

מבחן בגרות 2010

פרטים כלליים

- מועד הבחינה : בכל זמן
 מספר השאלון : 1
 משך הבחינה : 3 שעות
 חומר עזר בשימוש : הכל (ספרים ומחברות)
 המלצות : קרא המלצות לפני הבחינה ובדיקות אחרונות לפני מסירה (עמודים 8-11)

מבנה השאלון

| | | |
|---------------------------|---|----------------|
| פרק ראשון - עיצוב תוכנה | 2 שאלות (בחירה מ-4) לכל שאלה – 25 נקודות | סה"כ 50 נקודות |
| פרק שני - מודלים חישוביים | 2 שאלות (בחירה מ-4) לכל שאלה – 25 נקודות | סה"כ 50 נקודות |

תוכן עניינים של פתרון המבחן

פרק ראשון - עיצוב תוכנה

- שאלה 1 : מחסנית [פיתוח פעולה]
- שאלה 2 : רשימות [פיתוח פעולה]
- שאלה 3 : טיפוסים [פיתוח מחלקה ופיתוח פעולות]
- שאלה 4 : תור [מעקב וניתוח פעולות]

פרק שני - מודלים חישוביים

- שאלה 13 : רגולריות- אוטומט סופי דטרמיניסטי, פעולות על מילים ושפות [קביעת רגולריות]
- שאלה 14 : שפות רגולריות- אוטומט סופי [בניית מודל]
- שאלה 15 : שפות רגולריות, פעולות על מילים ושפות [שייכות מילים, זיהוי שפה, השלמת אוטומט]
- שאלה 16 : מכונת טיורינג

מספר פתרונות



פתרון שאלה 1

נושא מרכזי: מחסנית

סוג השאלה: פיתוח פעולה

נתוני השאלה: כתוב פעולה המקבלת שתי מחסניות st1, st2 שמכילות מספרים שלמים וגדולים מאפס. הפעולה מחזירה את סכום זוג האיברים הסמוכים מ-st1, הקרובים ביותר לראש המחסנית, הגדול מכל זוגות האיברים הסמוכים במחסנית st2.

גרסת פתרון 1

/* הפעולה מקבלת מחסנית st המכילה מספרים שלמים וגדולים מאפס. הפעולה מחזירה את סכום זוג האיברים הסמוכים הגדול ביותר ב-st. הנחה: במחסנית יש לפחות שני איברים */

```
public static int MaxStPairSum (Stack< Integer > st)
{
    int a = st.Pop();
    int b = st.Pop();
    int sum = a + b;
    while (!st.IsEmpty())
    {
        a = b;
        b = st.Pop();
        if (a + b > sum)
            sum = a + b;
    }
    return sum;
}
/* פעולה זו מחזירה את סכום זוג האיברים הסמוכים הקרוב ביותר לראש המחסנית st1 הגדול מהסכום של כל זוג איברים סמוכים ב-st2. */
public static int BigPairSum(Stack<int> st1, Stack<int> st2)
{
    int max = MaxStPairSum(st2);
    int a = st1.Pop();
    int b = st1.Pop();
    if (a + b > max)
        return a + b;
    while (!st1.IsEmpty())
    {
        a = b;
        b = st2.Pop();
        if (a + b > max)
            return a + b;
    }
    return 0;
}
```

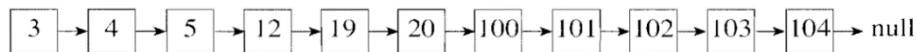
```
public static int MaxStPairSum (Stack<int> st)
{
    int val1 = st.Pop();
    int val2 = st.Pop();
    int max = val1 + val2;
    while ( ! st.IsEmpty() )
    {
        if ( val2 + st.Top() > max )
            max = val2 + st.Top();
        val2 = st.Pop();
    }
    return max;
}
```

פתרון שאלה 2

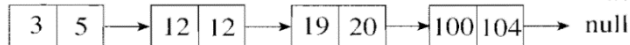
נושא מרכזי: רשימות
סוג השאלה: פיתוח פעולה

