

סוג הבחינה: מבחן מתכונת
מבחן: מתכונת 6
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)

לפניכם הפעולה what בשפת Java או What בשפת #C :

Java

Java

```
public static int what(int[] arr)
{
    int c = 1;
    int m = 1;
    for (int i = 1; i < arr.length; i++)
    {
        if (arr[i] > arr[i - 1])
            c++;
        else
            c = 1;
        m = Math.max(m, c);
    }
    return m;
}
```

#C

C#

```
public static int What(int[] arr)
{
    int c = 1;
    int m = 1;
    for (int i = 1; i < arr.Length; i++)
    {
        if (arr[i] > arr[i - 1])
            c++;
        else
            c = 1;
        m = Math.Max(m, c);
    }
    return m;
}
```

נתון המערך arr :

8	7	6	5	4	3	2	1	0	
4	12	9	7	5	5	3	8	6	arr

עקבו בעזרת טבלת מעקב אחר הפעולה what(arr) בשפת Java או What(arr) בשפת #C, וכתבו מה הפעולה מחזירה.

בטבלת המעקב יש לכלול עמודה בעבור כל אחד מן המשתנים האלה: m, c, i.

(20 נקודות)

2. (20 נקודות)

במפעל משקאות ממלאים בקבוקים בקו מילוי אוטומטי.

הפרמטר target הוא מספר שלם חיובי המייצג את כמות המשקה הרצויה במ"ל בכל בקבוק.

הפרמטר tol הוא מספר שלם חיובי המייצג את **הסטייה המותרת** במ"ל מן הכמות הרצויה.

- א.** כתבו פעולה חיצונית ששמה `getCheck` בשפת Java או `GetCheck` בשפת #C המקבלת מספר שלם חיובי - `volume`, המייצג את כמות המשקה במ"ל בבקבוק מסוים, ואת הפרמטרים `target` ו-`tol`. הפעולה תחזיר מספר שלם על פי התנאים האלה:
- אם כמות המשקה בבקבוק **קטנה** מן הכמות הרצויה ביותר מ-`tol` מ"ל, הפעולה תחזיר 1 (יש להשלים את המילוי).
 - אם כמות המשקה בבקבוק **גדולה** מן הכמות הרצויה ביותר מ-`tol` מ"ל, הפעולה תחזיר 2 (הבקבוק נפסל).
 - אם שני התנאים שלעיל אינם מתקיימים, הפעולה תחזיר 0 (הבקבוק תקין).

- ב.** המערך `volumes` הוא מערך מטיפוס שלם המכיל את כמויות המשקה במ"ל בבקבוקים שמולאו בקו (כל תא במערך מציג את הכמות בבקבוק אחר). המערך אינו מסודר בסדר מסוים.
- כתבו פעולה חיצונית ששמה `getAllChecks` בשפת Java או `GetAllChecks` בשפת #C המקבלת את המערך `volumes` ואת הפרמטרים `target` ו-`tol`.
- הפעולה תחזיר מערך **חדש** מטיפוס שלם בגודל זהה למערך `volumes`, המציג בכל תא את המספר 0 או 1 או 2 בהתאם לבקבוק שנמצא באותו האינדקס במערך `volumes` (על פי התנאים בסעיף א).
- דוגמה:** בעבור `target = 500`, `tol = 5`, והמערך `volumes`:

אינדקס	0	1	2	3	4	5	6
<code>volumes</code>	503	490	506	495	500	512	494

יוחזר המערך שלהלן:

אינדקס	0	1	2	3	4	5	6
	0	1	2	0	0	2	1

הערות: - יש להשתמש בפעולה שכתבתם בסעיף א.

- אין לשנות את המערך `volumes`.

