Kuwait University

Faculty of Science

Computer Science Department

CS 512: Automata and Formal Languages

Fall 2023/2024

Name: Abas Almaayofi

ID: 223124100

Homework: 01

Date: 10/10/2023

Q1-

<u>1.1.2.</u> What are these sets? Write them using braces, commas, and numerals only.

e) $(\{\mathbf{1},\mathbf{3},\mathbf{5}\} \cup \{\mathbf{3},\mathbf{1}\}) \cap \{\mathbf{3},\mathbf{5},\mathbf{7}\}$

$\boldsymbol{Solution}$: $\{1,\ 3,5\}\ \cap \{3,5,7\} = \{3,5\}$

e) $\cup \{\{\mathbf{3}\},\{\mathbf{3},\mathbf{5}\},\ \cap \{\{\mathbf{5},\mathbf{7}\},\{\mathbf{7},\mathbf{9}\}\}\}$

$\boldsymbol{Solution}$: $\cup\big\{\{3\},\{3,5\},\{7\}\big\} = \{3,5,7\}$

e) $(\{\mathbf{1},\mathbf{2},\mathbf{5}\} - \{\mathbf{5},\mathbf{7},\mathbf{9}\}) \cup (\{\mathbf{5},\mathbf{7},\mathbf{9}\} - \{\mathbf{1},\mathbf{2},\mathbf{5}\})$

$\boldsymbol{Solution}$: $\{1,2\} \cup \{7,9\} = \{1,2,7,9\}$

e) $\mathbf{2}^{(\mathbf{7},\mathbf{8},\mathbf{9})} - \mathbf{2}^{(\mathbf{7},\mathbf{9})}$

$\boldsymbol{Solution}$: $\big\{\emptyset,\{7\},\{8\},\{9\},\{7,8\},\{7,9\},\{8,9\},\{7,8,9\}\big\} - \big\{\emptyset,\{7\},\{9\},\{7,9\}\big\}$

$= \{\{8\},\{7,8\},\{8,9\},\{7,8,9\}\}$

e) $\mathbf{2}^{\emptyset}$

$\boldsymbol{Solution}$: $\{\emptyset\}$

<u>1.2.1.</u> write each of the following explicitly.

a) $\{\mathbf{1}\} \times \{\mathbf{1},\mathbf{2}\} \times \{\mathbf{1},\mathbf{2},\mathbf{3}\}$

$\boldsymbol{Solution}$: $\{(1,1,1),(1,1,2),(1,1,3),(1,2,1),(1,2,2),(1,2,3)\}$

b) $\emptyset \times \{\mathbf{1},\mathbf{2}\}$

$\boldsymbol{Solution}$: $\emptyset$

c) $\mathbf{2}^{\{\mathbf{1},\mathbf{2}\}} \times \{\mathbf{1},\mathbf{2}\}$

$\boldsymbol{Solution}$: $\big\{\emptyset,\{1\},\{2\},\{1,2\}\big\} \times \{1,2\} =$
$\{(\emptyset,1),(\emptyset,2),(\{1\},1),(\{1\},2),(\{2\},1),(\{2\},2),(\{1,2\},1),(\{1,2\},2)\}$

