//increase key operation(by creating max heap)

```
#include<stdio.h>
int n=0;

void create_max_heap(int arr[]);
void increase_key(int arr[],int id,int key);
void max_heapify(int arr[],int i);

int main(){
	int arr[20],i,id,key;
	printf("enter the size of array(heap): ");
	scanf("%d",&n);
	printf("\nenter the array eles\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]);
	}
	create_max_heap(arr);
	printf("\nafter max heapify the array(heap) is: ");
	for(i=0;i<n;i++){
		printf("%d ",arr[i]);
	}
	printf("\nenter the index of the element to be increased: ");
	scanf("%d",&id);
	printf("\nenter the increased val: ");
	scanf("%d",&key);
	increase_key(arr,id,key);
	printf("\nafter increase key the heap(max heap) is: ");
```

```
	for(i=0;i<n;i++){
		printf("%d ",arr[i]);
	}
	printf("\n");
}

void create_max_heap(int arr[]){
	int largest_non_leaf=(n-1)/2;
	for(int i=largest_non_leaf;i>=0;i--){
		max_heapify(arr,i);
	}
}

void increase_key(int arr[],int id,int key){
	if(arr[id]>key){
		printf("ERROR:the element is already greater than key\n");
	}
	else{
		arr[id]=key;
	   while(id>0 && arr[(id-1)/2]<arr[id])  //as key>arr[id],arr[id]>all eles of that subtree(root=id),so key>all eles of that subtree(root=id) so no need to apply mh from root to end,may the increased value is greater then its parent so apply mh to id's parent till root
		{
	   int temp=arr[(id-1)/2];
	   arr[(id-1)/2]=arr[id];
	   arr[id]=temp;
	   id=(id-1)/2;
	}
}
}

void max_heapify(int arr[],int i){
```

```
        int rc,lc,largest;
        lc=2*i+1;
        rc=2*i+2;
        if(lc<n && arr[i]<arr[lc]){
                largest=lc;
        }
        else{
                largest=i;
        }
        if(rc<n && arr[largest]<arr[rc]){
                largest=rc;
        }
        if(largest!=i){
                int temp=arr[i];
                arr[i]=arr[largest];
                arr[largest]=temp;
                max_heapify(arr,largest);
        }
}
```