פרק 7 – ביצוע-חוזר

הצגת הפרק

מטרת הפרק

הצגת הנושא ביצוע חוזר של תת-משימות באמצעות שני מבנים של הוראה לביצוע חוזר: המבנה *בצע מספר פעמים*... (משפט for) והמבנה *כל עוד*... *בצע*... (משפט while). בפרק זה מושם דגש על תבניות אלגוריתמיות כגון: מניה, צבירה, מציאת מקסימום או מינימום ופירוק מספר לספרותיו.

מושגים

- ביצוע חוזר
- לולאה וגוף הלולאה
- פעולת חלוקה בשלמים
- מונה
- צובר
- מקסימום
- מינימום
- ערך נלווה למקסימום או מינימום
- זקיף
- משתנה בוליאני
- קשר לוגי not

דגשים ודידקטיקה

- מפרק זה ואילך הפתרונות מאופיינים בכך שהם פונים לשימוש בתבניות אלגוריתמיות באופן משמעותי, פניה זו מוזכרת במפורש בתשובות לשאלות. אנו מציעים לך המורה להקפיד על כך בהוראת הפרק.
- סעיף 7.1 משמש כמדרגה להצגה הרחבה של ביצוע חוזר המופיעה בסעיף 7.4. לכן, הביצוע החוזר המוצג בסעיף 7.1 כולל את המשפט **for** בו מספר הפעמים של הביצוע החוזר נקבע **לפני** תחילת ביצועו. הדגש בסעיף זה הוא על ניסוח תת-משימה או קבוצת תת-משימות לביצוע חוזר, ניסוח שנעשה לפני כתיבת האלגוריתם המלא.
- בסעיף 7.4 מושם דגש על ניסוח מדויק של תנאי להמשך הביצוע החוזר; כלומר, תנאי הכניסה ללולאה. כמו כן, מורחב הדגש על ניסוח תת-משימות לביצוע חוזר.
- לסיכום סעיף 7.4 מומלץ להדגים כיצד ניתן לכתוב כל משפט **for** כמשפט **while** על מנת להמחיש את הדמיון ביניהם, אך יחד עם זאת, יש להבהיר שכאשר ניתן לדעת את מספר הביצועים החוזרים לפני תחילת ביצוע הלולאה, נהוג להשתמש במשפט **for**.
- בסעיף 7.7 מוצג קינון הוראות לביצוע חוזר. המבנה המקונן של לולאות המוצג בפרק כולל לולאת **for** בתוך לולאת **for** שהוא המבנה המקונן הפשוט ביותר של לולאות. המטרה היא להציג נושא זה בפשטות ולהדגיש את אופן חישוב הזמן לביצועו. אין מטרה להעמיק במבנים מקוננים של לולאות, ולכן אין קינון לולאות **while**. קינון לולאות יוצג שוב בהמשך הלימוד, בפרק 12, בו יוצגו מערכים דו-מימדים.
- כמו בפרקים הקודמים, גם בפרק זה מפותח אלגוריתם בשלבים. יש להקפיד במהלך

הפיתוח על ניסוח התת-משימות לביצוע חוזר ועל ניסוח תנאי מדויק לבצוע החוזר. כמו כן, יש להקפיד על בניית טבלאות מעקב מסודרות כדי להטמיע את מהלך הביצוע החוזר.

- הפרק כולל שאלות פיתוח ושאלות ניתוח. שאלות הניתוח מחדדות את ההבנה של האלגוריתם ומדגישות את החשיבות של הנושא. מדגישות את החשיבות של נושא נכונות של אלגוריתמים אשר הוצג בפרק הקודם.

פתרונות

שאלה 7.2

נסחו עבור כל אחת מן הבעיות האלגוריתמיות הבאות קבוצת תת-משימות לביצוע-חוזר:

א. הקלט הוא 20 מרחקים הנתונים במיילים, והפלט הוא 20 המרחקים בקילומטרים (1 מייל = 1.6 קילומטר).

ב. הקלט הוא עשר אותיות מן האי"ב הלועזי השונות מהאות Z, והפלט הוא עשר האותיות שעוקבות לאותיות הנתונות.

ג. הקלט הוא 40 זוגות של ציונים (זוג ציונים עבור כל תלמיד), והפלט הוא רשימה של ארבעים מספרים: כל מספר הוא הממוצע של זוג הציונים המתאים לו.

תשובה 7.2

א. קליטת מרחק במיילים
חישוב ערכו של המרחק בקילומטרים
הצגה כפלט של הערך המחושב

ב. קליטת אות
חישוב האות העוקבת
הצגה כפלט של האות העוקבת

ג. קליטת ציון ראשון
קליטת ציון שני
חישוב ממוצע שני הציונים
הצגה כפלט של הממוצע

שאלה 7.3

לפניכם קטע תוכנית:

```
System.out.print('X');
for (i = 0; i < 10; i++)
    System.out.print("*");
System.out.print('X');
```

מהו פלט קטע התוכנית?

תשובה 7.3

פלט קטע התוכנית הוא: `X**********X`

שאלה 7.6

שנו את המשפטים הנמצאים בגוף הלולאה שבתוכנית לפתרון בעיה 3, כך שיחושב ממוצע הערכים הגבוהים מ-50.

תשובה 7.6

```
// ההוראה לביצוע-חוזר
for (int i = 1; i <= length; i++)
{
     System.out.print("Enter a number: ");
    num = in.nextDouble();
    if (num > LIMIT)
    {
        sum = sum + num;
        counterLarge++;  // counterLarge=counterLarge+1 -שקול ל
     }
}// for
average = sum / counterLarge;
System.out.println("Average is " + average);
```

שאלה 7.7

לפעמים ניתן להשתמש בערכו של מונה לחישוב מספר הנתונים המאופיינים בצורה הפוכה לנתונים שמנינו. למשל, נניח שבבעיה 3 יש להציג גם את מספר הערכים ה**קטנים או שווים** ל-50.
דרך אחת לחישוב מספר זה היא באמצעות שימוש במונה נוסף counterSmall (נוסף ל-counterLarge), אשר ערכו יוגדל ב-1 בכל פעם שנקלוט ערך הקטן או שווה ל-50.
אך בעצם אין צורך במונה נוסף. ניתן לבצע את החישוב בתום ביצוע הלולאה, באמצעות המונה counterLarge המופיע כבר בתוכנית. כיצד? הוסיפו לתוכנית את ההוראה או את ההוראות המתאימות.

תשובה 7.7

```
System.out.println( (length – counterLarge) +
                                " numbers are smaller than " + LIMIT);
```

שאלה 7.8

ציינו עבור כל אחת מן הבעיות האלגוריתמיות הבאות אם נחוץ לפתרונה צובר, מונה או אף אחד מהשניים:

א. קלט: רשימת מחירים, פלט: סך כל המחירים.
ב. קלט: רשימת מחירים, פלט: מספר המחירים הגבוהים מ-100.
ג. קלט: רשימת מחירים, פלט: מספר המחירים שמרכיב האגורות בהם שונה מ-0.
ד. קלט: רשימת מספרים, פלט: הערך המוחלט של כל אחד מהמספרים.
ה. קלט: רשימת מספרים, פלט: הממוצע של הערכים המוחלטים של המספרים.

תשובה 7.8

א. צובר
ב. מונה
ג. מונה
ד. לא צובר ולא מונה
ה. צובר

שאלה 7.9

יש לקלוט סדרה של תווים, אשר אורכה שמור במשתנה listSize, ולמנות את מספר התווים שהם אותיות גדולות (capital letters) באי"ב האנגלי.
בחרו משתנים, כתבו הוראה לביצוע-חוזר אשר לפניה אתחול מתאים, וישמו אותה במשפט **for**.

תשובה 7.9

משתנים נוספים ל-listSize ו-i:
ch – תו, ישמור תו נקלט.
capLettersCount – שלם, מונה שישמור את מספר התווים שהם אותיות גדולות באי"ב האנגלי.

ההוראה לביצוע החוזר (ולפניה האיתחול):

1. *אתחל את* capLettersCount *ל-*0
2. *בצע* listSize *פעמים*
   2.1. *קלוט תו ל-*ch
   2.2. *אם* ch ≥ 'A' *ובאם* ch ≤ 'Z'
      2.2.1 *הגדל ב-*1 *את ערכו של* capLettersCount

משפט **for**:

```
capLettersCount = 0;
// ההוראה לביצוע-חוזר
for (int i = 1; i <= listSize ; i++)
{
     ch = in.next().charAt();
     if (ch >= 'A' && ch <= 'Z')
           capLettersCount++;
}
```

שאלה 7.10

מטרת קטע התוכנית הבא היא מניית מספר תווי קלט השונים מן האות A. המשתנים counter ו-listSize הם מטיפוס שלם והמשתנה letter הוא מטיפוס תווי.

```
int counter = __________ ;
for (int i = 1; i <= listSize; i++)
{
      System.out.print("Enter a char: ");
     letter = in.next().charAt(0);
     if (_____________________________)
          _____________________________
}// for
System.out.println("There are " + counter
                                  + "letters different than A");
```

קטע התוכנית כולל לולאה.

א. כמה פעמים תתבצע הלולאה עבור הערך 10 ב-listSize?

ב. כמה פעמים תתבצע הלולאה עבור הערך 1 ב-listSize?

ג. השלימו את קטע התוכנית.

תשובה 7.10

א. 10 פעמים.

ב. פעם אחת.

ג.

```
int counter = 0;
for (int i = 1; i <= listSize; i++)
{
      System.out.print("Enter a char: ");
     letter = in.next().charAt(0);
     if ( letter != 'A' )
         counter++;
}// for
System.out.println("There are " + counter
                                        + "letters different than A");
```

שאלה 7.11

נתונה התוכנית הבאה:

```
/*
קלט: טור של 16 תווים מטופס ספורטוטו
_____________________________________:פלט
*/
import java.util.Scanner;
public class Toto
{
    public static void main (String[] args)
    {
        final int NUM_OF_GAMES = 16;
        int d = 0;
        char score;
        Scanner in = new Scanner(System.in);
        for (int i = 0; i < NUM_OF_GAMES; i++)
        {
            System.out.print("Enter the score: ");
            score = in.next().charAt(0);
            if (score == 'X')
                d++;
        } // for
        System.out.println(d);
    } // main
} // Toto
```

קלט התוכנית הוא טור בטופס הספורטוטו. כלומר, 16 תווים שכל אחד מהם מציין תוצאת משחק (1, 2 או X).

א. מהו הפלט עבור הקלט: 1 2 X 1 2 X 1 2 X 1 2 X 1 2 X 2 1?

ב. האם התוכנית כוללת מונה או צובר? אם כן, מהו או מהם?

ג. כמה פעמים תתבצע לולאת התוכנית?

ד. הביאו דוגמת קלט אשר הפלט עבורה הוא 0.

ה. הביאו דוגמת קלט אשר הפלט עבורה הוא 15.

ו. מה מאפיין את הקלטים אשר הפלט עבורם הוא 1?

ז. מהם ערכי הפלט האפשריים?
ח. מהי מטרת התוכנית? בחרו שם משמעותי למשתנה d.

