

AMRITA SCHOOL OF COMPUTING

**DESIGN AND ANALYSIS OF
ALGORITHMS
(23CSE211)**

Name: P .JEEVAN SANDEEP

Roll No.: CH.SC.U4CSE24134

Class: BTech (CSE-B)

School: Amrita School of Computing,
Chennai Campus.

LAB-6

- 1) Quick Sort using first, last, and random pivot selection methods. Design a menu-driven program that allows the user to choose any method, prints the randomly selected pivot.

Code:

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  // JEEVAN SANDEEP
4  void swap(int *a, int *b){
5      int temp = *a;
6      *a = *b;
7      *b = temp;
8  }
9  int partitionFirst(int a[], int low, int high){
10     int pivot = a[low];
11     int i = low + 1, j = high;
12     while(i <= j){
13         while(i <= high && a[i] <= pivot){
14             i++;
15         }
16         while(a[j] > pivot){
17             j--;
18         }
19         if(i < j){
20             swap(&a[i], &a[j]);
21         }
22     }
23     swap(&a[low], &a[j]);
24     return j;
25 }
26 int partitionLast(int a[], int low, int high){
27     int pivot = a[high];
28     int i = low - 1;
29     for(int j = low; j < high; j++){
30         if(a[j] <= pivot){
31             i++;
32             swap(&a[i], &a[j]);
33         }
34     }
35     swap(&a[i + 1], &a[high]);
36     return i + 1;
37 }
```

```

37     int partitionRandom(int a[], int low, int high) {
38         int randomIndex = low + rand() % (high - low + 1);
39         printf("Randomly selected pivot: %d\n", a[randomIndex]);
40         swap(&a[randomIndex], &a[high]);
41         return partitionLast(a, low, high);
42     }
43
44     void quickSort(int a[], int low, int high, int choice){
45         if(low < high){
46             int p;
47             if(choice == 1){
48                 p = partitionFirst(a, low, high);
49             }
50             else if(choice == 2){
51                 p = partitionLast(a, low, high);
52             }
53             else{
54                 p = partitionRandom(a, low, high);
55             }
56             quickSort(a, low, p - 1, choice);
57             quickSort(a, p + 1, high, choice);
58         }
59     }
60     void copyArray(int src[], int dest[], int n){
61         for(int i = 0; i < n; i++){
62             dest[i] = src[i];
63         }
64     }
65     void display(int a[], int n){
66         for(int i = 0; i < n; i++){
67             printf("%d ", a[i]);
68         }
69         printf("\n");
70     }

```

```

71 int main(){
72     int n, choice;
73     printf("Enter number of elements: ");
74     scanf("%d", &n);
75     int original[n], temp[n];
76     printf("Enter elements:\n");
77     for(int i = 0; i < n; i++)
78         scanf("%d", &original[i]);
79     while(1){
80         printf("\n----- QUICK SORT MENU ----- \n");
81         printf("1. First Element as Pivot\n");
82         printf("2. Last Element as Pivot\n");
83         printf("3. Random Element as Pivot\n");
84         printf("4. Exit\n");
85         printf("Enter your choice: ");
86         scanf("%d", &choice);
87         if(choice == 4){
88             printf("Exiting program...\n");
89             break;
90         }
91         if(choice < 1 || choice > 3){
92             printf("Invalid choice! Try again.\n");
93             continue;
94         }
95         copyArray(original, temp, n);
96         printf("\nOriginal array:\n");
97         display(temp, n);
98         quickSort(temp, 0, n - 1, choice);
99         printf("Sorted array:\n");
100        display(temp, n);
101    }
102    return 0;
103 }

```

Output:

```
Enter number of elements: 5
Enter elements:
8
5
9
3
2
```

```
----- QUICK SORT MENU -----
1. First Element as Pivot
2. Last Element as Pivot
3. Random Element as Pivot
4. Exit
Enter your choice: 1
```

```
Original array:
8 5 9 3 2
Sorted array:
2 3 5 8 9
```

```
----- QUICK SORT MENU -----
1. First Element as Pivot
2. Last Element as Pivot
3. Random Element as Pivot
4. Exit
Enter your choice: 2
```

```
Original array:
8 5 9 3 2
Sorted array:
2 3 5 8 9
```

```
----- QUICK SORT MENU -----
1. First Element as Pivot
2. Last Element as Pivot
3. Random Element as Pivot
4. Exit
Enter your choice: 3
```

```
Original array:
8 5 9 3 2
Randomly selected pivot: 5
Randomly selected pivot: 3
Randomly selected pivot: 8
Sorted array:
2 3 5 8 9
```

```
----- QUICK SORT MENU -----
1. First Element as Pivot
2. Last Element as Pivot
3. Random Element as Pivot
4. Exit
Enter your choice: 4
Exiting program...
```

```
-----
Process exited after 27.02 seconds with return value 0
Press any key to continue . . . |
```

Space Complexity:

The space complexity of this program is $O(\log n)$ for best and average cases and $O(n)$ for worst case.

Time Complexity:

The time complexity of this program is $O(n \log n)$ for best and average cases and $O(n^2)$ for worst case.