# Report 2

**[Hash Function]**

a) Hash function: h(k) = k mod 7

```
Print Hash Table7
[0] -> 126 -> 189 -> 371 -> 210
[1] ->  43 ->  22 -> 421 ->  57 -> 288 -> 134
[2] -> 177 -> 499 -> 429 ->  44 -> 366 -> 100 -> 471
[3] -> 346 -> 136 -> 458 -> 311 -> 437 ->  17 -> 402 ->  52 -> 381
[4] -> 179 ->  39 -> 480 ->  81 -> 263 ->  32 -> 109 -> 165 -> 200 -> 116 ->   4 -> 424
[5] -> 341 -> 306 -> 117 -> 355 -> 187 -> 425 -> 474 -> 495
[6] -> 125 -> 223 -> 461 -> 307
The longest length of the chains: 12
The average length of the chains: 7.142857
```

b) Hash function: h(k) = k mod 11

```
Print Hash Table11
[0] -> 341 ->  22 -> 187 -> 429 ->  44 -> 165 -> 495
[1] -> 177 -> 100 -> 474 -> 210
[2] ->  57 -> 189 -> 288 -> 200 -> 134
[3] -> 179 -> 355 -> 421 -> 311 -> 223 -> 366
[4] -> 136 -> 499 ->  81 -> 125 ->   4
[5] -> 346 -> 126
[6] ->  39 ->  17 -> 116 -> 402 -> 424
[7] -> 117 -> 480 -> 458 -> 425 -> 381
[8] -> 371 -> 437 ->  52
[9] -> 306 -> 471
[10] ->  43 -> 263 ->  32 -> 109 -> 461 -> 307
The longest length of the chains: 7
The average length of the chains: 4.545455
```

c) Hash function: h(k) = k mod 17

```
Print Hash Table17
[0] -> 306 -> 136 -> 187 ->  17 -> 425
[1] -> 341 ->  52 -> 307
[2] -> 189 -> 223 -> 461 -> 495
[3]
[4] -> 480 -> 429 ->   4
[5] ->  39 ->  22 -> 311
[6] -> 346 -> 499 ->  57 -> 125 -> 210
[7] -> 177 -> 126 -> 109 -> 381
[8] -> 263
[9] -> 179 ->  43 -> 366
[10] ->  44
[11] -> 402
[12] -> 437 -> 165 -> 471
[13] -> 421 ->  81 -> 200
[14] -> 371 -> 116
[15] -> 117 -> 355 ->  32 -> 100 -> 474 -> 134
[16] -> 458 -> 288 -> 424
The longest length of the chains: 6
The average length of the chains: 2.941176
```

**[Open Address Hash Table]**

a) Linear probing

```
Print Hash Table of Linear Probing
0 43
1 431
2 475
3 347
4 262
5 47
6 135
7 172
8 empty
9 empty
10 397
11 53
12 98
13 13
14 185
15 14
16 188
17 318
18 483
19 empty
20 empty
21 empty
22 empty
23 empty
24 empty
25 412
26 empty
27 371
28 71
29 27
30 159
31 empty
32 204
33 empty
34 empty
35 121
36 36
37 209
38 252
39 250
40 381
41 empty
42 300
primary cluster length: 9
Average number of probes: 1.97
```

b) Quadratic probing

```
Print Hash Table of Quadratic Probing
0 43
1 431
2 475
3 347
4 262
5 empty
6 135
7 empty
8 47
9 172
10 397
11 empty
12 98
13 13
14 14
15 empty
16 188
17 185
18 empty
19 empty
20 empty
21 318
22 381
23 empty
24 53
25 412
26 empty
27 371
28 71
29 empty
30 159
31 27
32 204
33 empty
34 empty
35 121
36 36
37 209
38 empty
39 250
40 483
41 252
42 300
primary cluster length: 5
Average number of probes: 1.70
```