נתוני השאלה: L היא רשימה המכילה מספרים שלמים שונים זה מזה וממוינים בסדר עולה. רשימת הטווחים ב-L היא רשימה חדשה שנבנית באופן הזה: בעבור כל רצף של מספרים עוקבים ב-L יהיה ברשימת הטווחים איבר אחד שמכיל שני מספרים. מספר אחד הוא המספר הקטן ביותר ברצף, והמספר השני הוא המספר הגדול ביותר ברצף. רצף יכול להיות באורך 1 או יותר. אם הרצף הוא באורך 1, הוא מיוצג ברשימת הטווחים על-ידי איבר ששני המספרים בו שווים.

לדוגמה, בעבור הרשימה L שלפניך:



רשימת הטווחים של L תהיה:



טענת כניסה: הפעולה חיצונית מקבלת רשימה לא ריקה, המכילה מספרים שלמים שונים זה מזה ממוינים בסדר עולה ומחזירה את רשימת הטווחים שלה.

/* הפעולה מחזירה את רשימת הטווחים של list */

```
public static List<RangeNode> CreateRangeList(List<int> sourceList)
{
    List<RangeNode> lst = new List<RangeNode>();
    Node<RangeNode> pLst = null;
    Node<int> p = sourceList.GetFirst();
    int from, to;
    bool in;
    while ( p != null )
    {
        from = p.GetInfo();
        to = from;
        p = p.GetNext();
    }
}
```

```

in = true;
while (p != null && in)
{
    if (p.GetInfo() == to + 1)
    {
        to = p.GetInfo();
        p = p.GetNext();
    }
    else
        in = false;
}
pLst = lst.Insert (pLst, new RangeNode(from, to));
}
return lst;
}

```

פתרון שאלה 3

נושא מרכזי: טיפוסים

סוג השאלה: פיתוח מחלקה ופיתוח פעולות

א. כותרת המחלקה Message המייצגת הודעה ב- MessageBox ואת התכונות שלה:

```

class Message
{
    private int counter;           // מספור להודעות החדשות
    private string sender;        // שם השולח
    private string content;       // תוכן ההודעה
    private int id;               // מספר סידורי
    private int size;             // גודל ההודעה
}

```

ב. כותרת המחלקה MessageBox והתכונות שלה:

```

class MessageBox
{
    private string name;          // שם בעל התיבה
    private List<Message> inbox;  // רשימת ההודעות הפעילות
    private List<Message> bin;    // רשימת ההודעות הסל האשפה
}

```

ג. מימוש הפעולה addMessage המוצגת בממשק המחלקה MessageBox

```

public bool AddMessage (Message m)
{
    if (GetActiveSize() + GetBinSize() + m.GetSize() <=100)
    {
        inbox.Insert (null, m);
        return true;
    }
    if (GetBinSize()>=m.GetSize())
    {
        int sum=-0;
        while (sum<=m.GetSize())
            sum+=RemoveFromBin();
        inbox.Insert (null, m);
    }
}

```

```

return true;
}
return false;
}

```

- ד. לאחר שגולשים פטפטנים רבים ביקשו להגדיל את התיבה, הוחלט שפעם ביום תעבור המערכת על כל התיבות של הגולשים ותקטין או תגדיל את התיבות שלהם לפי הפעילות שלהם: תיבה שהתפוסה שלה יותר מ-80% תוגדל פי 2, ותיבה שהתפוסה שלה פחות מ-30% תוקטן בחצי.
- לאור החלטה זו צריך לשנות את הפעולה שנכתבה בסעיף ב.
 - השינויים: יש להוסיף תכונה שתייצג את גודל התיבה, ולא תחל את התכונה ל-100 בפעולה הבונה.
- (2) לאור החלטה זו צריך לשנות את הפעולה שנכתבה בסעיף ג.
- הסיבה: מאחר וכעת MaxSize שומר לכל תיבת דואר את הגודל שלה יש להשתמש בפעולה המאחזרת את הגודל.

