מדעי המחשב

פרק ראשון

שאלה 1

Java

```java
//---  פעולה המחזירה מערך חדש שבו כל האיברים  ---
//---                  שלא נמצאים במערך הנתון  ---
public static int[] Filter (int[]arr, int num)
{
    int n = countNums(arr, num);
    int[] a = new int[n];
    int j = 0;
    for(int i = 0; i < arr.length; i++)
    {
        if (arr[i] != num)
        {
            a[j] = arr[i];
            j++;
        }
    }
    return a;
}
```

```java
//---  פעולה הסופרת ומחזירה כמה מאיברי  ---
//---        המערך שונים מהמספר שהתקבל  ---
public static int countNums(int[]arr, int num)
{
    int count = 0;
    for (int i = 0; i < arr.length; i++)
        if (arr[i] != num)
            count++;
    return count;
}
```

C#

```csharp
//---  פעולה המחזירה מערך חדש שבו כל האיברים  ---
//---                  שלא נמצאים במערך הנתון  ---
public static int[] Filter (int[]arr, int num)
{
    int n = CountNums(arr, num);
    int[] a = new int[n];
    int j = 0;
    for(int i = 0; i < arr.Length; i++)
    {
        if (arr[i] != num)
        {
            a[j] = arr[i];
            j++;
        }
    }
    return a;
}
```

```csharp
//---  פעולה הסופרת ומחזירה כמה מאיברי  ---
//---        המערך שונים מהמספר שהתקבל  ---
public static int CountNums(int[]arr, int num)
{
    int count = 0;
    for (int i = 0; i < arr.Length; i++)
        if (arr[i] != num)
            count++;
    return count;
}
```

שאלה 2

Java

```java
public class ReportCard
{
    private String stuName;      //  שם התלמיד
    private Subject[] subArray; //  מערך הציונים

    //--- פעולה בונה ---
    public ReportCard(String name, int num)
    {
        this.stuName = name;
        this.subArray = new Subject[num];
    }


 //--- האם התלמיד מצטיין ---
 public boolean IsExcellent()
 {
     if (this.Average() < 85)
         return false;
     boolean grd100 = false;
     int grd;
     for (int i = 0; i < subArray.length; i++)
     {
         grd = subArray[i].getGrade();
         if (grd <= 54)
             return false;
         if (grd == 100)
             grd100 = true;
     }
     return grd100;       //  return grd100 == true;
 }



//--- פעולה המקבלת מערך של תעודות ומדפיסה את שמות התלמידים המצטיינים ---
public static void PrintExcellent(ReportCard[] array)
{
    for (int i = 0; i < array.length; i++)
    {
        if (array[i].IsExcellent())
            System.out.println(array[i].GetStuName());
    }
}
```

C#

```
class ReportCard
{
    private string stuName;     //  שם התלמיד
    private Subject[] subArray; //  מערך הציונים

    //--- פעולה בונה ---
    public ReportCard(string name, int num)
    {
        this.stuName = name;
        this.subArray = new Subject[num];
    }


//--- האם התלמיד מצטיין ---
public bool IsExcellent()
{
    if (this.Average() < 85)
        return false;
    bool grd100 = false;
    int grd;
    for (int i = 0; i < subArray.Length; i++)
    {
        grd = subArray[i].getGrade();
        if (grd <= 54)
            return false;
        if (grd == 100)
            grd100 = true;
    }
    return grd100;      //  return grd100 == true;
}


//--- פעולה המקבלת מערך של תעודות ומדפיסה את שמות התלמידים המצטיינים ---
public static void PrintExcellent(ReportCard[] array)
{
    for (int i = 0; i < array.Length; i++)
    {
        if (array[i].IsExcellent())
            Console.WriteLine(array[i].GetStuName());
    }
}
```

שאלה 3

Java

```java
//--- פעולה ההופכת מחרוזת להיות מחרוזת מיוחדת ---
public static MyString Special (MyString ms)
{
    MyString str = new MyString();
    char ch;
    int n;

    while( ! ms.isEmpty())
    {
        ch = ms.firstChar();          //  התו הראשון במחרוזת
        n = ms.countChar(ch);         //  מספר המופעים שלו
        ms.removeChar(ch);            //  מחיקתו ממחרוזת המקור

        //---   --- הוספת התו למחרוזת החדשה במספר המתאים של פעמים
        for (int i = 0; i < n; i++)
            str.appendChar(ch);
    }

    return str;
}
```

C#

```csharp
//--- פעולה ההופכת מחרוזת להיות מחרוזת מיוחדת ---
public static MyString Special (MyString ms)
{
    MyString str = new MyString();
    char ch;
    int n;

    while( ! ms.IsEmpty())
    {
        ch = ms.FirstChar();          //  התו הראשון במחרוזת
        n = ms.CountChar(ch);         //  מספר המופעים שלו
        ms.RemoveChar(ch);            //  מחיקתו ממחרוזת המקור

        //---   הוספת התו למחרוזת החדשה במספר המתאים של פעמים ---
        for (int i = 0; i < n; i++)
            str.AppendChar(ch);
    }

    return str;
}
```

