WEEK-5

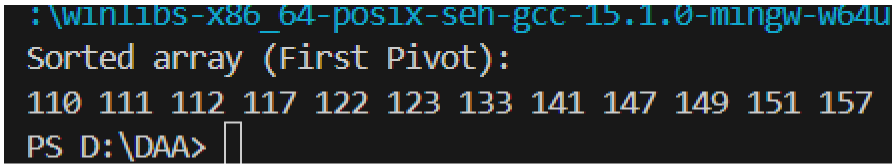
QUICK SORT

1. FIRST ELEMENT AS PIVOT ELEMENT

Code:

```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[] = {157,110,147,122,111,149,151,141,123,112,117,133};
    int n = 12;
    quickSortFirst(a, 0, n - 1);

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

    return 0;
}
```

Output:



```
:\winlibs-x86_64-posix-seh-gcc-15.1.0-mingw-w64u
Sorted array (First Pivot):
110 111 112 117 122 123 133 141 147 149 151 157
PS D:\DAA>
```

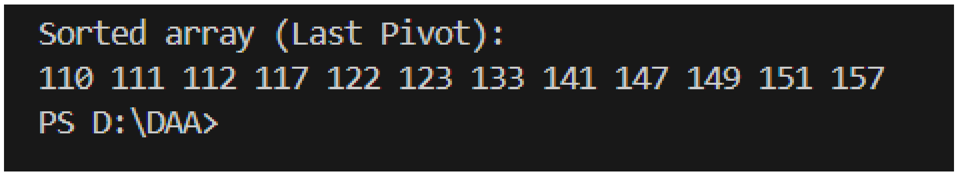
2. LAST ELEMENT AS PIVOT

CODE:

```c
#include <stdio.h>

int partitionLast(int a[], int low, int high) {
    int pivot = a[high];
    int i = low - 1, temp;

    for (int 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;
}
```

```c
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() {
    printf("Harish.r CH.SC.U4cse24163\n");

    int a[] = {157,110,147,122,111,149,151,141,123,112,117,133};
    int n = 12;

    quickSortLast(a, 0, n - 1);

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

    return 0;
}
```

Output:



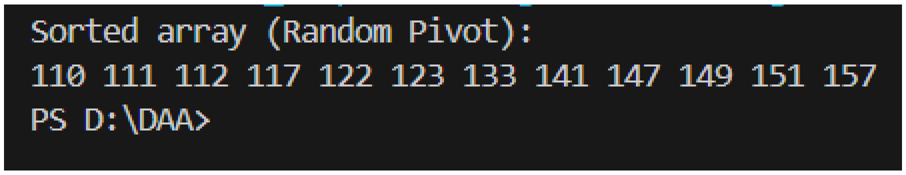
```
Sorted array (Last Pivot):
110 111 112 117 122 123 133 141 147 149 151 157
PS D:\DAA>
```

3. RANDOM ELEMENT AS PIVOT

CODE:

```c
#include <stdio.h>
#include <stdlib.h>

int partitionRandom(int a[], int low, int high) {
    int randomIndex = low + rand() % (high - low + 1);
    int temp = a[randomIndex];
    a[randomIndex] = a[high];
    a[high] = temp;

    int pivot = a[high];
    int i = low - 1;

    for (int 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[] = {157,110,147,122,111,149,151,141,123,112,117,133};
    int n = 12;

    quickSortRandom(a, 0, n - 1);

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

    return 0;
}
```

Output:



```
Sorted array (Random Pivot):
110 111 112 117 122 123 133 141 147 149 151 157
PS D:\DAA>
```