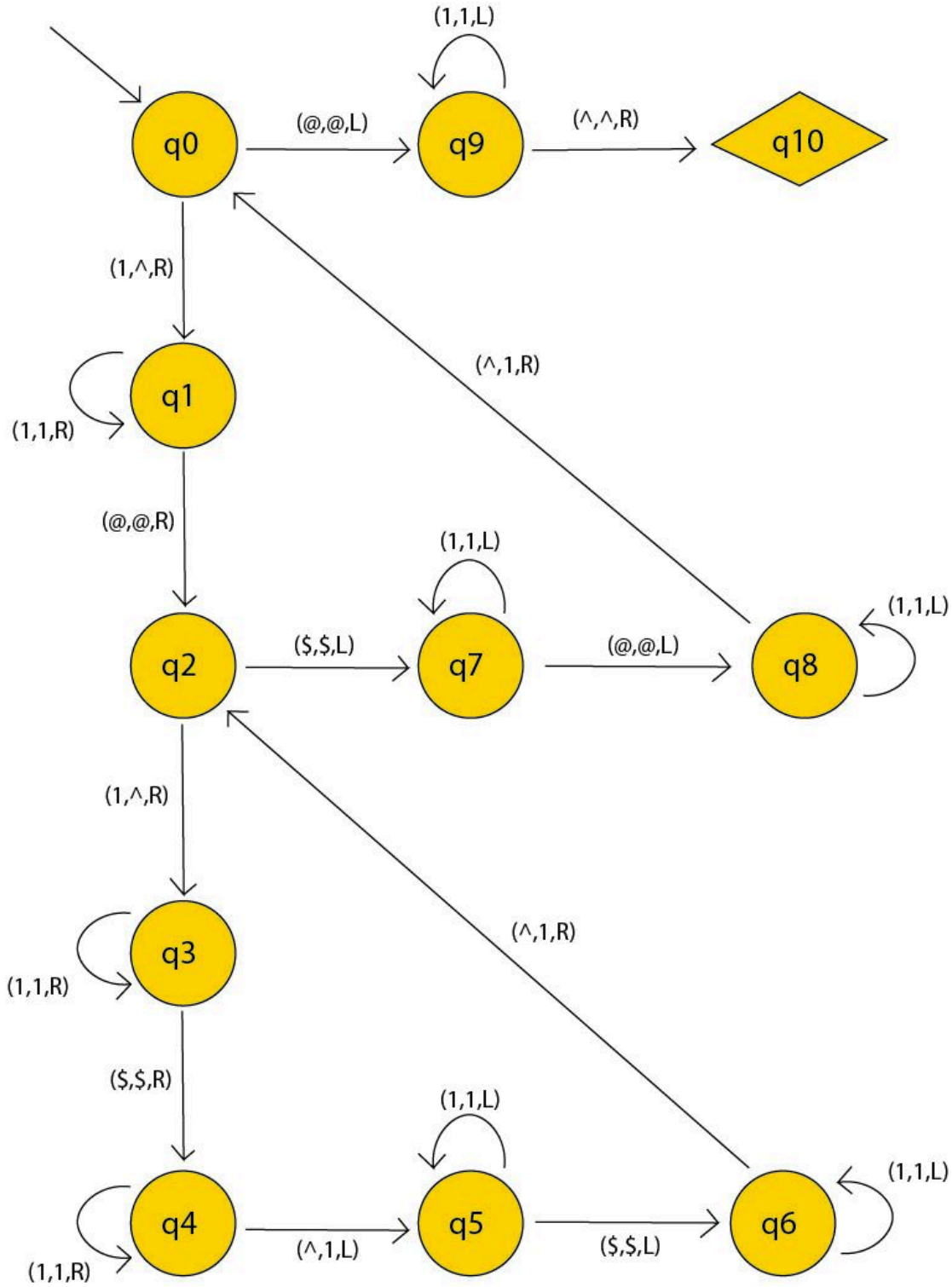
F29FB 2025 Coursework

- Shyam Sundar Velmurugan
- ssv2001@hw.ac.uk
- H00418621

Q1) Graph (1)


q0
q1
q2
q3
q4
q5
q6
q7
q8
q9
q10
(1,1,L)
(@,@,L)
(^,^,R)
(1,^,R)
(^,1,R)
(1,1,R)
(@,@,R)
($,$,L)
(@,@,L)
(1,^,R)
(^,1,R)
(1,1,R)
($,$,R)
(^,1,L)
($,$,L)
(1,1,R)

Q2) Highlighting the differences from the graph

So the symbols I have used instead of ♦ and ♥ are @ and $ respectively.

(0 , ^ , 9 , ^ , L) -> (0 , @ , 9 , @ , L)
(1 , ^ , 2 , ^ , R) -> (1 , @ , 2 , @ , R)
(2 , ^ , 7 , ^ , L) -> (2 , $, 7 , $, L)
(3 , ^ , 4 , ^ , R) -> (3 , $, 4 , $, R)
(5 , ^ , 6 , ^ , L) -> (5 , $, 6 , $, L)
(7 , ^ , 8 , ^ , L) -> (7 , @ , 8 , @ , L)

Q3) Formal definition of the proposed graph

States = {q0, q1, q2, q3, q4, q5, q6, q7, q8, q9, q10}
Symbol = {^, 1, @, $}
Start State = Lowest number state = q0

Initial State	Current Symbol	New State	New Symbol	Move
Mex q0	@	q9	@	L
Mex q0	1	q1	^	R
Mex q1	@	q2	@	R
Mex q1	1	q1	1	R
Mex q2	$	q7	b	L
Mex q2	1	q3	^	R
Mex q3	$	q4	$	R
Mex q3	1	q3	1	R
Mex q4	^	q5	1	L
Mex q4	1	q4	1	R
Mex q5	$	q6	$	L
Mex q5	1	q5	1	L
Mex q6	^	q2	1	R
Mex q6	1	q6	1	L
Mex q7	@	q8	@	L
Mex q7	1	q7	1	L

Mex q8	^	q0	1	R
Mex q8	1	q8	1	L
Mex q9	^	q10	^	R
Mex q9	1	q9	1	L

Q4) Logic of the proposed graph

States = {q0, q1, q2, q3, q4, q5, q6, q7, q8, q9, q10}

1) First we initialize at q0 (starting state)

The machine starts up at ‘q0’, which is the initial state.

In the tape if ‘@’ appears, it shifts to ‘q1’, updating for ‘^’, and then to the right.

2) Numbers Processed (q1 & q2)

q1:

Shifts to ‘q3’, keeping the blank and moves to right, if ‘^’ is encountered.

If ‘1’ is seen, it moves to ‘q2’, updating ‘1’ for blank and moves to the left.

q2:

Follows a loop in ‘q2’ but if ‘1’ is detected, it moves left.

If ‘^’ is detected, it moves to the right and remains at ‘q2’.

3) Second Symbol Sequence

At q3:

If it detects ‘1’, it moves right in a loop.

If it detects ‘$’ it moves right to ‘q4’.

At q4:

If it detects ‘1’, it moves to the right and goes back to ‘q1’.

4) Retracing & Adapting

At q5:

If it detects ‘^’, it moves left and updates to ‘1’.

If it detects ‘b’, it moves left to ‘q6’.

At q6:

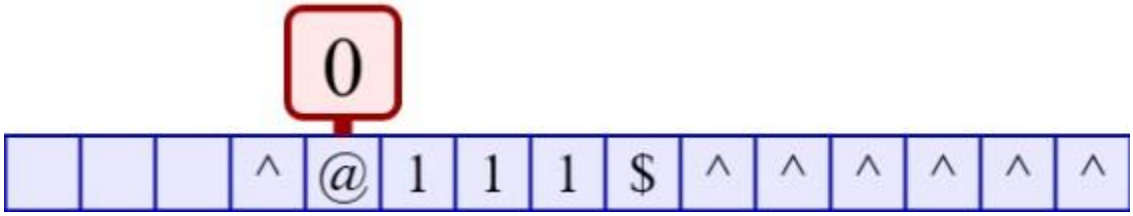
If it detects ‘1’, it moves left and loops.

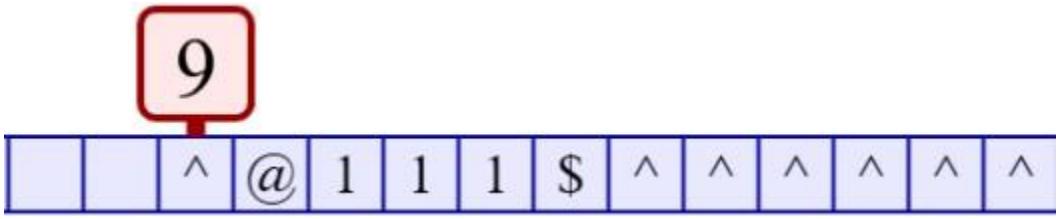
At q7:
If it detects ‘b’ or ‘1’, it moves left and loops.
At q8:
If it detects ‘a’ or ‘1’, it moves left and loops.

