

שאלה 1 (5 נק')::

לפניכם מימוש של המחלקה Point, כשחלק מהמימוש הושמט.

```
public class Point{
    protected int i,j;

    public Point(int i, int j) {
        this.i = i;
        this.j = j;
    }

    public int hashCode() {
        return i+j;
    }

    public boolean equals(Object obj) {
        Point other = (Point) obj;
        return this.i == other.i    &&    this.j == other.j;
    }

    // the rest of the code
}
```

נרצה לייצר מצב שבו לא יהיה קיים שום אובייקט שמקיים יחס is-a עם Point שאינו מטיפוס Point. בתשובתכם, עליכם להתייחס לכל האפשרויות שיכולות להופיע בקטע הקוד שהושמט.

טענה 1: אם נגדיר את Point להיות final, נשיג את מטרתנו בוודאות.

טענה 2: אם נגדיר את כל הבנאים (במידה ויש יותר מאחד) של Point להיות פרטיים, נשיג את מטרתנו בוודאות.

טענה 3: אם נשנה את הניראות של i,j לניראות פרטית, נשיג את מטרתנו בוודאות.

בחר\י בתשובה הטובה ביותר:

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

שאלה 2 (5 נק')::

שאלות 2-4 מתייחסות למחלקה FreqList שמימושה מופיע בהמשך. המחלקה FreqList מהווה סוג של רשימה גנרית, אשר בנוסף לשמירת כל האיברים שהוכנסו לרשימה, היא שומרת גם את מספר המופעים שלהם ברשימה, ויכולה להחזיר את מספר המופעים שלהם מבלי לעבור על תוכן הרשימה.

זוהי דוגמה לשימוש המצופה ב FreqList. קוד זה עושה שימוש ב Point אשר מומשה בשאלה הקודמת:

```
public static void main(String[] args) {
    FreqList<Point> fList =new FreqList<>();
    fList.addElement(new Point(1,2));
    fList.addElement(new Point(4,7));
    fList.addElement(new Point(1,2));
    System.out.println(fList.getFirstIndexofElement(new Point(4,7))); //1
    System.out.println(fList.getCountforElement(new Point(1,2))); //2
    System.out.println(
        fList.getIsPresent(new Point(3,3))); //False
}
```

להלן המימוש של FreqPoint:

```

/* @imp_inv: freqMap.size() == number of unique elements in innerList */
public class FreqList<T>{
    private List<T> innerList = new ArrayList<>();
    private Map<T, Integer> freqMap = new HashMap<>();

    public FreqList() {}

    public void addElement(T elem) {
        innerList.add(elem); //add to the end of the list
        freqMap.put(elem, getCountForElement(elem) +1);
    }

    public boolean getIsPresent(T element) {
        return freqMap.containsKey(element);
    }

    public int getCountForElement(T element) {
        /* getOrDefault - returns value for key. If key doesn't exist
        The specified default value is returned, 0 in this case. */
        return freqMap.getOrDefault(element,0);
    }

    public int getFirstIndexOfElement(T element) {
        for (int i = 0; i < innerList.size(); i++) {
            if (innerList.get(i) == element) {
                return i;
            }
        }
        return -1;
    }
}

```

לפניכם מספר טענות הנוגעות להדפסות ב main:

טענה 1: הקריאה ל `getFirstIndexOfElement` תייצר את הפלט הרצוי.

טענה 2: הקריאה ל `getCountForElement` תייצר את הפלט הרצוי.

טענה 3: הקריאה ל `getIsPresent` תייצר את הפלט הרצוי.

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

שאלה 3 (8 נקודות) – שאלה זו מצריכה נימוק:

נוסיף מתודה חדשה למחלקה:

```
/* @pre: getIsPresent() == true */
/* @pre: 0 < k <= getCountForElement(elem) */
/* @post: @prev(getCountForElement(elem)) - k ==
          getCountForElement(elem) */

public void removeKInstancesOfElem(T elem, int k) {
    freqMap.put(elem, getCountForElement(elem) - k);
    while (k > 0) {
        innerList.remove(elem); //remove first occurrence of elem
        k--;
    }
}
```

לפניכם מספר טענות:

טענה 1: המימוש של הפונקציה מקיים את תנאי ה post שלה.

טענה 2: אם נסיר את תנאי הקדם הראשון (`getIsPresent() == true`), המשתמשת עשויה לקבל שגיאת זמן ריצה בעת שימוש בפונק' על פי החוזה המעודכן.

טענה 3: הפונקציה מפרה את השמורה (invariant) של `FreqList`.