תשובה 7.11

א. 5
ב. התוכנית כוללת מונה, המשתנה d.
ג. לולאת התוכנית תתבצע 16 פעמים.
ד. 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
ה. X X X X X X X X X X X X X X X X 2
ו. קלטים אשר הפלט עבורם הוא 1 כוללים בדיוק X אחד.
ז. ערכי הפלטים האפשריים הם המספרים השלמים בין 0 ל-16.
ח. מטרת התוכנית היא הצגה כפלט של מספר ה-X-ים בקלט, כלומר מספר תוצאות התיקו בטור הספורטוטו הנתון. שם משמעותי למשתנה d: draws (כלומר, תוצאות תיקו).

שאלה 7.15

פתחו וישמו אלגוריתם שיקבל כקלט רשימה של 50 מספרים ויציג כפלט את **מכפלתם** של המספרים הקטנים מ-10.

תשובה 7.15

רעיון הפתרון:

באלגוריתם זה אנו נדרשים לכתוב לולאה המתבצעת 50 פעמים, ובכל מחזור פעולה נצבור את ערכי הקלט על ידי **מכפלתם** ולא על ידי סכימתם כפי שעשינו עד כה. בשל כך, עלינו לאתחל את הצובר ל-1, ולא ל-0 כפי שעשינו עד כה.

תבניות:

צבירה

בחירת משתנים:

num – שלם, משתנה הקלט
mult – שלם, מכפלת משתני הקלט הקטנים מ-10.

האלגוריתם:

1. *אתחל את* mult *ל-*1
2. *בצע* 50 *פעמים*
3. *קלוט מספר שלם ל-*num
4. *אם* num < 10
   4.1. *הכפל את* mult *ב-* num *והשם ב-* mult
5. *הצג כפלט את* mult

יישום האלגוריתם:

```
/* קלט: רשימה של 50 מספרים
10-פלט: מכפלת המספרים הקטנים מ- */
import java.util.Scanner;
public class Multiply
{
	public static void main(String[] args)
	{
		int num;
		int mult = 1;
		Scanner in = new Scanner(System.in);
		for (int i=1; i<=50; i++)
		{
			System.out.print("Enter a number: ");
			num = in.nextInt();
			if (num < 10)
				mult = mult * num;
		}
		System.out.println(mult);
	}// main
}// Multiply
```

שאלה 7.16

על פי הגדרת בעיה 4, יכול להינתן כנתון ראשון בקלט כל מספר שלם. תארו את מהלך ביצוע התוכנית `PrintNumbers` אם הערך הראשון בקלט הוא המספר השלם 0. תארו את מהלך הביצוע של התוכנית אם הערך הראשון בקלט הוא המספר השלם 2-.

תשובה 7.16

אם הערך הראשון בקלט הוא המספר השלם 0 או המספר השלם 2- הלולאה לא תתבצע, ולא יוצג כל פלט למסך.

שאלה 7.17

פתחו אלגוריתם שמקבל כקלט מספר חיובי שלם n, מחשב את n!, ומציג את הערך שחושב כפלט. ישמו את האלגוריתם בשפת Java.

להזכירכם: n! היא מכפלת המספרים מ-1 עד n, כלומר: $n! = 1 \cdot 2 \cdot \ldots \cdot n$.

תשובה 7.17

רעיון הפתרון:

הפתרון לשאלה זו מבוסס על התבנית **צבירה**, אך לא צבירה של סכום, אלא צבירת מכפלה. הלולאה תתבצע n פעמים (לפי המספר הנקלט), כאשר בכל ביצוע נוסף של הלולאה יוכפל צובר המכפלה במספר הביצועים החוזרים שהתבצעו עד כה. מספר הביצועים החוזרים שהתבצעו עד כה שמור במשתנה הבקרה של הלולאה i. יש לשים לב לאתחול צובר המכפלה ל-1.

ניתוח הבעיה באמצעות דוגמאות:

- קלט: 4 פלט: 24
- קלט: 6 פלט: 720

תבניות:
צבירה

בחירת משתנים:
n – שלם, גודל העצרת לחישוב
factorial– שלם, תוצאת החישוב: !n

האלגוריתם:
1. *אתחל את* factorial *ל-*0
2. *קלוט מספר שלם חיובי ב-*n
3. *עבור* i *מ-*1 *עד* n *בצע*:
   3.1. *הכפל את* factorial *ב-*i *והשם ב-*factorial
4. *הדפס את ערך* factorial *כפלט*

יישום האלגוריתם:

```
/* קלט: מספר שלם
*/ פלט: עצרת ערך הקלט
import java.util.Scanner;
public class Factorial
{
	public static void main(String[] args)
	{
		int n;
		int factorial = 1;
		Scanner in = new Scanner(System.in);
		System.out.print("Enter a number: ");
		n = in.nextInt();
		for (int i=1; i<=n; i++)
			factorial = factorial * i;
		System.out.println(n + "!=" + factorial);
	}// main
}// Factorial
```

שאלה 7.18

פתחו אלגוריתם שמקבל כקלט מספר חיובי שלם n, ומחשב את הסכום: $\frac{1}{1}+\frac{1}{2}+...+\frac{1}{n}$.

ישמו את האלגוריתם בשפת java.

תשובה 7.18

רעיון הפתרון:

הפתרון לשאלה זו מבוסס על התבנית **צבירה**. הלולאה תתבצע n פעמים, בכל ביצוע נוסף של הלולאה ייצבר (מספר הנתונים שנקלטו עד כה)/1. מספר הנתונים שנקלטו עד כה שמור במשתנה הבקרה של הלולאה (i). יש להקפיד על חלוקה שאינה חלוקה בשלמים.

ניתוח הבעיה באמצעות דוגמאות:

- קלט: 4 פלט: 2.083
- קלט: 6 פלט: 2.449

תבניות:

צבירה

בחירת משתנים:

n – שלם, גודל העצרת לחישוב

sum– ממשי, הסכום המצטבר

יישום האלגוריתם:

```
/* קלט: מספר שלם
   פלט: סכום סדרה מתוארת */
import java.util.Scanner;
public class EqSum
{
      public static void main(String[] args)
      {
            int n;
            double sum = 0;
            Scanner in = new Scanner(System.in);
            System.out.print("Enter a number: ");
            n = in.nextInt();
            for (int i=1; i<=n; i++)
                  sum = sum + (double)1/i;
            System.out.println("sum = " + sum);
      }// main
}// EqSum
```

שאלה 7.19

לפניכם קטע תוכנית הכתוב ב-java

```
for (int i = 1; i <= 50; i = i * 2)
    System.out.println(i);
```

א. עקבו באמצעות טבלת מעקב אחר קטע התוכנית? מה יודפס?

ב. נניח כי במקום הערך 50 מופיע ערך חיובי שלם כלשהו N. תארו מה מבצע קטע התוכנית כתלות בערך N.

תשובה 7.19

א.

המשפט לביצוע	i	i<=50	**פלט**
	?		
for (**int** i = 1; i <= 50; i = i * 2)	1	true	
System.out.println(i);	1		1
for (**int** i = 1; i <= 50; i = i * 2)	2	true	
System.out.println(i);	2		2
for (**int** i = 1; i <= 50; i = i * 2)	4	true	
System.out.println(i);	4		4
for (**int** i = 1; i <= 50; i = i * 2)	8	true	
System.out.println(i);	8		8
for (**int** i = 1; i <= 50; i = i * 2)	16	true	
System.out.println(i);	16		16
for (**int** i = 1; i <= 50; i = i * 2)	32	true	
System.out.println(i);	32		32
for (**int** i = 1; i <= 50; i = i * 2)	64	false	

יוצגו כל המספרים בין 1 ל-50 שהם חזקות של 2.

ב. קטע התוכנית מציג כפלט את כל המספרים בין 1 ל- N שהם חזקות של 2.

שאלה 7.21

בנו טבלת מעקב אחר מהלך ביצוע התוכנית `FindMax` לפתרון בעיה 5 עבור הקלט:

5 3 2 4 6 -9

כמה פעמים במהלך ביצוע התוכנית מושם ערך במשתנה `max`?

תשובה 7.21

מספר השורה	המשפט לביצוע	howMany	balance	max	i	i<=howMany	balance > max	פלט
		?	?	?	?			
1.	System.out.print("Enter the amount of ...");	?	?	?	?			Enter the amount of accounts:
2.	howMany = in.nextInt();	5	?	?	?			
3.	System.out.print("Enter the first balance: ");	5	?	?	?			Enter the first balance:
4.	balance = in.nextInt();	5	3	?	?			
5.	max = balance;	5	3	5	?			
6.	**for**(**int** i=2; i<=howMany; i++)	5	3	5	2	true		
6.1.	System.out.print("Enter balance of account" + i + ": ");	5	3	5	2			Enter balance of account 2:
6.2.	balance = in.nextInt();	5	2	5	2			
6.3.	**if** (balance > max)	5	2	5	2		false	
6	**for**(**int** i=2; i<=howMany; i++)	5	2	5	3	true		
6.1.	System.out.print("Enter balance of account" + i + ": ");	5	2	5	3			Enter balance of account 3:
6.2.	balance = in.nextInt();	5	4	5	3			
6.3.	**if** (balance > max)	5	4	5	3		false	
6	**for**(**int** i=2; i<=howMany; i++)	5	4	5	4	true		
6.1.	System.out.print("Enter balance of account" + i + ": ");	5	4	5	4			Enter balance of account 4:
6.2.	balance = in.nextInt();	5	6	5	4			
6.3.	**if** (balance > max)	5	6	5	4		true	
6.3.1.	max = balance;	5	6	6	4			
6	**for**(**int** i=2; i<=howMany; i++)	5	6	6	5	true		
6.1.	System.out.print("Enter balance of account" + i + ": ");	5	6	6	5			Enter balance of account 5:
6.2.	balance = in.nextInt();	5	-9	6	5			
6.3.	**if** (balance > max)	5	-9	6	5		false	
6	**for**(**int** i=2; i<=howMany; i++)	5	-9	6	6	false		
7	System.out.println("The maximum is " + max);	5	-9	6	6			The maximum is 6

במהלך התוכנית מושם ערך במשתנה max פעמיים.

שאלה 7.22

בפתרון בעיה 5 אותחל max לערכו של הנתון הראשון ברשימה. האם הפתרון היה נכון לו max היה מאותחל בערך קבוע כלשהו, למשל 0? הערה: זכרו כי ייתכן שזהו בנק לא מוצלח במיוחד וכל חשבונות הבנק בו במשיכת יתר, כלומר בעלי ערך שלילי.

תשובה 7.22

אם max היה מאותחל בערך קבוע כלשהו, למשל 0, האלגוריתם היה שגוי. אין באפשרותנו לדעת את טווח הערכים של הקלט ומכך לא נדע איזה ערך ניתן להציב במשתנה max כך שבוודאות יהיה ערך אחד לפחות הגדול ממנו. לדוגמה, אם נאתחל את max בערך הקבוע 0, וכל חשבונות הבנק בו נמצאים במשיכת יתר (זאת אומרת, בעלי ערך שלילי), לא ימצא ערך הגדול מ- max ולכן הוא יישאר 0, וזה יהיה הערך המוצג כפלט <u>למרות</u> שערך זה כלל לא היה בין קלטי התוכנית.

