**Homework 1** (To be delivered during the 1st midterm exam)

BLM 2502: Theory of Computations — Spring 2020

Print family (or last) name: ———————————————————————————

Print given (or first) name: ———————————————————————————

Print given student number: ———————————————————————————

I see that this homework has 17 questions in total 16 pages.

I agree that I have to submit my homework solution before the deadline (i.e., the time of the 1st midterm exam) otherwise my homework solution will not be graded. I accept that ***I will add the signed version of this instruction page as a first page into my homework solution***; otherwise my homework solution will not be graded. I know that ***I have to give my solutions in the empty-white spaces just below the questions***; otherwise my homework solution will not be graded. I will take care of the readability of my solutions, from which I may lose 10 points. For any proofs, I am sure to provide a step-by-step argument, with justifications for every step. I understand that, during solving this homework, it is prohibited to exchange information about solutions with any other person in any way, including by talking or exchanging solutions / papers.

I know that the course book is "Introduction to the theory of computation, 2nd Ed., Massachusetts Institute of Technology, by Micheal Sipser."

I have read, understand and accept all of the instructions above. On my honor, I pledge that I have not violated the provisions of the Academic Integrity Code of Yıldız Technical University.

_____

Signature and Date

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| **10 pts** | **20 pts** | **15 pts** | **15 pts** | **15 pts** | **15 pts** | **10 pts** | **20 pts** | **20 pts** | **20 pts** |
| | | | | | | | | | |

| 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|
| **20 pts** | **20 pts** | **20 pts** | **20 pts** | **20 pts** | **20 pts** | **20 pts** |
| | | | | | | |

| Total |
|---|
| **300 pts** |
| |

1) **[10 Points]** Why do we need computation? Why do we need programming language for computation? Why do we need automats / machines to recognizes/accept programming language?

Computation is any numerical or logical process/calculation that outputs something meaningful. So we need it to think, understand and build complex models such as algorithms. In the past people think only humans can do computation. But now when we say "computation" mostly computers will come to mind. Because of that; like we people need languages to communicate, we need such as languages to communicate with computers aka machines. We call this kind of languages "programming languages". If a machine can understand more complex language, this means it has more computational power. By "understanding" we mean recognizing/accepting the given input. So in the end we need machines which understands our "programming language" to do computations we want to get results.

2) **[20 Points]** For any $n \in \mathbb{N}$, prove that the following equality is valid.

$$1^6 + 2^6 + 3^6 + \cdots + n^6 = \frac{n}{42}(n+1)(2n+1)(3n^4 + 6n^3 - 3n + 1)$$

we're gonna prove this equation by induction. For this we need to prove it first for small numbers. lets prove it for 1 and 2.

$n=1$ $\quad 1^6 = \frac{1}{42}(1+1)(2+1)(3+6-3+1) \longrightarrow 1 = \frac{1}{42}\cdot 2\cdot 3\cdot 7 = \frac{42}{42} = 1$, true for 1.

$n=2$ $\quad 1^6+2^6 = \frac{2}{42}(2+1)(4+1)(3.16+6.8-3.2+1) \longrightarrow 65 = \frac{2}{42}\cdot 3\cdot 5\cdot 91 = \frac{15.91}{21} = 65$ true for 2.

now we're going to check that if this is true for $(n+1)$ when we assume that this is true for $n$.

we're going to use open form of formula $\longrightarrow \frac{n}{42}(n+1)(2n+1)(3n^4+6n^3-3n+1) = \frac{1}{42}(6n^7+21n^6+21n^5-7n^3+n)$

if the following is holds then this means this formula is true for $(n+1)$ when its true for $n$.

$$\frac{1}{42}(6n^7+21n^6+21n^5-7n^3+n) + (n+1)^6 = \frac{1}{42}\left(6(n+1)^7 + 21(n+1)^6 + 21(n+1)^5 - 7(n+1)^3 + (n+1)\right)$$

$$\frac{1}{42}\left(6n^7+21n^6+21n^5-7n^3+n+42(n^6+6n^5+15n^4+20n^3+15n^2+6n+1)\right) = \frac{1}{42}\begin{pmatrix} 6(n^7+7n^6+21n^5+35n^4+35n^3+21n^2+7n+1)+ \\ 21(n^6+6n^5+15n^4+20n^3+15n^2+6n+1)+ \\ 21(n^5+5n^4+10n^3+10n^2+5n+1)-7(n^3+3n^2+3n+1)+n+1 \end{pmatrix}$$

$$\frac{1}{42}\left(6n^7+63n^6+273n^5+630n^4+833n^3+630n^2+253n+42\right) = 2\,\frac{1}{42}\left(6n^7+63n^6+273n^5+630n^4+833n^3+630n^2+253n+42\right)$$

we know that formula true for 2 and any $n+1$ if it is for $n$. and since we can reach any number from 2, this is also for any $n \in \mathbb{N}$. proved by induction. □

3) **[15 Points]** Let $\alpha$ and $\beta$ be two positive integer numbers. If $\alpha^2 - \beta^2$ is not odd, then $\alpha + \beta \geq 2$

proof by contradiction.

