

Name : DVN Surya Kiran
 Roll no : CH.SC.U4CSE24116
 DAA – Quick Sort Using 3 Conditions

I) First Element as Pivot

Working :

Use Quick Sort and sort this array by using

- i) First Element as a Pivot
- ii) Last Element as a Pivot.

Pivot ↓ l > p r < p l < p r > p l > r l < r

i) 157 110 147 122 111 149 151 141 123 112 117 133

SWAP

STEP - 1:

Pivot ↓ l > p r > p

133 110 147 122 111 149 151 141 123 112 117 157

STEP - 2:

Pivot ↓ l > p l < p l > p r > p

133 110 117 147 122 111 149 151 141 123 112 147 157

STEP - 3:

Pivot ↓ l < p l < p l < p l < p l > p r > p

133 110 117 122 111 112 151 141 123 149 147 157

STEP - 4:

Pivot ↓ l < p l < p l < p l < p l < p l > p r > p l > p r > p

133 110 117 122 111 112 123 141 151 149 147 157

SWAP

STEP - 5:

Pivot ↓ l < p l < p l < p l < p l < p l > p r > p

123 110 117 122 111 112 133 141 151 149 147 157

133 141 123 151 149 147 157

No Swap

STEP - 6:

Pivot ↓ l < p l > p r < p l > p r > p

112 110 117 122 111 123 133 141 151 149 147 157

STEP - 7:

Pivot ↓ l < p l < p l > p r < p

112 110 111 122 117 123 133 141 147 149 151 157

SWAP

STEP - 8:

Pivot ↓ l < p l > p r < p l > p r > p

111 110 112 122 117 123 133 141 147 149 151 157

110 111 112 117 122 123 133 141 147 149 151 157

SWAP

STEP - 9:

110 111 112 117 122 123 133 141 147 149 151 157

It takes 9 steps to completely sort the unsorted array.
 Using first element as pivot element

Program :

```
1 #include <stdio.h>
2
3 int partitionFirst(int a[], int low, int high) {
4     int pivot = a[low];
5     int i = low + 1, j = high, temp;
6     while (i <= j) {
7         while (i <= high && a[i] <= pivot)
8             i++;
9         while (a[j] > pivot)
10            j--;
11         if (i < j) {
12             temp = a[j];
13             a[j] = a[i];
14             a[i] = temp;
15         }
16     temp = a[low];
17     a[low] = a[j];
18     a[j] = temp;
19 }
20 } return j;
21 }
22 void quickSortFirst(int a[], int low, int high) {
23     if (low < high) {
24         int p = partitionFirst(a, low, high);
25         quickSortFirst(a, low, p - 1);
26         quickSortFirst(a, p + 1, high);
27     }
28 }
29 int main() {
30     int a[] = {10, 7, 8, 9, 1, 5};
31     int n = 6, i
32
33     quickSortFirst(a, 0, n - 1);
34
35     printf("Sorted array:\n")
36     for (i = 0; i < n, i++)
37         printf("%d ", a[i]);
38     }
39     printf("\nCH.SC.U4CSE24116\n");
```

Output :

Output

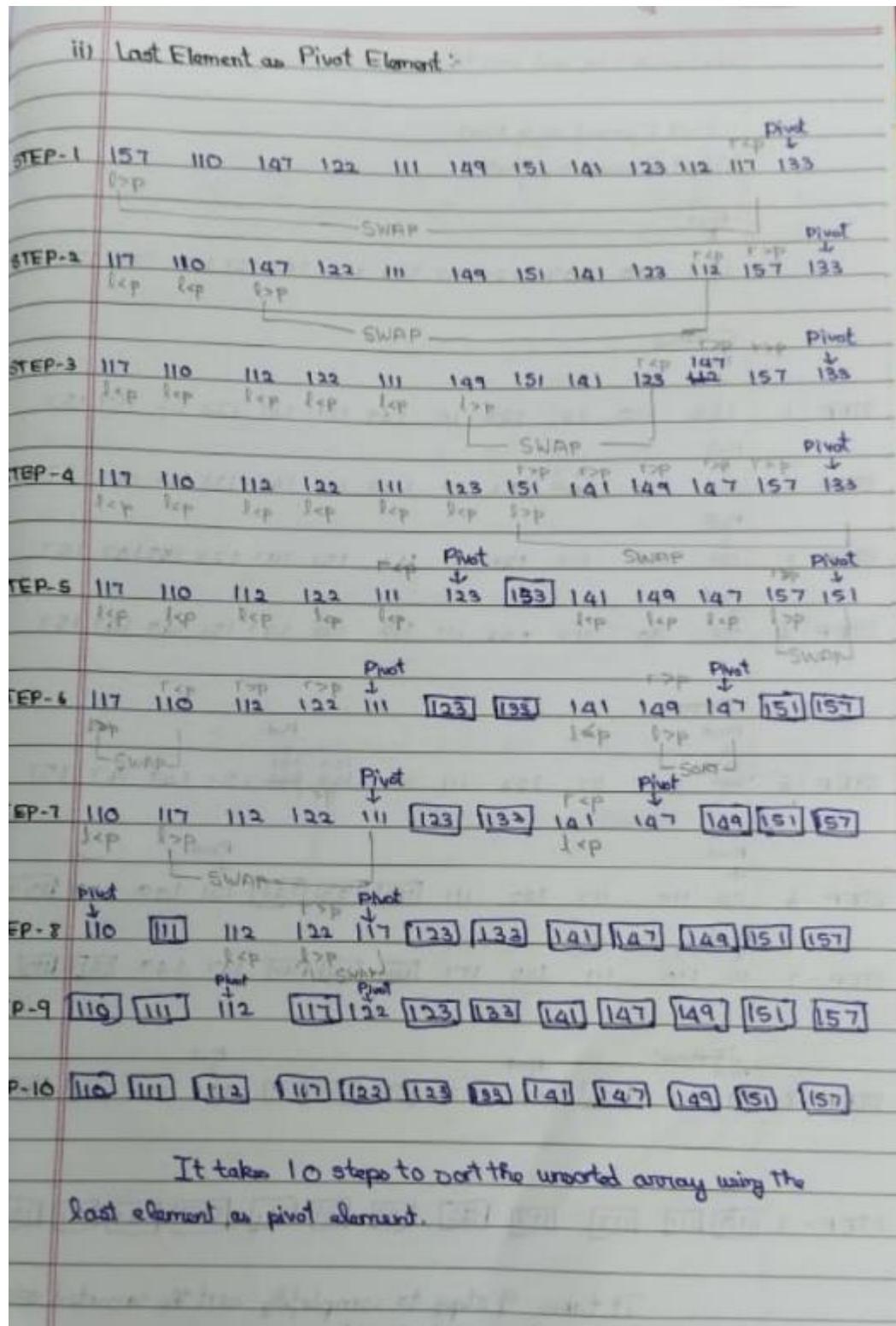
Sorted array:

1 5 7 8 9 10

CH.SC.U4CSE24116

II) Last Element as Pivot

Working :



Program :

```
2 * int partitionLast(int a[], int low, int high) {
3     int pivot = a[high];
4     int i = low - 1, j, temp;
5     for (j = low; j < high; j++) {
6         if (a[j] <= pivot) {
7             i++;
8             temp = a[j];
9             a[j] = a[i];
10            a[i] = temp;
11        }
12    }
13    temp = a[i + 1];
14    a[i + 1] = a[high];
15    a[high] = temp;
16
17    return i + 1;
18 }
19 void quickSortLast(int a[], int low, int high) {
20     if (low < high) {
21         int p = partitionLast(a, low, high);
22         quickSortLast(a, low, p - 1);
23         quickSortLast(a, p + 1, high);
24     }
25 }
26 int main() {
27     int a[] = {10, 7, 8, 9, 1, 5};
28     int n = 6, i;
29     quickSortLast(a, 0, n - 1);
30     printf("Sorted array:\n");
31     for (i = 0; i < n; i++)
32         printf("%d ", a[i]);
33     printf("\nCH.SC.U4CSE24116\n");
34
35     return 0;
}
```

Output :

Output

Sorted array:

1 5 7 8 9 10

CH.SC.U4CSE24116

III) Middle Element as Pivot

Program :

```
2  #include <stdio.h>
2  #include <stdlib.h>
3  int partitionRandom(int a[], int low, int high) {
4      int random = low + rand() % (high - low + 1);
5      int temp = a[random];
6      a[random] = a[high];
7      a[high] = temp;
8      int pivot = a[high];
9      for (j = low; j < high; j++) {
10         if (a[j] <= pivot) {
11             i++;
12             temp = a[i];
13             a[i] = a[j];
14             a[j] = temp;
15         }
16     }
17     temp = a[i + 1];
18     a[i + 1] = a[high];
19     a[high] = temp;
20     return i + 1;
21 }
22 void quickSortRandom(int a[], int low, int high) {
23     if (low < high) {
24         int p = partitionRandom(a, low, high);
25         quickSortRandom(a, low, p - 1);
26         quickSortRandom(a, p + 1, high);
27     }
28 }
30 int main() {
30     int a[] = {10, 7, 8, 9, 1, 5};
31     int n = 6, i;
32     quickSortRandom(a, 0, n - 1);
33     printf("Sorted array:\n")
34     for (i = 0, i < n, i++)
```

Output :

Output

Sorted array:

1 5 7 8 9 10

CH.SC.U4CSE24116