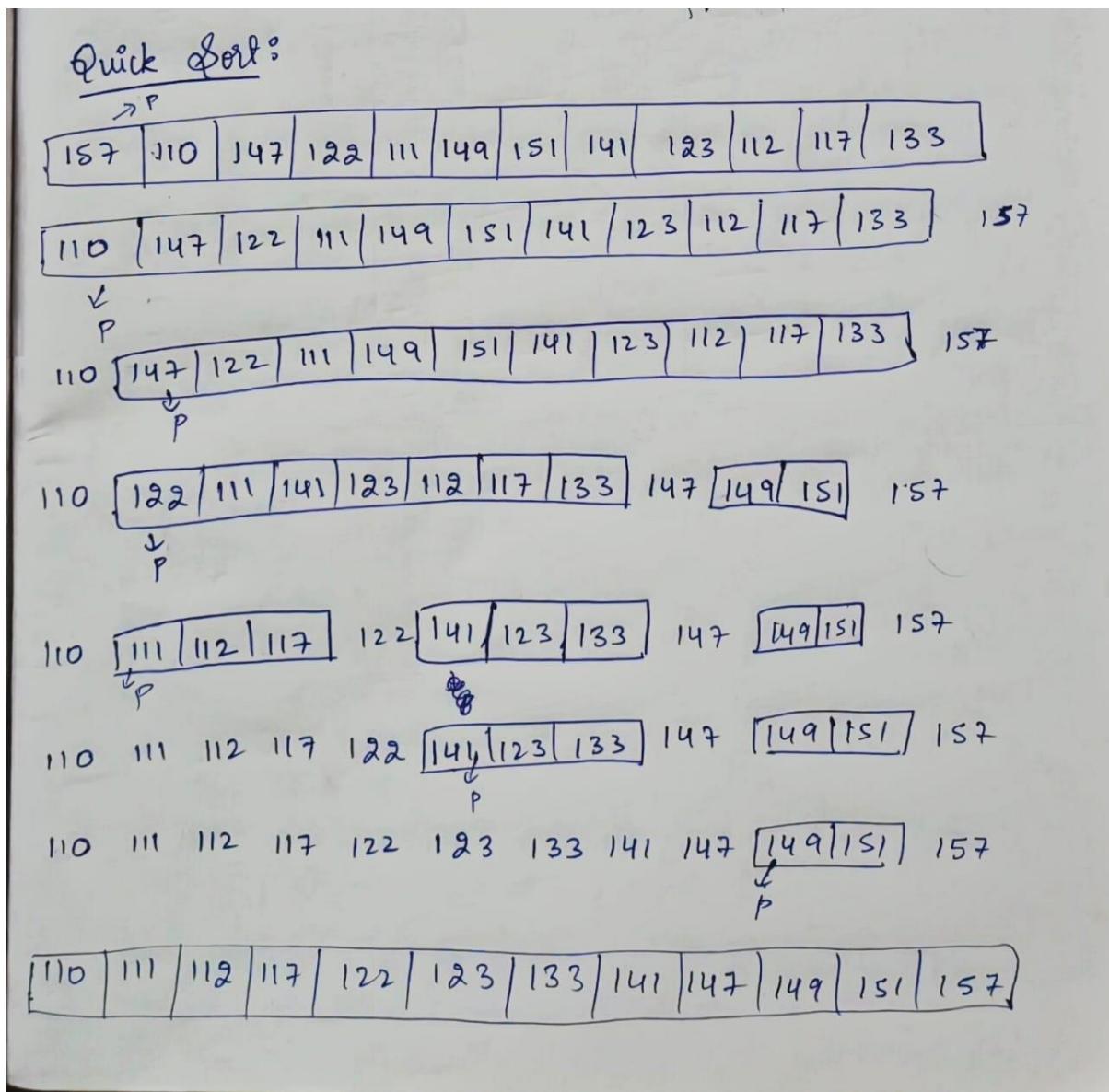


NAME : Mahalakshmi K

ROLL NO : CH.SC.U4CSE24160

WEEK - 4

Quick Sort :



Logic :

```
void quickSort(int a[], int low, int high){  
    if (low < high) {  
        int p = partition(a, low, high);  
        quickSort(a, low, p - 1);  
        quickSort(a, p + 1, high);  
    }  
}  
  
int partition(int a[], int low, int high){  
    int pivot = a[low];  
    int i = low + 1;  
    int j = high;  
    int 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;  
}
```

Code :