פתרון שאלה 4

נושא מרכזי: תור
סוג השאלה: מעקב וניתוח פעולות

נתוני השאלה: נתון התור myQueue מטיפוס `Queue<int>`:

הוצאת ערכים →

| | | | | |
|---|---|---|---|---|
| 9 | 8 | 1 | 3 | 4 |
|---|---|---|---|---|

 ← הכנסת ערכים

```

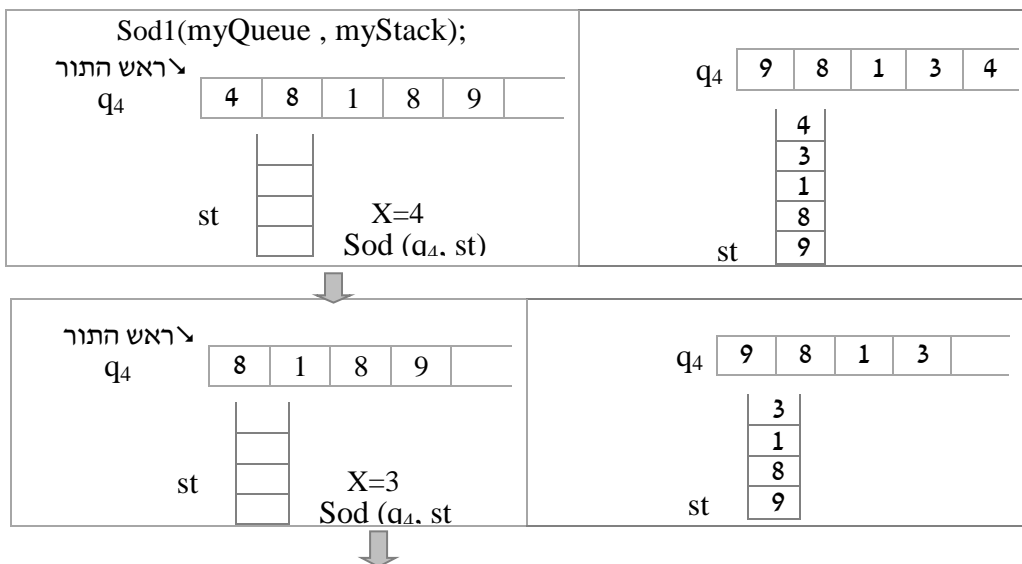
Stack<int> myStack = new Stack<int>();
Sod1(myQueue , myStack);
Sod2(myQueue , myStack);

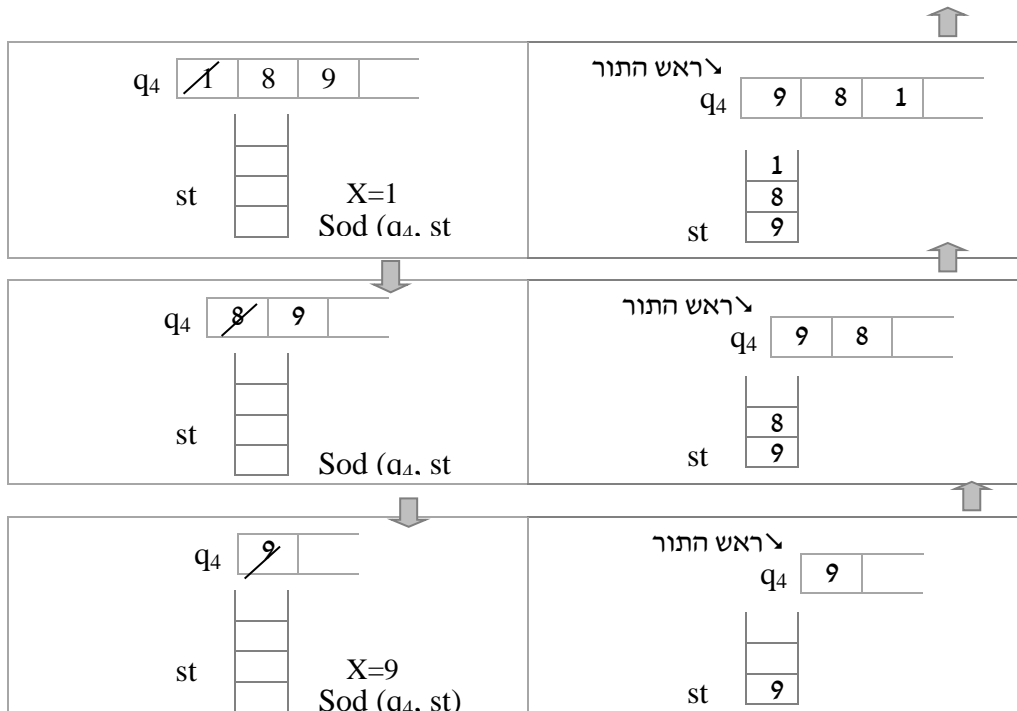
```

