Data Structures

Sorting Technique – Heap Sort

Team Emertxe

Build Maxheap

Build Maxheap

- arr[SIZE]

SIZE = 10

Build Maxheap

- arr[SIZE]

SIZE = 10

1	14	10	8	7	9	3	2	4	6

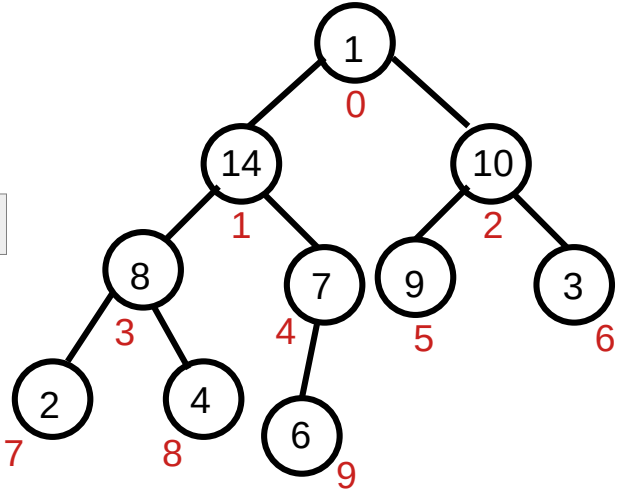
arr[0] arr[1] arr[2] arr[3] arr[4] arr[5] arr[6] arr[7] arr[8] arr[9]

Build Maxheap

- arr[SIZE]

SIZE = 10

1	14	10	8	7	9	3	2	4	6
arr[0]	arr[1]	arr[2]	arr[3]	arr[4]	arr[5]	arr[6]	arr[7]	arr[8]	arr[9]



Algorithm

build_maxheap(arr,size):

Input Specification:

arr : Array to hold elements

size : Length of the array

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

Algorithm

build_maxheap(arr,size):

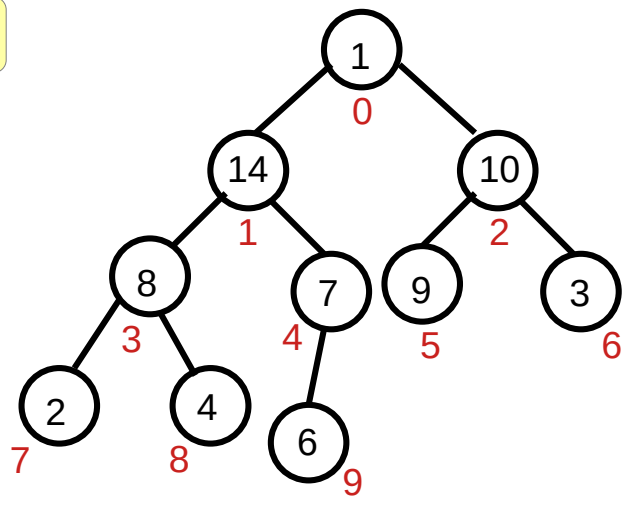
```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
                maxheapify(arr,index,size)
                Decrement index
```

SIZE = 10

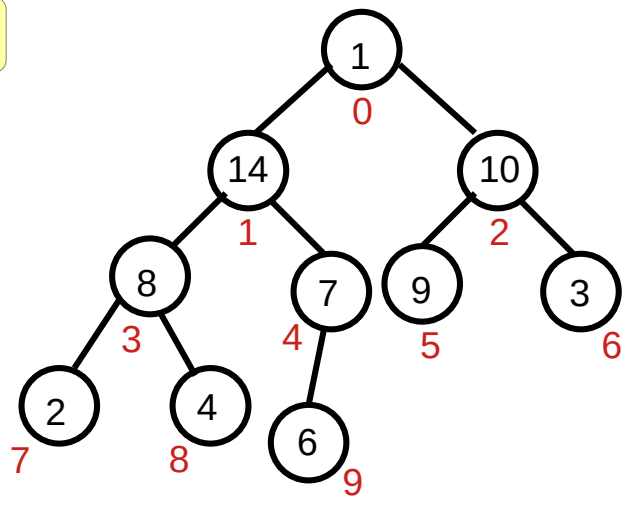


Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10



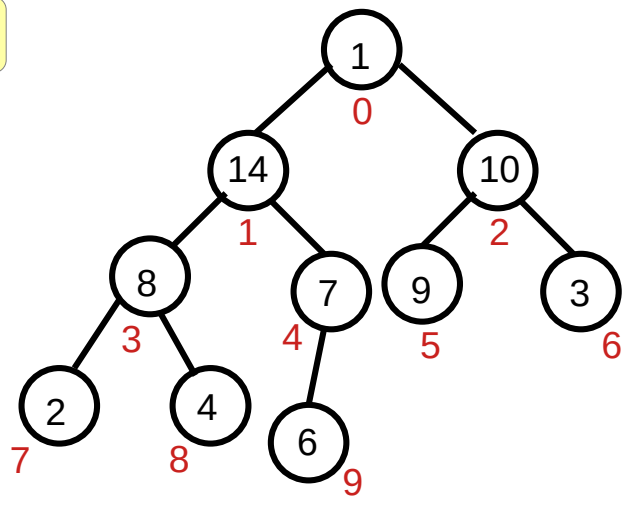
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 4



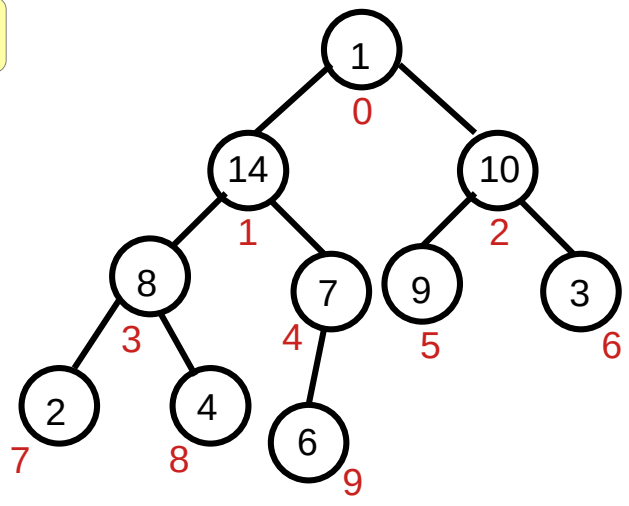
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 4



