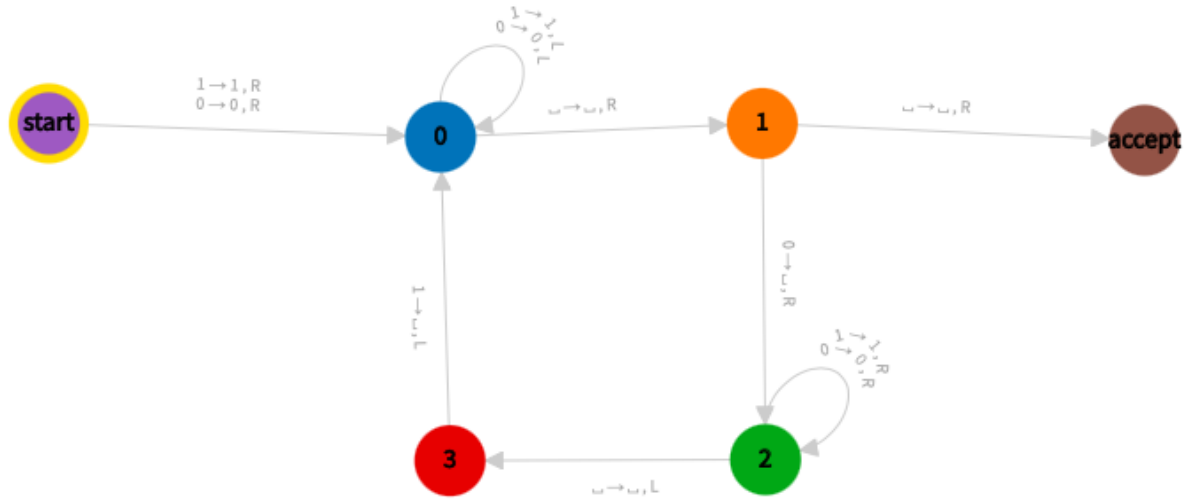
CENG 280
Fundamental Structures for Computation

Homework 2

Derya TINMAZ
2380947

Question 1

The Turing Machine



The machine basically replaces one 0 at the beginning and one 1 at the end with empty symbol in every cycle. If there is empty string at the end it accepts the string.

States

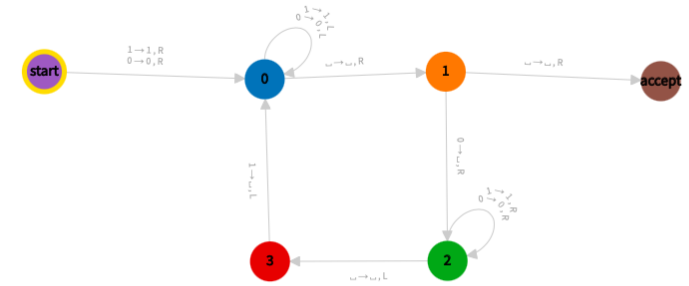
Start: The machine does not accept the empty string, that's why if there is any 0's or 1's then it moves the 0'th state, otherwise it crashes.
0: It moves the leftmost side of the string, and moves the state 1.
1: If there is a 0 at the beginning of the string, it replaces it with empty symbol and moves the state 2. If there is nothing left in the string at the end of the computation, it moves the accept state.
2: It moves the rightmost side of the string, and moves the state 3.
3: If there is a 1 at the end of the string, it replaces it with empty symbol and moves the state 2 with one left shift.
accept: The machine accepts the string only if the state at the end of the computation is accept state.

```
input: ''
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

Input Samples

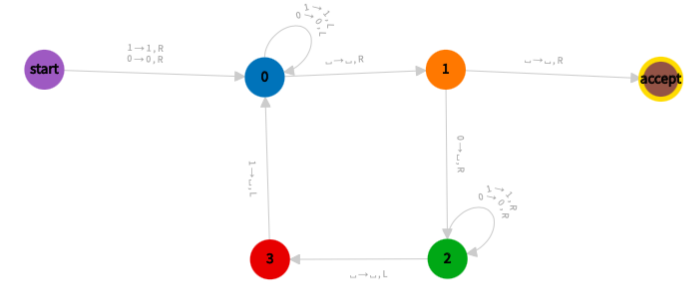
1) 000111

Initial state:




```
input: '000111'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, L}
    1      : {write: 1, L}
    ' '    : {write: ' ', R: 1}
  1:
    ' '    : {write: ' ', R: accept}
    0      : {write: ' ', R: 2}
  2:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 3}
  3:
    1      : {write: ' ', L: 0}
  accept:
```

End state:

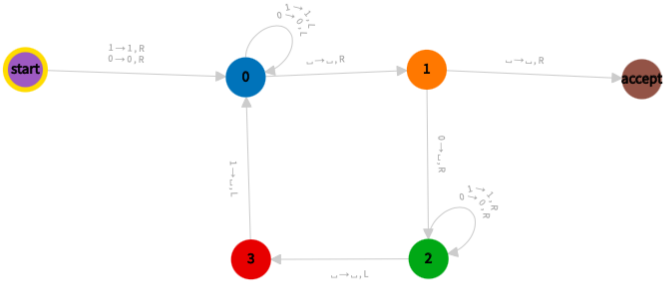



```
input: '000111'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, L}
    1      : {write: 1, L}
    ' '    : {write: ' ', R: 1}
  1:
    ' '    : {write: ' ', R: accept}
    0      : {write: ' ', R: 2}
  2:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 3}
  3:
    1      : {write: ' ', L: 0}
  accept:
```

It accepts the string.

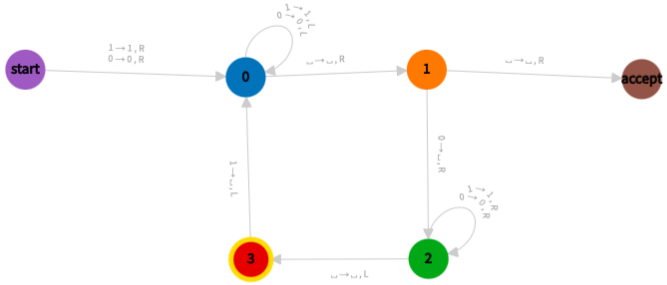
2) 0000111

Initial state:




```
input: '0000111'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

End state:

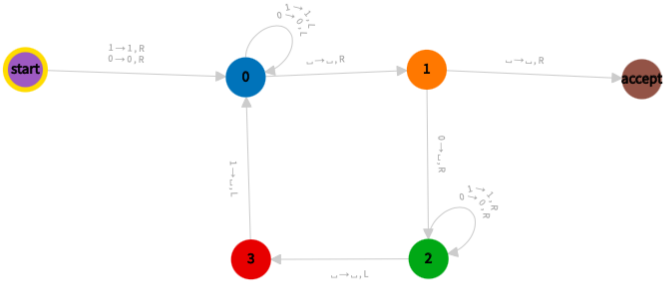



```
input: '0000111'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

It rejects the string.

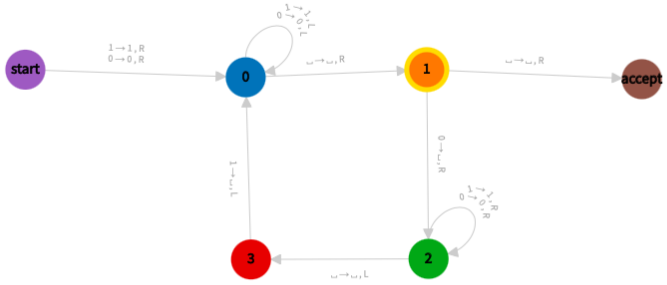
3) 000011111

Initial state:




```
input: '000011111'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

End state:

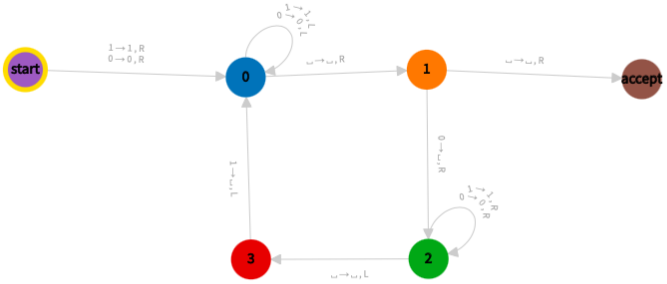



```
input: '000011111'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

It rejects the string.

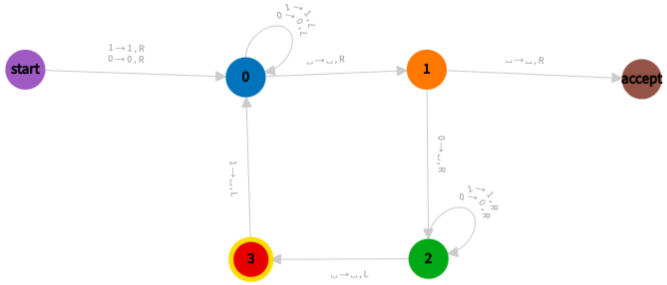
4) 0001110

Initial state:




```
input: '0001110'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

End state:

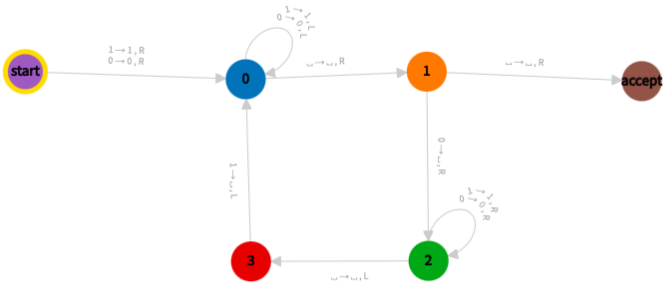



```
input: '0001110'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

It rejects the string.

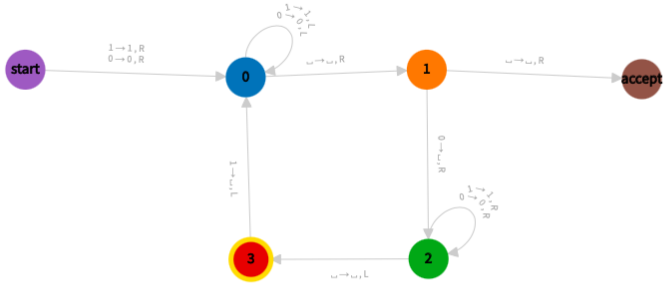
5) 001101

Initial state:




```
input: '001101'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

End state:

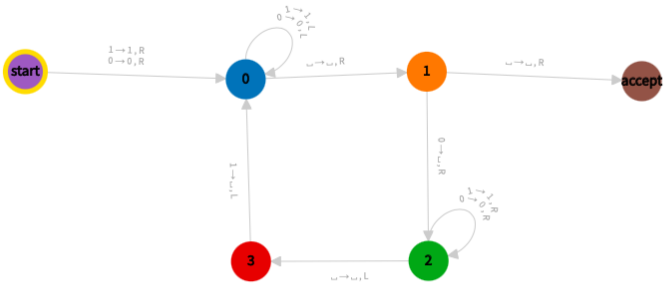



```
input: '001101'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

It rejects the string.

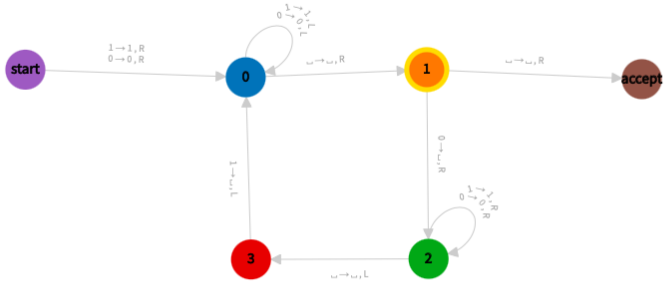
6) 100011

Initial state:




```
input: '100011'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

End state:

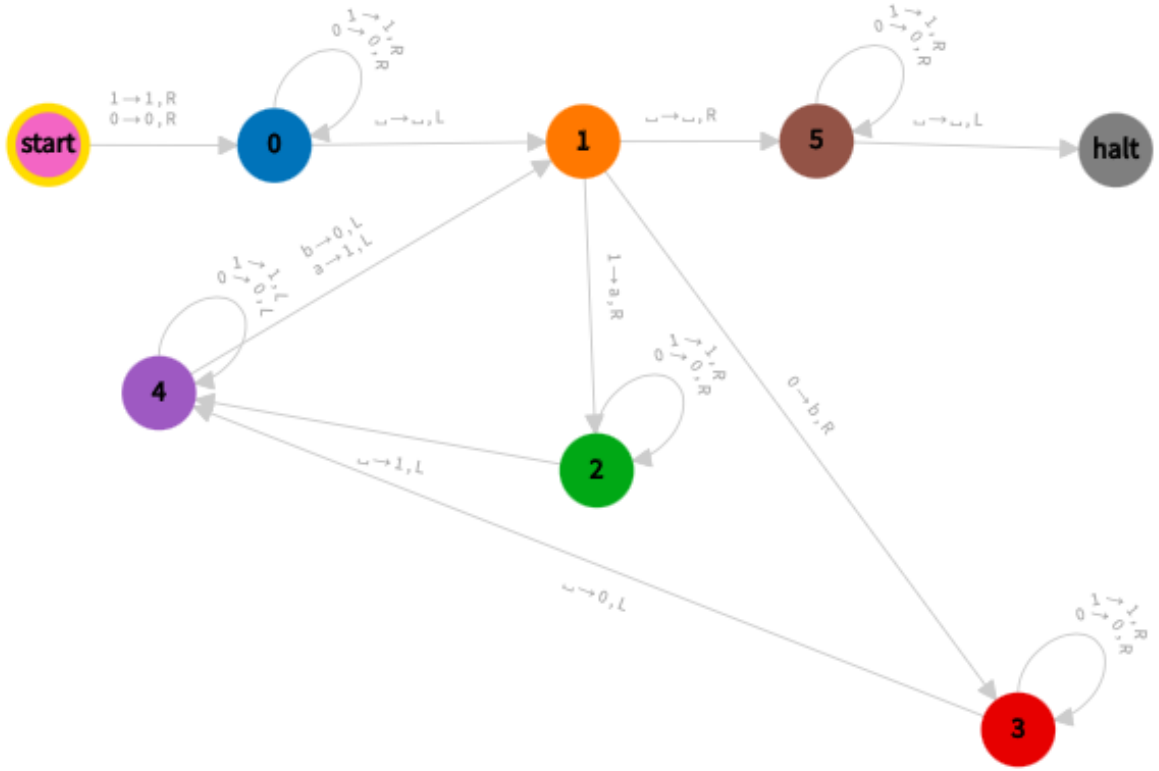



```
input: '100011'
blank: ' '
start state: start
table:
  start:
    0       : {write: 0, R: 0}
    1       : {write: 1, R: 0}
  0:
    0       : {write: 0, L}
    1       : {write: 1, L}
    ' '     : {write: ' ', R: 1}
  1:
    ' '     : {write: ' ', R: accept}
    0       : {write: ' ', R: 2}
  2:
    0       : {write: 0, R}
    1       : {write: 1, R}
    ' '     : {write: ' ', L: 3}
  3:
    1       : {write: ' ', L: 0}
  accept:
```