- ג.** מנהל הקו מקבל התראה אם **פחות** משלושה רבעים מן הבקבוקים תקינים.
- כתבו פעולה חיצונית ששמה `lineAlert` בשפת Java או `LineAlert` בשפת #C, המקבלת את המערך `volumes` ואת הפרמטרים `target` ו-`tol`.
- הפעולה תחזיר `true` אם פחות משלושה רבעים מן הבקבוקים במערך `volumes` תקינים, אחרת הפעולה תחזיר `false`.
- דוגמה:** בעבור המערך הנתון בסעיף ב הפעולה תחזיר `true`, כי מבין שבעת הבקבוקים במערך רק שלושה תקינים (הבקבוקים באינדקסים 0, 3, 4), ושלושה רבעים מ-7 הם 5.25.
- הערה:** יש להשתמש בפעולה שכתבתם בסעיף ב.

3. (20 נקודות)

בספריית בית הספר "דפים" כל תלמיד בוחר סיסמה לחשבון ההשאלה שלו. סיסמה מורכבת מאותיות באנגלית (קטנות או גדולות), מספרות ומתווים אחרים (כמו ! \$ # _), ואין בה רווחים. סיסמה **חזקה** היא סיסמה שאורכה 8 תווים לפחות, יש בה לפחות 2 ספרות, ויש בה לפחות תו אחר אחד (תו שאינו אות ואינו ספרה).

הערה: אין צורך לבדוק את תקינות הקלט.

לפותרים בשפת **Java**: הניחו שבתוכנית כתובה ההוראה:

`Scanner input = new Scanner(System.in);`

1. כתבו פעולה בשם `analyze` (ב- `Analyze - C#`) שתקבל מחרוזת `pw` המייצגת סיסמה.

הפעולה תחזיר מערך חד-ממדי של מספרים שלמים בגודל 3: בתא 0 - מספר האותיות בסיסמה,

בתא 1 - מספר הספרות בסיסמה, ובתא 2 - מספר התווים האחרים בסיסמה.

לדוגמה, בעבור הסיסמה "Gal#2026" הפעולה תחזיר את המערך: 3,4,1

2. כתבו פעולה בשם `checkPasswords` (ב- `CheckPasswords - C#`) שתקלוט את הסיסמאות שבחרו התלמידים.

הקלט מסתיים כאשר נקלטת סיסמה שאין בה אף אות.

הפעולה תדפיס את מספר הסיסמאות החזקות ואת מספר הסיסמאות שאינן חזקות. הסיסמה

שבה הסתיים הקלט אינה נספרת.

דוגמה להדפסה: 4,9

ניתן להשתמש בפעולה שכתבתם בסעיף 1.

פרק שני (60 נקודות)

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

4. נתונה המחלקה **Greenhouse**, המייצגת חממה לגידול ירקות. בחממה מותקן חיישן המודד את

הטמפרטורה פעם בשעה.

למחלקה ארבע תכונות:

· `name` - שם הגידול בחממה, מטיפוס מחרוזת

· `temps` - מערך מטיפוס שלם, המכיל את הטמפרטורות (במעלות צלזיוס) שנמדדו בחממה, לפי סדר

המדידה, שעה אחר שעה

· `low` - מספר שלם, הטמפרטורה הנמוכה ביותר המתאימה לגידול

· `high` - מספר שלם, הטמפרטורה הגבוהה ביותר המתאימה לגידול (`low < high`)

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

"שעה חמה" היא שעה שבה הטמפרטורה שנמדדה **גבוהה** מ- `high`, ו"שעה קרה" היא שעה שבה

הטמפרטורה שנמדדה **נמוכה** מ- `low`.

"רצף" הוא סדרה של שעות הסמוכות זו לזו (תאים סמוכים במערך).

א. כתבו פעולה פנימית ששמה `longestRun` בשפת `Java` או `LongestRun` בשפת `C#`, המקבלת פרמטר -

`hot` מטיפוס בוליאני, ומחזירה מספר שלם בהתאם לאפיון שלפניכם:

· אם הפרמטר `hot` שהתקבל הוא `true`, הפעולה תחזיר את אורך הרצף הארוך ביותר של "שעות

חמות".

