The Tasks

- In this assignment, you need to implement a class template for a hash table. Name your class **`MyHash`**.
  - Similar to **BST**, this class template needs two template parameters: The key type **K** and the element type **E**.
- In this assignment, you will do word counting with a hash table. This means that, given a long text, you need to count the number of occurrence of each unique word in the text.
- Use the STL class **`string`** for the key type **K**, and **`int`** for the element type **E**.
- Use chaining (with linked lists) to handle overflow.
- The hash function itself is defined outside of **`MyHash`**. The exact hash function is problem dependent. You need to design one yourself.

The Tasks

Example input:

She sells seashells by the seashore.
The shells she sells are surely seashells.
So if she sells shells on the seashore,
I am sure she sells seashore shells.

Example output:

she	4
sells	4
seashells	2
by	1
the	3
seashore	3
...	

Notes:

- Convert all the words to lower case.
- Ignore non-letter characters.

The Tasks

You need the following functions for **`MyHash`**:

- The constructor. It takes two inputs:
  - Prototype:

    ```
    MyHash(int(*hf)(const K&), int nb);
    ```

  - The first one is a **function pointer** to the hash function. This function should take an input of type **K** and return a value of type **`int`**.
  - The second input is the number of buckets.
- Internally, you use an array of linked lists (you can use the STL template **`list`**) to store the items.
  - Each node needs to store the whole item, including the key and the element (word count).
- Define the destructor if you use dynamic memory allocation.

The Tasks

You also need the following functions for **`MyHash`**:

- The function **`get_element`**:
  - Prototype: **`E* get_element(const K&);`**
  - Returns a pointer to the element. (Return **`NULL`** if the key is not in the hash table.)
  - The purpose that we have it return a pointer is so that you can directly update the value of the element.
- The function **`insert`**:
  - Prototype: **`void insert(const K&, const E&);`**
  - Inserts a new item with the given key and element. (Replace the original item if it exists.)

The Tasks

You also need the following functions for **`MyHash`**:

- The function **`get_items`**:
  - Prototype:

    ```
    void get_items(vector<pair<K,E>>&) const;
    ```

  - Retrieves a STL vector object containing all the items in the hash table.
  - Each item (in a **`pair`** object) includes the key and the element.

Notes on the Main Function

- **Important**: In this assignment, you need to write and submit the source file containing your own `main` function.
- Define your hash function in the same source file as your `main` function.
- The `main` function should take one command-line input, which is the path to a text file.
- What you need to do in your `main` function:
  - Initialize your hash table object.
  - Parse the content of the text file, and use the hash table to keep track of the word counts.
  - Retrieve the items in the hash table using `get_items`, and print out the words and their counts.

The Guidelines (Programming Part)

- Allowed programming environment: VS2015 only.
- STL class templates: The following classes / templates are allowed: `list`, `vector`, `string`, `pair`.
- Include documentation; this will be part of your grade.
- Demo: Only a randomly selected subset of students; the list will be announced separately after the due date.

Submission

- Use E3 only.
- Name your code **`P5_xxxxxxx.h`** and **`P5_xxxxxxx.cpp`**, where **`xxxxxxx`** is your ID. Your **`main`** function is in the **`cpp`** file. The whole class template of **`MyHash`**, including its functions, need to be implemented in the header file. Do not submit any file that is not your code (such as the *.sln file). No compressed file (*.zip, *.rar, etc.) accepted.
- Due date: **1/6/2017**. There's a grace period of 3 days with 10% deduction per day. (The deduction kicks in only when you have accumulated more than three days of delay during the semester.)