ונתון קטע התכנית

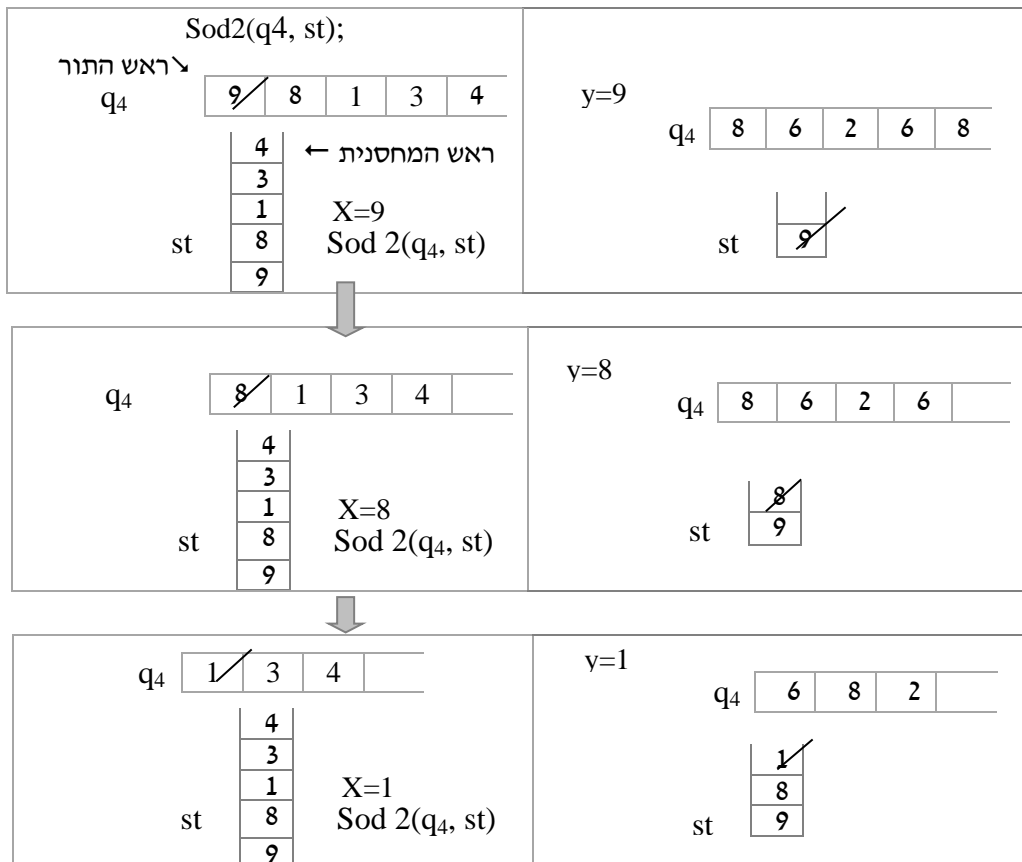
א. מעקב אחר הפעולה Sod1

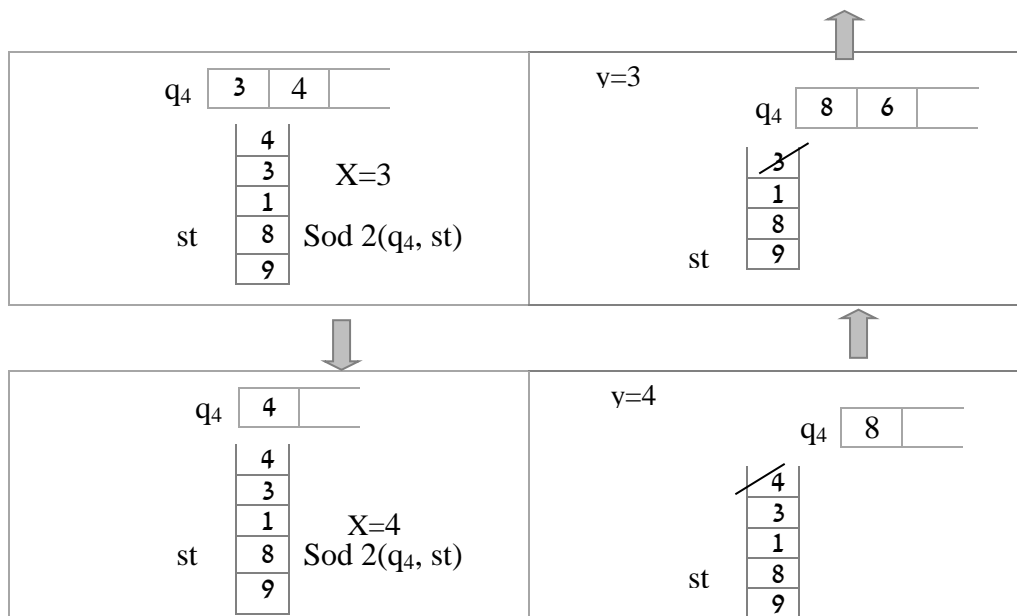
(1) התור myQueue והמחסנית myStack לאחר הזימון: `Sod1(myQueue , myStack)` מכילים:





מעקב אחר הפעולה Sod2





(2) התור myQueue והמחסנית myStack בסוף קטע התכנית מכילים

(i) myQueue מההתחלה לסוף, מימין לשמאל: 4,3,1,8,9

myStack מבפנים החוצה, מימין לשמאל: 4,3,1,8,9

(ii) myQueue מההתחלה לסוף, מימין לשמאל: 18,16,2,6,8

myStack : ריק

ב. בעבור תור כלשהו לא ריק myQueue מטיפוס Queue<int> ומחסנית myStack ריקה

מטיפוס Stack<int> הפעולה Sod1 מבצעת:

מחזירה במחסנית את אברי התור בסדר זהה כך שהאיבר שהיה בראש התור הוא בראש המחסנית, והופכת את סדר האיברים בתור.

ג. בעבור תור כלשהו לא ריק myQueue מטיפוס Queue<int> ומחסנית myStack ריקה

מטיפוס Stack<int> קטע התכנית מבצע:

הכפלה של ערכי כל האיברים בתור תוך שמירה על סדר האיברים.

כתרון סרק לני - מוצאים היילובים 2010

פתרון שאלה 13

נושא מרכזי: אוטומט סופי, רגולריות של שפה
סוג השאלה: ניתוח, בניית מודל

א. קביעת רגולריות של שפות ובניית מודל לשפות הרגולריות

| השפה | קביעת רגולריות + הסבר | אוטומט דטרמיניסטי סופי |
|---|---|------------------------|
| $L_1 = \{ a^n b^{n-1} \mid n \geq 1 \}$ | השפה L_1 אינה רגולרית. יש לזכור כמה a נקלטו כדי לבדוק את מספר ה- b . מספר ה- a תלוי במספר ה- b ימים כתלות אינסופית של מניה. | |
| $L_2 = \{ a^n b^k \mid n \geq 0, k \geq 0, \text{ החלוקה של } n \text{ ב-} 2 \text{ שונה משארית החלוקה של } k \text{ ב-} 2. \}$ | השפה L_2 רגולרית | |
| $L_3 = \{ a^k b^{2m} \mid k \geq 0, m \geq 0 \}$ | השפה L_3 רגולרית | |