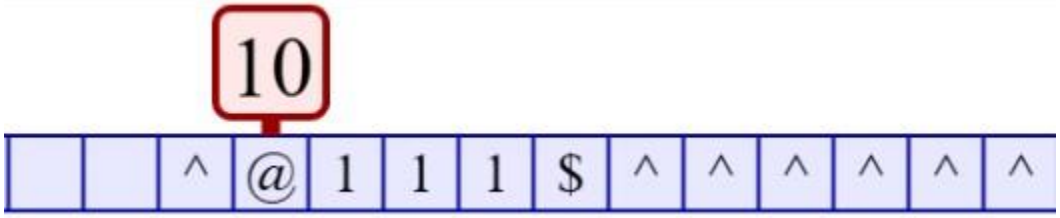
5) Taking in the input
At q9:
If it detects ‘^’, it moves right to ‘q10’.
The machine stops when it reaches the acceptable state ‘q10’.

Q6 Machine code

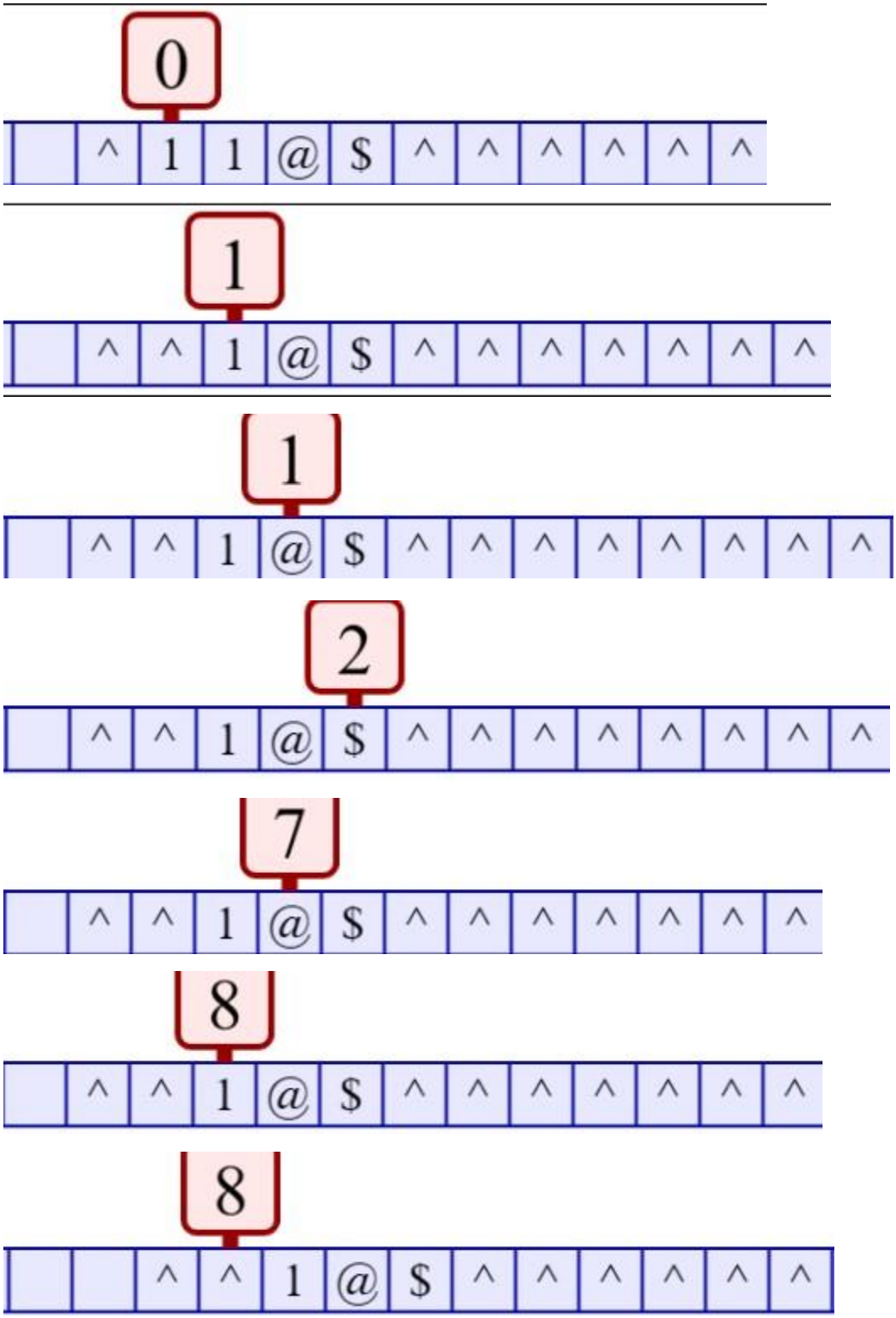
