Experiment No. 5

Aim: Implementation of selection sorting technique considering a real world application.

Objectives: ~~To imp~~
1. To impart knowledge of sorting and searching algorithms.
2. To develop an ability and analyze algorithms using various data structures.

Theory:

1) Introduction to sorting:-
- Sorting refers to the process of ordering data in an increasing or decreasing fashion according to some linear relationship among the data items.
- For eg - arr [8, 10, 7, 9, 4]
  Sorted array in ascending order - arr [4, 7, 8, 9, 10]

2) Types of sorting -
① Bubble sort - It is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order.

② Insertion sort - In insertion sort, the array is virtually divided into a sorted and unsorted part. Values from the unsorted part are picked and placed at the correct position.

③ Selection sort - Selection sort finds the smallest element in the array and place it on the first place of the list, then it finds the second smallest elements in the array and place it on the second place. This process continues until all elements are moved to their correct order.

④ Merge sort - In merge sort, the list is divided into sets of equal elements and then each half of the list is sorted. This approach of merge sort is known as divide and conquer approach.

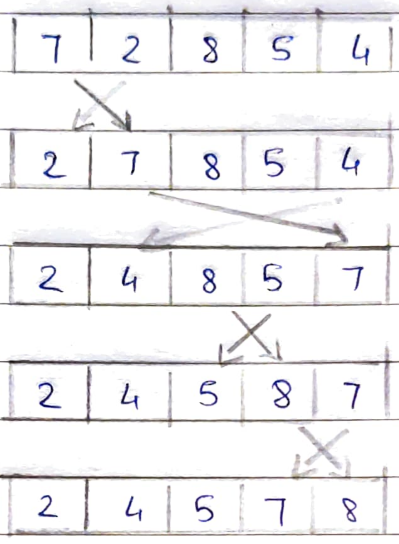
3) Introduction to selection sort:

Insertion sort is a simple sorting algorithm. This sorting algorithm is an inplace comparison based algorithm in which the list is divided into two parts, the sorted part at the left end and unsorted part at the right end. Initially, the sorted part is empty and the unsorted part is the entire list.

4) Algorithm:

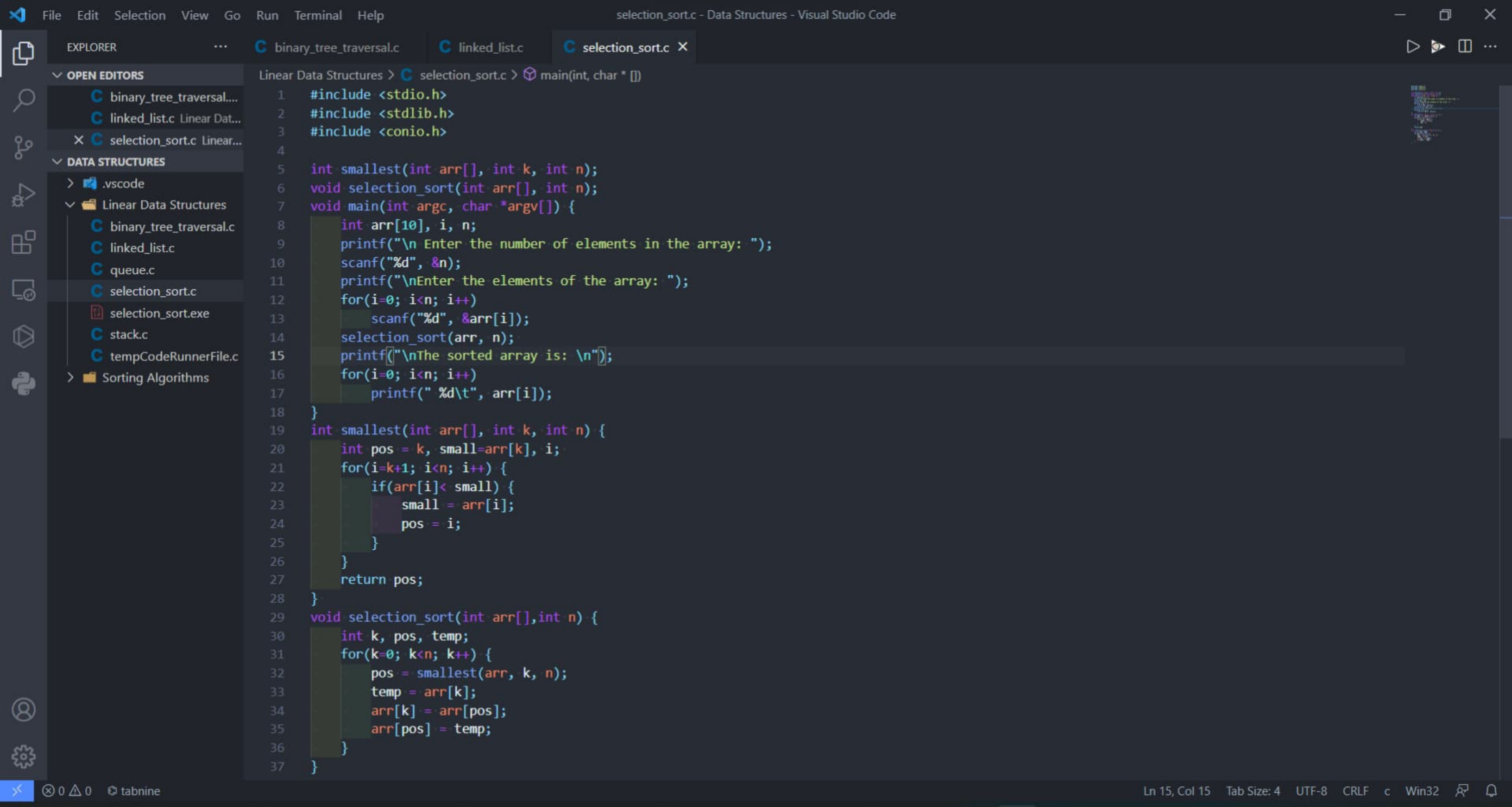
```
For i=0 to i=n-~~2~~1
    minimum_value ← i
    For j =i+1 to ~~i~~ i=n-1
        if arr[j] < arr [min] minimum_value ← j
    swap arr[i] and arr [minimum_value]
```

