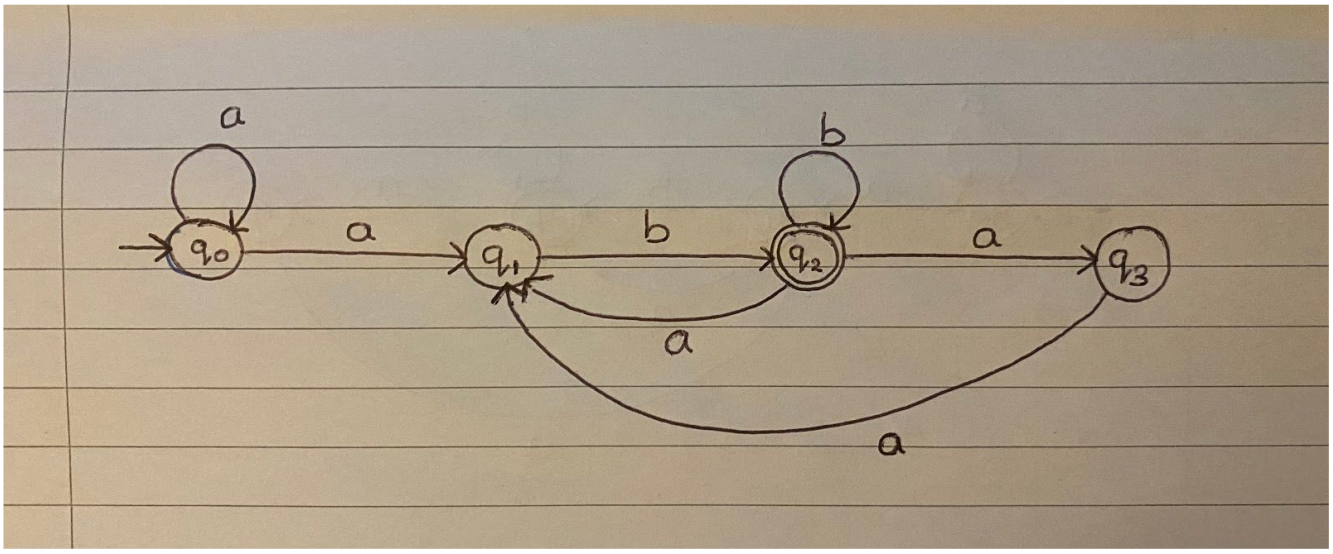
ECS421U Coursework 1 – FSA Algorithms

Name: Benyamin Babayani
ID: 190132565

Question 1

a)