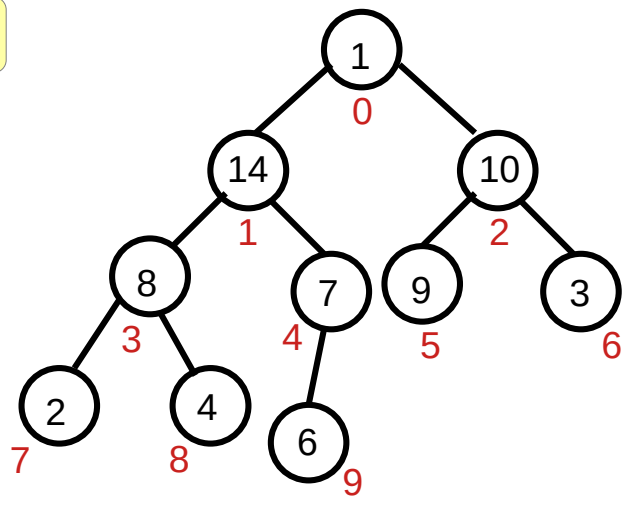
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 4



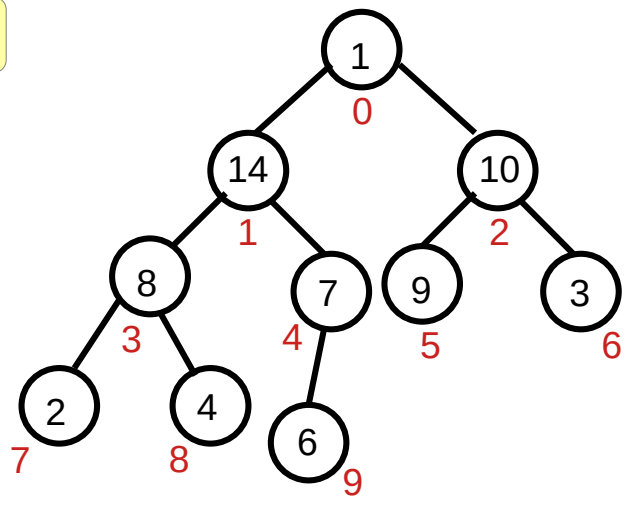
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 4



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

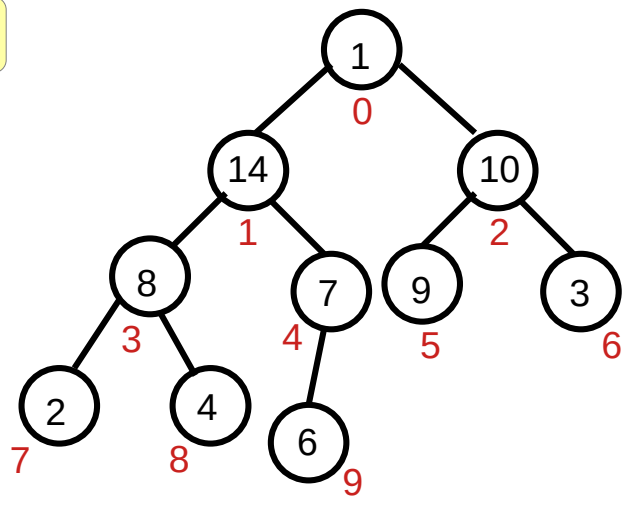
build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 4

L = 9



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

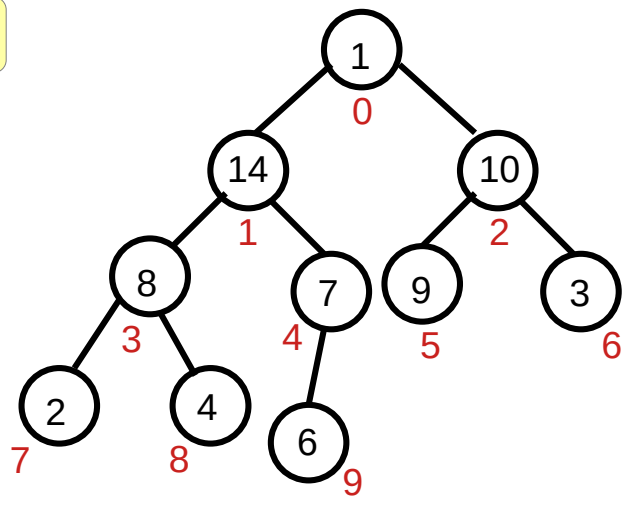
```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 4

L = 9

R = 10



maxheapify(arr,index,size)

```
 L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

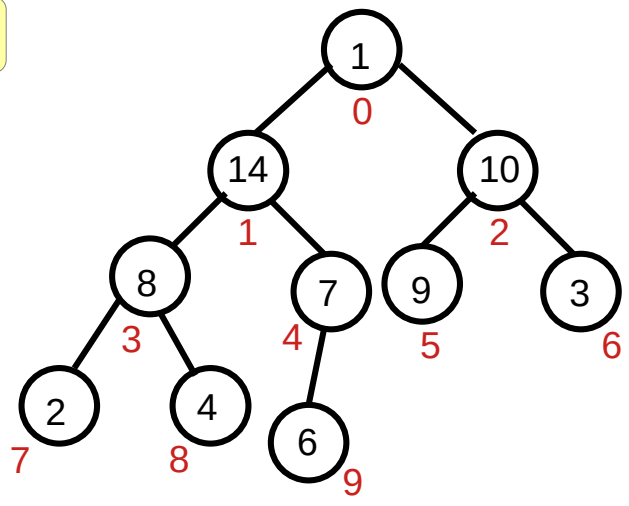
```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 4

L = 9

R = 10



maxheapify(arr,index,size)