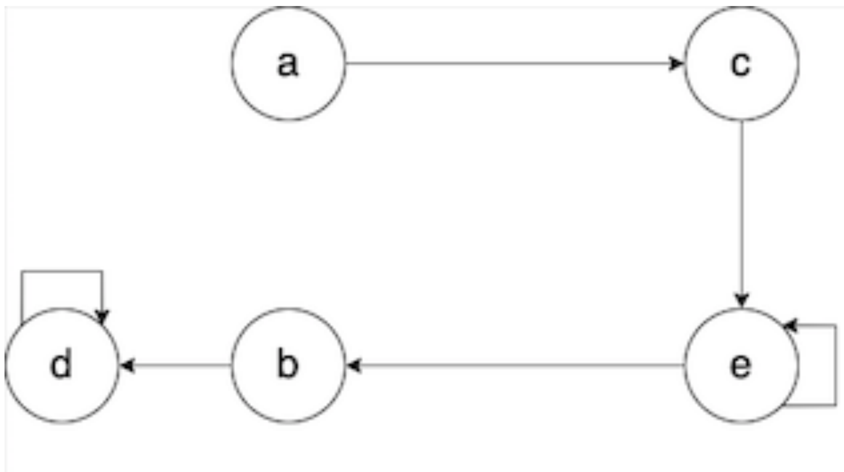
Q2-

<u>1.3.1.</u> Let $\boldsymbol{R} = \{(\boldsymbol{a}, \boldsymbol{c}), (\boldsymbol{c}, \boldsymbol{e}), (\boldsymbol{e}, \boldsymbol{e}), (\boldsymbol{e}, \boldsymbol{b}), (\boldsymbol{d}, \boldsymbol{b}), (\boldsymbol{d}, \boldsymbol{d})\}$.

$\boldsymbol{Draw\ directed\ graphs\ representing\ each\ of\ the\ following}$:

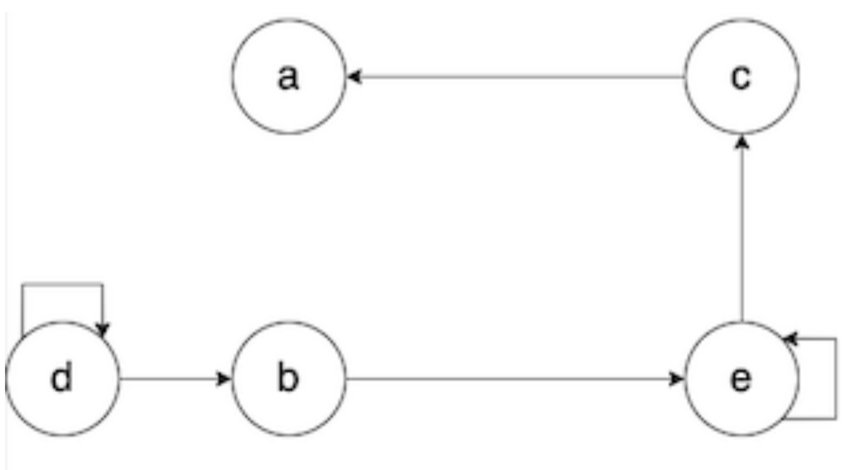
a) $\boldsymbol{R} = \{(\boldsymbol{a}, \boldsymbol{c}), (\boldsymbol{c}, \boldsymbol{e}), (\boldsymbol{e}, \boldsymbol{e}), (\boldsymbol{e}, \boldsymbol{b}), (\boldsymbol{d}, \boldsymbol{b}), (\boldsymbol{d}, \boldsymbol{d})\}$

Solution:



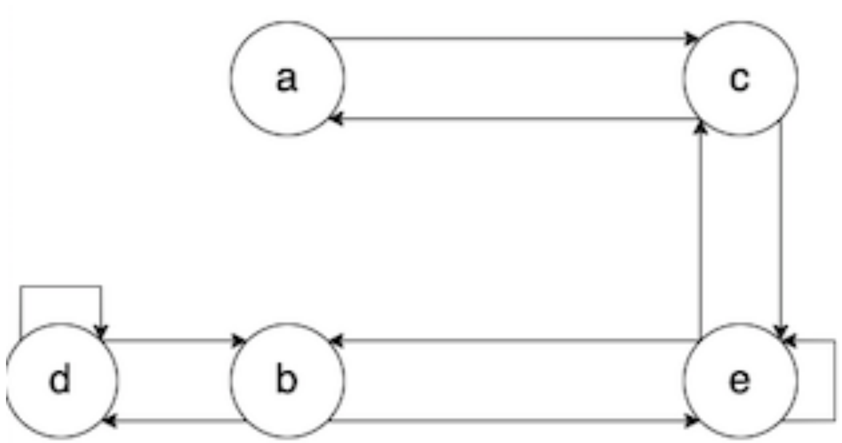
b) $\boldsymbol{R^{-1}} = \{(\boldsymbol{c}, \boldsymbol{a}), (\boldsymbol{e}, \boldsymbol{c}), (\boldsymbol{e}, \boldsymbol{e}), (\boldsymbol{b}, \boldsymbol{e}), (\boldsymbol{b}, \boldsymbol{d}), (\boldsymbol{d}, \boldsymbol{d})\}$