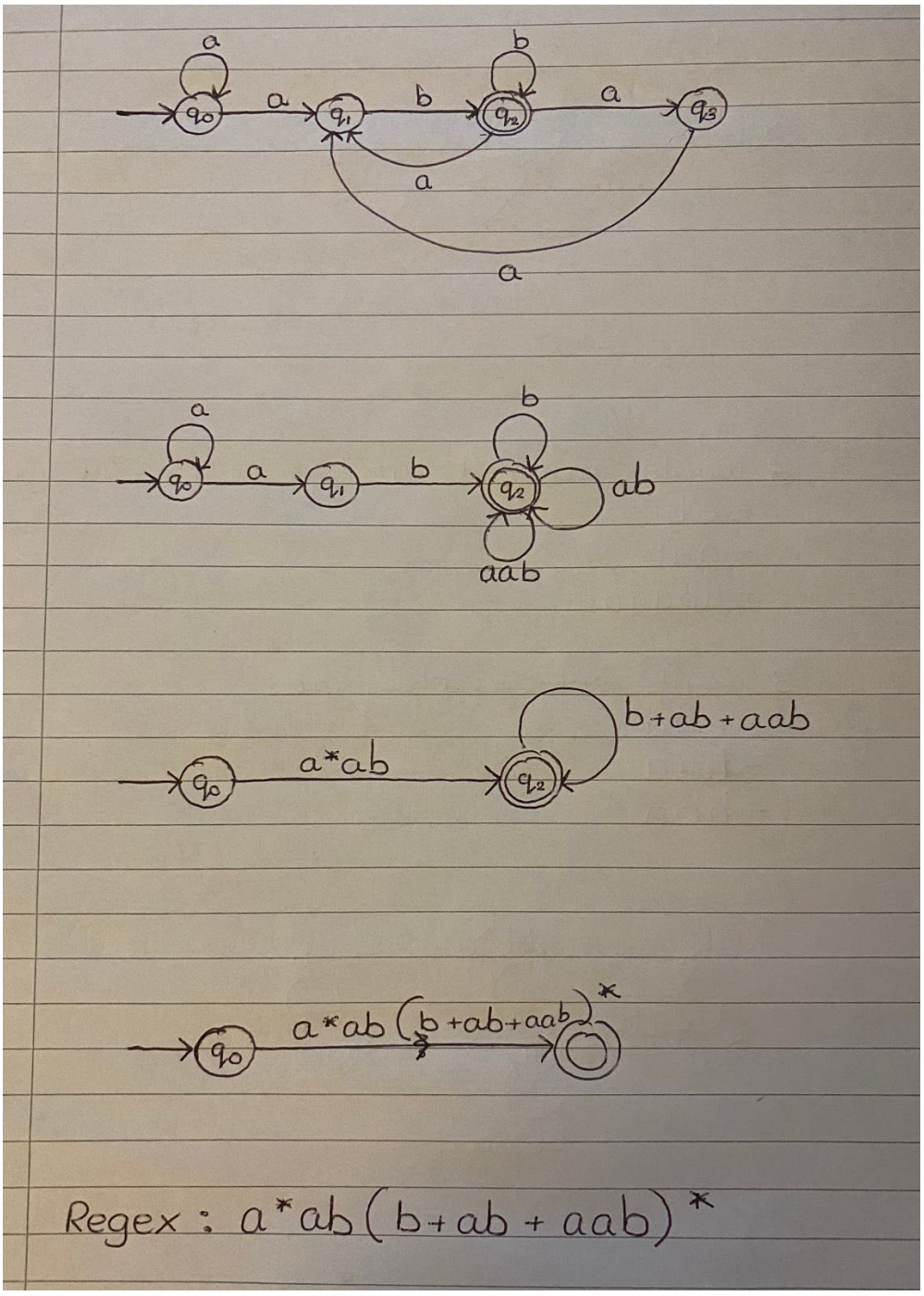
b) 3 words accepted are:

- ab
- aab
- abaab

3 words not accepted are:

- aaa
- baa
- bba

c)

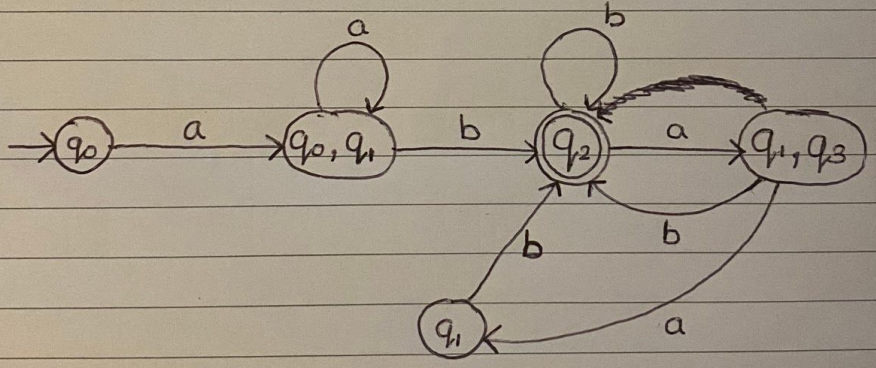


d) Automaton A is non-deterministic because there is more than one outgoing transition for the same letter. For example, the initial state, q0, has a transition (q0, a, q0) as well as a transition of (q0, a, q1).

