

Mastering Heap

March 15, 2017

Question: What are the minimum and maximum number of elements in a heap of height h ?

Answer: Minimum : 2^h , Maximum: $2^{h+1} - 1$

1 Build Max-Heap

We need to know that in a heap data structure the leaves start from $(\text{len}(A)/2 + 1)$. Because if you want to get the child of this leaf you would get $2 * (\text{len}(A) / 2 + 1) = \text{len}(A) + 2$, which is not possible. Even the last element of the array is the child of the array-index $\text{len}(A)/2$

We also need the procedure MAX-HEAPIFY(A, i)

```
In [57]: def max_heapify(A, i):
         l = 2 * i + 1
         r = 2 * i + 2
         if l >= len(A) and r >= len(A):
             return
         if l < len(A) and A[l] >= A[i]:
             largest = l
         else:
             largest = i
         if r < len(A) and A[r] >= A[largest]:
             largest = r
         if largest != i:
             A[largest], A[i] = A[i], A[largest]
             max_heapify(A, largest)
```

```
In [58]: def build_max_heap(A):
         i = len(A) / 2 - 1
         while i >= 0:
             max_heapify(A, i)
             i = i - 1
         return A
```

```
In [59]: build_max_heap([1, 2, 3, 4, 9, 16, 7])
```

```
Out[59]: [16, 9, 7, 4, 2, 3, 1]
```

```
In [60]: def min_heapify(A, i):
         l = 2*i + 1
```

```

    r = 2*i + 2
    if l >= len(A) and r >= len(A):
        return
    if l < len(A) and A[l] <= A[i]:
        smallest = l
    else:
        smallest = i
    if r < len(A) and A[r] <= A[smallest]:
        smallest = r
    if smallest != i:
        A[smallest], A[i] = A[i], A[smallest]
        min_heapify(A, smallest)

```

```

In [61]: def build_min_heap(A):
        i = len(A) // 2
        while i >= 0:
            min_heapify(A, i)
            i = i - 1
        return A

```

```

In [62]: build_min_heap([12, 3, 4, 5, 2, 1])

```

```

Out[62]: [1, 2, 4, 5, 3, 12]

```