פרק שני

שאלה 4

Java

```java
//---  פעולה המקבלת רשימה של מספרים שלמים באורך זוגי ---
//---  ומחזירה רשימה כפולה כך שבראשונה מחצית המספרים הגדולים ---
//--- ובשנייה שאר המספרים הקטנים ---
public static BiList GenerateBiList (Node<Integer>lst)
{
    int k = Size(lst)/2;
    BiList bl = new BiList();
    int max;

    //--- bl.lst1 -ל העברת מחצית הרשימה המכילה את המספרים הגדולים ---
    for(int i = 0; i < k; i++)
    {
        max = MaxInList(lst);
        lst = Delete(lst, max);
        bl.addNum(max, codeList: 1);
    }

    //--- bl.lst2 -ל העברת שאר האיברים ---
    while (lst != null)
    {
        bl.addNum(lst.getValue(), codeList: 2);
        lst = lst.getNext();
    }

    return bl;
}
```

```java
//--- פעולה המחזירה את אורך הרשימה ---
public static int Size(Node<Integer> lst)
{
    int count = 0;
    Node<Integer> pos = lst;
    while (pos != null)
    {
        count++;
        pos = pos.getNext();
    }
    return count;
}
```

```java
//--- פעולה המחזירה את המספר הגדול ביותר ברשימה ---
public static int MaxInList(Node<Integer> lst)
{
    int max = lst.getValue();
    Node<Integer> pos = lst.getNext();
    while (pos != null)
    {
        if (pos.getValue() > max)
            max = pos.getValue();
        pos = pos.getNext();
    }
    return max;
}
```

```java
//--- פעולה המוחקת את האיבר שהתקבל מהרשימה ---
//--- הפעולה נתונה ואין צורך לממש אותה במבחן ---
public static Node<Integer> Delete(Node<Integer>lst, int num)
{
    if (lst.getValue() == num)
        return lst.getNext();
    Node<Integer> prev = lst;
    Node<Integer> pos = lst.getNext();
    while (pos != null)
    {
        if (pos.getValue() == num)
        {
            prev.setNext(pos.getNext());
            return lst;
        }
        else
        {
            prev = pos;
            pos = pos.getNext();
        }
    }
    return lst;
}
```

C#

```
//--- פעולה המקבלת רשימה של מספרים שלמים באורך זוגי ---
//--- ומחזירה רשימה כפולה כך שבראשונה מחצית המספרים הגדולים ---
//--- ובשנייה שאר המספרים הקטנים ---
public static BiList GenerateBiList (Node<int>lst)
{
    int k = Size(lst)/2;
    BiList bl = new BiList();
    int max;

    //--- bl.lst1 -ל העברת מחצית הרשימה המכילה את המספרים הגדולים ---
    for(int i = 0; i < k; i++)
    {
        max = MaxInList(lst);
        lst = Delete(lst, max);
        bl.AddNum(max, 1);
    }

    //--- bl.lst2 -ל העברת שאר האיברים ---
    while (lst != null)
    {
        bl.AddNum(lst.GetValue(), 2);
        lst = lst.GetNext();
    }

    return bl;
}
```

```
//--- פעולה המחזירה את אורך הרשימה ---
public static int Size(Node<int> lst)
{
    int count = 0;
    Node<int> pos = lst;
    while (pos != null)
    {
        count++;
        pos = pos.GetNext();
    }
    return count;
}
```

```
//--- פעולה המחזירה את המספר הגדול ביותר ברשימה ---
public static int MaxInList(Node<int> lst)
{
    int max = lst.GetValue();
    Node<int> pos = lst.GetNext();
    while (pos != null)
    {
        if (pos.GetValue() > max)
            max = pos.GetValue();
        pos = pos.GetNext();
    }
    return max;
}
```

```
//--- פעולה המוחקת את האיבר שהתקבל מהרשימה ---
//--- הפעולה נתונה ואין צורך לממש אותה במבחן ---
public static Node<int> Delete(Node<int>lst, int num)
{
    if (lst.GetValue() == num)
        return lst.GetNext();
    Node<int> prev = lst;
    Node<int> pos = lst.GetNext();
    while (pos != null)
    {
        if (pos.GetValue() == num)
        {
            prev.SetNext(pos.GetNext());
            return lst;
        }
        else
        {
            prev = pos;
            pos = pos.GetNext();
        }
    }
    return lst;
}
```

שאלה 5

Java

```
//--- פעולה המבצעת סיבובים מעגליים על רשימה ---
//--- סיבוב מעגלי: העברת האיבר האחרון לתחילת הרשימה ---
public static Node<Integer> move(Node<Integer>lst, int n)
{
    for (int i = 0; i < n; i++)
    {
        int x = removeLast(lst);
        lst = new Node<Integer>(x, lst);
        show(lst);
    }
    return lst;
}

//--- פעולה המוחקת ומחזירה את האיבר האחרון ברשימה ---
//--- הנחה: האיבר האחרון אינו האיבר היחיד ברשימה ---
public static int removeLast(Node<Integer>lst)
{
    Node<Integer> prev = null;
    Node<Integer> pos = lst;
    int x;

    while(pos.hasNext())
    {
        prev = pos;
        pos = pos.getNext();
    }
    x = pos.getValue();
    prev.setNext(pos.getNext());
    return x;
}
```

יעילות הפעולה removeLast היא O(n) כאשר n מייצג את מספר האיברים ברשימה

הפעולה עוברת פעם אחת על הרשימה, ומבצעת פעולות (ממשק) שיעילות כל אחת מהן היא O(1)

יעילות הפעולה Move היא $O(n^2)$ כאשר n מייצג את מספר ההעברות שיש לבצע

(גם כן n כי ניתן לבצע העברות כמספר האיברים ברשימה, פחות, או יותר ממספר זה)

