Data Structures	Assignment 5
CSI 2103-02, Spring 2019	Due at 5:00pm, Thursday, June 6

Hand in your solutions electronically using YSCEC. As was announced in class, you are not allowed to use any data structure libraries, including the linked lists, stacks, queues, trees, binary search trees, heaps, graphs, and hash tables provided by JDK. When in doubt, contact the course staff.

Submit your source code(s), zipped as *yourStudentID*.zip. For example, if your student ID is 2019000000, then you must zip all your source code(s) into 2019000000.zip and submit this file. Each class should have its own .java file, of which the filename is the same as the class name. Do *not* include your student ID as part of the class names.

This assignment consists of two programming tasks.

(1) (50 points) Write a program that, given a connected graph $G = (V, E)$, determines the number of its biconnected components. Your program must read the input from input.txt in the current working directory, and output the answer to output.txt in the current working directory.

The first line of input.txt contains the number of vertices n ($\leq 2,103,000$) and the number of edges m ($\leq 5,000,000$), separated by a space. Each vertex is numbered from 0 to $n-1$. Each of the following m lines of the input file contains (the numeric IDs of) the two endpoints of each edge, also separated by a space. The input therefore consists of $m+1$ lines in total.

The output file consists of a single line containing the number of biconnected components.

You can assume that G is connected and $n \geq 2$. The entry point of your program must be Assignment51.main(). Your program must terminate within 12 seconds on the TA's computer.

Example

input.txt

```
9 11
7 2
2 5
5 1
1 4
4 8
8 1
1 3
2 3
6 7
6 0
0 7
```

output.txt

4

(2) (50 points) Write a class that implements a hash table of integers. With each integer, a string "payload" is to be associated. ***Use open addressing with linear probing.***

Your code must have two classes HEntry and HTable. The class HEntry represents each element of the hash table, and must have the following fields.

- public int i;
  Holds the integer.
- public String s;
  Holds the string associated with the integer.
- public boolean deleted;
  Is true if the entry has been "deleted"; false otherwise.

The class HTable that implements a hash table. It must have the following fields and methods.

- public HEntry[] table;
  Is the hash table. The size of the array must be 262, 139: that is, we use table[0], $\cdots$, table[262138].
- public HTable();
  Constructs an empty hash table.
- public boolean insert(int i, String s);
  If i already exists in the table, does nothing and returns false. Otherwise, i, along with the payload s, is inserted to the table and true is returned. You can assume that s is never null.
- public boolean delete(int i);
  If i exists in the table, it is deleted and true is returned. Otherwise, does nothing and returns false.
- public String query(int i);
  If i exists in the table, returns the associated payload. Otherwise, returns null.

Use the hash function that takes the remainder of the integer divided by 262, 139. For example, the first element of the table to be probed when the key is 262, 140 is table[1].

Initially, all entries of table must be null. ***Use the algorithms presented in class.*** In particular, when an item is deleted, the corresponding deleted flag must be set rather than deleting the HEntry object itself. This means that, once some table[i] is set to some non-null reference, it remains unchanged forever.

It is important to declare the fields and methods above as public, because our grading program will directly access them to check correctness.

You do not need to provide an entry point to your code. Our grading program will use your classes and check if your code works correctly. (Of course, you would need to write your own driver that uses your class to test your code before you submit your solution.)