Lets say this assumption is false and when $a^2-b^2$ is even $a+b<2$. if we try to prove this and fail, then we can say the original assumption is correct.

if $(a-b)(a+b)$ is even, this means;

a) $(a-b)(a+b)=0$

which means $a=b$, and since both positive min a and b can be 1,

$a+b \geq 1+1 \quad 1+1=2<2$ false

b) $(a-b)(a+b)=2k$ or $-2k$ → $(a-b)$ or $(a+b)$ or both need to be even

i) a-b even → $a=b+2k, k>0 \quad a+b=2b+2k<2$ false

$b=a+2k, k>0 \quad a+b=2a+2k<2$ false

ii) a+b even → a and b odd

$a=2k+1 \quad b=2n+1 \quad a+b=2k+1+2n+1<2$ false

a and b even

$a=2k \quad b=2n \quad a+b=2k+2n<2$ false

4) **[15 Points]** Let $a, b, c, d$ be integers. If $a > c$ and $b > c$, then prove that $\max(a, b) - c$ is always positive.

proof by contradiction. let say this statement is not true. in this case $\max(a,b)-c$ would be equal to 0 (zero) or negative number.

if $\max(a,b)-c=0$

$\max(a,b)=c$

since $a>c$ and $b>c$

a or b cannot be equal to c

} cannot be true

if $\max(a,b)-c<0$

$\max(a,b)<c$

since $a>c$ and $b>c$

a or b cannot be smaller than c

} cannot be true

since these statements cannot be true. we left with only one statement which means $\max(a,b)-c$ is always $>0$ aka positive.

5) **[15 points]** Given two sets $X$ and $Y$. The Cartesian product of $X$ and $Y$, written as $X \times Y$, is defined as the set of pairs $(x, y)$ where $x \in X$ and $y \in Y$. Then, find a mathematical closed-form expression to write $|X \times Y|$ in terms of $|X|$ and $|Y|$.

$$X \times Y = \{(x,y) \mid x \in X \text{ and } y \in Y\}$$

$$|X \times Y| = |X| \cdot |Y|$$

6) **[15 points]** Let us given two disjoint sets $X$ and $Y$, and then their joint set $S$ is $S = X \cup Y$. Sum of the elements in a set $S$ is denoted by $\sum(S)$ while their product is by $\prod(S)$. Accordingly, what are $\sum(S)$ and $\prod(S)$ in terms of $\sum(X)$, $\sum(Y)$, $\prod(X)$ and $\prod(Y)$. Conclude from this what $\sum(\emptyset)$ and $\prod(\emptyset)$ should be ($\emptyset$ is the empty set).

$\Sigma(S) = \Sigma(X \cup Y) = \Sigma(X) + \Sigma(Y) - \Sigma(X \cap Y)$ since $X \cap Y = \emptyset \rightarrow = \Sigma(X) + \Sigma(Y)$

$\Pi(S) = \Pi(X \cup Y) = \dfrac{\Pi(X) \cdot \Pi(Y)}{\Pi(X \cap Y)}$ since $X \cap Y = \emptyset \rightarrow = \Pi(X) \cdot \Pi(Y)$

This give us $\rightarrow$ $\Sigma(\emptyset) = 0$ and $\Pi(\emptyset) = 1$

$\Sigma(X) + \Sigma(Y) - \Sigma(\emptyset) = \Sigma(X) + \Sigma(Y)$

$-\Sigma(\emptyset) = 0 \rightarrow \Sigma(\emptyset) = 0$ ✓

$\dfrac{\Pi(X) \cdot \Pi(Y)}{\Pi(\emptyset)} = \Pi(X) \cdot \Pi(Y)$

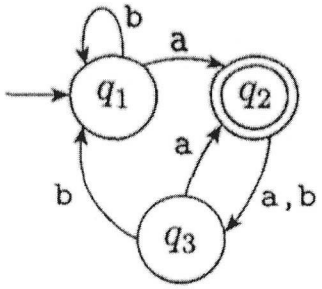
$\dfrac{1}{\Pi(\emptyset)} = 1 \rightarrow \Pi(\emptyset) = 1$

7) **[10 points]** What is the relation between programming language and the power of a machine that recognizes / accepts that programming language? Give an example in your explanation.

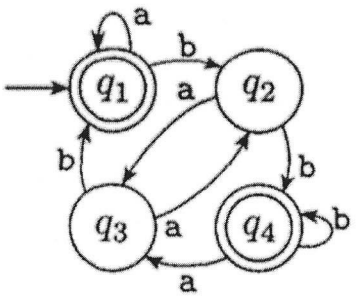
As programming language gets more complex; the machine that recognizes/accepts that programming language becomes more powerfull which means it has more computational capability/power.

ex: Let say we have $L_1 = \{\}$ and $L_2 = \{w \in \{0,1\}^* \mid w = (1^*(00)^*1^*)^+\}$ we can obviously see that $L_2$ is more complex than $L_1$. Lets say we also have two machines, $M_1$ that recognizes $L_1$ and $M_2$ that recognizes $L_2$. As we can see $M_1$ would have little computational power as it dont need to do much because it have only one state and no transitions at all. In the other hand $M_2$ has to deal multiple states with multiple transitions which has also loops in it. By that we can say that $M_2$ has more computational power than $M_1$; because $M_2$ has to recognize more complex language than $M_1$.

8) **[20 Points]** The following are the state diagrams of two DFAs, $M_1$ and $M_2$ . Answer the following questions about each of these machines.




$M_1$




$M_2$

a) What is the start state?

for $M_1$

$= q_1$

for $M_2$

$= q_1$

b) What is the set of accept states?

for $M_1$

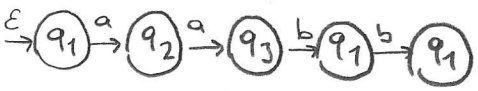
$F = \{q_2\}$

for $M_2$

$F = \{q_1, q_4\}$

c) What sequence of states does the machine go through on input *aabb* ?

for $M_1$




for $M_2$


d) Does the machine accept the string *aabb* ?

for $M_1$

NO

for $M_2$

YES

e) Does the machine accept the string $\varepsilon$?