Solution:



c) $\boldsymbol{R} \cup \boldsymbol{R}^{-1} = \{(\boldsymbol{a}, \boldsymbol{c}), (\boldsymbol{c}, \boldsymbol{a}), (\boldsymbol{c}, \boldsymbol{e}), (\boldsymbol{e}, \boldsymbol{c}), (\boldsymbol{e}, \boldsymbol{e}), (\boldsymbol{e}, \boldsymbol{b}), (\boldsymbol{b}, \boldsymbol{e}), (\boldsymbol{d}, \boldsymbol{b}), (\boldsymbol{b}, \boldsymbol{d}), (\boldsymbol{d}, \boldsymbol{d})\}$

Solution:



d) $\boldsymbol{R} \cap \boldsymbol{R}^{-1} = \{(\boldsymbol{e}, \boldsymbol{e}), (\boldsymbol{d}, \boldsymbol{d})\}$

Solution:

Q3-

<u>1.3.5.</u> Let $\boldsymbol{f} : \boldsymbol{A} \rightarrow \boldsymbol{B}$. Show that the following relation R is an equivalence relation on $\boldsymbol{A} : (\boldsymbol{a}, \boldsymbol{b}) \in \boldsymbol{R}$ if and only if $\boldsymbol{f}(\boldsymbol{a}) = \boldsymbol{f}(\boldsymbol{b})$.

Solution:

- **To show that R is an equivalence relation, we need to prove that it satisfies: Reflexivity, Symmetry, and Transitivity.**
  - **Reflexivity:** $we\ need\ to\ show\ that\ (a, a)\ \in R. we\ know\ that\ in\ order\ for\ a\ pair\ to\ be\ in\ R, f(a)\ must\ equal\ f(b), therefore, this\ stands\ true\ for\ all: a\ \in A, such\ that\ f(a) = f(a), so\ (a, a) \in R.$
  - **Symmetry: :** $we\ need\ to\ show\ that\ if\ (a, b\ )\ \in R, then\ (b, a)\ \in R.\ but\ if\ f(a) = f(b),\ this\ also\ means\ that\ f(b) = f(a), therefore, (b, a) \in R$**.**

- **Transitivity:** *we need to show that if* $(a, b) \in R,\ and\ (b, c) \in R$, *then* $(a, c) \in R$. *Since since* $f(a) = f(b), and\ f(b) = f(c)$, *then this also means that* $(a) = f(c)$, *therefore,* $(a, c) \in R$.

Q6-

<u>1.6.3.</u> Is the transitive closure of the symmetric closure of a binary relation necessarily reflexive? Prove it or give a counterexample.

Solution:

- No, it's not necessarily reflexive, counterexample: assume the following symmetric relation $R = \{(3, 4), (4, 3)\}$, we find the smallest transitive closure of the relation: $R = \{(3, 4), (4, 3), (3,3)\}$. We can clearly see that $(4,4) \notin R$, which makes the relation only symmetric.

<u>1.6.4.</u> Let $R \sqsubseteq A \times A$ be any binary relation.

a) Let $Q = \{(a, b) :\ a, b \in A$ and there are paths in R from a to b and from b to a $\}$. Show that Q is an equivalence relation on A.