5) Example:



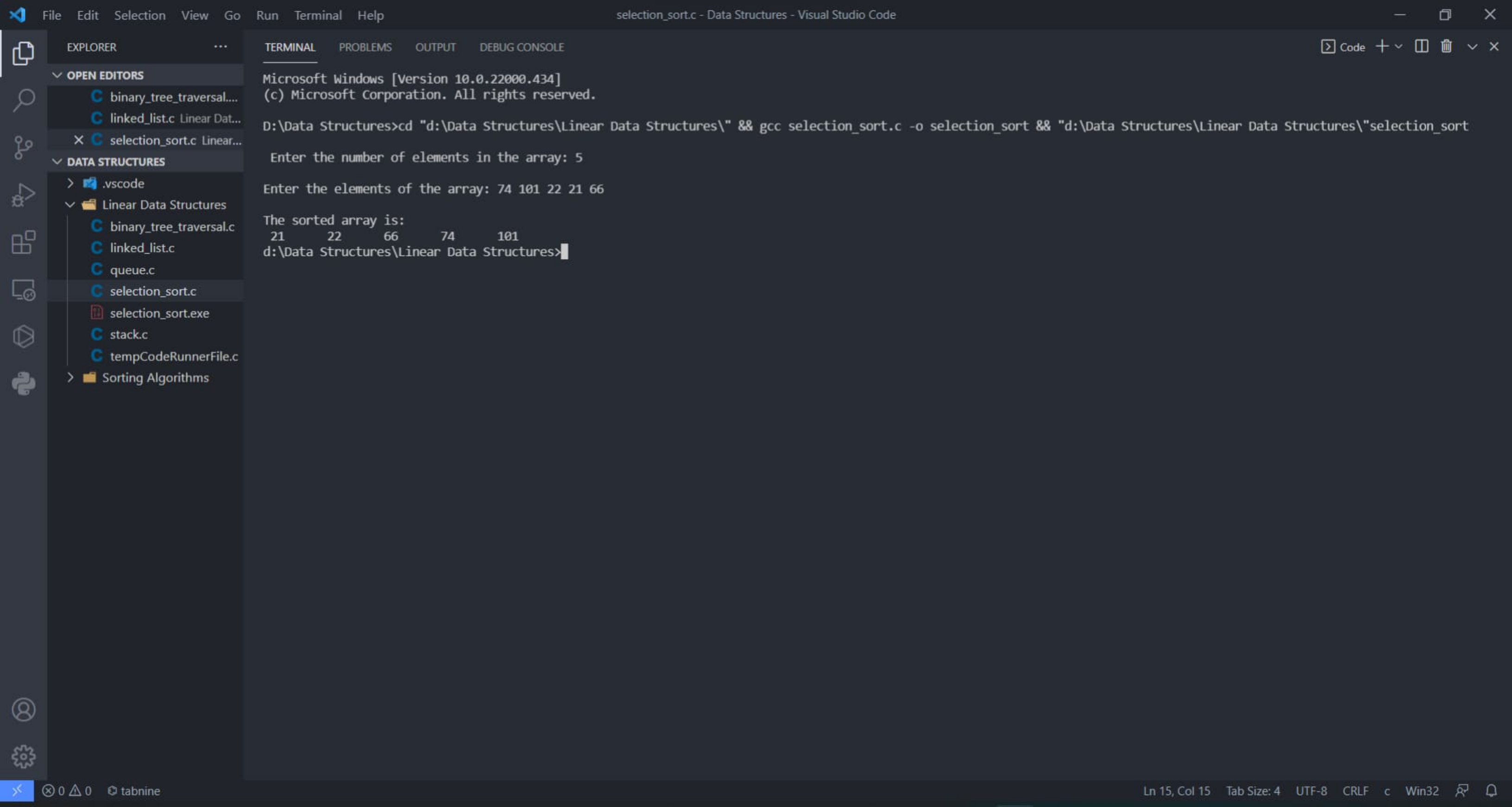
Conclusion: Despite being a simple algorithm, it is effective only on smaller data sets. It manifests the worst-case time complexity of O(n2).

