CT437 Assignment 3 – Breaking Hash Functions

Problem 1 – Searching for collisions

- The screenshot below is from running the code using hashF1 (i.e. usingNewFunction set to false in the checkAllStringsOfCertainLength function), using 2 methods: incrementing string lengths and random string lengths.
- The collisions found for each hash function are shown on the outputs.

```
Run:  CT437_HashFunction1
"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" ...
input = Bamb0 : Hash = 1079524045
Start searching for collisions
Checking in order...
Checking all strings of length: 1
Checking all strings of length: 2
Collision found: ar
Collision found: bq
Collision found: cp
Collision found: do
Collision found: en
Collision found: fm
Checking all strings of length: 3
Collision found: aaW
Collision found: abV
Collision found: acU
Collision found: acU
Total collisions found: 10
after 547 strings checked.
Checking random string lengths...
Checking all strings of length: 38
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa1
Checking all strings of length: 39
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaM
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabL
Checking all strings of length: 32
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaA2
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaB1
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaC0
Checking all strings of length: 34
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaA00
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa0A0
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa00A
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa089
Checking all strings of length: 57
Checking all strings of length: 27
Checking all strings of length: 49
Checking all strings of length: 10
Checking all strings of length: 17
Checking all strings of length: 44
Total collisions found: 10
after 195240 strings checked.

Process finished with exit code 0
```

Problem 2 – New hash function

- The screenshot below is from the code running in the same way as problem 1, but using the new hash function.
    - Using incrementing string lengths, it took 547 checks to find 10 collisions for hashF1, but 230345 for hashF2.
    - Using random string lengths, it took 195240 checks to find 10 collisions for hashF1, but 1000122 for hashF2.
- This suggests that hashF2 is more resistant to collisions

```
Run:  CT437_HashFunction1
"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" ...
input = Bamb0 : Hash = 1079524045
Start searching for collisions
Checking in order...
Checking all strings of length: 1
Collision found: u
Checking all strings of length: 2
Collision found: A4
Collision found: B3
Collision found: C2
Collision found: D1
Collision found: E0
Collision found: 0E
Checking all strings of length: 3
Checking all strings of length: 4
Collision found: aaaQ
Collision found: aabP
Collision found: aac0
Total collisions found: 10
after 230345 strings checked.
Checking random string lengths...
Checking all strings of length: 53
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaeyyy
Checking all strings of length: 56
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaqy
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaarx
Checking all strings of length: 9
Collision found: aaaaaaaaj
Collision found: aaaaaaabi
Collision found: aaaaaaach
Checking all strings of length: 33
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaI
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabH
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaacG
Collision found: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaadF
Checking all strings of length: 44
Checking all strings of length: 30
Checking all strings of length: 49
Checking all strings of length: 3
Checking all strings of length: 9
Checking all strings of length: 43
Total collisions found: 10
after 1000122 strings checked.

Process finished with exit code 0
```