הפעולה מבצעת n פעמים את הפעולה removeLast ולכן: $f(n) = n * O(n)$ ⇦ $O(n^2)$

פתרון נוסף: יצירת רשימה מעגלית וקידום lst:

```
//--- פעולה המבצעת סיבובים מעגליים על רשימה ---
//--- סיבוב מעגלי: העברת האיבר האחרון לתחילת הרשימה ---
public static Node<Integer> move(Node<Integer>lst, int n)
{
    int k = size(lst) - n;  //  אורך שאר הרשימה
    Node<Integer>pos = getLast(lst);
    pos.setNext(lst);       //  יצירת רשימה מעגלית, קישור החוליה האחרונה לראשונה

    //--- קידום lst למיקום תחילת הרשימה החדש ---
    pos = lst;
    lst = lst.getNext();
    k --;
    while (k > 0)
    {
        pos = lst;
        lst = lst.getNext();
        k --;
    }

    pos.setNext(null);  //  ניתוק המעגל
    return lst;
}

//--- פעולה המחזירה הפנייה לחוליה האחרונה ברשימה ---
public static Node<Integer> getLast(Node<Integer>lst)
{
    Node<Integer>pos = lst;
    while (pos.hasNext())
        pos = pos.getNext();
    return pos;
}

//--- פעולה המחזירה את אורך הרשימה ---
public static int size(Node<Integer> lst)
{
    int count = 0;
    Node<Integer> pos = lst;
    while (pos != null)
    {
        count++;
        pos = pos.getNext();
    }
    return count;
}
```

יעילות הפעולה Move היא O(n) כאשר n מייצג את אורך הרשימה.
הפעולות size ו- getLast עוברות כל אחת פעם אחת על הרשימה, O(n) כל אחת.
לאחר מכן מתקדם lst על שאר size-n האיברים כדי להפנות לחוליה שעתה היא הראשונה ברשימה - O(n)
פונקציית זמן הריצה: f(n) = 2n ולכן היעילות ליניארית O(n)

C#

```
//--- פעולה המבצעת סיבובים מעגליים על רשימה ---
//--- סיבוב מעגלי: העברת האיבר האחרון לתחילת הרשימה ---
public static Node<int>Move(Node<int>lst, int n)
{
    for (int i = 0; i < n; i++)
    {
        int x = RemoveLast(lst);
        lst = new Node<int>(x, lst);
        Show(lst);
    }
    return lst;
}

//--- פעולה המוחקת ומחזירה את האיבר האחרון ברשימה ---
//--- הנחה: האיבר האחרון אינו האיבר היחיד ברשימה ---
public static int RemoveLast(Node<int>lst)
{
    Node<int> prev = null;
    Node<int> pos = lst;
    int x;

    while(pos.HasNext())
    {
        prev = pos;
        pos = pos.GetNext();
    }
    x = pos.GetValue();
    prev.SetNext(pos.GetNext());
    return x;
}
```

יעילות הפעולה removeLast היא O(n) כאשר n מייצג את מספר האיברים ברשימה

הפעולה עוברת פעם אחת על הרשימה, ומבצעת פעולות (ממשק) שיעילות כל אחת מהן היא O(1)

יעילות הפעולה Move היא $O(n^2)$ כאשר n מייצג את מספר ההעברות שיש לבצע

(גם כן n כי ניתן לבצע העברות כמספר האיברים ברשימה, פחות, או יותר ממספר זה)

הפעולה מבצעת n פעמים את הפעולה removeLast ולכן: $f(n) = n * O(n) \Leftarrow O(n^2)$

פתרון נוסף: יצירת רשימה מעגלית וקידום lst:

```
//--- פעולה המבצעת סיבובים מעגליים על רשימה ---
//--- סיבוב מעגלי: העברת האיבר האחרון לתחילת הרשימה ---
public static Node<int> Move(Node<int> lst, int n)
{
    int k = Size(lst) - n;          //  אורך שאר הרשימה
    Node<int> pos = GetLast(lst);
    pos.SetNext(lst);               //  יצירת רשימה מעגלית, קישור החוליה האחרונה לראשונה

    //--- למיקום תחילת הרשימה החדש lst קידום ---
    pos = lst;
    lst = lst.GetNext();
    k--;
    while (k > 0)
    {
        pos = lst;
        lst = lst.GetNext();
        k--;
    }

    pos.SetNext(null);  //  ניתוק המעגל
    return lst;
}

//--- פעולה המחזירה הפנייה לחוליה האחרונה ברשימה ---
public static Node<int> GetLast(Node<int> lst)
{
    Node<int> pos = lst;
    while (pos.HasNext())
        pos = pos.GetNext();
    return pos;
}

//--- פעולה המחזירה את אורך הרשימה ---
public static int Size(Node<int> lst)
{
    int count = 0;
    Node<int> pos = lst;
    while (pos != null)
    {
        count++;
        pos = pos.GetNext();
    }
    return count;
}
```

יעילות הפעולה Move היא O(n) כאשר n מייצג את אורך הרשימה.
הפעולות size ו- getLast עוברות כל אחת פעם אחת על הרשימה, O(n) כל אחת.
לאחר מכן מתקדם lst על שאר size-n האיברים כדי להפנות לחוליה שעתה היא הראשונה ברשימה - O(n)
פונקציית זמן הריצה: f(n) = 2n ולכן היעילות ליניארית O(n)

