Problem 1

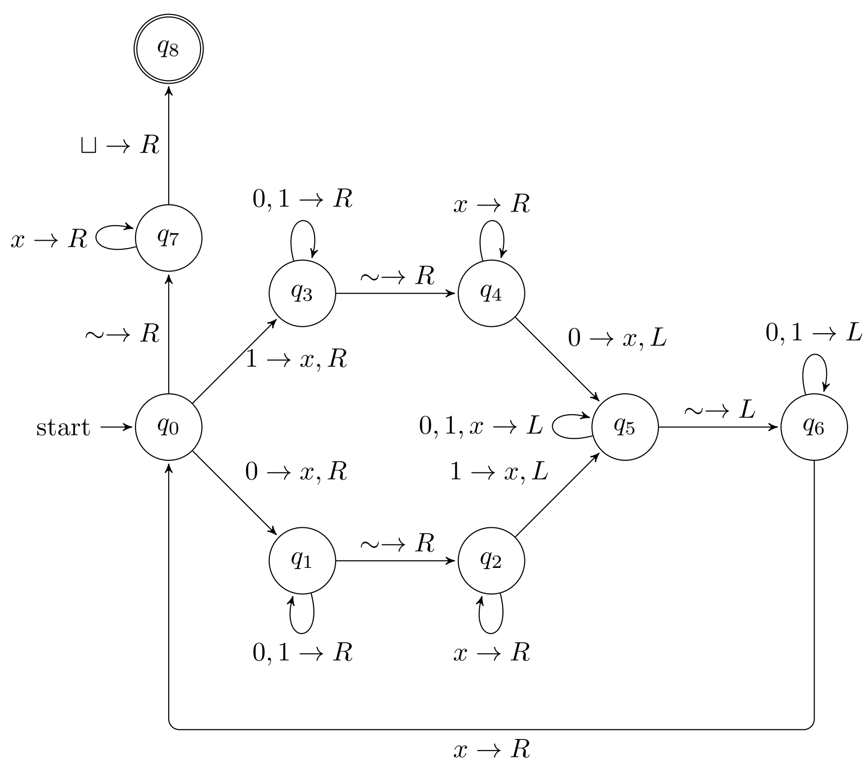
M_1 = On input string w:

1. Zig-zag across the tape to corresponding positions on either side of the "˜" symbol. If the symbols at these positions are different ($0 \to 1, 1 \to 0$), check off both of these positions. If the symbols in these positions are the same, reject immediately.

2. When all the symbols to the left of the "˜" have been marked, check for any unchecked symbols to the right of of the "˜" symbol. If there are any unchecked symbols to the right of the "˜" symbol, reject immediately. Otherwise accept.

Problem 2

$M = (A, \Sigma, \Gamma, \delta, q_0, q_9)$



Problem 3

```
states = {q0, q1, q2, q3, q4, q5, q6, q7, q8, q9}
input_alphabet = {0, 1, ~}
tape_alphabet_extra = {x, _}
start_state = q0
accept_state = q8
reject_state = q9
num_tapes = 1
delta =
  q0, 1 -> q3, x, R;
  q0, 0 -> q1, x, R;
  q0, ~ -> q7, ~, R;

  q1, 0 -> q1, 0, R;
  q1, 1 -> q1, 1, R;
  q1, ~ -> q2, ~, R;

  q2, x -> q2, x, R;
  q2, 1 -> q5, x, L;

  q3, 0 -> q3, 0, R;
  q3, 1 -> q3, 1, R;
  q3, ~ -> q4, ~, R;

  q4, x -> q4, x, R;
  q4, 0 -> q5, x, L;

  q5, 0 -> q5, 0, L;
  q5, 1 -> q5, 1, L;
  q5, x -> q5, x, L;
  q5, ~ -> q6, ~, L;

  q6, 0 -> q6, 0, L;
  q6, 1 -> q6, 1, L;
  q6, x -> q0, x, R;

  q7, x -> q7, x, R;
  q7, _ -> q8, _, R;
```

output:

1	0	~	0	1	_

state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				

output:

x	0	~	0	1	_

state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				

output:

x	0	~	0	1	_

state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				

output:

x	0	~	0	1	_

state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				

output:

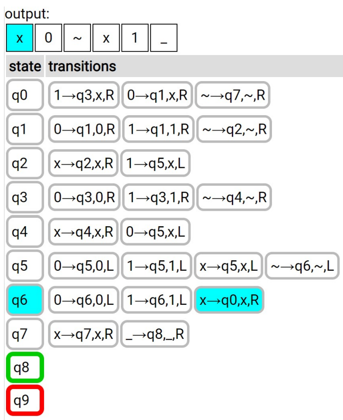
x	0	~	x	1	_

state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				

output:

x	0	~	x	1	_

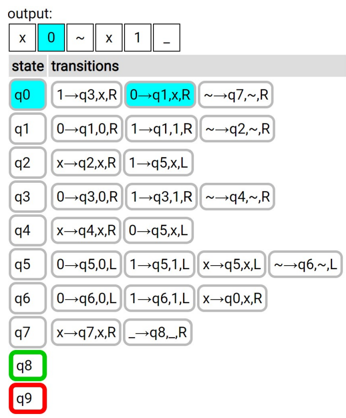
state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				



output:

x	0	~	x	1	_

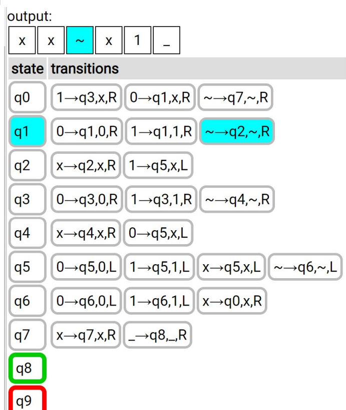
state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				



output:

x	0	~	x	1	_

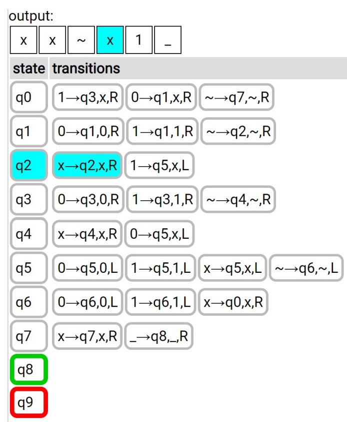
state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				



output:

x	x	~	x	1	_

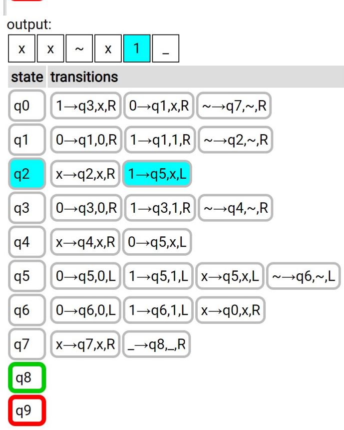
state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				



output:

x	x	~	x	1	_

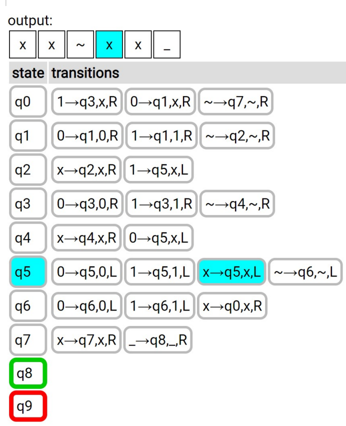
state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				



output:

x	x	~	x	1	_

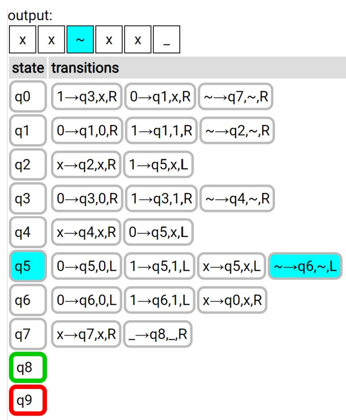
state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				

