CS2134 HOMEWORK 10
Fall 2016
Due* 11:00 p.m. on Sun. 4 Dec 27, 2016

Be sure to include your name at the beginning of each file! Assignment 10 include a programming portion and a written part. The programming portion must compile and consist of a single file (hw10.cpp). The written portion should consist of a single typed file (hw10written) in a .pdf format. Be sure to include your name at the beginning of each file! You must hand in the file via NYU Classes.

You must hand in both files via NYU Classes.

1. Implement a hash table where collisions are resolved by linear probing. You will implement the methods `find`, `insert`, and `rehash` for the class below. Your class will `rehash` when the load factor is greater or equal to 0.5. The hash function, `hf`, will be created when you instantiate a member of this class. But `hf` will return a positive integer whose range is much larger than the table size. The book says, "you will need to perform a final `mod` operation internally after the" hash function.

```
template< class HashedObj >
class HashTable
{
  public:
     explicit HashTable( int size = 101 ):currentSize(0){ array.resize(size); }
     std::hash<HashedObj> hf; // create a hash function object
     bool contains( const HashedObj & x ) const;
     void makeEmpty( );
     bool insert( const HashedObj & x );
     bool remove( const HashedObj & x);
     enum EntryType { ACTIVE, EMPTY, DELETED };
 private:
     struct HashEntry
     {
          HashedObj element;
          EntryType info;
          HashEntry( const HashedObj & e = HashedObj(), EntryType i = EMPTY )
          : element( e), info(i) {}
     };
     vector<HashEntry> array;
     int currentSize;
     void rehash( );
};
```

*5% extra credit will be given if you turn this assignment in on Sat. Dec 3 26, 2016 at 11:00 p.m.

2. Create an *adjacency list* for the graph of the NYC subway station. You will consider a subway station `sa` *adjacent* to subway station `sb` if there is a subway train that goes from `sa` directly to `sb` without going through any other stops, or you can transfer from `sa` to `sb`. You find what subway stations are adjacent[1] to other subway stations by using the information in the files `transfers.txt` and `MTA_train_stop_data.txt`. The file `transfers.txt` is a bit confusing! Here are some of the entries:

```
from_stop_id,to_stop_id,transfer_type,min_transfer_time
101,101,2,180
103,103,2,180
104,104,2,180
106,106,2,180
107,107,2,180
108,108,2,180
109,109,2,180
110,110,2,180
111,111,2,180
112,112,2,180
112,A09,2,180
113,113,2,180
...
```

In the file `transfers.txt`, observe you can transfer from subway station 112 to subway station $A09$. Thus $A09$ is adjacent to (a neighbor of) 112.

In the file `MTA_train_stop_data.txt` you can observe that subway stop 112 is one stop away from subway stop 111, and that subway stop 112 is one stop away from subway stop 113. Thus subway stops $A09, 111$, and 113 are the stops adjacent to (neighbors of) subway stop 112.

You will create an adjacency list as an `unordered_map<string, list<string>>` instead of a `vector<list<string>>` In the next homework assignment, you will use this adjacency list in the shortest path algorithm.

Written Part

1. Using big-oh notation, what is the worst case running time for your algorithm to create the adjacency list?

2. If your hash table size, n, was equal to 20 (or 40, or 2000), why might the following hash function *not* be a good idea: $h(k) = 4k \bmod n$?

3. For the following questions: Let the hash function be `H(x) = x mod M`, where the table size (`M`) is initially 5.

   (a) if the collision strategy was `linear probing`, what would the hash table look like
   - after inserting `4371, 6173`
   - then removing `6173`
   - then inserting `3327` and `26`
   - after resizing[2] to a table size of `M = 11`
   - then inserting `4199, 4340, 9679, 1323`

   (b) if the collision strategy was `separate chaining`, what would the hash table look like

[1] By looking at the MTA_train_stop_data you can determine which stations are one away from each other. Please note that $101, 101N$ and $101S$ are the same stop. The transfers are in the file transfers.txt

[2] where all the items must be inserted into the table by rehashing

- after inserting `4371, 1323, 6173, 4199, 4344, 9679`
- then removing `6173`
- then inserting `3324`
- then resizing[3] to a table size of `M = 11`
- inserting `21`

[3] using the algorithm on the slides