a) 0 and 3 representation

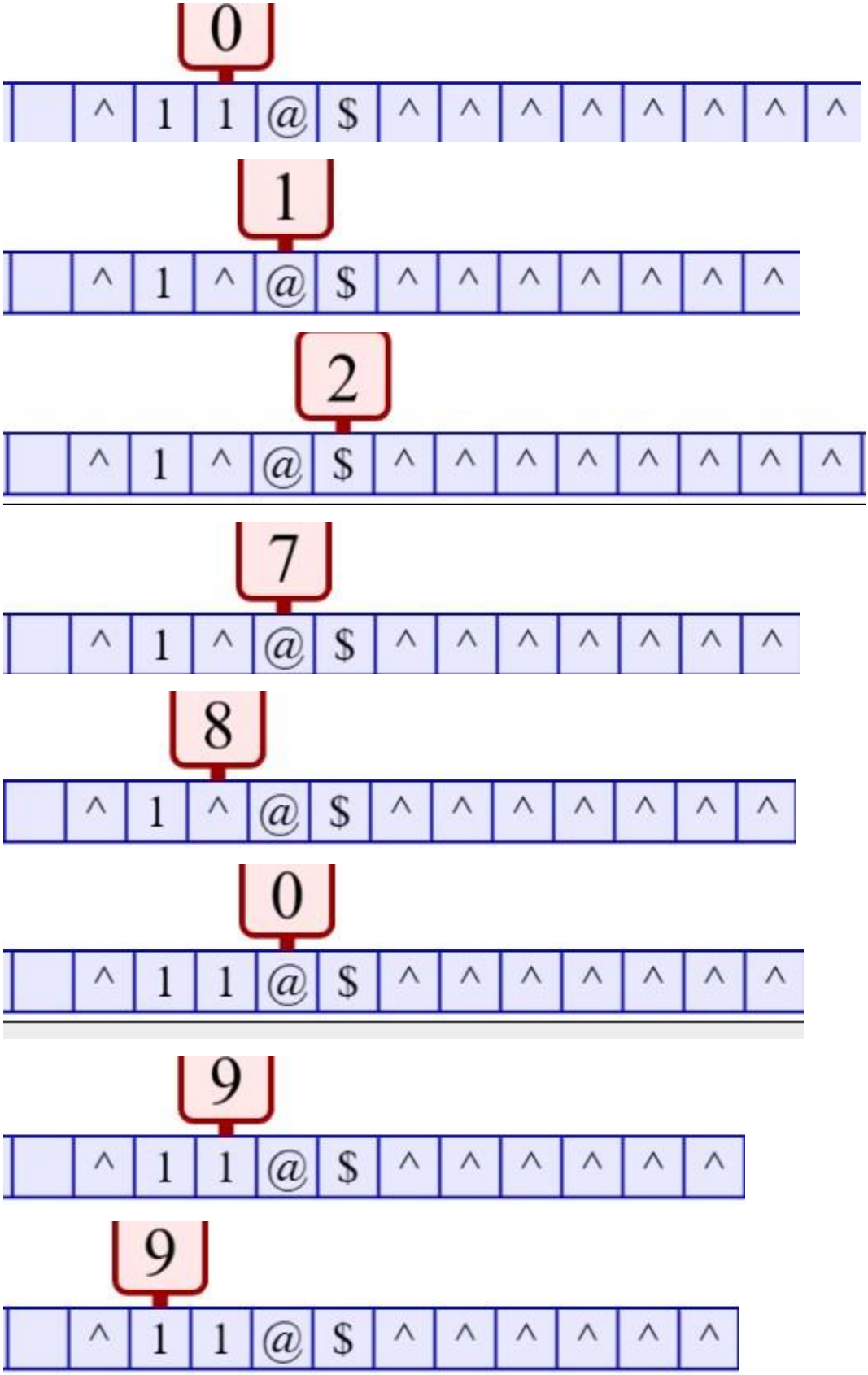




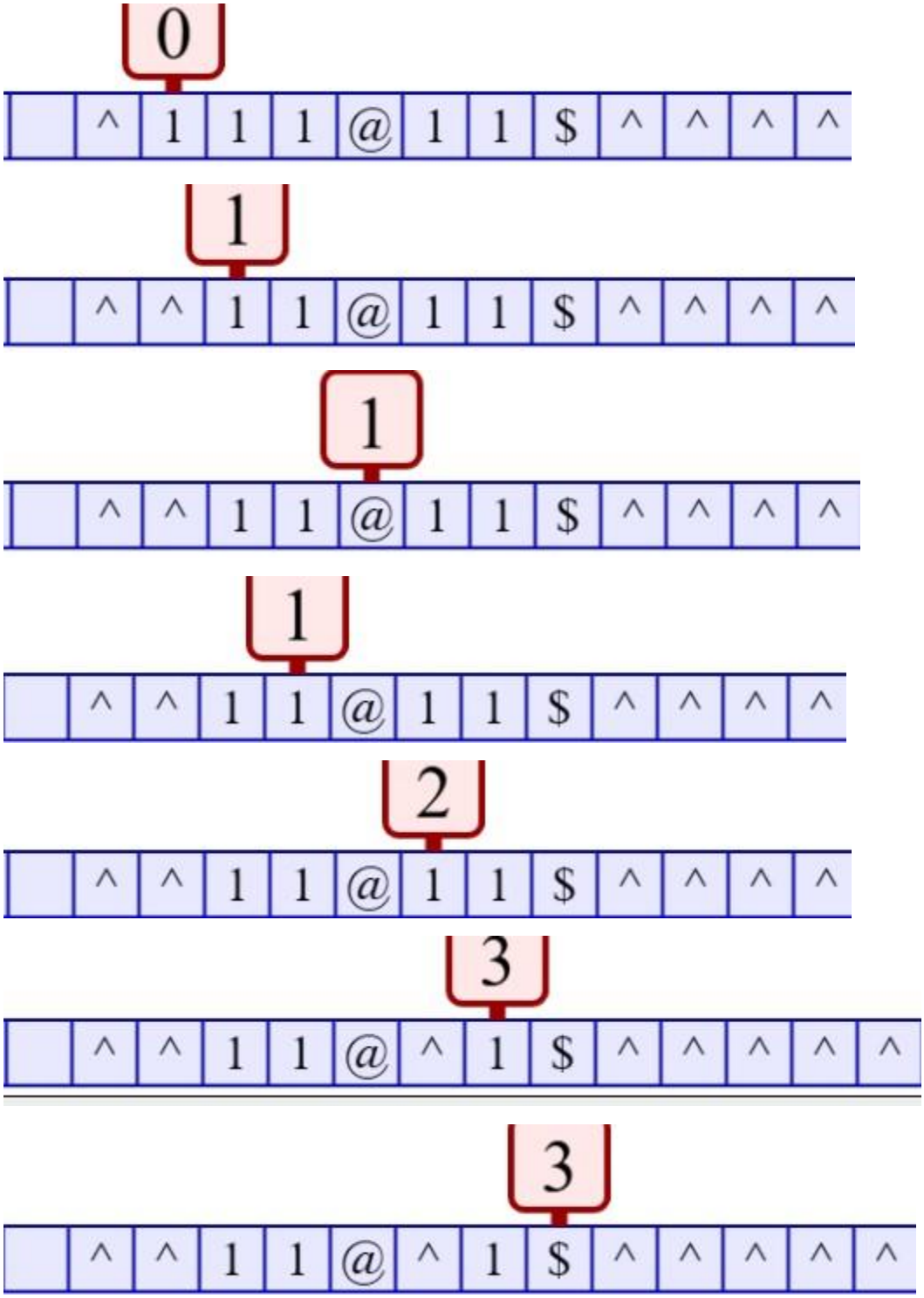


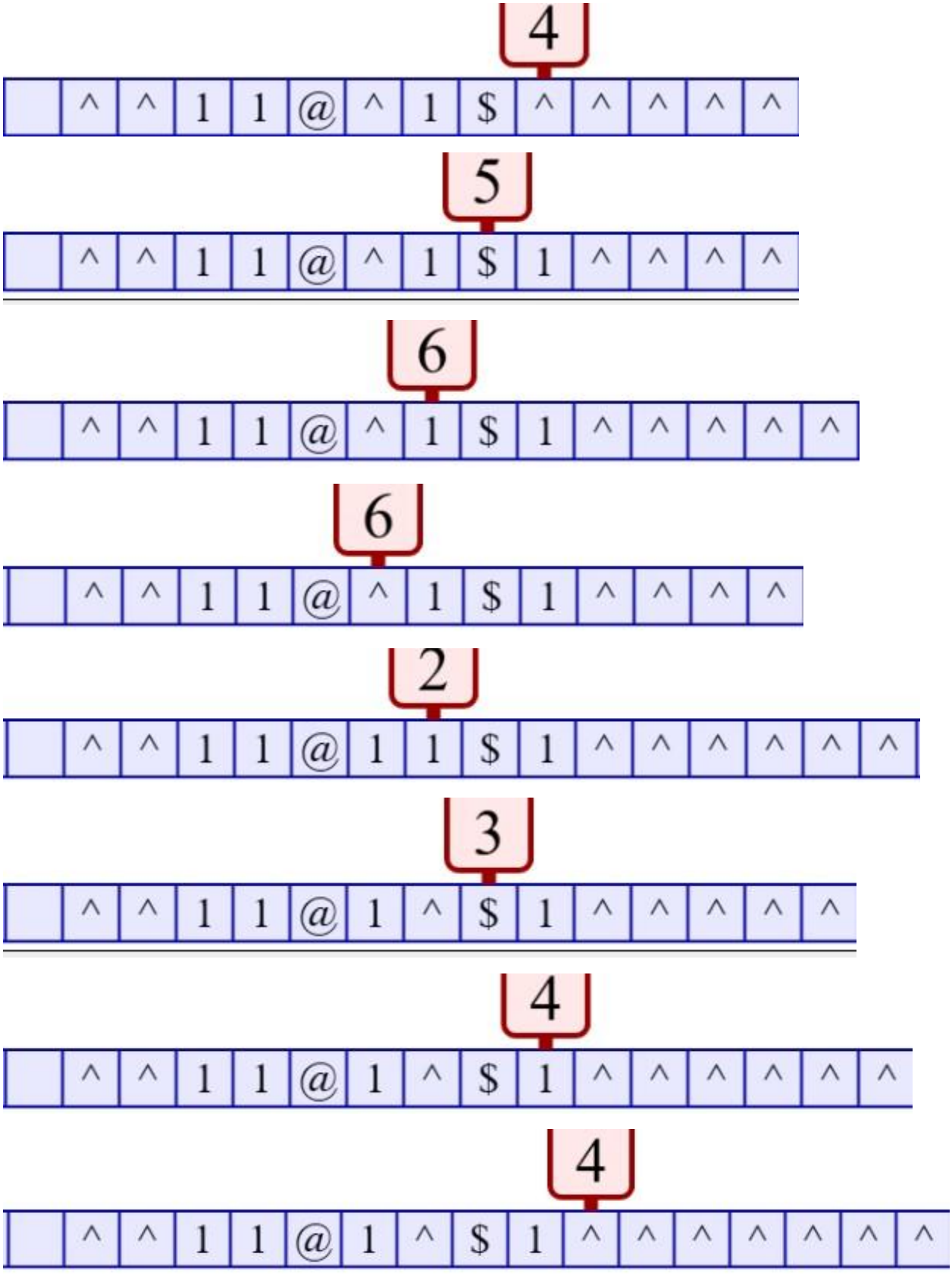
b) 2 and 0 representation
(First 15 steps)