It rejects the string.

Question 2:

The Turing Machine



The machine copies the 0's and 1's starting from the end of the string.

States

Start: The machine does not accept the empty string, that's why if there is any 0's or 1's then it moves the 0'th state, otherwise it crashes.

0: It moves the rightmost side of the string, and moves the state 1.

1: It copies the 0 or 1 in the input to the end of the string. If there is 1 in the input it replaces it with an a than moves the state 2. If there is 0 in the input it replaces it with an b than moves the state 3. If there is nothing left to copy in the input, it moves the state 5.

2: It moves the rightmost side of the string, and copies 1 then moves the state 4.

3: It moves the rightmost side of the string, and copies 0 then moves the state 4.

4: It moves the left until it finds the last copied input letter, if it finds an a replaces it with 1, else if it finds a b replaces it with 0, then moves the state 1 with left shift to copy next letter.

5: This state just moves the tape head to the end of the string before halting the machine.

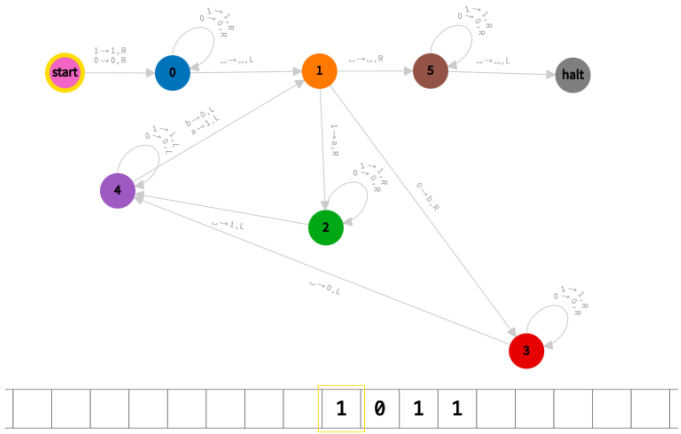
Halt: The machine halts.

```
input: ''
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
  halt:
```

Input Samples

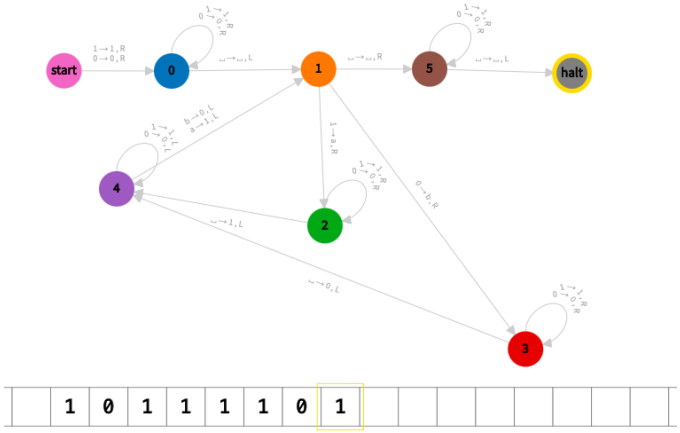
1) 1011

Initial state:




```
input: '1011'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
halt:
```

