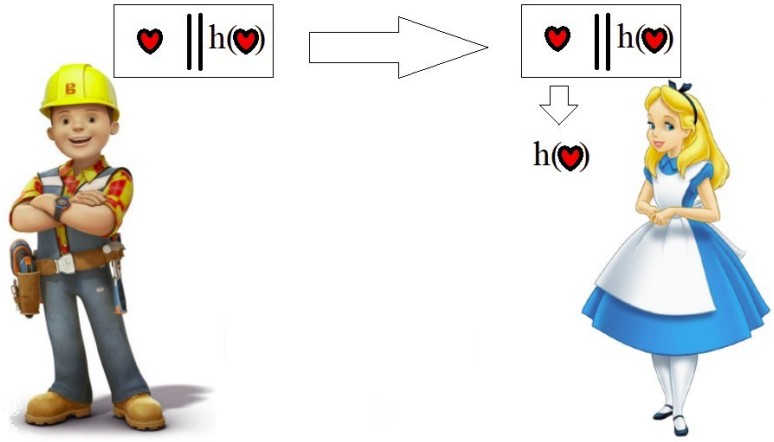
CSC 172
Data Structures and Algorithms
Spring 2019
Lab 8
Due: 04/14 11:59 PM

Introduction

We discussed hashing and the use of hash functions for storing data in hash tables. Hash functions are also used in cryptography for message authentication. In a simple scenario, Bob and Alice send encrypted messages to each other. They want to be able to decrypt messages they receive and make sure messages have not been modified during transmission. *Cryptographic* hash functions must have additional properties that are out of the scope of our course[1]. In this lab we'll investigate the property of *uniform distribution*: the probability for each element to hash into any of the slots is the same (*1/m*) for a table of size *m*. This is a desired property for any hash function.



Goal: Testing hash function *h(k) = (ak + b)* mod *m* for different choices of the parameters. Implement a rehashing of the hash table.

(7 *points*) You have to write Java code for the following:

1. compute the sequence of values *h(k)* for any sequence of integers given in input.
2. store the sequence computed in (1) in a text file called `output_sequence`
3. plot the pairs *(k, h(k))* to see how *well*/*bad* elements are distributed. Export your plots into PNG files.
4. implement a *rehashing* of the hash table when the load factor α reaches the value 0.75.

1 See more in https://en.wikipedia.org/wiki/Pseudorandom_number_generator

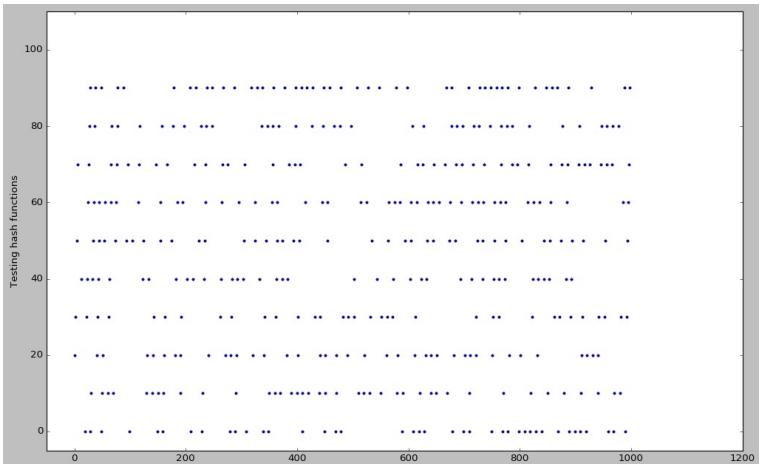
Assume your main class is **HashPlot**. The code will be executed as follows:

```
java HashPlot a b m input_sequence
```

Where

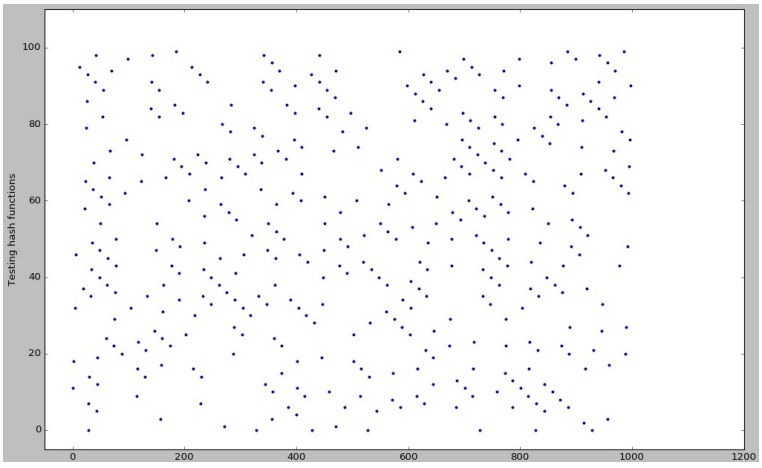
- $h(k) = (ak + b) \bmod m$ is the hash function you are testing
- `input_sequence` is a text file with a sequence of numbers divided by blank spaces.

Task 1: Identify 3 choices for *a*, *b* and *m*, for which you think the distribution of values is bad (*1.5 points*).



Example of plot for a "not so good" distribution

Task 2: Identify 3 different choices for *a*, *b* and *m*, for which you think the distribution of values is good. (*1.5 points*)



Example of plot for a "better" distribution

Note: Use sequences of **300-1000** numbers and table size (*m*) between **90 – 130**. In order to compare different hash functions you may want to look at number of collisions or other regularities in the distribution of values.

Submission

You have to submit a zip file named [Net_ID]_**Lab8.zip** including:

a) Your Java code in **HashPlot.java**

b) A README file with a description of your work, team members, the choices for the parameters in *Task 1* and *Task 2* with your comments about the quality of the distribution, your plots for each case in PNG format.

c) The input and output sequences (one text file for each) you used to get the results in (b).