שאלה 6

	0	1	2	3	4	length
a	5	4	15	12	2	5

java: sod1 (a, 8, a.length-1)

c#: Sod1 (a, 8, a.Length-1)

x	i	i== -1	arr[i]	arr[i]==x	משפט זימון	ערך מוחזר
8	4	F	2	F	sod(a, 8, 3)	F
8	3	F	12	F	sod(a, 8, 2)	
8	2	F	15	F	sod(a, 8, 1)	
8	1	F	4	F	sod(a, 8, 0)	
8	0	F	5	F	sod(a, 8, -1)	
8	-1	T				F
א. (1) ערך מוחזר: false						
א. (2) הפעולה מחזירה אמת אם x נמצא במערך ושקר אחרת						
א. (3) סיבוכיות הפעולה O(n) כאשר n מייצג את מספר האיברים במערך (n == a.length)						

java: sod2 (a, 16, a.length-1)

c#: Sod2 (a, 16, a.Length-1)

x	i	i== 0	arr[i]	x - arr[i]	sod1	משפט זימון	ערך מוחזר
16	4	F	2	14	F	sod2(a, 16, 3)	
16	3	F	12	4	T		True
א. (1) ערך מוחזר: true							
א. (2) הפעולה מחזירה אמת אם קיימים במערך שני מספרים שסכומם x ושקר אחרת							
א. (3) סיבוכיות הפעולה $O(n^2)$ כאשר n מייצג את מספר האיברים במערך (n == a.length) הפעולה מפעילה לכל היותר n פעמים את הפעולה sod1 שיעילותה O(n) ולכן: $f(n) = n * O(n)$ ⇨ $O(n^2)$							

שאלה 7

Java

```
//--- פעולה המחזירה אמת אם שני התורים בעלי אותו אורך ---
//--- ואותם איברים, גם לאחר העברת איבר מהסוף להתחלה ---
public static boolean IsSame(Queue<Integer> q1, Queue<Integer> q2)
{
    int k = size(q2);
    while(k > 0)
    {
        if (IsIdentical(q1, q2))
            return true;
        rotate(q2);
        k--;
    }
    return false;
}

//--- פעולה המחזירה אמת אם שני התורים זהים ושקר אחרת ---
public static boolean IsIdentical(Queue<Integer>q1, Queue<Integer>q2)
{
    Queue<Integer> qTemp1 = new Queue<Integer>();
    Queue<Integer> qTemp2 = new Queue<Integer>();
    boolean same = true;

    //--- העברת האיברים הזהים לתורי עזר ---
    while (! q1.isEmpty() && !q2.isEmpty() && same)
    {
        if (q1.head() != q2.head())
            same = false;
        qTemp1.insert(q1.remove());
        qTemp2.insert(q2.remove());
    }

    //--- אם אחד התורים לא ריק - האורכים שונים ---
    if (!q1.isEmpty() || !q2.isEmpty())
        same = false;

    //--- העברת שאר האיברים לתורי העזר ---
    while (!q1.isEmpty())
        qTemp1.insert(q1.remove());
    while (!q2.isEmpty())
        qTemp2.insert(q2.remove());

    //--- החזרת האיברים לתורים המקוריים ---
    while (!qTemp1.isEmpty())
        q1.insert(qTemp1.remove());
    while (!qTemp2.isEmpty())
        q2.insert(qTemp2.remove());

    return same;
}

//--- פעולה המעבירה את האיבר שבראש התור לסופו ---
public static void rotate(Queue<Integer>que)
{
    if (!que.isEmpty())
        que.insert(que.remove());
}
```

לולאה I: העברת כל האיברים הזהים.

הלולאה תסתיים כאשר:
* נמצא איבר שונה
* אחד התורים קצר מהאחר

במקרה כזה תתבצע לולאה II+III

אם שני התורים בעלי אותו אורך וכל איבריהם זהים, לא תתבצע לולאה II+III

C#

```
//--- פעולה המחזירה אמת אם שני התורים בעלי אותו אורך ---
//--- ואותם איברים, גם לאחר העברת איבר מהסוף להתחלה ---
public static bool IsSame(Queue<int> q1, Queue<int> q2)
{
    int k = Size(q2);
    while(k > 0)
    {
        if (IsIdentical(q1, q2))
            return true;
        Rotate(q2);
        k--;
    }
    return false;
}

//--- פעולה המחזירה אמת אם שני התורים זהים ושקר אחרת ---
public static bool IsIdentical(Queue<int>q1, Queue<int>q2)
{
    Queue<int> qTemp1 = new Queue<int>();
    Queue<int> qTemp2 = new Queue<int>();
    bool isSame = true;

    //--- העברת האיברים הזהים לתורי עזר ---
    while (! q1.IsEmpty() && !q2.IsEmpty() && isSame)
    {
        if (q1.Head() != q2.Head())
            isSame = false;
        qTemp1.Insert(q1.Remove());
        qTemp2.Insert(q2.Remove());
    }
    //--- אם אחד התורים לא ריק - האורכים שונים ---
    if (!q1.IsEmpty() || !q2.IsEmpty())
        isSame = false;

    //--- העברת שאר האיברים לתורי העזר ---
    while (!q1.IsEmpty())
        qTemp1.Insert(q1.Remove());
    while (!q2.IsEmpty())
        qTemp2.Insert(q2.Remove());

    //--- החזרת האיברים לתורים המקוריים ---
    while (!qTemp1.IsEmpty())
        q1.Insert(qTemp1.Remove());
    while (!qTemp2.IsEmpty())
        q2.Insert(qTemp2.Remove());

    return isSame;
}


//--- פעולה המעבירה את האיבר שבראש התור לסופו ---
public static void Rotate (Queue<int>que)
{
    if (!que.IsEmpty())
        que.Insert(que.Remove());
}
```

לולאה I: העברת כל האיברים הזהים.

הלולאה תסתיים כאשר:
* נמצא איבר שונה
* אחד התורים קצר מהאחר

במקרה כזה תתבצע לולאה II+III

אם שני התורים בעלי אותו אורך וכל איבריהם זהים, לא תתבצע לולאה II+III

פרק שלישי

לפניך שאלות מ- 4 מסלולים שונים: **מערכות מחשב ואסמבלי** (שאלות 8-9), **מבוא לחקר ביצועים** (שאלות 10-11), **מודלים חישוביים** (שאלות 12-13), **תכנות מונחה עצמים בשפת Java** (שאלות 14-15), **תכנות מונחה עצמים בשפת C#** (שאלות 16-17).

