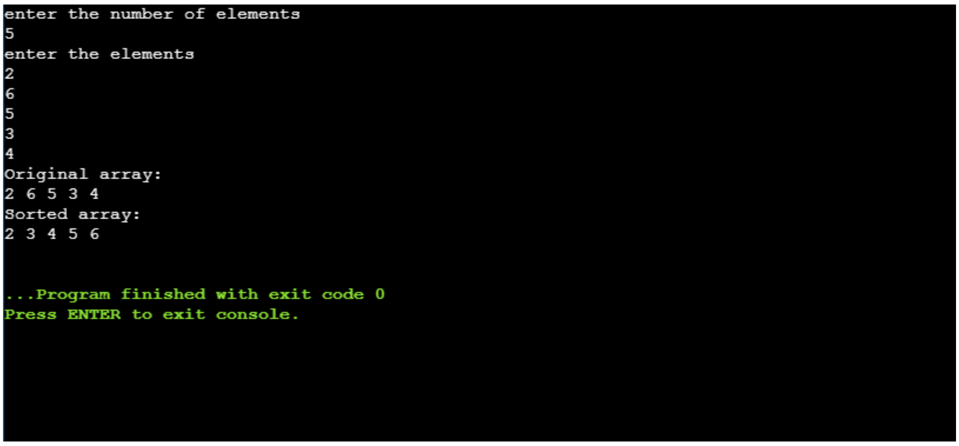
Code:

```
#include <stdio.h>

  void swap(int *a, int *b) {
    int tmp = *a;
    *a = *b;
    *b = tmp;
  }

  void heapify(int arr[], int n, int i) {
    int max = i; //Initialize max as root
    int leftChild = 2 * i + 1;
    int rightChild = 2 * i + 2;

    //If left child is greater than root
    if (leftChild < n && arr[leftChild] > arr[max])
      max = leftChild;

    //If right child is greater than max
    if (rightChild < n && arr[rightChild] > arr[max])
      max = rightChild;

    //If max is not root
    if (max != i) {
      swap(&arr[i], &arr[max]);
      //heapify the affected sub-tree recursively
      heapify(arr, n, max);
  }
}

// to perform heap sort
void heapSort(int arr[], int n) {
  //Rearrange array (building 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--) {
    swap(&arr[0], &arr[i]); //Current root moved to the end

    heapify(arr, i, 0); //calling max heapify on the heap reduced
  }
}

//print size of array
void display(int arr[], int n) {
  for (int i = 0; i < n; ++i)
    printf("%d ", arr[i]);
  printf("\n");
}


int main() {
  int arr[20],n;
  printf("enter the number of elements\n");

  scanf("%d",&n);

  printf("enter the elements\n");

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

  scanf("%d",&arr[i]);



  printf("Original array:\n");

  display(arr,n);

  heapSort(arr,n);


  printf("Sorted array:\n");

  display(arr,n);

 }
```

Output:



```
enter the number of elements
5
enter the elements
2
6
5
3
4
Original array:
2 6 5 3 4
Sorted array:
2 3 4 5 6


...Program finished with exit code 0
Press ENTER to exit console.
```