```
 L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

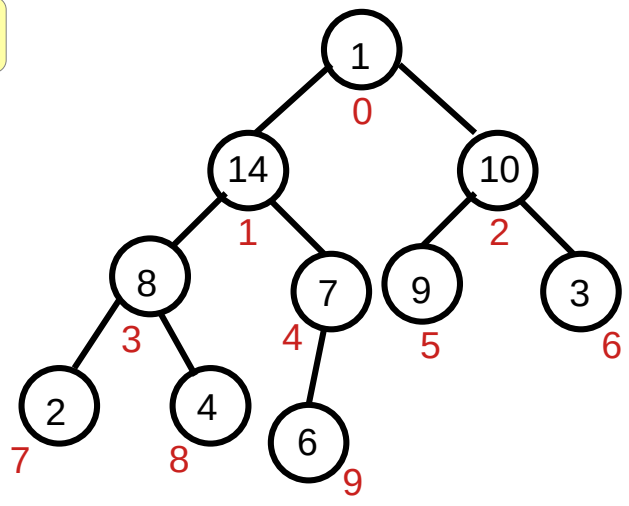
```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 4

L = 9

R = 10



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

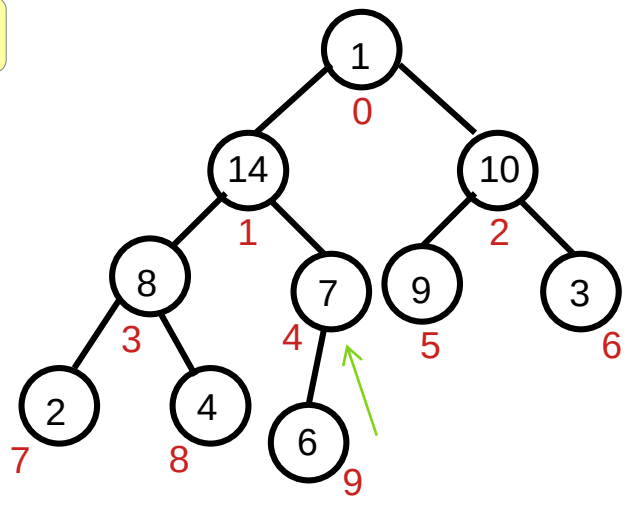
SIZE = 10

index = 4

L = 9

R = 10

large = 4



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

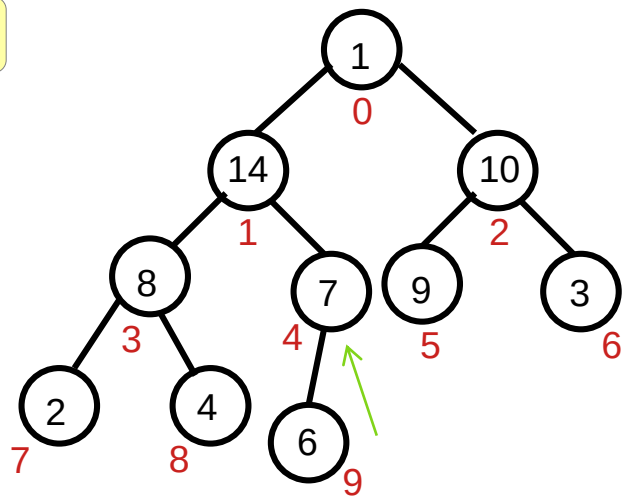
SIZE = 10

index = 4

L = 9

R = 10

large = 4



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

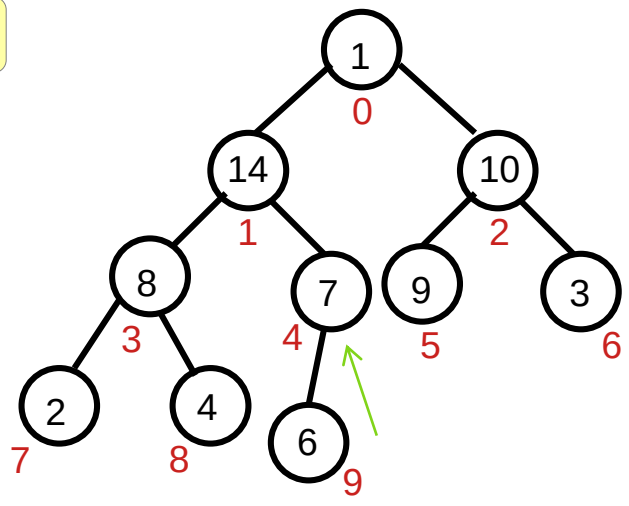
SIZE = 10

index = 4

L = 9

R = 10

large = 4



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

