```
//Heap sort
//Om Dattatray Gavande
//Class:-SY CSE A
//Roll No:-CS2145


#include <stdio.h>


void heapify(int arr[], int n, int i) {
  int largest = i;      // Initialize largest as root
  int l = 2 * i + 1;    // left child index
  int r = 2 * i + 2;    // right child index


  // If left child is larger than root
  if (l < n && arr[l] > arr[largest])
     largest = l;


  // If right child is larger than current largest
  if (r < n && arr[r] > arr[largest])
     largest = r;


  // If largest is not root
  if (largest != i) {
     int temp = arr[i];
     arr[i] = arr[largest];
     arr[largest] = temp;


     // Recursively heapify the affected subtree
     heapify(arr, n, largest);
  }
}


// Main function to perform heap sort
void heapSort(int arr[], int n) {
  // Build max heap
  for (int i = n/2 - 1; i >= 0; i--)
     heapify(arr, n, i);


  // Extract elements from heap one by one
  for (int i = n - 1; i > 0; i--) {
     // Move current root to end
     int temp = arr[0];
     arr[0] = arr[i];
     arr[i] = temp;
```

```
    // Call heapify on the reduced heap
    heapify(arr, i, 0);
  }
}


int main() {
  int arr[] = {9, 4, 3, 8, 10, 2, 5};
  int n = sizeof(arr) / sizeof(arr[0]);


  heapSort(arr, n);


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


  printf("\n");
  return 0;
}
```