for $M_1$

NO

for $M_2$

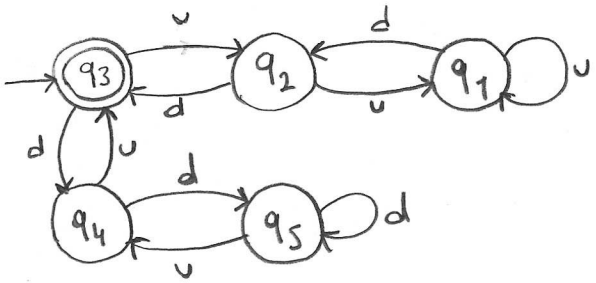
YES

9) **[20 Points]** The formal 5-tupple description of a DFA $M$ is

$$(\{q_1, q_2, q_3, q_4, q_5\}, \{u, d\}, \delta, q_3, \{q_3\}),$$

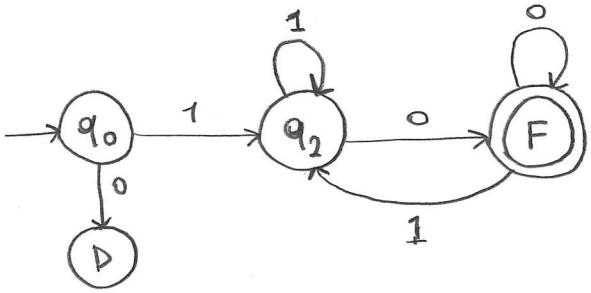
where $\delta$ is given by the following table. Give the state diagram of this machine.

| | u | d |
|---|---|---|
| $q_1$ | $q_1$ | $q_2$ |
| $q_2$ | $q_1$ | $q_3$ |
| $q_3$ | $q_2$ | $q_4$ |
| $q_4$ | $q_3$ | $q_5$ |
| $q_5$ | $q_4$ | $q_5$ |

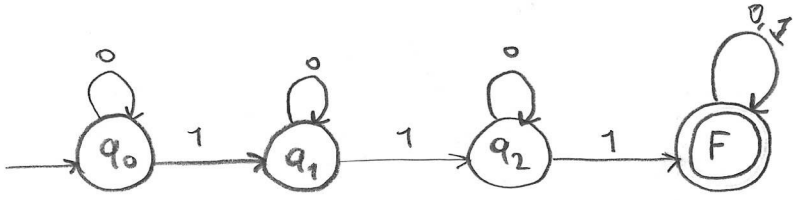



10) **[20 Points]** Give state diagrams of DFAs recognizing the following languages. In all parts the alphabet is {0,1}.

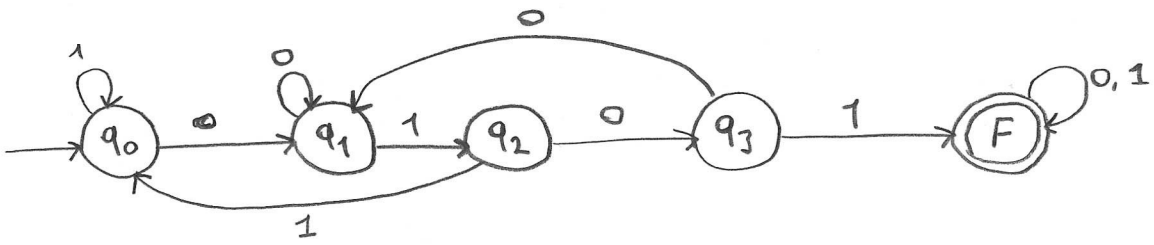
a) {w | w begins with a 1 and ends with a 0}



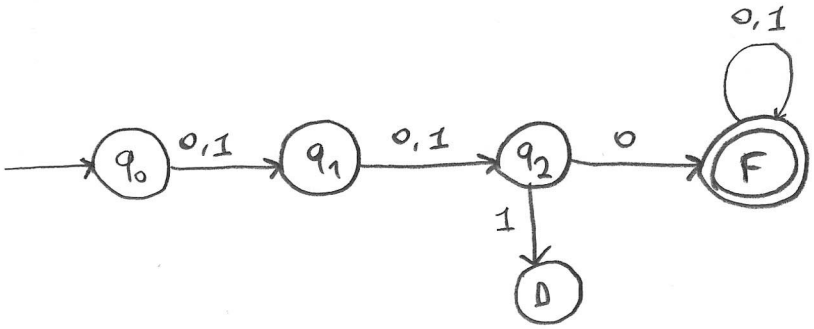

b) {w | w contains at least three 1s}



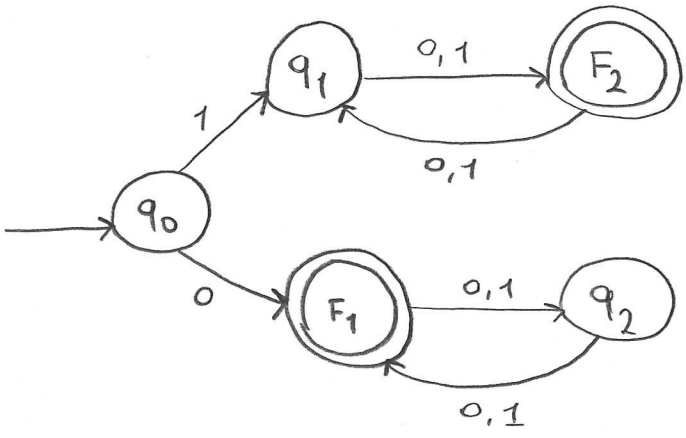
c) {w | w contains the substring 0101, i.e., w = x0101y for some x and y}