מערכות מחשב ואסמבלי

פתרון לפרק זה נכתב ע"י: רונית (מרציאנו) גל-אור

שאלה 8

סעיף א

(1)

AX		BX		CX		DX	
AH	AL	BH	BL	CH	CL	DH	DL
	06h	00h	5Ch				8Dh
		00h	5Bh		3Fh		
	05h				40h		
		00h	5Ah		0ABh		
	04h				0ACh		
		00h	59h		71h		
					72h		
	03h	00h	58h		0C5h		
					0C6h		
	02h	00h	57h		0E2h		
					0E3h		
	01h	00h	56h		59h		
					5Ah		
	00h	00h	55h				
		00h	56h				8Eh

address	56h	57h	58h	59h	5Ah	5Bh	5Ch
value	~~59h~~	0E2h	~~0C5h~~	~~71h~~	~~0ABh~~	~~3Fh~~	~~8Dh~~
	8Eh	5Ah	0E3h	0C6h	72h	0ACh	40h

(2) הקטע מבצע הזזה ציקלית 1 ימינה של כל הערכים, תוך כדי הוספת 1 לכל הערכים.

סעיף ב

```
REZ   DW ?

POP AX
POP BX
CMP AX,0
JG GO1
NEG AX
GO1:
CMP BX,0
JG  GO2
NEG BX
GO2:
CMP AX,BX
JG  GO3
MOV REZ,BX
JMP SOF
GO3:
MOV REZ, AX
SOF:
```

שאלה 9

```
ARR  DB 10 DUP(0)
 VAL  DB ?
 IND  DB ?

 MOV BX,08H
  CMP IND,9  ; אם האינדקס הוא אחרון יש לטפל בו בנפרד
  JNE AGAIN
  MOV AL,VAL
  MOV [BX+1],AL
  JMP SOF

AGAIN:
 MOV AL,[BX]
 MOV [BX+1],AL
 DEC BX
 CMP BL,IND
 JGE  AGAIN
 INC BX
 MOV AL,VAL
 MOV [BX],AL
SOF:
```

פתרון נוסף לשאלה 9
נכתב ע"י ארז אביב

```
MOV AH,VAL

MOV BL,IND
MOV BH,BH

AGAIN:
MOV AL,[BX]
MOV [BX],AH

MOV AH,AL
INC BX
CMP BX,09
JLE AGAIN
```

מבוא לחקר ביצועים

פתרון לפרק זה נכתב ע"י: **מרים קסמן**

שאלה 10

א. (1) X11=100, X12=100, X22=50, X23=250, X34=80

(2) u1=0, u2=5, u3=7, v1=10, v2=7, v3=-1, v4=0

הפתרון הנתון בטבלה איננו אופטימלי כיוון שע"פ רוטציית המחירים וחישוב הנוסחא c32 שלילי.
ע"מ שהפתרון יהיה אופטימלי יש להגדיל את C32 ב1 (מ13 ל14).

ב. (1) Ga={2,4,6,7}{1,5,3}

Gb = {3,4,2} {1,5}

Gc לא מתחלק

(2) ניתן להסיר את הקשתות (3,7)(4,5) ע"מ שהגרף ייחשב מתחלק, החלוקה תהיה:
{1,3,7}{2,6,4,5}

שאלה 11

א. (1) גרף

(2) רק"ח {a,b,c,d} 2 רכיבים נוספים המורכבים מקודקוד אחד כ"א {f}{e}

(3) קשת אחת <f,a> (מכוונת)

ב. (1) משקל 11 {a,b,c,d}

(2) אם השורש למסלול יכול להיות c, אין משמעות למשקל הקשת <a,c>.

מודלים חישוביים

פתרון לפרק זה נכתב ע"י: **רחל לודמר**

שאלה 12

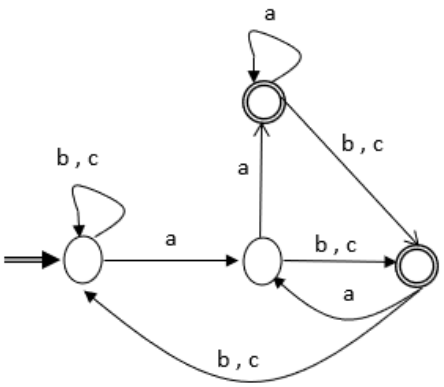
א.1. $cbccbcac \in L$

א.2. $bbc \notin L$

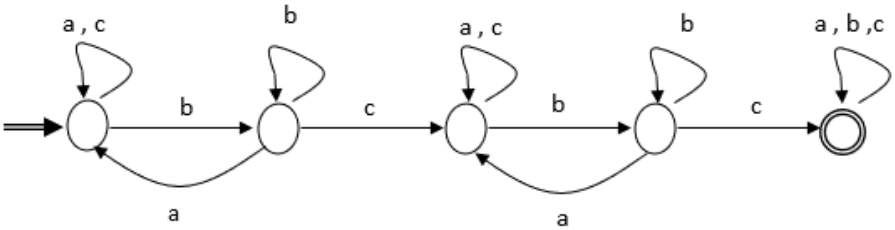
ב.

