

CSE221 Data Structures (Spring 2017)  
Instructor: Prof. Won-Ki Jeong  
Due date: May 11, 2017, 11:59 pm.

### **Assignment 3: Word frequency checker**

In this assignment, you will implement map data structure to check the frequency of word occurrence in a text file. A Map is an abstract data structure that stores key-value (k,v) pairs. There are no duplicate keys in a Map. Your program reads in a text file, parse the text into words, and build a map to calculate the frequency of each word occurring in the given text. That means, the key is an input word and the value is the number of occurrence of that word in the text. Below is a simple example (actual input text could be very long).

Input text:

```
I like a coffee and you like a tea.  
We like a coffee and a tea.
```

Output word frequency:

```
i:1  
like:3  
a:4  
coffee:2  
and:2  
you:1  
tea:2  
we:1
```

Note that we convert each word to lower-case letters and ignore non-alphabetic characters to make the problem simpler.

To calculate per-word occurrence number, we scan the input text word by word and count how many times each word appears. Hash map data structure is useful for this task. We can create a word-count pair for newly appeared word with 1 count, and if that word appears again later, increase the count by one.

#### **1. Hash Map data structure (50 pts)**

Hash Map is a template class defined as below:

```

template <class KeyType, class ValType>
class MapElem
{
public:
    typedef KeyType ktype;
    typedef ValType vtype;

    KeyType key;
    ValType val;

    MapElem* link;
};

template <class HashMapElemType>
class HashMap
{
public:
    HashMap();
    ~HashMap();
private:
    ...
};

```

Hash Map class should provide the functions below:

HashMap(unsigned int c)

Constructor with initial hash table size of c.

int size()

Return the size of the key-value pair

bool isEmpty()

Return true if no key-value pair is in the map

HashMapElem\* find(const key k)

If k is in the map, return the pair (k,v) as MapElem. If find is not successful, return NULL.

void insert(const key k, const value v)

Insert a pair (k, v) in the map. If k is already in the map, change its value with v.

```
bool remove(const key k)
```

If k is in the map, remove its pair (k,v) and return true. If k is not in the map, return false.

```
void print()
```

Print all key:value pair in decreasing order of value.

```
unsigned int hashfunction(const KeyType k)
```

Returns an integer hash value converted from a key k.

You need to build a hash map with a given word as a key. You are required to implement the static hashing with the division hash function coupled with chaining for overflow handling. For a given key, you can convert it to an integer number using the algorithm discussed in the class.

## 2. Checking word frequency (50 pts)

In class WordFrequency, you need to implement at least the following functions:

```
void ReadText(const char* filename, const MapType t)
```

Read in the input text file, parse it into words, and build a word-frequency map of a given type. You may use the following code snippet for parsing input string.

```
#include <iostream>
#include <sstream>
#include <string>
using namespace std;

// s : input string
istringstream iss(s);

do
{
    string sub;
    iss >> sub;
    cout << "Substring: " << sub << endl;
} while (iss);
```

To convert each string to lower-case, you can use the following code snippet:

```
#include <algorithm>
#include <string>
```

```
std::string data = "Hello";
std::transform(data.begin(), data.end(),
               data.begin(), ::tolower);
```

```
int GetFrequency(const std::string word)
```

Find the frequency of a given word. If the word is not found, return 0.

```
void IncreaseFrequency(const std::string word)
```

Find the frequency of a given word and increase it by 1. If the word is not found, set its frequency by 1.

```
void PrintAllFrequency()
```

Print out the frequency of all words in the text in decreasing order of its occurrence (most often occurred word should be printed out first) as shown in the example of this handout. It can be simply call print() function of the hash map class.

If your implementation is correct, the skeleton main.cpp will print out the result as follows:

```
# of occurrence of word 'you' : 5
Print all result
you:5
me:4
to:4
at:3
look:3
what:3
fun:3
have:3
did:3
and:3
the:3
that:2
know:2
do:2
tell:2
mother:2
```

```
in:2
but:1
is:1
it:1
now:1
asked:1
.
.
.
```

### 3. Compile and submit

You must log in `unio6~10.unist.ac.kr` for coding and submitting the assignment. You can compile the code using the included Makefile. You can simply `make` and then the code will be compiled. The output executable name is `assign3`.

Once you are ready to submit the code, use `dssubmit` script as follows:

```
> dssubmit assign3 assign3.zip
```

You need to submit `map.txx` and `wordfrequency.cpp`.

Good luck and have fun!