Programming Assignment #3
Due Date: 12/12/2023 at 13.30

In this assignment, you will be working with hashing algorithms and solutions for collisions.

Part 0: Introduction

Assume that you are asked to build an OBS (ogrenci bilgi sistemi) for a university. The OBS must find, delete, or insert a student into the database in the constant time complexity. The university has nine faculties, and every faculty has nine departments (the name of the faculties and departments are not important). Each department has admitted 100 students for the last ten years (So, you will have 10*9*9*100 = 81.000 students). The OBS stores the students with their name, last name, department, faculty, and a unique ID randomly generated but following the format given below:

YY-FF-DD-NNN, where YY is the year in when a student is admitted, FF and DD are the ID of the faculty and the department respectively, NNN is the student ID assigned randomly to each student. NNN of two students might be the same if at least one of their YY, or FF, or DD is different.

Part I: Random Student Generator

As it is not easy to find such database, you are expected to build one. Please generate students with info stated above.
You can find student names from the link:
https://www.cs.cmu.edu/Groups/AI/areas/nlp/corpora/names/male.txt
https://www.cs.cmu.edu/Groups/AI/areas/nlp/corpora/names/female.txt

You can find student last name from the link:

https://github.com/arineng/arincli/blob/master/lib/last-names.txt

An example Student class:

```
public class Student

{

	String name;

	String lastName;

	int ID;

	String department;

	String faculty;

}
```

An example Student object:

```
Name: Abagael

LastName: Smith

ID: 222515253 // Admitted in 2022, NNN = 253

Department: Computer Engineering // 15

Faculty: Faculty of Engineering // 25
```

Part II: Hashing

a) Hash Functions: Your hash functions for table size m, should be in the following format: $h(x) = (ax + b)/1001 \equiv \mod m \mid a, b \ll m$, where a, b and, m are relatively prime numbers. You can randomly select the value of a and b.
b) Probing: The probing should in the following format: $p(x, i) = h_1(x) + i.\, h_2(x)$, where h_1 and h_2 are two different hash functions following the format given in the bullet a.

Part III: Chaining:

Even though you know there will be at most 81000 students, you have a bad judgment, and you create an array (table) with 4001 spots, which is a prime number, to store all students. Then you decide to apply chaining to store colliding students. Individually measure the required time for each bullet given below.

a) Fetch the students from the ArrayList, then hash and allocate them into the table you created.
b) Search and bring hundred random students from the table.

Part IV: Open Addressing:

Even though you know there will be at most 81000 students, you have a bad judgment, and you create an array with 401 spots, which is a prime number, to store all students. Then you decide to apply open addressing to store colliding students. When there is not enough space in your table, you apply table doubling to increase the capacity of the table but instead of doubling the size of the table you pick the smallest prime number bigger than the twice of the size of the table. For instance, after you allocate 401 students into the table, the size of the new table will be 809. You can find the list of the prime numbers from the link:
http://compoasso.free.fr/primelistweb/page/prime/liste_online_en.php

Individually measure the required time for each bullet given below.

a) Fetch the students from the ArrayList, then hash and allocate them into the table you created.
b) Search and bring hundred random students from the table.

Part V: Submission

Submit the source code, the .jar file along with a readme file that contains the explanation of how to run your code, and screenshots of the console showing how long each bullet takes.