Lab2 – Readme File **Liu Yang** **yang.l**

File Includes Within This Project:

Main.java Record.java SeqReader.java StringTable.java
autofrader-results.txt JRE System Library Readme File.pdf

Algorithm Description & Project Issues:

Using Open Dressing Hash to store, search and remove records.
The record(input) in this project is *k-mres* of a DNA sequence. Every record has three variables:
key – substrings of length k.
position – a list of sequence positions at which that string occurs.
hashValue – integer result of applying the toHashValue procedure to the key.

Implement the doubling procedure

Create Record-type array named table and initialize it with size 2.
When implementing insert method, if load factor > 0.25, do the following steps:

- Call *doubleTable* method to double size of old table.
- Call *rehash* method to rehash each of the records in the old table.

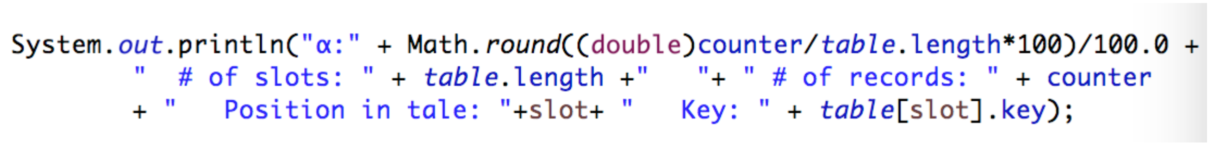
In order to compute load factor, set a public variable *counter* and initialized it to 0 to count the number of inputs in the table. After inserting a new input, *counter* increases by 1.

Challenging part

When I doubled the old table to get a new one, all records in the old table were cleaned up. Thus, I failed *rehash* implementation. It took me a great amount of time to debug. After figuring out the problem, I create a new table name *oldTable* to represent the original one. Then, rehash all records in *the oldTable* into the new table with diploid size.

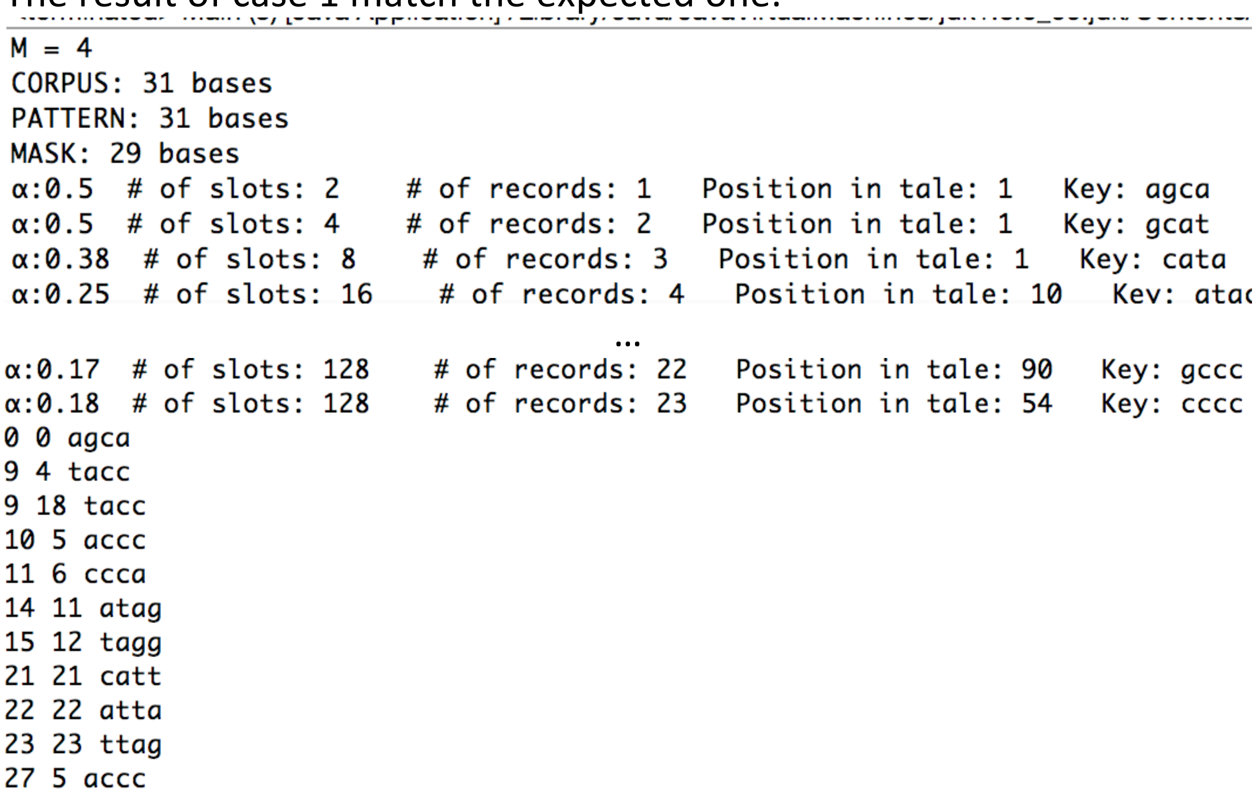
Check the Results

In order to find out bugs in my program, I printed out the results.



```
System.out.println("α:" + Math.round((double)counter/table.length*100)/100.0 +
        "  # of slots: " + table.length +"   "+ " # of records: " + counter
        + "   Position in tale: "+slot+ "   Key: " + table[slot].key);
```

The result of case 1 match the expected one.



```
M = 4
CORPUS: 31 bases
PATTERN: 31 bases
MASK: 29 bases
α:0.5  # of slots: 2    # of records: 1   Position in tale: 1   Key: agca
α:0.5  # of slots: 4    # of records: 2   Position in tale: 1   Key: gcat
α:0.38  # of slots: 8    # of records: 3   Position in tale: 1   Key: cata
α:0.25  # of slots: 16    # of records: 4   Position in tale: 10   Kev: atac
                                        ...
α:0.17  # of slots: 128    # of records: 22   Position in tale: 90   Key: gccc
α:0.18  # of slots: 128    # of records: 23   Position in tale: 54   Key: cccc
0 0 agca
9 4 tacc
9 18 tacc
10 5 accc
11 6 ccca
14 11 atag
15 12 tagg
21 21 catt
22 22 atta
23 23 ttag
27 5 accc
```

However, although the case2 to case4 printed common *k-mer* separately, the number of records didn't start from 1. Later on, I reset the preferences of Console, and got the expected results.