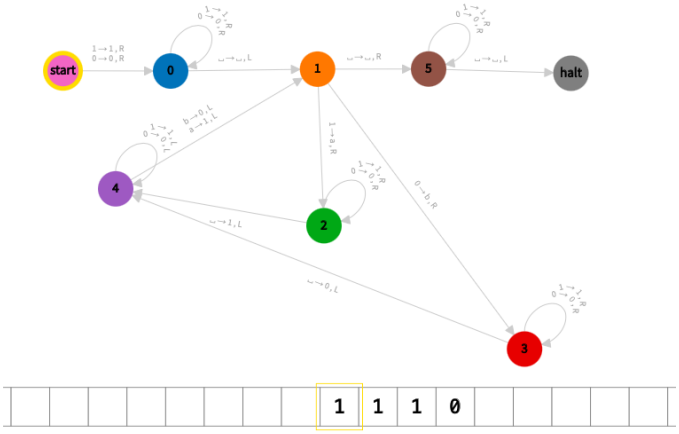
End state:




```
input: '1011'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
halt:
```

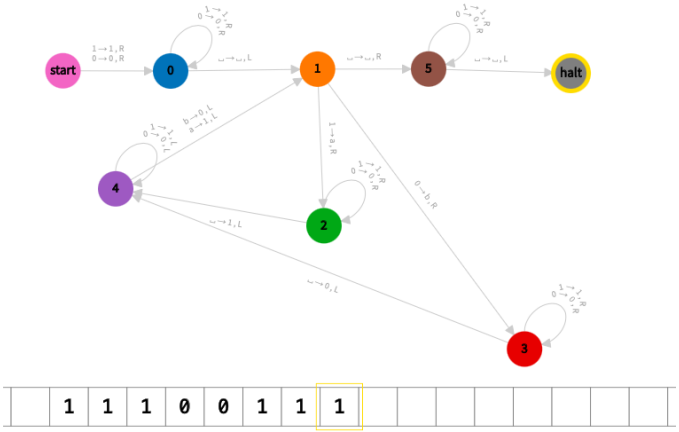
2) 1110

Initial state:




```
input: '1110'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
  halt:
```

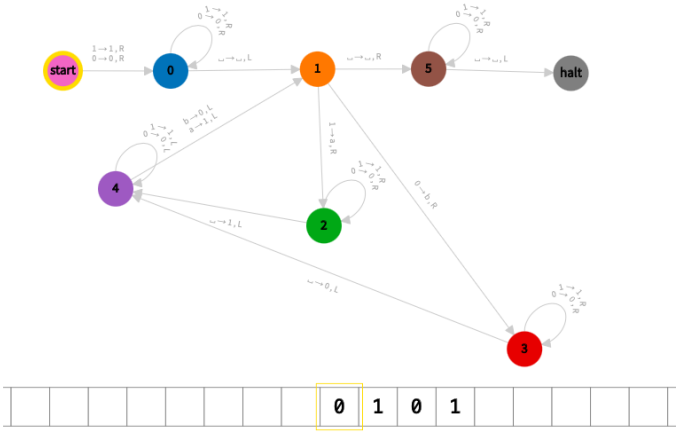
End state:




```
input: '1110'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
  halt:
```

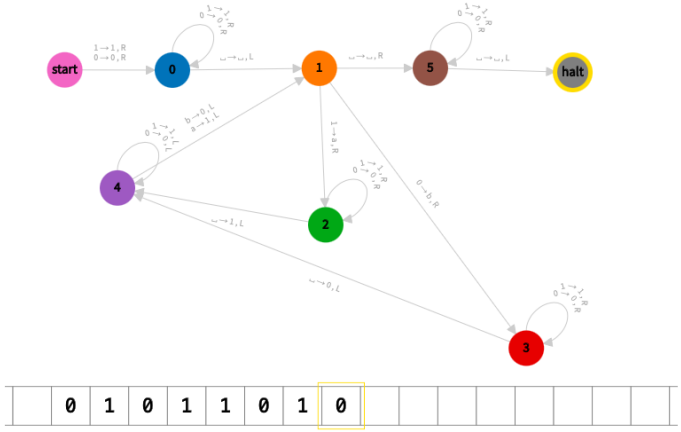
3) 0101

Initial state:




```
input: '0101'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
halt:
```