Solution:

- Q is an equivalence relation iff it satisfies reflexivity, symmetry, and transitivity.
  - **Reflexivity:** as stated in the definition of Q, that $\forall\ (a, b) \in Q$ there exists paths from a to b, for our own purpose we can assume that, $b = a$, and therefore, $(a, a) \in Q$.
  - **Symmetry:** the definition also states, that $\forall\ (a, b) \in Q$ there exists paths from a to b and from b to a, therefore $(a, b) \in Q$, and $(b, a) \in Q$, since the opposite is also true.
  - **Transitivity:** if $(a, b) \in R$, meaning there is a path from a to b, and $(b, c) \in R$, there is a path from b to c, then there must be a path from a to c, $(b, c) \in R$, therefore, if $(a, b) \in Q$, and $(b, c) \in Q$, then $(a, c) \in Q$.
- Hence, Q is an equivalence relation.

Q8-

<u>2.1.3.</u> Construct deterministic finite automata accepting each of the following languages.

a) $\{\boldsymbol{w} \in \{\boldsymbol{a}, \boldsymbol{b}\}^* : \boldsymbol{each\ a\ in\ w\ is\ immediately\ preceded\ by}$ **a** $\boldsymbol{b}\}$

***Solution*:**

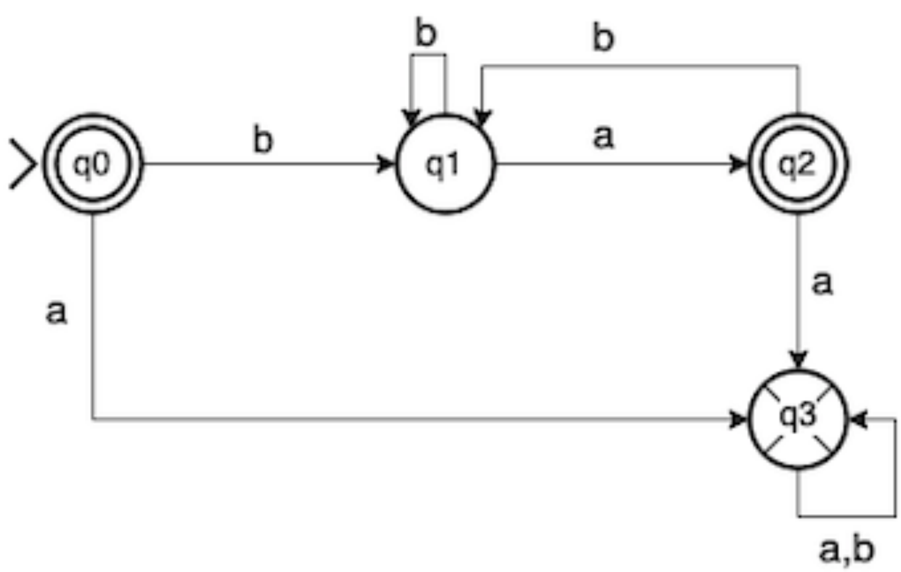
- $M = (K, \Sigma, \delta, s, F)$

  $K = \{q_{0,} q_1, q_2, q_{3,}\}$, $\Sigma = \{a, b\}$, $s = q_0$, $F = \{q_{0,} q_2\}$,

  and δ is given by the following table:

q	σ	$\delta(q, \sigma)$
q_0	a	q_3
q_0	b	q_1
q_1	a	q_2
q_1	b	q_1
q_2	a	q_3
q_2	b	q_1
q_3	a	q_3
q_3	b	q_3

DFA graph:



b) $\{\boldsymbol{w} \in \{\boldsymbol{a}, \boldsymbol{b}\}^* : \boldsymbol{w\ has\ abab\ as\ a\ substring}\}$

