Jordan (Yu-tin) Wang
V00786970

CSC226 Assignment #4

1.

		S	K	U	L	L	A	N	D	B	O	N	E	S
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	1	1	1	1	1	1	1	1	1	1
U	0	0	0	0	1	1	1	1	1	1	1	1	1	1
L	0	0	0	0	1	2	2	2	2	2	2	2	2	2
L	0	0	0	0	1	2	3	3	3	3	3	3	3	3
A	0	0	0	0	1	2	3	4	4	4	4	4	4	4
B	0	0	0	0	1	2	3	4	4	4	5	5	5	5
Y	0	0	0	0	1	2	3	4	4	4	5	5	5	6
B	0	0	0	0	1	2	3	4	4	4	5	5	5	6
A	0	0	0	0	1	2	3	4	4	4	5	5	5	6
B	0	0	0	0	1	2	3	4	4	4	5	5	5	6
I	0	0	0	0	1	2	3	4	4	4	5	5	5	5
E	0	0	0	0	1	2	3	4	4	4	5	5	5	6
S	0	1	1	1	2	3	4	4	4	5	5	5	6	7

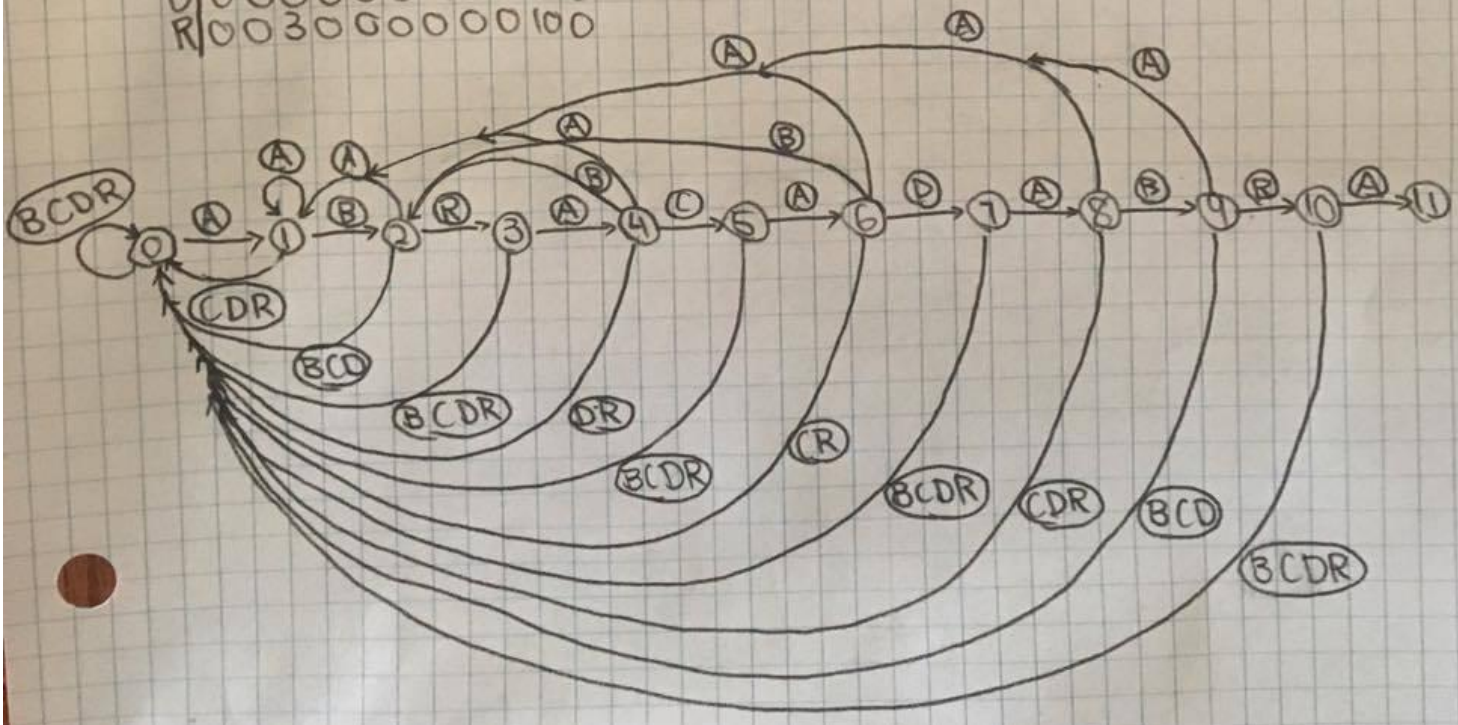
LONGEST COMMON SUBSEQUENCE : ULLABES

2.

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	right[ ]
		A	A	B	A	A	B	A	A	B	C	D	A	C	A	B	
A	-1	0	1	1	3	4	4	6	7	7	7	7	11	11	13	13	13
B	-1	-1	-1	2	2	2	5	5	5	8	8	8	8	8	8	14	14
C	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	9	9	9	12	12	12	12
D	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	10	10	10	10	10	10

3.

		A	B	R	A	C	A	D	A	B	R	A
A	1	1	1	4	1	6	1	8	1	1	11	
B	0	2	0	0	2	0	2	0	9	0	0	
C	0	0	0	0	5	0	0	0	0	0	0	
D	0	0	0	0	0	0	7	0	0	0	0	
R	0	0	3	0	0	0	0	0	0	10	0	



4.
```
printLCS(int[][] lcs, String X, String Y){
    int M = X.length-1;
    int N = Y.length-1;
    int In= lcs[M][N]-1;
    String Result;
    char Z = new [lcs[M][N]];
  while (lcs[M][N] != ∅){
    if (X.charAt(M) == Y.charAt(N)){
        Z[In] = X.charAt(M);
        In--; M--; N--;
    }else if (lcs[M][N] == lcs[M-1][N]){
        M--;
    }else{
        N--;
    }
  }
    Result = ArrayToString(Z);
    return Result;
}
```

RUNTIME : O(n+m+k)

5. change the Rabin-Karg algorithm to include 2 hashs.
   1. include characters from 1st to K, but not include K
   2. include characters from K+1 to end

   when PatHash & TxTHash are compared, insert PatHash1 and PatHash2 to TxTHash1 and TxTHash2.
   If match, assume string is correct result.