Experiment No. 5

Date: / /

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

Objectives: 1) To impart knowledge of sorting and serching algorithm.
2) To develop an ability to design and analyzie algorithms using various data structures.

Theory:

1) Introduction to sorting: Sorting is the process of arranging the elements at an array so that they can be placed either in ascending or descending order for eg consider any array $A = \{A_1, A_2, A_3, A_4 \dots A_n\}$, the array is called to be in ascending order if element of A are arranged like $A_1 > A_2 > A_3 > A_4 > \dots > A_n$

2) Types of sorting:

i) Bubble sort: It is the simplest sort method which perform sorting by repeatedly moving the largest element to the heighest index. of array. It comprises of comparing each element to its adjacent element and replace them accordingly.

ii) Insertion sort: The insertion sort insert each element of the array to its proper place. It is very simple sort method which is used to arrange the deck of cards while playing bridge

iii) Selection sort: Selection sort finds the smallest element in the array and place it on the first

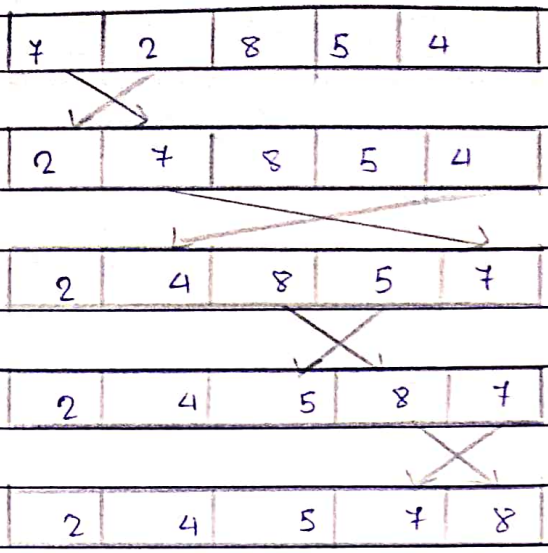
first place of the array and place it on the second place. This process continues until all elements are moved to their correct order.

iv) Merge sort: Merge sort follows divide and conquer approach in which the list is first divided into the sets of equal elements and then each half of the list is sorted by using merge sort.

3) Introduction to selection sort:
It is a simple sorting algorithm. This sorting algorithm is an inplace comparision 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. Initialy the sorted part is empty and the unsorted part is the entire list.

4) Algorithm:
Selection sort (A[0....n-1])
// sorts a given array by selection sort
// input: An array A[0....n-1] of orderable elements
// output: Array A[0....n-1] sorted in asending order.
For i←0 to n-2 do
    min ← i
    for j ← i+1 to n-1 do
        if A[j] < A[min] min ← j
    Swap A[i] and A[min]

5) Example :



Conclusion : Selection sort is sorting algorithm known by its simplicity Unfortunately it lacks efficiency on huge lists of items and also it does not stop unless the number of iterations has been achived. even through the list is already sorted.

Outcom : Implement sorting and searching techniques for real world application.

PROGRAM CODE

```c
/*************************************
 Implementation of Selection Sort
*************************************/
#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 Please Enter the total Number of Elements  : ");
scanf("%d", &n);
printf("\n Please Enter the Array Elements  :  ");
for(i=0;i<n;i++) {  scanf("%d", &arr[i]); }
selection_sort(arr, n);
printf("\n The 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;
}
}
```

OUTPUT:

```
C:\TURBOC3\BIN>TC

 Please Enter the total Number of Elements  : 5

 Please Enter the Array Elements  :  -1 3 10 1 0

 The sorted array is:
 -1      0       1       3       10
```