/*extract max element from max heap*/

```c
#include <stdio.h>

int n=0;


int extract_max_ele(int arr[]);

void max_heapify(int arr[],int i);

int main()

{


        int i,arr[20];   //just taking a random no of eles

        printf("Enter the array size (the array is an array representation of a heap): "); //so,heap size=arr size

scanf("%d", &n);

//int arr[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");

while (n > 0) { //as,if n=0 then no element in heap

int max = extract_max_ele(arr);

printf("Extracted max element: %d\n", max);

}

return 0;

}

int extract_max_ele(int arr[]){

if(n<0){
```

```c
printf("ERROR:heap underflow,there is no element in that heap\n");
}
int max=arr[0];
arr[0]=arr[n-1]; //as index is starting from 0
n--;
max_heapify(arr,0); //heapifying from the 1st index
return max;
}
void max_heapify(int arr[], int i) {
int lc, rc, largest=i;
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]) {
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); //Recursively heapify the affected sub-tree,subtree with root as largest
}
}
```

```
Enter the array size (the array is an array representation of a heap): 8
Enter the array elements:
25
14
19
7
9
13
12
2
the array (heap) is: 25 14 19 7 9 13 12 2
Extracted max element: 25
Extracted max element: 19
Extracted max element: 14
Extracted max element: 13
Extracted max element: 12
Extracted max element: 9
Extracted max element: 7
Extracted max element: 2

------------------------------------
Process exited after 8.015 seconds with return value 0
Press any key to continue . . .
```