken in Random order is: Time taken is 85769 microsecond  
Time taken in Descending order is: Time taken is 155582 microsecond  
PS C:\Users\HP\OneDrive\Desktop\c++ files\college> .\InsertionSort.exe  
Enter the size of the array: 100000  
Time taken in Ascending order is: Time taken is 997 microsecond  
Time taken in Random order is: Time taken is 7912851 microsecond  
Time taken in Descending order is: Time taken is 17076339 microsecond  
PS C:\Users\HP\OneDrive\Desktop\c++ files\college>
```
- STATUS BAR:** Shows the current line and column (Ln 36, Col 21) and the file encoding (UTF-8, CRLF).

Task - 3

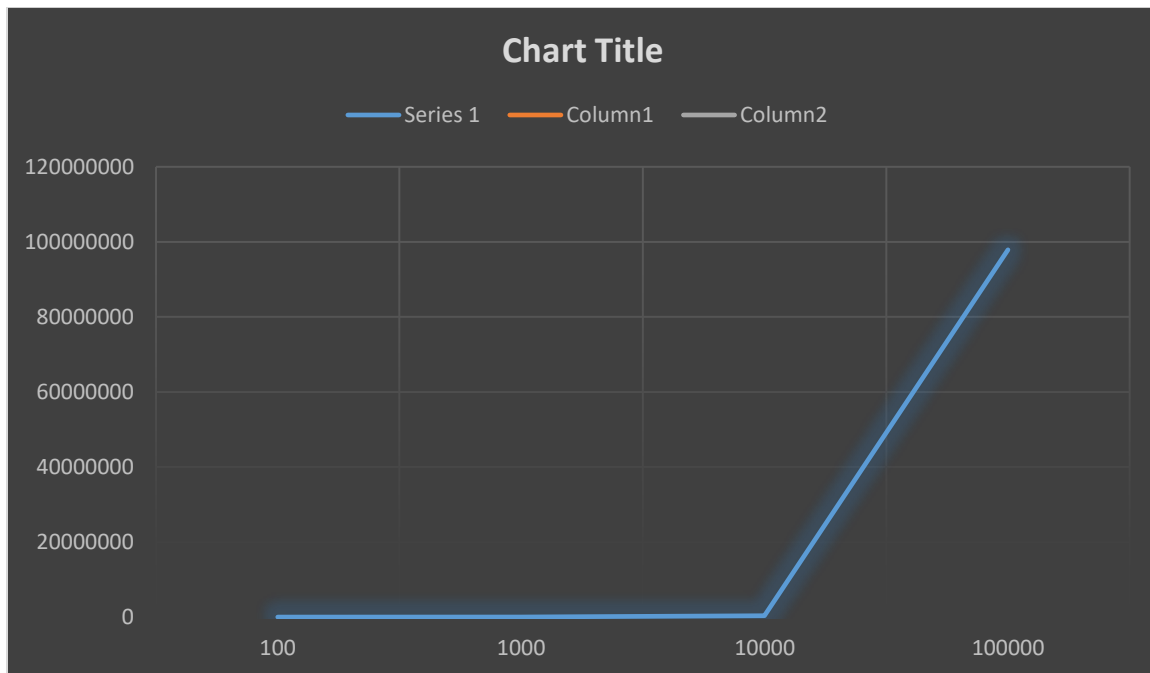
Linear Search:-



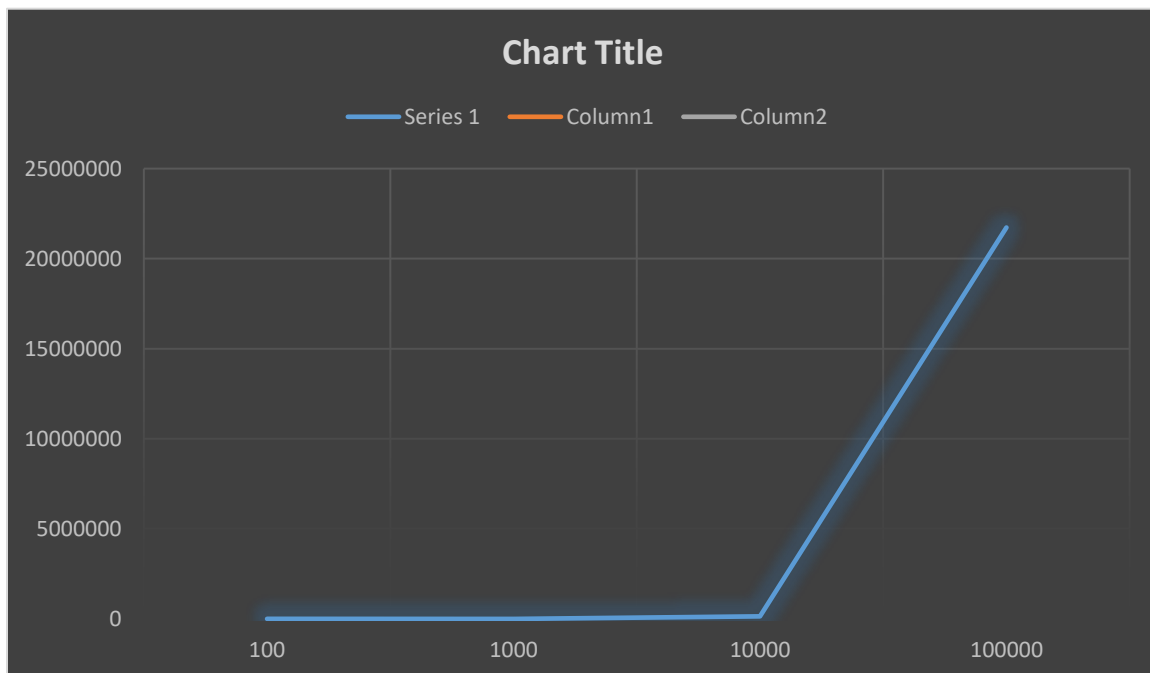
Binary Search:-



Bubble Sort:-



Selection Sort:-



Insertion Sort:-