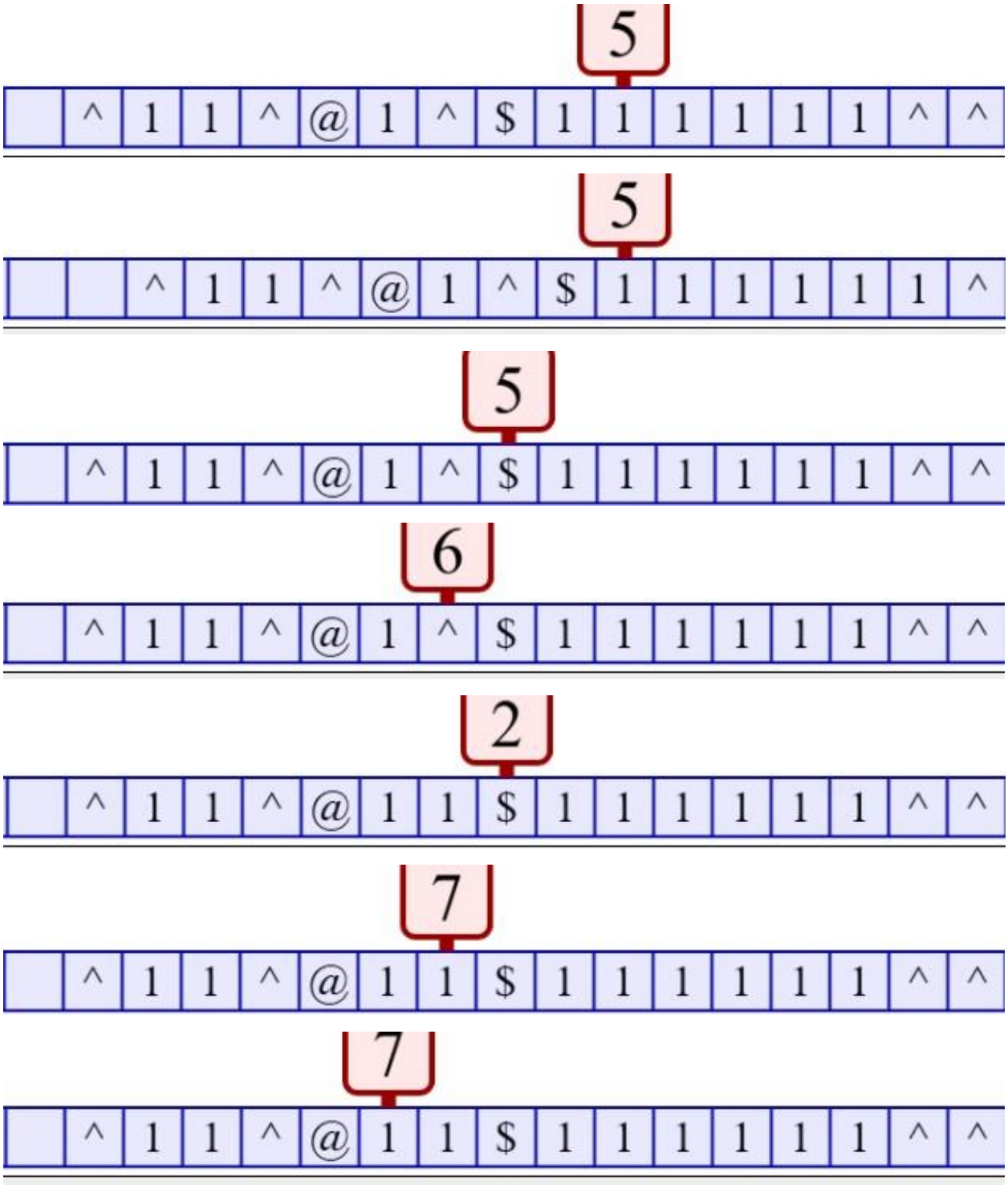

0
^ 1 1 @ $ ^ ^ ^ ^ ^ ^ ^ ^
1
^ 1 ^ @ $ ^ ^ ^ ^ ^ ^ ^
2
^ 1 ^ @ $ ^ ^ ^ ^ ^ ^ ^ ^
7
^ 1 ^ @ $ ^ ^ ^ ^ ^ ^ ^
8
^ 1 ^ @ $ ^ ^ ^ ^ ^ ^ ^
0
^ 1 1 @ $ ^ ^ ^ ^ ^ ^ ^
9
^ 1 1 @ $ ^ ^ ^ ^ ^ ^
9
^ 1 1 @ $ ^ ^ ^ ^ ^ ^

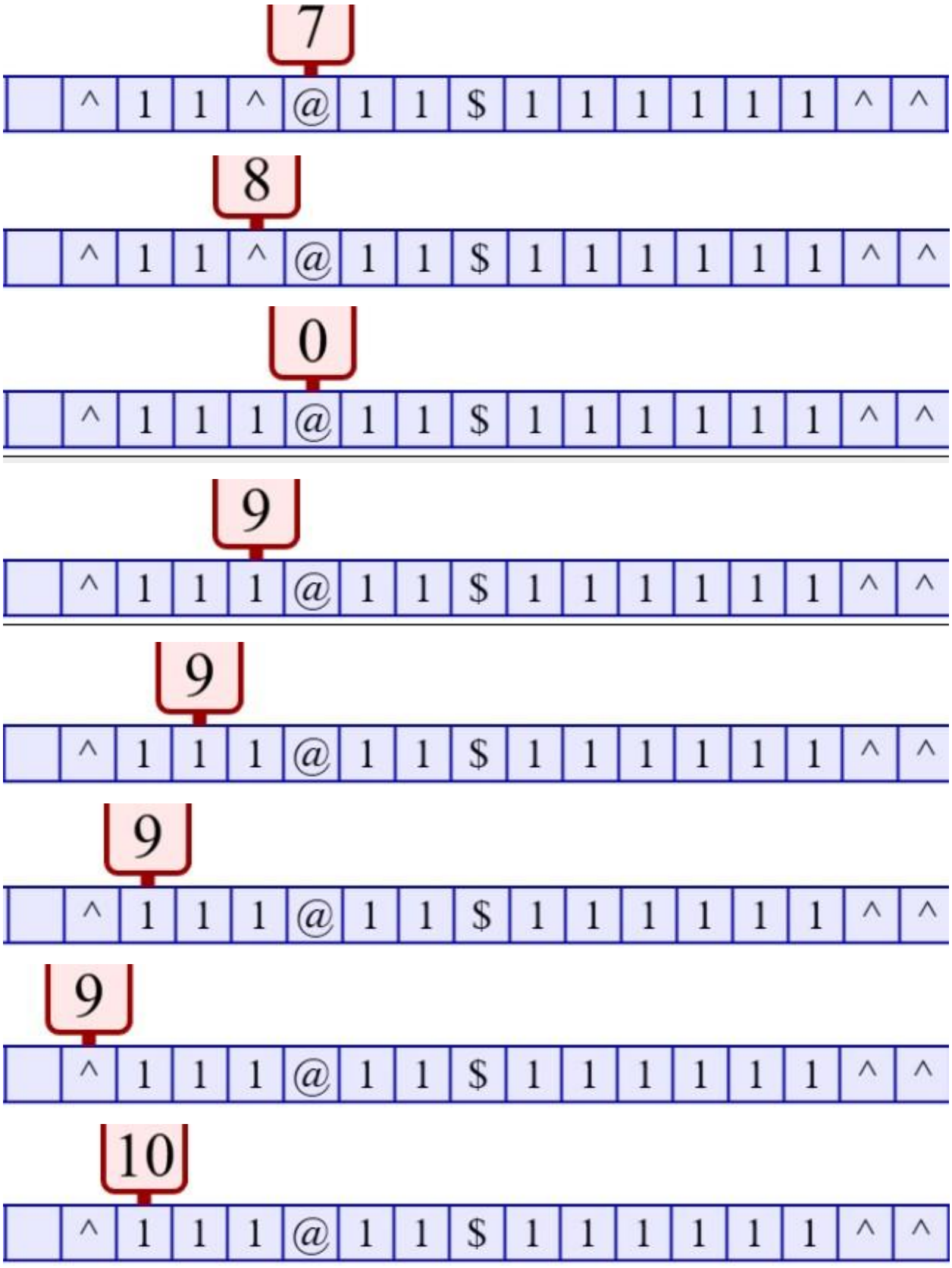
c) 3 and 2 representation
(First 15 steps)