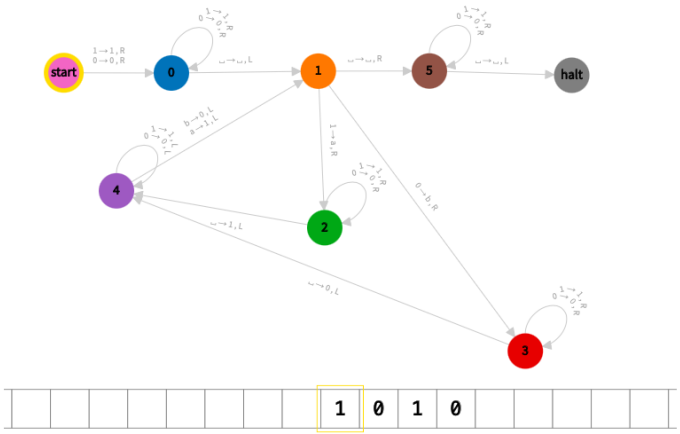
End state:




```
input: '0101'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
halt:
```

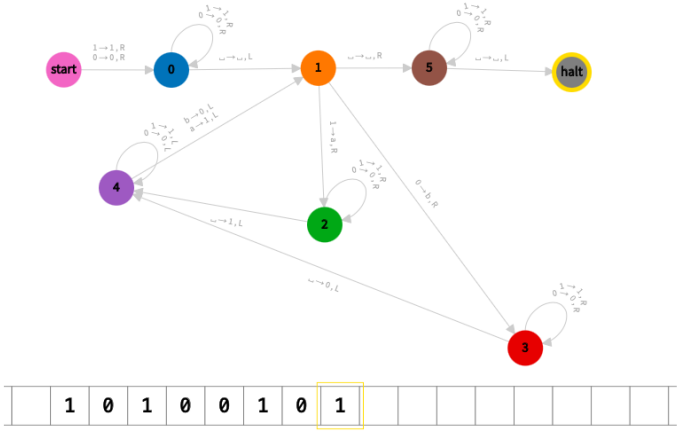
4) 1010

Initial state:




```
input: '1010'
blank: ' '
start state: start
table:
 start:
   0      : {write: 0, R: 0}
   1      : {write: 1, R: 0}
 0:
   0      : {write: 0, R}
   1      : {write: 1, R}
   ' '    : {write: ' ', L: 1}
 1:
   1      : {write: a, R: 2}
   0      : {write: b, R: 3}
   ' '    : {write: ' ', R: 5}
 2:
   1      : {write: 1, R}
   0      : {write: 0, R}
   ' '    : {write: 1, L: 4}
 3:
   1      : {write: 1, R}
   0      : {write: 0, R}
   ' '    : {write: 0, L: 4}
 4:
   0      : {write: 0, L}
   1      : {write: 1, L}
   a      : {write: 1, L: 1}
   b      : {write: 0, L: 1}
 5:
   0      : {write: 0, R}
   1      : {write: 1, R}
   ' '    : {write: ' ', L: halt}
 halt:
```

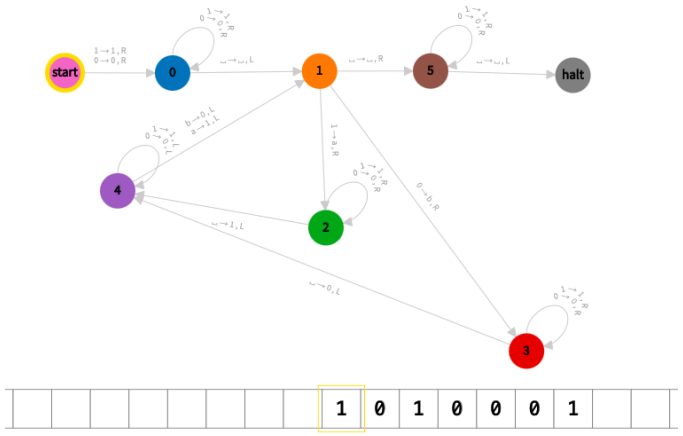
End state:




```
input: '1010'
blank: ' '
start state: start
table:
 start:
   0      : {write: 0, R: 0}
   1      : {write: 1, R: 0}
 0:
   0      : {write: 0, R}
   1      : {write: 1, R}
   ' '    : {write: ' ', L: 1}
 1:
   1      : {write: a, R: 2}
   0      : {write: b, R: 3}
   ' '    : {write: ' ', R: 5}
 2:
   1      : {write: 1, R}
   0      : {write: 0, R}
   ' '    : {write: 1, L: 4}
 3:
   1      : {write: 1, R}
   0      : {write: 0, R}
   ' '    : {write: 0, L: 4}
 4:
   0      : {write: 0, L}
   1      : {write: 1, L}
   a      : {write: 1, L: 1}
   b      : {write: 0, L: 1}
 5:
   0      : {write: 0, R}
   1      : {write: 1, R}
   ' '    : {write: ' ', L: halt}
 halt:
```

