AMRITA SCHOOL OF COMPUTING

DESIGN AND ANALYSIS OF ALGORITHMS

(23CSE211)

Name: BH. SUMITRANAND SHARMA

Roll No.: CH.SC.U4CSE24162

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:

```c
#include<stdio.h>
#include<stdlib.h>
void swap(int *a, int *b){
    int temp = *a;
    *a = *b;
    *b = temp;
}
int partitionFirst(int a[], int low, int high){
    int pivot = a[low];
    int i = low + 1, j = high;
    while(i <= j){
        while(i <= high && a[i] <= pivot){
            i++;
        }
        while(a[j] > pivot){
            j--;
        }
        if(i < j){
            swap(&a[i], &a[j]);
        }
    }
    swap(&a[low], &a[j]);
    return j;
}
```

```c
int partitionLast(int a[], int low, int high){
    int pivot = a[high];
    int i = low - 1;
    for(int j = low; j < high; j++){
        if(a[j] <= pivot){
            i++;
            swap(&a[i], &a[j]);
        }
    }
    swap(&a[i + 1], &a[high]);
    return i + 1;
}
int partitionRandom(int a[], int low, int high) {
    int randomIndex = low + rand() % (high - low + 1);
    printf("Randomly selected pivot: %d\n", a[randomIndex]);
    swap(&a[randomIndex], &a[high]);
    return partitionLast(a, low, high);
}

void quickSort(int a[], int low, int high, int choice){
    if(low < high){
        int p;
        if(choice == 1){
            p = partitionFirst(a, low, high);
        }
        else if(choice == 2){
            p = partitionLast(a, low, high);
        }
        else{
            p = partitionRandom(a, low, high);
        }
        quickSort(a, low, p - 1, choice);
        quickSort(a, p + 1, high, choice);
    }
}
void copyArray(int src[], int dest[], int n){
    for(int i = 0; i < n; i++){
        dest[i] = src[i];
    }
}
```

```c
void display(int a[], int n){
    for(int i = 0; i < n; i++){
        printf("%d ", a[i]);
    }
    printf("\n");
}
int main(){
    int n, choice;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    int original[n], temp[n];
    printf("Enter elements:\n");
    for(int i = 0; i < n; i++)
        scanf("%d", &original[i]);
    while(1){
        printf("\n----- QUICK SORT MENU -----\n");
        printf("1. First Element as Pivot\n");
        printf("2. Last Element as Pivot\n");
        printf("3. Random Element as Pivot\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        if(choice == 4){
            printf("Exiting program...\n");
            break;
        }
        if(choice < 1 || choice > 3){
            printf("Invalid choice! Try again.\n");
            continue;
        }
        copyArray(original, temp, n);
        printf("\nOriginal array:\n");
        display(temp, n);
        quickSort(temp, 0, n - 1, choice);
        printf("Sorted array:\n");
        display(temp, n);
    }
    return 0;
}
```

Output:

```
Enter number of elements: 12
Enter elements:
157 110 147 122 111 149 151 141 123 112 117 133

----- 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:
157 110 147 122 111 149 151 141 123 112 117 133
Sorted array:
110 111 112 117 122 123 133 141 147 149 151 157

----- 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:
157 110 147 122 111 149 151 141 123 112 117 133
Sorted array:
110 111 112 117 122 123 133 141 147 149 151 157

----- 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:
157 110 147 122 111 149 151 141 123 112 117 133
Randomly selected pivot: 149
Randomly selected pivot: 117
Randomly selected pivot: 111
Randomly selected pivot: 133
Randomly selected pivot: 122
Randomly selected pivot: 141
Randomly selected pivot: 151
Sorted array:
110 111 112 117 122 123 133 141 147 149 151 157

----- 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...
```

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.