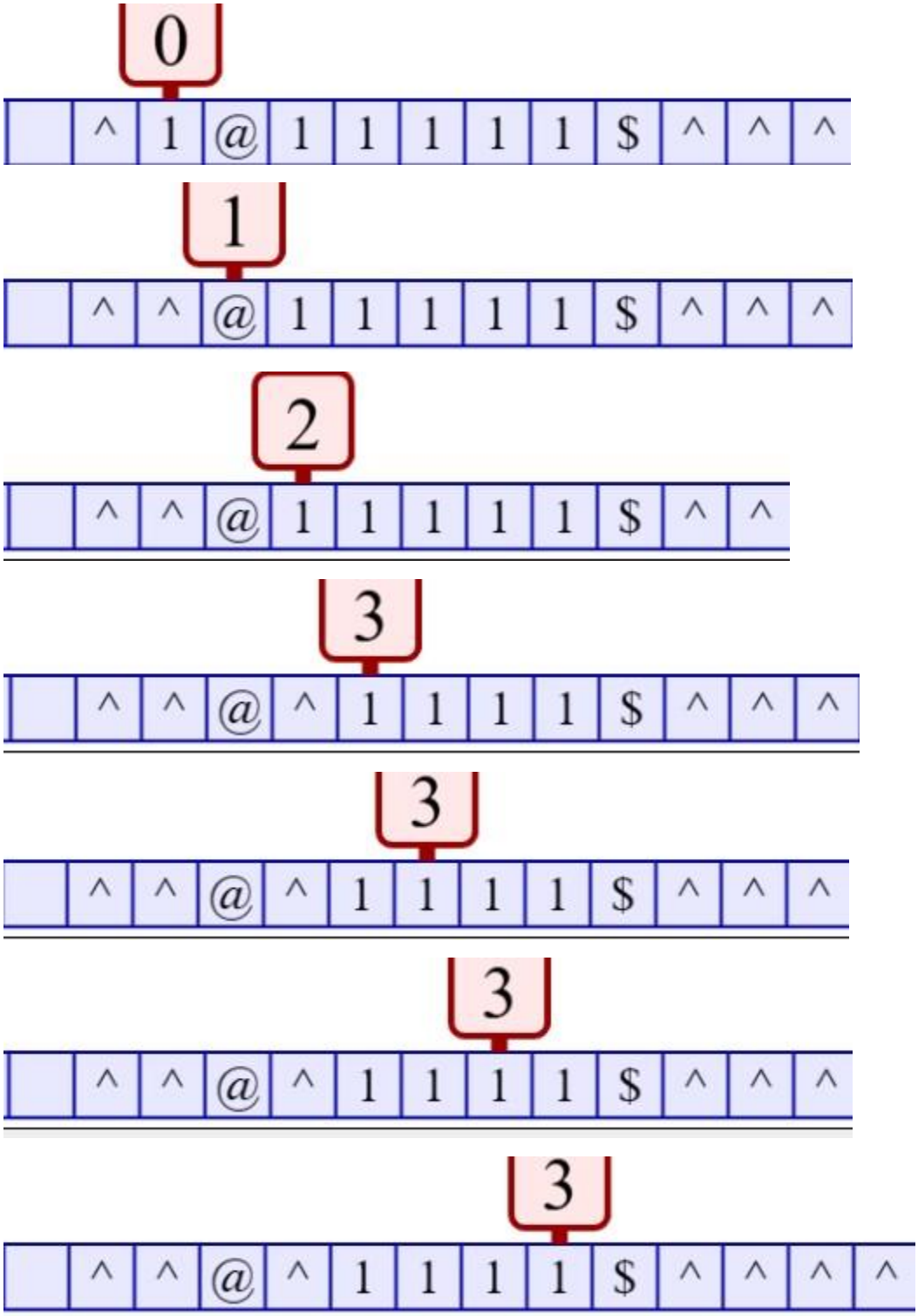

4
^ ^ 1 1 @ ^ 1 $ ^ ^ ^ ^ ^
5
^ ^ 1 1 @ ^ 1 $ 1 ^ ^ ^ ^
6
^ ^ 1 1 @ ^ 1 $ 1 ^ ^ ^ ^ ^
6
^ ^ 1 1 @ ^ 1 $ 1 ^ ^ ^ ^
2
^ ^ 1 1 @ 1 1 $ 1 ^ ^ ^ ^ ^ ^
3
^ ^ 1 1 @ 1 ^ $ 1 ^ ^ ^ ^ ^
4
^ ^ 1 1 @ 1 ^ $ 1 ^ ^ ^ ^ ^ ^
4
^ ^ 1 1 @ 1 ^ $ 1 ^ ^ ^ ^ ^ ^ ^

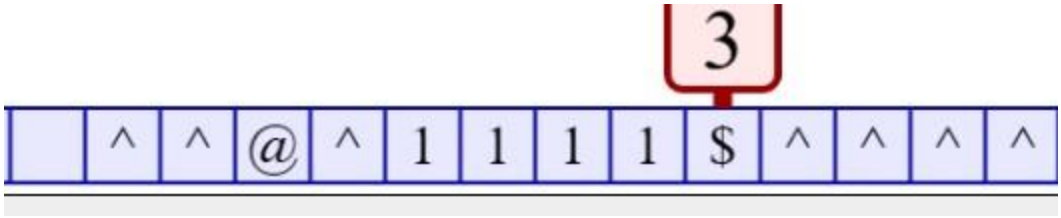
(Last 15 steps)

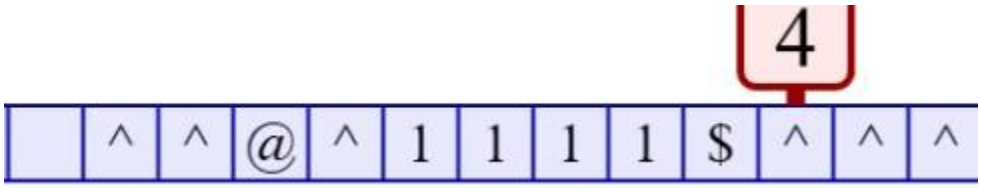


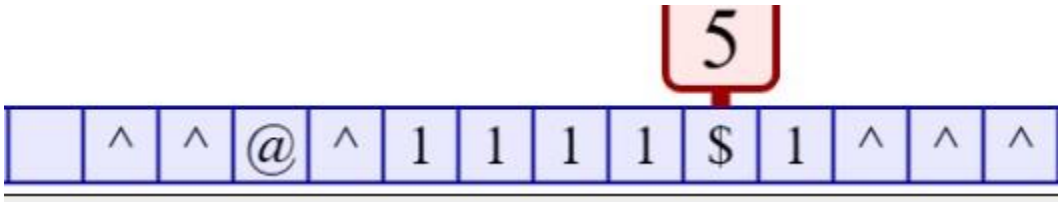

7
^ 1 1 ^ @ 1 1 $ 1 1 1 1 1 1 ^ ^
8
^ 1 1 ^ @ 1 1 $ 1 1 1 1 1 1 ^ ^
0
^ 1 1 1 @ 1 1 $ 1 1 1 1 1 1 ^ ^
9
^ 1 1 1 @ 1 1 $ 1 1 1 1 1 1 ^ ^
9
^ 1 1 1 @ 1 1 $ 1 1 1 1 1 1 ^ ^
9
^ 1 1 1 @ 1 1 $ 1 1 1 1 1 1 ^ ^
9
^ 1 1 1 @ 1 1 $ 1 1 1 1 1 1 ^ ^
10
^ 1 1 1 @ 1 1 $ 1 1 1 1 1 1 ^ ^

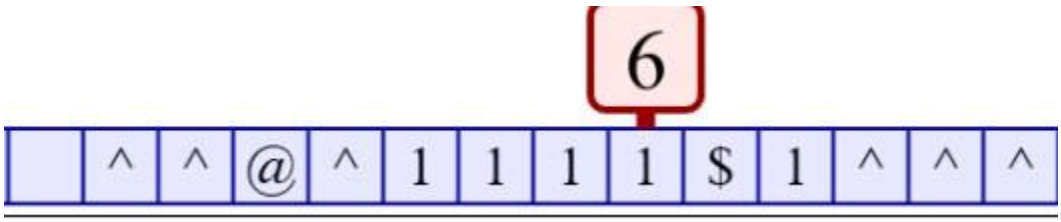
d) 1 and 5 representation
(First 15 steps)

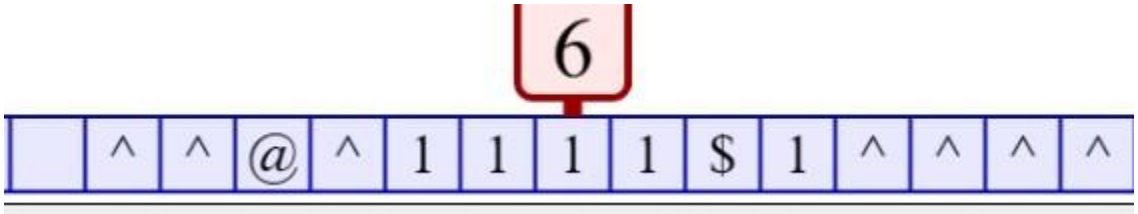


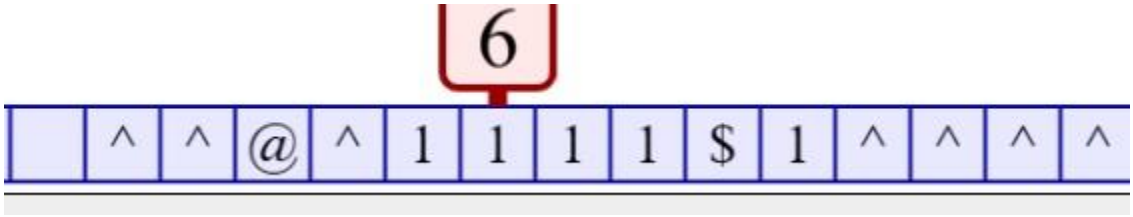

3
^ ^ @ ^ 1 1 1 1 $ ^ ^ ^ ^

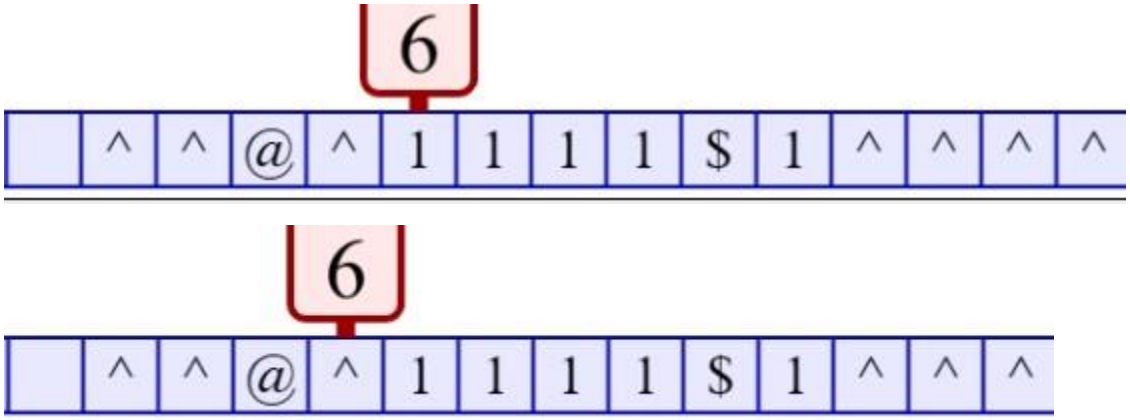

4
^ ^ @ ^ 1 1 1 1 $ ^ ^ ^


5
^ ^ @ ^ 1 1 1 1 $ 1 ^ ^ ^

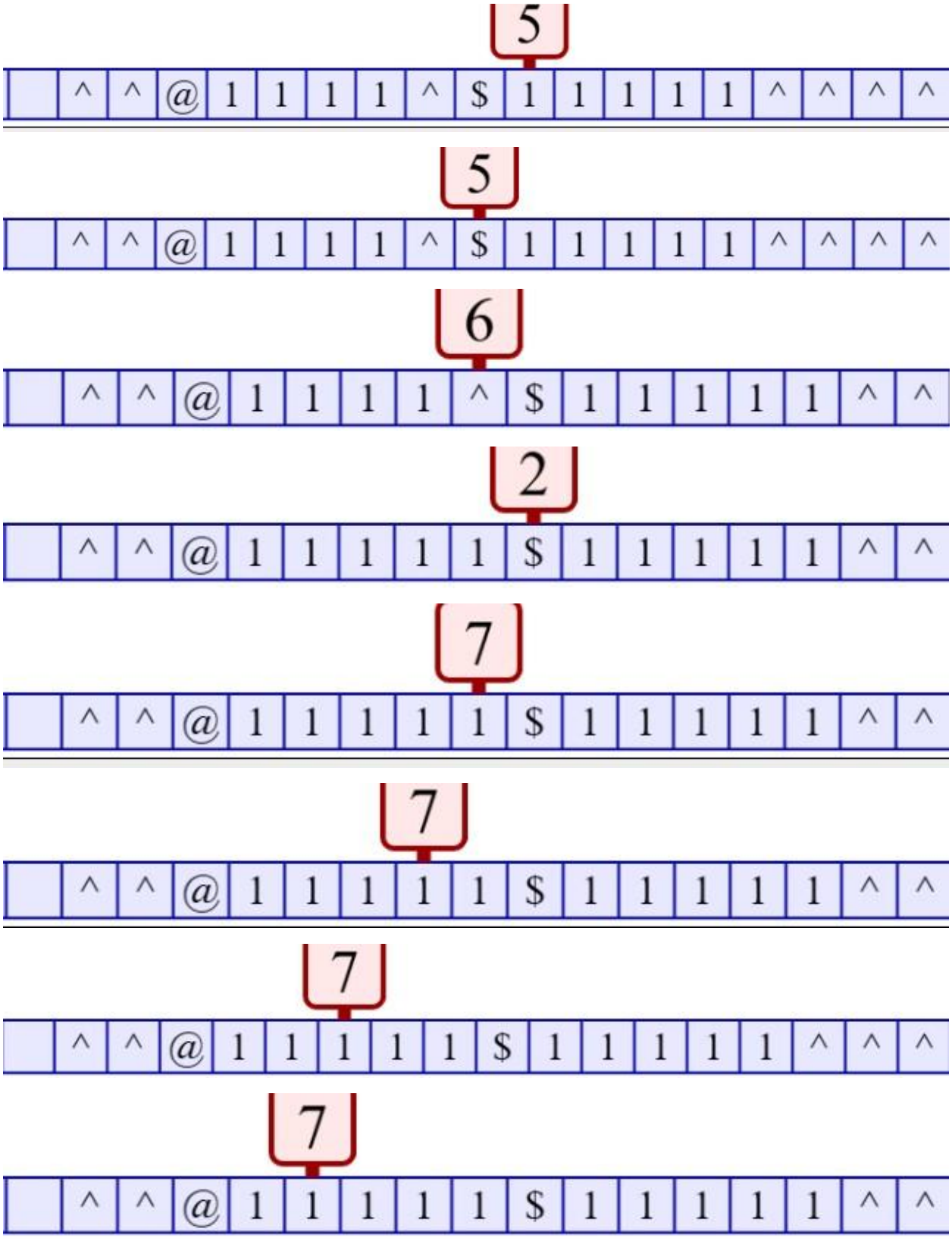

6
^ ^ @ ^ 1 1 1 1 $ 1 ^ ^ ^

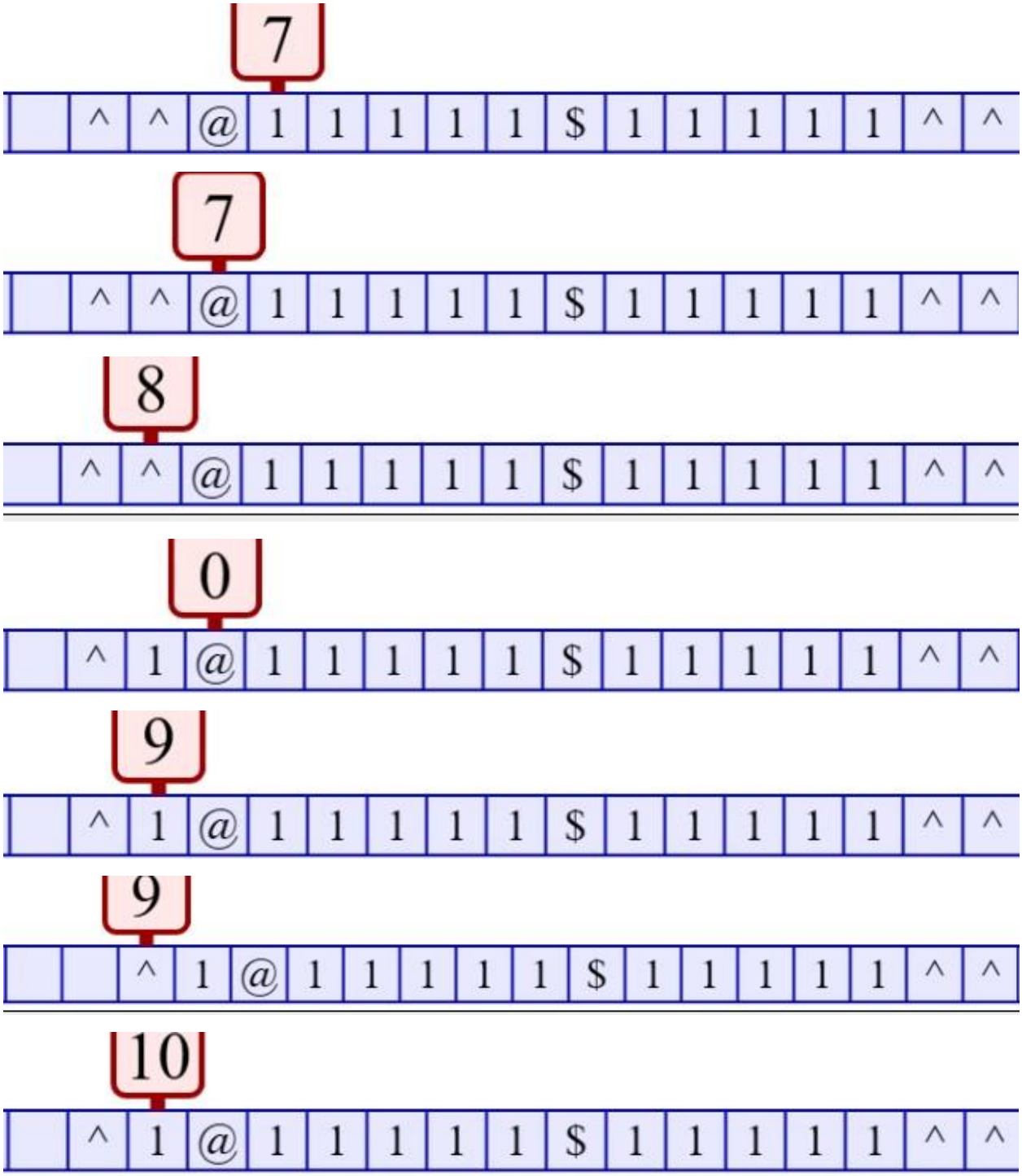

6
^ ^ @ ^ 1 1 1 1 $ 1 ^ ^ ^ ^


6
^ ^ @ ^ 1 1 1 1 $ 1 ^ ^ ^ ^


6
^ ^ @ ^ 1 1 1 1 $ 1 ^ ^ ^ ^
6
^ ^ @ ^ 1 1 1 1 $ 1 ^ ^ ^

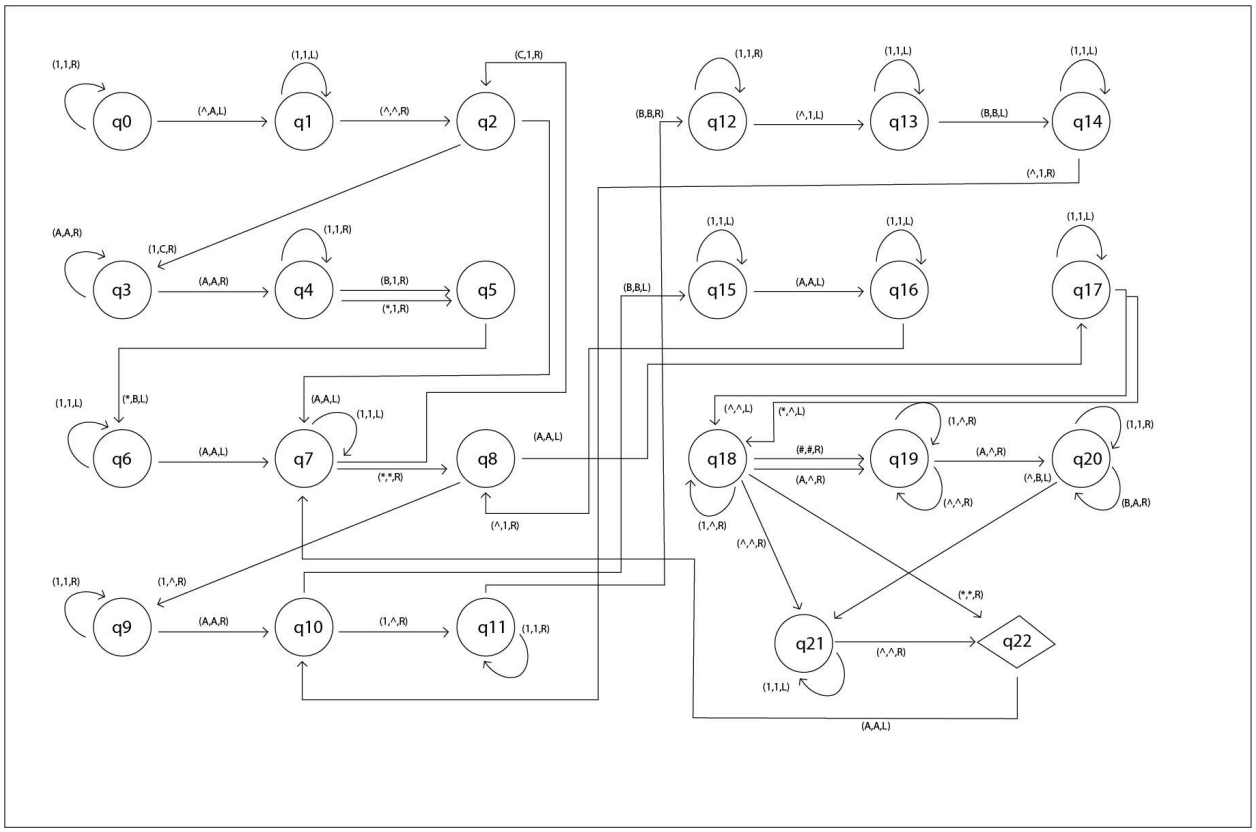
(Last 15 steps)