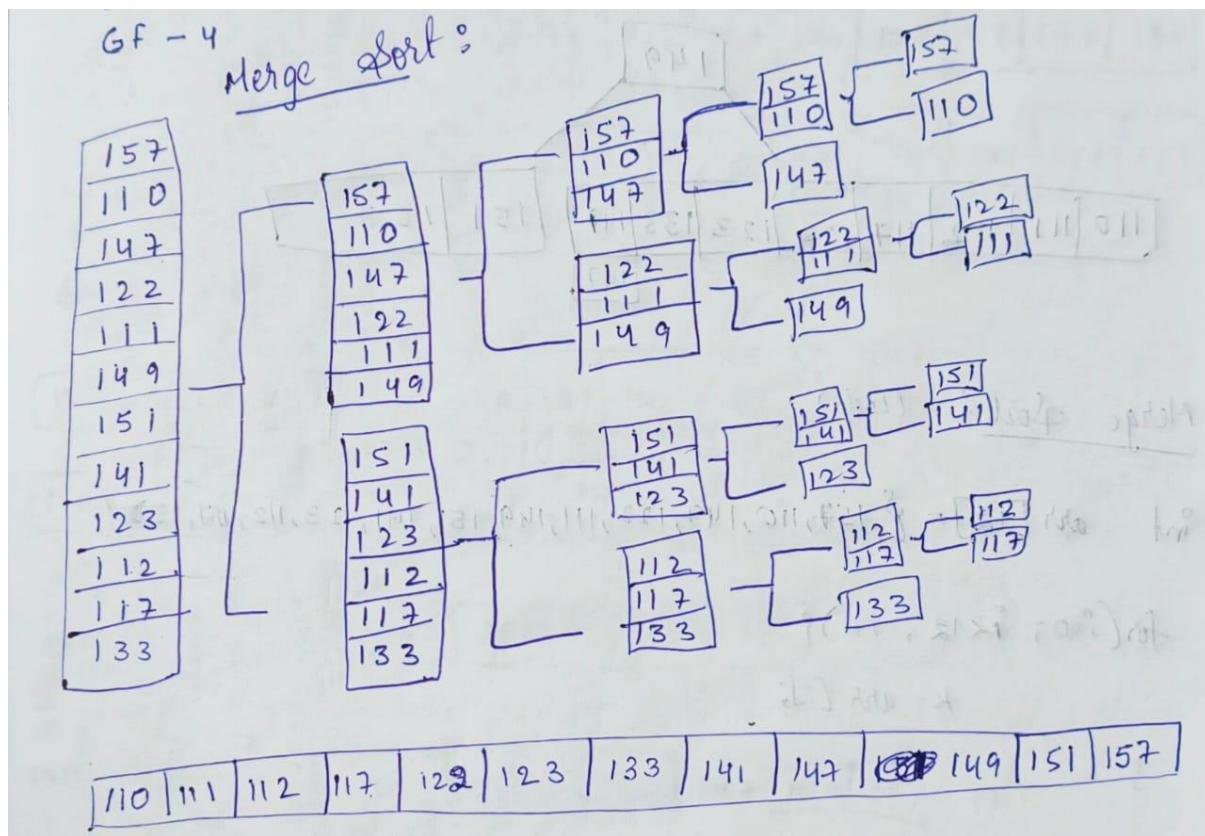
```
1 #include <stdio.h>
2 int partition(int a[], int low, int high)
3 {
4     int pivot = a[low];
5     int i = low + 1;
6     int j = high;
7     int temp;
8     while (i <= j)
9     {
10         while (i <= high && a[i] <= pivot)
11             i++;
12         while (a[j] > pivot)
13             j--;
14         if (i < j)
15         {
16             temp = a[i];
17             a[i] = a[j];
18             a[j] = temp;
19         }
20     }
21     temp = a[low];
22     a[low] = a[j];
23     a[j] = temp;
24
25     return j;
26 }
27 void quickSort(int a[], int low, int high)
28 {
29     if (low < high)
30     {
31         int p = partition(a, low, high);
32         quickSort(a, low, p - 1);
33         quickSort(a, p + 1, high);
34     }
35 }
36 int main()
37 {
38     int a[] = {157,110,147,122,111,149,151,141,123,112,117,133};
39     int n = sizeof(a) / sizeof(a[0]);
40     int i;
41     quickSort(a, 0, n - 1);
42     printf("Sorted array:\n");
43     for (i = 0; i < n; i++)
44         printf("%d ", a[i]);
45
46     return 0;
47 }
```

Output :

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

```
-----  
Process exited after 0.1601 seconds with return value 0  
Press any key to continue . . . |
```

Merge Sort :



Logic :

```
void merge(int a[], int low, int mid, int high){
```

```
    int i = low;
```

```
    int j = mid + 1;
```

```
    int k = low;
```

```
    int temp[100];
```

```
while (i <= mid && j <= high) {  
    if (a[i] <= a[j])  
        temp[k++] = a[i++];  
    else  
        temp[k++] = a[j++];  
}  
  
while (i <= mid)  
    temp[k++] = a[i++];  
  
while (j <= high)  
    temp[k++] = a[j++];  
  
for (i = low; i <= high; i++)  
    a[i] = temp[i];  
}
```

```
void mergeSort(int a[], int low, int high){  
    if (low < high)  
    {  
        int mid = (low + high) / 2;  
        mergeSort(a, low, mid);  
        mergeSort(a, mid + 1, high);  
        merge(a, low, mid, high);  
    }  
}
```

Code :

```

1 #include <stdio.h>
2 void merge(int a[], int low, int mid, int high)
3 {
4     int i = low;
5     int j = mid + 1;
6     int k = low;
7     int temp[100];
8     while (i <= mid && j <= high)
9     {
10         if (a[i] <= a[j])
11             temp[k++] = a[i++];
12         else
13             temp[k++] = a[j++];
14     }
15     while (i <= mid)
16         temp[k++] = a[i++];
17
18     while (j <= high)
19         temp[k++] = a[j++];
20
21     for (i = low; i <= high; i++)
22         a[i] = temp[i];
23 }
24 void mergeSort(int a[], int low, int high)
25 {
26     if (low < high)
27     {
28         int mid = (low + high) / 2;
29         mergeSort(a, low, mid);
30         mergeSort(a, mid + 1, high);
31         merge(a, low, mid, high);
32     }
33 }
34 int main()
35 {
36     int a[] = {157, 110, 147, 122, 111, 149, 151, 141, 123, 112, 117, 133};
37     int n = sizeof(a) / sizeof(a[0]);
38     int i;
39     mergeSort(a, 0, n - 1);
40     printf("Sorted array:\n");
41     for (i = 0; i < n; i++)
42         printf("%d ", a[i]);
43     return 0;
44 }

```

Output :

```

Sorted array:
110 111 112 117 122 123 133 141 147 149 151 157
-----
Process exited after 0.08793 seconds with return value 0
Press any key to continue . . .

```