e)

δ	a	b
q_0	$\{q_0, q_1\}$	$\emptyset$
q_1	$\emptyset$	$\{q_2\}$
q_2	$\{q_1, q_3\}$	$\{q_2\}$
q_3	$\{q_1\}$	$\emptyset$

δ'	a	b
$\{q_0\}$	$\{q_0, q_1\}$	$\emptyset$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_2\}$
$\{q_2\}$	$\{q_1, q_3\}$	$\{q_2\}$
$\{q_1, q_3\}$	$\{q_1\}$	$\{q_2\}$
$\{q_1\}$	$\emptyset$	$\{q_2\}$



f)

q_0, q_1	×			
q_2	×	×		
q_1, q_3	×	×	×	
q_1	×	×	×	×
	q_0	q_0, q_1	q_2	q_1, q_3

$(q_0, (q_0, q_1)) \xrightarrow{a} ((q_0, q_1), (q_0, q_1))$
$(q_0, (q_0, q_1)) \xrightarrow{b} (\emptyset, q_2)$

$(q_0, (q_1, q_3)) \xrightarrow{a} ((q_0, q_1), q_1)$
$(q_0, (q_1, q_3)) \xrightarrow{b} (\emptyset, q_2)$

$(q_0, q_1) \xrightarrow{a} ((q_0, q_1), \emptyset)$
$(q_0, q_1) \xrightarrow{b} (\emptyset, q_2)$

$((q_0, q_1), (q_1, q_3)) \xrightarrow{a} ((q_0, q_1), q_1)$
$((q_0, q_1), (q_1, q_3)) \xrightarrow{b} (q_2, q_2)$

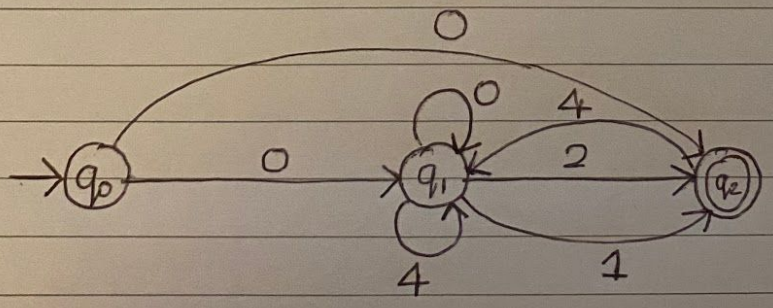
$((q_0, q_1), q_1) \xrightarrow{a} ((q_0, q_1), \emptyset)$
$((q_0, q_1), q_1) \xrightarrow{b} (q_2, q_2)$

$((q_1, q_3), q_1) \xrightarrow{a} (q_1, \emptyset)$
$((q_1, q_3), q_1) \xrightarrow{b} (q_2, q_2)$

