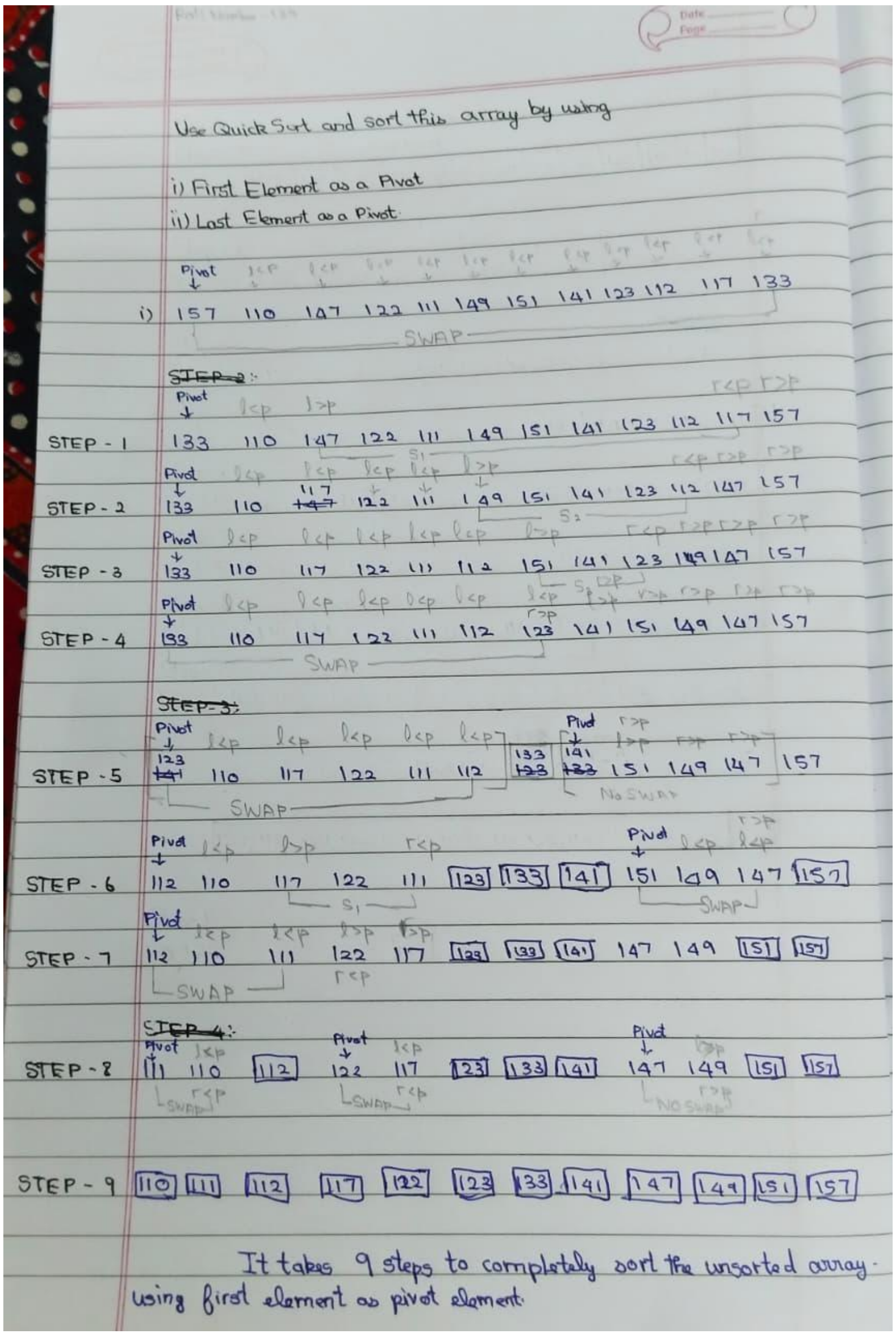
Name : DVN Surya Kiran
Roll no : CH.SC.U4CSE24116
DAA – Quick Sort Using 3 Conditions

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.

i) Pivot → 157 110 147 122 111 149 151 141 123 112 117 133 — SWAP

~~STEP-2~~:

