/*decrease key operation*/

```
#include <stdio.h>


int n = 0;


void decrease_key(int arr[], int id, int key);
void max_heapify(int arr[],int id);


int main()
{
   int i, arr[20], id=0, key=0; //taking an empty array of size 20
   printf("Enter the array size (the array is an array representation of a heap): "); //so, heap size = arr size
   scanf("%d", &n);
   printf("Enter the array elements:\n");
   for (i = 0; i < n; i++)
           {
      scanf("%d", &arr[i]);
   }
   printf("the array (heap) is: ");
   for (i = 0; i < n; i++)
           {
      printf("%d ", arr[i]);
   }
   printf("\n");
   printf("enter the index of the element to be decreased: ");
   scanf("%d", &id);
   printf("\nenter the key (decreased val): ");
   scanf("%d", &key);
   printf("\n");
   decrease_key(arr, id, key);
```

```
    printf("the heap after decrease key operation: ");
    for (i = 0; i < n; i++)
            {
        printf("%d ", arr[i]);
    }
    return 0;
}

void decrease_key(int arr[], int id, int key)
 {
    if (arr[id] < key)
            {
        printf("ERROR: node value already lesser than key");
        return;
    }
    arr[id] = key;
    max_heapify(arr,id);
}

void max_heapify(int arr[], int i)
{
    int lc, rc, largest;
    lc = 2 * i + 1;
    rc = 2 * i + 2;
    if (lc < n && arr[lc] > arr[i])
            {  //here arr.heapsize=n=arr.length as heap size=arr size
        largest = lc;
    } else {
        largest = i;
    }
    if (rc < n && arr[rc] > arr[largest])
```

```
        { //arr[rc] is larger then the largest element determined in just the prv if loop,ie arr[rc] is largest among 3 nodes(if have 3 nodes)
            largest = rc;
        }
        if (largest != i)
            { //swapp arr[i] with arr[largest]
            int temp = arr[i];
            arr[i] = arr[largest];
            arr[largest] = temp;
            max_heapify(arr,largest);
        }
}
```