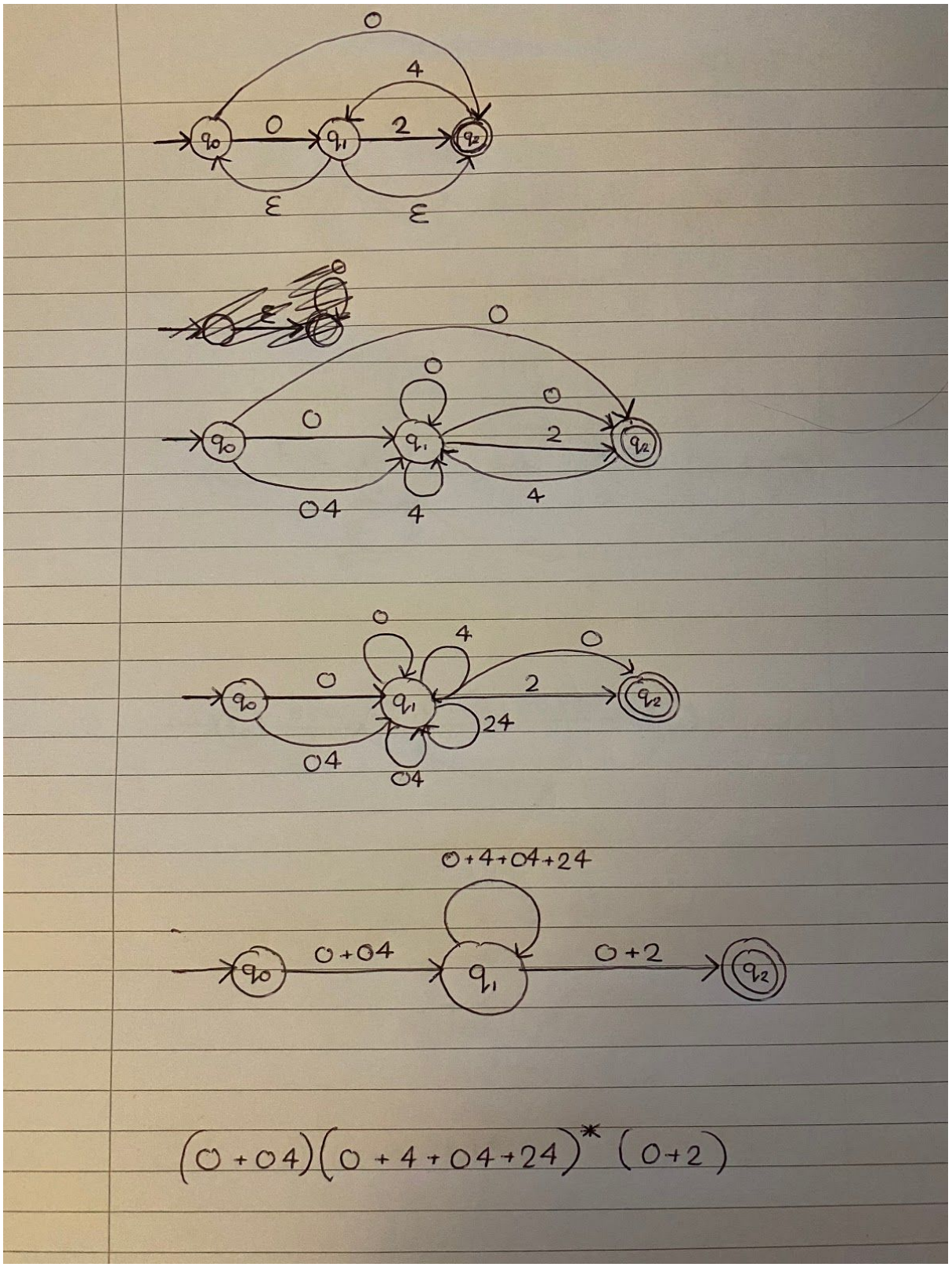
g)

δ	0	2	4	ε	ε*
q_0	$\{q_1, q_2\}$				$\{q_0\}$
q_1		$\{q_2\}$		$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$
q_2			$\{q_1\}$		$\{q_2\}$