Outcome: Implemented ~~insertion~~ sorting and searching techniques for real-world applications.


File Edit Selection View Go Run Terminal Help
selection_sort.c - Data Structures - Visual Studio Code
EXPLORER
OPEN EDITORS
binary_tree_traversal....
linked_list.c Linear Dat...
selection_sort.c Linear...
DATA STRUCTURES
.vscode
Linear Data Structures
binary_tree_traversal.c
linked_list.c
queue.c
selection_sort.c
selection_sort.exe
stack.c
tempCodeRunnerFile.c
Sorting Algorithms
binary_tree_traversal.c
linked_list.c
selection_sort.c
Linear Data Structures > selection_sort.c > main(int, char * [])
0 0 tabnine
Ln 15, Col 15 Tab Size: 4 UTF-8 CRLF c Win32

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

int smallest(int arr[], int k, int n);
void selection_sort(int arr[], int n);
void main(int argc, char *argv[]) {
    int arr[10], i, n;
    printf("\n Enter the number of elements in the array: ");
    scanf("%d", &n);
    printf("\nEnter the elements of the array: ");
    for(i=0; i<n; i++)
        scanf("%d", &arr[i]);
    selection_sort(arr, n);
    printf("\nThe sorted array is: \n");
    for(i=0; i<n; i++)
        printf(" %d\t", arr[i]);
}
int smallest(int arr[], int k, int n) {
    int pos = k, small=arr[k], i;
    for(i=k+1; i<n; i++) {
        if(arr[i]< small) {
            small = arr[i];
            pos = i;
        }
    }
    return pos;
}
void selection_sort(int arr[],int n) {
    int k, pos, temp;
    for(k=0; k<n; k++) {
        pos = smallest(arr, k, n);
        temp = arr[k];
        arr[k] = arr[pos];
        arr[pos] = temp;
    }
}
```


File Edit Selection View Go Run Terminal Help
selection_sort.c - Data Structures - Visual Studio Code
EXPLORER
OPEN EDITORS
binary_tree_traversal....
linked_list.c Linear Dat...
selection_sort.c Linear...
DATA STRUCTURES
.vscode
Linear Data Structures
binary_tree_traversal.c
linked_list.c
queue.c
selection_sort.c
selection_sort.exe
stack.c
tempCodeRunnerFile.c
Sorting Algorithms
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE
Code
0 0 tabnine
Ln 15, Col 15 Tab Size: 4 UTF-8 CRLF c Win32

```
Microsoft Windows [Version 10.0.22000.434]
(c) Microsoft Corporation. All rights reserved.

D:\Data Structures>cd "d:\Data Structures\Linear Data Structures\" && gcc selection_sort.c -o selection_sort && "d:\Data Structures\Linear Data Structures\"selection_sort

 Enter the number of elements in the array: 5

Enter the elements of the array: 74 101 22 21 66

The sorted array is:
 21      22      66      74      101
d:\Data Structures\Linear Data Structures>
```