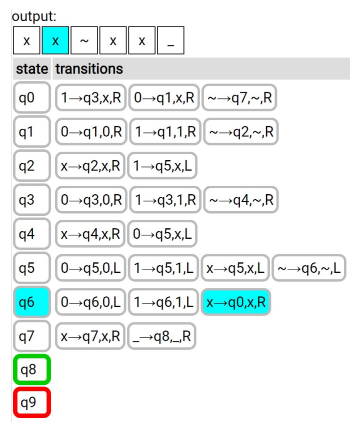


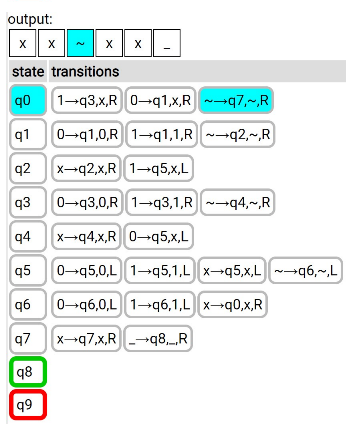
output:

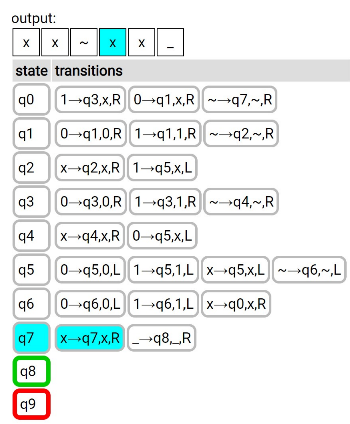
x	x	~	x	x	_

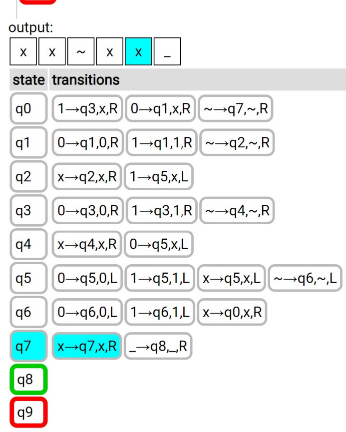
state	transitions			
q0	1→q3,x,R	0→q1,x,R	~→q7,~,R	
q1	0→q1,0,R	1→q1,1,R	~→q2,~,R	
q2	x→q2,x,R	1→q5,x,L		
q3	0→q3,0,R	1→q3,1,R	~→q4,~,R	
q4	x→q4,x,R	0→q5,x,L		
q5	0→q5,0,L	1→q5,1,L	x→q5,x,L	~→q6,~,L
q6	0→q6,0,L	1→q6,1,L	x→q0,x,R	
q7	x→q7,x,R	_→q8,_,R		
q8				
q9				

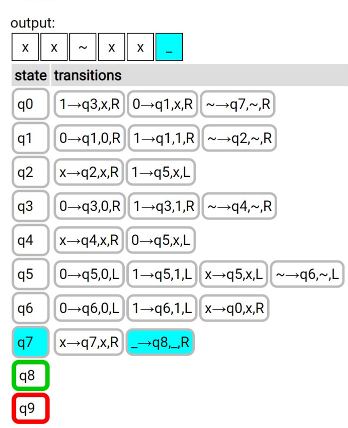

output:
x x ~ x x _
state transitions
q0 1→q3,x,R 0→q1,x,R ~→q7,~,R
q1 0→q1,0,R 1→q1,1,R ~→q2,~,R
q2 x→q2,x,R 1→q5,x,L
q3 0→q3,0,R 1→q3,1,R ~→q4,~,R
q4 x→q4,x,R 0→q5,x,L
q5 0→q5,0,L 1→q5,1,L x→q5,x,L ~→q6,~,L
q6 0→q6,0,L 1→q6,1,L x→q0,x,R
q7 x→q7,x,R _→q8,_,R
q8
q9