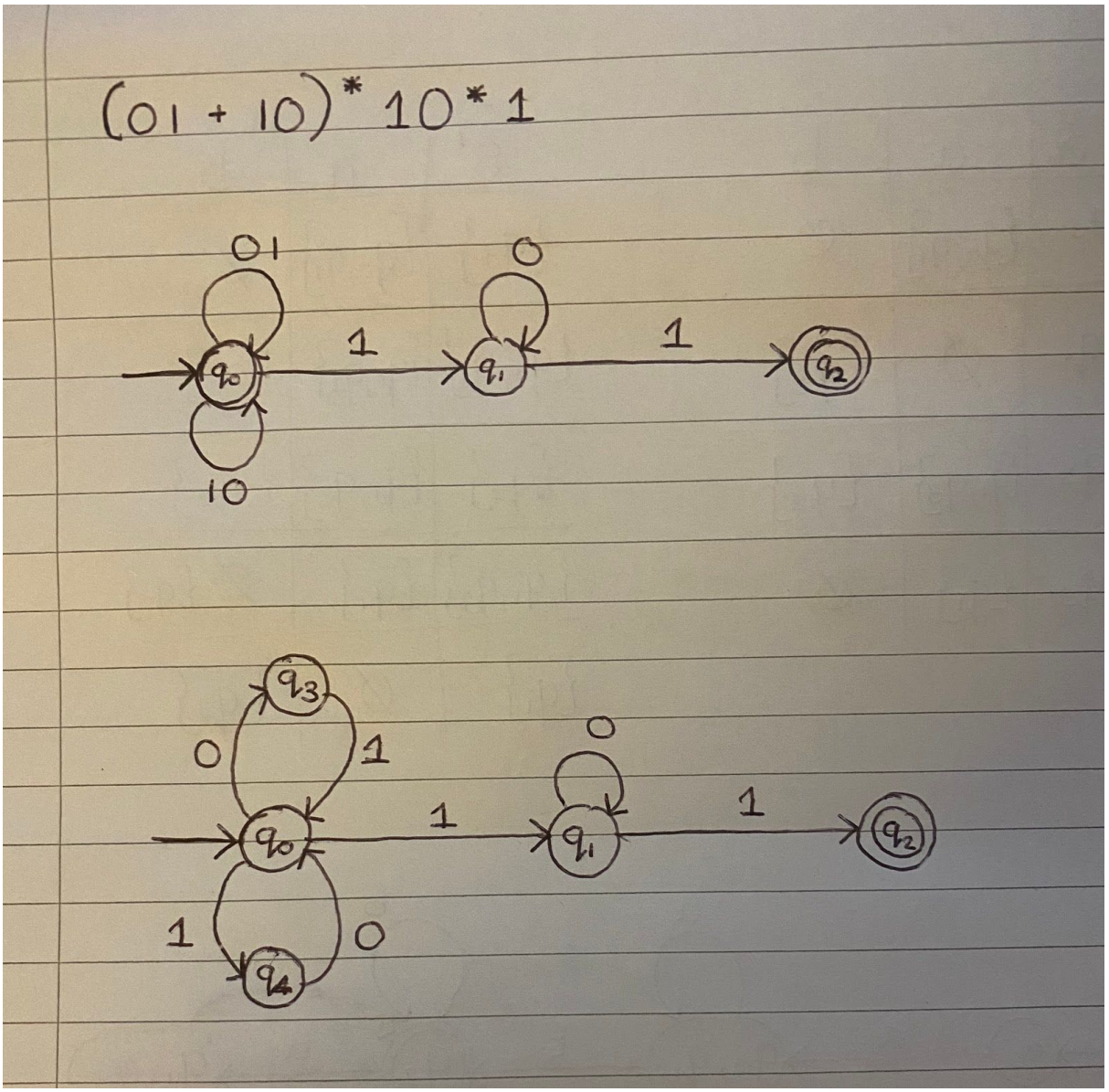
S'	0	2	4
q_0	$\{q_1, q_2\}$		
q_1	$\{q_1, q_2\}$	$\{q_2\}$	$\{q_1\}$
q_2			$\{q_1\}$



h)


0
4
0
2
q_0
q_1
q_2
ε
ε
0
ε
0
0
0
2
q_0
q_1
q_2
04
4
4
0
4
0
2
q_0
q_1
q_2
24
04
04
0+4+04+24
0+04
0+2
q_0
q_1
q_2
$(0+04)(0+4+04+24)^*(0+2)$

i)