הסבר הפתרון לאוטומט עבור השפה L_2 :

יש לשים לב שהשפה היא שפה של רצפים, בתחילתה ניתן לקלוט רק a ואז לעבור לקליטת b. המצב q_0 – רצף a-ים באורך זוגי (0 – מספר זוגי), עוד אין b – לכן רצף זוגי לכאורה, לכן אינו מצב מקבל.

המצב q_1 – רצף a-ים באורך אי-זוגי – אין תווי b, כלומר מספר b-ים זוגי – מצב מקבל

המצב q_2 – רצף a-ים היה באורך זוגי, רצף b-ים באורך אי זוגי – מצב מקבל

המצב q_3 – רצף a-ים באורך אי-זוגי, רצף b-ים באורך אי-זוגי

הסבר הפתרון לאוטומט עבור השפה L_3 :

השפה היא שפה של רצפים. המשמעות של רצף b באורך $2m$ – הוא שהרצף חייב להיות זוגי, כי עבור כל מספר שלם m, המספר $2m$ הוא זוגי.

ג. השפה L_5 היא:

נתונות השפות L_4 – L_5 מעל הא"ב {a, b}

$$L_4 = \{a^n b^{n-1} \mid n \text{ הוא אי-זוגי}, n \geq 1\}$$

$$L_5 = L_1 \cdot R(L_4)$$

$$L_5 = L_1 \cdot R(L_4) = \{a^n b^{n-1} b^{m-1} a^m \mid n \geq 1, m \geq 1\} = \{a^n b^{n+m-2} a^m \mid n \geq 1, m \geq 1\}$$

הסבר הפתרון לשפה L_5 :

השפה היא שרשור של שתי שפות שאינן תלויות זו בזו, לכן הדבר החשוב הוא לא לדרוש את אותו משתנה n בשתי השפות, אלא להשתמש במשתנה אחד המתאר את התלות בשפה הראשונה, ובמשתנה אחר לתיאור התלות בשפה השנייה.



פתרון שאלה 14

נושא מרכזי: שפות רגולריות- אוטומט סופי
סוג השאלה: בניית מודל

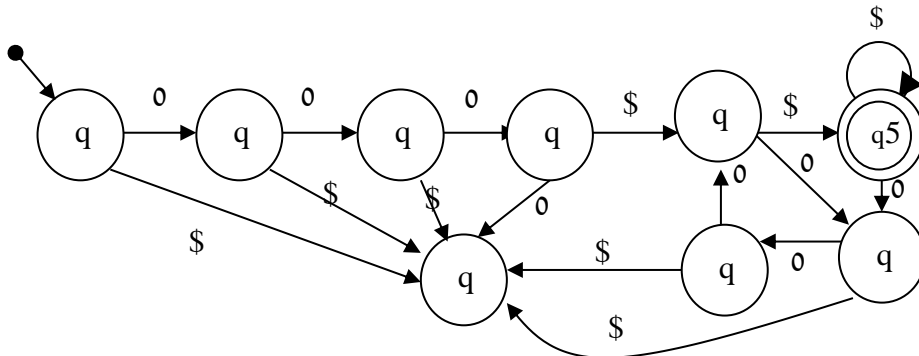
נתוני השאלה: נתונה השפה L מעל הא"ב {0, \$}:

$$L = \left\{ 0^3 \$ 0^{i_1} \$ 0^{i_2} \$ \dots 0^{i_k} \$ \mid \begin{array}{l} k \geq 1 \\ \text{לכל } m \text{ בין } 1 \text{ ל- } k: i_m \geq 0 \\ \text{ו- } i_m \text{ מתחלק ב- } 3 \text{ ללא שארית} \end{array} \right\}$$

א. המילה הקצרה ביותר בשפה L היא:

המילה תתקבל עבור $k=1$, זאת אומרת המילה היא מהצורה $0^3 \$ 0^i \$$, ולכן נבחר $i_1=0$, ונקבל את המילה $000 \$$, וזאת המילה הכי קצרה.