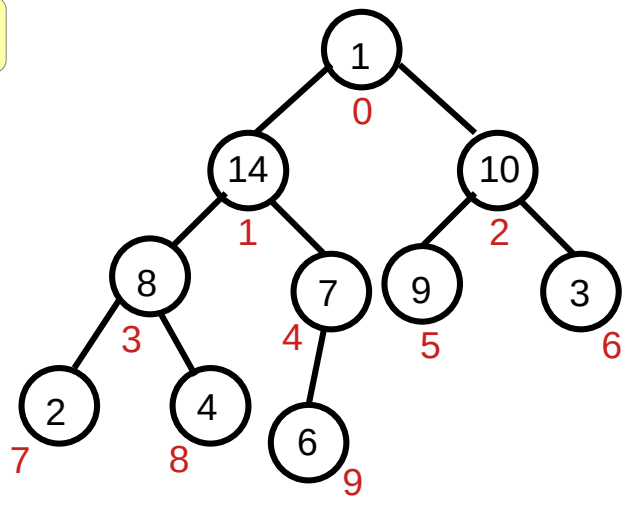
SIZE = 10

index = 4

L = 9

R = 10

large = 4



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

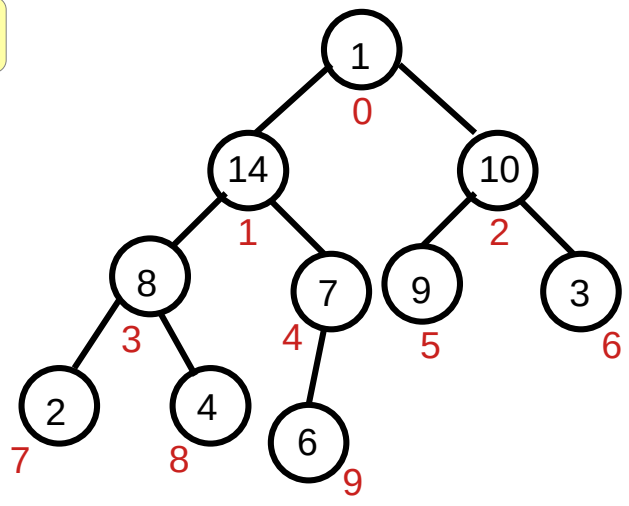
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 3



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

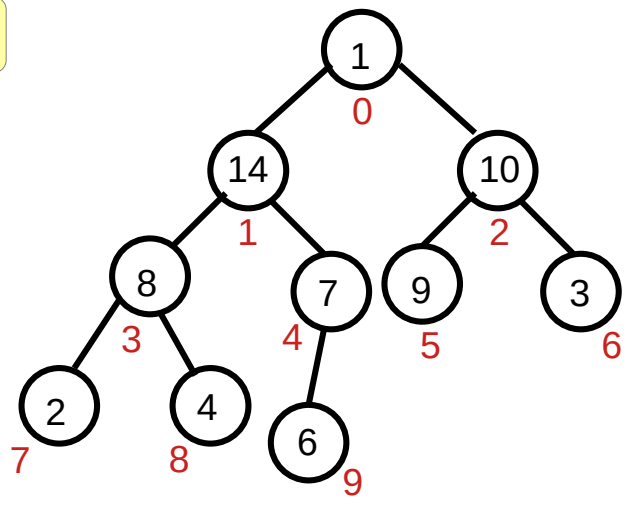
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 3



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

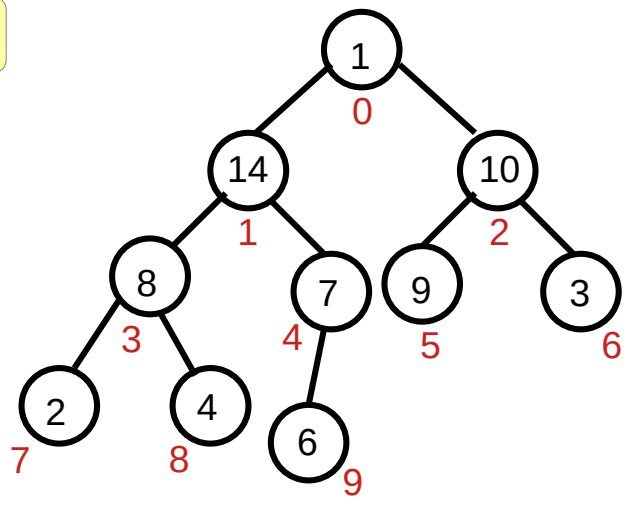
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 3



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

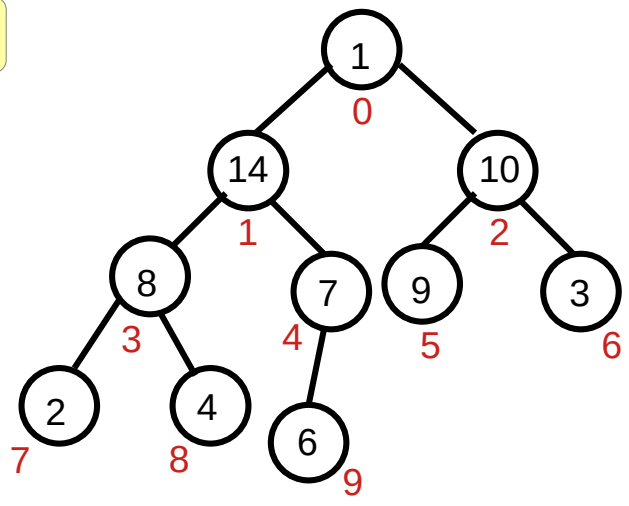
```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 3

