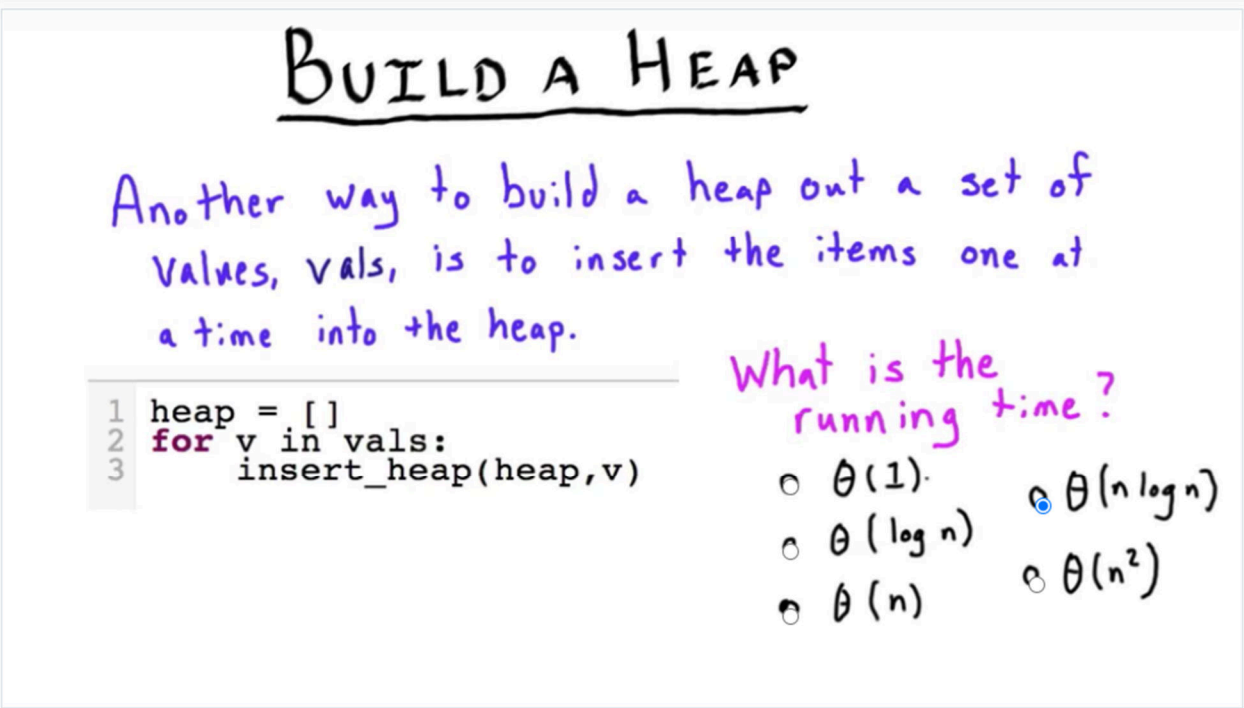
Build a Heap



Minimize Sum of Absolute Value

```
 9 def minimize_absolute(L):
10     L2 = L[:]
11     L2.sort()
12     if len(L2) % 2 == 0:
13         return (L2[(len(L2) - 1)/ 2] + L2[(len(L2) - 1)/ 2]) / 2.
14     else:
15         return L2[(len(L2) - 1)/ 2]
16
```

Minimize Sum of Squares

```
 7 #
 8 def minimize_square(L):
 9     return sum(L) * 1. / len(L)
```

Mode

```
def mode(L):
    counts = {}
    for each in L:
        if each not in counts:
            counts[each] = 0
        counts[each] += 1
    x = 0
    y = 0
    for each in counts:
        if counts[each] > x:
            x = counts[each]
            y = each
    return y
```

Up Heapify

```
def up_heapify(L, i):
    while i > 0 and L[i] < L[parent(i)]:
        L[parent(i)], L[i] = L[i], L[parent(i)]
        i = parent(i)
```

Actor Centrality