Prarthana Thyagi-CH.SC.U4CSE24137

Design Analysis and Algorithm – Lab Work Week 5

WEEK - 5

QUICK SORT

1.First element as pivot

Code:

```c
#include <stdio.h>
void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pivot = arr[low];
        int i = low;
        int j = high;
        int temp;
        while (i < j) {
            while (arr[j] > pivot && i < j) j--;
            while (arr[i] <= pivot && i < j) i++;
            if (i < j) {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
        temp = arr[low];
        arr[low] = arr[i];
        arr[i] = temp;

        quickSort(arr, low, i - 1);
        quickSort(arr, i + 1, high);
    }
}
int main() {
    int data[] = {57,10,47,22,22,49,51,41,23,12,17,33};
    int n = 12;
    quickSort(data, 0, n - 1);
    for (int i = 0; i < n; i++) printf("%d ", data[i]);
    return 0;
}
```

Output:

```
10 12 17 22 22 23 33 41 47 49 51 57
--------------------------------
```

2. Last element as pivot

Prarthana Thyagi-CH.SC.U4CSE24137

Code:

```c
#include <stdio.h>
void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pivot = arr[high];
        int i = low;
        int temp;
        for (int j = low; j < high; j++) {
            if (arr[j] < pivot) {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
                i++;
            }
        }
        temp = arr[i];
        arr[i] = arr[high];
        arr[high] = temp;

        quickSort(arr, low, i - 1);
        quickSort(arr, i + 1, high);
    }
}
int main() {
    int data[] = {57,10,47,22,22,49,51,41,23,12,17,33};
    int n = 12;
    quickSort(data, 0, n - 1);
    for (int i = 0; i < n; i++) printf("%d ", data[i]);
    return 0;
}
```

Output:

```
10 12 17 22 22 23 33 41 47 49 51 57
--------------------------------
```

3.Ramdom Element as pivot

Code:

```c
#include <stdio.h>
void quickSort(int arr[], int low, int high) {
    int i = low, j = high;
    int pivot = arr[(low + high) / 2];
    int temp;
    while (i <= j) {
        while (arr[i] < pivot) i++;
        while (arr[j] > pivot) j--;
        if (i <= j) {
            temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
            i++;
            j--;
        }
    }
    if (low < j) quickSort(arr, low, j);
    if (i < high) quickSort(arr, i, high);
}
int main() {
    int data[] = {57,10,47,22,22,49,51,41,23,12,17,33};
    int n = 12;
    quickSort(data, 0, n - 1);
    for (int i = 0; i < n; i++) printf("%d ", data[i]);
    return 0;
}
```

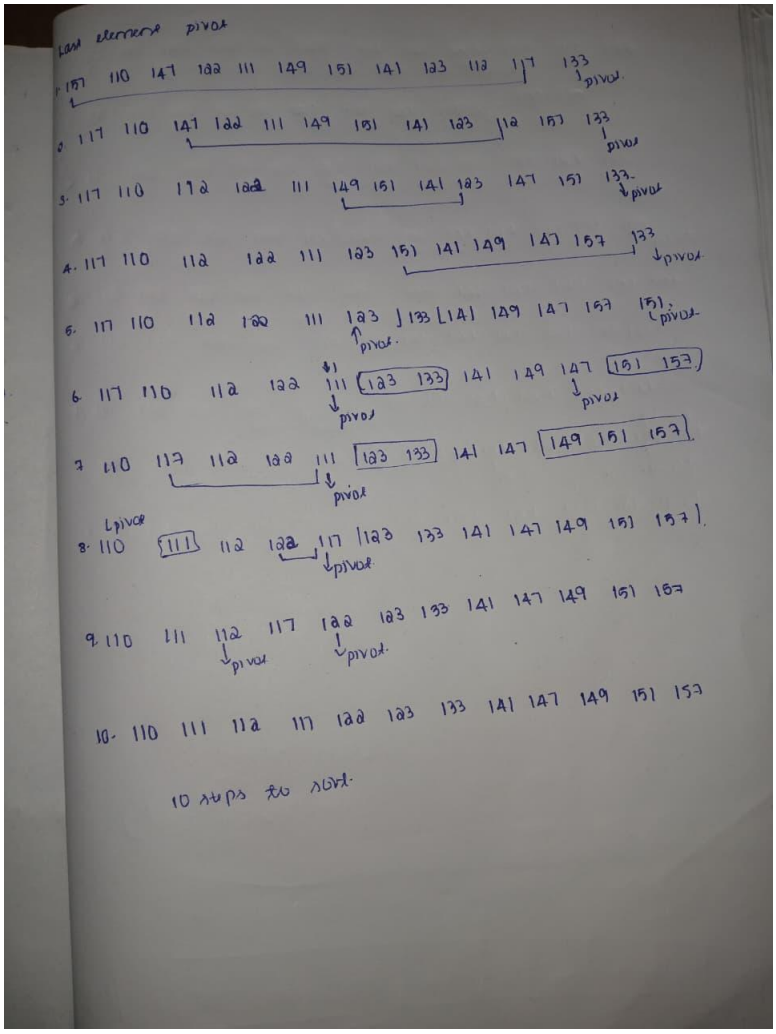
Output:

```
10 12 17 22 22 23 33 41 47 49 51 57
--------------------------------
```