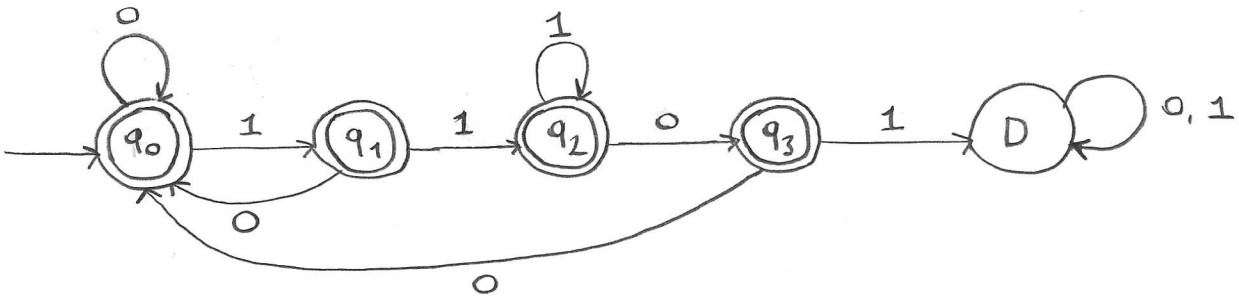

d) {w |w has length at least 3 and its third symbol is a 0}



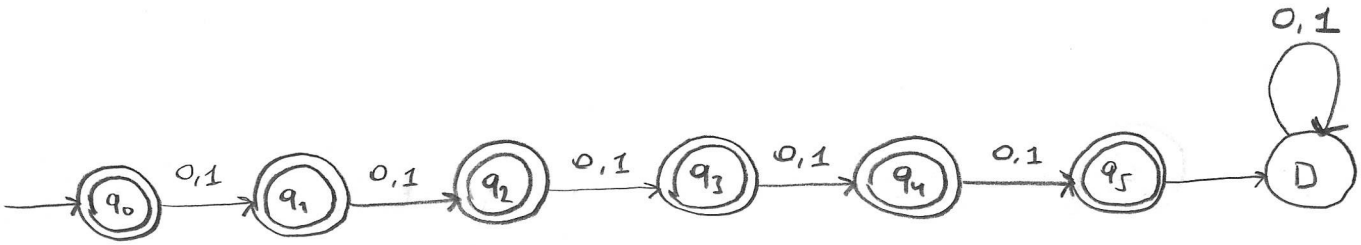

e) {w | w starts with 0 and has odd length, or starts with 1 and has even length}



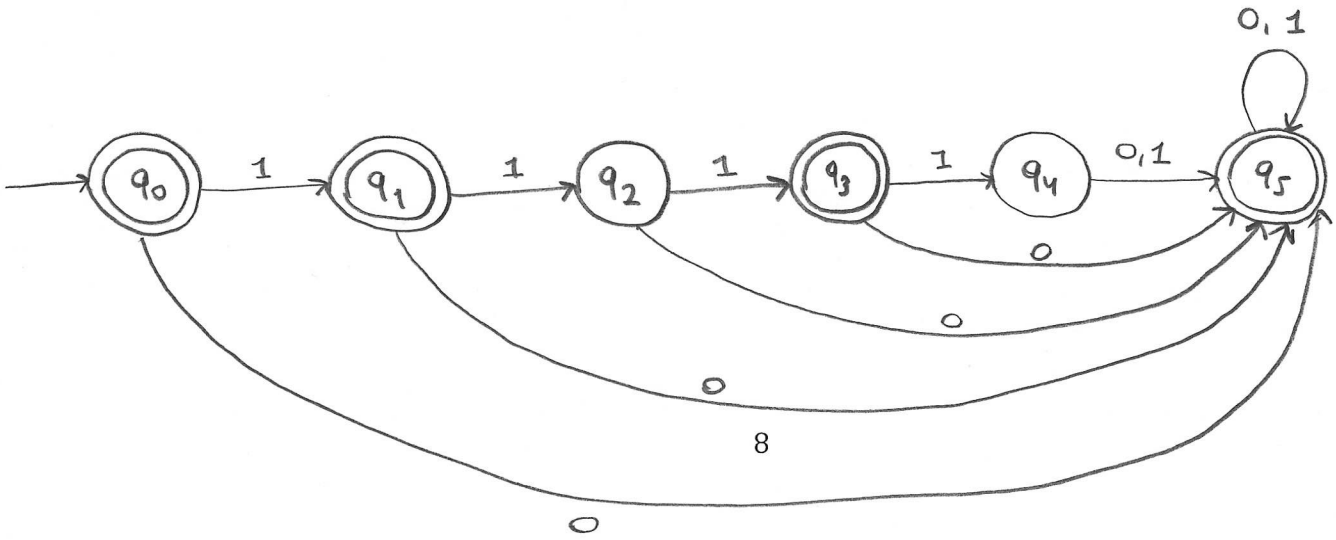
f) {w | w doesn't contain the substring 1101}