Question 2

a)

i)

```
public static FSA generateFSA1() {
    String[] alphabet = {"0", "1"};
    Transition[] delta = new Transition[] {
        new Transition(0, "1", 0);
        new Transition(0, "1", 1);
        new Transition(1, "0", 2);
        new Transition(1, "1", 3);
        new Transition(3, "0", 5);
        new Transition(5, "1", 4);
        new Transition(4, "1", 6);
        new Transition(6, "1", 5);
    };

    int[] finals = new int[] { 6 };
    FSA A = new FSA(7,alphabet,delta,finals);
    return A;
}
```

ii)

```
public static FSA generateFSA2() {
    String[] alphabet = {"0", "1", "2"};
    Transition[] delta = new Transition[] {
        new Transition(0, "2", 0);
        new Transition(0, "1", 1);
        new Transition(1, "1", 2);
        new Transition(2, "0", 3);
        new Transition(3, "1", 0);
        new Transition(0, "0", 4);
        new Transition(4, "1", 5);
        new Transition(5, "2", 0);
    };

    int[] finals = new int[] {0, 4};
    FSA A = new FSA(6,alphabet,delta,finals);
    return A;
}
```

iii)

```
public static FSA generateFSA3() {
    String[] alphabet = {"0", "1", "2", "3"};
    Transition[] delta = new Transition[] {
        new Transition(0, "3", 1);
        new Transition(0, "1", 1);
        new Transition(1, "1", 2);
        new Transition(1, "3", 0);
        new Transition(2, "0", 1);
        new Transition(2, "3", 1);
    };

    int[] finals = new int[] {1, 2};
    FSA A = new FSA(3,alphabet,delta,finals);
    return A;
}
```

b)

```
public static void printThem() {
    checkPrintFSA(generateFSA1(), "A1");
    checkPrintFSA(generateFSA2(), "A2");
    checkPrintFSA(generateFSA3(), "A3");
}
```

```
Printout(b)
A1 = ({0, 1}, {q0, q1, q2, q3, q4, q5, q6}, {(q0,1,q0), (q0,1,q1), (q1,0,q2), (q1,1,q3), (q3,0,q5), (
q5,1,q4), (q4,1,q6), (q6,1,q5)}, q0, {q6})
A2 = ({0, 1, 2}, {q0, q1, q2, q3, q4, q5}, {(q0,2,q0), (q0,1,q1), (q1,1,q2), (q2,0,q3), (q3,1,q0), (q
0,0,q4), (q4,1,q5), (q5,2,q0)}, q0, {q0, q4})
A3 = ({0, 1, 2, 3}, {q0, q1, q2}, {(q0,3,q1), (q0,1,q1), (q1,1,q2), (q1,3,q0), (q2,0,q1), (q2,3,q1)},
 q0, {q1, q2})
```

c)

```java
public static void runThem() {
    String[] w1 = {"1", "1", "0", "1", "1"};
    String[] w2 = {"1", "1", "0", "1", "0"};
    String[] w3 = {"1", "1", "0", "0", "1"};
    FSA A1 = generateFSA1();
    FSA A2 = generateFSA2();
    FSA A3 = generateFSA3();
    FSA[] A = {A1, A2, A3};
    String[] ALabel = {"A1", "A2", "A3"};

    for (int i = 0; i<A.length; i++) {
        System.out.print("\n" + ALabel[i] + " accepts: ");
        System.out.print("11011 ");
        if((isAccepted(A[i], w1)) == true) {
            System.out.print("yes, ");
        }
        else {
            System.out.print("no, ");
        }
        System.out.print("11010 ");
        if((isAccepted(A[i], w2)) == true) {
            System.out.print("yes, ");
        }
        else {
            System.out.print("no, ");
        }
        System.out.print("11001 ");
        if((isAccepted(A[i], w3)) == true) {
            System.out.print("yes\n");
        }
        else {
            System.out.print("no\n");
        }
    }
}
```

```
Printout(c)

A1 accepts: 11011 yes, 11010 no, 11001 no

A2 accepts: 11011 no, 11010 yes, 11001 no

A3 accepts: 11011 no, 11010 yes, 11001 no
```