7
^ ^ @ 1 1 1 1 1 $ 1 1 1 1 1 ^ ^
7
^ ^ @ 1 1 1 1 1 $ 1 1 1 1 1 ^ ^
8
^ ^ @ 1 1 1 1 1 $ 1 1 1 1 1 ^ ^
0
^ 1 @ 1 1 1 1 1 $ 1 1 1 1 1 ^ ^
9
^ 1 @ 1 1 1 1 1 $ 1 1 1 1 1 ^ ^
9
^ 1 @ 1 1 1 1 1 $ 1 1 1 1 1 ^ ^
10
^ 1 @ 1 1 1 1 1 $ 1 1 1 1 1 ^ ^

Q7 Graph and JSON (2)

Graph:



Explanation:

The turing machine takes input ‘n’ number in unary and returns ‘n’, ‘n2’ and ‘n3’ as the outputs, separated by A and B.

Symbols used:

A -> separator between ‘n’ and ‘n2’.

B -> separator between ‘n2’ and ‘n3’.

C -> a marker used temporarily for easier and faster copying.

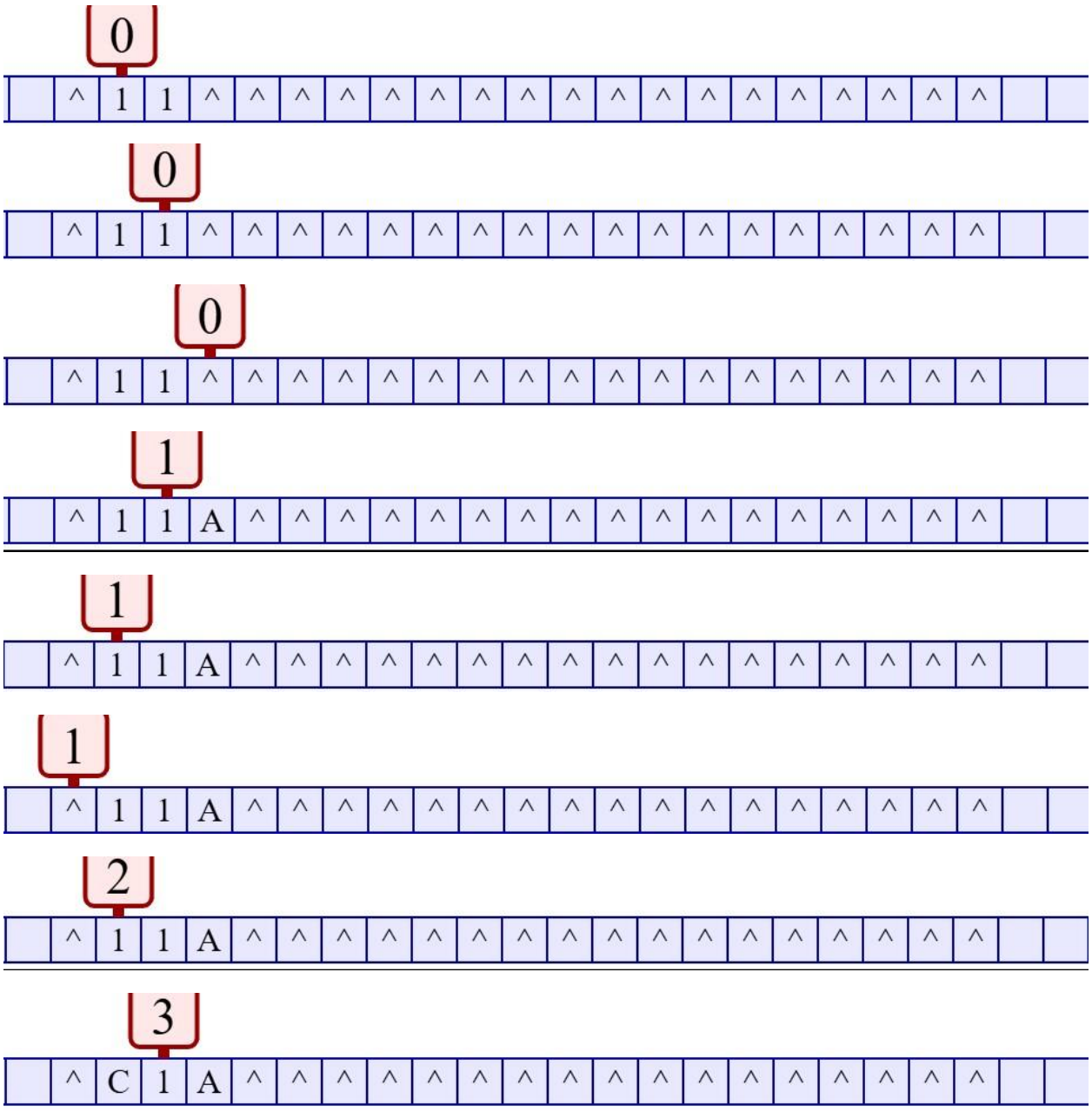
Main steps:

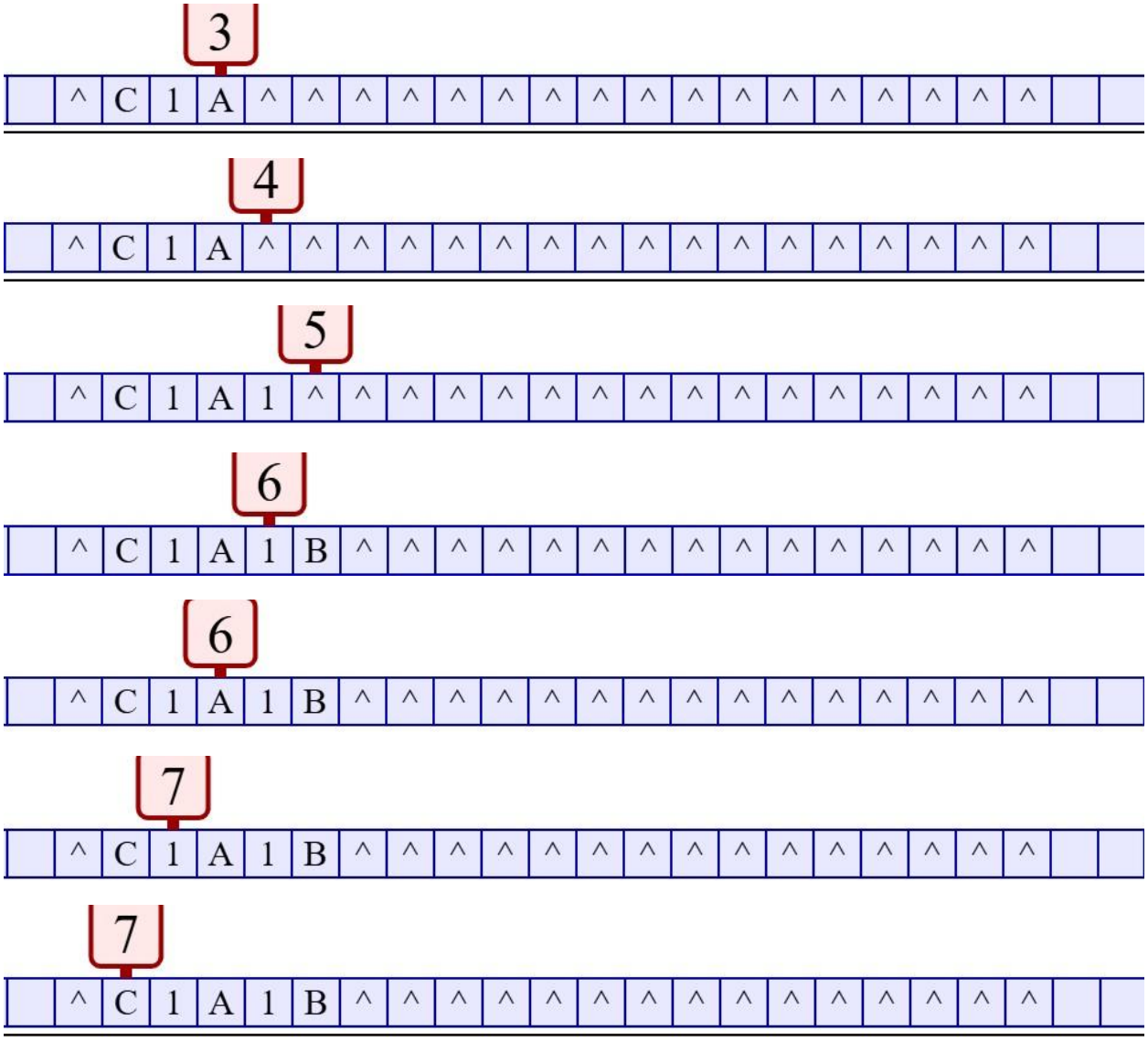
It duplicates ‘n’ and inserts ‘A’ as a separator.

