

סוג הבחינה: מבחן מתכונת
מבחן: מתכונת 1
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 קטע תוכנית, הכתוב בשפות Java ו-#C.

Java

Java

```
int a = 84;
int b = 30;
while (a != b) {
    int big = Math.max(a, b);
    int small = Math.min(a, b);
    a = big - small;
    b = small;
}
System.out.println(a);
```

#C

C#

```
int a = 84;
int b = 30;
while (a != b) {
    int big = Math.Max(a, b);
    int small = Math.Min(a, b);
    a = big - small;
    b = small;
}
Console.WriteLine(a);
```

- א. עקבו בעזרת טבלת מעקב אחרי ביצוע קטע התוכנית, וכתבו את המספר שקטע התוכנית מדפיס. בטבלת המעקב יש לכלול עמודה בעבור כל אחד מן המשתנים האלה: `a`, `b`, `big`, `small`.
- ב. כתבו דוגמה לשני מספרים שונים זה מזה, הגדולים מ-1, שמוצבים ב-`a` וב-`b` (בתחילת הקטע) שבעבורם קטע התוכנית מדפיס 1 (אין צורך להציג מעקב).

2. (20 נקודות)

ביישומון להשמעת מוזיקה נשמרת רשימת "הושמעו לאחרונה" - מערך מטיפוס שלם בגודל 8, ובו הקודים של שמונת השירים **השונים** שהשתמש השמיע לאחרונה. השירים שמורים לפי הסדר מן השיר שהושמע אחרון אל השיר שהושמע לפני הזמן הרב ביותר: השיר שהושמע אחרון נמצא בתא 0, והשיר שהושמע לפני הזמן הרב ביותר נמצא בתא האחרון במערך.

כאשר השתמש משמיע שיר:

- אם קוד השיר **אינו** מופיע במערך - השיר שבתא האחרון נמחק מן המערך, שאר השירים זזים ימינה כל אחד לתא שלידו, והשיר שהושמע מופיע בתא 0.
 - אם קוד השיר **כבר מופיע** במערך - השיר מועבר מן התא שבו הוא נמצא אל תא 0, וכל השירים שהיו בתאים שלפניו זזים ימינה כל אחד לתא שלידו. השירים שבתאים שאחריו נשארים במקומם.
- כתבו פעולה בוליאנית חיצונית ששמה `playSong` בשפת `Java` או `PlaySong` בשפת `C#`, המקבלת מערך מלא - `recent`, וקוד של שיר - `song`, מטיפוס שלם.
- הפעולה מעדכנת את המערך `recent` לפי הכללים שלעיל, ומחזירה `true` אם השיר `song` **לא** הופיע במערך לפני ההשמעה. אם השיר כבר הופיע במערך, הפעולה מחזירה `false`.
- נתון המערך `recent` שלפניכם:

7	6	5	4	3	2	1	0	
931	48	203	765	19	340	87	512	recent

דוגמה: עבור המערך `recent` ו-`song = 19`, הפעולה תחזיר `false`, והמערך ייראה כך בתום הפעולה:

7	6	5	4	3	2	1	0	
931	48	203	765	340	87	512	19	recent

הסבר: השיר 19 כבר הופיע במערך (באינדקס 3). הוא עבר לתא 0, השירים 512, 87, 340 זזו תא אחד ימינה, ושאר השירים נשארו במקומם.

דוגמה נוספת: עבור אותו המערך המקורי ו-`song = 600`, הפעולה תחזיר `true`, והמערך ייראה כך בתום הפעולה:

7	6	5	4	3	2	1	0	
48	203	765	19	340	87	512	600	recent

הסבר: השיר 600 לא הופיע במערך, ולכן השיר 931 שבתא האחרון נמחק.

כתבו פעולה בוליאנית חיצונית ששמה `playSong` בשפת `Java` או `PlaySong` בשפת `C#`, המקבלת מערך מלא - `recent`, וקוד של שיר - `song`, מטיפוס שלם.

הפעולה מעדכנת את המערך `recent` לפי הכללים שלעיל, ומחזירה `true` אם השיר `song` **לא** הופיע במערך לפני ההשמעה. אם השיר כבר הופיע במערך, הפעולה מחזירה `false`.