STEP - 1: Pivot → 133 110 147 122 111 149 151 141 123 112 117 157 — S₁

STEP - 2: Pivot → 133 110 ~~147~~ 117 122 111 149 151 141 123 112 147 157 — S₂

STEP - 3: Pivot → 133 110 117 122 111 112 151 141 123 149 147 157 — S₃

STEP - 4: Pivot → 133 110 117 122 111 112 123 141 151 149 147 157 — SWAP

~~STEP-3~~:

STEP - 5: Pivot → ~~141~~ 123 110 117 122 111 112 | ~~123~~ 133 | Pivot → ~~133~~ 141 151 149 147 | 157 — SWAP — No SWAP

STEP - 6: Pivot → 112 110 117 122 111 $\boxed{123}$ $\boxed{133}$ $\boxed{141}$ Pivot → 151 149 147 $\boxed{157}$ — S₁ — SWAP

STEP - 7: Pivot → 112 110 111 122 117 $\boxed{123}$ $\boxed{133}$ $\boxed{141}$ 147 149 $\boxed{151}$ $\boxed{157}$ — SWAP

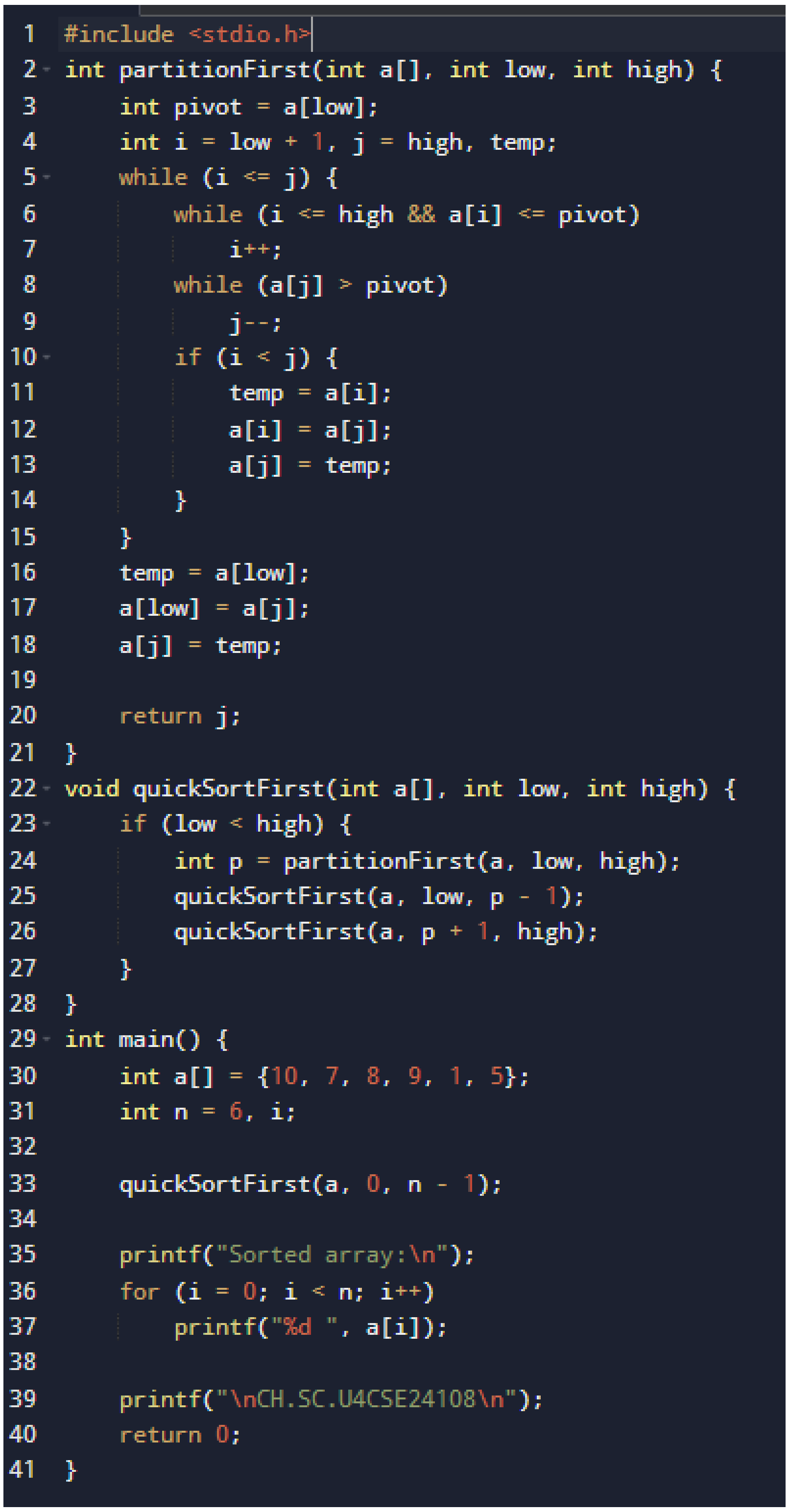
~~STEP-4~~:

STEP - 8: Pivot → 111 110 $\boxed{112}$ Pivot → 122 117 $\boxed{123}$ $\boxed{133}$ $\boxed{141}$ Pivot → 147 149 $\boxed{151}$ $\boxed{157}$ — SWAP — SWAP — NO SWAP

STEP - 9: $\boxed{110}$ $\boxed{111}$ $\boxed{112}$ $\boxed{117}$ $\boxed{122}$ $\boxed{123}$ $\boxed{133}$ $\boxed{141}$ $\boxed{147}$ $\boxed{149}$ $\boxed{151}$ $\boxed{157}$

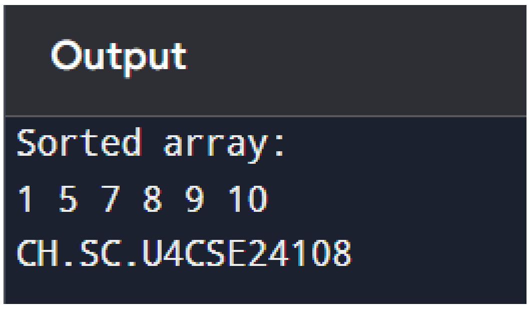
It takes 9 steps to completely sort the unsorted array. using first element as pivot element.

Program :



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

    return j;
}
void quickSortFirst(int a[], int low, int high) {
    if (low < high) {
        int p = partitionFirst(a, low, high);
        quickSortFirst(a, low, p - 1);
        quickSortFirst(a, p + 1, high);
    }
}
int main() {
    int a[] = {10, 7, 8, 9, 1, 5};
    int n = 6, i;

    quickSortFirst(a, 0, n - 1);

    printf("Sorted array:\n");
    for (i = 0; i < n; i++)
        printf("%d ", a[i]);