L = 7

R = 8



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

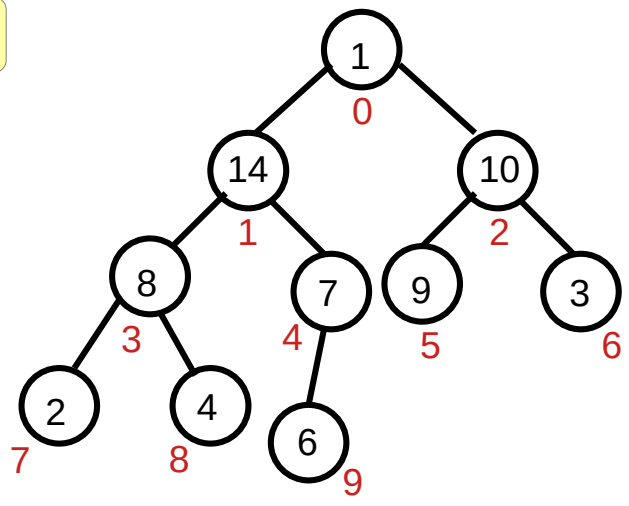
```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 3

L = 7

R = 8



maxheapify(arr,index,size)

```
 L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

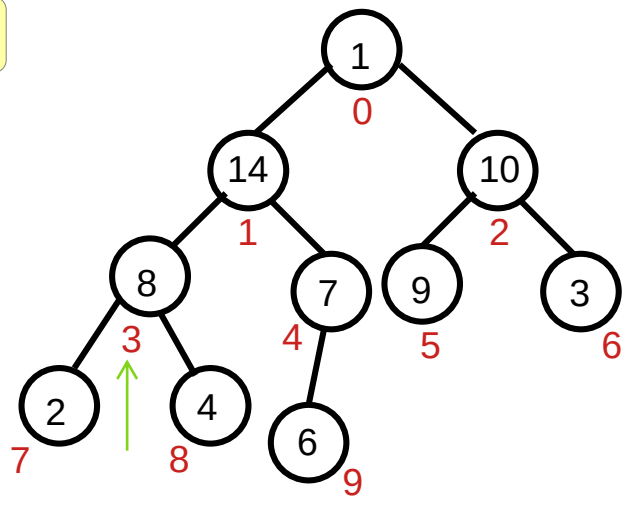
SIZE = 10

index = 3

L = 7

R = 8

large = 3



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

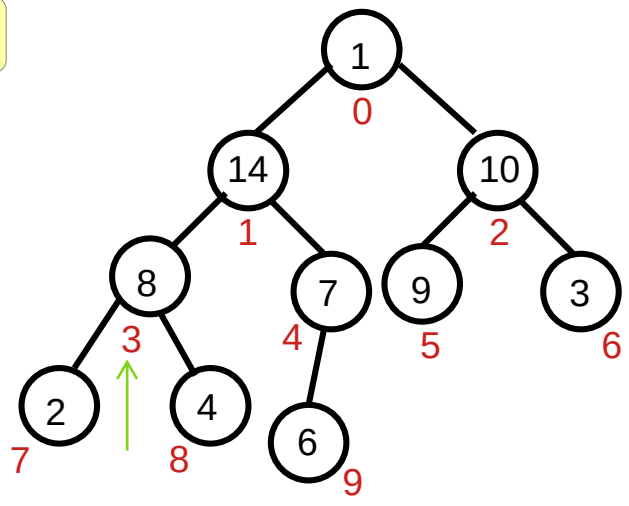
SIZE = 10

index = 3

L = 7

R = 8

large = 3



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

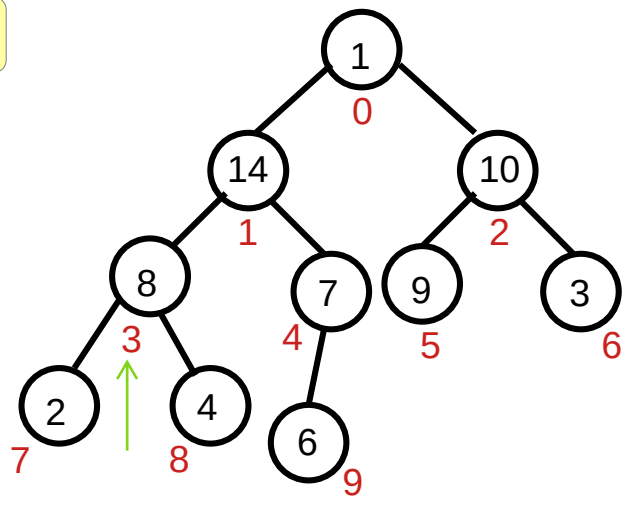
SIZE = 10

index = 3

L = 7

R = 8

large = 3



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

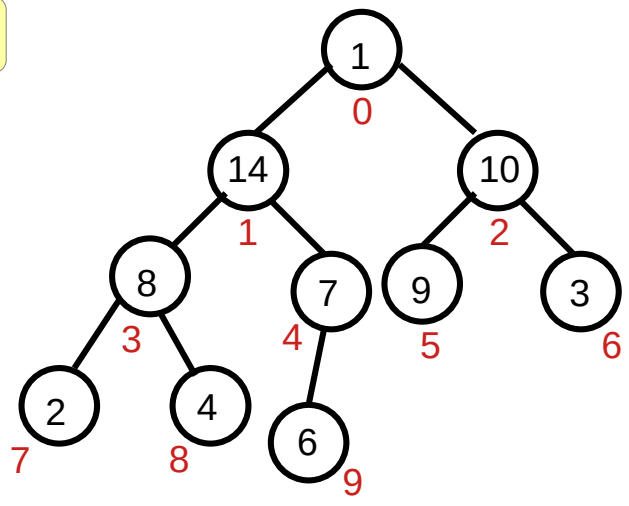
SIZE = 10

index = 3

L = 7

R = 8

large = 3



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

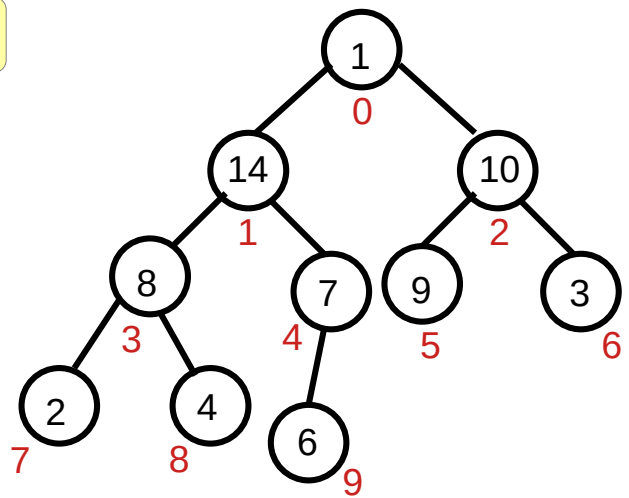
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 2



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

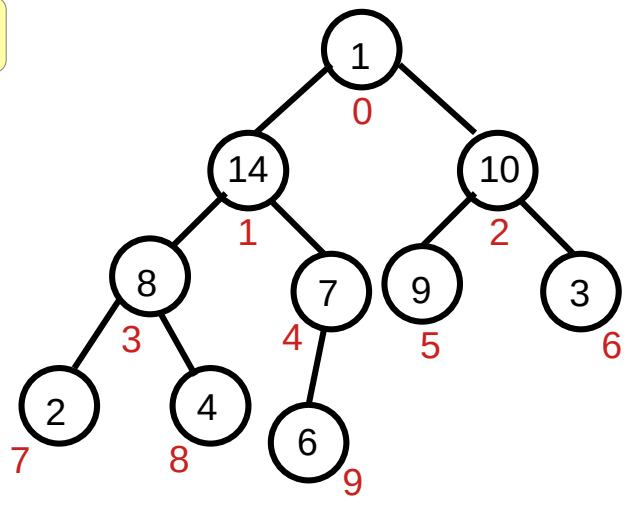
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 2