Then computes ‘n2’ using the multiplication method and separates using ‘B’.
Finally it computes ‘n3’ by multiplying ‘n2’ and ‘n’.

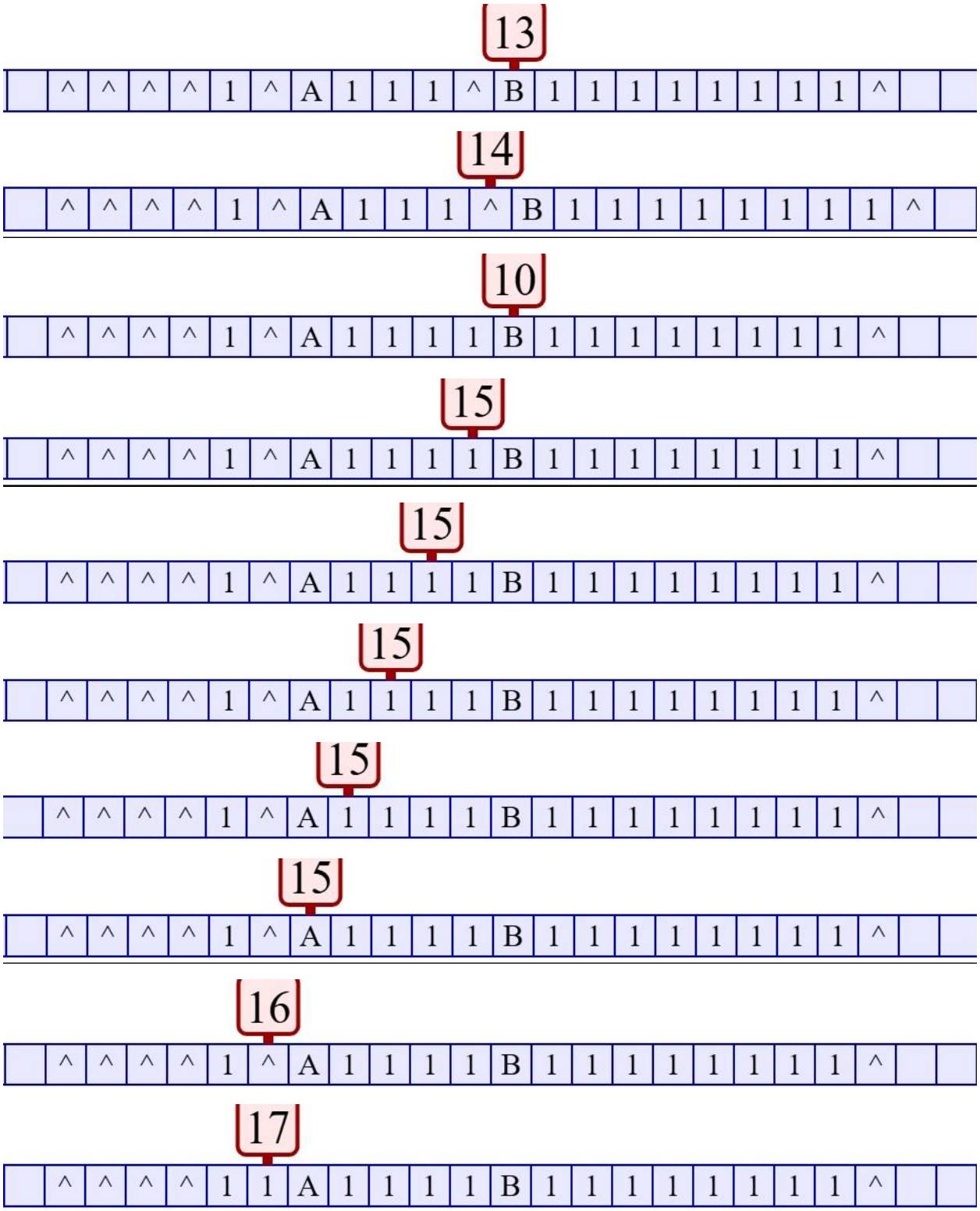
Final Output : 11A1111B11111111

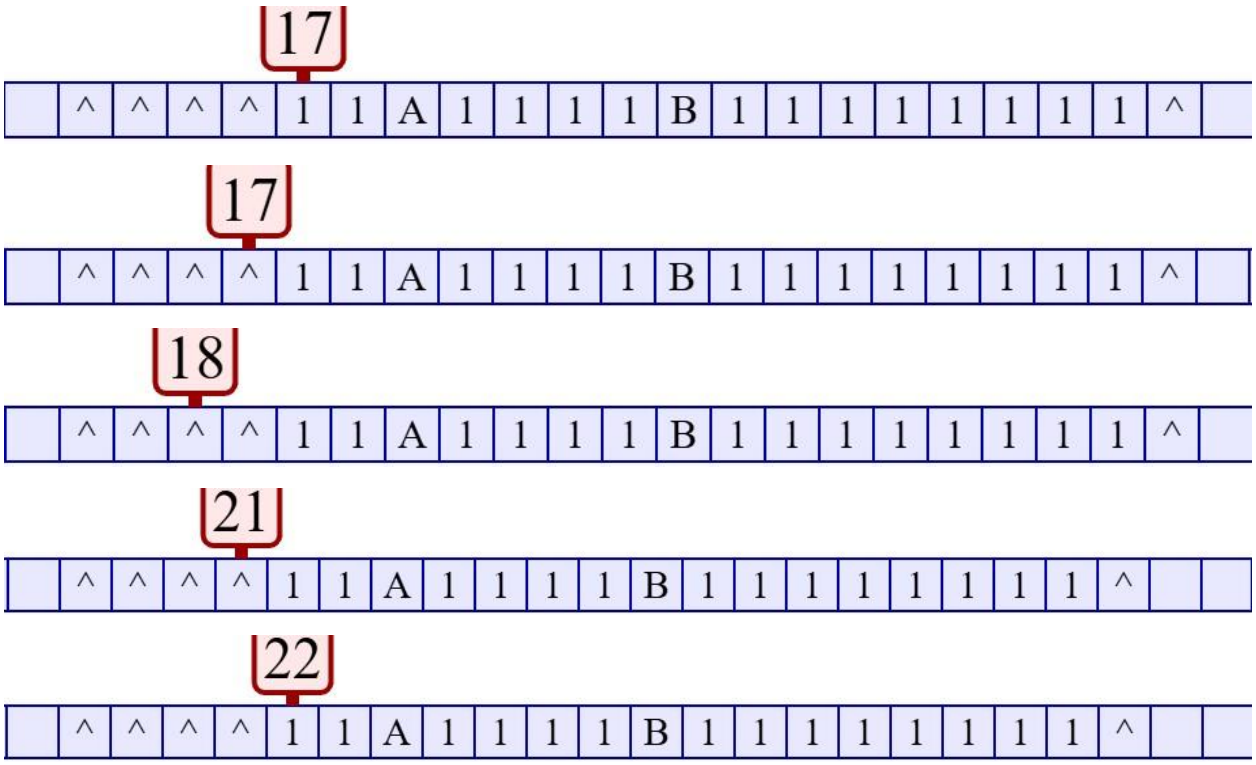
(First 15 steps)




3
^ C 1 A ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
4
^ C 1 A ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
5
^ C 1 A 1 ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
6
^ C 1 A 1 B ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
6
^ C 1 A 1 B ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
7
^ C 1 A 1 B ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
7
^ C 1 A 1 B ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^

(Last 15 steps)




17
^ ^ ^ ^ 1 1 A 1 1 1 1 B 1 1 1 1 1 1 1 1 ^
17
^ ^ ^ ^ 1 1 A 1 1 1 1 B 1 1 1 1 1 1 1 1 ^
18
^ ^ ^ ^ 1 1 A 1 1 1 1 B 1 1 1 1 1 1 1 1 ^
21
^ ^ ^ ^ 1 1 A 1 1 1 1 B 1 1 1 1 1 1 1 1 ^
22
^ ^ ^ ^ 1 1 A 1 1 1 1 B 1 1 1 1 1 1 1 1 ^