ב. אוטומט סופי דטרמיניסטי שמקבל את השפה L:





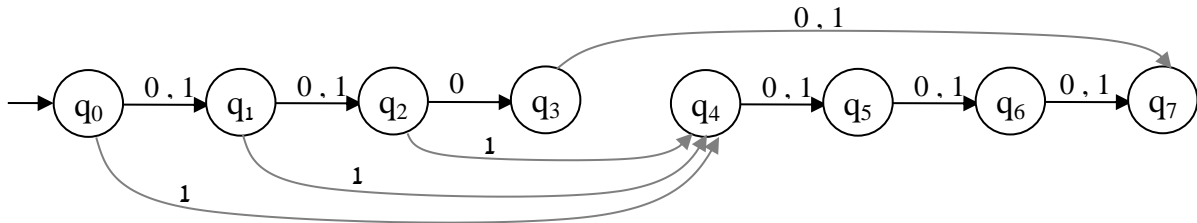
הסבר לתיאור האוטומט:

- אחרי ה-\$ הראשון, יש להקפיד בכל מקרה שיתקבלו רצפים של 0 באורך 3.
- רצף יכול להיות גם באורך 0, ולכן יש לאפשר קלטה חוזרת של \$.

פתרון שאלה 15

נושא מרכזי: פעולות על מילים ושפות
סוג השאלה: זיהוי מילים, זיהוי שפה רגולרית/לא רגולרית

נתוני השאלה: נתון אוטומט סופי לא דטרמיניסטי הבא:



(1) קביעה האם המילה מתקבלת ע"י האוטומט:

- המילה 001001 מתקבלת ע"י האוטומט. דוגמה למסלול: $q_0 \xrightarrow{0} q_1 \xrightarrow{0} q_2 \xrightarrow{1} q_4 \xrightarrow{0} q_5 \xrightarrow{0} q_6 \xrightarrow{1} q_7$
- המילה 01010 מתקבלת ע"י האוטומט. דוגמה למסלול: $q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_4 \xrightarrow{0} q_5 \xrightarrow{1} q_6 \xrightarrow{0} q_7$
- המילה 0101 לא מתקבלת ע"י האוטומט.

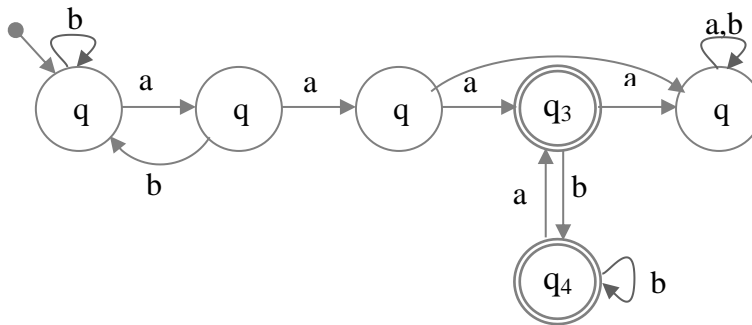
(2) האורך המינימלי של מילה שמתקבלת ע"י האוטומט הוא 4, למשל המילה 1111.

(3) האורך המקסימלי של מילה שמתקבלת ע"י האוטומט הוא 6, למשל המילה 001001.