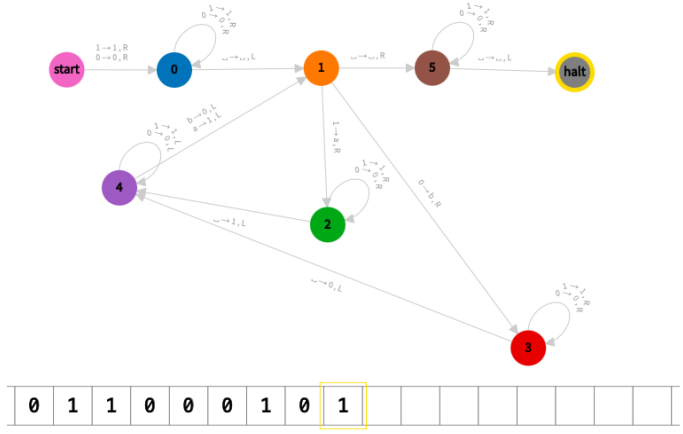
5) 1010001

Initial state:




```
input: '1010001'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
halt:
```

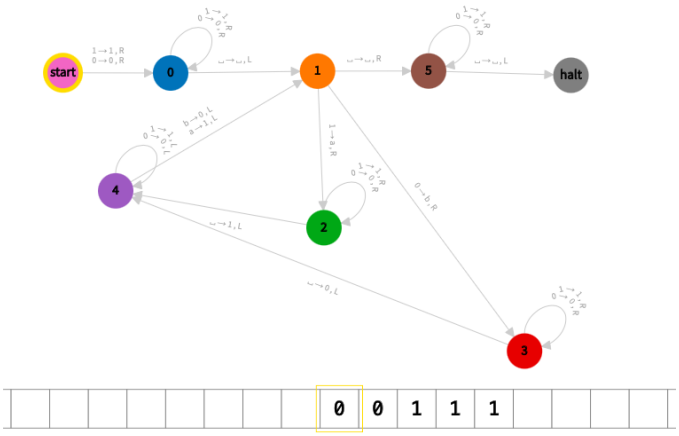
End state:




```
input: '1010001'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
halt:
```

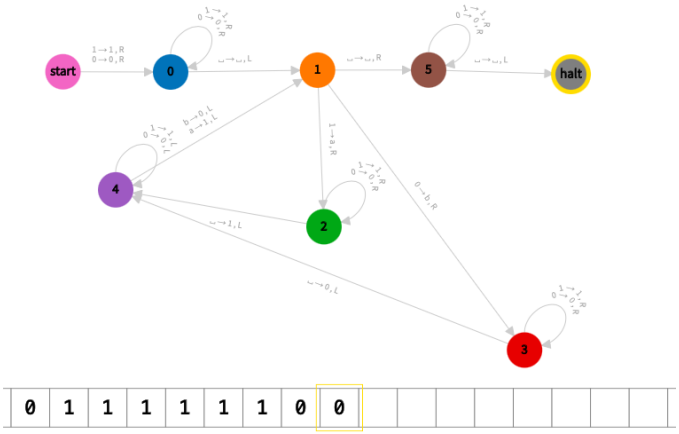
6) 00111

Initial state:




```
input: '00111'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
halt:
```

End state:




```
input: '00111'
blank: ' '
start state: start
table:
  start:
    0      : {write: 0, R: 0}
    1      : {write: 1, R: 0}
  0:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: 1}
  1:
    1      : {write: a, R: 2}
    0      : {write: b, R: 3}
    ' '    : {write: ' ', R: 5}
  2:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 1, L: 4}
  3:
    1      : {write: 1, R}
    0      : {write: 0, R}
    ' '    : {write: 0, L: 4}
  4:
    0      : {write: 0, L}
    1      : {write: 1, L}
    a      : {write: 1, L: 1}
    b      : {write: 0, L: 1}
  5:
    0      : {write: 0, R}
    1      : {write: 1, R}
    ' '    : {write: ' ', L: halt}
halt:
```

Question 3

Formal Definition of a Turing machine with a 2-dimensional tape:
$M = (K, \Sigma, \delta, s, H)$
$K =$ finite set of states
$\Sigma =$ an alphabet containing the blank symbol $\sqcup$, the tape-bottom marker $\triangle$ and the left end symbol $\triangleright$, but not containing the symbols $\rightarrow$ and $\leftarrow$
$\delta =$ function from K x Σ to K x $(\Sigma \cup \{\leftarrow, \uparrow, \downarrow, \rightarrow\})$, such that $\delta(q_1, \triangleright) = (q_2, \rightarrow)$ and $\delta(q_1, \triangle) = (q_2, \uparrow)$ for all q_1.
$s =$ initial state $(s \in K)$
$H =$ the set of halting states. $(H \subseteq K)$