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

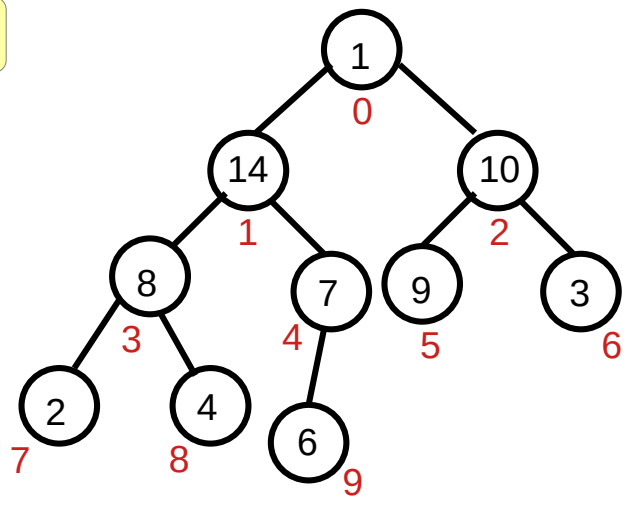
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 2



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

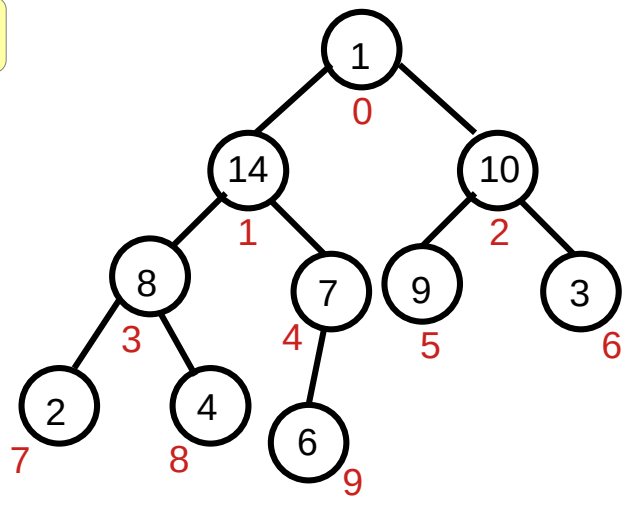
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 2



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

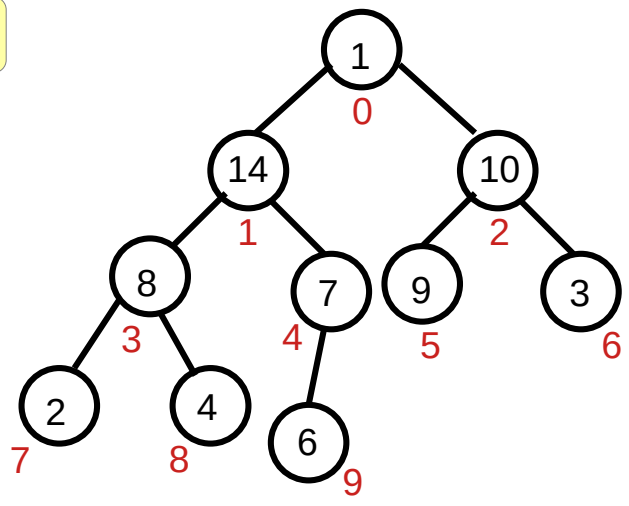
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 1



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

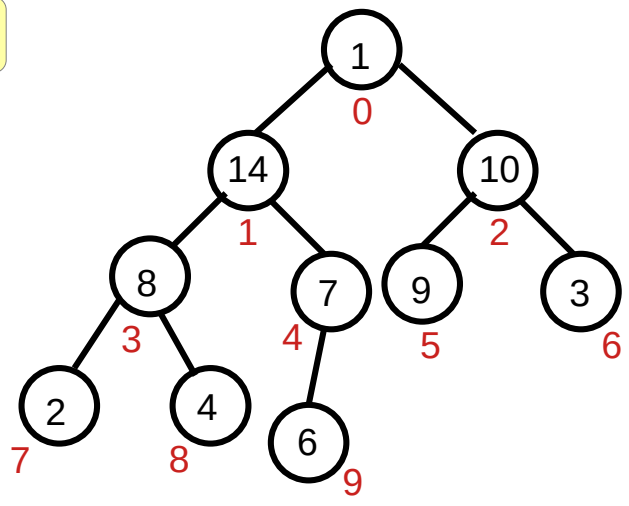
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 1



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

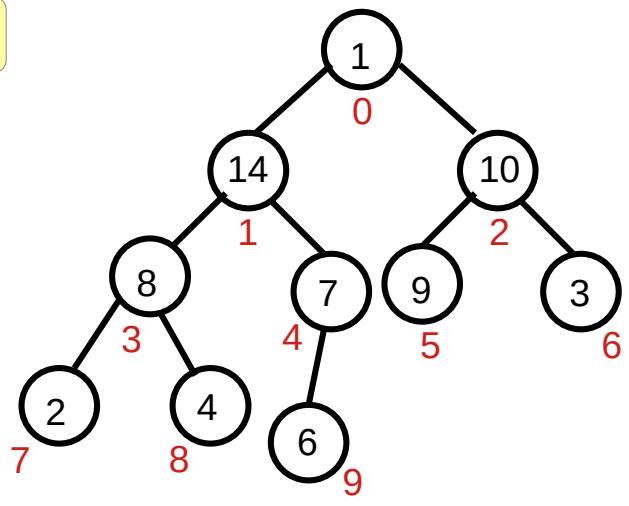
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 1



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

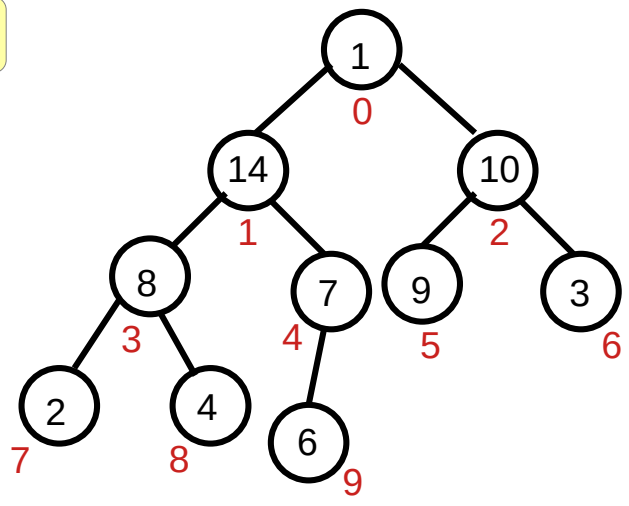
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 1



