Programming Assignment #2 - Part 1

학번 / 학과 : B889047 / 소프트웨어 융합학과

작성자 : 윤준호

Part 1

Topic : Implement Priority Queue using 'Heap' ADT

Least required operations

- Insert(Q,k) : 키값 k를 큐 Q에 추가한다
- Delete(Q,k) : 키값 k를 큐 Q에서 제거한다
- ExtractMin(Q) : 가장 키값이 작은 원소의 키값을 제거하고 그 키값을 리턴한다
- IncreaseKey(Q,k,v) : 키값 k 를 가진 원소의 키 값을 증가된 키값 v로 바꾼다
- DecreaseKey(Q,k,v) : 키값 k 를 가진 원소의 키 값을 감소된 키값 v로 바꾼다

Implement Priority Queue using Polymorhpic type

- 여기서 다형성 준수는, 분기문에 의한 다형성 준수이다.

Install dependency packages from pip

In [1]:

```
!pip3 install -r requirements.txt
```

```
Requirement already satisfied: treelib in /Users/hoplin/opt/anaconda3/
lib/python3.9/site-packages (from -r requirements.txt (line 1)) (1.6.
1)
Requirement already satisfied: future in /Users/hoplin/opt/anaconda3/l
ib/python3.9/site-packages (from treelib->-r requirements.txt (line
1)) (0.18.2)
```

Priorty Queue 모듈 import 하기

- 이 Priority Queue모듈은 Heap모듈에 의존성을 가집니다

In [2]:

```
from PriQueue.priorityqueue import PriorityQueue
```

- Class Document를 출력합니다

In [3]:

```
help(PriorityQueue)
```

```
Help on class PriorityQueue in module PriQueue.priorityqueue:

class PriorityQueue(builtins.object)
 |  PriorityQueue(initial_sequence: MutableSequence, heap_type: str
= 'max')
 |
 |  < Class Document : Priority Queue >
 |
 |  - 부모클래스 : object
 |  - 의존성 : Heap,MaxHeap,MinHeap
 |
 |  < Exception Class >
 |
 |  - WrongConstructorOptionException
 |      Description:
 |          PriortyQueue생성자에게 heap_type매개변수를 넘길때 잘못된 타입을 넘
기게 될 경우에 이 예외가 발생됩니다
 |
 |  < Methods >
```

Priority Queue using Max Heap

- PriorityQueue 인스턴스를 생성합니다. 최대힙을 이용한 우선순위큐를 생성해 봅니다.

In [4]:

```
pq = PriorityQueue([],"max")
```

Insert(k) 연산 수행하기

- PriorityQueue에 10개의 원소를 insert해보겠습니다

In [5]:

```
pq.insert(6)
pq.insert(4)
pq.insert(5)
pq.insert(9)
pq.insert(1)
pq.insert(10)
pq.insert(4)
pq.insert(3)
pq.insert(1)
pq.insert(8)
```

- 우선순위큐를 출력해보겠습니다

In [6]:

```
pq()
```

Out[6]:

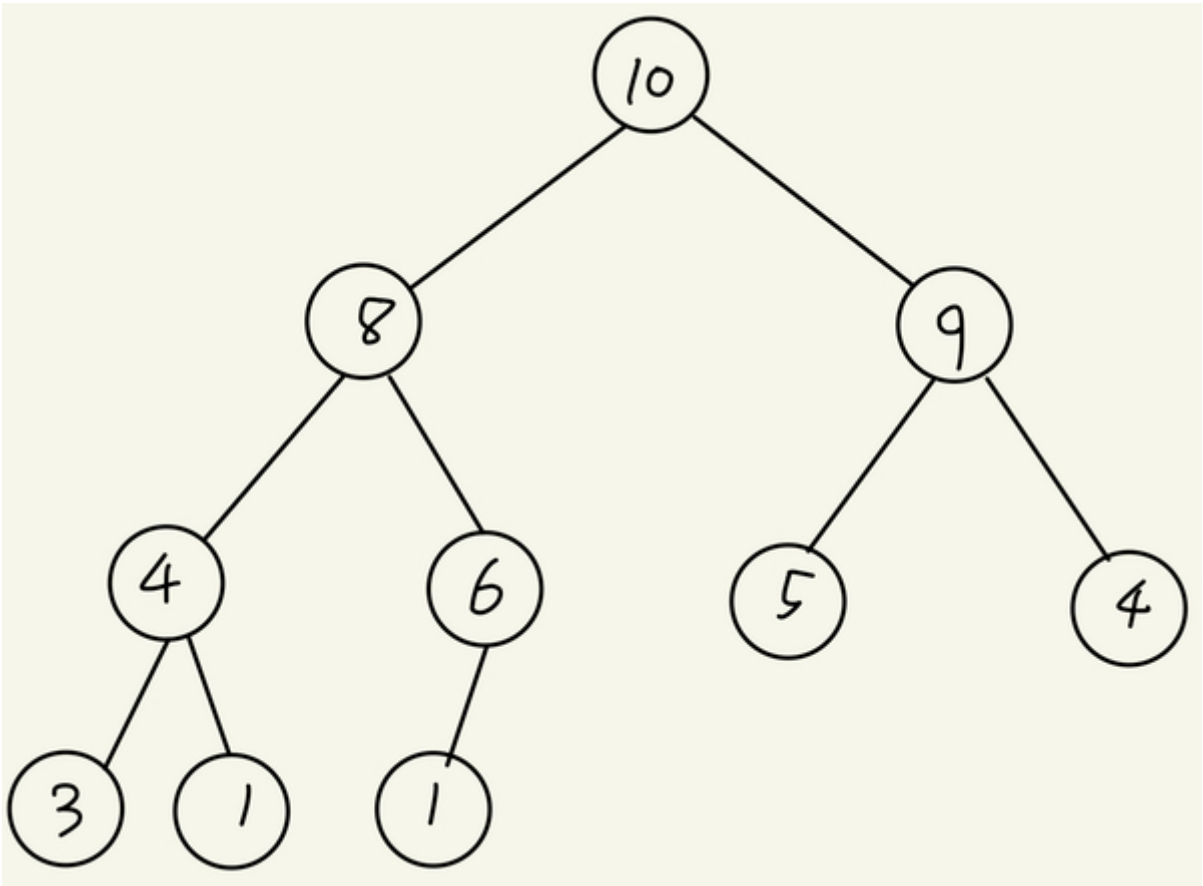
```
[10, 8, 9, 4, 6, 5, 4, 3, 1, 1]
```

- 이 우선순위 큐를 완전 이진트리로 출력하면 아래와 같습니다. 완전 이진트리를 만족하면서,Max Heap을 만족하고 있는것을 볼 수 있습니다.

In [7]:

```
pq.show_in_tree()
```

```
10
├── 8
│   ├── 4
│   │   ├── 1
│   │   └── 3
│   └── 6
│       └── 1
└── 9
    ├── 4
    └── 5
```



Delete(k)연산 수행하기

- 3,6,9를 삭제해 보겠습니다. 각 단계별로 우선순위 큐가 어떻게 변하는지 출력해 보겠습니다
- Delete연산을 수행 하고 난 이후에도 완전 이진트리 형태를 갖추고 있어야 합니다.

In [8]:

```
pq.delete(3)
pq()
```

Out[8]:

```
[10, 8, 9, 4, 6, 5, 4, 1, 1]
```

In [9]:

```
pq.show_in_tree()
```

```
10
├─── 8
│    ├─── 4
│    │    ├─── 1
│    │    └─── 1
│    └─── 6
└─── 9
     ├─── 4
     └─── 5
```

In [10]:

```
pq.delete(6)
pq()
```

Out[10]:

```
[10, 8, 9, 4, 1, 5, 4, 1]
```

In [11]:

```
pq.show_in_tree()
```

```
10
├─── 8
│    ├─── 1
│    └─── 4
│         └─── 1
└─── 9
     ├─── 4
     └─── 5
```

In [12]:

```
pq.delete(9)
pq()
```

Out[12]:

```
[10, 8, 5, 4, 1, 1, 4]
```

In [13]:

```
pq.show_in_tree()
```

```
10
├─── 5
│    ├─── 1
│    └─── 4
└─── 8
     ├─── 1
     └─── 4
```

IncreaseKey(k,v)연산 수행하기

- 1을 7로, 8을 11로 바꿔보겠습니다
- 만약 v값이 k보다 작거나 같은 값인 경우에는 Heap.WrongLogicException을 발생시킵니다

In [14]:

```
#pq.increase_key(1,0) #이 문을 실행하면 Heap.WrongLogicException을 발생시킨다

pq.increase_key(1,7)
pq()
```

Out[14]:

```
[10, 8, 5, 4, 7, 1, 4]
```

In [15]:

```
pq.show_in_tree()
```

```
10
├─── 5
│    ├─── 1
│    └─── 4
└─── 8
     ├─── 4
     └─── 7
```

In [16]:

```
pq.increase_key(8,11)
pq()
```

Out[16]:

```
[11, 10, 5, 4, 7, 1, 4]
```

In [17]:

```
pq.show_in_tree()
```

```
11
├─── 5
│    ├─── 1
│    └─── 4
└─── 10
     ├─── 4
     └─── 7
```