נבנה אוטומטים סופיים דטרמיניסטיים לשפות הבאות:

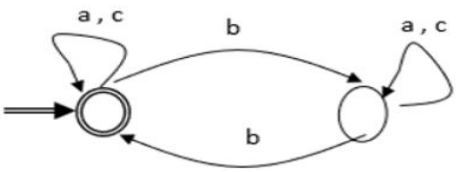
L_1 = {האות שלפני האחרונה היא a}



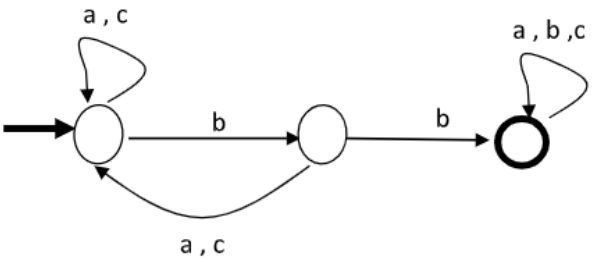
L_2 = {מכיל לפחות פעמיים bc}



L_3 = {מספר b זוגי}



L_4 = {מכיל bb}



לשפות: L_1, L_2, L_3, L_4 בנינו אוטומט סופי דטרמיניסטי לכן הן רגולריות.
מתכונות סגירות של שפות רגולריות, משלים של שפה רגולרית הוא רגולרי, וחיתוך של שפות רגולריות הוא רגולרי.
לכן השפה L רגולרית

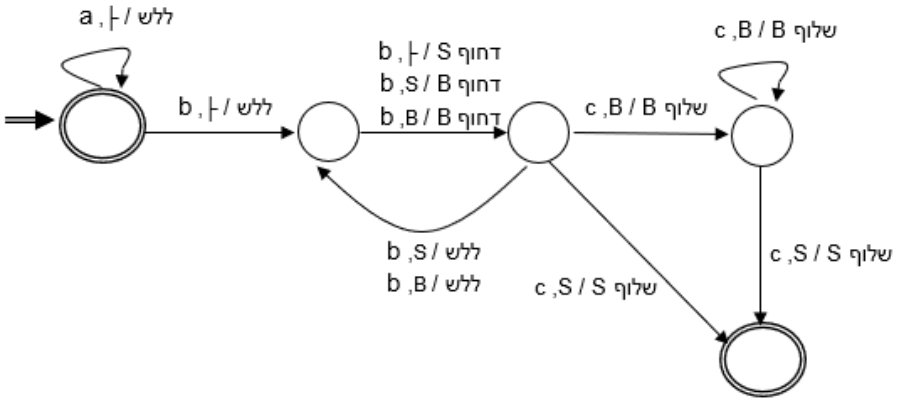
מכאן $L = L_1 \cap L_2 \cap L_3 \cap \overline{L_4}$ היא רגולרית.

שאלה 13

$$L_1 = \{a^n b^{2k} c^k \mid n,k \geq 0\}$$
$$L_2 = \{a^n b^m c^k \mid n,m,k > 0,\ n\%2 = m\%2 = k\%2\}$$

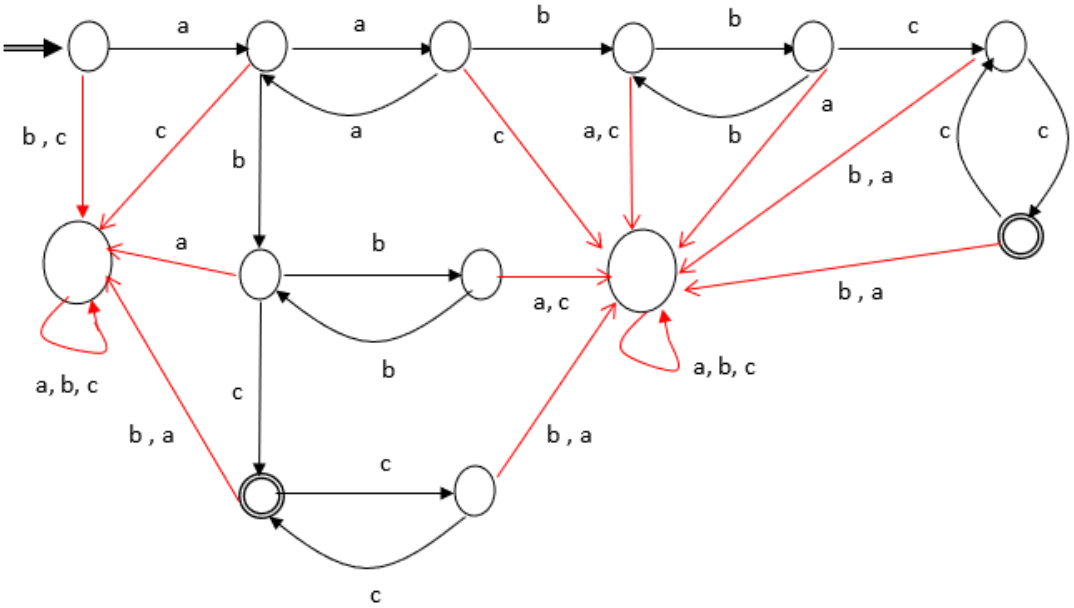
א.

השפה L_1 היא לא רגולרית, קיימת תלות של מניה בין רצפי b לבין רצפי c. לכן נבנה אוטומט מחסנית.



ב.

השפה L_2 היא רגולרית. כל הרצפים הם זוגיים או אי זוגיים. לכן נבנה אס"ד מתאים.