    printf("\nCH.SC.U4CSE24108\n");
    return 0;
}
```

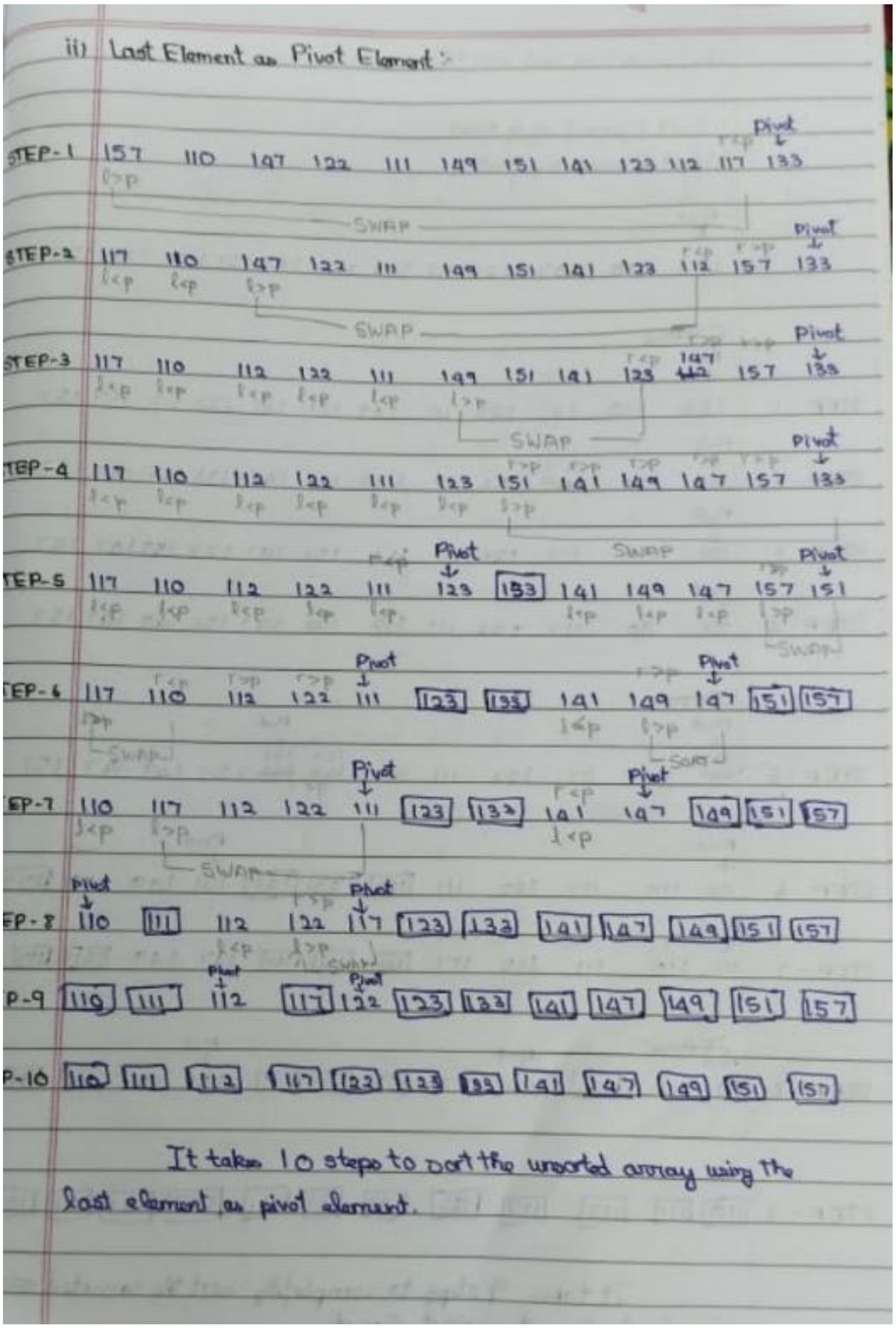
Output :



