SEM 3\Exp9\Heap_Sort.c

```c
// Function to heapify a subtree rooted at index i
void heapify(int arr[], int size, int i)
{
    int largest = i;       // Initialize largest as root
    int left = 2 * i + 1;  // left = 2*i + 1
    int right = 2 * i + 2; // right = 2*i + 2

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

    // If right child is larger than largest so far
    if (right < size && arr[right] > arr[largest])
        largest = right;

    // 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, size, largest);
    }
}

// Function to perform heap sort
void heapSort(int arr[], int size)
{
    // Build heap (rearrange array)
    for (int i = size / 2 - 1; i >= 0; i--)
        heapify(arr, size, i);

    // One by one extract an element from heap
    for (int i = size - 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);
    }
}

```