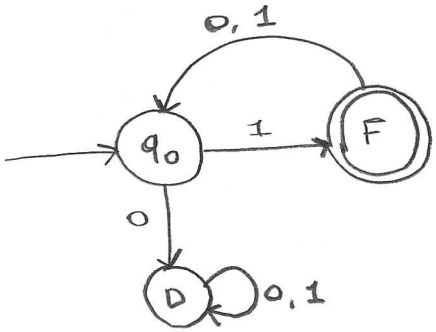

g) {w | w the length of w is at most 5}



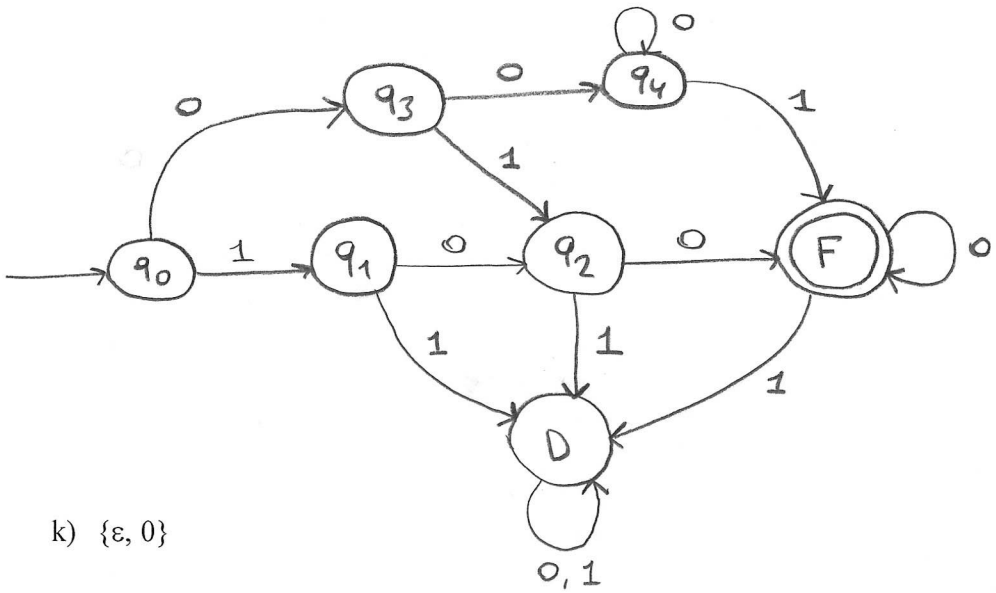

h) {w | w is any string except 11 and 1111}



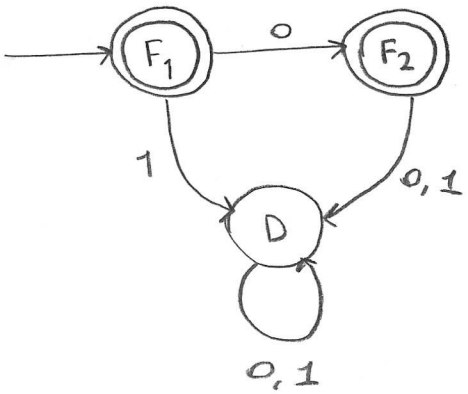
i) {w | every odd position of w is a 1}



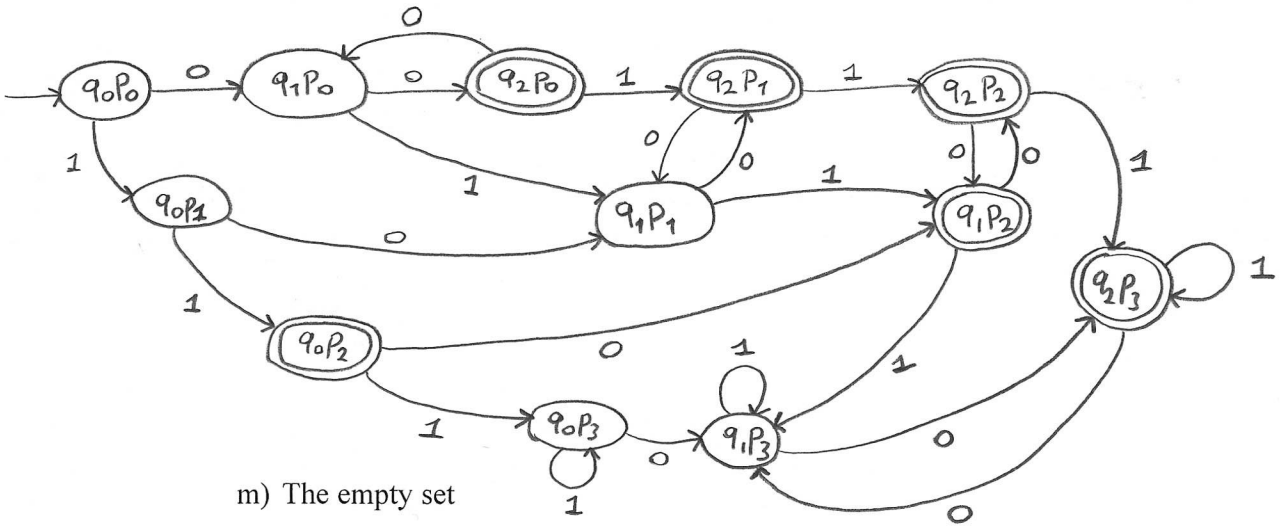

j) {w | w contains at least two 0s and at most one 1}



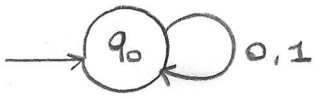

k) {ε, 0}



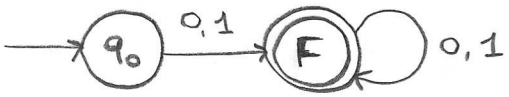
l) {w | w contains an even number of 0s, or contains exactly two ls}




m) The empty set




n) All strings except the empty string



11) **[20 Points]** Show by giving an example that if $M$ is a DFA that recognizes language $C$, swapping the final and non-final states in $M$ yields a new DFA that recognizes $\bar{C}$.

Lets say M is a DFA

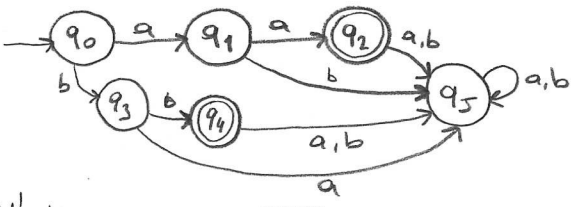
and $L(M) = \{aa, bb\}$ over $\Sigma = \{a, b\}$

then $\overline{L(M)}$ would be;

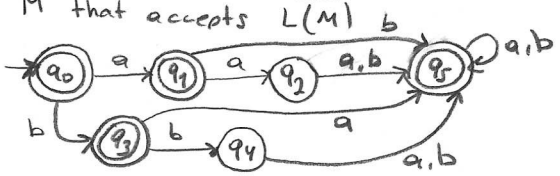
$\overline{L(M)} = \Sigma^* \setminus L(M) = \{a,b\}^* \setminus \{aa, bb\}$

Lets draw state diagrams

M that accepts L(M)