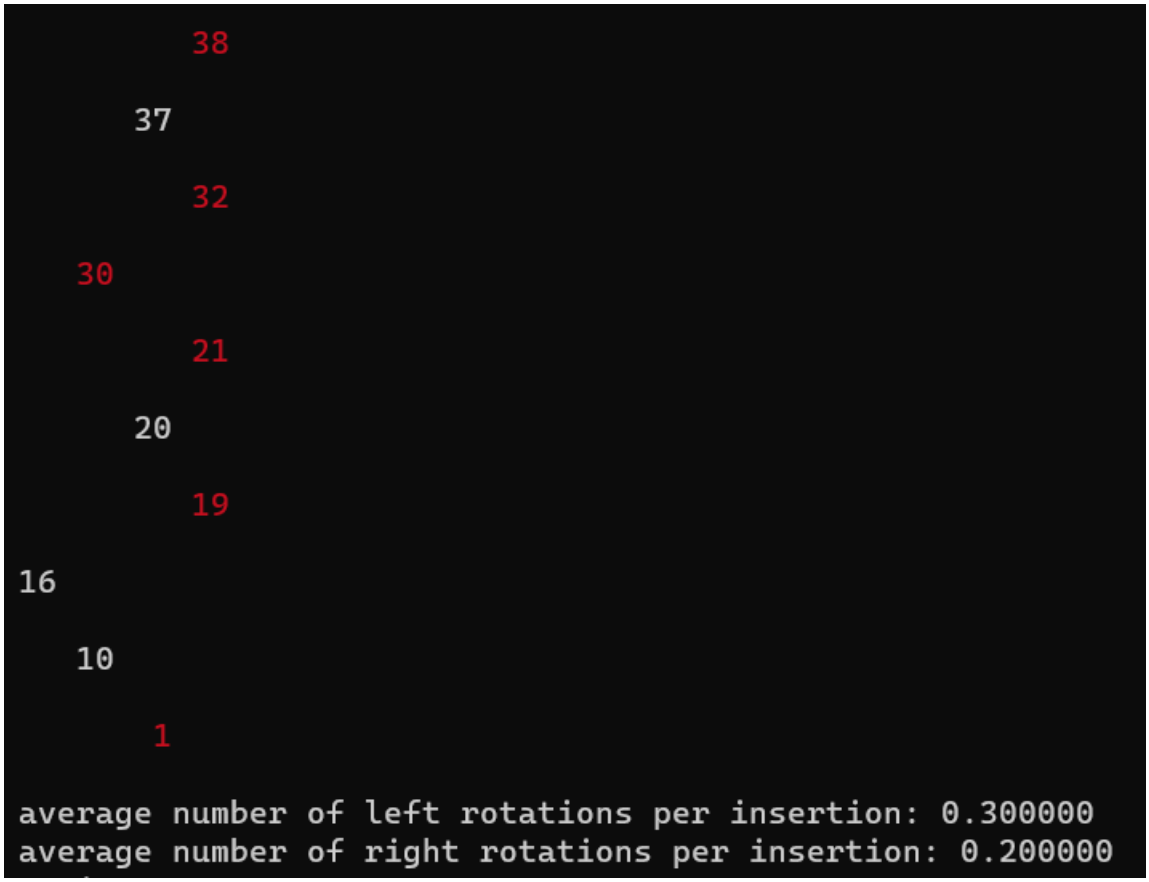
c) Double hashing

```
Print Hash Table of Double Hashing
0 43
1 431
2 475
3 347
4 262
5 172
6 135
7 empty
8 empty
9 empty
10 397
11 483
12 98
13 13
14 14
15 188
16 47
17 318
18 empty
19 empty
20 empty
21 empty
22 53
23 empty
24 empty
25 412
26 empty
27 371
28 71
29 empty
30 159
31 185
32 204
33 250
34 empty
35 121
36 36
37 209
38 252
39 empty
40 27
41 381
42 300
primary cluster length: 8
Average number of probes: 1.47
```

**[Red-Black Tree]**

Using RB-INSERT(T, k) construct a RBT with the keys in A[10] and print the T. Fill in A[10] by rand()%50.