אם הפרמטר hot שהתקבל הוא false , הפעולה תחזיר את אורך הרצף הארוך ביותר של "שעות קרות".

אם אין אף שעה מהסוג המבוקש, הפעולה תחזיר 0 .

לדוגמה, עבור חממה שבה high = 30 , low = 18 והמערך temps שלפניכם, אם hot הוא true הפעולה תחזיר 3 , ואם hot הוא false הפעולה תחזיר 2 .

9	8	7	6	5	4	3	2	1	0	
31	29	15	25	32	33	31	22	16	17	temps

הסבר: התאים באינדקסים 9 , 5 , 4 , 3 מייצגים "שעות חמות". הרצף הארוך ביותר שלהן הוא באינדקסים 3-5 , ואורכו 3 .

התאים באינדקסים 7 , 1 , 0 מייצגים "שעות קרות". הרצף הארוך ביותר שלהן הוא באינדקסים 0-1 , ואורכו 2 .

ב. כתבו פעולה פנימית ששמה isSafe בשפת Java או IsSafe בשפת #C , המחזירה true אם בחממה לא היה רצף של יותר משתי "שעות חמות" וגם לא היה רצף של יותר משתי "שעות קרות". אחרת הפעולה תחזיר false .

למשל, בדוגמה שלעיל הפעולה תחזיר false , כיוון שהיה רצף של 3 "שעות חמות".
חובה להיעזר בפעולה שכתבתם בסעיף א.

5. המחלקה Size מייצגת מידות של תיבה, ולה שלוש תכונות:

length - מספר שלם חיובי, האורך בסנטימטרים

width - מספר שלם חיובי, הרוחב בסנטימטרים

height - מספר שלם חיובי, הגובה בסנטימטרים

הניחו שקיימות פעולות get/Set ו־ set/Set עבור תכונות המחלקה, ופעולה בונה המקבלת ערכים עבור תכונות המחלקה.

לפניכם ממשק המחלקה. אפשר להשתמש בפעולות המחלקה בלי לממש אותן.

תיאור הפעולה	כותרת הפעולה
הפעולה מחזירה את הנפח של העצם הנוכחי בסמ"ק (אורך × רוחב × גובה).	Java - public int volume () C# - public int Volume ()
הפעולה מחזירה true אם הנפח של העצם הנוכחי גדול מהנפח של העצם other , אחרת (כלומר אם הנפח שווה לו או קטן ממנו) הפעולה מחזירה false .	Java - public boolean biggerThan (Size other) C# - public bool BiggerThan (Size other)

המחלקה **Shipment** מייצגת חבילה במרכז הפצה של חנות מקוונת, ולה שלוש תכונות:

code - קוד החבילה, מטיפוס מחרוזת

box - מידות קרטון האריזה, מטיפוס Size

item - מידות המוצר שארוז בקרטון, מטיפוס Size

הניחו שקיימות פעולות get/Set ו־ set/Set עבור תכונות המחלקה, ושתכונות המחלקה אינן null .
הניחו שהמוצר נכנס תמיד לקרטון שלו.

נתון מערך חד־ממדי shipments מטיפוס **Shipment** המכיל מידע על כל החבילות שנארזו במרכז ההפצה ביום מסוים. **המערך אינו ממוין** ואינו מכיל ערכי null .

א. כתבו פעולה חיצונית ששמה largestBox בשפת Java או LargestBox בשפת C#, המקבלת את המערך shipments, ומדפיסה את קוד החבילה שהקרטון שלה הוא בעל הנפח הגדול ביותר. הניחו שאין שני קרטונים בעלי אותו נפח.