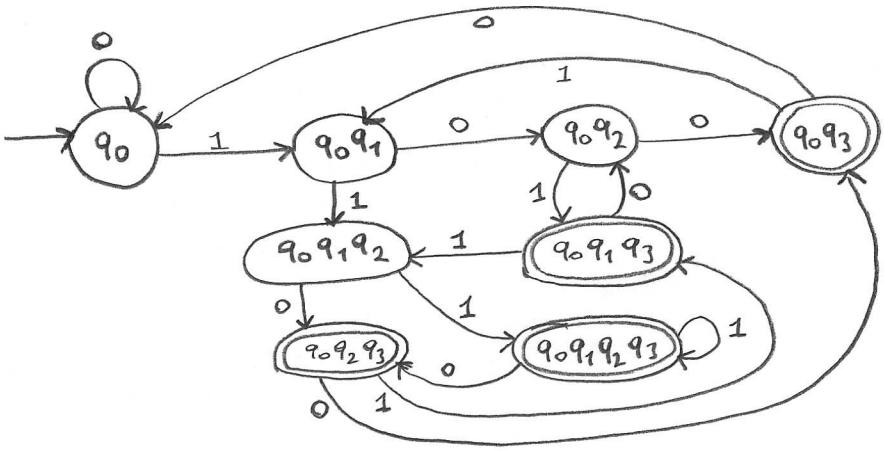

M' that accepts $\overline{L(M)}$



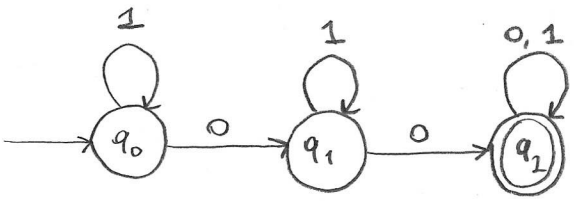

As you can see, by swapping final and non-final states we made a new DFA that accepts every thing except L(M). This means our M' accepts $\overline{L(M)}$.

12) **[20 Points]** Design automata (DFA) to accept the following languages:

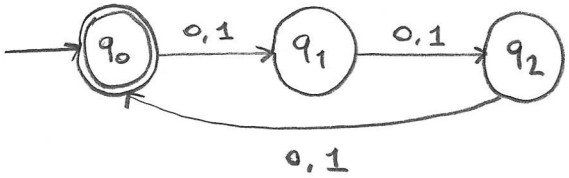
a) A = {w ∈ {0, 1}* : w has a 1 in the third position from the right}.



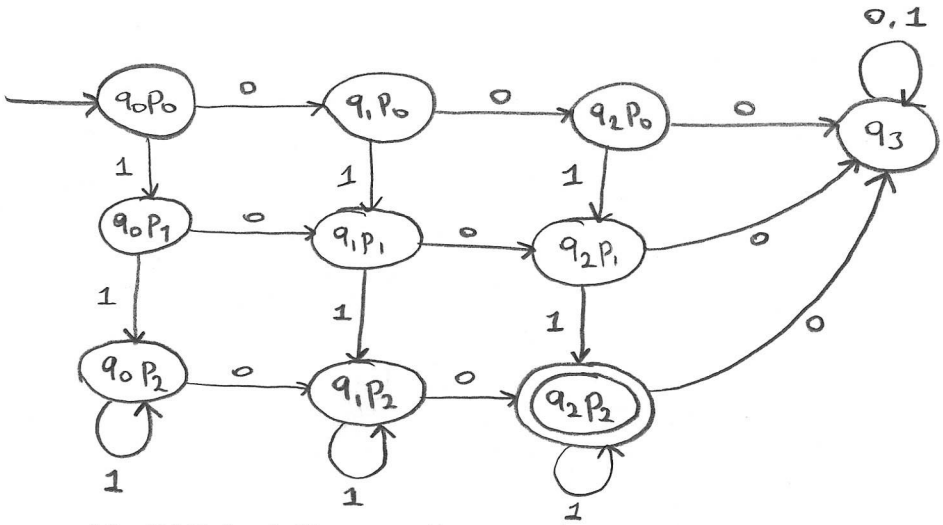

b) B = {w ∈ {0, 1}* : w contains at least two 0s}



c) C = {w ∈ {0, 1}* : the length of w is divisible by three}

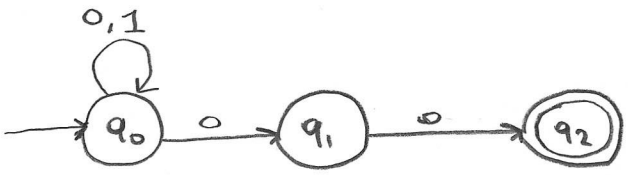



d) D = {w ∈ {0, 1}* : w contains exactly two 0s and at least two 1s}.

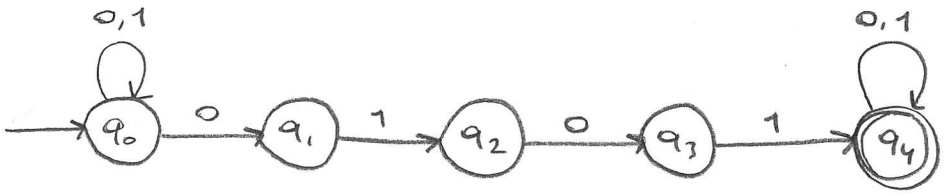



13) **[20 Points]** Give state diagrams of NFAs with the specified number of states recognizing each of the following languages. In all parts the alphabet is {0,1}.

