Dept. of Computer Science, University of California, Davis
ECS122b Instructor: Rob Gysel 3/7/18

Programming Assignment #2*
Due date: 3/16/18 11:59pm

Programs are to be submitted using `handin` by the due date; be sure your code compiles and works on the CSIF. If you are unfamiliar with `handin` or the CSIF, see `http://csifdocs.cs.ucdavis.edu/`. Use the command:

```
handin rsgysel 122b-Program2 file1 file2 ... fileN
```

You may work in groups of up to 2 people. Programs submitted up to 24 hours late will still be accepted but incur a 10% grade penalty.

Overview

In this project, you will:

1. Implement the random matrix method for universal hashing.
2. Implement a Bloom filter.
3. Add basic building instructions to `CMakeLists.txt` for `CMake`.
4. Write sanity checks.

The development environment is identical to that used for Project1.
You do not need to create any files from scratch for this project. Your programming will consist of writing tests and implementing the body of pre-defined functions.
Throughout this project, when you compile, do the following steps in your project's root directory[1]. In the following commands, I assume you are starting in the root directory.

1. Create a directory called `build` using `mkdir`, or change directory to `build` and delete its contents with `rm -r *`.
2. Run CMake in `build` with `CMake ..`
3. Run Make in `build` with `make` (any unit tests you have defined will execute during this stage.)

*Last updated March 11, 2018
[1]The root directory will have all of your .cpp, .h files etc.

References

The following are helpful references for this project.

C++ Standard Library `http://www.cplusplus.com/reference/` (has tutorials) and `http://en.cppreference.com/w/` (terse)

Google Test Documentation This defines `EXPECT_*` and `ASSERT_*` macros that you must use for your tests.

http://cheezyworld.com/wp-content/uploads/2010/12/PlainGoogleQuickTestReferenceGuide1.pdf

CMake Documentation You should not need to deep-dive into CMake. Instead, you should be able to copy and paste CMake code and make the appropriate changes where necessary (e.g. change which source files are used, target names, etc.). However, if you feel that you need a reference, refer to `https://cmake.org/documentation/`.

Part1: Universal Hashing

Files to modify: `CMakeLists.txt`, `RandomMatrixHash.cpp`, `RandomMatrixHash.h`, `RandomMatrixHashSanityCheck.cpp`

Instructions: Complete the following steps, *in this order*.

1. Complete the code in the `RandomMatrixHash.*` files to implement the random matrix hash functions. Assume that the universe of keys are the set of `int`s (i.e. everything integer in between `std::numeric_limits<int>::min()` and `std::numeric_limits<int>::max()`, inclusive). The constructor must take in `m`, the number of slots in the hash table.
2. Write the sanity checks in `RandomMatrixHashSanityCheck.cpp`. Be sure your sanity checks function with working code and break with non-working code. Note the macros defined at the top of the file: you must use these values for your tests, which are probabilistic in nature and have error thresholds.

Part2: Bloom Filter

Files to modify: `CMakeLists.txt`, `BloomFilter.cpp`, `BloomFilter.h`, `BloomFilterSanityCheck.cpp`

Instructions: Complete the following steps, *in this order*.

1. Complete the code in the `BloomFilter.*` files to implement a Bloom filter. The constructor must take in `m`, the number of bits used in the filter, and `expectedSetSize`, the size of the set inserted into the filter. Use these values to compute `k`, the optimal number of hash functions. Use the `RandomMatrixHash` class for your hash functions.

2. Write the sanity checks in `RandomMatrixHashSanityCheck.cpp`. Be sure your sanity checks function with working code and break with non-working code. Note the macros defined at the top of the file: you must use these values for your tests, which are probabilistic in nature and have error thresholds.
3. Your sanity checks may occasionally fail for this section, which is fine. During grading we will re-run the sanity check a number of times to ensure that your sanity check generally passes.