**Course CSC 541 Advanced Data Structure**

**Spring 2019, Project 3**

**Due: April 24th 11:55pm – Hard Deadline**

Implement a structure that maintains a Bloom Filter for 2,000,000 strings with an error rate of ~0.025%, using only 2Mbyte of memory. To achieve this, you create eight bit arrays, each of 2,000,000 bits (this is, 250,000 char). For each of these, you select a random hash function hi from a universal family. To insert a string *s*, you set the *hi*(S)-th bit to one in the *i*-th bit array, for *i* = 0,…,7. To query whether a string *q* is contained in the set, you check whether *hi*(q) is one in the *i*-th bit array, for all *i*.

The structure must support the following operations

* **bf\_t \* create\_bf()** creates a Bloom filter with the above specification.
* **void insert\_bf(bf\_t \*b, char \*s)** inserts the string \*s into the Bloom filter \*b.
* **int is\_element(bf\_t \*b, char \*q)** returns 1 if the string \*q is accepted by the Bloom filter, and 0 else.

Submission instructions:

Project can be done in pairs. You should submit a single .c file through Moodle and rename it as UnityID1\_UnityID2.c. A file “**bloomfiltertest.c**” with test cases will be made available soon. Grading will be done using the code in bloomfiltertest.c, so you should integrate that into your main function to ensure you have tested your code thoroughly.

**Note** that the theoretical error rate here should be 2.55% or 0.025. However, due to rounding off errors and other slight variability due to implementation issues, a relaxed (~) error rate was given. In practical terms, ~2.5 means, not greater than 2.599. Error rates above this will be penalized up to 30%.

# Note:

**Sharing your code with others or copying code on line will be treated as academic dishonesty and be dealt with very severely.**

**Late submissions will not be accepted.**