```
//Implement a priority queue using Heap
#include <stdio.h>
#define MAX 100
#define MIN -10000

struct Heap
{
    int A[MAX];
    int heap_size;
};

void swap(int &a, int &b)
{
    int t = a;
    a = b;
    b = t;
}

void max_heapify(Heap *heap, int index)
{
    int left = 2*index+1, right = 2*index+2, largest=index;
    if (left <= heap->heap_size && heap->A[left] > heap->A[index])
    {
        largest = left;
    }
    if (right <= heap->heap_size && heap->A[right] > heap->A[index])
    {
        largest = right;
    }
    if(left <= heap->heap_size && right <= heap->heap_size)
    {
        if(heap->A[left] > heap->A[right] && heap->A[left] > heap->A[index])
            largest = left;
        else if(heap->A[right] > heap->A[left] && heap->A[right] > heap->A[index])
            largest = right;
    }
    if (largest != index)
    {
        swap(heap->A[index], heap->A[largest]);
        max_heapify(heap, largest);
    }
}

void build_maxheap(Heap *heap)
{
    for(int i=heap->heap_size/2 +1 ; i>=0 ; i--)
        max_heapify(heap ,i);
}

int heap_maximum(Heap heap)
{
    return heap.A[0];
}

int extract_maximum(Heap *heap)
{
    if(heap->heap_size < 1)
    {
        printf("Heap Underflow\n");
        return -1;
    }
    int maxa = heap->A[0];
    heap->A[0] = heap->A[heap->heap_size];
    heap->heap_size--;
    max_heapify(heap, 0);
    return maxa;
}

void increase_key(Heap *heap, int index, int key)
```

```
{
    if (key < heap->A[index])
    {
        printf("Error\n");
        return;
    }
    heap->A[index] = key;
    while(index > 1 && heap->A[(index-1)/2] < heap->A[index])
    {
        swap(heap->A[index], heap->A[(index-1)/2]);
        index = (index-1)/2;
    }
}

void max_heap_insert(Heap *heap, int key)
{
    heap->heap_size++;
    heap->A[heap->heap_size] = MIN;
    increase_key(heap, heap->heap_size, key);
}

int main()
{
    Heap heap;
    for(int i=0 ; i<5 ; i++)
        heap.A[i] = i+1;
    heap.heap_size = 4;
    build_maxheap(&heap);
    printf("Maximum element in heap : %d\n", heap_maximum(heap));
    printf("After inserting 6 in heap\n");
    max_heap_insert(&heap, 6);
    printf("Maximum element in heap : %d\n",heap_maximum(heap));
}
/*
OUTPUT
Maximum element in heap : 5
After inserting 6 in heap
Maximum element in heap : 6
*/
```