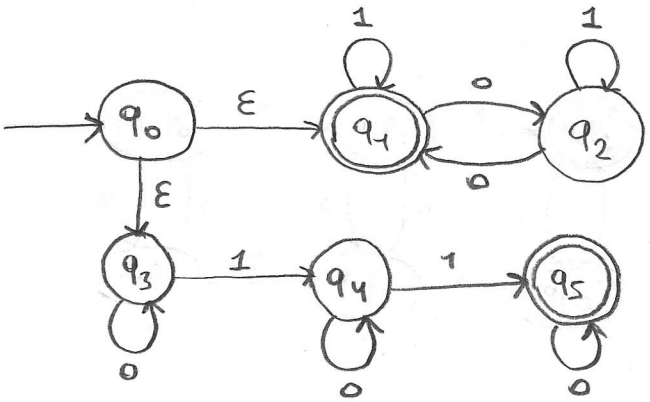
a) The language {w | w ends with 00} with three states



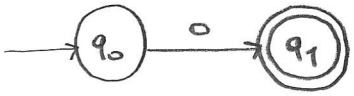
b) The language {w | w contains the substring 0101, i.e., w = x0101y for some x and y} with five states



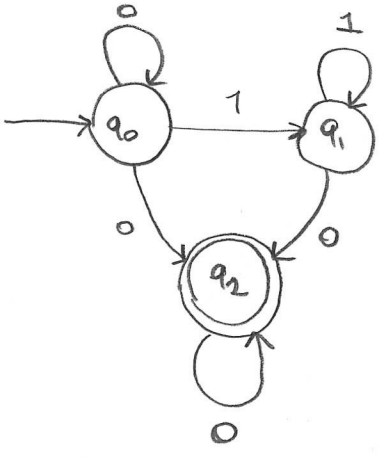

c) The language {w | w contains an even number of 0s, or contains exactly two ls} with six states



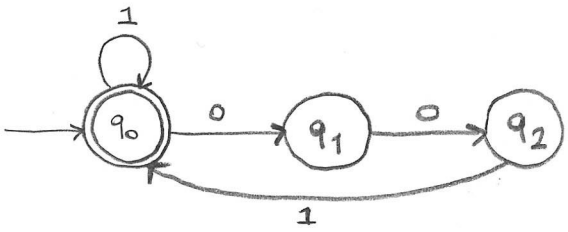

d) The language {0} with two states




e) The language $0^*1^*0^+$ with three states



f) The language $1^*(001^+)^*$ with three states

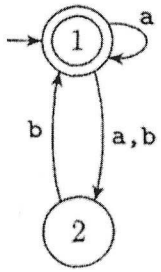



g) The language $\{\varepsilon\}$ with one state

h) The language $0^*$ with one state


14) **[20 Points]** Use the construction given in Theorem 1.39 in the book to convert the following two non-deterministic finite automata to equivalent deterministic finite automata.

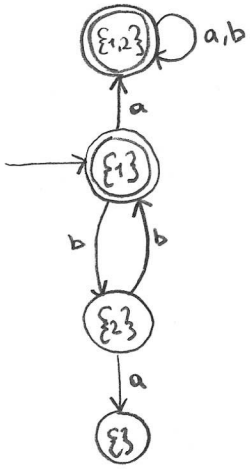
$Q=\{1,2\}$ over $\Sigma=\{a,b\}$




| $\delta$ | a | b |
|---|---|---|
| 1 | $\{1,2\}$ | $\{2\}$ |
| 2 | $\{\}$ | $\{1\}$ |

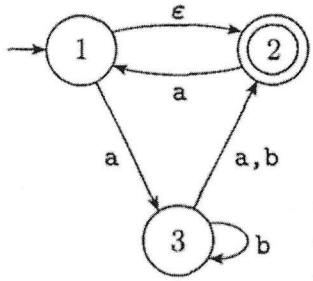
(a)

$P(Q)=\{\{\},\{1\},\{2\},\{1,2\}\}=Q_{DFA}$




| $\delta_{DFA}$ | a | b |
|---|---|---|
| $\{1\}$ | $\{1,2\}$ | $\{2\}$ |
| $\{2\}$ | $\{\}$ | $\{1\}$ |
| $\{1,2\}$ | $\{1,2\}$ | $\{1,2\}$ |

DFA $q_0=\{1\}$




$Q=\{1,2,3\}$ over $\Sigma=\{a,b\}$

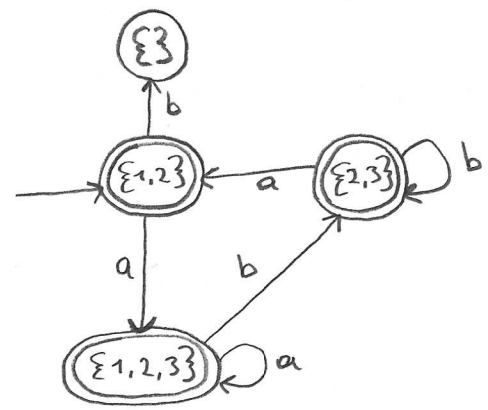
| $\delta$ | a | b |
|---|---|---|
| 1 | $\{1,2,3\}$ | $\{\}$ |
| 2 | $\{1,2\}$ | $\{\}$ |
| 3 | $\{2\}$ | $\{2,3\}$ |

(b)

$P(Q)=Q_{DFA}=\{\{\},\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\}$

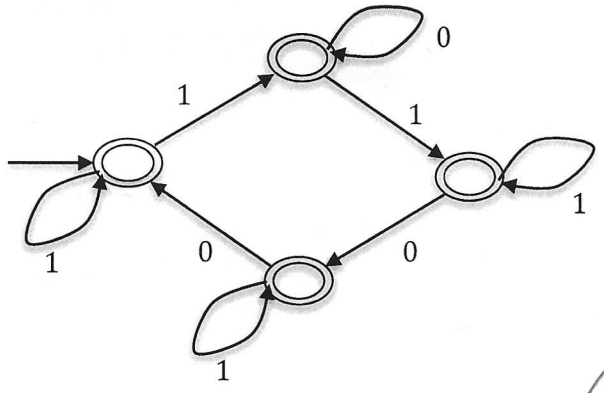
| $\delta_{DFA}$ | a | b |
|---|---|---|
| $\{1,2\}$ | $\{1,2,3\}$ | $\{\}$ |
| $\{2,3\}$ | $\{1,2\}$ | $\{2,3\}$ |
| $\{1,2,3\}$ | $\{1,2,3\}$ | $\{2,3\}$ |

DFA $q_0=\{1,2\}$



$r = (1^* + 1^*10^* + 1^*10^*11^* + 1^*10^*11^*01^* + 1^*10^*11^*01^*01^*)^*$

15) **[20 points]** For the alphabet $\Sigma_1 = \{0,1\}$, answer the following questions for the automata machine shown below




$L = \{w \in \Sigma^* \mid w \in L(reg)\}$ reg is obtained in ophan (b)

a) Is the machine DFA or NFA? Why?