***Solution*:**

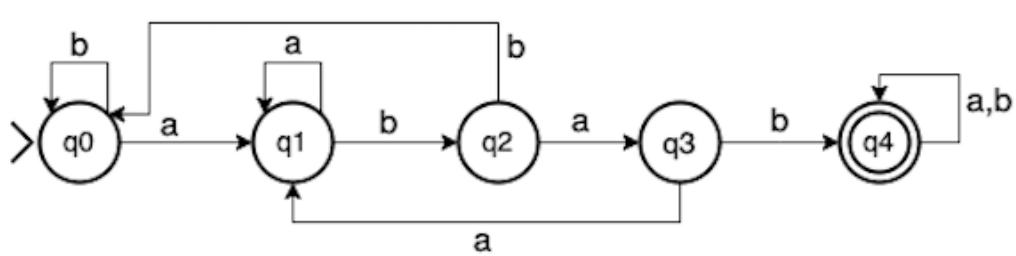
- $M = (K, \Sigma, \delta, s, F)$

  $K = \{q_0, q_1, q_2, q_3, q_4\}$, $\Sigma = \{a, b\}$, $s = q_0$, $F = \{q_4\}$,

  and δ is given by the following table:

q	σ	$\delta(q, \sigma)$
q_0	a	q_1
q_0	b	q_0
q_1	a	q_1
q_1	b	q_2
q_2	a	q_3
q_2	b	q_0
q_3	a	q_1
q_3	b	q_4
q_4	a	q_4
q_4	b	q_4

DFA graph:



e) $\{\boldsymbol{w} \in \{\boldsymbol{a}, \boldsymbol{b}\}^* : \boldsymbol{w\ has\ both\ ab\ and\ ba\ as\ substrings}\}$

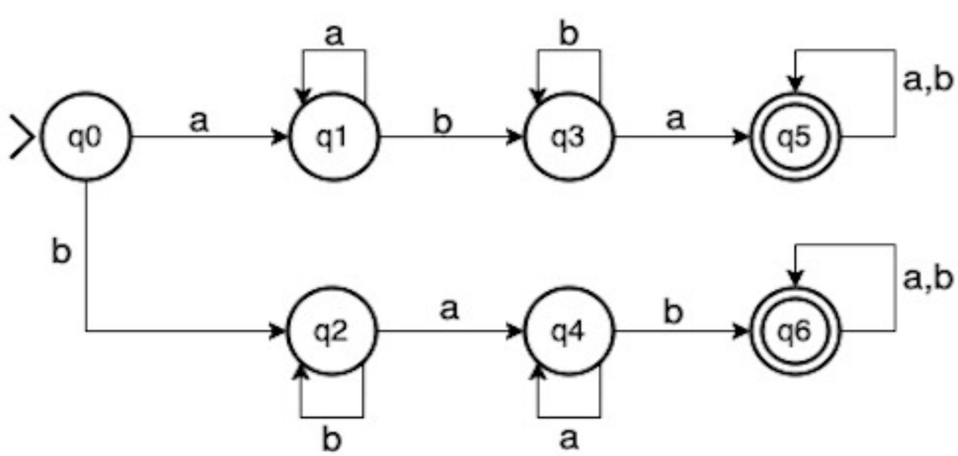
***Solution*:**

- $M = (K, \Sigma, \delta, s, F)$

  $K = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$, $\Sigma = \{a, b\}$, $s = q_0$, $F = \{q_5, q_6\}$, and δ is given by the following table:

q	σ	$\delta(q,\sigma)$
q_0	a	q_1
q_0	b	q_2
q_1	a	q_1
q_1	b	q_3
q_2	a	q_4
q_2	b	q_2
q_3	a	q_5
q_3	b	q_3
q_4	a	q_4
q_4	b	q_6
q_5	a	q_5
q_5	b	q_5
q_6	a	q_6
q_6	b	q_6

DFA graph:



Q9.