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

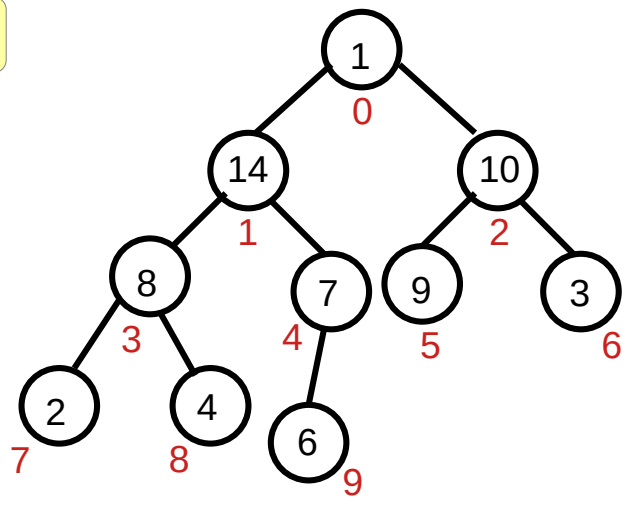
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 0



maxheapify(arr,index,size)

```
 L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

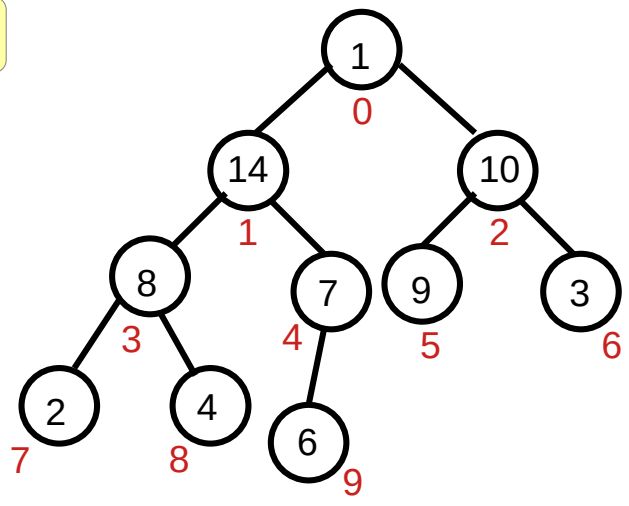
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 0



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

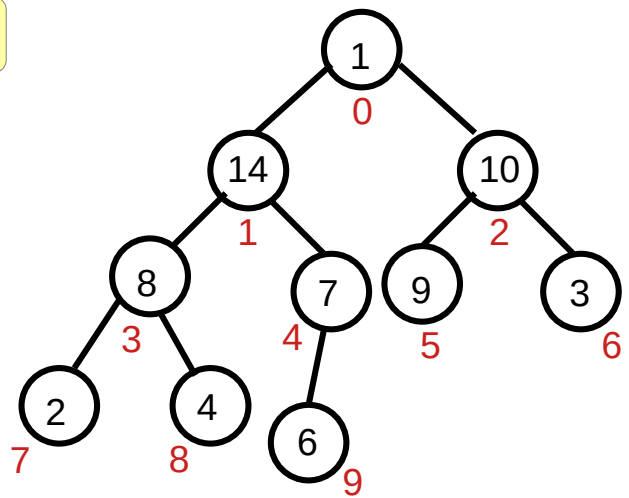
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 0



maxheapify(arr,index,size)

```
 L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

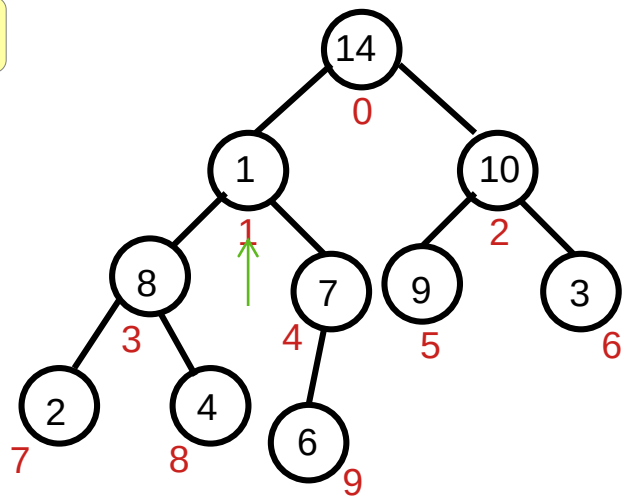
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 0



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

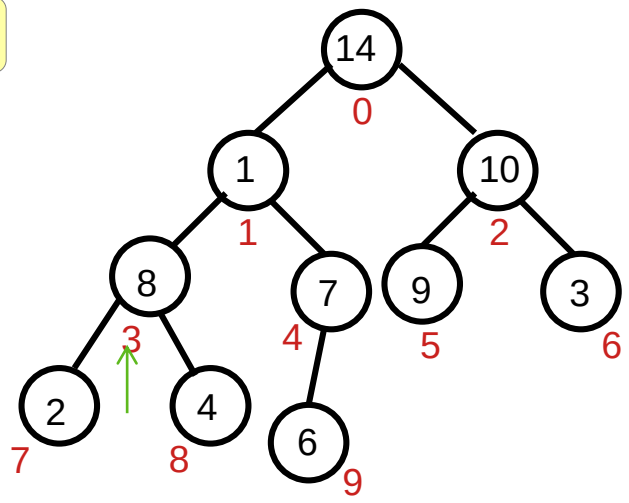
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 0