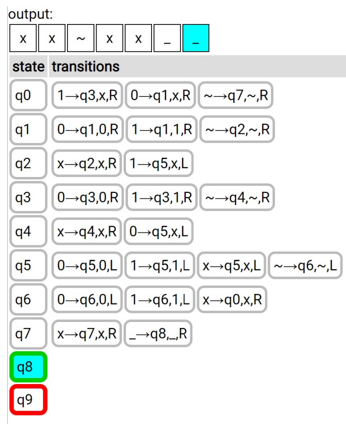

output:
x x ~ x x _
state transitions
q0 1→q3,x,R 0→q1,x,R ~→q7,~,R
q1 0→q1,0,R 1→q1,1,R ~→q2,~,R
q2 x→q2,x,R 1→q5,x,L
q3 0→q3,0,R 1→q3,1,R ~→q4,~,R
q4 x→q4,x,R 0→q5,x,L
q5 0→q5,0,L 1→q5,1,L x→q5,x,L ~→q6,~,L
q6 0→q6,0,L 1→q6,1,L x→q0,x,R
q7 x→q7,x,R _→q8,_,R
q8
q9


output:
x x ~ x x _
state transitions
q0 1→q3,x,R 0→q1,x,R ~→q7,~,R
q1 0→q1,0,R 1→q1,1,R ~→q2,~,R
q2 x→q2,x,R 1→q5,x,L
q3 0→q3,0,R 1→q3,1,R ~→q4,~,R
q4 x→q4,x,R 0→q5,x,L
q5 0→q5,0,L 1→q5,1,L x→q5,x,L ~→q6,~,L
q6 0→q6,0,L 1→q6,1,L x→q0,x,R
q7 x→q7,x,R _→q8,_,R
q8
q9


output:
x x ~ x x _
state transitions
q0 1→q3,x,R 0→q1,x,R ~→q7,~,R
q1 0→q1,0,R 1→q1,1,R ~→q2,~,R
q2 x→q2,x,R 1→q5,x,L
q3 0→q3,0,R 1→q3,1,R ~→q4,~,R
q4 x→q4,x,R 0→q5,x,L
q5 0→q5,0,L 1→q5,1,L x→q5,x,L ~→q6,~,L
q6 0→q6,0,L 1→q6,1,L x→q0,x,R
q7 x→q7,x,R _→q8,_,R
q8
q9


output:
x x ~ x x _
state transitions
q0 1→q3,x,R 0→q1,x,R ~→q7,~,R
q1 0→q1,0,R 1→q1,1,R ~→q2,~,R
q2 x→q2,x,R 1→q5,x,L
q3 0→q3,0,R 1→q3,1,R ~→q4,~,R
q4 x→q4,x,R 0→q5,x,L
q5 0→q5,0,L 1→q5,1,L x→q5,x,L ~→q6,~,L
q6 0→q6,0,L 1→q6,1,L x→q0,x,R
q7 x→q7,x,R _→q8,_,R
q8
q9


output:
x x ~ x x _
state transitions
q0 1→q3,x,R 0→q1,x,R ~→q7,~,R
q1 0→q1,0,R 1→q1,1,R ~→q2,~,R
q2 x→q2,x,R 1→q5,x,L
q3 0→q3,0,R 1→q3,1,R ~→q4,~,R
q4 x→q4,x,R 0→q5,x,L
q5 0→q5,0,L 1→q5,1,L x→q5,x,L ~→q6,~,L
q6 0→q6,0,L 1→q6,1,L x→q0,x,R
q7 x→q7,x,R _→q8,_,R
q8
q9



Problem 4

Proof.
Let M_1 be a Turing machine which decides L_1
Let M_2 be a Turing machine which decides L_2
We can define M_3, the Turing machine which decides $L_1 \cap L_2$, like so:

$$M_3 = \text{On input string } w\text{:}$$

1. Run both M_1 and M_2, one right after the other
2. If M_1 accepts w and M_2 accepts w, accept w
3. If either M_1 or M_2 reject w, reject w

M_3 accepts when $w \in L_1 \cap L_2$ and rejects when $w \notin L_1 \cap L_2$. Because M_1 and M_2 are gaurenteed to halt (L_1 and L_2 are decidable), M_3 is also gaurenteed to halt. Therefore, $L_1 \cap L_2$ is decidable.