```
Sorted array:
1 5 7 8 9 10
CH.SC.U4CSE24108
```

II) Last Element as Pivot

Working :



ii) Last Element as Pivot Element :-

STEP-1 157 110 147 122 111 149 151 141 123 112 117 133 (Pivot: 133; l>p under 157, r<p over 117) — SWAP

STEP-2 117 110 147 122 111 149 151 141 123 112 157 133 (Pivot: 133; l<p, l<p, l>p; r<p over 112) — SWAP

STEP-3 117 110 112 122 111 149 151 141 123 ~~112~~ 147 157 133 (Pivot: 133; l<p l<p l<p l<p l<p l>p; r<p over 123) — SWAP

TEP-4 117 110 112 122 111 123 151 141 149 147 157 133 (Pivot: 133; l<p l<p l<p l<p l<p l<p l>p) — SWAP

TEP-5 117 110 112 122 111 123 $\boxed{133}$ 141 149 147 157 151 (Pivot: 123; Pivot: 151; l>p under 157) — SWAP

TEP-6 117 110 112 122 111 $\boxed{123}$ $\boxed{133}$ 141 149 147 $\boxed{151}$ $\boxed{157}$ (Pivot: 111; Pivot: 147; l>p under 117; l<p under 141, l>p under 149) — SWAP — SORT

EP-7 110 117 112 122 111 $\boxed{123}$ $\boxed{133}$ 141 147 $\boxed{149}$ $\boxed{151}$ $\boxed{157}$ (Pivot: 111; Pivot: 147; l<p, l>p; l<p under 141) — SWAP

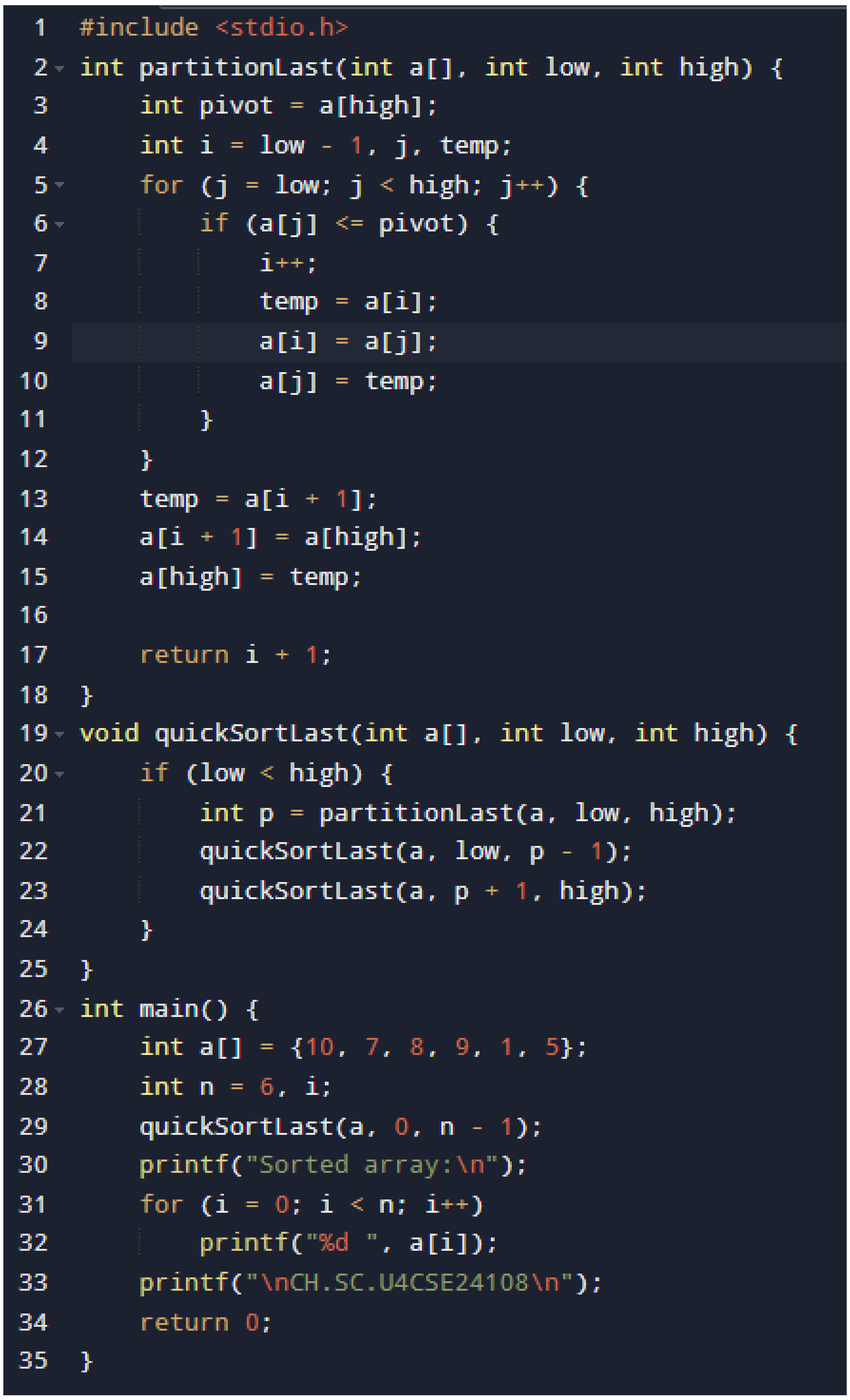
EP-8 110 $\boxed{111}$ 112 122 117 $\boxed{123}$ $\boxed{133}$ $\boxed{141}$ $\boxed{147}$ $\boxed{149}$ $\boxed{151}$ $\boxed{157}$ (Pivot: 110; Pivot: 117; l<p under 112, l>p under 122) — SWAP

P-9 $\boxed{110}$ $\boxed{111}$ 112 $\boxed{117}$ 122 $\boxed{123}$ $\boxed{133}$ $\boxed{141}$ $\boxed{147}$ $\boxed{149}$ $\boxed{151}$ $\boxed{157}$ (Pivot: 112; Pivot: 122)

P-10 $\boxed{110}$ $\boxed{111}$ $\boxed{112}$ $\boxed{117}$ $\boxed{122}$ $\boxed{123}$ $\boxed{133}$ $\boxed{141}$ $\boxed{147}$ $\boxed{149}$ $\boxed{151}$ $\boxed{157}$

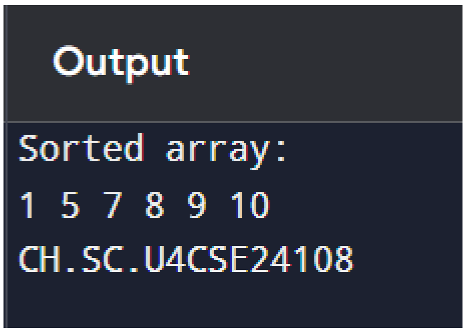
It takes 10 steps to sort the unsorted array using the last element as pivot element.

Program :



```c
#include <stdio.h>
int partitionLast(int a[], int low, int high) {
    int pivot = a[high];
    int i = low - 1, j, temp;
    for (j = low; j < high; j++) {
        if (a[j] <= pivot) {
            i++;
            temp = a[i];
            a[i] = a[j];
            a[j] = temp;
        }
    }
    temp = a[i + 1];
    a[i + 1] = a[high];
    a[high] = temp;

    return i + 1;
}
void quickSortLast(int a[], int low, int high) {
    if (low < high) {
        int p = partitionLast(a, low, high);
        quickSortLast(a, low, p - 1);
        quickSortLast(a, p + 1, high);
    }
}
int main() {
    int a[] = {10, 7, 8, 9, 1, 5};
    int n = 6, i;
    quickSortLast(a, 0, n - 1);
    printf("Sorted array:\n");
    for (i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\nCH.SC.U4CSE24108\n");
    return 0;
}
```

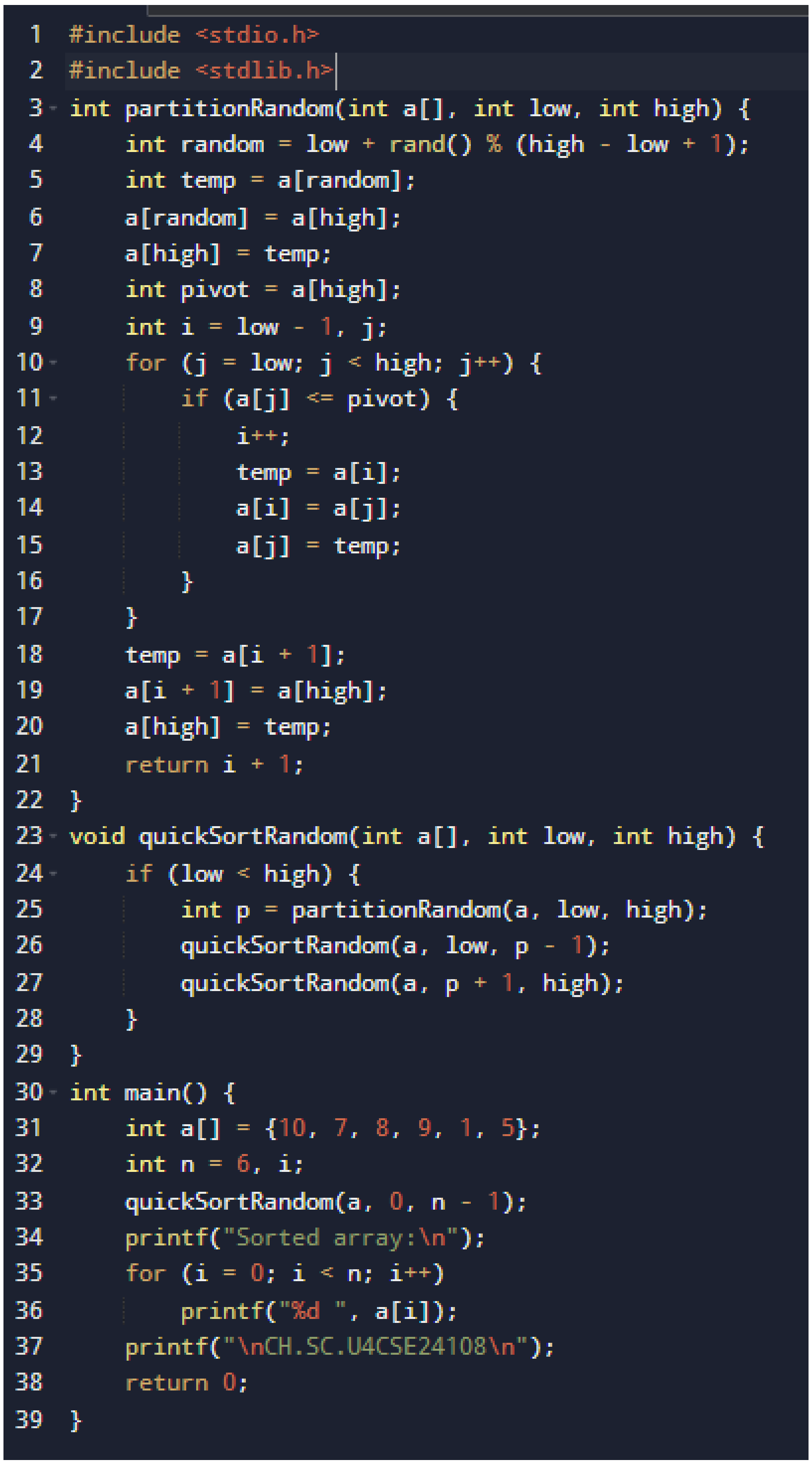
Output :



```
Sorted array:
1 5 7 8 9 10
CH.SC.U4CSE24108
```

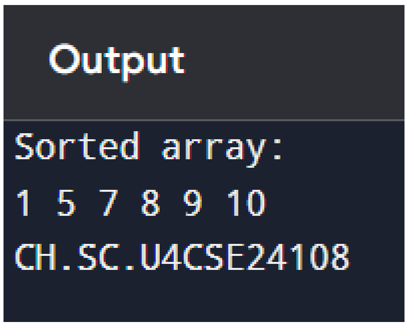
III) Middle Element as Pivot

Program :



```c
#include <stdio.h>
#include <stdlib.h>
int partitionRandom(int a[], int low, int high) {
    int random = low + rand() % (high - low + 1);
    int temp = a[random];
    a[random] = a[high];
    a[high] = temp;
    int pivot = a[high];
    int i = low - 1, j;
    for (j = low; j < high; j++) {
        if (a[j] <= pivot) {
            i++;
            temp = a[i];
            a[i] = a[j];
            a[j] = temp;
        }
    }
    temp = a[i + 1];
    a[i + 1] = a[high];
    a[high] = temp;
    return i + 1;
}
void quickSortRandom(int a[], int low, int high) {
    if (low < high) {
        int p = partitionRandom(a, low, high);
        quickSortRandom(a, low, p - 1);
        quickSortRandom(a, p + 1, high);
    }
}
int main() {
    int a[] = {10, 7, 8, 9, 1, 5};
    int n = 6, i;
    quickSortRandom(a, 0, n - 1);
    printf("Sorted array:\n");
    for (i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\nCH.SC.U4CSE24108\n");
    return 0;
}
```

Output :



```
Sorted array:
1 5 7 8 9 10
CH.SC.U4CSE24108
```