Data Structures Lab

Experiment 2: Sorting Algorithms

Project By:

Mohammed Rabeeh
Roll No: 35
TVE18CS036

Contents

1 Aim

To create a C program implementing different sorting algorithms and verifying their outputs.

2 Problem Description

Sorting algorithm is any algorithm which is used to arrange data in either ascending order or descending order. There are a number of sorting algorithms which vary in their efficiency. Commonly used sorting algorithms are Bubble sort, Selection sort, Insertion sort, Quick sort, Heap sort and Merge sort.

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order.

Selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning. The algorithm maintains two subarrays in a given array.

1. The subarray which is already sorted.
2. Remaining subarray which is unsorted.

In every iteration of selection sort, the minimum element (considering ascending order) from the unsorted subarray is picked and moved to the sorted subarray.

3 Algorithm

3.1 Bubble Sort

1. Take the input array.
2. Starting with the first element(index = 0), compare the current element with the next element of the array.
3. If the current element is greater than the next element of the array, swap them.
4. If the current element is less than the next element, move to the next element.
5. Repeat for each element in the array.

3.2 Selection Sort

1. Take the input array.
2. Set the minimum to the left most element.
3. Search the minimum in the remaining list.
4. Swap the minimum with the current location.
5. Increment the minimum and repeat the steps.

4 Program Code

```
#include <stdio.h>
#define MAX_SIZE 100
void bubbleSort(int arr[], int n) {
   int i, j, temp;
   for (i = 0; i < n-1; i++) {
      for (j = 0; j < n-i-1; j++) {
         if (arr[j] > arr[j+1]) {
            temp = arr[j];
            arr[j] = arr[j+1];
```

```
            arr[j+1] = temp;
          }
       }
    }

}

void selectionSort(int arr[], int n)
{
    int i, j, min_idx, temp;
    for (i = 0; i < n-1; i++) {
        min_idx = i;
        for (j = i+1; j < n; j++) {
          if (arr[j] < arr[min_idx]) {
            min_idx = j;
          }
      }
        temp = arr[min_idx];
      arr[min_idx] = arr[i];
      arr[i] = temp;
    }
}

void main() {
   int arr[MAX_SIZE], n, option, i;
   printf("Sorting...\n1.Bubble Sort\n2.Selection Sort\nChoose an
       option: ");
   scanf("%d", &option);
   if(option != 1 && option != 2) {
      printf("Invalid option\n");
   } else {
      printf("Enter the no. of elements: ");
      scanf("%d", &n);
      printf("Enter the elements: \n");
      for(i = 0; i < n; i++) {
         scanf("%d", &arr[i]);
      }
      if(option == 1) {
         bubbleSort(arr, n);
      } else {
```

```
            selectionSort(arr,n);
        }
        printf("Sorted array: ");
        for(i = 0; i < n; i++) {
            printf("%d ", arr[i]);
        }
        printf("\n");
    }
}
```

5 Output

```
rabeehrz@BatPC:~/college/s3dslab/cycle1$ ./a.out
Sorting...
1.Bubble Sort
2.Selection Sort
Choose an option: 1
Enter the no. of elements: 5
Enter the elements:
9 8 10 6 -8
Sorted array: -8 6 8 9 10
rabeehrz@BatPC:~/college/s3dslab/cycle1$ ./a.out
Sorting...
1.Bubble Sort
2.Selection Sort
Choose an option: 2
Enter the no. of elements: 7
Enter the elements:
0 -3 7 19 4 -3 6
Sorted array: -3 -3 0 4 6 7 19
```

6 Result

The sorting algorithms **Bubble Sort** and **Linear Search** were implemented in C and the output was verified.