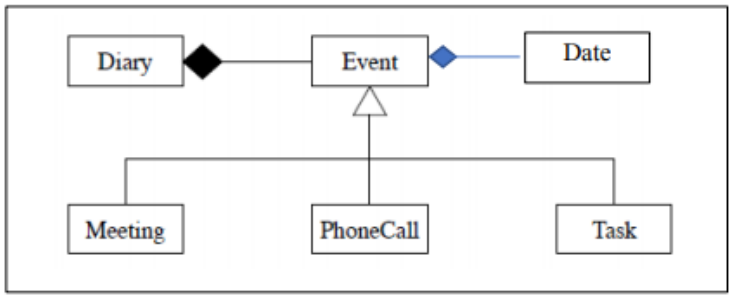


תכנות מונחה עצמים בשפת Java

פתרון לפרק זה נכתב ע"י: **אביטל Evi גרינוואלד**

שאלה 14

סעיף א: היררכית המחלקות



כותרת המחלקה	`public class Event`
תאריך האירוע	`protected Date date;`
שעת התחלת האירוע	`protected int hour;`

סעיף ב: פעולת המחלקה Diary

תנאי קדם: עצם מטיפוס Date שמייצג תאריך שיחת טלפון

תנאי בתר: מערך בגודל 100 מטיפוס PhoneCall אשר מכיל את כל שיחות הטלפון שהתקיימו בתאריך date .

הנחות: יש בדיוק 100 שיחות טלפון שהתקיימו ביום date

אין null במערך האירועים.

```
public PhoneCall[] allCalls(Date date)
{
    PhoneCall[] calls = new PhoneCall[NUM_CALLS_DATE];
    int index = 0;  // index to new array
    PhoneCall phonecall;
    for (int event = 0; event < this.arr.length; event++)
    {
        if (this.arr[event] instanceof PhoneCall)
        {
            phonecall = (PhoneCall)this.arr[event];
            if (phonecall.getDate().same(date))
            {
                calls[index] = new PhoneCall(phonecall);
                index++;
            }
        }
    }// end for
    return calls;
}
```

<u>סעיף ג</u>

תנאי קדם: שם

תנאי בתר: אמת אם משתתף בפגישה או שמתקשים אליו

מחלקה Event	`public boolean match(String name)`
	`{`
	`  return false;`
	`}`

מחלקה Meeting	`public boolean match(String name)`
	`{`
	`  for (int p=0; p < this.arrNames.length; p++)`
	`  {`
	`  if (this.arrNames[p].equals(name))`
	`  return true;`
	`  }`
	`  return false;`
	`}`

מחלקה PhoneCall	`public boolean match(String name)`
	`{`
	`  return this.name.equals(name);`
	`}`

שאלה 15

השאלה עוסקת ב-
ירושה, דריסת פעולות ופולימורפיזם.
מטרת השאלה: זיהוי שגיאות של קטעי קוד בפעולה ראשית ופלט למקרה שתקין.

מספר	קוד הקטע	תקין / לא תקין	פלט אם תקין
1	`A a = new A();` `a.foo(2);`	**תקין**. למחלקה A יש פעולה כזו עם חתימה כזו	ctor A A foo int 2
2	`A a = new A()` `((B)a).foo(3)`	**לא תקין. שגיאת ריצה** העצם נוצר מטיפוס A. בשורה שנייה מנסים להמיר אותו לעצם מטיפוס B. בזמן ריצה התוכנית תגלה זאת. המרות מתבצעות בזמן ריצה.	
3	A a = new A(); B b = a; b.foo(2);	**לא תקין. שגיאת תחביר** העצם a הוא טיפוס A. בשורה שנייה מנסים להציב אותו בעצם מטיפוס B. אין התאמה בטיפוסים. טיפוס A אינו מכיר את טיפוס B.	
4	A x = new B(); x.foo(2);	**תקין** A a = new A(); B יורש מ A ולכן ניתן ליצור עצם מטיפוס תת המחלקה ולבצע המרה בלתי מפורשת למחלקת העל. במחלקה A קיימת הפעולה עם החתימה המתאימה הסבר לפלט: בפעולה הבונה של B מופעל הבנאי הריק של מחלקת העל בשורה ראשונה גם אם לא כתוב מפורשות. ולכן תופיעה שורה ראשונה בפלט לאחריו הפלט שמופיע בפעולה הבונה של B. מאחר ולפעולה foo(int x) יש מימוש שונה במחלקה B ומחר והעצם נוצר ממחלקה B, תזומן הפעולה foo שכתובה במחלקה ממנה נוצר העצם והפלט יהיה השורה השלישית.	ctor A ctor B B foo int 2
5	A x = new B(); x.bar();	**לא תקין. שגיאת תחביר** בשורה ראשונה נוצר עצם מטיפוס B אבל קיימת המרה כלפי מעלה למחלקה A. בשורה שנייה מנסים להפעיל על העצם פעולה שלא קיימת במחלקה A	
6	A a = new A(); a.bar(3);	**תקין** במחלקה A יש פעולה כזו עם חתימה כזו	ctor A A bar 3 A foo int 3
7	A a = new A(); a.bar();	**לא תקין. שגיאת תחביר** למחלקה A אין פעולה כזו עם חתימה כזו.	

