HW7

陳卉縈 112356043@nccu.edu.tw
王瀚 111306078@g.nccu.edu.tw
劉亭妤113356048@g.nccu.edu.tw

HW7

Maintain a keyword heap.

- A keyword is a triple [String name, Integer count, Double weight]
- Heap Order: n.count >= n.parent.count
- Use java.util.PriorityQueue
  - http://download.oracle.com/javase/1.5.0/docs/api/java/util/PriorityQueue.html
    https://docs.oracle.com/javase/8/docs/api/java/util/PriorityQueue.html
    https://docs.oracle.com/javase/8/docs/api/java/util/Comparator.html
- Here's an example of a priority queue sorting by string length
- Reuse your code in HW4

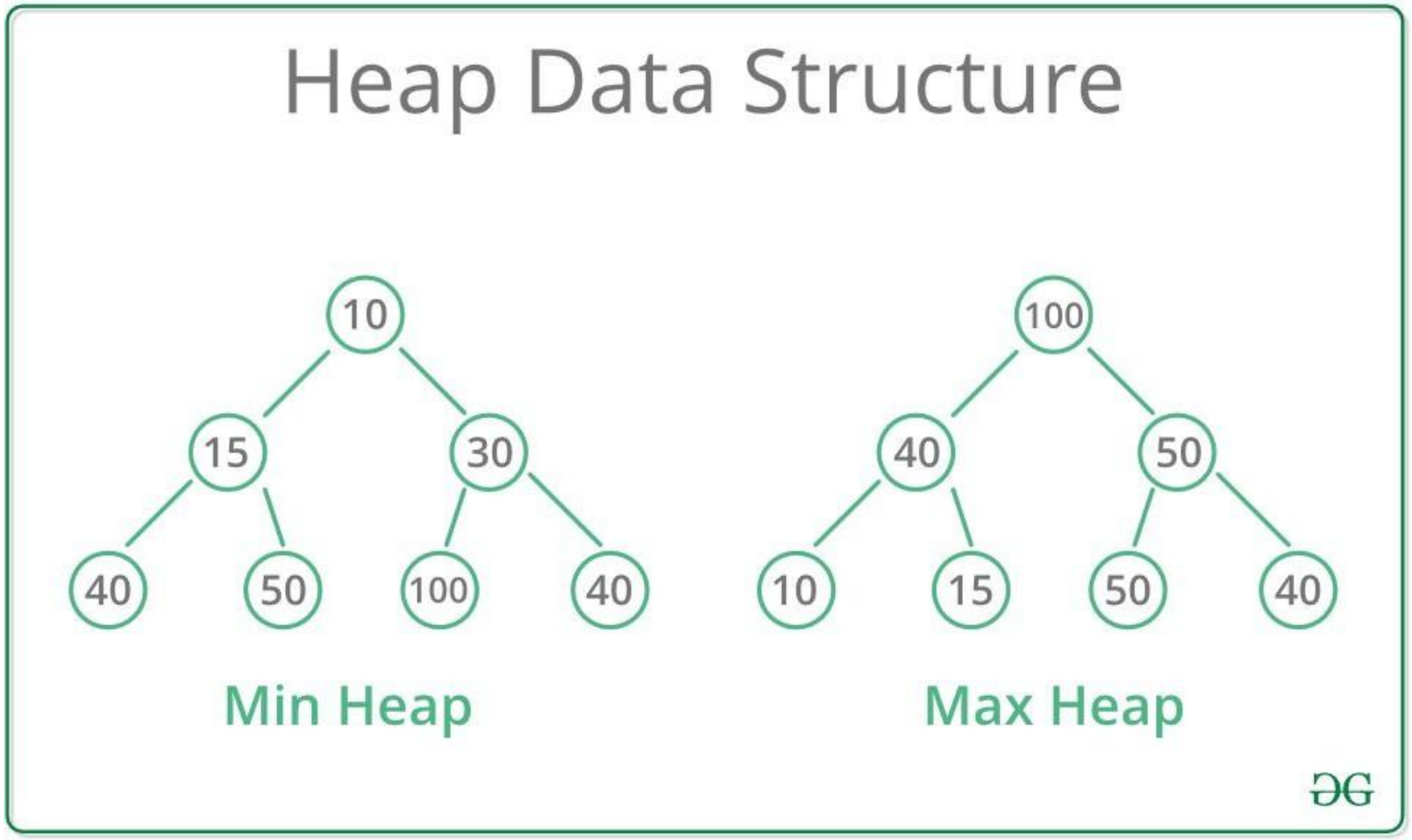
Requirements

- Maintain a keyword heap
- Heap order

  n.count >= n.parent.count (MIN-HEAP)