שאלה זו מצריכה נימוק! נמקו בקצרה את הנכונות או את אי הנכונות של כל טענה.

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

שאלה 4 (6 נקודות) – שאלה זו מצריכה נימוק.

הסטודנטית שי רוצה לממש את הפונקציה `getFreqIterator` אשר מחזירה איטרטור אשר עובר על כל מספרי המופעים של האלמנטים ב `FreqList` מהגדול לקטן.

דוגמת שימוש באיטרטור:

```
FreqList<Integer> fListStr = new FreqList<>();
fListStr.addElement(10);
fListStr.addElement(13);
fListStr.addElement(15);
fListStr.addElement(10);
Iterator<Integer> it = fListStr.getFreqIterator();
while(it.hasNext()) {
    System.out.println(it.next());
    // 2 1 1
}
```

להלן המימוש של שי. שימו לב שהמימוש צריך להיות תקין ל `FreqList` מטיפוסים שונים.

```
public Iterator<T> getFreqIterator(){    /**
    List<Integer> sortedValues = new ArrayList<>();
    sortedValues.addAll(freqMap.values()); /**
    sortedValues.sort((x,y) -> - Integer.compare(x,y)); /**
    return sortedValues.iterator();
}
```

האם המימוש של שי תקין?

טענה 1: נדרש לבצע תיקון בשורה *

טענה 2: נדרש לבצע תיקון בשורה **

טענה 3: נדרש לבצע תיקון בשורה ***

שאלה זו מצריכה נימוק! נמקו בקצרה את תשובתכם לגבי נכונותה או אי נכונותה של כל טענה.

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

שאלה 5 (6 נקודות) – שאלה זו מצריכה נימוק

הקוד הבא מתייחס ל A,B,C שהם מחלקות או ממשקים. בנוסף, נתון מימוש חלקי של הפונקציה main המתאר קריאה ל func. השאלה מתייחסת גם למימוש של A,B,C וגם להשלמת המימוש של main. כל התייחסות לשגיאות זמן ריצה מתייחסת לשגיאות של ClassCastException.

```
public static void func(A a1, A a2){
    B b1 = (B)a1; //line 1
    B b2 = (B)a2; //line 2
    C c1 = (C)a1; //line 3
    C c2 = (B)a1; //line 4
}

public static void main(String[] args){
    A a1 = /* missing code */ ;
    A a2 = /* missing code */ ;
    func(a1, a2);
}
```

לפנים 3 טענות על המימושים האפשריים של A,B,C:

- טענה 1: קיים מימוש של A,B,C כך ש:
- הקוד מתקמפל.
 - A,B,C הן מחלקות (class-ים).
 - קיימים שני מימושים שונים של main שעבור הראשון, שורה 1 של func תזרוק שגיאה ושורה 2 של func לא תזרוק שגיאה, ומימוש שני שבו שורה 2 תזרוק שגיאה ושורה 1 לא.

- טענה 2: קיים מימוש של A,B,C והשלמה של הפונק' main כך ש:
- לא מתקיים B instanceof C וגם לא מתקיים C instanceof B.
 - הקוד מתקמפל ורץ ללא שגיאות.

- טענה 3: קיים מימוש של A,B,C כך ש:
- שורה 4 מתקמפלת ושורה 3 לא.

שאלה זו מצריכה נימוק! נמקו בקצרה את תשובתכם לגבי נכונותה או אי נכונותה של כל טענה.

בחר'י בתשובה הטובה ביותר:

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

```

public class Q6 {
    public static void main(String[] args) {
        Stream<Integer> s = Stream.generate(new MyNumbers());
        s = s.filter(x-> x%2 == 0)
            .peek(x->System.out.print("x "));

        // #
        /* System.out.println(s.allMatch(x->x < 5)); */

        // ##
        /* System.out.println(s.anyMatch(x->x >= 5)); */

        // ###
        /* System.out.println(
            s.map(x-> x < 5).limit(1)
                .reduce((x,y) -> x & y).get()); */
    }
}

public class MyNumbers implements Supplier<Integer>{
    int i = 0;

    @Override
    public Integer get() {
        i = (i + 1) % 7;
        return i;
    }
}

```

נרצה להוסיף פעולות לזרם s שנוצר ב main כך שבעת הרצתו יודפס הפלט הבא: x x x false. לפניכם 3 אופציות להשלמת הפעולות על s. עליכם לבחון כל אחת בנפרד.

- טענה 1: הוצאת הפקודה # מהערה תייצר את הפלט הנדרש.
 טענה 2: הוצאת הפקודה ## מהערה תייצר את הפלט הנדרש.
 טענה 3: הוצאת הפקודה ### מהערה תייצר את הפלט הנדרש.