מספר	קוד הקטע	תקין / לא תקין	פלט אם תקין
8	B b = new B(); b.bar();	**תקין**	ctor A ctor B B bar B foo int 2
9	B b = new B(); b.bar(3);	**תקין** שני שורות ראשונות קשורות לפעולה הבונה למחלקה B אין פעולה דרוסה של פעולה bar עם פרמטר שלם ולכן מפעילה את הפעולה שקבלה בירושה ממחלקה A. בפעולה bar יש זימון לפעולה foo(3) , מאחר והעצם נוצר מטיפוס B, תופעל הפעולה foo שנכתבה במחלקה B.	ctor A ctor B A bar 3 B foo int 3

10	B b = new B();
	b.foo(2);
	b.foo(2.0);
		תקין	ctor A
			ctor B
			B foo int 2
			A foo double 2.0

11	A a = new A();
	a.foo(2);
	a.foo(2.0);
		תקין	ctor A
			A foo int 2
			A foo double 2.0

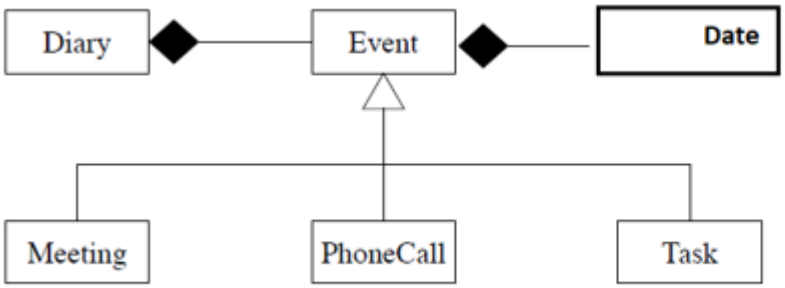
12	B b = new A(); b.another(2);	**לא תקין. שגיאת תחביר** יוצרת עצם מטיפוס A ומנסים להציב אותו בטיפוס שאינו מכיר. אין התאמה בטיפוסים.	
13	A a = new A(); a.another(2);	**לא תקין. שגיאת תחביר** למחלקה A אין פעולה כזו.	
14	B x = new B(); x.another(2);	**תקין** בפעולה another מפעילים את super.foo(x) ולכן הוראת הדפסת שורה 4 מאחר ובמחלקה B אין פעולה foo שמקבלת מספר ממשי, היא מפעילה את הפעולה שכתובה במחלקה העל. גורם להדפסת שורה 4.	ctor A ctor B B another 2 A foo int 2 A foo double 4.0
15	A x = new B(); x.another(2);	**לא תקין. שגיאת תחביר** בשורה ראשונה יש המרה למחלקה A. למחלקה A אין פעולה another ולכן לא מכירה אותה.	

תכנות מונחה עצמים בשפת #C

פתרון לפרק זה נכתב ע"י: **דיתה אוהב ציון**

שאלה 16

א.




```
public  class Event
{
      private Date date;
      private int hour;
```

ב.

```
//הנחה: יש בדיוק 100 שיחות טלפון ביום
 public PhoneCall[]  AllCalls(Date date)
 {
      PhoneCall[] arr = new PhoneCall[100];
      int index = 0;
      foreach( Event e in arr)
      {
         if(e is PhoneCall)
         {
            if(e.GetDate().Same(date) )
            {
               string p = ((PhoneCall)e).GetPhone();
               string n = ((PhoneCall)e).GetName();
               arr[index++] = new PhoneCall(date, e.GetHour(),p , n);

            }
         }
      }
      return arr;
 }
```

ג.

במחלקה Event

```
public virtual bool Match(string name) {   return false;   }
```

במחלקה Meeting

```
public override bool Match(string name)
{
        for (int i = 0; i < arrNames.Length; i++)
        {
           if (arrNames[i] == name) return true;

        }
        return false;
}
```

במחלקה PhoneCall

```
public override bool Match(string name)
{
        return this.name == name;
}
```

שאלה 17

1	`A a = new A();` `a.Foo(2);`	ctor A A Foo int2

2	`A a = new A();`
	
	`((B)a).Foo(3);`
		ctor A שורה 1 תקין
		
		לא תקין. שגיאת זמן ריצה בהמרה. אי אפשר להמיר לעצם לא מוכר. הפניה מטיפוס A המחזיקה עצם מטיפוס A לא מכירה את B.

3	`A a = new A();`
	`B b = a;`
	`b.Foo(2);`
		שגיאת קומפילציה בשורה 2. בן לא יכול להחזיק אבא.

4	`A x = new B();`
	
	`x.Foo(2);`
		ctor A
		ctor B
		B Foo int2

5	`A x = new B();`
	
	`x.Bar();`
		שגיאת קומפילציה בשורה 2. הפניה מטיפוס A מפעילה את פעולה Bar שלה , וחסר פרמטר הנדרש בחתימת הפעולה.

6	`A a = new A();`
	
	`a.Bar(3);`
		ctor A
		A Bar3
		A Foo int3

7	`A a = new A();` `a.Bar();`	שגיאת קומפילציה. חסר פרמטר לפעולה Bar
8	`B b = new B();` `b.Bar();`	ctor A ctor B B Bar B Foo int2
9	`B b = new B();` `b.Bar(3);`	ctor A ctor B A Bar3 B Foo int3

10	`B b = new B();`
	`b.Foo(2);`
	`b.Foo(2.0);`
		ctor A
		ctor B
		B Foo int2
		A Foo double2

11	`A a = new A();`
	`a.Foo(2);`
	`a.Foo(2.0);`
		ctor A
		A Foo int2
		A Foo double2

12	`B b = new A();` `b.Another(2);`	שגיאת קומפילציה בשורה הראשונה. הפניה מטיפוס הבן לא יכולה להחזיק עצם מטיפוס האב.
13	`A a = new A();` `a.Another(2);`	שגיאת קומפילציה. הפעולה אינה מוגדרת במחלקה A
14	`B x = new B();` `x.Another(2);`	ctor A ctor B B Another2 A Foo int2 A Foo double4
15	`A x = new B();` `x.Another(2);`	שגיאת קומפילציה. הפניה מטיפוס A אינה מכירה פעולות מהמחלקות היורשות. (חסרה המרה מפורשת - `((B)x).Another(2);`

בהצלחה!