SNU CSE: Data Structure Due Date : 11 pm, Apr 18 2019

Assignment 2

Lecturer: Prof. Satti

Number 1: Linear Probing and Hashing with Chain (60p)

Implement hash table classes `LinearProbingHashTable` and `Hashingwithchain` (`LinearProbingHashTable.java` and `Hashingwithchain.java`) with linear probing and hashing with chain supporting the methods `insert`, `remove` and `get` (as discussed in the class). Assume that the keys are integers (but stored in `String`s: to convert the keys to integers, you can use the code discussed in the class.) and the elements are `String`s.

The skeleton code (to implement the table is given partially) is provided just for `LinearProbingHashTable` But you must also consider the same skeleton for `Hashingwithchain` in the description file. If needed, you may modify pre-written part of the skeleton code. You are asked to complete it, referring to the instructions given below:

- Use linear probing in the class `LinearProbingHashTable` to handle the overflows for Linear (a probe sequence in which the gap between two probes is fixed (usually 1)).
- Using the chains to manage the collision in the class `Hashingwithchain`. Using a one-dimensional array of sorted chains.
- Define the following methods for both classes:
    - `int hashCode(String key)`, to get hash code of a given key.
    - `void insert(String key, String val)`, to insert key-value pair.
    - `String get(String key)`, to get value for a given key.
    - `void remove(String key)`, to remove key and its value.

In addition, write a Java code in `Test.java` that contains a single method `main`, so that it

- reads the input elements from the file `hash.txt`, which has an integer per line. The Keys and values are the same.
- The hash.txt is provided
- shows the result of hashing, using both pre-implemented function `printHashTable`.

- Test your code with the following hash function:

  `h(x) = x mod 7393`, using a hash table of size `7393`.

  Measure the running time (in nanoseconds) of the `get`, `insert` and `remove` methods for both `LinearProbingHashTable` and `Hashingwithchain` for provided keys (for each methods separately) in the given `indices.txt` file. Your output style should be same as follows:

  `Key is X , method is Y, running time for Z is W nanoseconds.`

  X∈ indices.txt

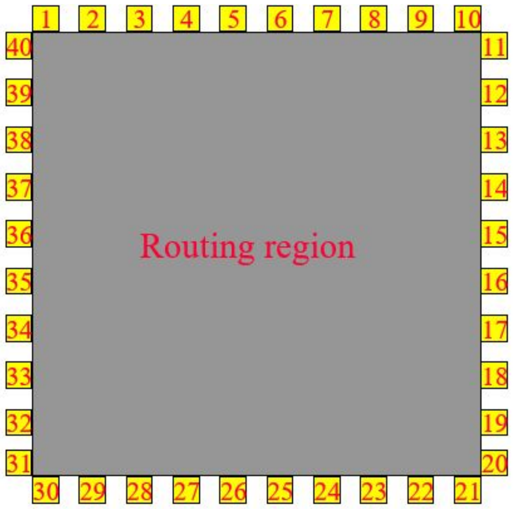
  Y∈ {get(),insert(),remove()}

  Z∈ {LinearProbingHashTable ,Hashingwithchain}

<u>Number 2: Switch Box(40p)</u>

In this problem, you need to write a program SwitchBox.java to read a set of integer pairs from files named `index1.txt` and `index2.txt`, there are no input files rather than these two, and need to check the validity of the pairs with respect to the SwitchBox Routing problem discussed in the class.

The input file contains a set of integer pairs of form `1<x,y<100`, one on each line. The output (in the screen) should be "`Routable`" if all the pairs can be connected with no `crossing'; otherwise, the output should be two pairs that crossover, you do not need to consieder multicrossovers (it means three or more lines cross each other) ,each one printed as (x,y) separated by a newline character.



The output for the first example in the above figure should be:

[Example 1]
```
cat index1.txt
2,100
3,75
3,6
14,33
40,66
java SwitchBox
Routable
```

[Example 2]
```
cat index2.txt
1,37
24,62
83,63
45,50
java SwitchBox
(1,37)
(24,62)
```

Assessment:

- Copying others work will result zero.
- If your program doesn’t compile, then you will get a zero score.
- Grading will be done in the Linux environment with Java 11. If you comment or use any Korean character, there will be an error during process of compiling.

Submission:

- By eTL. Make sure you write your name and student number as a comment on top of Test.java and Switchbox.java.
- Compress your LinearProbingHashTable.java and Hashingwithchain.java Test.java and Switchbox.java.
- Do NOT email for changing in the assignment after you have submitted.
- Late submissions will not be accepted