

DATA STRUCTURES AND ALGORITHMS LAB – 2 ASSIGNMENT

Student Details

- Name: Pratyush Deo Singh
 - Registration Number: 25BCE5101
 - Course / Branch: B. Tech CSE (CORE)
 - Semester: 2nd
 - Subject: Data Structures and Algorithms
 - Faculty Name: Dr. Malini A
-

Program 1: Quick Sort

Aim

To write a C program to sort elements using Quick Sort technique.

Algorithm

1. Read number of elements
2. Read array elements
3. Select a pivot element
4. Partition the array
5. Apply quick sort recursively

Program Code

```
C quick_sort.c > main()
1  #include <stdio.h>
2  void quickSort(int a[], int low, int high) {
3      int i = low, j = high;
4      int pivot = a[(low + high) / 2];
5      int temp;
6      while (i <= j) {
7          while (a[i] < pivot)
8              i++;
9          while (a[j] > pivot)
10             j--;
11         if (i <= j) {
12             temp = a[i];
13             a[i] = a[j];
14             a[j] = temp;
15             i++;
16             j--;
17         }
18     }
19     if (low < j)
20         quickSort(a, low, j);
21     if (i < high)
22         quickSort(a, i, high);
23 }
24 int main() {
25     int n, i;
26     printf("Enter number of elements: ");
27     scanf("%d", &n);
28     int a[n];
29     printf("Enter elements:\n");
30     for (i = 0; i < n; i++)
31         scanf("%d", &a[i]);
32     quickSort(a, 0, n - 1);
33     printf("Sorted array:\n");
34     for (i = 0; i < n; i++)
35         printf("%d ", a[i]);
36     return 0;
37 }
```

Output

```
Enter number of elements: 5
Enter elements:
1328
341123
132
1
3413
Sorted array:
1 132 1328 3413 341123
```

Program 2: Merge Sort

Aim

To write a C program to sort elements using Merge Sort technique.

Algorithm

1. Read number of elements
2. Divide array into subarrays
3. Sort subarrays recursively
4. Merge the sorted subarrays

Program Code

```
C merge_sort.c > ...
1  #include <stdio.h>
2
3  void merge(int a[], int l, int m, int r) {
4      int i, j, k;
5      int n1 = m - l + 1;
6      int n2 = r - m;
7
8      int L[n1], R[n2];
9
10     for (i = 0; i < n1; i++)
11         L[i] = a[l + i];
12     for (j = 0; j < n2; j++)
13         R[j] = a[m + 1 + j];
14
15     i = 0;
16     j = 0;
17     k = l;
18
19     while (i < n1 && j < n2) {
20         if (L[i] <= R[j]) {
21             a[k] = L[i];
22             i++;
23         } else {
24             a[k] = R[j];
25             j++;
26         }
27         k++;
28     }
29 }
```

Ln 70, Col 1 Spaces: 4 UTF-8 CRLF { } C Win32

C merge_sort.c > ...

```
3 void merge(int a[], int l, int m, int r) {
29
30     while (i < n1) {
31         a[k] = L[i];
32         i++;
33         k++;
34     }
35
36     while (j < n2) {
37         a[k] = R[j];
38         j++;
39         k++;
40     }
41 }
42
43 void mergeSort(int a[], int l, int r) {
44     if (l < r) {
45         int m = (l + r) / 2;
46         mergeSort(a, l, m);
47         mergeSort(a, m + 1, r);
48         merge(a, l, m, r);
49     }
50 }
51
52 int main() {
53     int n, i;
54     printf("Enter number of elements: ");
55     scanf("%d", &n);
56
```

Ln 70, Col 1 Spaces: 4 UTF-8 CRLF { } C Win32

C merge_sort.c > ...

```
52 int main() {
54     printf("Enter number of elements: ");
55     scanf("%d", &n);
56
57     int a[n];
58     printf("Enter elements:\n");
59     for (i = 0; i < n; i++)
60         scanf("%d", &a[i]);
61
62     mergeSort(a, 0, n - 1);
63
64     printf("Sorted array:\n");
65     for (i = 0; i < n; i++)
66         printf("%d ", a[i]);
67
68     return 0;
69 }
70
```

Ln 70, Col 1 Spaces: 4 UTF-8 CRLF { } C Win32

Output

```
Enter number of elements: 10
Enter elements:
123467
12341
1233
542
524552
687
8435268
4352
23452
245255
Sorted array:
542 687 1233 4352 12341 23452 123467 245255 524552 8435268
```

Program 3: Bubble Sort

Aim

To write a C program to sort elements using Bubble Sort technique.

Algorithm

1. Read number of elements
2. Compare adjacent elements
3. Swap if required
4. Repeat until sorted

Program Code

```
C bubble_sort.c > ...
1  #include <stdio.h>
2
3  int main() {
4      int n, i, j, temp;
5      printf("Enter number of elements: ");
6      scanf("%d", &n);
7
8      int a[n];
9      printf("Enter elements:\n");
10     for (i = 0; i < n; i++)
11         scanf("%d", &a[i]);
12
13     for (i = 0; i < n - 1; i++) {
14         for (j = 0; j < n - i - 1; j++) {
15             if (a[j] > a[j + 1]) {
16                 temp = a[j];
17                 a[j] = a[j + 1];
18                 a[j + 1] = temp;
19             }
20         }
21     }
22
23     printf("Sorted array:\n");
24     for (i = 0; i < n; i++)
25         printf("%d ", a[i]);
26
27     return 0;
28 }
29
```

Ln 29, Col 1 Spaces: 4 UTF-8 CRLF {} C Win32

Output

```
Enter number of elements: 5
Enter elements:
123
134
453
10
124
Sorted array:
10 123 124 134 453
```

Result

Thus, the given sorting algorithms were successfully implemented and executed.