2. Bloom Filter and hash functions

ReadMe for code:

We used the following external jar to read the strings from the excel file.

```
<dependency>
          <groupId>org.apache.poi</groupId>
          <artifactId>poi-ooxml</artifactId>
          <version>3.15</version>
     </dependency>
```

Build command: javac BloomFilter.java

Run command: java BloomFilter.java

Explanation for implementation:

- Given that we need to use 400Kbytes ie 400000 bytes ie 3200000 bits. We use a bitset which uses 1 bit per boolean value. Therefore the size of our Bloom Filter hashTable is 3200000. Hence n which is the size of the bloom filter is n=3200000

- Implementation of Bloom Filter
  - A bitset of size n was intialised.
  - 7 hash functions of the format of (ax+b)%p%n were implemented
  - For each string in the word list from the xl sheet, hash the word and set the bits in bloom filter.
  - Then, generated a list of 100 random words of size 5 and cross checked for false positivity in the bloom filter.

Observations:

- Expected Positive Rate was generated by theoretical formula: [1-e^(-km/n)]^k
- False Positive Rate is the output after running the program

No of hash functions	False Positive Rate	Expected Positive Rate
2	0.04	0.0361
3	0.02	0.0188
4	0.02	0.0133
5	0.01	0.011
6	0.0	0.0094
7	0.01	0.0102

The optimal number of hash functions were observed to be 6.