d)

```
public static Boolean isDeterministic(FSA A) {
    int count = 0;
    for (int i=1; i<(A.delta).length; i++) {
        if ((A.delta[i-1]).from == (A.delta[i]).from) {
            if ((A.delta[i-1]).label == (A.delta[i]).label) {
                count += 1;
                break;
            }
        }
    }

    if (count > 1) {
        return false;
    }
    else {
        return true;
    }
}
```

My code uses a “for loop” to traverse through the delta array, which is of the type Transition. I have used two “if statements”, one which compares the state component between two consecutive Transition objects and the other “if statement” that checks the alphabet component. If the state and alphabet component have the same value for both consecutive Transition objects, a variable called “count” is incremented by 1. The “count” variable is important as it helps us to differentiate between the deterministic and non-deterministic FSAs. If the “count” variable stores a value greater than 1, it means that the FSA is non-deterministic otherwise it must be deterministic.

```
Printout(d)
Test 1: ({0, 1, 2}, {q0, q1, q2}, {(q0,1,q0), (q0,0,q1), (q0,2,q2), (q1,1,q2), (q2,0,q2), (q2,1,q2)}, q0, {q2})
Result: 1
Test 2: ({0, 1, 2}, {q0, q1, q2}, {(q0,0,q0), (q0,0,q1), (q0,2,q2), (q1,1,q2), (q2,0,q2), (q2,1,q2)}, q0, {q2})
Result: 0
Test 3: ({0, 1, 2}, {q0, q1, q2}, {(q0,0,q0), (q0,0,q1), (q0,0,q2), (q1,1,q2), (q2,0,q2), (q2,1,q2)}, q0, {q2})
Result: 0
Test 4: ({0, 1, 2}, {q0, q1, q2}, {(q0,1,q0), (q0,0,q1), (q0,2,q2), (q1,1,q2), (q2,0,q2), (q2,0,q1)}, q0, {q2})
Result: 0
Test 5: ({0, 1, 2}, {q0, q1, q2}, {(q0,1,q0), (q0,0,q1), (q0,2,q2), (q1,0,q2), (q2,0,q2), (q2,0,q0), (q2,0,q1)}, q0, {q2})
Result: 0
Test 6: ({0, 1, 2, a, b}, {q0, q1, q2}, {(q0,1,q0), (q0,0,q1), (q0,2,q2), (q1,0,q2), (q2,a,q2), (q2,b,q0), (q2,c,q1)}, q0,
{q2})
Result: 1
Test 7: ({0, 1, 2, a, b}, {q0, q1, q2}, {(q0,1,q0), (q0,0,q1), (q0,2,q2), (q1,0,q2), (q2,a,q2), (q2,b,q2), (q2,c,q2)}, q0,
{q2})
Result: 1
Test 8: ({0, 1, 2, a, b}, {q0, q1, q2}, {(q0,1,q0), (q0,1,q1), (q0,2,q2), (q1,0,q2), (q2,a,q2), (q2,b,q2), (q2,a,q1)}, q0,
{q2})
Result: 0
Test 9: ({a, b, c, d}, {q0, q1, q2, q3}, {(q0,a,q0), (q0,b,q1), (q0,c,q2), (q0,d,q3), (q1,a,q0), (q1,b,q1), (q1,c,q2), (q1,
d,q3), (q2,a,q0), (q2,b,q1), (q2,c,q2), (q2,d,q3), (q3,a,q0), (q3,b,q1), (q3,c,q2), (q3,d,q3)}, q0, {q0, q1, q2, q3})
Result: 1
Test 10: ({0, 1, 2, a, b}, {q0}, {}, q0, {})
Result: 1
Total: 5
```