Write a program that reads a string over {0, 1} * and accepts (outputs "ACCEPTED") only strings that have a substring 0110 and do not have 1001 as a substring. Your program must decide to accept or not based only on a deterministic finite state automaton (So you must encode the automaton in your program, see the example below). You may use C or C++. Submit only the source and two sample run outputs for each case (accept/reject).

Solution:

Source code of the program (using C):

```
#include <stdio.h>
#include <string.h>
#include<stdlib.h>

int main(int argc, const char * argv[]) {
    char *word = malloc(256);
    printf("Enter word {0,1}*:");
    scanf("%255s", word);
    int state = 0;
    char symbol;
    char *output = "REJECTED";
    do {
        symbol = word[0];
        word = word + 1;
        switch (state) {
            // state q0
            case 0:
                switch (symbol) {
                    case '0':
                        state = 1;
                        break;
                    case '1':
                        state = 2;
                        break;
                }
                break;
            // state q1
            case 1:
                switch (symbol) {
                    case '0':
                        state = 1;
                        break;
                    case '1':
                        state = 3;
                        break;
                }
                break;
            // state q2
            case 2:
                switch (symbol) {
```

```
        case '0':
            state = 4;
            break;
        case '1':
            state = 2;
            break;
    }
    break;
// state q3
case 3:
    switch (symbol) {
        case '0':
            state = 4;
            break;
        case '1':
            state = 5;
            break;
    }
    break;
// state q4
case 4:
    switch (symbol) {
        case '0':
            state = 6;
            break;
        case '1':
            state = 3;
            break;
    }
    break;
// state q5
case 5:
    switch (symbol) {
        case '0':
            state = 7;
            break;
        case '1':
            state = 2;
            break;
    }
    break;
// state q6
case 6:
    switch (symbol) {
        case '0':
            state = 1;
            break;
        case '1':
            state = 8;
            break;
    }
    break;
// state q7 : final state
case 7:
```

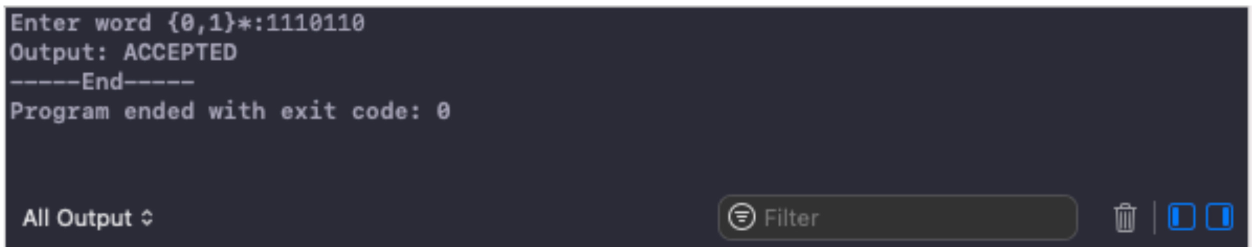
```
        output = "ACCEPTED";
        switch (symbol) {
            case '0':
                state = 9;
                break;
            case '1':
                state = 10;
                break;
        }
        break;
    // state q8 : trap state
    case 8:
        output = "REJECTED";
        switch (symbol) {
            case '0':
                state = 8;
                break;
            case '1':
                state = 8;
                break;
        }
        break;
    // state q9 : final state
    case 9:
        switch (symbol) {
            case '0':
                state = 11;
                break;
            case '1':
                state = 8;
                break;
        }
        break;
    // state q10: finale state
    case 10:
        switch (symbol) {
            case '0':
                state = 7;
                break;
            case '1':
                state = 10;
                break;
        }
        break;
    // state q11: finale state
    case 11:
        switch (symbol) {
            case '0':
                state = 11;
                break;
            case '1':
                state = 10;
                break;
        }
```

```
                break;

            }
        } while (symbol != '\0');
        printf("Output: %s\n", output);
        printf("-----End-----\n");
        return 0;
}
```

Run Outputs:

- Case “ACCEPTED”:



- Case “REJECTED”: