פרק 12 – תבניות אלגוריתמיות

הצגת הפרק

מטרת הפרק

הצגת מבנה הנתונים מערך דו-מימדי. הכרת האלגוריתם לחיפוש בינרי, ואלגוריתמים שונים למיון של סדרת נתונים.

מושגים

- מערך דו-מימדי
- מערך דו-מימדי ריבועי
- אלכסון ראשי
- אלכסון משני
- חיפוש בינרי
- מיון בחירה
- מיון הכנסה
- מיון בועות
- מיזוג

דגשים ודידקטיקה

- בפרק זה רוב השאלות עוסקות במערך דו-מימדי כתכונה בתוך מחלקה. ניתן לעסוק בנושא גם ללא מחלקות כלל, כשם שנלמד הנושא מערך חד-מימדי (תוכנית ראשית בלבד).
- מומלץ להדגיש בפני התלמידים מדוע ישנם מספר אלגוריתמי מיון, וכן להדגיש את חשיבות נושא המיון במדעי המחשב.

פתרונות

שאלה 12.1

בנו מחלקה ובה מערך דו-ממדי המכיל מספרים שלמים. המחלקה תכיל את הפעולות הבאות:

א. פעולה בונה המקבלת את ממדי המערך הדו-ממדי ומקצה עבורו מקום.

ב. פעולת גישה לאתחול תא במערך הדו-ממדי. הפעולה מקבלת מספר שורה, מספר עמודה ומספר שלם ומעדכנת את התא המתאים.

ג. פעולה המקבלת מספר שורה ומחזירה את סכום איברי השורה.

ד. פעולה המציגה את ערכי האיברים הנמצאים בשורות הזוגיות.

ה. פעולה המקבלת מספר שורה ומחזירה "אמת" אם כל איברי השורה מתחלקים ב-3.

תשובה 12.1

להלן המחלקה המבוקשת, ובה מערך דו-מימדי: (כפי שנאמר בתחילת הפרק, יש אפשרות לבצע פעולות אלה בתוך בתוכנית הראשית)

```
public class Quest1
{
    private int[][] mat;
    // פעולה בונה
    public Quest1(int rows, int cols)
    {
        mat = new int[rows][cols];
    }
    // עדכון תא בודד
    public void setCell(int row, int col, int val)
    {
        mat[row][col] = val;
    }
    // סכום שורה
    public int sumRow(int row)
    {
        int sum = 0;
        for (int i = 0; i < mat[row].length; i++)
            sum += mat[row][i];
        return sum;
    }
    // הדפסת שורות זוגיות
    public void evenElement()
    {
        int i = 0;
        while (i < mat.length)
        {
            for (int j = 0; j < mat[i].length; j++)
                System.out.print(mat[i][j]);
            System.out.println();
            i += 2;
        }
    }
    // האם כל ערכי השורה מתחלקים בשלוש
    public boolean isThree(int row)
    {
```

```
        for (int i = 0; i < mat[row].length; i++)
        {
            if (mat[row][i] % 3 != 0)
                return false;
        }
        return true;
    }
}
```

שאלה 12.2

א. פתחו אלגוריתם המאחסן בכל תא של מערך דו-ממדי מספר אשר ספרת העשרות שלו שווה למספר השורה של התא, וספרת האחדות שלו שווה למספר העמודה של התא. למשל, מערך דו-ממדי בגודל 3x4 יראה כך:

0	1	2	3
10	11	12	13
20	21	22	23

ב. הגדירו מחלקה הכוללת מערך דו-ממדי כתכונה ואת הפעולות הבאות:

- פעולה בונה המקבלת את ממדי המערך, מקצה עבורו מקום ומיישמת את האלגוריתם שפיתחתם בסעיף הקודם.
- פעולה להצגת תוכן המערך הדו-ממדי בצורת טבלה.

ג. הגדירו פעולה ראשית הקולטת מהמשתמש את ממדי המערך הרצוי, מייצרת עצם מהסוג שהוגדר בסעיף הקודם ומציגה את המערך שנוצר.

תשובה 12.2

להלן המחלקה המבוקשת

```
public class Quest2
{
    private int[][] mat;

    // פעולה בונה המשתמשת באלגוריתם המבוקש
    public Quest2(int rows, int cols)
    {
        mat = new int[rows][cols];
        for(int i = 0; i < rows; i++)
            for(int j = 0; j < cols; j++)
                mat[i][j] = i*10 + j;
    }
    // פעולה להצגת המטריצה כטבלה
    public void print()
    {
        for(int i = 0; i < mat.length; i++)
        {
            for(int j = 0; j < mat[i].length; j++)
                System.out.print(mat[i][j] + " ");
            System.out.println();
        }
    }
}
```

שאלה 12.4

פתחו וישמו אלגוריתם לניהול הזמנת מקומות במטוס. הקלט הוא סדרה של בקשות להזמנת מושבים במטוס. כל בקשה מורכבת ממספר שורה ומאות המייצגת את המושב ('a', 'b', 'c', ...). לאחר כל בקשה תוצג הודעה: "בקשתך התקבלה" או "המקום שביקשת תפוס". לאחר סיום העיבוד של כל בקשה יוצג מצב התפוסה של כל המושבים במטוס, כאשר A מסמן מושב פנוי ו-N מסמן מושב תפוס. הקלט יסתיים כאשר יוקלד המושב A 1 (והוא שייך כידוע לדיילת הראשית).

הדרכה: הגדירו מחלקה המייצגת מטוס. המחלקה תכלול מערך דו-ממדי בוליאני המייצג את המושבים ואת הפעולות הבאות:

א. פעולה בונה המקבלת את ממדי המטוס (מספר שורות ומספר מושבים בכל שורה). הפעולה תקצה את מערך המושבים ותאתחל את כל המושבים כך שיהיו פנויים.

ב. פעולת הזמנה המקבלת מספר שורה ואות המייצגת את המושב. הפעולה מסמנת את המקום כתפוס ומחזירה ערך בוליאני המציין את הצלחת הפעולה או את כישלונה.
<u>זכרו</u>: כדי להמיר את התווים למקומות במערך ('a' הוא 0, 'b' הוא 1 וכו') יש לחסר מהתו המבוקש את התו 'a'. כלומר אם התו מאוחסן במשתנה ch, נכתוב: 'a'-ch. חשבו מדוע! (ראו פרק 4).

ג. פעולה להצגת מצב מושבי המטוס.

הגדירו פעולה ראשית הכוללת את קליטת ממדי המטוס, את יצירת העצם מטוס, את קליטת הזמנות המשתמשים ואת ביצוע ההזמנות.

תשובה 12.4

המחלקה לניהול מושבי המטוס:

```
public class Quest4
{
    private boolean[][] plane;
    public Quest4(int rows, int cols)
    {
        plane = new boolean[rows][cols];
        for(int i = 0; i < rows; i++)
            for(int j = 0; j < cols; j++)
                plane[i][j] = false;
    }
    // פעולה לתפיסת מקום, שבודקת לפני האם המקום פנוי
    public boolean request(int row, char sit)
    {
        if(plane[row-1][sit-'a']==false)
        {
            plane[row-1][sit-'a'] = true;
            return true;
        }
        return false;
    }
    // הצגת מושבי המטוס
    public void print()
    {
        for(int i = 0; i < plane.length; i++)
        {
            for(int j = 0; j < plane[i].length; j++)
            {
```

```
                if (plane[i][j])
                        System.out.print('N' + " ");
                else
                        System.out.print('A' + " ");
            }
            System.out.println();
        }
    }
}
```

יישום המחלקה הראשית המנהלת את בקשות הנוסעים:

```
import java.util.Scanner;
public class Quest4Test
{
    public static void main(String[] args)
    {
        Scanner in = new Scanner(System.in);
        // הקצאת מקום לפי גודל המטוס
        Quest4 plane = new Quest4(10,5);
        plane.print();
        System.out.print("Enter the place: ");
        int row = in.nextInt();
        char sit = in.next().charAt(0);
        while(row!=1 || sit!='A')
        {
            System.out.println("You asked for place " + row + sit);
            if(plane.request(row, sit))
                System.out.println("OK");
            else
                System.out.println("This place is occupied");
            plane.print();
            System.out.print("Enter the place: ");
            row = in.nextInt();
            sit = in.next().charAt(0);
        }
        System.out.println("The End");
        plane.print();
    }
}
```

שאלה 12.5

חגית משחקת שש-בש עם לירון בכל יום בשבוע, בכל יום 10 משחקים. תוצאות המשחקים נשמרות במערך דו-ממדי בוליאני בגודל 7×10: כל שורה במערך מייצגת יום בשבוע, השורה תכיל את תוצאות עשרת המשחקים ששיחקו באותו היום. למשל אם חגית ניצחה במשחק השני ביום חמישי אז התא השני בשורה החמישית יכיל את הערך `true`, אם לירון ניצח באותו משחק, תא זה יכיל את הערך `false`. פתחו אלגוריתם שיחשב ויציג:

א. למי מהשניים כמות הניצחונות הגדולה ביותר.

ב. הודעה המציינת את יום המזל של חגית (**היום** שבו לחגית היו הכי הרבה ניצחונות). אם יש יותר מיום אחד כזה ההודעה תציין את היום הראשון שנמצא.

ג. הודעה המציינת את משחק המזל של לירון (מכיוון שביום מתקיימים עשרה משחקים נמספר אותם מ-1 עד 10. **משחק המזל** הוא זה שמספר הניצחונות בו במהלך השבוע הוא הגדול ביותר). אם יש יותר ממשחק אחד כזה ההודעה תציין את המשחק הראשון שנמצא.

הדרכה: הגדירו מחלקה המייצגת את תוצאות המשחקים. המחלקה תכלול מערך דו-ממדי בוליאני, ואת הפעולות הבאות:

- פעולה בונה ליצירת מערך התוצאות.
- פעולת גישה לעדכון תוצאה של משחק מסוים.

פעולות הבודקות: למי יש יותר ניצחונות, מהו יום המזל של חגית, ומהו משחק המזל של לירון. הפעולות מחזירות ערכים בהתאם.

תשובה 12.5

רעיון הפתרון:

נבנה מחלקה אשר תמנה את הניצחונות של המשחקים. על מחלקה זו לשמור את כל משחקי השבוע: 7 ימים בשבוע, 10 משחקים בכל יום. לשם כך נשתמש במערך דו-מימדי בוליאני, בגודל 7x10. תא אשר יכיל את הערך true פירושו שחגית ניצחה במשחק זה. כדי לבדוק את יום המזל של חגית (סעיף ב') מתאים להשתמש בתבנית **מציאת מקסימום**. הסדרה בה נחפש את המקסימום היא כמות הניצחונות עבור שורה במטריצה, אותם נמנה על ידי שימוש בתבנית **מניה**. כדי למצוא את משחק המזל של לירון מתאים להשתמש בתבנית **מציאת מקסימום**. הסדרה בה נחפש את המקסימום היא כמות הניצחונות עבור עמודה במטריצה, אותם נמנה על ידי שימוש בתבנית **מניה**.

יישום המחלקה ופעולותיה העיקריות:

```
public class Quest5
{
    private boolean[][]wins;
    public Quest5()
    {
        wins = new boolean[7][10];
    }
    public int maxWin()
    {
        int counterTrue = 0;  // משתנה המונה את ניצחונותיה של חגית
        for(int i = 0; i < wins.length; i++)
            for(int j = 0; j < wins[i].length; j++)
                if(wins[i][j])
                    counterTrue++;

        if(counterTrue > (wins.length * wins[i].length) / 2)
            return 1;
        return 0;
    }
    public int dayOfLuck()
    {
        int sumDay;
        int max = 0;
        int maxDay = 0;
        for(int i = 0; i < wins.length; i++)
        {
            sumDay = 0;
```

```
            for(int j = 0; j < wins[i].length; j++)
            {
                if(wins[i][j])
                    sumDay++;
            }
            // בתום סכימת השורה בודקים אם היא מקסימלית
            if(sumDay > max)
            {
                max = sumDay;
                maxDay = i+1;
            }
        }
        return maxDay;
    }
    public int gameOfLuck()
    {
        int sumGame;
        int max = 0;
        int maxGame = 0;
        for(int i = 0; i < wins[0].length; i++)
        {
            sumGame = 0;
            for(int j = 0; j < wins.length; j++)
            {
                if(!wins[j][i])
                    sumGame++;
            }
            // בתום סכימת העמודה בודקים אם היא מקסימלית
            if(sumGame > max)
            {
                max = sumGame;
                maxGame = i+1;
            }
        }
        return maxGame;
    }
}
```

שאלה 12.6

לפניכם כמה קטעי תוכניות בשפת Java, עבור כל אחד מהקטעים כתבו מה יוצג על המסך ובאיזו צורה.

הניחו ש-mat מוגדרת כמטריצה ריבועית בגודל 5×5 והיא מכילה את הערכים הבאים:

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

```
for(int i = 0; i < mat.length; i++)
{
```

```
for(int i = 0; i < mat.length; i++)
{
```

`System.out.print(mat[2][i]);` `}`	`System.out.print(mat[i][2]);` `}`

`for(int i = 0; i < mat.length; i++)`
`{`
`System.out.print(mat[5][5]);`
`}`
	`for(int i = 0; i < mat.length; i++)`
	`{`
	`System.out.print(mat[i][i]);`
	`}`

תשובה 12.6

תוצג השורה השלישית במטריצה, כלומר: 11 12 13 14 15	תוצג העמודה השלישית במטריצה, כלומר: 3 8 13 18 23
חריגה מגבולות המערך, התוכנית לא תרוץ.	יוצג האלכסון הראשי במטריצה, כלומר: 1 7 13 19 25

שאלה 12.7

פתחו וישמו אלגוריתם הבודק לגבי מטריצה ריבועית אם היא:

א. "שוות שורות" – מטריצה "שוות שורות" היא מטריצה שסכום כל שורה בה שווה.

ב. "שוות עמודות" – מטריצה "שוות עמודות" היא מטריצה שסכום כל עמודה בה שווה.

ג. "ריבוע קסם" – "ריבוע קסם" הוא מטריצה "שוות שורות" ו"שוות עמודות" ובנוסף סכום כל שורה שווה לסכום כל עמודה, וסכום כל אלכסון שווה לסכום שורה ולסכום עמודה. הנה דוגמה לריבוע קסם:

8	1	6
3	5	7
4	9	2

הדרכה: הגדירו מחלקה המכילה מערך דו-ממדי של שלמים ואת הפעולות הבאות: פעולה בונה ליצירת המטריצה הריבועית, פעולת גישה לאתחול תא במטריצה ושלוש פעולות בוליאניות הבודקות אם המטריצה היא שוות שורות, שוות עמודות וריבוע קסם.

שימו ♥: לצורך בדיקת ריבוע קסם, ניתן להיעזר בפעולות הקודמות שמימשתם.
הגדירו פעולה ראשית הקולטת מספרים עבור מטריצה בגודל 3×3 ובודקת אם המטריצה היא שוות שורות, אם היא שוות עמודות ואם היא ריבוע קסם.

תשובה 12.7

רעיון הפתרון: בפתרון שאלה זו נשתמש בתבנית "**האם כל הערכים בסדרה מקיימים תנאי**", כיוון שעלינו לבדוק האם כל השורות/העמודות הן בעלות סכום שווה.

```
public class Quest7
{
    private int[][]mat;

    public Quest7(int size)
    {
        mat = new int[size][size];
    }
    // פעולה המחשבת סכום שורה מבוקשת
    public int sumRow(int row)
```

```
{
    int sum = 0;
    for(int i = 0; i < mat.length; i++)
        sum+= mat[row][i];
    return sum;
}
// פעולה המחשבת סכום עמודה מבוקשת
public int sumCol(int col)
{
    int sum = 0;
    for(int i = 0; i < mat.length; i++)
        sum+= mat[i][col];
    return sum;
}
// פעולה שמחזירה "אמת" אם סכום כל השורות שווה, ו"שקר" אחרת
public boolean isEqualsRow()
{
    int firstRow = sumRow(0);
    for(int i = 1; i < mat.length; i++)
    {
        if(sumRow(i) != firstRow)
            return false;
    }
    return true;
}
// פעולה שמחזירה "אמת" אם סכום כל העמודות שווה, ו"שקר" אחרת
public boolean isEqualsCol()
{
    int firstCol = sumCol(0);
    for(int i = 1; i < mat.length; i++)
    {
        if(sumCol(i) != firstCol)
            return false;
    }
    return true;
}
// פעולה המחשבת סכום אברי אלכסון ראשי
public int sumMainDiagonal()
{
    int sum = 0;
    for(int i = 1; i < mat.length; i++)
    {
        sum+= mat[i][i];
    }
    return sum;
}
// פעולה המחשבת סכום אברי אלכסון משני
public int sumSecondDiagonal()
{
    int sum = 0;
    for(int i = 1; i < mat.length; i++)
    {
        sum+= mat[i][mat.length-i-1];
```

```
        }
        return sum;
    }
    // פעולה שמחזירה "אמת" אם זהו ריבוע קסם, ו"שקר" אחרת
    public boolean magicSquare()
    {
        int firstRow = sumRow(0);
        //  את השורה הבאה אפשר לבצע גם באמצעות תנאי
        return(isEqualsCol() && isEqualsRow() &&
               sumMainDiagonal() == firstRow &&
               sumSecondDiagonal()== firstRow);
    }
}
```

שאלה 12.9

עלינו לפתח מערכת ממוחשבת לרכישת כרטיסי קולנוע. בבית הקולנוע יש שני אולמות: אולם שביט ואולם ארמון. באולם שביט מוצג בימים אלה הסרט מלחמת הכוכבים ובאולם ארמון מוצג בימים אלה הסרט שרק. בכל אולם יש 10 שורות, ו-20 מושבים בכל שורה.

המערכת מציגה למשתמש את הסרטים שמוצגים בבית הקולנוע בימים אלה ומבקשת ממנו לציין את שם הסרט שהוא רוצה לראות. לאחר מכן המערכת שואלת את המשתמש כמה כרטיסים הוא רוצה לרכוש. המערכת מציגה לפני המשתמש את המושבים הפנויים ומבקשת ממנו לבחור מושבים.

בסוף ההזמנה המערכת מדפיסה: נרכשו <מספר כרטיסים> כרטיסים לסרט <שם הסרט> המוצג באולם <שם האולם שהסרט מציג בו>

לפניכם הגדרת המחלקה Cinema. השלימו את המקומות החסרים, והוסיפו מחלקה ראשית שתטפל בפעולות הקולנוע:

```
public class Cinema
{
    private String name;   // שם אולם הקולנוע
    private String movie;  // שם הסרט המוצג
    private boolean[][] freeSeats;   // רשימת מושבי האולם ומצבם
    private final int ROWS = 10;
    private final int COLS = 20;
    // הפעולה הבונה
    public Cinema (String aName, String aMovie)
    {
        name = aName;
        movie = aMovie;
        freeSeats = new boolean[ROWS][COLS];
        // מסמנים את כל המושבים כפנויים
        for (int i = 0; i < ROWS; i++)
             for (int j = 0; j < COLS; j++)
                  _____________________________;
    }
    // פעולת גישה: מחזירה את שם הסרט המוצג באולם
    public String getMovie ()
    {
        return ____________________;
    }
    // פעולת גישה: מחזירה את שם אולם הקולנוע
```

```
    public String getName()
    {
        return ____________________;
    }
    // פעולת גישה: מעדכנת את שם הסרט המוצג באולם
    public void setMovie(String aMovie)
    {
        ____________________________;
    }
    // פעולה בוליאנית הבודקת אם סרט נתון מוצג באולם הקולנוע
    public boolean isShowing(String aMovie)
    {
        return (____________________________)
    }
    // פעולה המחזירה מחרוזת שמתארת את רשימת המושבים הפנויים
    public String getAvailableSeats()
    {
        String availableSeats = new String("");
        for (int i = 0; i <freeSeats.length; i++)
             for (int j = 0; j <freeSeats[i].length; j++)
                  if (____________________________)
                        availableSeats = availableSeats +
                                          ____________________________________;
         return availableSeats;
    }
    // פעולה המקבלת מספר מושב פנוי ומעדכנת את מצבו כתפוס
    // ומחזירה "אמת" אם הפעולה הצליחה ו"שקר" אחרת
    public boolean setSeatAsTaken(int rowSeat, int colSeat)
    {
        if (____________________)
        {
            _____________________________ = false;
            return _________________;
        }
        else
            return _________________;
      }
} // class Cinema
```

תשובה 12.9

```
public class Cinema
{
    private String name;   // שם אולם הקולנוע
    private String movie;  // שם הסרט המוצג
    private boolean[][] freeSeats;   // רשימת מושבי האולם ומצבם
    private final int ROWS = 10;
    private final int COLS = 20;
    // הפעולה הבונה
    public Cinema (String aName, String aMovie)
    {
        name = aName;
```

```
        movie = aMovie;
        freeSeats = new boolean[ROWS][COLS];
        // מסמנים את כל המושבים כפנויים
        for (int i = 0; i < ROWS; i++)
             for (int j = 0; j < COLS; j++)
                  freeSeats[i][j] = 0;
    }
    // פעולת גישה: מחזירה את שם הסרט המוצג באולם
    public String getMovie ()
    {
        Return movie;
    }
    // פעולת גישה: מחזירה את שם אולם הקולנוע
    public String getName()
    {
        return name;
    }
    // פעולת גישה: מעדכנת את שם הסרט המוצג באולם
    public void setMovie(String aMovie)
    {
        Movie = aMovie;
    }
    // פעולה בוליאנית הבודקת אם סרט נתון מוצג באולם הקולנוע
    public boolean isShowing(String aMovie)
    {
        return (aMovie == movie)
    }
    // פעולה המחזירה מחרוזת שמתארת את רשימת המושבים הפנויים
    public String getAvailableSeats()
    {
        String availableSeats = new String("");
        for (int i = 0; i < freeSeats.length; i++)
             for (int j = 0; j < freeSeats[i].length; j++)
                  if (freeSeats[i][j])    // או: freeSeats[i][j]==true
                       availableSeats = availableSeats +
                                                 i + " " + j + "\n";
         return availableSeats;
    }
    // פעולה המקבלת מספר מושב פנוי ומעדכנת את מצבו כתפוס
    // ומחזירה "אמת" אם הפעולה הצליחה ו"שקר" אחרת
    public boolean setSeatAsTaken(int rowSeat, int colSeat)
    {
            if (freeSeats[rowSeat][colSeat])
            {
                  freeSeats[rowSeat][colSeat] = false;
                  return true;
            }
            else
                  return false;
     }
} // class Cinema
```

שאלה 12.13

לפניכם קטע קוד שמבצע חיפוש בינרי במערך arr:

הניחו כי המשתנים high ו-low מאותחלים בגבולות המערך 0 ו-N.

```
boolean flag = true;
while (low <= high)
{
      middle = (low + high) / 2;
      if (arr[middle] != num)
            flag = false;
      if (arr[middle] > num)
            high = middle - 1;
      else
            low = middle + 1;
}
if (flag)
      System.out.println("The Value Is In The Array");
else
      System.out.println("The Value Is Not In The Array");
```

הקטע שגוי. מצאו את השגיאה ותקנו את הקטע כך שיבצע את הנדרש.

תשובה 12.13

האלגוריתם המוצע מניח כי הערך אותו מחפשים קיים במערך, ובודק האם הערך התורן (הערך האמצעי בין גבולות החיפוש) **אינו** שווה לערך אותו מחפשים, ומשנה את ערך המשתנה הבוליאני flag בהתאם. לא קיימת בדיקה האם הערך שאנו מחפשים אכן נמצא במערך. ברגע שהערך הראשון שייבדק ימצא כשגוי (אינו שווה לערך אותו מחפשים) ישתנה המשתנה הבוליאני להיות "שקר" ויישאר להיות כזה עד לסיום האלגוריתם. האלגוריתם הנכון לשאלה זו הוא להניח כי הערך אותו מחפשים אינו קיים במערך, ובמידה ובמהלך החיפוש נמצא ערך זה, ישתנה המשתנה להיות "אמת". הנה התיקון:

```
boolean flag = false;
while (low <= high  && !flag)
{
      middle = (low + high) / 2;
      if (arr[middle] == num)
            flag = true;
      else if (arr[middle] > num)
            high = middle - 1;
      else
            low = middle + 1;
}
if (flag)
      System.out.println("The Value Is In The Array");
else
      System.out.println("The Value Is Not In The Array");
```

שאלה 12.14

נתון מערך ממוין A. כתבו קטע תוכנית בשפת Java שיבדוק כמה פעמים מופיע הערך 25 במערך A. כתבו את הקטע יעיל ככל שתוכלו.

תשובה 12.14

רעיון הפתרון:

אם הערך 25 נמצא במערך יותר מפעם אחת, כל הפעמים יופיעו ברצף כיוון שהמערך ממוין. לכן, נשתמש בחיפוש בינרי כדי למצוא האם המספר 25 נמצא במערך. אם הוא נמצא נצעד ימינה ושמאלה החל ממקום זה עד אשר נפגוש במספר שונה מ-25.

```
low = 0;
high = a.length-1;

boolean found = false;
while (low <= high && !found)
{
     middle = (low + high) / 2;
     if (a[middle] == 25)
     {
         found = true;
         place = middle;
     }
     else if (a[middle] > 25)
          high = middle - 1;
     else
         low = middle + 1;
}
if(!found)
   System.out.println("The number 25 is not in the array");
else
{
    int counter = 1;
    int curr = place + 1;
    while(curr < a.length && a[curr]==25)
    {
       counter++;
       curr++;
    }
    curr = place - 1;
    while(curr >= 0 && a[curr]==25)
    {
        counter++;
        curr--;
    }
    System.out.println("The number 25 appear: " + counter +
                                                        " times ");
}
```

שאלה 12.16

נתון מערך ממוין A בגודל N ובו מספרים שלמים חיוביים (בסדר עולה). כתבו קטע תוכנית שהפלט שלו הוא הודעה אם רוב איברי המערך גדולים מן הממוצע של האיבר הראשון והאחרון.

תשובה 12.16

מומלץ לפתור שאלה זו בכיתה, כדי לתת לתלמידים לנסות ולהציע דרכים שונות לפתרונה ולדון ביעילות של כל אחת מהדרכים שהוצעו לעומת הדרך המוצעת כאן.

רעיון הפתרון היעיל ביותר:

לאחר חישוב הממוצע המבוקש, ניגש לאיבר האמצעי במערך (יש להתייחס גם לאורך מערך זוגי שבו יש שני איברים אמצעיים). אם איבר זה גדול מהממוצע, הרי שכל האיברים מימינו גם גדולים מהממוצע (מכיוון שהמערך ממוין), ולכן רוב האיברים גדולים מהממוצע. אם איבר זה אינו גדול מהממוצע, רוב איברי המערך אינם גדולים מהממוצע.

```
double average = (double)(a[0] + a[a.length-1])/2;

if(a[(a.length-1)/2] > average)

      System.out.println("Yes");

else

      System.out.println("No");
```

שאלה 12.19

"סדרת הפרשים עולה" היא סדרת ערכים שבה ההפרשים בין כל זוג ערכים סמוכים עולים-ממש. כלומר ההפרש הגדול ביותר הוא בין האיבר האחרון בסדרה וזה שלפניו, וההפרש הקטן ביותר הוא ההפרש בין האיבר הראשון לשני. דוגמה ל"סדרת הפרשים עולה": 97, 66, 40, 29, 20, 15. עליכם לכתוב אלגוריתם שיקלוט "סדרת הפרשים עולה" במערך ומספר נוסף k. האלגוריתם יבדוק אם קיים זוג ערכים סמוכים במערך שהפרשם הוא k. ממשו את האלגוריתם בשפת Java. עליכם לכתוב את האלגוריתם יעיל ככל שתוכלו מבלי להשתמש במערך עזר.

תשובה 12.19

רעיון הפתרון:

פתרון שאלה זו מבוסס על חיפוש בינרי בסדרת ההפרשים. יש לשים לב כי המערך אינו בהכרח ממוין, אך סדרת ההפרשים **כן** ממוינת. בכל שלב בחיפוש הבינרי נבדוק את ההפרש בין שני איברים סמוכים במקום את האיבר עצמו.

```
low = 0;
high = a.length - 1;
boolean found = false;
while (low < high && !found)
{
      middle = (low + high) / 2;
      if (a[middle+1] - a[middle] == k)
          found = true;
      else if (a[middle+1] - a[middle] > k)
          high = middle - 1;
      else
          low = middle + 1;
 }
 if(!found)
```

```
     System.out.println("No");
 else
     System.out.println("Yes");
```

שאלה 12.22

האם מספר הפעמים שהלולאה מתבצעת במיון בחירה תלוי בתוכן המערך?

תשובה 12.22

לא. בכל מקרה מתרחשת מציאת מינימום. שתי הלולאות (החיצונית והפנימית) עוברות **בכל מקרה** על כל אברי המערך.

שאלה 12.23

כמה פעולות החלפה מתבצעות במיון בחירה?

תשובה 12.23

עבור כל מינימום שאנחנו מוצאים מתבצעת החלפה, לכן מספר ההחלפות שמתבצעות הוא כמספר הפעמים שאנו מחפשים מינימום והוא: $n-1$ כאשר n הוא מספר אברי המערך.

שאלה 12.25

כתבו אלגוריתם המקבל כקלט סדרת מספרים בסדר כלשהו. האלגוריתם יציג את מספר הערכים שמיקומם המקורי הוא גם מיקומם בסדרה <u>המסודרת</u> של הערכים הנתונים.

הדרכה: קלטו את האיברים במערך, מיינו במיון בחירה ומנו תוך כדי כך כמה איברים נמצאו במיקומם.

תשובה 12.25

רעיון הפתרון:

לאחר מציאת המינימום במיון בחירה, נבדוק האם הוא נמצא במקום הסידורי המבוקש. נמנה את האיברים שהיו במקומם ולא היה צורך להחליפם עם איבר אחר. שימו ♥: אלגוריתם זה מוודא שהחלפות מיותרות לא תתבצענה.

יישום האלגוריתם:

```
int iMin, temp;
int counter = 0;
for (int i = 0; i < a.length - 1; i++)
{
     iMin = i;
     for (int j = i + 1; j < a.length; j++)
          if (a[j] < a[iMin])
              iMin = j;
     // אם האיבר במקומו, מונים אותו. אחרת מבצעים את ההחלפה
     if (iMin == i)
          counter++;
     else
     {
          temp = a[iMin];
          a[iMin] = a[i];
          a[i] = temp;
     }
 }
```

```
System.out.print(counter);
```

שאלה 12.30

תארו מה קורה במיון הכנסה כאשר מנסים למיין מערך ממוין.

תשובה 12.30

כאשר מנסים למיין מערך ממוין במיון הכנסה, כל איבר נמצא כבר במקומו ולכן פעולת הזזת האיברים לא תתבצע כלל.

שאלה 12.31

אפיינו את המערך הגורם לביצועים הגרועים ביותר במיון הכנסה.

תשובה 12.31

מערך ממוין בסדר **הפוך** מהמבוקש יגרום לביצועים הגרועים ביותר במיון הכנסה, כיוון שהפעולה insertElement בכל זימון תבצע את מקסימום ההזזות האפשריות.

שאלה 12.35

לפניכם קטע תוכנית הממיין בשיטת "מיון בועות" את המערך A ובו מספרים שלמים. השלימו את החלקים החסרים במיון: (שימו ♥: המערך צריך להיות ממוין בסדר עולה).

```
int temp;
for (int i = 1 ; i < ____________; i++)
{
      for (int j = 0 ; j < _____________; j++)
      {
            if (A[____] > A[____])
            {
                  temp = ___________;
                  A[____] = ___________;
                  A[____] = _________;
            }
      }
}
```

תשובה 12.35

```
int temp;
for (int i = 1 ; i < A.length-1 ; i++)
{
      for (int j = 0 ; j < A.length-i ; j++)
      {
            if (A[j] > A[j+1])
            {
                  temp = A[j];
                  A[j] = A[j+1];
                  A[j+1] = temp;
            }
      }
}
```

שאלה 12.39

לפניכם קטע תוכנית בשפת Java, למיזוג המערכים A ו-B במערך C, השלימו את הקטע במקומות החסרים:

```
int iA = _______;
int iB = _______;
int iC = _______;
while (iA < A.length && iB < B.length)
{
    if(A[iA] ____ B[iB])
    {
        C[_____] = A[_____];
        _____++;
    }
    else
    {
        C[_____] = B[_____];
        _____++;
    }
    _____++;
}
while (_________________)
{
    C[iC] = A[iA];
    iA++;
    iC++;
}
while (_________________)
{
    C[iC] = B[iB];
    iB++;
    iC++;
}
```

תשובה 12.39

```
int iA = 0 ;
int iB = 0 ;
int iC = 0 ;
while (iA < A.length && iB < B.length)
{
    if(A[iA] < B[iB])
    {
        C[iC] = A[iA];
        iA++;
    }
    else
    {
        C[iC] = B[iB];
        iB++;
    }
    iC++;
}
```

```
while (iA < A.length)
{
    C[iC] = A[iA];
    iA++;
    iC++;
}
while (iB < B.length)
{
    C[iC] = B[iB];
    iB++;
    iC++;
}
```

שאלה 12.40

תנו דוגמאות של ערכים התחלתיים במערכים A ו-B שעבורן:

א. ערכו של iA נשאר 0 לאורך ביצוע לולאת ה-**while** הראשונה.

ב. ערכו של iB נשאר 0 לאורך ביצוע לולאת ה-**while** הראשונה.

ג. ערכי iA ו-iB עולים לסירוגין: ערכו של iA עולה ב-1, אחר כך ערכו של iB עולה ב-1, אחר כך ערכו של iA עולה ב-1, וכן הלאה עד ל"זנב".

תשובה 12.40

א. ערכו של iA נשאר 0 לאורך ביצוע לולאת ה-**while** הראשונה, כיוון שהאיבר הקטן ביותר במערך A גדול יותר מהאיבר הגדול ביותר במערך B

A[0]	A[1]	A[2]	A[3]
12	27	70	93

B[0]	B[1]	B[2]	B[3]	B[4]
1	3	7	9	11

ב. ערכו של iB נשאר 0 לאורך ביצוע לולאת ה-while הראשונה, כיוון שהאיבר הקטן ביותר במערך B גדול יותר מהאיבר הגדול ביותר במערך A.

A[0]	A[1]	A[2]	A[3]
1	5	7	8

B[0]	B[1]	B[2]	B[3]	B[4]
10	35	73	92	100

ג. ערכי iA ו-iB עולים לסירוגין: ערכו של iA עולה ב-1, אחר כך ערכו של iB עולה ב-1, אחר כך ערכו של iA עולה ב-1, וכן הלאה עד ל"זנב". אם נגדיר את מיקומו של איבר במערך כ-i. כל ערך A[i] קטן יותר מהערך הנמצא ב-[i] B והערך B[i] גדול יותר מ-A[i+1].

A[0]	A[1]	A[2]	A[3]
1	5	9	13

B[0]	B[1]	B[2]	B[3]	B[4]
3	7	11	17	100

שאלה 12.42

נתונים שני מערכים ממוינים sortedA ו-soredtB. כתבו קטע תוכנית בשפת Java שיציג את ערכו של האיבר הקטן ביותר המופיע בשני המערכים.

למשל עבור המערכים:

sortedA = 1 ,5 ,6 ,8 ,10

sortedB = 2 ,4 ,6 ,10 ,20

יודפס הערך: 6

תשובה 12.42

רעיון הפתרון:

כיוון שהמערכים ממוינים הרי שאם ערך תורן במהלך אלגוריתם המיזוג קטן מהערך התורן במערך השני לא יתכן שערך זה הוא הערך המבוקש, כיוון ששאר הערכים במערך השני בוודאות גדולים ממנו. לכן, נסרוק את המערכים באותו אופן בו אנו סורקים אותם במיזוג, עד אשר נמצא שני ערכים תורנים זהים או עד אשר נסיים את הסריקה באחד המערכים.

```
boolean found = false;
int iA = 0 ;
int iB = 0 ;
while (iA < sortedA.length && iB < sortedB.length && !found)
{
     if(sortedA[iA] == sortedB[iB])
          found=true;
     else if(sortedA[iA] < sortedB[iB])
           iA++;
     else
           iB++;
}
if(found)
    System.out.println(sortedA[iA]);
else
    System.out.println("No common number! ");
```

שאלה 12.44

נתונים שני מערכים ממוינים, אחד בסדר עולה והשני בסדר יורד. כתבו קטע תוכנית הממזג את שני המערכים כך שהמערך הממוזג יהיה ממוין בסדר עולה.

תשובה 12.44

רעיון הפתרון:

נמזג את המערכים באופן דומה למיזוג רגיל, בהבדל אחד: במערך שממוין בסדר הפוך נתחיל מהסוף להתחלה, הנה כך:

```
int iA = 0 ;
int iB = B.length-1 ;                // האתחול מתחיל מהסוף
int iC = 0 ;
while (iA < A.length && iB >= 0)     // שימו לב לתנאי עצירת הלולאה
{
     if(A[iA] < B[iB])
     {
          C[iC] = A[iA];
          iA++;
```

```
      }
      else
      {
            C[iC] = B[iB];
            iB--;
      }
      iC++;
}
// העתקת הזנבות
while (iA < A.length)
{
        C[iC] = A[iA];
        iA++;
        iC++;
}
while (iB >=0)
{
       C[iC] = B[iB];
       iB--;
       iC++;
}
```

שאלות נוספות

שאלות נוספות לסעיף 12.2

שאלה 1

נתון מערך A בגודל N, אשר ערכיו הראשונים הם 0 והערכים שאחריהם הם 1. כתבו קטע תוכנית המוצא את מציין האיבר האחרון שערכו הוא 0 (ניתן להניח שיש לפחות 0 אחד).

תשובה 1

רעיון הפתרון:

נשתמש בגרסה של חיפוש בינרי (בדומה לשאלה 12.19) בכל שלב נבדוק שני ערכים סמוכים: אם שניהם שווים 1, הערך המבוקש נמצא משמאל, אם שניהם שווים 0, הערך המבוקש נמצא מימין, ואם ערכיהם שונים: מצאנו את האיבר המבוקש.

שאלה 3

נתון מערך ממוין A בגודל N ובו ערכים חוזרים. כתבו קטע תוכנית שהפלט שלו הוא הודעה אם יש ערך המופיע יותר מ-N/2 פעמים.

תשובה 3

רעיון הפתרון:

כיוון שהמערך ממוין, אם ערך כלשהו מופיע יותר מ-N/2 פעמים, הרי שערך האיבר האמצעי במערך וודאי שווה לו, נתבונן בשני המערכים הנתונים בגודל 10: הערך X נמצא בשניהם יותר מ-5 פעמים, ולכן וודאי ערכו של האיבר האמצעי הוא בהכרח X.

		X	X	X	X	X	X		

				X	X	X	X	X	X

שאלות נוספות לסעיף 12.4

שאלה 1

הניחו שאין ערכים זהים בתוך מערך A ושאין ערכים זהים בתוך מערך B, אך ייתכן שיש ערכים הקיימים גם ב-A וגם ב-B. שנו את פעולת **המיזוג** כך שערכים זהים יופיעו פעם אחת בלבד במערך הממוזג.

תשובה 1

רעיון הפתרון:

בשונה מהמיזוג המוכר, כאשר נמצא זוג איברים שווים נדלג באחד המערכים על איבר זה.

```
int iA = 0;
int iB = 0;
int iC = 0;
while (iA < A.length && iB < B.length)
{
     if (A[iA] == B[iB])
         iA++;
     else
     {
         if (A[iA] < B[iB])
         {
             C[iC] = A[iA];
             iA++;
         }
         else
         {
             C[iC] = B[iB];
             iB++;
         }
         iC++;
     }
 }
```

שאלה 4

נתונים שני מערכים ממוינים ובהם יש ערכים חוזרים. כתבו קטע תוכנית המוצא ומדפיס את הערך המופיע מספר מרבי של פעמים בשתי הרשימות יחדיו (אם יש יותר מאחד כזה, אז נדפיס את אחד מן הערכים האלה).

תשובה 4

רעיון הפתרון:

נבצע מיזוג של שני המערכים, ולאחר המיזוג יהיה ברשותנו מערך C ממוין, כעת נעבור על מערך זה, ונבדוק מהו הערך השכיח (שימו לב כי המערך ממוין, ולכן ערכים זהים נמצאים בסמיכות).