ב. (1) את החלל הריק בקרטון ממלאים בחומר ריפוד. עלות הריפוד נקבעת לפי נפח החלל הריק בקרטון:

. עד 2000 סמ"ק (כולל) - ללא עלות

. יותר מ- 2000 סמ"ק ועד 8000 סמ"ק (כולל) - 5 שקלים

. יותר מ- 8000 סמ"ק - 12 שקלים

כתבו פעולה פנימית במחלקה Shipment ששמה emptySpace בשפת Java או EmptySpace בשפת C#, המחזירה את נפח החלל הריק בקרטון (נפח הקרטון פחות נפח המוצר).

(2) כתבו פעולה חיצונית ששמה fillerCost בשפת Java או FillerCost בשפת C#, המקבלת את המערך shipments, ומחזירה את העלות הכוללת של חומר הריפוד לכל החבילות. אפשר להשתמש בפעולה שכתבתם בסעיף ב(1).

6. (30 נקודות)

נתונה פעולה חיצונית ששמה maxDigit בשפת Java או MaxDigit בשפת C#, המקבלת מספר שלם שאינו שלילי - num ומחזירה את הספרה הגדולה ביותר במספר. אפשר להשתמש בפעולה **בלי לממש אותה**. **לדוגמה:** עבור $num = 1708$, הפעולה תחזיר 8.

"מספר הצעדים לאפס" של מספר שלם שאינו שלילי מחושב באופן שלהלן: בכל צעד מחסירים מן המספר את הספרה הגדולה ביותר שבו, וחוזרים על כך עד שמתקבל 0. "מספר הצעדים לאפס" הוא מספר ההחסרות שבוצעו.

דוגמאות:

"מספר הצעדים לאפס" של המספר 0 הוא 0.

"מספר הצעדים לאפס" של המספר 7 הוא $1 (7 - 7 = 0)$.

"מספר הצעדים לאפס" של המספר 24 הוא 5, כמפורט להלן:

$$20 = 4 - 24$$

$$18 = 2 - 20$$

$$10 = 8 - 18$$

$$9 = 1 - 10$$

$$0 = 9 - 9$$

א. (1) כתבו פעולה חיצונית ששמה stepsToZero בשפת Java או StepsToZero בשפת C#, המקבלת מספר שלם num שאינו שלילי ומחזירה את "מספר הצעדים לאפס" שלו.

(2) "מספר הצעדים לאפס" יכול להיות זוגי (למשל 0 עבור המספר 0) או אי-זוגי (למשל 5 כמו בדוגמה השלישית שלעיל). יש הטוענים כי בטווח שבין 1 ל-9999 יש יותר מספרים שעבורם "מספר הצעדים לאפס" אי-זוגי ממספרים שעבורם "מספר הצעדים לאפס" זוגי.

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 isPrime בשפת Java או IsPrime בשפת #C, המקבלת מספר שלם שאינו שלילי - num, ומחזירה true אם num הוא מספר ראשוני, ואחרת מחזירה false (המספרים 0 ו-1 אינם ראשוניים).

אפשר להשתמש בפעולה **בלי לממש אותה**.

כתבו פעולה חיצונית ששמה onlyOnePrime בשפת Java או OnlyOnePrime בשפת #C המקבלת שני מספרים שלמים שאינם שליליים: num1 ו- num2. הפעולה מחזירה true אם **בדיוק אחד** מבין "מספר הצעדים לאפס" של num1 ו"מספר הצעדים לאפס" של num2 הוא מספר ראשוני, אחרת היא מחזירה false.

למשל עבור num1 = 24 ו- num2 = 7, הפעולה תחזיר true כי "מספר הצעדים לאפס" של num1 הוא 5 (ראשוני) ו"מספר הצעדים לאפס" של num2 הוא 1 (אינו ראשוני). עבור num1 = 24 ו- num2 = 10 הפעולה תחזיר false, כי שני המספרים 5 ו-2 ראשוניים.
הערה: חובה להיעזר בפעולה שכתבתם בסעיף א(1).

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