HW4

蔣其叡 111356024@nccu.edu.tw
陳卉縈 112356043@nccu.edu.tw

HW4

HW4 (Due on 10/13)

Maintain an ordered keyword list.

- A keyword is a tuple of [String name, Integer count, Double weight]
- Keep the list in order by its count while adding or deleting elements
- For the list structure, you can
  - Use java.util.ArrayList, or
  - java.util.LinkedList, or
  - Develop it by yourself
- Given a sequence of operations in a txt file, parse the txt file and execute each operation accordingly

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]

Add and Output

operations	description
add(Keyword k)	Insert k to the list in order
outputIndex(int i)	Output the ith keyword in the list
outputCount(int c)	Output all keywords whose count is equal to c
outputHas(string s)	Output all keywords whose name contains s
outputName(string s)	Output all keywords whose name is equal to s
outputFirstN(int n)	Output the first n keywords
outputScore()	Output the score of the whole list

I/O Example: Add

- To do: Insert a keyword [k,c,w] to the list in order
- Input:
  - Token1 : a constant “add”
  - Token2 : keyword name **k**
  - Token3 : keyword count **c**
  - Token4 : keyword weight **w**
  - EX: add Fang 3 1.5
- Smaller count placed in the front. If equal, smaller weight is placed in the front.

```
[MIS,2,1.2] [UCSB,2,2.2] [Food,3,0.1] [Data,3,0.3] [NCCU,3,0.8] [Fang,3,1.5] [Structure,4,2.1] [Badminton,4,2.3] [Yu,5,1.2]
```

I/O Example: outputIndex

- To do: Output the ith keyword in the list
- Input:
  - Token1 : a constant “outputIndex”
  - Token2 : an index **i** in our keyword list
  - EX: outputIndex 3
- Output:
  - If **i** is out of bound, simply output a line of “InvalidOperation”:

    InvalidOperation

  - If **i** is legal:

    [NCCU,4,9.9]

I/O Example: outputCount

- To do: Output all keywords whose count is equal to c in order
- Input:
  - Token1 : a constant “outputCount”
  - Token2 : an integer **c**
  - EX: outputCount 4
- Output:
  - If there is no keyword whose count is equal to **c,** simply output a line of constant “NotFound”:

    NotFound

  - If there are any (separated by one space):

    [OK,4,2.2] [MIS,4,3.3] [NCCU,4,9.9]

I/O Example: outputHas

- To do: Output all keywords whose name contains s
- Input:
  - Token1 : a constant “outputHas”
  - Token2 : a pattern string **s**
  - EX: outputHas ang
- Output:
  - If there is no keyword whose name contains **s,** simply output a line of constant “NotFound”:

    NotFound

  - If there are any (separated by one space):

    [Stanger,4,2.2] [Rang,4,3.3] [Fang,4,9.9]

I/O Example: outputName

- To do: Output all keywords whose name is equal to s
- Input:
  - Token1 : a constant “outputName”
  - Token2 : a string **s**
  - EX: outputName Fang
- Output:
  - If there is no keyword whose name is equal to **s,** simply output a line of constant “NotFound”:

    NotFound
  - If there are any (separated by one space):

    [Fang,4,9.9]

I/O Example: outputFirstN

- To do: Output the first N Keywords, if N <= size of list
- Input:
  - Token1 : a constant “outputFirstN”
  - Token2 : an signed integer **N**
  - EX: outputFirstN 3
- Output:
  - If N > **size of keyword list,** simply output a line of constant “InvalidOperation”:

    InvalidOperation
  - If N is legal (separated by one space):

    [Stanger,4,2.2] [Rang,4,3.3] [Fang,4,9.9]

I/O Example: outputScore

- To do: Output the score of the whole list
  - **Σ(count*weight)**
- Input:
  - Token1 : a constant “outputScore”
  - EX: outputScore
- Output:
  - Simply output a line of score
  - EX: 108.5

Delete

operations	description
deleteIndex(int i)	Delete the ith keyword in the list
deleteCount(int c)	Delete all keywords whose count is equal to c
deleteHas(string s)	Delete all keywords whose name contains s
deleteName(string s)	Delete all keywords whose name is equal to s
deleteFirst(int n)	Delete the first n keywords

I/O Example: deleteIndex

- To do: Delete the ith keyword in the list
- Input:
  - Token1 : a constant “deleteIndex”
  - Token2 : an index **i** in our keyword list
  - EX: deleteIndex 3

I/O Example: deleteCount

- To do: Delete all keywords whose count is equal to c
- Input:
  - Token1 : a constant “deleteCount”
  - Token2 : an integer **c**
  - EX: deleteCount 4

I/O Example: deleteHas

- To do: Delete all keywords whose name contains s
- Input:
  - Token1 : a constant “deleteHas”
  - Token2 : a pattern string **s**
  - EX: deleteHas ang

I/O Example: deleteName

- To do: Delete all keywords whose name is equal to s
- Input:
  - Token1 : a constant “deleteName”
  - Token2 : a string **s**
  - EX: deleteName Fang

I/O Example: deleteFirstN

- To do: Delete the first N Keywords, if N <= size of list
- Input:
  - Token1 : a constant “deleteFirstN”
  - Token2 : an signed integer **N**
  - EX: deleteFirstN 2

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 1.5
add Yu 5 1.2
add NCCU 3 0.8
add UCSB 2 2.2
add MIS 2 1.2
add Badminton 4 2.3
add Food 3 0.1
add Data 3 0.3
add Structure 4 2.1
outputScore
deleteCount 3
outputCount 2
outputName Yu
deleteName Yu
outputHas a
deleteHas a
outputIndex 2
deleteIndex 4
deleteFirstN 1
outputFirstN 3
deleteAll
```

Output

```
38.5
[MIS,2,1.2] [UCSB,2,2.2]
[Yu,5,1.2]
[Badminton,4,2.3]
[Structure,4,2.1]
InvalidOperation
```