$\square$

Problem 5

Proof.
Let M_1 be a Turing machine which decides L_1
Let M_2 be a Turing machine which decides L_2
We can define M_3, the Turing machine which decides L_1L_2, like so:

$$M_3 = \text{On input string } w\text{:}$$

1. Split the string w into two parts, LHS and RHS
2. Run M_1 on LHS and M_2 on RHS.
3. If M_1 accepts LHS and M_2 accepts RHS, accept w
4. If Either M_1 or M_2 reject, return to step one and shift the split position to the right by one. If this goes beyond the end of w, reject.

M_3 accepts when $w \in L_1L_2$ and rejects when $w \notin L_1L_2$. Because M_1 and M_2 are gaurenteed to halt (L_1 and L_2 are decidable), M_3 is also gaurenteed to halt. Therefore, L_1L_2 is decidable.

□

Problem 6

Proof.
Let M_1 be a Turing machine which recognizes L_1
Let M_2 be a Turing machine which recognizes L_2
We can define M_3, the Turing machine which recognizes $L_1 \cap L_2$, like so:

$$M_3 = \text{On input string } w\text{:}$$

1. Run both M_1 and M_2, one right after the other
2. If M_1 accepts w and M_2 accepts w, accept w
3. If either M_1 or M_2 reject w, reject w
4. If M_1 loops, then M_3 is also looping (M_3 runs M_1)
5. If M_2 loops, then M_3 is also looping (M_3 runs M_2)

M_3 accepts when $w \in L_1 \cap L_2$ and rejects (or loops) when $w \notin L_1 \cap L_2$. Because M_1 and M_2 are gaurenteed to accept, reject, or loop (L_1 and L_2 are recognizable), M_3 is also gaurenteed to accept, reject, or loop. Therefore, $L_1 \cap L_2$ is recognizable.

□

Problem 7

Proof.
Let M_1 be a Turing machine which recognizes L_1
Let M_2 be a Turing machine which recognizes L_2
We can define M_3, the Turing machine which recognizes L_1L_2, like so:

$$M_3 = \text{On input string } w\text{:}$$

1. Split the string w into two parts, LHS and RHS
2. Run M_1 on LHS and M_2 on RHS.
3. If M_1 accepts LHS and M_2 accepts RHS, accept w
4. If Either M_1 or M_2 reject, return to step one and shift the split position to the right by one. If this goes beyond the end of w, reject.

M_3 accepts when $w \in L_1L_2$ and rejects or loops when $w \notin L_1L_2$. Because M_1 and M_2 are gaurenteed to accept or loop (L_1 and L_2 are recognizable), M_3 is also gaurenteed to halt or loop. Therefore, L_1L_2 is recognizable.

□

Problem 8

Proof.
If $L(A) = \Sigma^*$, then every state of A must be an accepting state. For each state in A, check to see if it is an accepting state. If it is not, then reject. If all states are accepting, accept. Because a DFA has a finite number of states, the machine will always halt, so this language is decidable.

□

Problem 9

Proof.
The language is decidable. Construct a machine which performs the following operations.

1. Convert R to an NFA, R'
2. Convert R' to a DFA, R''
3. Construct a new DFA, R_Δ, which accepts the symmetric difference of $L(D)$ and $L(R'')$

4. If the language of R_Δ is $\emptyset$, then accept, otherwise reject. Note that it is sufficent to check weather there are any paths to an accepting state to determine this.

$\square$