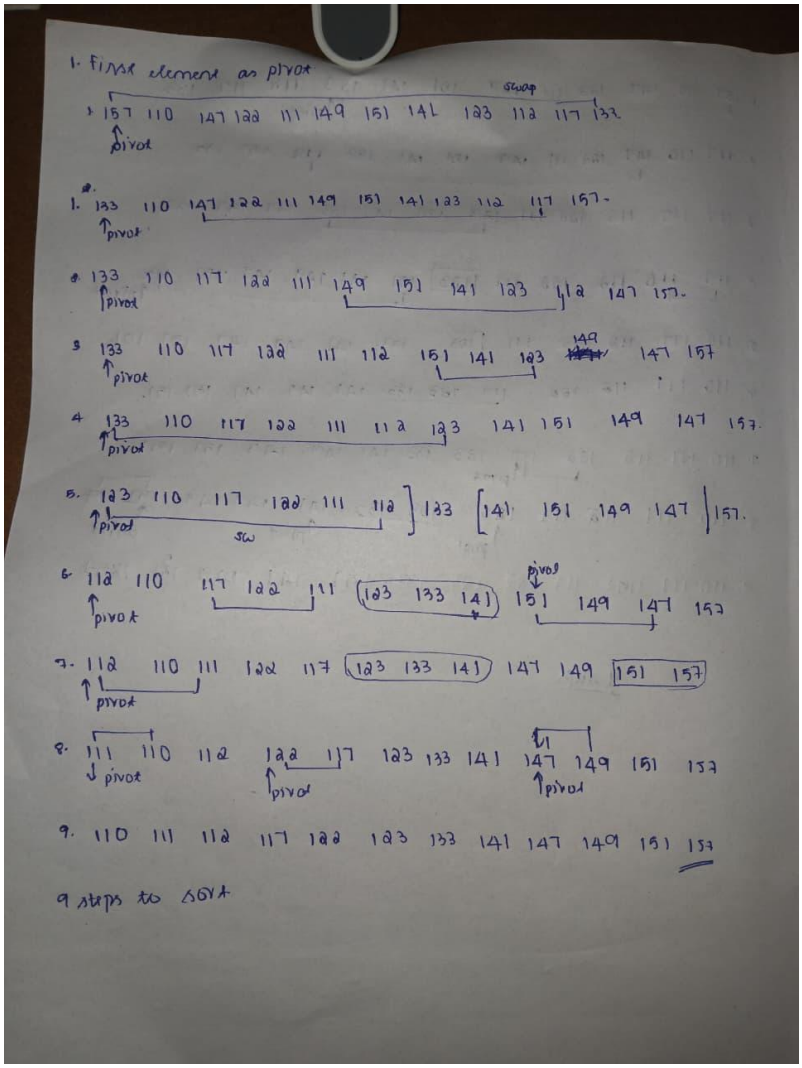
Space Complexity: O(n) -In the worst-case scenario (like a sorted list), the recursion depth can go as deep as the number of elements in the array.

Time Complexity : O(n^2)-The pivot is always the smallest or largest element (like in a sorted list). This creates very lopsided splits.

Prarthana Thyagi-CH.SC.U4CSE24137



Prarthana Thyagi-CH.SC.U4CSE24137



Prarthana Thyagi-CH.SC.U4CSE24137