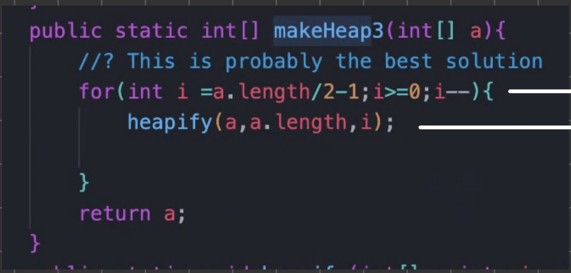
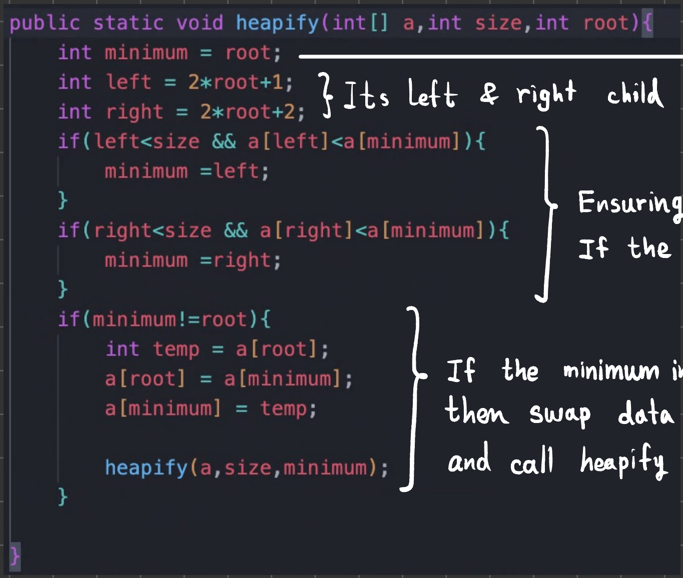


```java
public static int[] makeHeap3(int[] a){
    //? This is probably the best solution
    for(int i =a.length/2-1;i>=0;i--){
        heapify(a,a.length,i);

    }
    return a;
}
```

Start at the last non leaf node

call heapify on each root.



```java
public static void heapify(int[] a,int size,int root){
    int minimum = root;
    int left = 2*root+1;
    int right = 2*root+2;
    if(left<size && a[left]<a[minimum]){
        minimum =left;
    }
    if(right<size && a[right]<a[minimum]){
        minimum =right;
    }
    if(minimum!=root){
        int temp = a[root];
        a[root] = a[minimum];
        a[minimum] = temp;

        heapify(a,size,minimum);
    }
}
```

Intitialize min index as root

} Its left & right child index

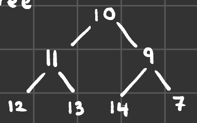
Ensuring the index is less than the size of the array.
If the child is less than current minimum change minimum's index

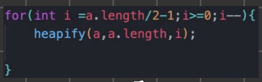
If the minimum index isn't root (root is already heapified)
then swap data of the root & minimum index
and call heapify on the minimum index

Example

array = [10, 11, 9, 12, 13, 14, 7]

Tree



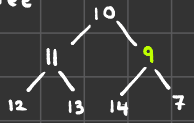


```java
for(int i =a.length/2-1;i>=0;i--){
    heapify(a,a.length,i);
}
```

start i = 7/2 - 1
= 3 - 1
= 2

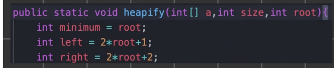
array = [10, 11, 9, 12, 13, 14, 7]

Tree



heapify (array, 7, 2)

heapify (array, 7, 2)

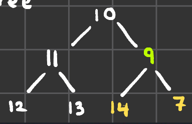


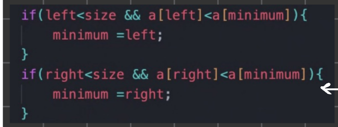
```java
public static void heapify(int[] a,int size,int root){
    int minimum = root;
    int left = 2*root+1;
    int right = 2*root+2;
```

minimum ↓ Left ↓ Right ↓

array = [10, 11, 9, 12, 13, 14, 7]

Tree





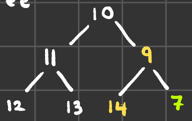
```java
if(left<size && a[left]<a[minimum]){
    minimum =left;
}
if(right<size && a[right]<a[minimum]){
    minimum =right;
}
```

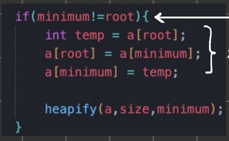
← condition met

root ↓ Left ↓ minimum ↓

array = [10, 11, 9, 12, 13, 14, 7]

Tree





```java
if(minimum!=root){
    int temp = a[root];
    a[root] = a[minimum];
    a[minimum] = temp;

    heapify(a,size,minimum);
}
```

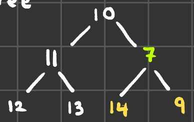
← condition met

} swap

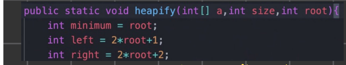
→ heapify (array, 7, 6) [Does nothing]

array = [10, 11, 7, 12, 13, 14, 9]

Tree



heapify (array, 7, 1) [Does nothing]

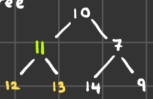


```java
public static void heapify(int[] a,int size,int root){
    int minimum = root;
    int left = 2*root+1;
    int right = 2*root+2;
```

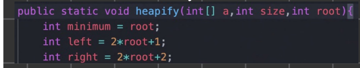
min Left right

array = [10, 11, 7, 12, 13, 14, 9]

Tree



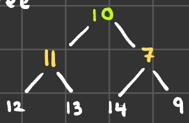
heapify (array, 7, 0)



```java
public static void heapify(int[] a,int size,int root){
    int minimum = root;
    int left = 2*root+1;
    int right = 2*root+2;
```

array = [10, 11, 7, 12, 13, 14, 9]

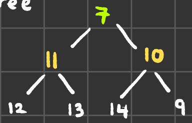
Tree



Swap occur and

array = [7, 11, 10, 12, 13, 14, 9]

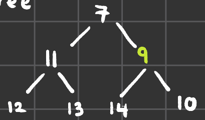
Tree



heapify (array, 7, 2) called
and swap occur.

array = [7, 11, 9, 12, 13, 14, 10]

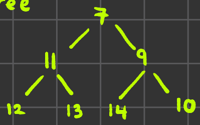
Tree



array = [7, 11, 9, 12, 13, 14, 10]

Done!

Tree


7
11
9
12
13
14
10