Data Structures

Built-in Heap

Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Using **heapq** module

- The same concepts, but follow this style
- The list is used
  as the internal array
- So data in the list

```
import heapq

minHeap = []
lst = [2, 17, 22, 10, 8, 37, 14,
       19, 7, 6, 5, 12, 25, 30]

for val in lst:
    heapq.heappush(minHeap, val)

print(minHeap)
# 2, 5, 12, 8, 6, 14, 22, 19, 17, 10, 7, 37, 25, 30

for step in range(len(minHeap)):
    print(heapq.heappop(minHeap), end=', ')
# 2, 5, 6, 7, 8, 10, 12, 14, 17, 19, 22, 25, 30, 37
```

O(n) heap creation

- We can heapify a list in O(n)

```
minHeap = [2, 17, 22, 10, 8, 37, 14,
           19, 7, 6, 5, 12, 25, 30]

heapq.heapify(minHeap)  # O(n) creation

print(minHeap)
# 2, 5, 12, 8, 6, 14, 22, 19, 17, 10, 7, 37, 25, 30

for step in range(len(minHeap)):
    print(heapq.heappop(minHeap), end=', ')
# 2, 5, 6, 7, 8, 10, 12, 14, 17, 19, 22, 25, 30, 37
```

Largest and Smallest K elements

- Internal list of length K is created. O(k) to heapify the first K elements
  - Then some smartness to get the largest/smallest elements in O(nlogk)

```
lst = [2, 17, 22, 10, 8, 37, 14,
           19, 7, 6, 5, 12, 25, 30]

# observe it is lst. No need to be heap

# Find the largest K elements
k = 3
print(heapq.nlargest(k, lst))   # [37, 30, 25]
print(heapq.nlargest(k, lst))   # [37, 30, 25]
print(heapq.nsmallest(k, lst))  # [2, 5, 6]

print(lst)  # NOT changed

# O(k) memory. O(nlogk) time. We will implement in homework
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."