- For the heap structure, you can
  - Use java.util.ArrayList
  - java.util.PriorityQueue
  - Or develop it by yourself

Heap



From https://www.geeksforgeeks.org/heap-data-structure/

Operations

operations	description
add(Keyword k)	Insert a keyword k to the heap (use offer())
peek()	Output the keyword with the minimal count (use peek())
removeMin()	Return and remove the keyword of the root (the one with the minimal count) (use poll())
output()	Output all keywords in order

Keyword

- A keyword is a tuple of [String *name*, Integer *count*, Double *weight*]
  - For example:

    {
        name: “Fang”,
        count: 3,
        weight: 5.5
    }
- A keyword should output in format **[name, count, weight]** :
  - [Fang,3,5.5]

I/O Example: add

- To do: Insert a keyword [k,c,w] to the heap in order
- Input:
  - Token1 : a constant “add”
  - Token2 : keyword name **k**
  - Token3 : keyword count **c**
  - Token4 : keyword weight **w**
  - EX: add Fang 3 0.5

I/O Example: peek

- To do: Output the keyword with the minimal count
- Input:
  - Token1 : a constant “peek”
  - EX: peek
- Output:
  - If heap is empty, then output “InvalidOperation”:

    InvalidOperation
  - If it is legal to peek:

    [NCCU,4,9.9]

I/O Example: removeMin

- To do: Output and Remove the keyword of the root
- Input:
  - Token1 : a constant “removeMin”
  - EX: removeMin
- Output:
  - If heap is empty, then output “InvalidOperation”:

    InvalidOperation
  - If it is legal to remove:

    [NCCU,4,9.9]

I/O Example: output

- To do: Output and Remove all the keywords in order (ascending)
- Input:
  - Token1 : a constant “output”
  - EX: output
- Output:
  - If heap is empty, then output “InvalidOperation”:

    InvalidOperation

  - If heap is not empty:

    [NCCU,4,9.9] [MIS,5,6.8] [DS,6,1.6]

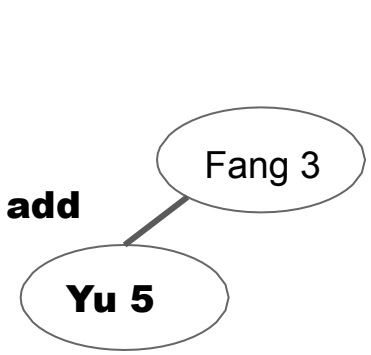
Input file

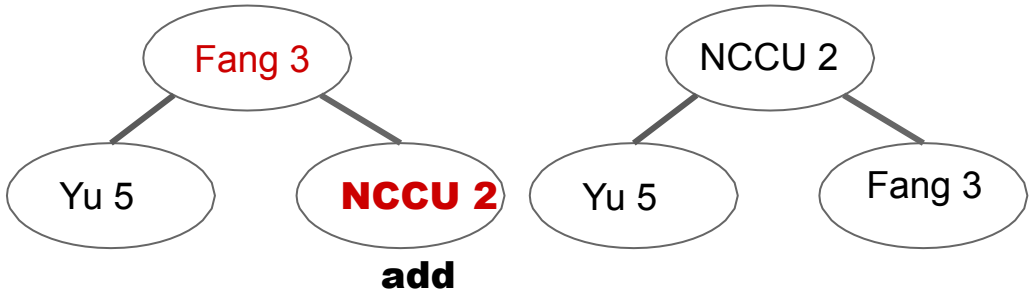
- You need to read the sequence of operations from a txt file
- The format is firm
- Raise an exception if the input does not match the format