הערה – כל הקוד מתקמפל ורץ.

שאלה זו מצריכה נימוק! נמקו בקצרה את תשובתכם לגבי נכונותה או אי נכונותה של כל טענה.

בחר/י בתשובה הטובה ביותר:

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

```

public class Q7 {
    public static void main(String[] args) {
        List<Integer> lst = Arrays.asList(1,2,3,4,5);
        Iterator<Integer> it = new MyCrazyIterator<>(lst);
        while(it.hasNext()) {
            System.out.print(it.next() + " ");
        }
    }
}

public class MyCrazyIterator<T> implements Iterator<T>{
    Iterator<T> innerIt;
    List<T> chooseFrom = new ArrayList<>();
    Random rand = new Random();

    public MyCrazyIterator(Collection<T> it) {
        innerIt = it.iterator();
    }

    @Override
    public boolean hasNext() {
        return innerIt.hasNext();
    }

    @Override
    public T next() {
        // next random boolean
        if (rand.nextBoolean() || chooseFrom.size() == 0) {
            T next = innerIt.next();
            chooseFrom.add(next);
            return next;
        }
        else {
            /* nextInt(b) returns a random element from
            the group {0,1,2,... b-1} */
            int randIndex = rand.nextInt(chooseFrom.size());
            return chooseFrom.get(randIndex);
        }
    }
}

```

לפניכם שלוש טענות על הקוד:

- טענה 1: בכל ריצה שמסתיימת המספר 5 יודפס פעם אחת בלבד ויהיה האיבר האחרון שיודפס.
- טענה 2: ניתן לקבל ריצה שהסתיימה ובה האיבר 3 מודפס לפני הפעם הראשונה שהאיבר 2 הודפס.
- טענה 3: ניתן לקבל ריצה שהסתיימה ובה הודפסו פחות מ 5 מספרים.

בחר/י בתשובה הטובה ביותר:

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

שאלה 8 (5 נק'):

```
class Outer{
    private int i;
    public class Inner{
        private int j;

        /*
        public int getNum(Outer o) {
            return o.i + j;
        } */
    }
}
```

לפניכם מספר טענות המתייחסות למחלקות Inner ו Outer

טענה 1: ניתן לייצר אובייקטים מטיפוס Inner רק בתוך קוד המחלקה Outer.

טענה 2: אם נוציא את הפונקציה getNum מהערה, הקוד יתקמפל.

טענה 3: אם נמיר את Inner להיות static class וגם נוציא את getNum מההערה, הקוד יתקמפל.

בחר/י בתשובה הטובה ביותר:

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

שאלה 9 (8 נקודות) – שאלה זו מצריכה נימוק

מה יקרה בהרצת התוכנית הבאה?

```
public class Q9 {
    public static void main(String[] args) {
        System.out.println(func());
    }

    public static int func() {
        int sum = 0;
        try {
            List<Integer> ints = Arrays.asList(1,2,3);
            for (int i : ints){
                try {
                    if (i == 2) {
                        throw new EOFException();
                    }
                    if (i == 1) {
                        throw new Exception();
                    }
                    sum += 1;
                }
                catch(EOFException exp) {
                    sum += 2;
                    throw new Exception();
                }
                catch(Exception exp) {
                    sum += 3;
                }
            }
            return sum;
        }
        catch (Exception exp) {
            return sum += 7;
        }
    }
}
```

שאלה זו מצריכה נימוק – הסבירו בקצרה מה קורה בכל איטרציה של הריצה.

מה יקרה בסוף ריצת התוכנית?

- א. בזמן ריצת התוכנית תיזרק שגיאת זמן ריצה.
- ב. יודפס 5
- ג. יודפס 6
- ד. יודפס 7
- ה. יודפס 8
- ו. יודפס 9

- ז. יודפס 10
- ח. יודפס 11
- ט. יודפס 12
- י. יודפס 13
- יא. יודפס 14

שאלה 10 (5 נק')::

```
public class A<K,V>{
    public <T> void func1(List<? extends T> l1, List<? extends T> l2){}
    public static <T> void func2(List<K> l1, List<V> l2){}
    public void func3(List l1, List<? extends V> l2){}

    public static void main(String[] args) {
        A<String, Integer> a = new A<>();
        List<String> lStr = new ArrayList<>();
        List<Integer> lInt = new ArrayList<>();
        a.func1(lStr, lInt);
        a.func2(lStr, lInt);
        a.func3(lStr, lInt);
    }
}
```

לפניכם 3 טענות על הקוד המצורף:

- טענה 1: הפונקציה func1 מתקמפלת וגם הקריאה ל func1 מתקמפלת.
- טענה 2: הפונקציה func2 מתקמפלת וגם הקריאה ל func2 מתקמפלת.
- טענה 3: הפונקציה func3 מתקמפלת וגם הקריאה ל func3 מתקמפלת.