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

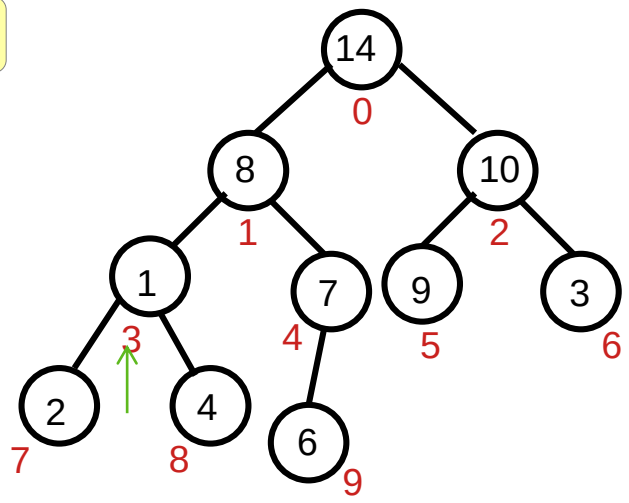
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 0



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

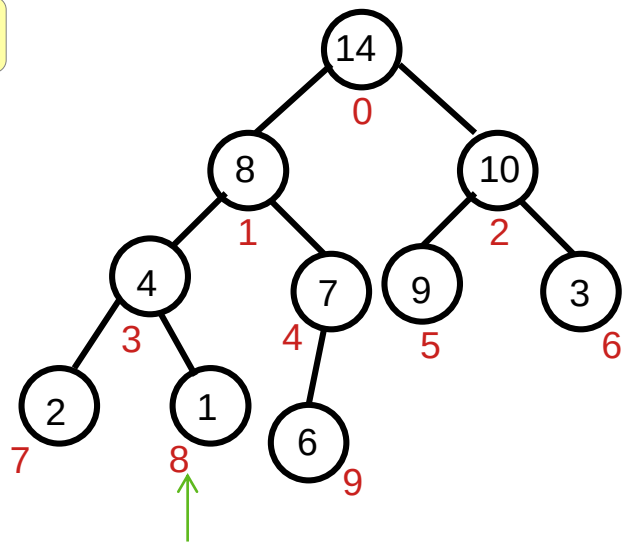
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 0



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

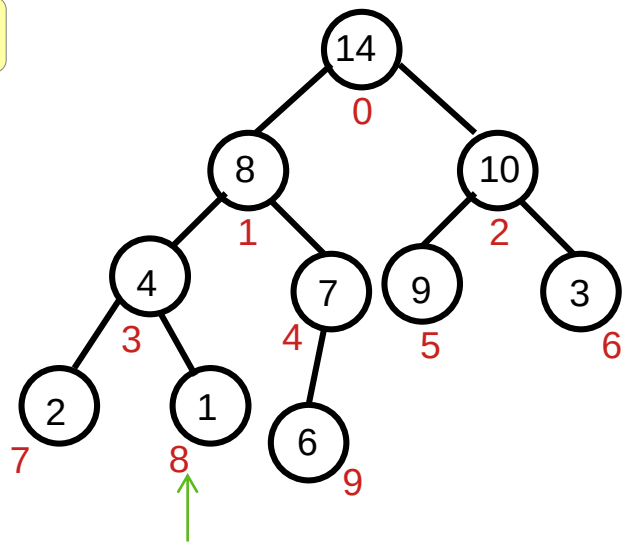
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = 0



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

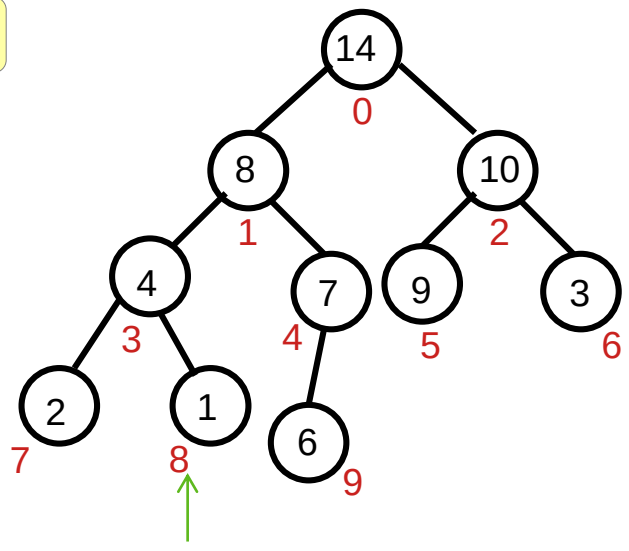
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = -1



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

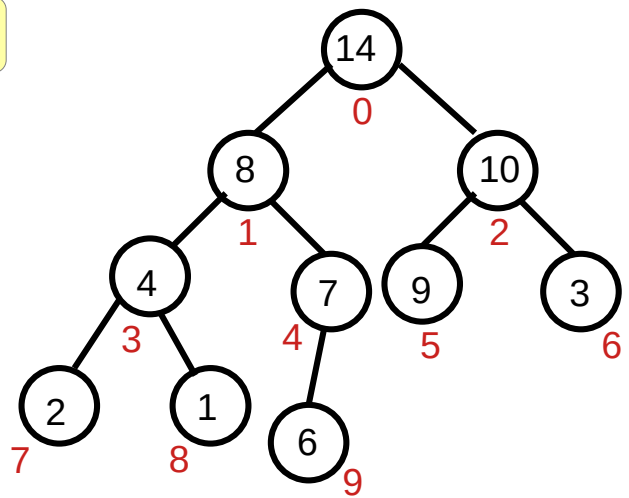
Algorithm

build_maxheap(arr,size):

```
index = size/2 – 1
while(index >= 0 )
        maxheapify(arr,index,size)
        Decrement index
```

SIZE = 10

index = -1



maxheapify(arr,index,size)

```
L = 2*index +1
R = 2*index+2
if(arr[index] < arr[L] AND L < size)
  large = L
else
  large = index
if(arr[large]< arr[R] AND R < size)
  large = R
if(index != large)
  swap(arr[large] ,arr[index])
  maxheapify(arr,large,size)
```

EMERTXE

Heap Sort