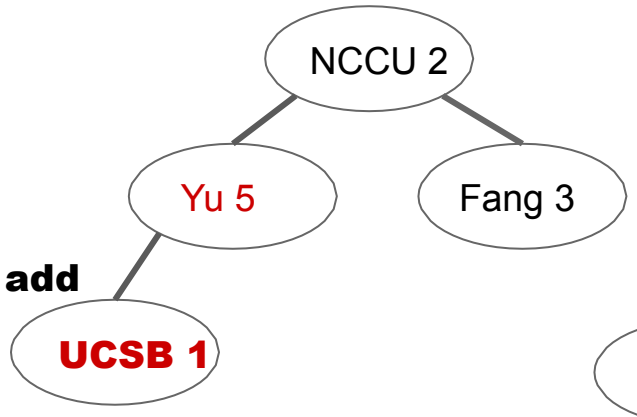
```
add Fang 3 1.2
add Yu 5 1.8
add NCCU 2 0.6
add UCSB 1 1.9
peek
add MIS 4 2.2
removeMin
add Badminton 1 0.6
output
```

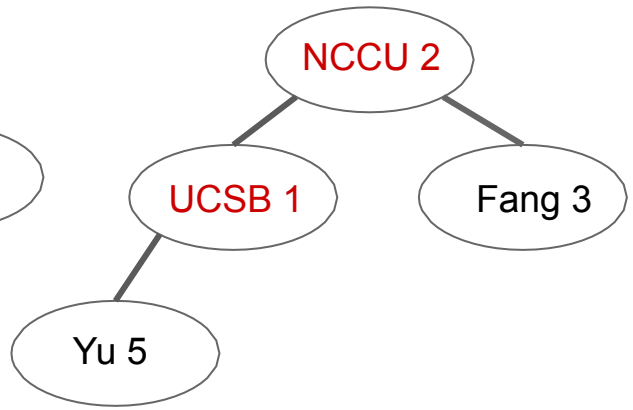
Keyword Heap

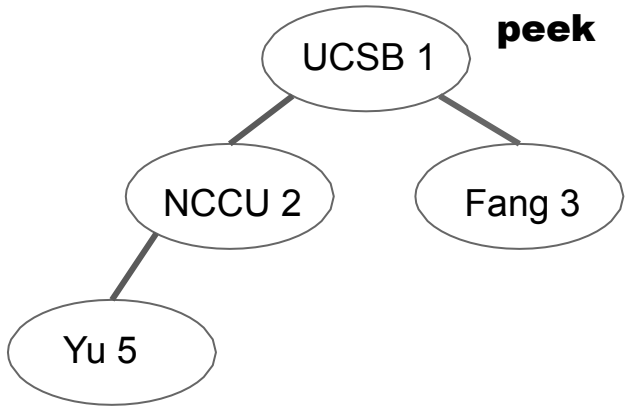
```
add Fang 3 1.2
add Yu 5 1.8
add NCCU 2 0.6
add UCSB 1 1.9
peek
add MIS 4 2.2
removeMin
add Badminton 1 0.6
output
```