בחר\י בתשובה הטובה ביותר:

- א. רק טענה 1 נכונה.
- ב. רק טענה 2 נכונה.
- ג. רק טענה 3 נכונה.
- ד. רק טענות 1+2 נכונות.
- ה. רק טענות 1+3 נכונות.
- ו. רק טענות 2+3 נכונות.
- ז. כל הטענות נכונות.
- ח. כל הטענות לא נכונות.

שאלה 11 (5 נקודות).

נתונות המחלקות Sub ו Base. נרצה להוסיף ל Sub מימוש לפונקציה func. מבין המימושים המוצעים, איזה מימוש יתקמפל? עליכם להתייחס לכל מימוש בנפרד.

```
public class Base{
    public String func(String str, List<String> lst) {return null;}
}

public class Sub extends Base{
    //op1
    /* public String func(String str, List<Integer> lst) {return null;} */

    //op2
    /* public String func(Object str, List<String> lst) throws Exception
        {return null;} */

    //op3
    /* public Object func(String str, List<String> lst) {return null;} */
}
```

בחר/י בתשובה הטובה ביותר:

- א. רק אופציה 1 מתאימה.
- ב. רק אופציה 2 מתאימה.
- ג. רק אופציה 3 מתאימה.
- ד. רק אופציות 1+2 מתאימות.
- ה. רק אופציות 2+3 מתאימות.
- ו. רק אופציות 1+3 מתאימות.
- ז. כל האופציות מתאימות.
- ח. כל האופציות לא מתאימות.

שאלה 12 (5 נק'):

נממש enum חדש באופן הבא:

```
public enum MyEnum{
    One(1),
    Two(2),
    Three(3),
    Four(4);

    int value;
    private MyEnum(int i){
        this.value = i;
    }

    public int getValue() {
        return this.value;
    }
}
```

נרצה להשתמש ב enum שהגדרנו בתוכנית הבאה:

```
public class EnumTest{

    public static void main(String[] args) {
        System.out.println(func(MyEnum.One) + func(MyEnum.Two));
    }
    public static int func(MyEnum e) {
        int sum = 0;
        switch(e) {
            case Two:
                sum += e.value*2;
            case Three:
                sum += e.value*3;
                break;
            default:
                sum += e.value;
        }
        return sum;
    }
}
```

בחר/י בתשובה הטובה ביותר:

- א. יש שגיאת קומפילציה ב MyEnum בגלל שאסור להגדיר שדות.
- ב. יש שגיאת קומפילציה ב MyEnum בגלל שאסור להגדיר בנאים.
- ג. יש שגיאת קומפילציה ב MyEnum בגלל שאסור להגדיר פונקציות.
- ד. הקוד מתקמפל אך בזמן ריצה תהיה שגיאה בגלל אי טיפול במקרה של One.
- ה. הקוד יתקמפל, ירוץ וידפיס 5
- ו. הקוד יתקמפל, ירוץ וידפיס 6
- ז. הקוד יתקמפל, ירוץ וידפיס 7
- ח. הקוד יתקמפל, ירוץ וידפיס 8
- ט. הקוד יתקמפל, ירוץ וידפיס 9
- י. הקוד יתקמפל, ירוץ וידפיס 10
- יא. הקוד יתקמפל, ירוץ וידפיס 11
- יב. מלבד תשובה זו יש יותר מתשובה אחת נכונה.

שאלה 13 (5 נק):

```
public class A{
    public static int i = 5;
    public int k = 10;

    public void func(int[] arr) {
        this.k += num;
    }
}
```

שאלה זו מתייחסת למחלקה A ולשימושים אפשריים במחלקה זו במערכת כלשהי.

בחר/י בתשובה הטובה ביותר:

- א. i לא ייוצר בזכרון לפני שבוצעה קריאה לבנאי של A באיזשהו מקום בקוד.
- ב. k נשמר על המחסנית (stack).
- ג. בעת הקריאה ל func, נוצר עותק (copy) של המערך שמועבר כפרמטר, ו arr יצביע לעותק זה.
- ד. לאחר הקריאה ל func, אם נפעיל את ה Garbage Collector בצורה יזומה, המערך שאליו הצביע arr יעלם מהזכרון.
- ה. מלבד תשובה זו יש יותר מתשובה נכונה אחת.
- ו. מלבד תשובה זו כל התשובות לא נכונות.

