//Ayush sahu

//cse 11911058

// Merge sort

#include <stdio.h>

// Merge two subarrays L and M into arrays

void merge(int arr[], int p, int q, int r) {

// Create L and M

int n1 = q - p + 1;

int n2 = r - q;

int L[n1], M[n2];

for (int i = 0; i < n1; i++) //assigning elements to sub arrays

L[i] = arr[p + i];

for (int j = 0; j < n2; j++)//assigning elements to sub arrays

M[j] = arr[q + 1 + j];

int i, j, k;

i = 0;

j = 0;

k = p;

// Until we reach either end of either L or M, pick larger among

while (i < n1 && j < n2) {

if (L[i] <= M[j]) {

arr[k] = L[i];

i++;

} else {

arr[k] = M[j];

j++;

}

k++;

}

// pick up the remaining elements

while (i < n1) {

arr[k] = L[i];

i++;

k++;

}

while (j < n2) {

arr[k] = M[j];

j++;

k++;

}

}

// main merge sort

void mergeSort(int arr[], int l, int r) {

if (l < r) {

int m = l + (r - l) / 2;

mergeSort(arr, l, m);

mergeSort(arr, m + 1, r);

merge(arr, l, m, r);//merging arrays

} }

void printArray(int arr[], int size) {

for (int i = 0; i < size; i++)

printf("%d ", arr[i]);

printf("\n"); }

// Driver program

int main() {

int arr[] = {10,12,6,5,8,1};

int size = sizeof(arr) / sizeof(arr[0]);

mergeSort(arr, 0, size - 1);

printf("Sorted array: \n");

printArray(arr, size); }