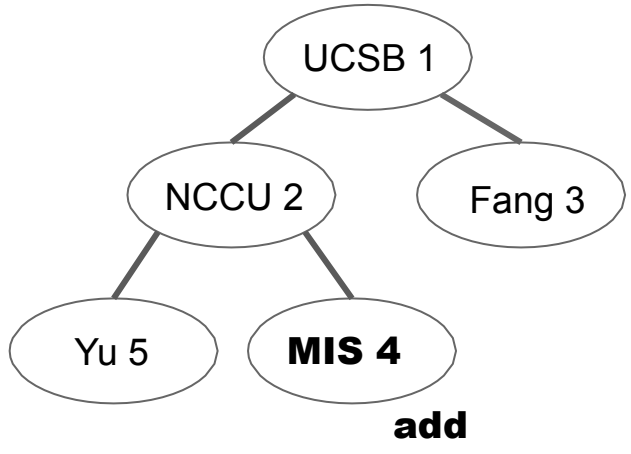






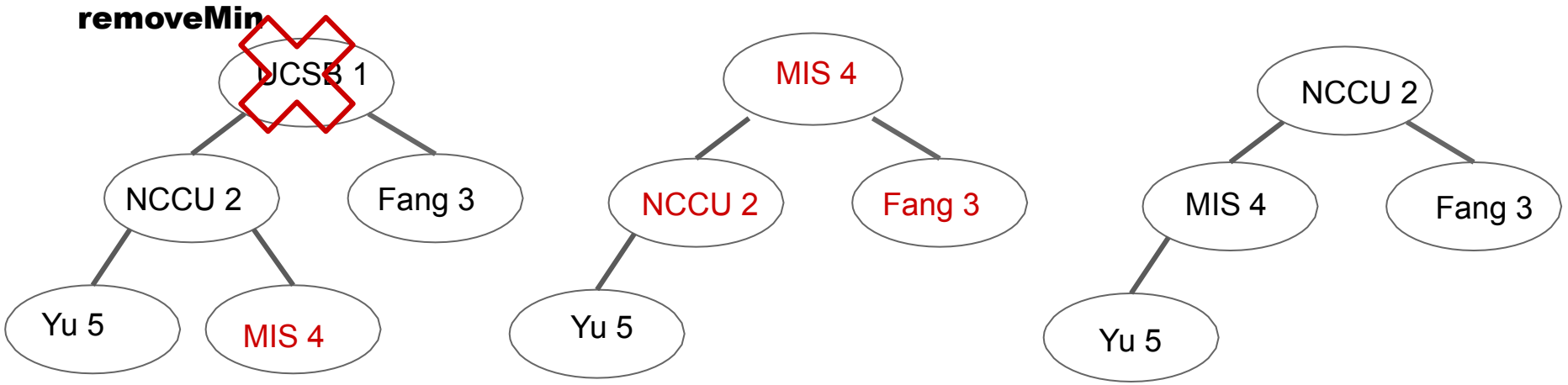


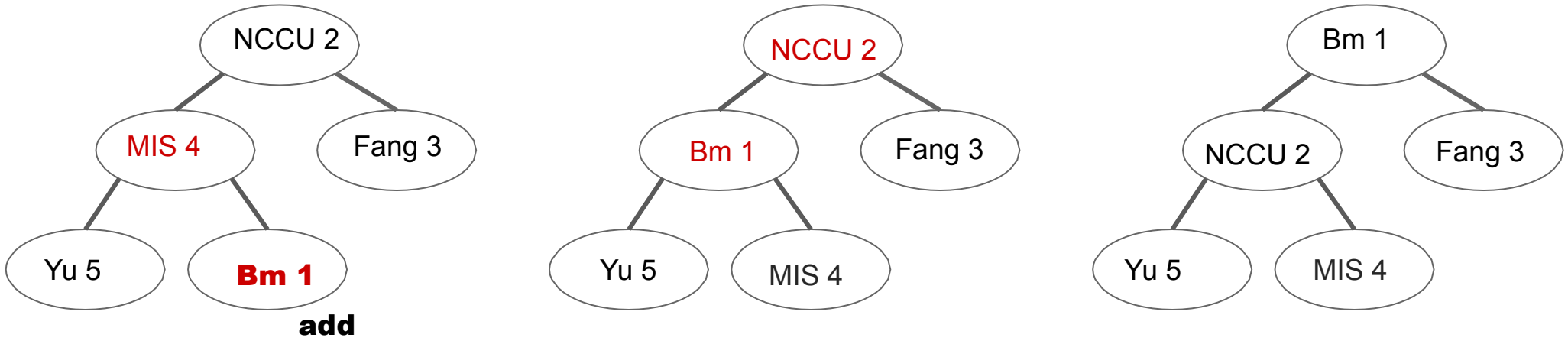




Keyword Heap

```
add Fang 3 1.2
add Yu 5 1.8
add NCCU 2 0.6
add UCSB 1 1.9
peek
add MIS 4 2.2
removeMin
add Badminton 1 0.6
output
```





Output

```
[UCSB,1,1.9]
[UCSB,1,1.9]
[Badminton,1,0.6][NCCU,2,0.6][Fang,3,1.2][MIS,4,2.2][Yu,5,1.8]
```

No Bonus for this HW