שאלה 14 (5 נק):

שאלה זו מתייחסת לירושה ולמנשקים:

- א. מחלקה לא יכולה לממש שני מנשקים שונים אם בשניהם מוגדרת פונקציה אבסטרקטית עם אותה החתימה.
- ב. מחלקה אבסטרקטית יכולה לרשת ממחלקה רגילה.
- ג. מנשק יכול לרשת ממחלקה אבסטרקטית אם היא לא מכילה שדות ומימושים.
- ד. במחלקה אבסטרקטית ניתן להגדיר פונקציות אבסטרקטיות בניראות פרטית (private).
- ה. מלבד תשובה זו כל התשובות לא נכונות.
- ו. מלבד תשובה זו יש יותר מתשובה נכונה אחת.

שאלה 15 (6 נק):

לפניכם 3 טענות המתייחסות לקוד הבא:

```
public static void func() {  
    int[] arr1 = new int[]{1,2,3};  
    int[] arr2 = arr1;  
    arr1 = new int[] {3,4,5};  
    func();  
}
```

- טענה 1: המשתנה arr1 נשמר על ה stack והמערך {1,2,3} נשמר על ה heap.
טענה 2: לאחר ביצוע השורה השלישית הביטוי arr1 == arr2 יחזיר True.
טענה 3: אם נריץ את func, תוך כדי ריצה יגדל גם השטח התפוס על ה heap וגם השטח התפוס על ה stack.
- א. רק טענה 1 נכונה.
 - ב. רק טענה 2 נכונה.
 - ג. רק טענה 3 נכונה.
 - ד. רק טענות 1+2 נכונות.
 - ה. רק טענות 1+3 נכונות.
 - ו. רק טענות 2+3 נכונות.
 - ז. כל הטענות נכונות.
 - ח. כל הטענות לא נכונות.

שאלה 16 (6 נק):

```
class Test{  
  
    public interface TestService {  
        *****  
    }  
  
    public static void func(TestService tS, int i) {  
        tS.printMessage("Test " + i);  
    }  
  
    public static void main(String[] args) {  
        func(s -> System.out.println(s), 1);  
    }  
}
```

בשביל שהתוכנית תרוץ תייצר את ההדפסה Test 1, כיצד צריך להחליף את ***** ?
בחר/י בתשובה הטובה ביותר:

- א. public String printMessage(String str);
- ב. public String printMessage(String str, int i);
- ג. public void printMessage();
- ד. public String printMessage();
- ה. public void printMessage(String str);
- ו. public void printMessage(String str, int i);

שאלה 17 (5 נק):

לפניכם מספר טענות על Factory. בתסריט שלנו נתון המנשק MyI שבו משתמשת הלקוחה MyClient, ונרצה לבנות Factory בשביל יצירת אובייקטים מטיפוס MyI.

בחר'י בתשובה הטובה ביותר:

- א. ה Factory נועד לאפשר ל MyClient לייצר אובייקטים מטיפוס MyI מבלי שקוד הלקוחה יכיר את המימושים השונים של MyI.
- ב. ה Factory נועד לאפשר לספקית של MyI לעשות שימוש חוזר במימושים השונים ל MyI, ובכך למנוע שכפול קוד.
- ג. ה Factory מאפשר לספקית של MyI לכתוב מימושים קונקרטיים של MyI מבלי לממש בנאים למחלקות אלה.
- ד. מלבד תשובה זו כל התשובות לא נכונות.
- ה. מלבד תשובה זו יש יותר מתשובה נכונה אחת.

שאלה 18 (5 נק):

```
public static void main(String[] args) {  
    MyInterface myI = new MyInterfaceImp();  
    myI.func();  
}
```

נתון שהקוד הבא מתקמפל. מה ניתן להגיד על המנשק MyInterface ועל המחלקה MyInterfaceImp?
בחר'י בתשובה הטובה ביותר:

- א. יתכן שהשירות ממומש ב MyInterfaceImp אך לא מופיע ב MyInterface.
- ב. יתכן שהשירות מוגדר כ abstract ב MyInterface, ואין לו מימוש ב MyInterfaceImp ולא בשום מחלקה ש MyInterface יורשת ממנה.
- ג. יתכן שהשירות מוגדר כ default ב MyInterface, ואין לו מימוש ב MyInterfaceImp ולא בשום מחלקה ש MyInterface יורשת ממנה.
- ד. מלבד תשובה זו יש יותר מתשובה נכונה אחת.
- ה. מלבד תשובה זו כל התשובות לא נכונות.