Trabalho Prático MPEI

- Bruno Caseiro – 88804
- João Simões – 88930

ContadorEstocástico.java

- 2 construtores
  - ContadorEstocastico() - p = ½
  - ContadorEstocastico(int anotherWay) - p = 2^-n
- 2 toString(), um para cada contador
- Getters

CountingBloomFilter.java

- 3 construtores
  - CountingBloomFilter(int n) - Buckets
  - CountingBloomFilter(int n, double p) - Buckets, probabilidade de “false positives”
  - CountingBloomFilter(int n, int k) - Buckets, nº de hash functions

- Getters, toString, equals
- initialize, insert, isMember, delete, count, reset

CountingBloomFilter.java

- OptimalValueK()
  - k = (m/n) * ln(2)
  - p = 0.01 (default value)

- 2 hash functions, stringToHash, stringToHashOne
  - stringToHash usa a função hashCode do Java
  - strignToHashOne usa uma função hash criada por nós

MinHash.java

- 2 construtores
  - MinHash(int n, int k) - n-shingle, nº de hash functions
  - MinHash(int n) - n-shingle, 500 hash functions
  - MinHash(int n, double e) - n-shingle, erro esperado e, 500 hash functions

- Shingling
  - charShingle(String s) e wordShingle(String s)
  - Ambos retornam um ArrayList<String>

MinHash.java

- JSim(int set1, int set2); JDis(int set1, int set2)
  - Compara os sets de índice (set1 - 1) e (set2 - 1) do ArrayList shingleSaver
- JSimMH, JDisMH
  - Compara signatures em vez dos sets

MinHash.java

- getSignature
  - id = (index no ArrayList shingleSaver) + 100
  - É usado Universal Hashing

- uniHash(int id, int a, int b)
  - p = 211; a e b variam apenas por hash function
  - (a*id + b) % p

- Getters, updateMatrix

JaccardsClinic.java

- 1) Begin diagnosis
  - Deteta a doença do paciente de acordo com os sintomas descritos
- 2) Show user count
  - Mostra a contagem de utilizadores segundo ambos os contadores estocásticos
- 3) Show disease graph
  - Pequeno gráfico com a frequência de cada doença, segundo o counting bloom filter

JaccardsClinic.java

- Sintomas.txt
  - Mini “database” com algumas doenças e os respetivos sintomas