

הצעה לפתרון
אלגוריתמיקה אביב 2025 מועד א

שאלה 1:

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        int sum = 0;
        int count = 0;
        int positiveCount = 0;

        while (sum <= 100 && count < 15) {
            System.out.print("הכנס מספר שלם: ");
            int num = scanner.nextInt();

            sum += num;
            count++;

            if (num > 0) {
                positiveCount++;
            }
        }

        double average = (count == 0) ? 0 : (double) sum / count;

        System.out.println("הממוצע של המספרים שהוזנו הוא: " + average);
        System.out.println("כמות המספרים החיוביים: " + positiveCount);
    }
}
```

```
}  
}
```

שאלה 2:

```
import java.util.Scanner;  
  
public class Main {  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
  
        String longest = "";  
        String shortest = null;  
        int capitalCount = 0;  
  
        System.out.println("הכנס 30 מחרוזות:");  
  
        for (int i = 0; i < 30; i++) {  
            String input = scanner.nextLine();  
  
            if (input.length() > longest.length()) {  
                longest = input;  
            }  
  
            // עדכון הקצרה ביותר  
            if (shortest == null || input.length() < shortest.length()) {  
                shortest = input;  
            }  
  
            // ASCII בדיקת אותיות ראשונה ואחרונה לפי  
            if (!input.isEmpty()) {  
                char first = input.charAt(0);
```

```

char last = input.charAt(input.length() - 1);

if ((first >= 'A' && first <= 'Z') || (last >= 'A' && last <= 'Z')) {
    capitalCount++;
}
}
}

System.out.println("המחרוזת הארוכה ביותר היא: " + longest);
System.out.println("המחרוזת הקצרה ביותר היא: " + shortest);
System.out.println("כמות המחרוזות שמתחילות או מסתיימות באות גדולה: " +
capitalCount);
}
}

```

שאלה 3:

```

import java.util.Random;

public class Main {
    public static void main(String[] args) {
        Random rand = new Random();

        int evenCount = 0;
        int hasEvenDigitCount = 0;

        for (int i = 0; i < 60; i++) {
            int number = rand.nextInt(900) + 100; // מספרים בין 100 ל-999 (תלת ספרתיים)
            System.out.println("המספר שנוצר: " + number);

            // בדיקה אם המספר זוגי
            if (number % 2 == 0) {
                evenCount++;
            }
        }
    }
}

```

```

    }

    // בדיקה אם יש לפחות ספרה זוגית אחת במספר
    if (hasEvenDigit(number)) {
        hasEvenDigitCount++;
    }
}

System.out.println("כמות המספרים הזוגיים: " + evenCount);

System.out.println("כמות המספרים שיש בהם לפחות ספרה זוגית אחת: " +
hasEvenDigitCount);
}

// פונקציה שבודקת אם למספר יש לפחות ספרה זוגית אחת
public static boolean hasEvenDigit(int num) {
    while (num > 0) {
        int digit = num % 10;
        if (digit % 2 == 0) {
            return true;
        }
        num /= 10;
    }
    return false;
}
}

```

שאלה 4:

סעיף א

```

public boolean IsItSuitable(char gender, int age) {
    return this.gender == gender && this.age == age;
}

```

סעיף ב

```
public static void Print(Dog[] arr) {  
    for (Dog dog : arr) {  
        if (dog.IsItSuitable('M', 1) || dog.IsItSuitable('M', 2)) {  
            if (dog.isImmuned()) {  
                System.out.println(dog.getName());  
            }  
        }  
    }  
}
```

סעיף ג:

```
public static int CountNoAge(Dog[] arr) {  
    int count = 0;  
  
    for (Dog dog : arr) {  
        if (!dog.isImmuned() && dog.getAge() == 0) {  
            count++;  
        }  
    }  
  
    return count;  
}
```

שאלה 5:

סעיף א:

```
public static boolean IsNice (int[] arr) {  
    int mid = arr.length / 2;  
    int leftSum = 0;
```

```

int rightSum = 0;

for (int i = 0; i < mid; i++) {
    leftSum += arr[i];
}

for (int i = mid + 1; i < arr.length; i++) {
    rightSum += arr[i];
}

return leftSum == arr[mid] && rightSum == arr[mid];
}
}

```

סעיף ב: הערה – האיבר האמצעי יכול להיות גדול מ 999 בשל העובדה שהשאלה לא הגבילה את ה
size

```

public static int[] DoltNice(int size) {

    Random rand = new Random();
    int[] arr = new int[size];
    int mid = size / 2;

    int leftSum = 0;

    for (int i = 0; i < mid; i++) {
        arr[i] = rand.nextInt(900) + 100; // בין 100 ל-999
        leftSum += arr[i];
    }

    arr[mid] = leftSum;

    int rightSum = 0;
    for (int i = mid + 1; i < size; i++) {
        if (i == size - 1) {
            arr[i] = leftSum - rightSum;
        } else {
            int max = Math.min(999, leftSum - rightSum - (size - 1 - i) * 100);
            int min = 100;

```

```

        if (max < min) max = min;
        arr[i] = rand.nextInt(max - min + 1) + min;
    }
    rightSum += arr[i];
}

return arr;
}
}

```

שאלה 6:

סעיף א

```

public static int Excellent(int[] gradesCountingArray) {
    int count = 0;
    for (int i = 91; i <= 100; i++) {
        count += gradesCountingArray[i];
    }

    return count;
}

```

סעיף ב:

```

public static int HigherGrade(int[] gradesCountingArray) {
    for (int i = 100; i >= 0; i--) {
        if (gradesCountingArray[i] > 0) {
            return i;
        }
    }

    return -1;
}

```

סעיף ג:

```
public static int MostCommonScore(int[] gradesCountingArray) {  
    int maxCount = 0;  
    int scoreWithMaxCount = -1;  
  
    for (int i = 0; i < gradesCountingArray.length; i++) {  
        if (gradesCountingArray[i] > maxCount) {  
            maxCount = gradesCountingArray[i];  
            scoreWithMaxCount = i;  
        } else if (gradesCountingArray[i] == maxCount && i > scoreWithMaxCount) {  
            scoreWithMaxCount = i;  
        }  
    }  
  
    return scoreWithMaxCount;  
}
```


סעיף א

Index	s[i]	s[i+1]	אות גדולה?	התנאי מתקיים?	נוסף לפלט?
0	K	%	כן	כן	כן
1	%	%	לא	כן	כן
2	%	A	לא	לא	לא
3	A	B	כן	כן	כן
4	B	B	כן	לא	לא
5	B	2	כן	כן	כן
6	2	3	לא	לא	לא
7	3	C	לא	לא	לא
8	C	C	כן	לא	לא
9	C	C	כן	לא	לא
10	C	?	כן	כן	כן
11	?	?	לא	כן	כן
12	?	D	לא	לא	לא
13	D	D	כן	לא	לא
14	D	—	—	—	כן (תו אחרון)

תוצאה סופי: "K%ABC?D"

סעיף ב:

תנאים שצריכים להתקיים בכל מיקום במחרוזת (מלבד האחרון):

עבור תו באינדקס i:

- אם $s[i] \neq s[i+1]$: חייב להיות גדולה
- אם $s[i] == s[i+1]$: חייב להיות לא אות גדולה

המחרוזת "AB" כן מקיימת.

סעיף ג:

כן. Str1="M" str2="MM"

סעיף ג:

הפחתת כפילויות או ניקוי תווים מיותרים, אבל באופן חכם ומבחין בין סוגי תווים:

- היא שומרת על אותיות גדולות (ייחודיות) לא חוזרות פעמיים רצוף.
- היא שומרת רק על תווים לא-אותיים כשהם מופיעים ברצף כמו %% או ??

יוצרת מעין "חתימה דחוסה" של הטקסט:

נותנת גרסה מקוצרת אך יציבה מבחינת הדפוס של רצפים – אפשר לראות בזה סוג של קידוד או סינון לפלט מוגן.

שאלה 8

סעיף א:

```

public static int Median(int[] arr) {
    int n = arr.length;
    int target = (n - 1) / 2;

    for (int i = 0; i < n; i++) {
        int smaller = 0;
        int greater = 0;

        for (int j = 0; j < n; j++) {
            if (arr[j] < arr[i]) smaller++;
            if (arr[j] > arr[i]) greater++;
        }

        if (smaller == target && greater == target) {
            return arr[i]; // מצאנו את החציון
        }
    }

    return -1;
}
  
```

סעיף ב

```
public static int[] BuildSpecial(int[] arr) {  
    int n = arr.length;  
    int median = findMedian(arr);  
  
    int[] result = new int[n];  
    int leftIndex = 0;  
    int rightIndex = n - 1;  
    for (int i = 0; i < n; i++) {  
        if (arr[i] < median) {  
            result[leftIndex++] = arr[i];  
        } else if (arr[i] > median) {  
            result[rightIndex--] = arr[i];  
        }  
    }  
  
    result[n / 2] = median;  
  
    return result;  
}
```

סעיף ג

סיבוכיות:

- לולאה חיצונית $O(n)$:
- לולאה פנימית $O(n)$:
- סך הכול:

$$(O(n^2))$$

שאלה 9:

סעיף א

```
public int compareTo(Baby other) {  
    boolean thisHeavier = this.weight > other.weight;  
    boolean thisTaller = this.height > other.height;  
    boolean sameWeight = this.weight == other.weight;  
    boolean sameHeight = this.height == other.height;  
  
    if (sameWeight && sameHeight) {  
        return 0;  
    } else if (thisHeavier && thisTaller) {  
        return 1;  
    } else if (thisHeavier) {  
        return 2;  
    } else if (thisTaller) {  
        return 3;  
    } else {  
        return 4;  
    }  
}
```

סעיף ב:

```
public boolean AreTwins(Baby other) {  
    // תנאי 1: שם משפחה זהה  
    if (!this.lastName.equals(other.lastName)) return false;  
  
    // תנאי 2: שם פרטי שונה  
    if (this.firstName.equals(other.firstName)) return false;  
  
    // תנאי 3: ת"ז שונה  
    if (this.id.equals(other.id)) return false;
```

```
// תנאי 4: תאריך זהה או בהפרש של יום
Date d1 = this.birth;
Date d2 = other.birth;

if (d1.equals(d2) || d1.tomorrow().equals(d2) || d2.tomorrow().equals(d1)) {
    return true;
}

return false;
}
```

סעיף ג:

```
public static void Print(Baby[] babies, Baby baby) {
    int count = 0;

    for (int i = 0; i < babies.length; i++) {
        int cmp = babies[i].compareTo(baby);
        if (cmp == 0) {
            count++;
        }
    }

    System.out.println("מספר התינוקות עם גובה ומשקל זהים או טובים יותר" + count);
}
```

סעיף ד:

```
public static int CountTwins(Baby[] babies) {
    int count = 0;

    for (int i = 0; i < babies.length; i++) {
        for (int j = i + 1; j < babies.length; j++) {
```

```

if (babies[i].AreTwins(babies[j])) {
    count++;
}
}
}

return count;
}

```

שאלה 10:

סעיף א:

Num=9630

הפעולה בודקת האם כל ספרה במספר הנתון **גדולה מהספרה שמימינה**, כלומר:

האם המספר הוא בסדר ספרות **יורד** משמאל לימין

אם כן → מוחזרת true
אם לא → מוחזרת false

סעיף ב:

טבלת מעקב:

אחרי res	? Value > arr[res]	לפני res	One(Value)	Value	Index
0	כן	1-	true	21	0
0	לא	0	true	13	1
0	-	0	false	34783	2
3	כן (21 < 68)	0	true	68	3
3	-	3	false	131	4
3	-	3	false	468	5
3	-	3	false	761	6

תוצאה סופית:

- המספרים שענו על 21, 13, 68 One:
- הגדול מביניהם 68 :
- מיקומו **index 3** :

מחזירה את אינדקס המספר הגדול ביותר במערך שספרותיו בסדר יורד מימין לשמאל, כלומר כזה שעבורו One(num) מחזירה אמת.
 אם אין אף מספר כזה – מוחזרת התוצאה -1

שאלה 11

סעיף א

```

public boolean isBlack() {
    return red == 0 && green == 0 && blue == 0;
}
  
```

סעיף ב

```

public boolean isBalancedPicture() {
    int blackCount = 0;
    int whiteCount = 0;

    for (int i = 0; i < 256; i++) {
        for (int j = 0; j < 256; j++) {
            if (pixels[i][j] != null) {
                if (pixels[i][j].isBlack()) {
                    blackCount++;
                } else if (pixels[i][j].isWhite()) {
                    whiteCount++;
                }
            }
        }
    }
}
  
```

```
return blackCount == whiteCount;  
}
```

סעיף ג

```
public boolean isBlackWhite() {  
    for (int i = 0; i < 256; i++) {  
        for (int j = 0; j < 256; j++) {  
            if (pixels[i][j] != null) {  
                if (!pixels[i][j].isBlack() && !pixels[i][j].isWhite()) {  
                    return false;  
                }  
            }  
        }  
    }  
    return true;  
}
```

סעיף ד

```
public void setWhiteBorder() {  
    // RGB בניח שקיים בנאי לפיקסל שמקבל שלושת ערכי  
    Pixel whitePixel = new Pixel(255, 255, 255);  
  
    // שורות גבול (0 ו-255)  
    for (int j = 0; j < 256; j++) {  
        pixels[0][j] = whitePixel;  
        pixels[255][j] = whitePixel;  
    }  
  
    // עמודות גבול (0 ו-255)  
    for (int i = 1; i < 255; i++) {
```



```
pixels[i][0] = whitePixel;
pixels[i][255] = whitePixel;
}
}
```

שאלה 12

סעיף א

```
public class Clinic {
    private String name;
    private String location;
    private int status; // 1-ילדים 2-מבוגרים 3-גם וגם

    public Clinic(String name, String location, int status) {
        this.name = name;
        this.location = location;
    }
}
```

```
public class Center {
    private String name;
    private String location;
    private Clinic[] clinics;
    private int numClinics;

    public Center(String name, String location) {
        this.name = name;
        this.location = location;
        this.clinics = new Clinic[15]; // קיבולת מקסימלית
        this.numClinics = 0;
    }
}
```

```

}

public class Chain
{
    private Center[] centers;
    private int numCenter;
    public Chain() {
        this.center = new Center[40];
        this.numCenter = 0;
    }
}

```

סעיף ב

```

public boolean isClinicForChildren(String clinicName) {
    for (int i = 0; i < numClinics; i++) {
        Clinic c = clinics[i];
        if (c != null && c.getName().equals(clinicName)) {
            return c.getStatus() == 1;
        }
    }
    return false; // לא נמצאה מרפאה בשם הזה או שאינה לילדים
}

```

סעיף ג

```

public void updateOrAddEyeClinic() {
    boolean foundEyeClinic = false;

    for (int i = 0; i < numCenters; i++) {
        Center center = centers[i];
    }
}

```

```

for (int j = 0; j < center.getNumClinics(); j++) {
    Clinic clinic = center.getClinics()[j];
    if (clinic.getName().contains("עיניים")) {
        foundEyeClinic = true;

        if (clinic.getStatus() != 1 && clinic.getStatus() != 3) {
            clinic.setStatus(3);
        }
    }
}

if (!foundEyeClinic) {
    if (center.getNumClinics() < 15) {
        Clinic newClinic = new Clinic("ילדים", "מרפאת עיני ידוע",
            center.getLocation());
        center.getClinics()[center.getNumClinics()] = newClinic;
        center.setNumClinics(center.getNumClinics() + 1);
    }
}
}

```

שאלה 13

סעיף א

קריאה	s1	s2	s1[0] vs s2[0]	תוצאה
Mystery("HELLO", "HELP")	'H'	'H'	שווים	ממשיכים רקורסיה
Mystery("ELLO", "ELP")	'E'	'E'	שווים	ממשיכים רקורסיה
Mystery("LLO", "LP")	'L'	'L'	שווים	ממשיכים רקורסיה
Mystery("LO", "P")	'L'	'P'	L < P	מחזיר 1-

Mystery("HELLO", "HELP") ⇒ -1

סעיף ב

הסבר:

הפונקציה Mystery מחזירה:

- 1- אם $s1 < s2$ לקסיקוגרפית
- 1 אם $s1 > s2$
- 0 אם הן שוות

כאן "HELLO" – קטנה מ- "HELP" לכן מתקבל 1-

סעיף ג

מיקום במערך temp	תוצאה	Mystery מול "JAVA"	ערך
temp[0]	קטנה	1-	"HELLO"
temp[5]	גדולה	1	"WORLD"
temp[1]	קטנה	1-	"GOOD"
temp[2]	קטנה	1-	"DAY"
temp[3]	קטנה	1-	"FOR"
temp[4]	גדולה	1	"TEST"

תוצאה סופית:

`temp = { "HELLO", "GOOD", "DAY", "FOR", "TEST", "WORLD" }`

הפונקציה Secret ממיינת את arr לתוך מערך חדש temp, שכל מחרוזת שקטנה מ- "JAVA" לפי Mystery תיכנס מימין לשמאל (מהאינדקס 0 ומעלה), וכל מחרוזת שגדולה או שווה ל- "JAVA" תיכנס משמאל לימין (מהסוף ויורד).

סעיף ד:

הפונקציה מחלקת את המערך לשני חלקים:

- בצד שמאל: כל המחרוזות ש"קטנות" מ-s
- בצד ימין: כל המחרוזות ש"גדולות או שוות" ל-s

הסדר בתוך כל צד נשמר כפי שהיה ב- arr כי עוברים בסדר מקורי.

שאלה 14

```
public class MatrixUtils {

    public static boolean isSortedCandidate(int[][] matrix) {

        int rows = matrix.length;
        int cols = matrix[0].length;

        // תנאי 1: בדיקה אם ריבועי
        if (rows != cols) {
            return false;
        }

        // תנאים 2 ו-3:
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                int value = matrix[i][j];

                // תנאי 2: מספרים חיוביים בלבד
                if (value <= 0) {
                    return false;
                }

                // תנאי 3: המספר צריך להכיל את הספרה של העמודה
                if (!containsDigit(value, j)) {
                    return false;
                }
            }
        }

        return true;
    }
}
```

```
}  
  
// פונקציית עזר: בודקת אם מספר מכיל ספרה מסוימת  
private static boolean containsDigit(int num, int digit) {  
    while (num > 0) {  
        if (num % 10 == digit) {  
            return true;  
        }  
        num /= 10;  
    }  
    return false;  
}  
}
```

סעיף ב

סיבוכיות: $O(n^{2*m})$

הסבר:

הפעולה בודקת כל תא במערך, ולכל אחד מבצעת בדיקה של ספרות. מספר באורך m . לכן, זמן הריצה פרופורציונלי לגודל המערך כולו – כלומר ריבועי במספר השורות/עמודות.