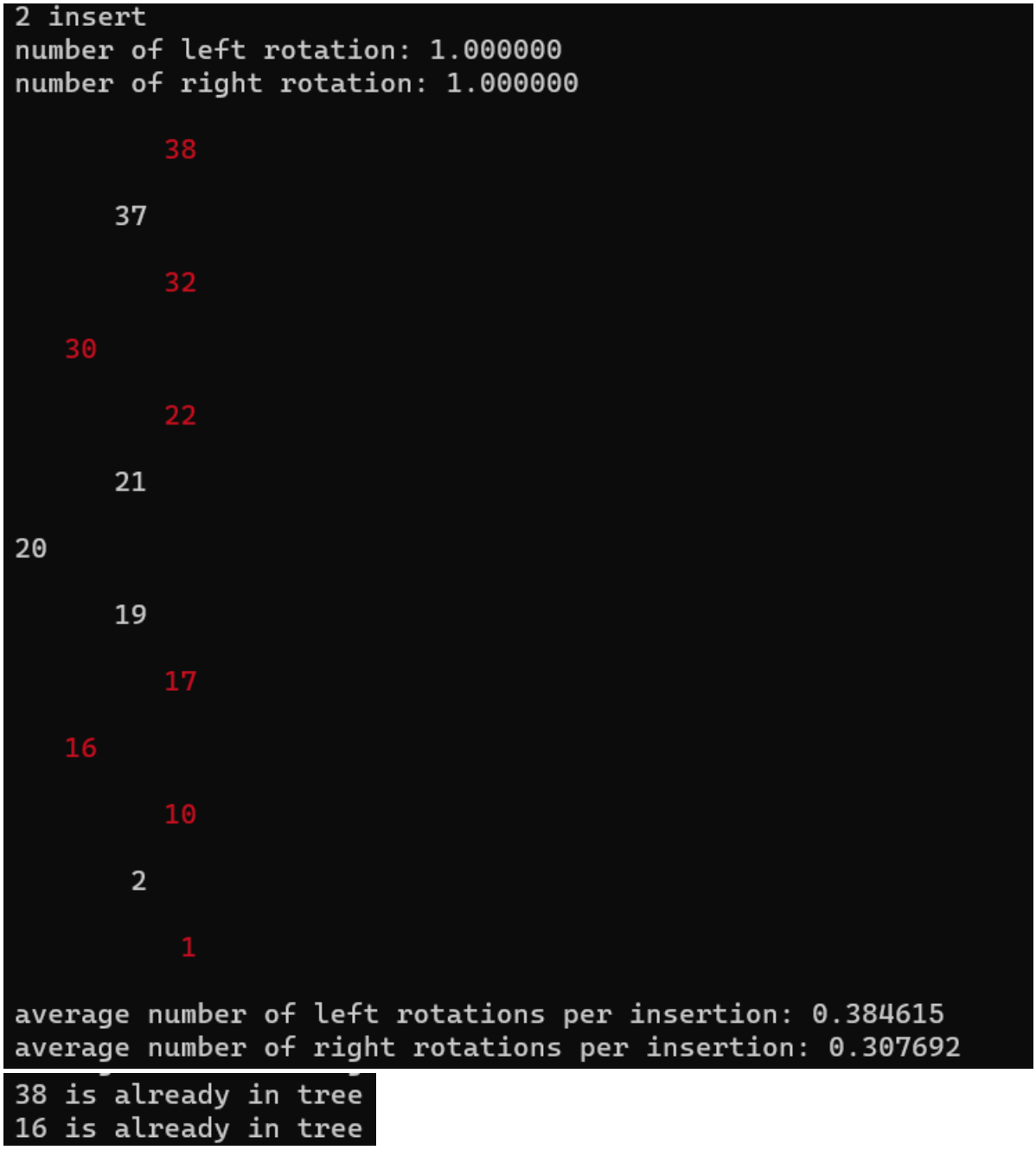
```
10 insert
number of left rotation: 0.000000
number of right rotation: 0.000000
16 insert
number of left rotation: 0.000000
number of right rotation: 0.000000
19 insert
number of left rotation: 1.000000
number of right rotation: 0.000000
1 insert
number of left rotation: 0.000000
number of right rotation: 0.000000
30 insert
number of left rotation: 0.000000
number of right rotation: 0.000000
38 insert
number of left rotation: 1.000000
number of right rotation: 0.000000
37 insert
number of left rotation: 0.000000
number of right rotation: 0.000000
21 insert
number of left rotation: 0.000000
number of right rotation: 0.000000
32 insert
number of left rotation: 0.000000
number of right rotation: 1.000000
20 insert
number of left rotation: 1.000000
number of right rotation: 1.000000
```




Execute RB-INSERT(T, 17), RB-INSERT(T, 22), RB-INSERT(T, 2), RB-INSERT(T, 38), RB-INSERT(T, 16) sequentially. Execute PRINT-BST, each time after the insertion is performed. (If a key is already in T, the insertion is ignored.)

```
17 insert
number of left rotation: 1.000000
number of right rotation: 1.000000

         38

      37

         32

   30

      21

20

      19

         17

   16

      10

          1

average number of left rotations per insertion: 0.363636
average number of right rotations per insertion: 0.272727
```

```
22 insert
number of left rotation: 0.000000
number of right rotation: 0.000000

         38

      37

         32

   30

         22

      21

20

      19

         17

   16

      10

          1

average number of left rotations per insertion: 0.333333
average number of right rotations per insertion: 0.250000
```




Execute RB-DELETE(T, 17), RB-DELETE(T, root-key), RB-DELETE(T, 38), RB-DELETE(T, root-key), RB-DELETE(T, root-key) sequentially. Execute PRINT-BST(T) each time after the deletion is performed.

```
17 delelte
number of left rotation: 0.000000
number of right rotation: 0.000000

         38

      37

         32

   30

         22

      21

20

      19

   16

         10

       2

          1

average number of left rotations per deletion: 0.000000
average number of right rotations per deletion: 0.000000
```

```
root key delete
number of left rotation: 0.000000
number of right rotation: 0.000000

         38

      37

         32

   30

      22

21

      19

   16

         10

       2

          1

average number of left rotations per deletion: 0.000000
average number of right rotations per deletion: 0.000000
```

```
38 delelte
number of left rotation: 0.000000
number of right rotation: 0.000000

      37

         32

   30

      22

21

      19

   16

         10

       2

          1

average number of left rotations per deletion: 0.000000
average number of right rotations per deletion: 0.000000
```

```
root key delete
number of left rotation: 1.000000
number of right rotation: 1.000000

      37

   32

      30

22

      19

   16

         10

       2

          1

average number of left rotations per deletion: 0.250000
average number of right rotations per deletion: 0.250000
```

```
root key delete
number of left rotation: 0.000000
number of right rotation: 0.000000

      37

   32

30

      19

   16

         10

       2

          1

average number of left rotations per deletion: 0.200000
average number of right rotations per deletion: 0.200000
```