A configuration is K x N x N x T, where S is the set of functions from N x N to Σ that have $t(0, y) = \triangleright$ for all $y \in N$, $t(x, 0) = \uparrow$ for all $x > 0$ and have $t(x, y) = 0$ for all but a finite number of (x, y) pairs. We thus implicitly represent a configuration by the current state, current head position, and a list of all non-blank squares on the tape. We say that $(q_1, x_1, y_1, t_1) \vdash (q_2, x_2, y_2, t_2)$ if $\delta(q_1, t_1(x_1, y_1)) = (q_2, \sigma)$ and one of the following:
$x_1 = x_2$, $y_1 + 1 = y_2$, $t_1 = t_2$, and $\sigma = \rightarrow$
$x_1 = x_2$, $y_1 - 1 = y_2$, $t_1 = t_2$, and $\sigma = \leftarrow$
$x_1 + 1 = x_2$, $y_1 = y_2$, $t_1 = t_2$, and $\sigma = \uparrow$
$x_1 - 1 = x_2$, $y_1 + 1 = y_2$, $t_1 = t_2$, and $\sigma = \downarrow$
$x_1 = x_2$, $y_1 = y_2$, $t_2(x, y) = t_1(x, y)$ for all pairs (x, y), and $\sigma \notin \{\leftarrow, \uparrow, \downarrow, \rightarrow\}$

Given a string w, let $t_w \in T$ be the function that has $t(i + 1, 1) = w(i)$ for $0 < i < |w|$,$t(0, y) = \triangleright$ for $y \in N$, $t(x, 0) = \triangle$ for all $x > 0$, and $t(x, y) = \sqcup$ otherwise. Then if we have a two dimensional tape Turing machine M with two distinguished halting states y and n such that for any string w either $(s, 1, 1, t_w) \vdash_M^* (y, i, j, t')$ or $(s, 1, 1, t_w) \vdash_M^* (n, i, j, t')$ for some $i, j \in N$ and $t' \in T$, we let the language decided by M be the set of strings for which M thus halts in the y state.

We note that from the definition of T, ant $t \in T$ can be encoded as a finite set of ordered triples (x, y, σ) for which t differs from the function which has end markers along the left and bottom edges and is otherwise everywhere blank. Since t is a function, there can be only one such σ given an x and a y, so if we encode these values along a tape, we can use dovetailing to assign a unique square for each possible x, y value. We then fill in the non-blank entries of t in this tabular format, so that $t(x, y)$ is

stored in the $[(i+j)^2+3i+j]$th square (with the first $\sqcup$ to the right of the $\triangleright$ being considered as 0. Again, we reiterate, at any time during a computation of M, only finitely many squares of this encoding tape will be non-blank.

Therefore, we simulate M with a three-tape Turing machine M'. M' uses one tape for calculation, one to hold the encoding of the tape of M as described above, and takes its input on the third. The calculations M needs to pefrom are of the following kinds: it will need to increment and decrement two "registers" i, j, (to indicate the current location of M' on its tape, and then to carry out simple multiplications and additions to convert this number into the binary which tells M' where the symbol in that cell is stored. M' begins by copying out the tape of M into its internal representation by scanning forward through its actual input and marking each symbol in the encoded location M would receive that symbol as input.

In each step of the computation, M' looks at the current symbol on the encoding tape and at the current state of M. If the instructions are to write a symbol, M' writes that symbol and proceeds. If the instructions are to move, M' increments or decrements i or j as appropriate, then calculates the value of the dovetail function. M' then moves the encoding-tape head all the way to the left, then out to the appropriate square.

Now for the time analysis. At any given time, if i and j represent the largest x and y coordinates that M has been in, then each of i and j is certainly no larger than t. Thus, the process of simulating one move takes the time required to increment or decrement a binary register, compute a few binary sums and products, and move one head by a number of spaces that is a quadratic polynomial in i and j, and thus is bounded by a quadratic polynomial in t. The increments and arithmetic are fast, and can be done in time polynomial in t (actually, much faster, since the length of the encodings are only logarithmic in the numbers they represent, so that even the worst case, in which one register grows by a digit and we need to move the rest of the tape to the right by one square, remains polynomial with ease). Thus the time to carry out t steps, once we have finished the initial setup, is $O(t^3)$. The initial setup itself can be regarded as an 2n-step computation of alternating writes and moves, so that it can be carried out in time $O(n^3)$ by the same reasoning. Thus the overall simulation proceeds in time polynomial in t and n.