DecreaseKey(k,v) 수행하기

- 10을 2로,5를 3으로 바꿔보겠습니다
- 만약 v값이 k보다 크거나 같으면 Heap.WrongLogicException을 발생시킵니다

In [18]:

```
# pq.decrease_key(10,20) # 이 문을 실행시키면 Heap.WrongLogicException을 발생시킵니다
pq.decrease_key(10,2)
pq()
```

Out[18]:

```
[11, 7, 5, 4, 2, 1, 4]
```

In [19]:

```
pq.show_in_tree()
```

```
11
├─── 5
│    ├─── 1
│    └─── 4
└─── 7
     ├─── 2
     └─── 4
```

In [20]:

```
pq.decrease_key(5,3)
pq()
```

Out[20]:

```
[11, 7, 4, 4, 2, 1, 3]
```

In [21]:

```
pq.show_in_tree()
```

```
11
├─── 4
│    ├─── 1
│    └─── 3
└─── 7
     ├─── 2
     └─── 4
```

Priority Queue using Min Heap

- 위에서 사용하였던 우선순위큐인스턴스 pq에 대해 convert_heap_type()으로 MinHeap 사용 우선순위큐로 바꿉니다
- 그 후 큐를 비워줍니다

In [22]:

```
print(pq.heap)
pq.convert_heap_type()
print(pq.heap)
```

```
<MaxHeap_Object_0x105b29970 [11, 7, 4, 4, 2, 1, 3]>
<MinHeap_Object_0x105b50580 [1, 2, 3, 4, 7, 4, 11]>
```

In [23]:

```
pq.clear_queue()
```

In [24]:

```
print(pq.heap)
```

```
<MinHeap_Object_0x105b50580 []>
```

Insert(k) 연산 수행하기

- 위에서 추가했던 10개의 원소를 동일하게 insert해보겠습니다

In [25]:

```
pq.insert(6)
pq.insert(4)
pq.insert(5)
pq.insert(9)
pq.insert(1)
pq.insert(10)
pq.insert(4)
pq.insert(3)
pq.insert(1)
pq.insert(8)
```

In [26]:

```
pq()
```

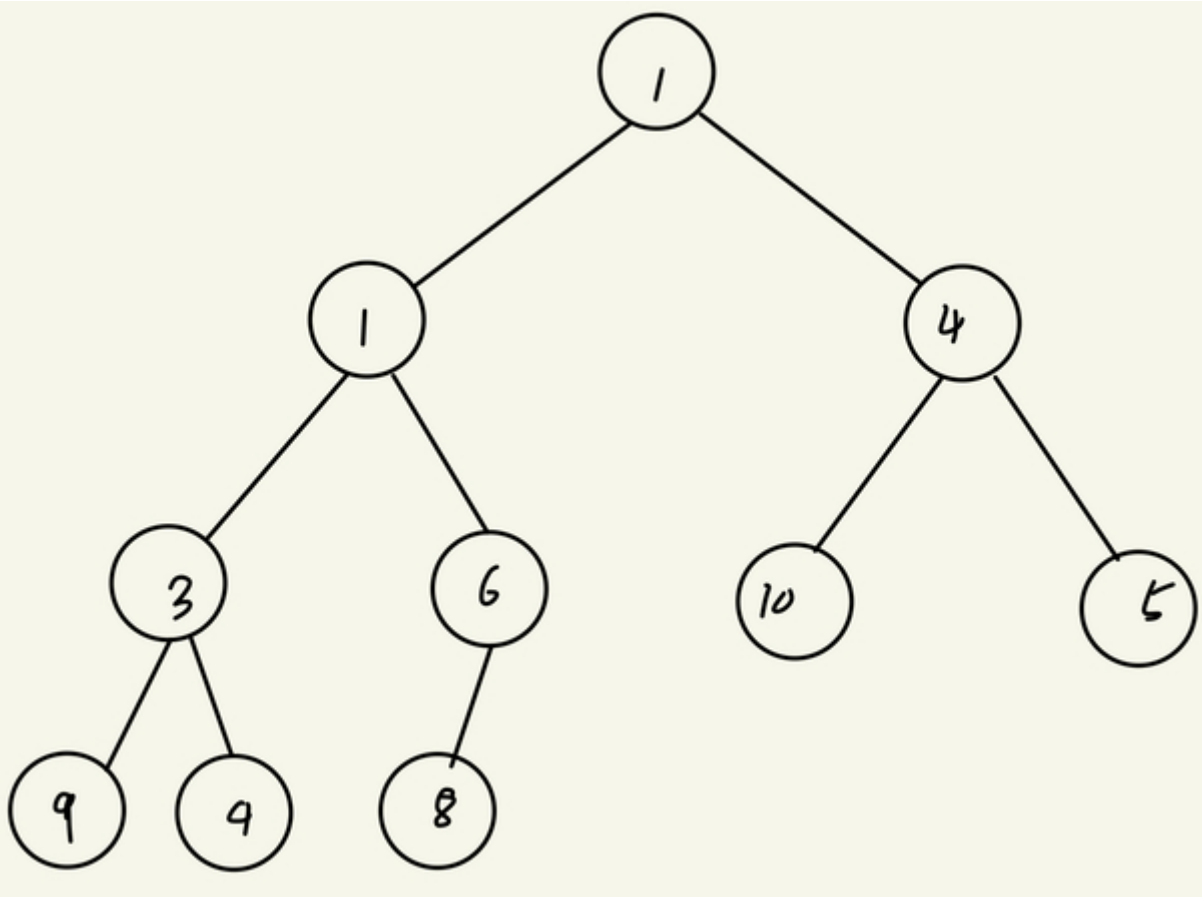
Out[26]:

```
[1, 1, 4, 3, 6, 10, 5, 9, 4, 8]
```

