Lab Sheet 3: Implement Sorting Algorithms – Bubble Sort and Selection Sort

Aim:

To implement Bubble Sort and Selection Sort algorithms in C and to understand their working and time complexity.

Procedure:

• Open Turbo C and create a new C program.

• Include the required header file stdio.h.

• Read the number of elements and array values from the user.

• Implement the Bubble Sort algorithm to sort the array in ascending order.

• Display the sorted array obtained using Bubble Sort.

• Implement the Selection Sort algorithm on the same set of elements.

• Display the sorted array obtained using Selection Sort.

• Compile and run the program.

• Observe and compare the results of both sorting techniques.

Program: Bubble Sort and Selection Sort

```
#include <stdio.h>

#include <time.h>

// Function to perform Bubble Sort

void bubbleSort(int a[], int n) {

int i, j, temp;

for(i = 0; i < n - 1; i++) {

for(j = 0; j < n - i - 1; j++) {

if(a[j] > a[j + 1]) {

temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp;

}

}

}

}
```

```
// Function to perform Selection Sort
void selectionSort(int a[], int n) {
int i, j, minIndex, temp;
for(i = 0; i < n - 1; i++) {
minIndex = i;
for(j = i + 1; j < n; j++) {
if(a[j] < a[minIndex])
minIndex = j;
}
temp = a[i];
a[i] = a[minIndex];
a[minIndex] = temp;
}
}
// Function to display array
void displayArray(int a[], int n) {
for(int i = 0; i < n; i++)
printf("%d ", a[i]);
printf("\n");
}
int main() {
int a[20], n, original[20];
clock_t start, end;
double time_taken;
// Input array elements
printf("Enter number of elements: ");
scanf("%d", &n);
printf("Enter array elements:\n");
for(int i = 0; i < n; i++) {
scanf("%d", &a[i]);
original[i] = a[i]; // store original array for fair comparison
}
// Bubble Sort
start = clock();
bubbleSort(a, n);
end = clock();
time_taken = ((double)(end - start)) / CLOCKS_PER_SEC;
printf("\nSorted array using Bubble Sort:\n");
displayArray(a, n);
printf("Time taken by Bubble Sort: %f seconds\n", time_taken);
// Restore original array for Selection Sort
for(int i = 0; i < n; i++)
a[i] = original[i];
// Selection Sort
start = clock();
selectionSort(a, n);
end = clock();
time_taken = ((double)(end - start)) / CLOCKS_PER_SEC;
printf("\nSorted array using Selection Sort:\n");
displayArray(a, n);
printf("Time taken by Selection Sort: %f seconds\n", time_taken);
return 0;
}
```

Result:

Bubble Sort and Selection Sort algorithms were implemented successfully. Both algorithms sorted the given array

correctly. It was observed that both algorithms have $O(n^2)$ time complexity and are suitable for small input sizes.