(4) השפה המוגדרת על-ידי האוטומט היא $L = \{ \{0,1\}^n \cdot 1 \cdot \{0,1\}^3 \mid 0 \leq n \leq 2 \}$

ניתן להגדיר את השפה גם באופן מילולי: השפה מקבלת מילים מעל הא"ב $\{0,1\}$ בהן התו הרביעי מן הסוף הוא 1, ולפניו יכולים להיות עד שני תווים.

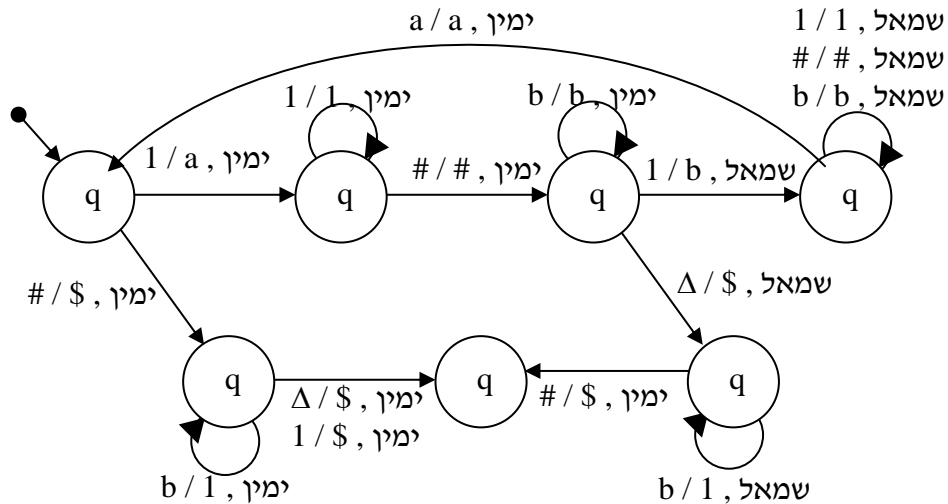
השלמת אוטומט



פתרון שאלה 16

נושא מרכזי: מכונת טיורינג
סוג השאלה: זיהוי מילים, זיהוי שפה רגולרית/לא רגולרית

א. השלמת סרטוט של מכונה



ההחלפות שבוצעו:

- א. ימין, $\#/\#$
- ב. ימין, $\#/\$$
- ג. שמאל, $\Delta/\$$

ב. תהליך החישוב של המכונה בעבור הקלט: $n=1, m=1$.

| | | | | | | |
|-------|---|----|---|----------|----------|-----|
| q_0 | 1 | # | 1 | Δ | Δ | ... |
| q_1 | a | # | 1 | Δ | Δ | ... |
| q_2 | a | # | 1 | Δ | Δ | ... |
| q_3 | a | # | b | Δ | Δ | ... |
| q_3 | a | # | b | Δ | Δ | ... |
| q_0 | a | # | b | Δ | Δ | ... |
| q_4 | a | \$ | b | Δ | Δ | ... |
| q_4 | a | \$ | 1 | Δ | Δ | ... |
| q_5 | a | \$ | 1 | \$ | Δ | ... |

הסבר של פעולת המכונה להעשרה: המכונה מסמנת בהתאמה, 1-ים מן המספר הראשון ב-a, ו-1-ים מהמספר השני ב-b. מסמנת תו אחד מן המספר הראשון, עוברת למספר השני ומסמנת תו אחד ממנו. אם במצב q_2 , אין עוד ספרה 1 לסימון, סימן שהמספר השני קטן יותר, אך בכל מקרה מספר התווים שהוחלפו בשני המספרים הוא באותו גודל – כגודל המספר הקטן. בשלב זה עובר האוטומט ל"סידור" הפלט. כלומר לרשום שני \$ משני הצדדים ולהפוך חזרה את תווי b לתווי 1. אם במצב q_0 אין עוד ספרה 1 לסימון, סימן שהמספר הראשון קטן יותר, אך עדיין מספר תווי b זהה למספר תווי a, שהוא כגודל המספר הקטן. גם כאן יש מעבר ל"סידור" הפלט.