In [27]:

```
pq.show_in_tree()
```

```
1
├─── 1
│    ├─── 3
│    │    ├─── 4
│    │    └─── 9
│    └─── 6
│         └─── 8
└─── 4
     ├─── 5
     └─── 10
```



Delete(k)연산 수행하기

- 3,6,9를 삭제해 보겠습니다. 각 단계별로 우선순위 큐가 어떻게 변하는지 출력해 보겠습니다
- Delete연산을 수행 하고 난 이후에도 완전 이진트리 형태를 갖추고 있어야 합니다.

In [28]:

```
pq.delete(3)
pq()
```

Out[28]:

```
[1, 1, 4, 4, 6, 10, 5, 9, 8]
```

In [29]:

```
pq.show_in_tree()
```

```
1
├── 1
│   ├── 4
│   │   ├── 8
│   │   └── 9
│   └── 6
└── 4
    ├── 5
    └── 10
```

In [30]:

```
pq.delete(6)
pq()
```

Out[30]:

```
[1, 1, 4, 4, 8, 10, 5, 9]
```

In [31]:

```
pq.show_in_tree()
```

```
1
├─── 1
│    ├─── 4
│    │    └─── 9
│    └─── 8
└─── 4
     ├─── 5
     └─── 10
```

In [32]:

```
pq.delete(9)
pq()
```

Out[32]:

```
[1, 1, 4, 4, 8, 10, 5]
```

In [33]:

```
pq.show_in_tree()
```

```
1
├─── 1
│    ├─── 4
│    └─── 8
└─── 4
     ├─── 5
     └─── 10
```

IncreaseKey(k,v)연산 수행하기

- 1을 7로, 8을 11로 바꿔보겠습니다
- 만약 v값이 k보다 작거나 같은 값인 경우에는 Heap.WrongLogicException을 발생시킵니다

In [34]:

```
pq.increase_key(1,7)
pq()
```

Out[34]:

```
[1, 4, 4, 7, 8, 10, 5]
```

In [35]:

```
pq.show_in_tree()
```

```
1
├─── 4
│    ├─── 7
│    └─── 8
└─── 4
     ├─── 5
     └─── 10
```

In [36]:

```
pq.increase_key(8,11)
pq()
```

Out[36]:

```
[1, 4, 4, 7, 11, 10, 5]
```

In [37]:

```
pq.show_in_tree()
```

```
1
├─── 4
│    ├─── 7
│    └─── 11
└─── 4
     ├─── 5
     └─── 10
```

DecreaseKey(k,v) 수행하기

- 10을 2로,5를 3으로 바꿔보겠습니다
- 만약 v값이 k보다 크거나 같으면 Heap.WrongLogicException을 발생시킵니다

In [38]:

```
pq.decrease_key(10,2)
pq()
```

Out[38]:

```
[1, 4, 2, 7, 11, 4, 5]
```

In [39]:

```
pq.show_in_tree()
```

```
1
├─── 2
│    ├─── 4
│    └─── 5
└─── 4
     ├─── 7
     └─── 11
```

In [40]:

```
pq.decrease_key(5,3)
pq()
```

Out[40]:

```
[1, 4, 2, 7, 11, 4, 3]
```

In [41]:

```
pq.show_in_tree()
```

```
1
├─── 2
│    ├─── 3
│    └─── 4
└─── 4
     ├─── 7
     └─── 11
```