

פתרון JAVA-

אלגוריתמיקה ותכנות

קיץ 2025 מועד א' 97104

פתרון שאלה 1: (12 נקודות)

```
public static String encrypt(String message) {

    String noSpaces = "";
    for (int i = 0; i < message.length(); i++) {
        char c = message.charAt(i);
        if (c != ' ') {
            noSpaces=noSpaces+c;
        }
    }
    String swapped = "";
    int i = 0;
    while (i < noSpaces.length() - 1) {
        swapped = swapped+noSpaces.charAt(i+1)+noSpaces.charAt(i);
        i += 2;
    }
    if (i < noSpaces.length()) {
        swapped = swapped+noSpaces.charAt(i);
    }
    String reversed = "";
    for (int j = swapped.length()-1; j >= 0; j--) {
        reversed = reversed + swapped.charAt(j);
    }
    return reversed;
}
```

פתרון שאלה 2 (12 נקודות)

```
public static int findBalancePoint(int[] arr) {
    int n = arr.length;

    for (int k = 0; k < n; k++) {
        int leftProduct = 1;
        for (int i = 0; i < k; i++) {
            leftProduct *= arr[i];
        }

        int rightProduct = 1;
        for (int j = k + 1; j < n; j++) {
            rightProduct *= arr[j];
        }

        if (leftProduct == rightProduct) {
            return k;
        }
    }

    return -1;
}
```

פתרון שאלה 3 (12 נקודות)

סעיף א (4 נק)

arrA = {12, 6, 3, 17, 4, 5, 2, 8, 10, 13}

הערות m c? זוגי arr[i] i

0 12 בן 1 0 ++c → זוגי

1 6 בן 2 0 ++c → זוגי

2 3 לא 0 2 m ל־2 מעודכן → אי-זוגי

3 17 לא 0 2 → אי-זוגי אין שינוי

4 4 בן 1 2 התחלה של רצף חדש

5 5 לא 0 2 → אי-זוגי אין שינוי

6 2 בן 1 2 התחלה של רצף חדש

7 8 בן 2 2 המשך רצף

8 10 בן 3 2 המשך רצף

9 13 לא 0 3 m ל־3 מעודכן → אי-זוגי

what(arrA) = 3

סעיף ב (3 נק)

{2, 4, 6, 8, 3, 1}

יחזיר 4

סעיף ג (3 נק)

{1, 3, 5, 7, 9, 11}

יחזיר 0

סעיף ד (2 נק)

הסבר: הפעולה מחשבת את האורך של רצף המספרים הזוגיים הרצוף הארוך ביותר במערך.

פתרון שאלה 4 (12 נקודות)

סעיף א: (3 נקודות)

```
public int getSalary() {
    int baseHours = Math.min(160, this.hours);
    int extraHours = Math.max(0, this.hours - 160);
    int baseRate = (this.status == 2) ? 80 : 150;
    return (baseHours * baseRate) + (extraHours * 70);
}
```

סעיף ב: (3 נקודות)

```
Scanner in = new Scanner(System.in);
int s = 0;
String n = in.next();           // (1) קלט של מספר זהות
int t = in.nextInt();           // (2) קלט של סאטטוס
Employee emp = new Employee(n, t); // (3) יצירת אובייקט מסוג Employee

for (int i = 1; i <= 25; i++) {
    int a = in.nextInt();
    int b = in.nextInt();
    s = s + (b - a);             // (4) צבירת שעות עבודה ליום
}

emp.setHours(s);                // (5) עדכון סך השעות באובייקט
System.out.println(emp.getSalary()); // (6) הדפסת השכר
```

סעיף ג: (6 נקודות)

```
public static void printWorkersAboveEngineerAvg(Employee[] employees) {
    int sum = 0;
    int count = 0;
    for (int i = 0; i < employees.length; i++) {
        if (employees[i].getStatus() == 1) {
            sum += employees[i].getSalary();
            count++;
        }
    }
    if (count == 0) {
        System.out.println("nothing to compare");
        return;
    }
    double avg = (double) sum / count;

    System.out.println("high salary workers:");
    for (int i = 0; i < employees.length; i++) {
        if (employees[i].getStatus() == 2 && employees[i].getSalary() > avg) {
            System.out.println(employees[i].getId());
        }
    }
}
```

פתרון שאלה 5 (15 נקודות)

סעיף א: (4 נקודות)

```
public static boolean isPrimary(int num) {
    if (num < 2) return false;
    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0) return false;
    }
    return true;
}
```

סעיף ב: (4 נקודות)

```
public static int countPrimaryPairs(int num) {
    int count = 0;
    for (int i = 1; i <= num / 2; i+=2) {
        if (isPrimary(i) && isPrimary(num - i)) {
            count++;
        }
    }
    return count;
}
```

סעיף ג: (4 נקודות)

```
public static int[] allPrimaryPairs(int num) {
    int count = countPrimaryPairs(num);
    int[] result = new int[count * 2];
    int index = 0;

    for (int i = 1; i <= num / 2; i+=2) {
        int j = num - i;
        if (isPrimary(i) && isPrimary(j)) {
            result[index++] = i;
            result[index++] = j;
        }
    }

    return result;
}
```

סעיף ד: (3 נקודות)

prime - $O(\sqrt{n})$
countprimepairs = $O(n \cdot \sqrt{n})$
allPrimaryPairs = $O(n \cdot \sqrt{n})$

פתרון שאלה 6 (15 נקודות)

סעיף א: (3 נקודות)

```
public Warehouse() {
    flags = new Flag[100];
    quantities = new int[100];
    currentFlags = 0;
}
```

סעיף ב: (4 נקודות)

```
// פעולת עזר
public boolean equals(Flag other) {
    return this.country.equals(other.country) &&
           this.length == other.length &&
           this.width == other.width;
}

public void add(String country, int length, int width, int quant) {
    Flag newFlag = new Flag(country, length, width);
    for (int i = 0; i < currentFlags; i++) {
        if (flags[i].equals(newFlag)) {
            quantities[i] += quant;
            return;
        }
    }
    if (currentFlags >= 100) {
        System.out.println("Warehouse is full. Cannot add new flag.");
        return;
    }
    flags[currentFlags] = newFlag;
    quantities[currentFlags] = quant;
    currentFlags++;
}
```

סעיף ג: (4 נקודות)

```
public void flagsWithMinQuantity(int minQuantity) {
    System.out.println("Flags with quantity less than " + minQuantity + ":");
    for (int i = 0; i < currentFlags; i++) {
        if (quantities[i] < minQuantity) {
            System.out.println(flags[i].toString()+"-Quantity: "+quantities[i]);
        }
    }
}
```

סעיף ד: (4 נקודות)

```
public void printFlags(String[] countries) {
    System.out.println("Total flag quantities per country:");
    for (String country : countries) {
        int total = 0;
        for (int i = 0; i < currentFlags; i++) {
            if (flags[i].getCountry().equals(country)) {
                total += quantities[i];
            }
        }
        System.out.println(country + ": " + total + " flag(s)");
    }
}
```

פתרון שאלה 7 (15 נקודות)

סעיף א: (5 נקודות)

1.

קריאה	num	s (sum of digits)	c (count of digits)
1	6123	$0 + 3 = 3$	$0 + 1 = 1$
2	612	$3 + 2 = 5$	$1 + 1 = 2$
3	61	$5 + 1 = 6$	$2 + 1 = 3$
4	6	$6 + 6 = 12$	$3 + 1 = 4$
5	0		

return $12 / 4 = 3.0$

2.

5546

תחזיר 5

3.

ממוצע ספרות

סעיף ב (10 נקודות)

1.

int[] a = {24, 126, 9, 35, 2684, 8941};

int p = 3; // one(a[3]) = one(35)

index k	value	one(value)	one(p) = 4.0	condition	temp location
0	24	$(2+4)/2 = 3.0$	< 4.0	left $\leftarrow 0$	temp[0] = 24
1	126	$(1+2+6)/3 = 3.0$	< 4.0	left $\leftarrow 1$	temp[1] = 126
2	9	$9/1 = 9.0$	> 4.0	right $\leftarrow 5$	temp[5] = 9
3	35	4.0	$=$	left $\leftarrow 2$	temp[2] = 35
4	2684	$2+6+8+4 = 20 / 4 = 5.0$	> 4.0	right $\leftarrow 4$	temp[4] = 2684
5	8941	$8+9+4+1 = 22 / 4 = 5.5$	> 4.0	right $\leftarrow 3$	temp[3] = 8941

temp = {24, 126, 35, 8941, 2684, 9}

2.

כן, שגיאה תתרחש אם p לא נמצא בטווח התקני של האינדקסים במערך כלומר $p < 0$ או $p \geq a.length$

3.

int[] arr = {111, 111, 111, 111, 111, 111};

int[] arr = {123, 114, 3320, 141, 312, 213};

4. הפעולה three(arr, p) מחזירה מערך חדש, שבו כל המספרים ממוקמים מחדש בהתאם לממוצע הספרות של כל מספר בהשוואה לממוצע הספרות של המספר שנמצא במקום p במערך. הקטנים לפיו והגדולים אחריו.

פתרון שאלה 8 (17 נקודות)

```
public static boolean exist(char[][] arr, String word) {
    int rows = arr.length;
    int cols = arr[0].length;
    int len = word.length();

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            if (checkRight(arr, word, i, j, rows, cols, len) ||
                checkLeft(arr, word, i, j, len) ||
                checkDown(arr, word, i, j, rows, len) ||
                checkUp(arr, word, i, j, len)) {
                return true;
            }
        }
    }
    return false;
}

// בדיקה מימין לשמאל
private static boolean checkRight(char[][] arr, String word, int i, int j,
int rows, int cols, int len) {
    if (j + len > cols) return false;
    for (int k = 0; k < len; k++) {
        if (arr[i][j + k] != word.charAt(k)) return false;
    }
    return true;
}

// בדיקה משמאל לימין
private static boolean checkLeft(char[][] arr, String word, int i, int j,
int len) {
    if (j - len + 1 < 0) return false;
    for (int k = 0; k < len; k++) {
        if (arr[i][j - k] != word.charAt(k)) return false;
    }
    return true;
}

// בדיקה מלמעלה למטה
private static boolean checkDown(char[][] arr, String word, int i, int j,
int rows, int len) {
    if (i + len > rows) return false;
    for (int k = 0; k < len; k++) {
        if (arr[i + k][j] != word.charAt(k)) return false;
    }
    return true;
}

// בדיקה מלמטה למעלה
private static boolean checkUp(char[][] arr, String word, int i, int j, int len)
{
    if (i - len + 1 < 0) return false;
    for (int k = 0; k < len; k++) {
        if (arr[i - k][j] != word.charAt(k)) return false;
    }
    return true;
}
```


פתרון נוסף של שאלה 8

```
public static String rowToString(char[][] arr, int row)
{
    String s = "";
    for(int j = 0; j<arr[row].length; j++)
        s+=arr[row][j];
    return s;
}

public static String colToString(char[][] arr, int col)
{
    String s = "";
    for(int i = 0; i<arr.length; i++)
        s+=arr[i][col];
    return s;
}

public static String reverse(String str)
{
    String s = "";
    for(int i =0; i<str.length(); i++)
        s = str.charAt(i)+s;
    return s;
}

public static boolean exist(char[][] arr, String word)
{
    String reverseWord = reverse(word);
    for(int i= 0; i<arr.length; i++)
    {
        String rowString = rowToString(arr, i);
        if(rowString.indexOf(word)!=-1) return true;
        if(rowString.indexOf(reverseWord)!=-1) return true;
    }
    for(int j= 0; j<arr[0].length; j++)
    {
        String colString = colToString(arr, j);
        if(colString.indexOf(word)!=-1) return true;
        if(colString.indexOf(reverseWord)!=-1) return true;
    }
    return false;
}
```

פתרון שאלה 9 (17 נקודות)

סעיף א: (2 נקודות)

```
public Account(String username, String initialPass) {
    this.username = username;
    this.currentPass = initialPass;
    this.passHistory = new String[10];
    this.passHistory[0] = initialPass;
    this.historySize = 1;
}
```

סעיף ב: (4 נקודות)

```
public boolean isValidPassword(String password) {
    int criteriaMet = 0;

    if (password.length() >= 8)
        criteriaMet++;
    int upperCount = 0;
    for (int i = 0; i < password.length(); i++) {
        char c = password.charAt(i);
        if (c >= 'A' && c <= 'Z') {
            upperCount++;
        }
    }
    if (upperCount >= 2)
        criteriaMet++;

    boolean hasDigit = false;
    for (int i = 0; i < password.length(); i++) {
        char c = password.charAt(i);
        if (c >= '0' && c <= '9') {
            hasDigit = true;
            break;
        }
    }
    if (hasDigit)
        criteriaMet++;
    boolean hasSpecial = false;
    for (int i = 0; i < password.length(); i++) {
        char c = password.charAt(i);
        if (!(c >= 'A' && c <= 'Z') ||
            (c >= 'a' && c <= 'z') ||
            (c >= '0' && c <= '9')) {
            hasSpecial = true;
            break;
        }
    }
    if (hasSpecial)
        criteriaMet++;
    boolean noRepeats = true;
    for (int i = 1; i < password.length(); i++) {
        if (password.charAt(i) == password.charAt(i - 1)) {
            noRepeats = false;
            break;
        }
    }
    if (noRepeats)
        criteriaMet++;

    return criteriaMet >= 3;
}
```

סעיף ג: (4 נקודות)

```
public void updatePassword(String newPass) {
    if (!isPasswordValid(newPass)) {
        System.out.println("האבטחה בדרישות עומדת אינה החדשה הסיסמה");
        return;
    }
    if (newPass.equals(currentPass)) {
        System.out.println("הנוכחית לסיסמה זהה החדשה הסיסמה");
        return;
    }
    for (int i = 0; i < historySize; i++) {
        if (newPass.equals(passHistory[i])) {
            System.out.println("בעבר בשימוש הייתה כבר הסיסמה");
            return;
        }
    }
    for (int i = Math.min(historySize, 9); i > 0; i--) {
        passHistory[i] = passHistory[i - 1];
    }
    passHistory[0] = newPass;
    currentPass = newPass;
    if (historySize < 10) historySize++;
    System.out.println("בהצלחה עודכנה הסיסמה");
}
```

סעיף ד: (3 נקודות)

```
public void addUser(String username, String password) {
    if (numOfAccounts >= 100) {
        System.out.println("מלאה המערכת - נוסף משתמש להוסיף ניתן לא");
        return;
    }
    for (int i = 0; i < numOfAccounts; i++) {
        if (accounts[i].getUsername().equals(username)) {
            System.out.println("במערכת קיים כבר המשתמש שם");
            return;
        }
    }
    accounts[numOfAccounts] = new Account(username, password);
    numOfAccounts++;
    System.out.println("בהצלחה נוסף משתמש");
}
```

סעיף ה: (4 נקודות)

- דרישה ל-4 מתוך 5 תנאים ישנים במקום 3 מ 5
- ובנוסף חובה שתהיה לפחות אות קטנה אחת

```
public boolean isValid(String password) {
    int criteriaMet = 0;
    // תנאי 1: אורך לפחות 8
    if (password.length() >= 8)
        criteriaMet++;
    // תנאי 2: לפחות 2 אותיות גדולות
    int upperCount = 0;
    for (int i = 0; i < password.length(); i++) {
        char c = password.charAt(i);
        if (c >= 'A' && c <= 'Z') {
            upperCount++;
        }
    }
    if (upperCount >= 2)
        criteriaMet++;
    // תנאי 3: לפחות ספרה אחת
    boolean hasDigit = false;
    for (int i = 0; i < password.length(); i++) {
        char c = password.charAt(i);
        if (c >= '0' && c <= '9') {
            hasDigit = true;
            break;
        }
    }
    if (hasDigit)
        criteriaMet++;
    // תנאי 4: לפחות תו מיוחד
    boolean hasSpecial = false;
    for (int i = 0; i < password.length(); i++) {
        char c = password.charAt(i);
        if (!((c >= 'A' && c <= 'Z') ||
            (c >= 'a' && c <= 'z') ||
            (c >= '0' && c <= '9')))) {
            hasSpecial = true;
            break;
        }
    }
    if (hasSpecial)
        criteriaMet++;

    // תנאי 5: אין תווים זהים צמודים
    boolean noRepeats = true;
    for (int i = 1; i < password.length(); i++) {
        if (password.charAt(i) == password.charAt(i - 1)) {
            noRepeats = false;
            break;
        }
    }
    if (noRepeats)
        criteriaMet++;
    // תנאי חובה: לפחות אות קטנה אחת
    boolean hasLowercase = false;
    for (int i = 0; i < password.length(); i++) {
        char c = password.charAt(i);
        if (c >= 'a' && c <= 'z') {
            hasLowercase = true;
            break;
        }
    }

    // לפחות 4 מתוך 5 + חובה אות קטנה
    return (criteriaMet >= 4 && hasLowercase);
}
```

פתרון שאלה 10 (17 נקודות)

arr = [10, 3, 6, 8, 2, 11, 5];

סעיף א (3 נקודות)

mystery(arr, 1, 4)

מצב לפני:

[10, 3, 6, 8, 2, 11, 5]

תהליך:

היפוך טווח [3, 6, 8, 2] → [2, 8, 6, 3]

תוצאה:

[10, 2, 8, 6, 3, 11, 5]

סעיף ב (3 נקודות)

secret(arr, 3)

מצב לפני:

[10, 3, 6, 8, 2, 11, 5]

שלבים:

mystery(0,6) → [5, 11, 2, 8, 6, 3, 10]

mystery(0,2) → [2, 11, 5, 8, 6, 3, 10]

mystery(3,6) → [2, 11, 5, 10, 3, 6, 8]

תוצאה :

[2, 11, 5, 10, 3, 6, 8]

סעיף ג (2 נקודות)

הסבר לפעולה :

secret(a, d)

הפונקציה מבצעת סיבוב שמאלה של המערך ב־ d מקומות. (left rotation)

היא עושה זאת בעזרת שלושה היפוכים:

היפוך של כל המערך

היפוך של החלק הראשון 0 עד d

היפוך של החלק השני d עד הסוף

סעיף ד (3 נקודות)

– what(arr, 3)

על המערך ההתחלתי

מצב לפני:

[10, 3, 6, 8, 2, 11, 5]

שלבי הרקורסיה: (why)

i j = (i+3)%7 rest = a[j]

0 3 8

1 4 2

2 5 11

3 6 5

4 0 10

5 1 3

6 2 6

שלב חזור – תוצאה:

[8, 2, 11, 5, 10, 3, 6]

סעיף ה (3 נקודות)

לאחר what(brr, 2) מתקבל:

[60, 50, 40, 30, 20, 10]

המערך שעליו פעל:

[40, 30, 20, 10, 60, 50]

סעיף ו (3 נקודות)

מה עושה

What(a,d)

הפעולה יוצרת סיבוב ימינה ב-d מקומות במערך

C#- פתרון

אלגוריתמיקה ותכנות

קיץ 2025 מועד א' 97104

פתרון שאלה 1: (12 נקודות)

```
public static string Encrypt(string message)
{
    string noSpaces = "";
    foreach (char c in message)
    {
        if (c != ' ')
            noSpaces += c;
    }
    string swapped = "";
    int i = 0;
    while (i < noSpaces.Length - 1)
    {
        swapped += noSpaces[i + 1];
        swapped += noSpaces[i];
        i += 2;
    }
    if (i < noSpaces.Length)
        swapped += noSpaces[i];

    char[] reversedArray = swapped.ToCharArray();
    Array.Reverse(reversedArray);
    return new string(reversedArray);
}
```

פתרון שאלה 2 (12 נקודות)

```
public static int FindBalancePoint(int[] arr)
{
    int n = arr.Length;
    for (int k = 0; k < n; k++)
    {
        int leftProduct = 1;
        for (int i = 0; i < k; i++)
            leftProduct *= arr[i];

        int rightProduct = 1;
        for (int j = k + 1; j < n; j++)
            rightProduct *= arr[j];

        if (leftProduct == rightProduct)
            return k;
    }
    return -1;
}
```


פתרון שאלה 3 (12 נקודות)

סעיף א (4 נק)

arrA = {12, 6, 3, 17, 4, 5, 2, 8, 10, 13}

הערות m c? זוגי arr[i] i

0 12 בן 1 0 ++c → זוגי

1 6 בן 2 0 ++c → זוגי

2 3 לא 0 2 m ל־2 מעודכן → אי-זוגי

3 17 לא 0 2 → אי-זוגי אין שינוי

4 4 בן 1 2 התחלה של רצף חדש

5 5 לא 0 2 → אי-זוגי אין שינוי

6 2 בן 1 2 התחלה של רצף חדש

7 8 בן 2 2 המשך רצף

8 10 בן 3 2 המשך רצף

9 13 לא 0 3 m ל־3 מעודכן → אי-זוגי

what(arrA) = 3

סעיף ב (3 נק)

{2, 4, 6, 8, 3, 1}

יחזיר 4

סעיף ג (3 נק)

{1, 3, 5, 7, 9, 11}

יחזיר 0

סעיף ד (2 נק)

הסבר: הפעולה מחשבת את האורך של רצף המספרים הזוגיים הרצוף הארוך ביותר במערך.

פתרון שאלה 4 (12 נקודות)

סעיף א: (3 נקודות)

```
public int GetSalary()
{
    int baseHours = Math.Min(160, this.hours);
    int extraHours = Math.Max(0, this.hours - 160);
    int baseRate = (this.status == 2) ? 80 : 150;
    return (baseHours * baseRate) + (extraHours * 70);
}
```

סעיף ב: (3 נקודות)

```
int s = 0;
String n = Console.ReadLine();// (1) קלט של מספר זהות
int t = int.Parse(Console.ReadLine());// (2) קלט של סאטטוס
Employee emp = new Employee(n, t); // (3) יצירת אובייקט מסוג Employee

for (int i = 1; i <= 25; i++) {
    int a = int.Parse(Console.ReadLine());
    int b = int.Parse(Console.ReadLine());
    s = s + (b - a); // (4) צבירת שעות עבודה ליום
}

emp.SetHours(s); // (5) עדכון סך השעות באובייקט
System.out.println(emp.GetSalary()); // (6) הדפסת השכר
```

סעיף ג: (6 נקודות)

```
public static void PrintWorkersAboveEngineerAvg(Employee[] employees)
{
    int sum = 0;
    int count = 0;
    foreach (var emp in employees)
    {
        if (emp.GetStatus() == 1)
        {
            sum += emp.GetSalary();
            count++;
        }
    }
    if (count == 0)
    {
        Console.WriteLine("nothing to compare");
        return;
    }

    double avg = (double)sum / count;

    Console.WriteLine("high salary workers:");
    foreach (var emp in employees)
    {
        if (emp.GetStatus() == 2 && emp.GetSalary() > avg)
        {
            Console.WriteLine(emp.GetId());
        }
    }
}
```

פתרון שאלה 5 (15 נקודות)

סעיף א: (4 נקודות)

```
public static bool IsPrimary(int num)
{
    if (num < 2) return false;
    for (int i = 2; i * i <= num; i++)
        if (num % i == 0) return false;
    return true;
}
```

סעיף ב: (4 נקודות)

```
public static int CountPrimaryPairs(int num)
{
    int count = 0;
    for (int i = 1; i <= num / 2; i += 2)
    {
        if (IsPrimary(i) && IsPrimary(num - i))
            count++;
    }
    return count;
}
```

סעיף ג: (4 נקודות)

```
public static int[] AllPrimaryPairs(int num)
{
    int count = CountPrimaryPairs(num);
    int[] result = new int[count * 2];
    int index = 0;
    for (int i = 1; i <= num / 2; i += 2)
    {
        int j = num - i;
        if (IsPrimary(i) && IsPrimary(j))
        {
            result[index++] = i;
            result[index++] = j;
        }
    }
    return result;
}
```

סעיף ד: (3 נקודות)

prime - $O(\sqrt{n})$

countprimepairs = $O(n \cdot \sqrt{n})$

allPrimaryPairs = $O(n \cdot \sqrt{n})$

פתרון שאלה 6 (15 נקודות)

סעיף א: (3 נקודות)

```
public class Warehouse
{
    private Flag[] flags = new Flag[100];
    private int[] quantities = new int[100];
    private int currentFlags = 0;
}
```

סעיף ב: (4 נקודות)

```
public bool Equals(Flag other)
{
    return this.Country == other.Country &&
           this.Length == other.Length &&
           this.Width == other.Width;
}
public void Add(string country, int length, int width, int quant)
{
    Flag newFlag = new Flag(country, length, width);
    for (int i = 0; i < currentFlags; i++)
    {
        if (flags[i].Equals(newFlag))
        {
            quantities[i] += quant;
            return;
        }
    }

    if (currentFlags >= 100)
    {
        Console.WriteLine("Warehouse is full. Cannot add new flag.");
        return;
    }

    flags[currentFlags] = newFlag;
    quantities[currentFlags] = quant;
    currentFlags++;
}
```

סעיף ג: (4 נקודות)

```
public void FlagsWithMinQuantity(int minQuantity)
{
    Console.WriteLine($"Flags with quantity less than {minQuantity}:");
    for (int i = 0; i < currentFlags; i++)
    {
        if (quantities[i] < minQuantity)
            Console.WriteLine(flags[i] + " - Quantity: " + quantities[i]);
    }
}
```

סעיף ד: (4 נקודות)

```
public void PrintFlags(string[] countries)
{
    Console.WriteLine("Total flag quantities per country:");
    foreach (string country in countries)
    {
        int total = 0;
        for (int i = 0; i < currentFlags; i++)
            if (flags[i].Country == country)
                total += quantities[i];
        Console.WriteLine($"{country}: {total} flag(s)");
    }
}
```

פתרון שאלה 7 (15 נקודות)

סעיף א: (5 נקודות)

1.

קריאה	num	s (sum of digits)	c (count of digits)
1	6123	$0 + 3 = 3$	$0 + 1 = 1$
2	612	$3 + 2 = 5$	$1 + 1 = 2$
3	61	$5 + 1 = 6$	$2 + 1 = 3$
4	6	$6 + 6 = 12$	$3 + 1 = 4$
5	0		

return $12 / 4 = 3.0$

2.

5546

תחזיר 5

3.

ממוצע ספרות

סעיף ב (10 נקודות)

1.

int[] a = {24, 126, 9, 35, 2684, 8941};

int p = 3; // one(a[3]) = one(35)

index k	value	one(value)	one(p) = 4.0	condition	temp location
0	24	$(2+4)/2 = 3.0$	< 4.0	left $\leftarrow 0$	temp[0] = 24
1	126	$(1+2+6)/3 = 3.0$	< 4.0	left $\leftarrow 1$	temp[1] = 126
2	9	$9/1 = 9.0$	> 4.0	right $\leftarrow 5$	temp[5] = 9
3	35	4.0	$=$	left $\leftarrow 2$	temp[2] = 35
4	2684	$2+6+8+4 = 20 / 4 = 5.0$	> 4.0	right $\leftarrow 4$	temp[4] = 2684
5	8941	$8+9+4+1 = 22 / 4 = 5.5$	> 4.0	right $\leftarrow 3$	temp[3] = 8941

temp = {24, 126, 35, 8941, 2684, 9}

2.

כן, שגיאה תתרחש אם p לא נמצא בטווח התקני של האינדקסים במערך כלומר $p < 0$ או $p \geq a.length$

3.

int[] arr = {111, 111, 111, 111, 111, 111};

int[] arr = {123, 114, 3320, 141, 312, 213};

4. הפעולה three(arr, p) מחזירה מערך חדש, שבו כל המספרים ממוקמים מחדש בהתאם לממוצע הספרות של כל מספר בהשוואה לממוצע הספרות של המספר שנמצא במקום p במערך. הקטנים לפיו והגדולים אחריו.

פתרון שאלה 8 (17 נקודות)

```
public static bool Exist(char[][] arr, string word)
{
    int rows = arr.Length;
    int cols = arr[0].Length;
    int len = word.Length;

    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < cols; j++)
        {
            if (CheckRight(arr, word, i, j, cols, len) ||
                CheckLeft(arr, word, i, j, len) ||
                CheckDown(arr, word, i, j, rows, len) ||
                CheckUp(arr, word, i, j, len))
            {
                return true;
            }
        }
    }

    return false;
}

private static bool CheckRight(char[][] arr, string word, int i, int j, int cols, int len)
{
    if (j + len > cols) return false;
    for (int k = 0; k < len; k++)
        if (arr[i][j + k] != word[k]) return false;
    return true;
}

private static bool CheckLeft(char[][] arr, string word, int i, int j, int len)
{
    if (j - len + 1 < 0) return false;
    for (int k = 0; k < len; k++)
        if (arr[i][j - k] != word[k]) return false;
    return true;
}

private static bool CheckDown(char[][] arr, string word, int i, int j, int rows, int len)
{
    if (i + len > rows) return false;
    for (int k = 0; k < len; k++)
        if (arr[i + k][j] != word[k]) return false;
    return true;
}

private static bool CheckUp(char[][] arr, string word, int i, int j, int len)
{
    if (i - len + 1 < 0) return false;
    for (int k = 0; k < len; k++)
        if (arr[i - k][j] != word[k]) return false;
    return true;
}
```

פתרון נוסף של שאלה 8

```
public static string RowToString(char[][] arr, int row)
{
    string s = "";
    for (int j = 0; j < arr[row].Length; j++)
        s += arr[row][j];
    return s;
}
public static string ColToString(char[][] arr, int col)
{
    string s = "";
    for (int i = 0; i < arr.Length; i++)
        s += arr[i][col];
    return s;
}
public static string Reverse(string str)
{
    string s = "";
    for (int i = 0; i < str.Length; i++)
        s = str[i] + s;
    return s;
}

public static bool Exist(char[][] arr, string word)
{
    string reverseWord = Reverse(word);
    for (int i = 0; i < arr.Length; i++)
    {
        string rowString = RowToString(arr, i);
        if (rowString.Contains(word) || rowString.Contains(reverseWord))
            return true;
    }

    for (int j = 0; j < arr[0].Length; j++)
    {
        string colString = ColToString(arr, j);
        if (colString.Contains(word) || colString.Contains(reverseWord))
            return true;
    }

    return false;
}
```

פתרון שאלה 9 (17 נקודות)

סעיף א: (2 נקודות)

```
public Account(string username, string initialPass)
{
    this.username = username;
    this.currentPass = initialPass;
    this.passHistory[0] = initialPass;
    historySize = 1;
}
```

סעיף ב: (4 נקודות)

```
public bool IsPasswordValid(string password)
{
    int criteriaMet = 0;

    if (password.Length >= 8)
        criteriaMet++;

    if (password.Count(char.IsUpper) >= 2)
        criteriaMet++;

    if (password.Any(char.IsDigit))
        criteriaMet++;

    if (password.Any(c => !char.IsLetterOrDigit(c)))
        criteriaMet++;

    bool noRepeats = true;
    for (int i = 1; i < password.Length; i++)
    {
        if (password[i] == password[i - 1])
        {
            noRepeats = false;
            break;
        }
    }
    if (noRepeats)
        criteriaMet++;

    return criteriaMet >= 3;
}
```

סעיף ג: (4 נקודות)

```
public void UpdatePassword(string newPass)
{
    if (!IsPasswordValid(newPass))
    {
        Console.WriteLine("הסיסמה החדשה אינה עומדת בדרישות האבטחה.");
        return;
    }
    if (newPass == currentPass)
    {
        Console.WriteLine("הסיסמה החדשה זהה לסיסמה הנוכחית");
        return;
    }
    for (int i = 0; i < historySize; i++)
    {
        if (passHistory[i] == newPass)
        {
            Console.WriteLine("הסיסמה כבר הייתה בשימוש בעבר");
            return;
        }
    }
}
```



```

        for (int i = Math.Min(historySize, 9); i > 0; i--)
            passHistory[i] = passHistory[i - 1];

        passHistory[0] = newPass;
        currentPass = newPass;
        if (historySize < 10) historySize++;

        Console.WriteLine("הסיסמה עודכנה בהצלחה.");
    }

```

סעיף ד: (3 נקודות)

```

public void AddUser(string username, string password)
{
    if (numOfAccounts >= 100)
    {
        Console.WriteLine("לא ניתן להוסיף משתמש נוסף - המערכת מלאה.");
        return;
    }

    for (int i = 0; i < numOfAccounts; i++)
    {
        if (accounts[i].GetUsername() == username)
        {
            Console.WriteLine("שם המשתמש כבר קיים במערכת.");
            return;
        }
    }

    accounts[numOfAccounts++] = new Account(username, password);
    Console.WriteLine("משתמש נוסף בהצלחה.");
}

```

סעיף ה: (4 נקודות)

- דרישה ל-4 מתוך 5 תנאים ישנים במקום 3 מ 5
- ובנוסף חובה שתהיה לפחות אות קטנה אחת

```
public bool IsPasswordValid(string password)
{
    int criteriaMet = 0;
    // תנאי 1: אורך  $\geq 8$ 
    if (password.Length >= 8)
        criteriaMet++;
    // תנאי 2: לפחות 2 אותיות גדולות
    int upperCount = 0;
    foreach (char c in password)
    {
        if (c >= 'A' && c <= 'Z')
            upperCount++;
    }
    if (upperCount >= 2)
        criteriaMet++;
    // תנאי 3: לפחות ספרה אחת
    bool hasDigit = false;
    foreach (char c in password)
    {
        if (c >= '0' && c <= '9')
        {
            hasDigit = true;
            break;
        }
    }
    if (hasDigit)
        criteriaMet++;
    // תנאי 4: לפחות תו מיוחד
    bool hasSpecial = false;
    foreach (char c in password)
    {
        if (!((c >= 'A' && c <= 'Z') ||
              (c >= 'a' && c <= 'z') ||
              (c >= '0' && c <= '9'))))
        {
            hasSpecial = true;
            break;
        }
    }
    if (hasSpecial)
        criteriaMet++;
    // תנאי 5: אין תווים זהים צמודים
    bool noRepeats = true;
    for (int i = 1; i < password.Length; i++)
    {
        if (password[i] == password[i - 1])
        {
            noRepeats = false;
            break;
        }
    }
    if (noRepeats)
        criteriaMet++;
    // תנאי חובה: לפחות אות קטנה אחת
    bool hasLowercase = false;
    foreach (char c in password)
    {
        if (c >= 'a' && c <= 'z')
        {
            hasLowercase = true;
            break;
        }
    }
    return (criteriaMet >= 4 && hasLowercase);
}
```

פתרון שאלה 10 (17 נקודות)

arr = [10, 3, 6, 8, 2, 11, 5];

סעיף א (3 נקודות)

mystery(arr, 1, 4)

מצב לפני:

[10, 3, 6, 8, 2, 11, 5]

תהליך:

היפוך טווח [3, 6, 8, 2] → [2, 8, 6, 3]

תוצאה:

[10, 2, 8, 6, 3, 11, 5]

סעיף ב (3 נקודות)

secret(arr, 3)

מצב לפני:

[10, 3, 6, 8, 2, 11, 5]

שלבים:

mystery(0,6) → [5, 11, 2, 8, 6, 3, 10]

mystery(0,2) → [2, 11, 5, 8, 6, 3, 10]

mystery(3,6) → [2, 11, 5, 10, 3, 6, 8]

תוצאה :

[2, 11, 5, 10, 3, 6, 8]

סעיף ג (2 נקודות)

הסבר לפעולה :

secret(a, d)

הפונקציה מבצעת סיבוב שמאלה של המערך ב־ d מקומות. (left rotation)

היא עושה זאת בעזרת שלושה היפוכים:

היפוך של כל המערך

היפוך של החלק הראשון 0 עד d

היפוך של החלק השני d עד הסוף

סעיף ד (3 נקודות)

– what(arr, 3)

על המערך ההתחלתי

מצב לפני:

[10, 3, 6, 8, 2, 11, 5]

שלבי הרקורסיה: (why)

i j = (i+3)%7 rest = a[j]

0 3 8

1 4 2

2 5 11

3 6 5

4 0 10

5 1 3

6 2 6

שלב חזור – תוצאה:

[8, 2, 11, 5, 10, 3, 6]

סעיף ה (3 נקודות)

לאחר what(brr, 2) מתקבל:

[60, 50, 40, 30, 20, 10]

המערך שעליו פעל:

[40, 30, 20, 10, 60, 50]

סעיף ו (3 נקודות)

מה עושה

What(a,d)

הפעולה יוצרת סיבוב ימינה ב-d מקומות במערך