```
[12/15, 9:35 /AM]  R a g h uØ>#iàØ>clude <stdio.h>

// Function to swap two elements
void swap(int a, int b) {
int temp = *a;
a = b;
*b = temp;
}

// Partition function
int partition(int arr[], int low, int high) {
int pivot = arr[high];   // pivot element
int i = low - 1;

for (int j = low; j < high; j++) {
if (arr[j] <= pivot) {
i++;
swap(&arr[i], &arr[j]);
}
}
swap(&arr[i + 1], &arr[high]);
return i + 1;
}

// Quick Sort function
void quickSort(int arr[], int low, int high) {
if (low < high) {
int pi = partition(arr, low, high);

quickSort(arr, low, pi - 1);
quickSort(arr, pi + 1, high);
}
}

// Main function
int main() {
int arr[] = {10, 7, 8, 9, 1, 5};
int n = sizeof(arr) / sizeof(arr[0]);

quickSort(arr, 0, n - 1);

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

return 0;
}
#include <stdio.h>

void merge(int a[], int l, int m, int r)
{
int i = l, j = m + 1, k = l;
int temp[100];

while (i <= m && j <= r)
{
if (a[i] <= a[j])
temp[k++] = a[i++];
else
temp[k++] = a[j++];
}

while (i <= m)
temp[k++] = a[i++];

while (j <= r)
temp[k++] = a[j++];

for (i = l; i <= r; i++)
a[i] = temp[i];
}

void mergeSort(int a[], int l, int r)
{
if (l < r)
{
int m = (l + r) / 2;
mergeSort(a, l, m);
mergeSort(a, m + 1, r);
merge(a, l, m, r);
}
}

int main()
```

```
{
int a[100], n, i;

printf("Enter number of elements: ");
scanf("%d", &n);

printf("Enter elements:\n");
for (i = 0; i < n; i++)
scanf("%d", &a[i]);

mergeSort(a, 0, n - 1);

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

return 0;
}
```