שאלה 7.23

נתון קטע התוכנית החלקי הבא לחישוב המספר **הקטן** ביותר ברשימת מספרים **חיוביים** נתונה, אשר אורכה שמור במשתנה len.

```
min = _______________
System.out.print("Enter the list size: ");
len = in.nextInt();
for (int i = 1; i <= len; i++)
{
      _____________________
      _____________________
      _____________________
}
System.out.println("The minimum is: " + min);
```

השלימו את קטע התוכנית (הוסיפו משתנה או משתנים במידת הצורך).
במשפט הראשון אתחלו את min לערך קבוע כלשהו (ולא לערך הראשון ברשימה).

תשובה 7.23

```
min = Integer.MAX_VALUE;
System.out.print("Enter the list size: ");
len = in.nextInt();
for (int i = 1; i <= len; i++)
{
      int num = in.nextInt();
      if (num < min)
            min = num;
}
System.out.println("The minimum is: " + min);
```

הערה: בשאלה זו נדרש שימוש בערך הקבוע `Integer.MAX_VALUE`. ערך זה אינו מוסבר בספר ומומלץ להציגו לתלמידים.

שאלה 7.24

פתחו וישמו אלגוריתם שמקבל כקלט רשימה של ציונים במדעי המחשב של 40 תלמידים. הפלט הוא הציון הגבוה ביותר מבין התלמידים **שנכשלו** במבחן (ציון נכשל הוא ציון הנמוך מ-55).
שימו לב: כאן הציון התורן הנקרא מהקלט אינו מושווה בכל מקרה למקסימום הנוכחי, אלא רק כאשר הוא קטן מ-55.

תשובה 7.24

רעיון הפתרון:

בפתרון השאלה מתאים להשתמש בתבנית **מציאת מינימום**. כפי שנאמר בשאלה, הציון התורן הנקרא מהקלט אינו מושווה בכל מקרה למקסימום הנוכחי, אלא רק כאשר הוא קטן מ-55. לכן, כל ציון ייבדק בתנאי המורכב משני תנאים: האם התלמיד נכשל **וגם** האם הציון גדול יותר מהמקסימלי שנשמר עד כה. המשתנה השומר את הציון הגבוה ביותר מבין אלה שנכשלו יאותחל ל-0.

תבניות:

מציאת מקסימום.

בחירת משתנים:

garde – שלם, ציון נקלט

maxFail – שלם, לשמירת הציון הגבוה ביותר מבין אלה שנכשלו

PASS_GRADE – שלם קבוע, ציון עובר מינימלי

יישום האלגוריתם:

```
/* קלט: ציוני 40 תלמידים
 */ פלט: הציון הגבוה ביותר מבין התלמידים שנכשלו במבחן
import java.util.Scanner;
public class MaxGradeOutOfFails
{
      public static void main (String[] args)
      {
            int grade;
            int maxFail  = 0;
            final int PASS_GRADE = 55;
            Scanner in = new Scanner(System.in);
            for (int i=1; i<=40; i++)
            {
                  System.out.print("Enter grade for student number "+
                                                          i + ": ");
                  grade = in.nextInt();
                  if (grade < PASS_GRADE && grade > maxFail)
                        maxFail = grade;
            }
            System.out.println("Maximum grade out of fails is: " +
                                                           maxFail);
      } //main
}// MaxGradeOutOfFails
```

שאלה 7.25

פתחו אלגוריתם שיקבל כקלט מספר חיובי שלם המציין את מספר שחקני מכבי ת״א, ואחר כך יקלוט רשימה של נתוני קליעות של השחקנים: עבור כל שחקן ייקלט מספרו (המספר שעל החולצה שלו), ומספר הנקודות שקלע במהלך העונה. פלט האלגוריתם יהיה **מספרו** של השחקן שקלע הכי הרבה נקודות במהלך העונה.

שימו ♥: במקרה זה, אנחנו לא מתבקשים להציג את מיקומו של השחקן שקלע הכי הרבה נקודות אלא את מספרו, לכן משתנה הבקרה המצביע על ה״מיקום״ של השחקן התורן בקלט אינו

מתאים כי הוא אינו מספרו הסידורי של השחקן. את מספרו של השחקן יש לקרוא מהקלט.

תשובה 7.25

רעיון הפתרון:

בפתרון שאלה זו מתאים להשתמש בתבנית **מציאת ערך נלווה למקסימום**. כפי שנאמר בשאלה, אנחנו לא מתבקשים להציג את מיקומו של השחקן שקלע הכי הרבה נקודות אלא את **מספרו**, לכן **בנוסף** למשתנה השומר את מספר הנקודות הגדול ביותר שקלע שחקן מבין נתוני הקלט, נשמור על ידי משתנה נוסף את מספרו של שחקן זה. המשתנה הנוסף השומר את מספרו של השחקן זה אינו דורש איתחול מבחינה אלגוריתמית כיוון שאינו משתתף בהשוואה אלא מקבל ערך ברגע שערכו של המשתנה השומר את מספר הנקודות הגדול ביותר מקבל ערך חדש. אבל, אם לא נאתחל משתנה זה נקבל שגיאת הידור כיוון שהמהדר מזהה משתנה זה כמשתנה שיתכן ויודפס ללא קבלת ערך. בשל אילוץ טכני זה נאתחל את המשתנה ל-0.

ניתוח הבעיה באמצעות דוגמאות:

- קלט: 27 3 37 9 14 5 3 פלט: 9
- קלט: 14 6 35 12 24 8 36 11 4 פלט: 11

תבניות:

מציאת ערך נלווה למקסימום

בחירת משתנים:

numOfPlayers – שלם, מספר השחקנים

points – שלם, מספר נקודות נקלט

player – שלם, מספר שחקן נקלט

maxPoints – שלם, לשמירת מספר הנקודות הגדול ביותר שקלע שחקן מבין נתוני הקלט

maxPointsPlayer – שלם, מספרו של השחקן שקלע את מספר הנקודות הגדול ביותר.

יישום האלגוריתם:

```
/*  קלט: מספר שחקנים, ועבור כל שחקן מספר שחקן ומספר הנקודות שקלע
 */ פלט: מספר השחקן שקלע את מספר הנקודות הגדול ביותר
import java.util.Scanner;
public class MaxPoibtsPlayer
{
      public static void main (String[] args)
      {
            int numOfPlayers;
            int player, points;
            int maxPoints = 0, maxPointsPlayer = 0;
            Scanner in = new Scanner(System.in);
            System.out.print("Enter number of players: ");
            numOfPlayers = in.nextInt();
            for (int i=1; i<=numOfPlayers; i++)
            {
                  System.out.print("Enter player shirt number: ");
                  player = in.nextInt();
                  System.out.print("Enter number of points for player
                                          number " + player + ": ");
                  points = in.nextInt();
                  if (points > maxPoints)
                  {
                        maxPoints = points;
                        maxPointsPlayer = player;
                  }
            }
            System.out.println("Player number " + maxPointsPlayer +
                                       " has highest number of points");
      }// main
}// MaxPoibtsPlayer
```

שאלה 7.26

תארו את הפלט עבור כל אחד מן הקלטים הבאים (משמאל לימין):

א. 1- 9 7 10 12

ב. 1- 20

תשובה 7.26

א. 38

ב. 20

שאלה 7.27

נתון קטע התוכנית הבא. הקלט הוא רשימת תווים המהווים מילה באנגלית ובסופה הזקיף '*'.

```
int i = 0;
char letter;
System.out.print("Enter the first letter of the word: ");
letter = in.next().charAt(0);
while (letter != '*')
{
    i = i + 1;
    System.out.print("Enter the next letter of the word: ");
    letter = in.next().charAt(0);
}
System.out.println(i);
```

מהי מטרת קטע התוכנית?

תשובה 7.27

מטרת קטע התוכנית היא למנות את מספר התווים במילה.

שאלה 7.29

אלון ובני מתחרים על תפקיד יושב ראש ועדת קישוט של הכיתה, מועמד זוכה אם **יותר** מחצי מן הבוחרים הצביעו עבורו. פתחו וישמו אלגוריתם אשר הקלט שלו הוא סדרה של התווים A ו-B (A עבור אלון, B עבור בני), המבטאת את קולות הבוחרים, ומסתיימת בזקיף '#'. הפלט שלו הוא הודעה אם אלון זכה או לא זכה ברוב קולות (כלומר ביותר מחצי מקולות הבוחרים).

למשל, עבור הקלט #ABBAABBAA הפלט המתאים הוא: Alon wins, ועבור הקלט #ABBABBAA הפלט המתאים הוא: Alon didn't win.

תשובה

רעיון הפתרון:

בפתרון בשאלה זו מתאים להשתמש בתבנית **איסוף בקיזוז**. נעזר במונה שתפקידו לשמור את ההפרש בין מספר ההצבעות לאלון לבין מספר ההצבעות לבני. מונה זה יאותחל ל-0. עבור כל בוחר שהצביע עבור אלון, **נוסיף** 1 למונה, ועבור כל בוחר שלא הצביע עבור אלון **נחסיר** 1 מהמונה. בסיום הקלט, אם ערך המונה הוא מספר חיובי, הרי שיותר ממחצית הבוחרים הצביעו עבור אלון. את הקלט נבצע על ידי לולאת זקיף.

ניתוח הבעיה באמצעות דוגמאות:

- קלט: #ABBABAB פלט: Alon didn't win
- קלט: #ABBABA פלט: Alon didn't win
- קלט: #ABBAA פלט: Alon won

תבניות:

איסוף בקיזוז

בחירת משתנים:

vote – תו, הצבעת בוחר

count – שלם, מונה המקזז את ההצבעות לאלון

האלגוריתם:

1. אתחל את count ל-0
2. קלוט תו ל-vote
3. כל עוד vote שונה מ-'#' בצע:
   3.1. אם vote שווה ל-'A'
      3.1.1. הגדל את ערכו של count ב-1
   3.2. אחרת
      3.2.1. הקטן את ערכו של count ב-1
   3.3. קלוט תו ל-vote
4. אם count > 0
   4.1. הדפס כפלט "אלון ניצח"
5. אחרת
   5.1. הדפס כפלט "אלון לא ניצח"

יישום האלגוריתם:

```
/*  קלט: סדרה של התווים המבטאת את קולות הבוחרים לועדת הקישוט
 */ פלט: האם אלון זכה ברב קולות
import java.util.Scanner;
public class ClassVotes
{
      public static void main (String[] args)
      {
            char vote;
            int count = 0;
            Scanner in = new Scanner(System.in);
            System.out.print("Enter vote: ");
            vote = in.next().charAt(0);
            while (vote != '#')
            {
                  if (vote == 'A')
                        count++;
                  else
                        count--;
                  System.out.print("Enter vote: ");
                  vote = in.next().charAt(0);
            }
            if (count > 0)
                  System.out.println("Alon won");
            else
                  System.out.println("Alon didn't win");

      }// main
}// ClassVotes
```

שאלה 7.33

מהו הפלט המתאים עבור כל אחד מן הקלטים הבאים:

א. 200 1000 800 600 600

ב. 1000 1200 800 900

תשובה 7.33

א. 200 5

ב. 900 4

שאלה 7.34

א. ישמו את האלגוריתם בשפת Java.

ב. בנו טבלת מעקב אחר ביצוע התוכנית (לפתרון בעיה 8) עבור הקלט:

500 1200 800 300 300

כמה פעמים יתבצע גוף הלולאה?

שימו ♥: בכל ביצוע-חוזר של הלולאה נקלט מרחק טיסה אחד, ולכן הערך השמור במשתנה counter הוא בעצם מספר הפעמים שמתבצעת הלולאה.

תשובה 7.34

א.

```
/*  קלט: סדרה של מרחקי טיסה
/* פלט: כמה טיסות נדרשו לקבלת הפרס, ומה יתרת הקילומטרים שנותרה ללקוח */
import java.util.Scanner;
public class Distances
{
      public static void main (String[] args)
      {
            int dist;
            int sum = 0;
            int counter = 0;
            int WIN_DISTANCE = 3000;
            Scanner in = new Scanner(System.in);
            while (sum <= WIN_DISTANCE)
            {
                  System.out.print("Enter new distance: ");
                  dist = in.nextInt();
                  sum += dist;
                  counter++;
            }
            System.out.println(counter + " " + (sum - WIN_DISTANCE));
      }// main
}// Distances
```

ב.

המשפט לביצוע	dist	sum	counter	sum <= WIN_DISTANCE	פלט
	?	0	0		
while(sum<=WIN_DISTANCE)	?	0	0	true	
System.out.print("Enter new distance: ");	?	0	0		Enter new distance:
dist = in.nextInt();	500	0	0		
sum += dist;	500	500	0		
counter++;	500	500	1		
while(sum<=WIN_DISTANCE)	500	500	1	true	
System.out.print("Enter new distance: ");	500	500	1		Enter new distance:
dist = in.nextInt();	1200	500	1		
sum += dist;	1200	1700	1		
counter++;	1200	1700	2		
while(sum<=WIN_DISTANCE)	1200	1700	2	true	
System.out.print("Enter new distance: ");	1200	1700	2		Enter new distance:
dist = in.nextInt();	800	1700	2		
sum += dist;	800	2500	2		
counter++;	800	2500	3		
while(sum<=WIN_DISTANCE)	800	2500	2	true	
System.out.print("Enter new distance: ");	800	2500	2		Enter new distance:
dist = in.nextInt();	300	2500	2		
sum += dist;	300	2800	2		
counter++;	300	2800	3		
while(sum<=WIN_DISTANCE)	800	2800	3	true	
System.out.print("Enter new distance: ");	300	2800	3		Enter new distance:
dist = in.nextInt();	300	2800	3		
sum += dist;	300	3100	3		
counter++;	800	3100	4		
while(sum<=WIN_DISTANCE)	800	3100	4	false	
System.out.println(counter + " " + (sum – WIN_DISTANCE));	800	3100	4		4 100

גוף הלולאה יתבצע 5 פעמים.

שאלה 7.35

נסחו תנאי כניסה בוליאני מתאים עבור כל אחד מן התיאורים הבאים של ביצוע-חוזר:

א. סכימת משקלי מכוניות (לצורך מעבר במעבורת) **כל עוד** הסכום אינו עולה על 100 טון.

ב. סכימת המספרים החיוביים השלמים המתחילים ב-1 **עד אשר** הסכום גדול מהערך הנתון כקלט.

ג. קריאת אותיות **עד אשר** נקלטות 10 אותיות A.

תשובה 7.35

א. כל עוד carWeight קטן או שווה ל-100 בצע:

ב. כל עוד sum קטן או שווה ל-input בצע:

ג. כל עוד counter קטן מ-10 בצע:

שאלה 7.36

בכל אחד מן הסעיפים הבאים מופיע קטע תוכנית הכולל משפט **while** ובו תנאי הכניסה חסר. השלימו את תנאי הכניסה לפי מטרת קטע התוכנית, ובנו טבלת מעקב אחר ביצוע הקטע השלם.

א. מטרת הקטע: הצגה של המספר הזוגי הקטן ביותר אשר גדול מנתון הקלט בהינתן שהקלט הוא מספר חיובי.

```
System.out.print("Enter a number: ");
num = in.nextInt();
i = 0;
while (_____________)
    i = i + 2;
System.out.println(i);
```

בנו טבלת מעקב אחר ביצוע קטע התוכנית השלם עבור הקלט 9.

ב. מטרת הקטע: הצגה של מכפלת שני נתוני הקלט, בהינתן שהם מספרים שלמים חיוביים.

```
System.out.print("Enter a number: ");
x = in.nextInt();
System.out.print("Enter a number: ");
y = in.nextInt();
counter = 0;
sum = 0;
while (_____________)
{
    sum = sum + x;
    counter = counter + 1;
}
System.out.println(sum);
```

בנו טבלת מעקב אחר ביצוע קטע התוכנית השלם עבור הקלט 3 4.

תשובה 7.36

א.

```
System.out.print("Enter a number: ");
num = in.nextInt();
i = 0;
while (i <= num)
    i = i + 2;
System.out.println(i);
```

טבלת מעקב אחר ביצוע קטע התוכנית עבור הקלט 9.

המשפט לביצוע	num	i	i <= num	פלט
	?	?		
System.out.print("...");	?	?		Enter a number:
num = in.nextInt();	9	?		
i = 0;	9	0		
while (i <= num)	9	0	true	
i = i + 2;	9	2		
while (i <= num)	9	2	true	
i = i + 2;	9	4		
while (i <= num)	9	4	true	
i = i + 2;	9	6		
while (i <= num)	9	6	true	
i = i + 2;	9	8		
while (i <= num)	9	8		
i = i + 2;	9	10		
while (i <= num)	9	10	false	
System.out.println(i);	9	10		10

ב. מטרת הקטע: הצגה של מכפלת שני נתוני הקלט, בהינתן שהם מספרים שלמים חיוביים.

```
System.out.print("Enter a number: ");
x = in.nextInt();
System.out.print("Enter a number: ");
y = in.nextInt();
counter = 0;
sum = 0;
while (counter < y)
{
    sum = sum + x;
    counter = counter + 1;
}
System.out.println(sum);
```

טבלת מעקב אחר ביצוע קטע התוכנית עבור הקלט 3 4.

המשפט לביצוע	x	y	counter	sum	counter < y	פלט
	?	?	?	?		
`System.out.print("...");`	?	?	?	?		`Enter a number:`
`x = in.nextInt();`	3	?	?	?		
`System.out.print("...");`	3	?	?	?		`Enter a number:`
`x = in.nextInt();`	3	4	?	?		
`counter = 0;`	3	4	0	?		
`sum = 0;`	3	4	0	0		
`while` `(counter < y)`	3	4	0	0	`true`	
`sum = sum + x;`	3	4	0	3		
`counter = counter + 1;`	3	4	1	3		
`while` `(counter < y)`	3	4	1	3	`true`	
`sum = sum + x;`	3	4	1	6		
`counter = counter + 1;`	3	4	2	6		
`while` `(counter < y)`	3	4	2	6	`true`	
`sum = sum + x;`	3	4	2	9		
`counter = counter + 1;`	3	4	3	9		
`while` `(counter < y)`	3	4	3	9	`true`	
`sum = sum + x;`	3	4	3	12		
`counter = counter + 1;`	3	4	4	12		
`while` `(counter < y)`	3	4	4	12	`false`	
`System.out.println(sum);`	3	4	4	12		`12`

שאלה 7.37

נתון קטע התוכנית הבא אשר הקלט שלו הוא מספר חיובי שלם:

```
System.out.print("Enter a number: ");
limit = in.nextInt();
s = 0;
c = 0;
while (s < limit)
{
    c = c + 1;
    s = s + c;
}
System.out.println(c);
```

קטע התוכנית כולל מונה `c` וצובר `s`. הצובר צובר את ערכיו של המונה במהלך הביצוע-החוזר.

א. מהו הפלט עבור הקלט 1? מהו הפלט עבור הקלט 11?

ב. ציינו שני קלטים שונים שעבורם יהיה הפלט 5. מהו מספר הפעמים לביצוע-החוזר עבור קלטים אלה?

ג. מהי מטרת קטע התוכנית?

היעזרו בטבלת מעקב כדי לענות על סעיפים א ו-ב.

תשובה 7.37

א. עבור הקלט 1 הפלט הוא 1

עבור הקלט 11 הפלט הוא 5

ב. שני קלטים שונים עבורם יהיה הפלט 5 : 12 15

עבור קלטים אלה מספר הפעמים לביצוע-חוזר הוא 5.

ג. מטרת קטע התוכנית היא הצגה כפלט של מספר המספרים השלמים העוקבים המתחילים ב-1 אשר סכומם גדול או שווה לנתון הקלט.

שאלה 7.38

נתון קטע התוכנית הבא אשר הקלט שלו הוא מספר חיובי שלם, ו-`TOP_LIMIT` הוא קבוע שערכו 100 :

```
System.out.print("Enter a number: ");
num = in.nextInt();
mult = 1;
i = 0;
while ( (i < num) && (mult < TOP_LIMIT) )
{
    i = i + 1;
    mult = mult * i;
}
System.out.println(mult);
```

קטע התוכנית הנתון כולל צובר `mult` אשר צובר ערך של מכפלה, והוא מאותחל בערך 1.

א. מהו הפלט עבור הקלט 2 ? ומהו הפלט עבור הקלט 5?

ב. מהו הקלט שעבורו יהיה הפלט 24? ומהו מספר הפעמים של הביצוע-החוזר עבור קלט זה?

ג. מהי מטרת קטע התוכנית?

היעזרו בטבלת מעקב כדי לענות על סעיפים א ו-ב.

תשובה 7.38

א. עבור הקלט 1 הפלט הוא 2

עבור הקלט 5 הפלט הוא 120

ב. הפלט 24 יתקבל עבור הקלט 4. עבור קלט זה מספר הפעמים לביצוע-חוזר הוא 4.

ג. מטרת קטע התוכנית היא הצגה כפלט של num! ולכל היותר של הערך הקטן ביותר הגדול מהגבול של `TOP_LIMIT` (1 * 2 * 3 * ... * num) עבור קלט נתון שערכו num.

שאלה 7.39

פתחו אלגוריתם אשר הקלט שלו הוא מספר חיובי שלם, והפלט שלו הוא החזקה הקטנה ביותר של 2 אשר גדולה מנתון הקלט. למשל : עבור הקלט 7 הפלט הדרוש הוא 8 (כי $2^3=8$), ועבור הקלט 8 הפלט הדרוש הוא 16 (כי $2^4=16$). ישמו את האלגוריתם בשפת Java.

במהלך הפיתוח הקפידו על ניסוח תת-משימות לביצוע-חוזר ועל ניסוח תנאי לביצוע-חוזר.

שימו ♥ : באלגוריתם זה יש להשתמש בצובר של מכפלה, ולא בפעולת החזקה pow המוגדרת במחלקה Math.

תשובה 7.39

רעיון הפתרון:

באלגוריתם זה אנו נדרשים להכפיל את הערך פי 2 עד אשר ערך התוצאה גדולה מנתון הקלט. לשם כך נשתמש בצובר מכפלה.

ניתוח הבעיה באמצעות דוגמאות:

- קלט: 4 פלט: 8
- קלט: 20 פלט: 32

תבניות:

צבירה

בחירת משתנים:

num – שלם, נתון הקלט

pow– שלם, צובר המכפלה

יישום האלגוריתם:

```
/* קלט: מספר שלם
*/ פלט: החזקה הקטנה ביותר של 2 אשר גדולה מנתון הקלט
import java.util.Scanner;
public class Power2
{
	public static void main(String[] args)
	{
		int num;
		int pow = 1;
		Scanner in = new Scanner(System.in);
		System.out.print("Enter a number: ");
		num = in.nextInt();
		while (pow <= num)
			pow = pow * 2;
		System.out.println(pow);
	}// main
}// Power2
```

שאלה 7.40

א. שנו את התוכנית `DigitCount` לפתרון בעיה 9, כך שהפלט יכלול את **סכום** ספרות המספר, למשל עבור הקלט 153 הפלט המתאים הוא 9.

ב. שנו את התוכנית `DigitCount` לפתרון בעיה 9, כך שהפלט יכלול רק את **מספר** הספרות **האי-זוגיות** במספר הנתון. למשל עבור הקלט 150 הפלט המתאים הוא 2.

ג. שנו את התוכנית `DigitCount` לפתרון בעיה 9, כך שהפלט יכלול רק את **מכפלת** ספרות המספר. שימו לב לאתחול נכון של המשתנה השומר את המכפלה.

תשובה 7.40

א.

```
/*
קלט: מספר חיובי שלם
פלט: סכום ספרות המספר הנתון
*/
import java.util.Scanner;
public class DigitSum
{
    public static void main (String [] args)
    {
        int sum = 0;
        int num;
        Scanner in = new Scanner(System.in);
        System.out.print("Enter a number: ");
        num = in.nextInt();
        while (num > 0)
        {
            sum = sum + (num % 10);
            num = num / 10;
        }
        System.out.println("The sum of the digits is " + sum);
    }// main
}//DigitSum
```

ב.

```
/*
קלט: מספר חיובי שלם
פלט: מספר הספרות האי זוגיות במספר הנתון
*/
import java.util.Scanner;
public class DigitOddCount
{
    public static void main (String [] args)
    {
        int oddCount = 0;
        int num;
        Scanner in = new Scanner(System.in);
        System.out.print("Enter a number: ");
        num = in.nextInt();
        while (num > 0)
        {
            int digit = num % 10;
            if ( digit % 2 == 1)
                  oddCount++;
            num = num / 10;
        }
        System.out.println("The number of odd digits is " + oddCount);
    }// main
}//DigitOddCount
```

ג.

```
/*
קלט: מספר חיובי שלם
פלט: מכפלת ספרות המספר הנתון
*/
import java.util.Scanner;
public class DigitMult
{
    public static void main (String [] args)
    {
        int mult = 1;
        int num;
        Scanner in = new Scanner(System.in);
        System.out.print("Enter a number: ");
        num = in.nextInt();
        while (num > 0)
        {
            mult = mult * (num % 10);
            num = num / 10;
        }
        System.out.println("The multiply of the digits is " + mult);
    }// main
}//DigitMult
```

שאלה 7.41

נתון קטע התוכנית החלקי הבא, אשר הקלט שלו הוא שני מספרים שלמים חיוביים x ו-y ומטרתו היא הצגת שארית החלוקה בשלמים של x ב-y (כלומר תוצאת החישוב x%y). בקטע התוכנית מתבצע החישוב הדרוש באמצעות פעולת **חיסור**.
השלימו את קטע התוכנית.

```
System.out.print("Enter a number: ");
x = in.nextInt();
System.out.print("Enter a number: ");
y = in.nextInt();
while (_______________)
    x = x - y;
System.out.println(_____________);
```

תשובה 7.41

```
System.out.print("Enter a number: ");
x = in.nextInt();
System.out.print("Enter a number: ");
y = in.nextInt();
while (x >= y)
    x = x - y;
System.out.println(x + " % " +  y + " = " + x );
```

שאלה 7.42

שנו את הקטע הנתון בשאלה הקודמת כך שיציג גם את מנת החלוקה של x ב-y, (כלומר את תוצאת החישוב x/y). יש לבצע את החישובים הדרושים באמצעות פעולות חיבור וחיסור בלבד!

תשובה 7.42

```
System.out.print("Enter a number: ");
x = in.nextInt();
System.out.print("Enter a number: ");
y = in.nextInt();
count = 0;
while (x >= y)
{
    x = x - y;
    count++;
}
System.out.println(x + " / " +  y + " = " + count );
System.out.println(x + " % " +  y + " = " + x );
```

שאלה 7.43

שני תלמידים המשחקים זה נגד זה מותחים בתחילת המשחק קו באורך N סנטימטרים (N > 1). השחקנים מחליפים תורות לסירוגין. כל שחקן מקצר בתורו את הקו לחצי מאורכו. השחקן אשר מקצר בתורו את הקו לאורך של פחות מסנטימטר אחד מנצח במשחק.

למשל אם אורכו של הקו הוא 8 ס״מ, השחקן הראשון יקצר את הקו ל-4 ס״מ והשחקן השני יקצר את הקו ל-2 ס״מ, השחקן הראשון יקצר את הקו לס״מ אחד והשחקן השני יקצר את הקו ל-0.5 ס״מ וינצח במשחק.

פתחו וישמו אלגוריתם אשר הקלט שלו הוא אורכו ההתחלתי של הקו, והפלט שלו הוא הודעה מיהו השחקן המנצח (הראשון או השני).

תשובה 7.43

רעיון הפתרון:

לפתרון שאלה זו אנו נדרשים לחלק את הערך ב-2 עד אשר ערך התוצאה קטן מ-1. לשם כך מתאים להשתמש בתבנית **צבירה**, כך שערך הצובר יחולק ב-2 בכל ביצוע חוזר. בנוסף, עלינו לקבוע מי היה השחקן אשר ביצע את הפעולה האחרונה. לשם כך נשמור במשתנה נוסף את מספר השחקן המבצע את הפעולה, ובכל ביצוע חוזר נחליף את הערך של משתנה זה.

ניתוח הבעיה באמצעות דוגמאות:

- קלט: 3 פלט: player number 2 wins
- קלט: 18 פלט: player number 1 wins

תבניות:

צבירה

בחירת משתנים:

length– ממשי, צובר החילוק

player – שלם, מספר השחקן

יישום האלגוריתם:

```
/* קלט: אורך קו
/* פלט: מספר השחקן אשר יקצר את הקו לאורך הקו מ-1 ס"מ */
import java.util.Scanner;
public class LineLength
{
      public static void main(String[] args)
      {
            double length;
            int player = 0;   // will be changed to 1 in first loop
            Scanner in = new Scanner(System.in);
            System.out.print("Enter line length: ");
            length = in.nextDouble();
            while (length >= 1)
            {
                  length = length / 2;
                  player = 3 - player; // change player number
            }
            System.out.println("The winner is number: " + player);
      }// main
}// LineLength
```

שאלה 7.44

במשחק אסימונים שחקן מניח 2 אסימונים בתור הראשון, 4 אסימונים בתור השני, 8 אסימונים בתור השלישי, וכך הלאה – בכל תור מוכפל מספר האסימונים.

נתון קטע התוכנית הבא אשר הקלט שלו הוא מספר האסימונים ההתחלתי של השחקן, והפלט שלו צריך להיות המספר הסידורי של התור אשר בו לא ניתן להמשיך לשחק לפי השיטה המתוארת. למשל, עבור הקלט המייצג מספר אסימונים התחלתי 9, הפלט הדרוש הוא 3, כיוון שאחרי שהניח 2 אסימונים בתור הראשון ו-4 אסימונים נוספים בתור השני יוותרו לשחקן רק 3 אסימונים (ולא 8 אסימונים, כפי שנדרש לתור השלישי). קטע התוכנית שגוי.

```
int turn = 0;                  //המספר הסידורי של התור הנוכחי
int tokensInCurrentTurn = 0;   //מספר אסימונים למשחק בתור הנוכחי
int totalPlayedTokens = 0;  //סכום אסימונים כולל בתורות שהתבצעו עד כה
int startTokens;               //מספר אסימונים התחלתי
System.out.print("Enter number of tokens to start with: ");
startTokens = in.nextInt();
while (totalPlayedTokens <= startTokens)
{
    turn = turn + 1;
    tokensInCurrentTurn = tokensInCurrentTurn * 2;
    totalPlayedTokens = totalPlayedTokens + tokensInCurrentTurn;
}
System.out.println("The game cannot go on after " + turn + "turns");
```

ציינו מהי השגיאה ותקנו אותה.

תשובה 7.44

השגיאה היא שצובר המכפלה מאותחל ל-0, ולכן ישאר תמיד 0. יש לאתחלו ל-1.

```
int tokensInCurrentTurn = 1;  //מספר אסימונים למשחק בתור הנוכחי
```

שאלה 7.46

כתבו לולאה להצגת פלט של 50 כוכביות.

א. כתבו לולאת **for** להצגת הפלט הדרוש.

ב. כתבו לולאת **while** להצגת הפלט הדרוש.

ג. איזה מן הפתרונות פשוט יותר?

תשובה 7.46

א.

```
for (int i=1; i<=50; i++)
      System.out.print("*");
```

ב.

```
int i=1;
while (i<=50)
{
      System.out.print("*");
      i++
}
```

ג. פתרון א' פשוט יותר. מקובל להשתמש בלולאת **for** כאשר מספר הביצועים החוזרים ידוע מראש.

שאלה 7.47

א. כתבו לולאת **for** המציגה כפלט את 20 הכפולות החיוביות הראשונות של 5 (כלומר הפלט הוא: 5 10 15 ... 100).

ב. כתבו לולאת **for** נוספת המבצעת אותו דבר אבל בעלת כותרת שונה וגוף לולאה שונה.

ג. כתבו לולאת **while** המבצעת אותו דבר.

תשובה 7.47

א.

```
for (int i=1; i<=20; i++)
      System.out.print(i*5 + " ");
```

ב.

```
for (int i=5; i<=100; i = i + 5)
      System.out.print(i + " ");
```

ג.

```
int i=5;
while (i<=100)
{
      System.out.print(i + " ");
      i += 5;
}
```

שאלה 7.48

ציינו עבור כל אחת מן הבעיות הבאות אם לפתרונה מתאים יותר להשתמש בלולאת **for** או בלולאת **while**. נמקו את תשובותיכם.

א. הקלט: מספר שלם חיובי num. הפלט: רשימת המספרים השלמים **החיוביים** מ-1 עד num.

ב. הקלט: מספר שלם חיובי num. הפלט: רשימת המספרים השלמים **השליליים** מ-1- עד num-.

ג. הקלט: מספר שלם חיובי num. הפלט: רשימת המספרים השלמים החיוביים מ-1 עד המספר הקטן ביותר k אשר עבורו מתקיים $1+2+3+...+k > num$.

תשובה 7.48

א. לולאת **for** מתאימה יותר, כיוון שאפשר לחשב לפני ביצוע לולאה את מספר הפעמים שתתבצע. במקרה זה, מספר הפעמים הוא num.

ב. לולאת **for** מתאימה יותר, כיוון שגם כאן אפשר לחשב לפני ביצוע לולאה את מספר הפעמים שתתבצע. גם במקרה זה, מספר הפעמים הוא num.

ג. לולאת **while** מתאימה יותר, כיוון שאי אפשר לחשב לפני ביצוע לולאה את מספר הפעמים שתתבצע. הסיום נקבע באמצעות תנאי הבודק את גודל סכום סדרה לעומת ערך המשתנה num.

שאלה 7.49

עבור כל אחת מן הבעיות הבאות ציינו אם לפתרונה מתאים יותר להשתמש בלולאת `for` או בלולאת `while`. נמקו בקצרה את תשובותיכם.

שימו ♥: בעיות אלו מתאימות לתבניות מקסימום, מינימום, מקום המקסימום ומקום המינימום, אך לא תמיד גודל התחום שמופעלת בו התבנית ידוע מראש.

א. הקלט הוא סדרת אותיות מהאלף-בית ובסופה '*', והפלט הוא האות הגדולה ביותר שמופיעה בקלט.

ב. הקלט הוא מספר שלם חיובי המציין אורך סדרה ואחריו סדרת אותיות מהאלף-בית באורך המצוין, והפלט הוא האות הקטנה ביותר שמופיעה בקלט.

ג. הקלט הוא כמו הקלט לסעיף ב, והפלט הוא המקום הסידורי של ההופעה הראשונה (אולי יש יותר מאחת) של האות הגדולה ביותר שמופיעה בקלט.

ד. הקלט הוא כמו הקלט לסעיף ב, והפלט הוא המקום הסידורי של ההופעה האחרונה (אולי יש יותר מאחת) של האות הקטנה ביותר שמופיעה בקלט.

ה. מהם ההבדלים ביישום האלגוריתם בין סעיף ג ל-ד?

תשובה 7.49

א. לולאת **while** מתאימה יותר, כיוון שלא ניתן לדעת את מספר הפעמים שתתבצע הלולאה. לאחר כל ביצוע חוזר של הלולאה יש לבדוק האם התו הנקלט הוא '*'.

ב. לולאת **for** מתאימה יותר, כיוון שמיד לאחר קליטת נתון הקלט הראשון מספר הפעמים שתתבצע הלולאה נתון.

ג. לולאת **for** מתאימה יותר, כיוון שמיד לאחר קליטת נתון הקלט הראשון מספר הפעמים שתתבצע הלולאה נתון.

ד. לולאת **for** מתאימה יותר, כיוון שמיד לאחר קליטת נתון הקלט הראשון מספר הפעמים שתתבצע הלולאה נתון.

ה. בסעיף ג' אנו מחפשים את המקום הסידורי של האות הגדולה ביותר ולכן נשתמש בתבנית מציאת ערך נלווה למקסימום.

בסעיף ד' אנו מחפשים את המקום הסידורי של האות הקטנה ביותר ולכן נשתמש בתבנית מציאת ערך נלווה למינימום.
בסעיף ג' אנו מחפשים את המקום הסידורי של ההופעה הראשונה, ולכן נחליף את משתנה המקסימום רק אם מצאנו ערך גדול לערך משתנה המקסימום.
בסעיף ד' אנו מחפשים את המקום הסידורי של ההופעה האחרונה, ולכן נחליף את משתנה המינימום רק אם מצאנו ערך קטן **או שווה** לערך משתנה המינימום.

שאלה 7.50

פתחו אלגוריתם אשר הקלט שלו הוא סדרת אותיות מן האלף-בית האנגלי שמסתיימת ב-'*', והפלט שלו הוא האות **הגדולה** ביותר מבין האותיות B עד I המופיעות בקלט.
למשל עבור הקלט *SCHOOL הפלט המתאים הוא H. אמנם יש בקלט אותיות גדולות מ-H, למשל S, אך אותיות אלו לא כלולות בסדרת האותיות B עד I.
הניחו שבקלט מופיעה לפחות אות אחת בתחום B עד I.
ישמו את האלגוריתם בשפת Java.

תשובה 7.50

רעיון הפתרון:
הפתרון לשאלה זו מבוסס על התבנית **מציאת מקסימום**, כאשר אנו מגבילים את טווח הערכים הנבדקים עבור משתנה המקסימום, על פי דרישת השאלה. את משתנה המקסימום מומלץ לאתחל לערך-'A' מכיוון שהערך המקסימלי הנדרש הינו בטווח הערכים 'B' עד 'I'. באופן זה בוודאות תמצא אות הגדולה ממנו אשר תוצב במשתנה המקסימום. הקלט יתבצע עד אשר יקלט התו '*' ולכן נשתמש בלולאת זקיף.

ניתוח הבעיה באמצעות דוגמאות:

- קלט: LETS GO HOME* פלט: H
- קלט: TO BE OR NOT TO BE* פלט: E

תבניות:
מקסימום

בחירת משתנים:
ch – תו, התו התורן ברשימת תווי הקלט
max– תו, לשמירת התו הגדול ביותר מבין אלה שנקלטו ונמצאים בטווח שבין B ל- I

יישום האלגוריתם:

```
/* קלט: רשימת תווים המסתיימת ב-*
פלט: התו הגדול ביותר מבין אלה שנקלטו ונמצאים בטווח שבין B ל-I
*/
import java.util.Scanner;
public class BiggestChar
{
	public static void main(String[] args)
	{
		char ch;
		char max = 'A';
		Scanner in = new Scanner(System.in);
		System.out.print("Enter a character: ");
		ch = in.next().charAt(0);
		while (ch != '*')
		{
			if (ch >= 'B' && ch <= 'I' && ch > max)
				max = ch;
			System.out.print("Enter next character: ");
			ch = in.next().charAt(0);
		}
		System.out.println(max);
	}// main
}// BiggestChar
```

שאלה 7.51

תקנו את התוכנית `OddNumbers` כך שתסתיים לאחר הצגת כל המספרים האי-זוגיים החיוביים שקטנים מן המספר הנתון.

תשובה 7.51

```
/*
קלט: מספר חיובי שלם
פלט: כל המספרים האי-זוגיים החיוביים הקטנים מן המספר הנתון
*/
import java.util.Scanner;
public class OddNumbers
{
     public static void main (String[] args)
     {
            int limit; // המספר הנתון
            int oddN = 1; //המספר האי-זוגי החיובי הראשון
            Scanner in = new Scanner(System.in);
            System.out.print("Enter a number");
            limit = in.nextInt();
            System.out.println("The odd numbers that are smaller " +
                                        "than the given number are: ");
            while (oddN < limit)
            {
                  System.out.print(oddN + " ");
                  oddN = oddN + 2;
            }
      } //main
}// OddNumbers
```

שאלה 7.52

ציינו עבור כל אחת מן הלולאות הבאות אם היא לולאה אינסופית.

אם הלולאה סופית ציינו את מספר הפעמים שהיא תתבצע. היעזרו בטבלת מעקב לביצוע החישובים הדרושים.

א.
```
int i = 0;
while (i < 30)
    i = i + 4;
```

ב.
```
int j = 0;
while (j < 50)
    j = j - 10;
```

ג.
```
int j = 0;
while (Math.abs(j) < 30)
    j = j - 10;
```

ד.
```
int k = 0;
for (int j = 1; j < 10; j++)
    k = k + 11;
```

ה. num הוא מספר שלם חיובי כלשהו
```
while (num != 10)
{
    System.out.println(num);
    num = num + 1;
}
```

ו. num הוא מספר שלם חיובי כלשהו
```
while (num != 0)
{
    System.out.println(num);
    num = num - 1;
}
```

תשובה 7.52

א. לולאה סופית, תתבצע 8 פעמים.

ב. לולאה אינסופית, כיוון שערכו של j הולך וקטן.

ג. לולאה סופית, כיוון שערכו המוחלט של j הולך וגדל. תתבצע 3 פעמים.

ד. לולאה סופית, תתבצע 10 פעמים.

ה. לולאה אינסופית, תתבצע אינסוף פעמים עבור כל קלט גדול מ-10. (מומלץ לדון בכיתה גם על

המקרים בהם הקלט הוא בין 1 ל-10, שבהם הלולאה מתבצעת num-10 פעמים).

ו. לולאה סופית, תתבצע num פעמים.

שאלה 7.53

נתון קטע התוכנית הבא אשר הקלט שלו הוא מספר שלם חיובי:

```
System.out.print("Enter a number: ");
num = in.nextInt();
while (num != 0)
{
    num = num % 3;
    System.out.println(num);
}
```

הלולאה בקטע התוכנית תתבצע מספר סופי של פעמים עבור חלק מן הקלטים, ומספר אינסופי של פעמים עבור שאר הקלטים. לכן הלולאה שבקטע התוכנית היא לולאה אינסופית.

א. תנו דוגמת קלט שעבורה תתבצע הלולאה מספר סופי של פעמים.

מה מאפיין את הקלטים שעבורם תתבצע הלולאה מספר סופי של פעמים?

כמה פעמים תתבצע הלולאה עבור קלטים אלה?

ב. תנו דוגמת קלט שעבורה תתבצע הלולאה מספר אינסופי של פעמים.

מה מאפיין את הקלטים שעבורם תתבצע הלולאה מספר אינסופי של פעמים?

תשובה 7.53

א. 6. הלולאה תתבצע בדיוק פעם אחת עבור כל קלט שהוא כפולה של 3.

ב. 4. הלולאה תתבצע אינסוף פעמים עבור כל קלט שאיננו כפולה של 3.

שאלה 7.54

נתון קטע התוכנית הבא אשר הקלט שלו הוא שני מספרים שלמים:

```
System.out.print("Enter a number: ");
x = in.nextInt();
System.out.print("Enter a number: ");
y = in.nextInt();
while (x != y)
{
    y = y - 1;
    x = x + 1;
}
System.out.println(x);
```

א. תנו שתי דוגמאות קלט שונות שעבורן **לא יתבצע** גוף הלולאה.

ב. תנו שתי דוגמאות קלט שונות שעבורן יתבצע גוף הלולאה **בדיוק פעם אחת**.

ג. תנו שתי דוגמאות קלט שונות שעבורן יתבצע גוף הלולאה **בדיוק חמש פעמים**.

ד. תנו שתי דוגמאות קלט שונות שעבורן יתבצע גוף הלולאה **אינסוף** פעמים.

ה. נניח שערכו של x הוא 0. עבור אילו ערכים של y יתבצע גוף הלולאה **אינסוף** פעמים?

תשובה 7.54

א. 5 5 ; 8 8

ב. 1 3 ; 2 4

ג. 1 11 ; 2 12

ד. 1 2 ; 2 1

ה. עבור כל מספר שלילי, או עבור כל מספר חיובי אי-זוגי.

שאלה 7.56

בקטע התוכנית הבא המשתנה length שומר מספר חיובי שלם המבטא אורך רשימה. הקטע קולט רשימת תוצאות הטלה של קובייה (תוצאת הטלה של קובייה היא מספר שלם בין 1 ל-6). רשימת התוצאות עלולה לכלול מספרים שגויים (כלומר שאינם בין 1 ל-6). המשתנה valid הוא מטיפוס בוליאני ושאר המשתנים הם מטיפוס שלם.

```
i = 1;
s = 0;
valid = true;
while(valid && (i <= length))
{
    System.out.print("Enter the number on the die: ");
    die = in.nextInt();
    if((die >= 1) && (die <= 6))
        s = s + die;
    else
        valid = false;
    i = i + 1;
} // while
if (valid)
    System.out.println(s);
else
    System.out.println(die);
```

א. תארו את הפלט עבור הקלט: 3 2 1 3.

ב. תארו את הפלט עבור הקלט: 3 0 1 3.

ג. מהו תפקיד המשתנה הבוליאני valid?

ד. מהי מטרת קטע התוכנית?

תשובה 7.56

א. הפלט הוא סכום ארבע ההטלות שנקלטו: 9.

ב. הפלט הוא הטלה שאינה חוקית: 0. שימו לב שהקלט האחרון (3) כלל לא נקלט במהלך ריצת התוכנית.

ג. תפקיד המשתנה הבוליאני valid הוא לעצור את מהלך ריצת התוכנית לאחר קלט שאינו תקין.

ד. מטרת קטע התוכנית: להציג את סכום ההטלות, אם כל הערכים שנקלטו הינם חוקיים. במידה ונקלט ערך לא חוקי, תפסק מהלך ריצת התוכנית ויוצג הערך השגוי שהפסיק את ריצתה.

שאלה 7.57

פתחו אלגוריתם שהקלט שלו הוא רשימה של 20 מספרים חיוביים. הקלט נחשב חוקי אם **כל** המספרים בו הם זוגיים. האלגוריתם בודק אם הקלט חוקי. אם הקלט חוקי האלגוריתם מציג כפלט את סכום המספרים שבקלט, ואחרת תוצג הודעה כי הקלט איננו חוקי. ישמו את האלגוריתם בשפת Java.

הדרכה: השתמשו במשתנה בוליאני שיאותחל ב-**true**. אם יימצא בקלט מספר אי-זוגי יושם במשתנה זה הערך **false**.

תשובה 7.57

רעיון הפתרון:

לפתרון שאלה זו מומלץ להשתמש בתבנית **האם כל הערכים בסדרה מקיימים תנאי?** על מנת לבדוק אם אכן כל הערכים זוגיים ויש להציג את סכומם כפלט. אם ישנו ערך אחד לפחות שאינו זוגי, יש להציג הודעה כי הקלט אינו חוקי.

תבניות:

האם כל הערכים בסדרה מקיימים תנאי?

צבירה

בחירת משתנים:

num – שלם, המספר התורן ברשימת מספרי הקלט

sum– שלם, צובר לסכימת מספרי הקלט

count – שלם, מונה למניית מספרי הקלט

valid – בוליאני, שומר האם המספר הנקלט הוא חוקי

האלגוריתם:

1. *אתחל את* sum *ל-*0
2. *אתחל את* count *ל-*0
3. *אתחל את* valid *ל-*true
4. *כל עוד* count<20 *ו*valid *שווה ל-*true
   4.1. *קלוט מספר שמור ב-*num
   4.2. *הגדל את* count *ב-*1
   4.3. *אם* num *זוגי*
      4.3.1. *הוסף את ערכו של* num *ל-*sum
   4.4. *אחרת*
      4.4.1. *השם ב-*valid *את הערך* false
5. *אם* valid *שווה ל-*true
   5.1. *הדפס כפלט את* sum
6. *אחרת*
   6.1. *הדפס כפלט "הקלט אינו תקין"*

יישום האלגוריתם:

```
/* קלט: רשימת תווים המסתיימת ב-*
פלט: האם הקלט מהווה סיסמה חוקית
*/
import java.util.Scanner;
public class ValidNumbers
{
	public static void main(String[] args)
	{
		int num;
		int sum = 0;
		int count = 0;
		boolean valid = true;
		Scanner in = new Scanner(System.in);
		System.out.print("Enter a number: ");
		while(count < 20 && valid)
		{
			num = in.nextInt();
			count++;
			if (num % 2 == 0)
				sum += num;
			else
				valid = false;
		}
		if (valid)
			System.out.println("The sum is: " + sum);
		else
			System.out.println("Illegal number");
	}// main
}// ValidNumbers
```

שאלה 7.58

פתחו אלגוריתם שהקלט שלו הוא סיסמה, המורכבת מרשימת אותיות לועזיות המסתיימת ב-'*'. הפלט הוא הודעה המציינת אם הסיסמה תקינה. סיסמה תקינה כוללת לפחות 6 תווים, ובהם לפחות אות אחת קטנה (בתחום a..z) ולפחות אות אחת גדולה (בתחום A..Z). ישמו את האלגוריתם בשפת Java.

תשובה 7.58

רעיון הפתרון:

על מנת לקבוע האם הסיסמה תקינה עלינו לבדוק האם מתקיימים שלושת התנאים: האם הסיסמה בעלת 6 תווים לפחות, האם היא כולל אות לועזית קטנה והאם היא כוללת אות לועזית גדולה. לבדיקת מספר תווי הסיסמה נשתמש בתבנית **מניה**. לבדיקה האם הסיסמה כוללת אות לועזית קטנה ואות לועזית גדולה נשתמש בשני משתנים בוליאניים. משתנה בוליאני אחד ישמור האם נקלטה אות לועזית קטנה עד כה, משתנה בוליאני שני ישמור האם נקלטה אות לועזית גדולה עד כה. כל אחד ממשתנים אלה יאותחל ל-false, וישתנה ל-true אם התנאי המתאים עבורו התקיים. בסיום הקלט נדע שהסיסמה תקינה בתנאי שהתקיימו שלושת התנאים.

ניתוח הבעיה באמצעות דוגמאות:

- קלט: Password* פלט: Valid password
- קלט: password* פלט: Not valid password
- קלט: Pass* פלט: Not valid password

תבניות:

מנייה

בחירת משתנים:

ch – תו, התו התורן ברשימת תווי הקלט

count– שלם, מונה למניית תווי הקלט

hasSmallLetter – בוליאני, שומר האם נקלטה אות לועזית קטנה

hasCapitlLetter – בוליאני, שומר האם נקלטה אות לועזית גדולה

יישום האלגוריתם:

```
/* קלט: רשימת תווים המסתיימת ב-*
פלט: האם הקלט מהווה סיסמה חוקית
*/
import java.util.Scanner;
public class ValidPassword
{
	public static void main(String[] args)
	{
		char ch;
		int count = 0;
		boolean hasSmallLetter = false;
		boolean hasCapitalLetter = false;
		Scanner in = new Scanner(System.in);
		System.out.print("Enter a character: ");
		ch = in.next().charAt(0);
		while (ch != '*')
		{
			count++;
			if (ch >= 'a' && ch <= 'z')
				hasSmallLetter = true;
			else if (ch >= 'A' && ch <= 'Z')
				hasCapitalLetter = true;
			System.out.print("Enter next character: ");
			ch = in.next().charAt(0);
		}
		if (count>=6 && hasCapitalLetter && hasSmallLetter)
			System.out.println("Valid password");
		else
			System.out.println("Not valid password");
	}// main
}// ValidPassword
```

שאלה 7.59

בקטע התוכנית הבא length מכיל מספר שלם חיובי. הקלט לקטע התוכנית הוא סדרה באורך length של מילים בנות שלוש אותיות כל אחת.

מטרת קטע התוכנית היא להציג כל מילה בסדרה בסדר אותיות הפוך כל עוד אותיות המילה שונות זו מזו. ברגע שנקלטת מילה שבה לפחות שתי אותיות זהות יש לעצור את מהלך הביצוע.

המשתנה diff הוא מטיפוס **בוליאני**, המשתנים length ו-i מטיפוס שלם והמשתנים let1, let2 ו-let3 הם מטיפוס תווי.

```
diff = ___________
i = 1;
while(___________________)
{
    System.out.print("Enter the first letter of the word: ");
    let1 = in.next().charAt(0);
    System.out.print("Enter the second letter of the word: ");
    let2 = in.next().charAt(0);
    System.out.print("Enter the third letter of the word: ");
    let3 = in.next().charAt(0);
    if (_________________)
        System.out.println(let3 + " " + let2 + " " + let1);
    else
        diff = _________
} // while
```

השלימו את קטע התוכנית.

תשובה 7.59

```
diff = true;___________
i = 1;
while(i <= length && diff)
{
    System.out.print("Enter the first letter of the word: ");
    let1 = in.next().charAt(0);
    System.out.print("Enter the second letter of the word: ");
    let2 = in.next().charAt(0);
    System.out.print("Enter the third letter of the word: ");
    let3 = in.next().charAt(0);
    if (let1 != let2 && let2 != let3 && let1 != let3)
        System.out.println(let3 + " " + let2 + " " + let1);
    else
        diff = false;
} // while
```

שאלה 7.60

נניח שהמשתנים a ו-b הם מטיפוס שלם. עבור כל ביטוי מן הביטויים הבוליאניים הבאים, תנו דוגמה לערכים למשתנים שעבורם יהיה ערך הביטוי **true**, ודוגמה לערכי המשתנים שעבורם יהיה ערך הביטוי **false**.

	ערך הביטוי **true**	ערך הביטוי **false**
א. `!(a != b)`	a = b =	a = b =
ב. `!(Math.abs(a)== a)`	a =	a =

תשובה 7.60

	ערך הביטוי **true**	ערך הביטוי **false**
א. `!(a != b)`	a = 8 b = 8	a = 7 b = 8
ב. `!(Math.abs(a)== a)`	a = -8	a = 8

שאלה 7.61

נניח שהמשתנים a ו-b הם מטיפוס שלם והמשתנה c הוא מטיפוס בוליאני. עבור כל ביטוי מן הביטויים שבמשפטי ההשמה הבאים, תנו דוגמה לערכי המשתנים שעבורם יושם ב-c הערך **true**, ודוגמה לערכי המשתנים שעבורם יושם ב-c הערך **false**.

	ערך c הוא **true**	ערך c הוא **false**
א. `c =(x == Math.sqrt(x))`	x =	x =
ב. `c =(c && (a == b))`	a = b = c =	a = b = c =
ג. `c =(!((a + b)>= 5))`	a = b =	a = b =

תשובה 7.61

	ערך c הוא **true**	ערך c הוא **false**
א. `c =(x == Math.sqrt(x))`	x = 1	x = 4
ב. `c =(c && (a == b))`	a = 4 b = 4 c = true	a = 3 b = 4 c= true
ג. `c =(!((a + b)>= 5))`	a = 2 b = 3	a = 3 b = 4

שאלה 7.62

במשחק הניחושים מגריל המחשב מספר בתחום 1-100, והשחקן צריך לנחש אותו. בתום המשחק יודיע המחשב לשחקן כמה ניסיונות ניסה עד שהצליח לנחש.

לפניכם קטע תוכנית ב-Java. השלימו את הקטע כך שיבצע את הנדרש:

```
Random rnd = new Random();
int num = _____________________;       //המספר שהמחשב יגריל
int numOfGuesses = _____________;          //   מונה   לספירת   מספר
הניחושים
boolean found = ____________;
while (!found)
{
    numOfGuesses++;
    System.out.print("Enter your guess: ");
    guess = in.nextInt();
    if (_____________)
    {
        System.out.println("Correct!! ");
        found = true;
    }
    else if (_______________)
        System.out.println("Your guess is too big");
    else
        System.out.println("Your guess is too small");
} // while
System.out.println("It took you " + _________________ + " guesses");
```

תשובה 7.62

```
Random rnd = new Random();
int num = rnd.nextInt(100) + 1;     // הגרלת המספר
int numOfGuesses = 0;               // מונה לספירת מספר הניחושים
boolean found = false;
while (!found)
{
    numOfGuesses++;
    System.out.print("Enter your guess: ");
    guess = in.nextInt();
    if ( guess == num )
    {
        System.out.println("Correct!! ");
        found = true;
    }
    else if ( guess > num )
        System.out.println("Your guess is too big");
    else
        System.out.println("Your guess is too small");
} // while
System.out.println("It took you " + numOfGuesses + " guesses");
```

שאלה 7.65

נניח שבאלגוריתם דומה לאלגוריתם המתואר בבעיה 12, משתנה הבקרה של הלולאה החיצונית i יתחיל מהערך 2, ויגדל בכל ביצוע של הלולאה ב-1, עד הגיעו לערך 6. מה יהיה הפלט של אלגוריתם זה? ומה יהיה מספר הפעמים הכולל שיתבצע גוף הלולאה הפנימית שבו?

תשובה 7.65

הפלט של אלגוריתם זה יהיה שורות מספר 2 עד 6 של לוח הכפל:

2 4 6 20
3 6 9 30
4 8 12 40
5 10 15 50
6 12 18..... 60

גוף הלולאה הפנימית יתבצע 10 פעמים בכל אחד מ-5 הביצועים של הלולאה החיצונית. סה"כ 50 פעמים.

שאלה 7.66

תארו עבור כל אחד מקטעי התוכניות הבאים את פלט קטע התוכנית ואת מספר הפעמים הכולל שמתבצע גוף הלולאה הפנימית:

א.
```
for (int i = 1; i <= 5; i++)
{
    for (int j = 1; j <= 5; j++)
        System.out.print("*");
    System.out.println();
}
```

ב.
```
for (int i = 1; i <= 5; i++)
{
    for (int j = 1; j <= i; j++)
        System.out.print("*");
    System.out.println();
}
```

ג.
```
for (int i = 1; i <= 5; i++)
{
    for (int j = 1; j <= 5; j++)
        System.out.print(j);
    System.out.println();
}
```

ד.
```
for (int i = 1; i <= 5; i++)
{
    for (int j = 5; j >= i; j--)
        System.out.print("*");
    System.out.println();
}
```

תשובה 7.66

א.

```
*****
*****
*****
*****
*****
```

גוף הלולאה הפנימית יתבצע 25 פעמים.

ב.

```
*
**
***
****
*****
```

גוף הלולאה הפנימית יתבצע 15 פעמים.

ג.

```
12345
12345
12345
12345
12345
```

גוף הלולאה הפנימית יתבצע 25 פעמים.

ד.

```
*****
****
***
**
*
```

גוף הלולאה הפנימית יתבצע 15 פעמים.

שאלה 7.67

עבור כל אחד מהסעיפים הבאים פתחו וישמו אלגוריתם אשר הפלט שלו הוא כפי שמופיע בתיאור הסעיף. בכל אחד מהאלגוריתמים מותר לכתוב **רק הוראה אחת** המדפיסה תו או ספרה ו**הוראה אחת** למעבר שורה בפלט (מכאן נובע כי עליכם להשתמש בלולאות מקוננות):

א.

```
*
**
***
****
*****
```

ב.

```
*****
****
***
**
*
```

ג.

```
*****
****
***
**
*
**
***
****
*****
```

ד.

```
1
12
123
1234
12345
```

ה.

```
1
22
333
4444
55555
```

ו.

```
12345
1234
123
12
1
```

ז.

```
55555
4444
333
22
1
```

ח.

```
11111
2222
333
44
5
```

תשובה 7.67

א.
```
for (int i = 1; i <= 5; i++)
{
    for (int j = 1; j <= i; j++)
        System.out.print("*");
    System.out.println();
  }
```

ב.
```
for (int i = 1; i <= 5; i++)
{
    for (int j = 5; j >= i; j--)
        System.out.print("*");
    System.out.println();
  }
```

ג.
```
int k = i;
for (int i = 1; i < 10; i++)
{
      if (i>5)
            k=10-i;
      for (int j=5; j >= k; j--)
           System.out.print("*");
      System.out.println();
}
```

ד.
```
for (int i = 1; i <= 5; i++)
{
      for (int j=1; j <= i; j++)
            System.out.print(j);
      System.out.println();
}
```

ה.
```
for (int i = 1; i <= 5; i++)
{
      for (int j=1; j <= i; j++)
            System.out.print(i);
      System.out.println();
}
```

ו.
```
for (int i = 5; i >= 1; i--)
{
    for (int j = 1; j <= i; j++)
        System.out.print(j);
    System.out.println();
}
```

ז.
```
for (int i = 5; i >= 1; i--)
{
    for (int j = 1; j <= i; j++)
        System.out.print(i);
    System.out.println();
}
```

ח.
```
for (int i = 1; i <= 5; i++)
{
    for (int j = 5; j >= i; j--)
        System.out.print(i);
    System.out.println();
}
```

שאלה 7.68

בכיתה 40 תלמידים, כל תלמיד לומד 20 מקצועות. לפניכם תוכנית המחשבת עבור כל תלמיד את ממוצע ציוניו:

```
/*
קלט: 20 הציונים עבור כל אחד מ-40 תלמידי הכיתה
פלט: הממוצע של כל תלמיד ותלמיד
*/
import java.util.Scanner;
public class Grades
{
    public static void main (String[] args)
    {
        final int STUDENT_NUM = 40;
        final int GRADE_NUM = 20;
        double grade;
        double sum;
        double average;
        Scanner in = new Scanner(System.in);
        for(int i = 1; i <= STUDENT_NUM; i++)
        {
            for(int j = 1; j <= GRADE_NUM; j++)
            {
              System.out.print("Enter next student grade: ");
                grade = in.nextInt();
                sum = sum + grade;
            } // for j
        } // for i
    } // main
} // Grades
```

המשפטים הבאים חסרים בתוכנית הנתונה, היכן צריך לשלב את המשפטים האלה כדי שהתוכנית תבצע את הנדרש?

א. `sum = 0;`

ב. `System.out.println(sum / 20);`

תשובה 7.68

א. את הצבירה אנו מבצעים עבור כל תלמיד בנפרד, ולכן יש לאפס את הצובר **לפני** תחילת הזנת הציונים עבור כל תלמיד. לכן, את משפט זה יש לשלב בתוך הלולאה החיצונית, לפני תחילת הביצוע של הלולאה הפנימית.

ב. את הצגת הממוצע אנו מבצעים עבור כל תלמיד בנפרד, לאחר שסיימנו לקלוט ולצבור את כל ציוניו. לכן, את משפט זה יש לשלב בתוך הלולאה החיצונית, לאחר סיום הביצוע של הלולאה הפנימית.

להלן התוכנית המלאה:

```
/*
קלט: 20 הציונים עבור כל אחד מ-40 תלמידי הכיתה
פלט: הממוצע של כל תלמיד ותלמיד
*/
import java.util.Scanner;
public class Grades
{
    public static void main (String[] args)
    {
        final int STUDENT_NUM = 40;
        final int GRADE_NUM = 20;
        double grade;
        double sum;
        double average;
        Scanner in = new Scanner(System.in);
        for(int i = 1; i <= STUDENT_NUM; i++)
        {
            sum = 0;
            for(int j = 1; j <= GRADE_NUM; j++)
            {
                System.out.print("Enter next student grade: ");
                grade = in.nextInt();
                sum = sum + grade;
            } // for j
            System.out.println(sum / 20);
        } // for i
    } // main
} // Grades
```

שאלה 7.70

במפעל לייצור נעליים יש עובדים רבים. פתחו אלגוריתם שיקבל כקלט את מספר העובדים במפעל, ואחר כך רשימה של כל המשכורות של השנה האחרונה (12 משכורות) **לכל אחד** מעובדי המפעל. האלגוריתם יחשב וידפיס את **המשכורת האחרונה** של העובד המסכן שסכום משכורותיו כל השנה היה הנמוך ביותר. ישמו את האלגוריתם בשפת Java.

תשובה 7.70

רעיון הפתרון:

לפתרון שאלה זו עלינו לשלב שימוש בשתי תבניות שונות: **צבירה ומציאת ערך נלווה למינימום בסדרה**. משתנה הצבירה יצבור עבור כל עובד את סכום משכורותיו. בשל כך, יש לתת תשומת לב מיוחד לאתחול משתנה הצבירה **לפני** תחילת קליטת המשכורת הראשונה של העובד התורן. בסיום צבירת 12 המשכורות של העובד התורן, נשתמש בתבנית **מציאת ערך נלווה למינימום בסדרה** על מנת לשמור את משכורתו האחרונה של העובד התורן במידה וסכום משכורותיו הוא המינימלי עד כה. את התהליך המתואר לעיל נבצע עבור כל אחד מעובדי המפעל. המבנה התכנותי המתאים לביצוע משימה זו הוא לולאה מקוננת.

תבניות:

צבירה

מציאת ערך נלווה למינימום בסדרה

בחירת משתנים:

workers – שלם, מספר העובדים במפעל

salery – שלם, המשכורת התורנית ברשימת משכורות הקלט

sum– שלם, צובר משכורות הקלט

minSumSalery – שלם, שומר את סכום המשכורות הנמוך ביותר

minLastSalery – שלם, שומר את המשכורות האחרונה של העובד בעל סכום המשכורות הנמוך ביותר

NUM_OF_SALARY – קבוע שלם, שומר את מספר המשכורות לכל עובד

האלגוריתם:

1. *אתחל את* minSumSalery *ל-*0
2. *קלוט מספר שלם חיובי ב-*workers
3. **עבור** i *מ-*1 *עד* workers **בצע**:
   1. *אתחל את* sum *ל-*0
   2. **עבור** j *מ-*1 *עד* NUM_OF_SALARY **בצע**:
      1. *קלוט מספר שלם חיובי ב-*salery
      2. *הוסף את ערכו של* salery *ל-*sum
   3. **אם** minSumSalery *שווה ל-*0 *או* sum *קטן מ-*minSumSalery
      1. *השם את* sum *ב-*minSumSalery
      2. *השם את* salery *ב-*minLastSalery
4. *הדפס את ערך* minLastSalery *כפלט*

יישום האלגוריתם:

```
/*
קלט: מספר עובדי המפעל, ו12 משכורות עבור כל עובד
פלט: המשכורות האחרונה של העובד בעל סכום המשכורות הנמוך ביותר
*/
import java.util.Scanner;
public class Saleries
{
      public static void main (String[] args)
      {
            int salary=0;
            int minSumSalery=0;
            int minLastSalery=0;
            final int NUM_OF_SALARY = 12;
            Scanner in = new Scanner(System.in);
            System.out.print("Enter number of workers: ");
            int workers  = in.nextInt();
            for(int i = 1; i <= workers ; i++)
            {
                  int sum = 0;
                  System.out.println("Worker number " + i + ": ");
                  for(int j = 1; j <= NUM_OF_SALARY; j++)
                  {
                        System.out.print("\tEnter salary number "+j);
                        salary = in.nextInt();
                        sum = sum + salary;
                  } // for j

                  if (minSumSalery==0 || sum < minSumSalery )
                  {
                        minSumSalery = sum;
                        minLastSalery = salary;
                  }
            } // for i
            System.out.println("Last salary of poorest worker is: " +
                                                      minLastSalery) ;
      } // main
} // Saleries
```