Machine is an NFA but not a DFA because for some states; there is more than one transition outs from the state for a symbol in alphabet. For example initial state has two transition for symbol 1

b) Give its regular expression.

$r = (1^+0^*1^+01^*0)^*1^* \cup 1^+(0^*1^+01^*01^+)^*0^* \cup 1^+0^*1^+(01^*01^+0^*1^+)^* \cup 1^+0^*1^+01^*(01^+0^*1^+01^*)^*1^*$

c) Write the language which is a set consisting of strings that are recognized by this automaton.

$L = \{w \in \{0,1\}^* \mid w = (1^+0^*1^+01^*0)^*1^*\} \cup \{w \in \{0,1\}^* \mid w = 1^+(0^*1^+01^*01^+)^*0^*\}$
$\cup \{w \in \{0,1\}^* \mid w = 1^+0^*1^+(01^*01^+0^*1^+)^*\}$
$\cup \{w \in \{0,1\}^* \mid w = 1^+0^*1^+01^*(01^+0^*1^+01^*)^*1^*\}$

16) **[20 Points]** Give regular expressions describing the following languages:

a) A = { w ∈ {0, 1}* : w contains at least three 1s}.

$r = 0^*10^*10^*10^*1^*$ ✗ $(0+1)^*1(0+1)^*1(0+1)^*1(0+1)^*$ ✓

b) B = { w ∈ {0, 1}* : w contains at least two 1s and at most one 0},

$r = 011^+ \cup 101^+ \cup 11^+01^*$ ✓

c) C = { w ∈ {0, 1}* : w contains an even number of 0s and exactly two 1s}.

$r = (00)^*0101(00)^* \cup 0(00)^*0101(00)^*0 \cup (00)^*11(00)^* \cup 0(00)^*11(00)^*0 \cup (00)^*1010(00)^* \cup 0(00)^*1010(00)^*0$

d) D = { w ∈ {0, 1}* : w contains an even number of 0s and each 0 is followed by at least one}

$r = (1^*(00)^*1^*)^*$ ~~$1^*(01^*01^*)^*1^*$~~ ✓

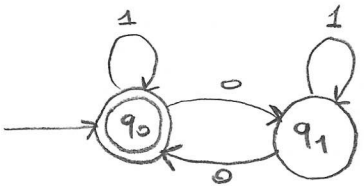
17) **[20 Points]** Design a DFA or NFA for the following languages. $n_0(w)$ denotes the number of zeros in the string w.

a) $L_1 = \{ w \in \{0, 1\}^* : n_0(w) \bmod 2 = 0 \}$,

$Q_1 = \{q_0, q_1\}$ $\Sigma = \{0,1\}$ start state $= q_0$ Final states $= \{q_0\}$

| $\delta_1$ | 0 | 1 |
|---|---|---|
| $q_0$ | $q_1$ | $q_0$ |
| $q_1$ | $q_0$ | $q_1$ |

state diagram =

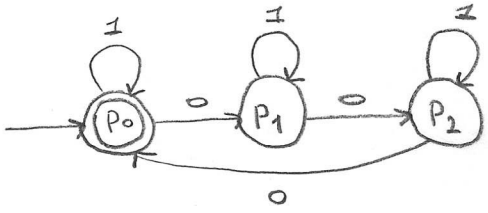



b) $L_2 = \{ w \in \{0, 1\}^* : n_0(w) \bmod 3 = 0 \}$,

$Q_2 = \{P_0, P_1, P_2\}$ $\Sigma = \{0,1\}$ start state $= P_0$ Final States $= \{P_0\}$

| $\delta_2$ | 0 | 1 |
|---|---|---|
| $P_0$ | $P_1$ | $P_0$ |
| $P_1$ | $P_2$ | $P_1$ |
| $P_2$ | $P_0$ | $P_2$ |

state diagram =




c) Based on using the NFA and DFA you designed in the options a and b, design an NFA that recognized the language $L_3 = \{ w \in \{0, 1\}^* : n_0(w) \bmod 6 = 0\}$.

Hint: De Morgan's Laws $L_1 \cap L_2 = \overline{(\overline{L_1} \cup \overline{L_2})}$ can be used for designing an NFA that recognizes the intersection of languages.

$Q_3 = Q_1 \times Q_2 = \{(q_0,P_0), (q_0,P_1), (q_0,P_2), (q_1,P_0), (q_1,P_1), (q_1,P_2)\}$ $\Sigma = \{0,1\}$

start state $= (q_0,P_0)$ Final States $= \{(q_0,P_0)\}$ (only intersection)

| $\delta_3$ | 0 | 1 |
|---|---|---|
| $q_0P_0$ | $q_1P_1$ | $q_0P_0$ |
| $q_0P_1$ | $q_1P_2$ | $q_0P_1$ |
| $q_0P_2$ | $q_1P_0$ | $q_0P_2$ |
| $q_1P_0$ | $q_0P_1$ | $q_1P_0$ |
| $q_1P_1$ | $q_0P_2$ | $q_1P_1$ |
| $q_1P_2$ | $q_0P_0$ | $q_1P_2$ |

state diagram: