

סוג הבחינה: מבחן מתכונת
מבחן: מתכונת 8
מספר השאלון: 899371
שם התלמיד/ה: _____

מדעי המחשב

יסודות מדעי המחשב – 1-2 יחידות לימוד

הוראות

- א. משך הבחינה: שעה וחצי.
- ב. מבנה השאלון ומפתח ההערכה: בשאלון זה שני פרקים.
פרק ראשון – (2×20) – 40 נקודות
פרק שני – (2×30) – 60 נקודות
סך הכול – 100 נקודות
- ג. חומר עזר מותר בשימוש: כל חומר עזר, חוץ ממחשבון שיש בו אפשרות תכנות.
- ד. הוראה מיוחדת:
את כל התוכניות שיש לכתוב בשפת מחשב כתבו בשפה אחת בלבד – Java או C#.
הערה: לא יורדו נקודות אם תכתבו בתוכניות אות גדולה במקום אות קטנה או להפך.
יש לכתוב במחברת הבחינה בלבד. יש לרשום "טיוטה" בראש כל עמוד המשמש טיוטה.
כתיבת טיוטה בדפים שאינם במחברת הבחינה עלולה לגרום לפסילת הבחינה.
השאלות בשאלון זה מנוסחות בלשון רבים, אף על פי כן על כל תלמידה וכל תלמיד להשיב עליהן באופן אישי.
בהצלחה!
השאלות
בשאלון זה שני פרקים.
יש לענות על שאלות משני הפרקים, לפי ההוראות בכל פרק.
הערה: בכל שאלה שנדרשת בה קליטה, אין צורך לבדוק את תקינות הקלט.
לפותרים בשפת Java: בכל שאלה שנדרשת בה קליטה, הניחו שבתוכנית כתובה ההוראה:
`Scanner scan = new Scanner(System.in);`

השאלות

פרק ראשון (40 נקודות)

ענו על שתיים מן השאלות 1-3 (לכל שאלה - 20 נקודות).

1. (20 נקודות)

נתונים שני מערכים מטיפוס שלם: a בגודל 10, ו- b בגודל 5.
המערך a מכיל מספרים גדולים מ-0, והמערך b מכיל כולו אפסים.

9	8	7	6	5	4	3	2	1	0	
2	11	6	3	8	3	6	15	4	9	a

4	3	2	1	0	
0	0	0	0	0	b

נתון קטע התכנית שלפניך, הכתוב ב-Java וב-C#.

Java

Java

```
int k = 0;
for(int i=0; i<a.length/2; i++)
{
    int j = a.length-1-i;
    if(a[i] > a[j])
    {
        b[k] = a[i] - a[j];
        k++;
    }
}
```

#C

C#

```
int k = 0;
for(int i=0; i<a.Length/2; i++)
{
    int j = a.Length-1-i;
    if(a[i] > a[j])
    {
        b[k] = a[i] - a[j];
        k++;
    }
}
```

1. עקבו בעזרת טבלת מעקב אחר ביצוע קטע התכנית בעבור המערכים a ו- b הנתונים, ורישמו את המערך b לאחר ביצוע קטע התכנית.

בטבלת המעקב יש לכלול עמודה לכל אחד מהמשתנים האלה: i , j , $a[i]$, $a[j]$, k , $b[k]$.

2. תנו דוגמה מייצגת למערך a בגודל 10 המכיל מספרים גדולים מ-0, שבעבורו, לאחר ביצוע קטע התכנית, אף אחד מתאי המערך b לא יכיל את הערך 0.

2. (20 נקודות)

ממשו את הפעולה keepFirst בשפת Java או KeepFirst בשפת C# :

Java - public static void keepFirst (int[] arr)

C# - public static void KeepFirst (int[] arr)

בכניסה למוזיאון סורקים את כרטיסי המבקרים. במערך arr שמתקבל רשומים מספרי הכרטיסים שנשרקו, לפי סדר הסריקה. כל מספרי הכרטיסים הם מספרים חיוביים. כרטיס עשוי להיסרק יותר מפעם אחת, ולכן מספר עשוי להופיע במערך כמה פעמים.

הפעולה מעבירה לתחילת המערך arr כל מספר כרטיס **פעם אחת בלבד**, לפי הסדר שבו הוא הופיע **לראשונה** במערך, ומציבה את הערך 1- בכל התאים שנותרו אחריהם.

דוגמה:

בעבור המערך arr שלפניכם:

	7	6	5	4	3	2	1	0	
arr	5	30	7	7	5	12	7	12	

המערך arr ייראה כך בסיום הפעולה:

	7	6	5	4	3	2	1	0	
arr	1-	1-	1-	1-	30	5	7	12	

הסבר: במערך יש ארבעה מספרי כרטיסים שונים: 12, 7, 5, 30 (לפי הסדר שבו הופיעו לראשונה). הם הועברו לתחילת המערך, ובארבעת התאים הנותרים הוצב 1-.

ממשו את הפעולה keepFirst בשפת Java או KeepFirst בשפת C#, לפי התיאור שלמעלה.

(20 נקודות)

3. (20 נקודות)

מספר חיובי - num ייקרא "מספר מרגל" אם מתקיים התנאי שלפניכם:

סכום כל הספרות של num שווה למכפלת כל הספרות של num .

דוגמה ל"מספר מרגל": המספר 1124, כי סכום הספרות שלו הוא $1 + 1 + 2 + 4 = 8$, ומכפלת הספרות שלו היא $1 \cdot 1 \cdot 2 \cdot 4 = 8$.

דוגמה נוספת ל"מספר מרגל": המספר 123, כי $1 + 2 + 3 = 6$ וגם $1 \cdot 2 \cdot 3 = 6$.

דוגמה למספר שאינו "מספר מרגל": המספר 1234, כי סכום הספרות שלו הוא $1 + 2 + 3 + 4 = 10$, ואילו מכפלת הספרות שלו היא $1 \cdot 2 \cdot 3 \cdot 4 = 24$.

א. כתבו פעולה חיצונית ששמה isSpy בשפת Java או IsSpy בשפת C#, המקבלת מספר מטיפוס שלם - num הגדול מ-0. הפעולה מחזירה true אם המספר הוא "מספר מרגל", אחרת היא מחזירה false.

ב. כתבו פעולה חיצונית ששמה printSpies בשפת Java או PrintSpies בשפת C#, המקבלת מספר מטיפוס שלם - n הגדול מ-0. הפעולה מדפיסה את כל "מספרי המרגל" שיש מ-1 ועד n (כולל).
הערה: יש להשתמש בפעולה שכתבתם בסעיף א.

ענו על **שתיים** מן השאלות 4-6 (לכל שאלה - 30 נקודות).

4. נתונה המחלקה **RiverGauge**, המייצגת תחנה המודדת את גובה פני המים בנחל. למחלקה שתי תכונות:

- station - שם תחנת המדידה, מטיפוס מחרוזת
- levels - מערך מטיפוס שלם, המכיל את גובה פני המים (בסנטימטרים) שנמדד בתחנה פעם ביום, לפי סדר הימים

הניחו שקיימות פעולות `get/Set` ו- `set/Set` עבור תכונות המחלקה.

"שיא" הוא יום שבו גובה המים היה **גבוה** גם מגובה המים ביום הקודם וגם מגובה המים ביום שאחריו. "שפל" הוא יום שבו גובה המים היה **נמוך** גם מגובה המים ביום הקודם וגם מגובה המים ביום שאחריו. היום הראשון והיום האחרון בתקופת המדידה אינם נחשבים "שיא" או "שפל".

א. כתבו פעולה פנימית ששמה `countTurns` בשפת Java או `CountTurns` בשפת #C, המקבלת פרמטר - `peak` מטיפוס בוליאני, ומחזירה מספר שלם בהתאם לאפיון שלפניכם:

אם הפרמטר `peak` שהתקבל הוא `true`, הפעולה תחזיר את מספר הימים שנחשבים "שיא".

אם הפרמטר `peak` שהתקבל הוא `false`, הפעולה תחזיר את מספר הימים שנחשבים "שפל".

לדוגמה, עבור המערך `levels` שלפניכם, אם `peak` הוא `true` הפעולה תחזיר 3, ואם `peak` הוא `false` הפעולה תחזיר 1.

8	7	6	5	4	3	2	1	0	
120	133	131	150	140	128	128	135	120	levels

הסבר: התאים באינדקסים 7, 5, 1 מייצגים ימים שנחשבים "שיא". התא באינדקס 6 מייצג יום שנחשב "שפל".

התא באינדקס 2 אינו "שפל", כי גובה המים בו (128) אינו נמוך מגובה המים ביום שאחריו (128).

ב. נחל ייקרא "סוער" אם מספר הימים שנחשבים "שיא" או "שפל" (יחד) גדול ממחצית מספר ימי המדידה.

כתבו פעולה פנימית ששמה `isRestless` בשפת Java או `IsRestless` בשפת #C, המחזירה `true` אם הנחל "סוער". אחרת הפעולה תחזיר `false`.

למשל, בדוגמה שלעיל הפעולה תחזיר `false`, כיוון שיש 4 ימים שנחשבים "שיא" או "שפל", ו- 4 אינו גדול ממחצית מספר ימי המדידה (9 ימים).

חובה להיעזר בפעולה שכתבתם בסעיף א.

5. (30 נקודות)

נתונה המחלקה **Seat** - מושב באולם מופעים, ולה שתי תכונות:

- row - מספר השורה (מספר בין 1 ל-30), מטיפוס שלם
 - number - מספר המושב בשורה (מספר בין 1 ל-40), מטיפוס שלם
- הניחו שקיימות פעולות `get/Set` ו- `set/Set` לתכונות המחלקה.

- א.** שני מושבים הם **צמודים** אם הם באותה שורה ומספרי המושבים שלהם עוקבים (ההפרש ביניהם הוא 1 בדיוק).
 כתבו במחלקה **Seat** פעולה פנימית ששמה `isNextTo` בשפת `Java` או `IsNextTo` בשפת `C#` המקבלת עצם אחר - `other` מטיפוס **Seat**.
 הפעולה מחזירה `true` אם שני המושבים צמודים, אחרת היא מחזירה `false`.
למשל, המושב בשורה 7 מספר 12 והמושב בשורה 7 מספר 13 צמודים; המושב בשורה 7 מספר 12 והמושב בשורה 8 מספר 13 אינם צמודים.
- ב.** (1) אולם המופעים העירוני מוכר כרטיסים במערכת ממוחשבת. לשם כך נבנתה המחלקה **Ticket** - כרטיס, ולה שלוש תכונות:
 • `buyerCode` - הקוד של הקונה (מספר בן 4 ספרות), מטיפוס `int`.
 • `section` - האות של האגף באולם (אות גדולה באנגלית בין 'A' ל-'F'), מטיפוס `char`.
 • `seat` - המושב, מטיפוס **Seat**.
 בכל אגף יש מספור שורות ומושבים משלו. הניחו שקיימות פעולות `get/Get` ו-`set/Set` לתכונות המחלקה.
 כתבו במחלקה **Ticket** פעולה פנימית ששמה `sameSection` בשפת `Java` או `SameSection` בשפת `C#`, המקבלת עצם אחר - `other` מטיפוס **Ticket**. הפעולה מחזירה `true` אם שני הכרטיסים הם לאותו אגף. אחרת הפעולה מחזירה `false`.
 (2) שני כרטיסים "יושבים יחד" אם הם לאותו אגף וגם המושבים שלהם צמודים (אם רק תנאי אחד מתקיים, הכרטיסים **אינם** "יושבים יחד").
 כתבו במחלקה **Ticket** פעולה פנימית ששמה `sitTogether` בשפת `Java` או `SitTogether` בשפת `C#`, המקבלת עצם אחר - `other` מטיפוס **Ticket**. הפעולה מחזירה `true` אם שני הכרטיסים "יושבים יחד", אחרת הפעולה מחזירה `false`.
 יש להשתמש בפעולות שכתבתם בסעיף א ובתת-סעיף ב(1).
- ג.** כרטיס נקרא "כרטיס אמצעי" אם יש במערך **שני** כרטיסים אחרים ש"יושבים יחד" איתו (כפי שהוגדר בתת-סעיף ב(2)), כלומר יש לו שכנים משני צדדיו.
 כתבו פעולה חיצונית ששמה `printMiddle` בשפת `Java` או `PrintMiddle` בשפת `C#` המקבלת מערך - `tickets` מטיפוס **Ticket** של כל הכרטיסים שנמכרו למופע מסוים. המערך אינו מסודר לפי סדר כלשהו, אין בו ערכי `null`, ואין בו שני כרטיסים לאותו מושב באותו אגף. הפעולה **תדפיס** את הקוד של הקונה של כל "כרטיס אמצעי" במערך (כל הדפסה בשורה חדשה).
 יש להשתמש בפעולה שכתבתם בתת-סעיף ב(2).

6. (30 נקודות)

נתונה פעולה חיצונית ששמה `digitProduct` בשפת Java או `DigitProduct` בשפת C#, המקבלת מספר שלם שאינו שלילי - `num` ומחזירה את מכפלת כל הספרות במספר. אפשר להשתמש בפעולה **בלי לממש אותה**.

לדוגמה: עבור $num = 4448$, הפעולה תחזיר $512 (8 \times 4 \times 4 \times 4)$.

"שורש המכפלה" של מספר כלשהו הוא מספר **חד-ספרתי** המתקבל באופן שלהלן: מחשבים את מכפלת הספרות שוב ושוב עד שמתקבל מספר חד-ספרתי.

דוגמאות:

"שורש המכפלה" של המספר 7 הוא 7.

"שורש המכפלה" של המספר 25 הוא 5 ($10 = 5 \times 2$), ואז $0 = 0 \times 1$.

"שורש המכפלה" של המספר 39 הוא 4, כמפורט להלן:

$$27 = 9 \times 3$$

$$14 = 7 \times 2$$

$$4 = 4 \times 1$$

א. (1) כתבו פעולה חיצונית ששמה `productRoot` בשפת Java או `ProductRoot` בשפת C# המקבלת

מספר שלם `num` שאינו שלילי ומחזירה את "שורש המכפלה" שלו.

(2) "שורש המכפלה" של כל מספר שיש בו את הספרה 0 הוא 0, אבל גם מספרים רבים שאין בהם

את הספרה 0 מגיעים ל"שורש מכפלה" 0 (למשל 25 שבדוגמה שלעיל). יש הטוענים כי בטווח

שבין 1 ל-99999, ליותר מ-80 אחוזים מן המספרים יש "שורש מכפלה" 0.

כתבו פעולה חיצונית ששמה `isCorrect` בשפת Java או `IsCorrect` בשפת C# המחזירה `true` אם

הטענה נכונה, ואחרת מחזירה `false`.

הערה: חובה להיעזר בפעולה שכתבתם בסעיף א(1).

ב. נתונה פעולה חיצונית נוספת, ששמה `countDigit` בשפת Java או `CountDigit` בשפת C#, המקבלת

מספר שלם חיובי - `num` וספרה - `digit`. הפעולה מחזירה את מספר הפעמים שהספרה `digit`

מופיעה במספר `num` (ו-0 אם אינה מופיעה בו).

אפשר להשתמש בפעולה **בלי לממש אותה**.

כתבו פעולה חיצונית ששמה `isStronger` בשפת Java או `IsStronger` בשפת C# המקבלת שני מספרים

שלמים חיוביים: `num1` ו-`num2`. הפעולה מחזירה `true` אם "שורש המכפלה" של `num1` מופיע

במספר `num2` **יותר פעמים** מאשר "שורש המכפלה" של `num2` מופיע במספר `num1`, אחרת היא

מחזירה `false`.

למשל עבור $num1 = 39$ ו- $num2 = 4448$, הפעולה תחזיר `true` כי "שורש המכפלה" של (4) `num1`

מופיע במספר (4448) `num2` שלוש פעמים, ו"שורש המכפלה" של (0) `num2` אינו מופיע במספר

(39) `num1` כלל.

הערה: חובה להיעזר בפעולה שכתבתם בסעיף א(1).

כל הבגרויות במדעי המחשב עם פתרונות מלאים: www.guyai.co.il/bagrut/computer-science