3. (20 נקודות)

בפסטיבל קיץ מוכרים כרטיסי הגרלה. על כל כרטיס מודפס מספר שלם וגדול מ-0. כרטיס **זוכה** הוא כרטיס שכל הספרות במספר שלו שוות זו לזו (למשל: 7, 444, 8888).

הערה: אין צורך לבדוק את תקינות הקלט.

לפותרים בשפת `Java`: הניחו שבתוכנית כתובה ההוראה:

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

1. כתבו פעולה בשם `digitCount` (ב- `DigitCount` - C#) שתקבל מספר שלם `num` הגדול מ-0. הפעולה תחזיר מערך חד-ממדי של מספרים שלמים בגודל 10. התא שהמציין שלו $d \leq 0$ (כאשר $d \geq 0$) יכיל את מספר הפעמים שהספרה `d` מופיעה במספר `num`. לדוגמה, בעבור `num = 2026` הפעולה תחזיר את המערך: `1,0,2,0,0,0,1,0,0,0`.
2. כתבו פעולה בשם `raffle` (ב- `Raffle` - C#) שתקלוט את מספרי הכרטיסים שנמכרו בפסטיבל, לפי סדר מכירתם. הקלט מסתיים כאשר נקלט כרטיס זוכה. הפעולה תדפיס את הספרה שהופיעה הכי הרבה פעמים בכל הכרטיסים שנקלטו לפני הכרטיס הזוכה. אם יש כמה ספרות כאלה, תודפס הקטנה מביניהן. הניחו כי לפני הכרטיס הזוכה נקלט לפחות כרטיס אחד. **דוגמה:** אם נקלטו המספרים `1203, 3315, 90, 444` - יודפס 3. ניתן להשתמש בפעולה שכתבתם בסעיף 1.

פרק שני (60 נקודות)

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

4. (30 נקודות)

- נתונה המחלקה **Parking**, המייצגת חנייה של רכב באחד מחניוני העירייה, ולה ארבע תכונות:
- `lotID` - מספר החניון, מטיפוס שלם בין 0 ל-29 (כולל)
 - `ratePerHour` - התעריף לשעת חנייה, מטיפוס מספר ממשי
 - `hours` - מספר שעות החנייה, מטיפוס שלם
 - `resident` - האם הנהג הוא תושב העיר, מטיפוס בוליאני (`true` - בעבור תושב העיר, `false` - בעבור נהג שאינו תושב העיר)
- נהג שהוא תושב העיר מקבל הנחה של 40% מסכום החנייה. נוסף על כך, אם הסכום לתשלום (אחרי ההנחה, אם יש הנחה) **גבוה** מ-60.00 ש"ח - הנהג משלם 60.00 ש"ח בלבד (תעריף יומי מרבי). התעריף המרבי חל על כל הנהגים.
- הניחו שקיימות פעולות `get/Set` ו- `set/Set` לתכונות המחלקה.
- א. כתבו במחלקה **Parking** פעולה בונה המקבלת את מספר החניון - `lotID` ואת התעריף לשעה - `ratePerHour`. הפעולה תאתחל את תכונות המחלקה כך שתקבל חנייה בחניון `lotID`, בתעריף של `ratePerHour` לשעה, למשך שעה אחת, של נהג שאינו תושב העיר.
 - ב. (1) כתבו במחלקה **Parking** פעולה פנימית ששמה `basePrice` בשפת Java או `BasePrice` בשפת C#, המחזירה מספר ממשי של סכום החנייה לפי התעריף לשעה ומספר השעות, **ללא הנחה וללא התעריף היומי המרבי**.
 - (2) כתבו במחלקה **Parking** פעולה פנימית ששמה `finalPrice` בשפת Java או `FinalPrice` בשפת C#, המחזירה מספר ממשי של הסכום **הסופי** שהנהג צריך לשלם, לפי ההנחה לתושבי העיר והתעריף היומי המרבי שתוארו לעיל.
- הערה:** יש להשתמש בפעולה שכתבתם בתת-סעיף ב(1).

ג. נתון מערך - dayParkings מטיפוס **Parking** המכיל את כל החניות שהיו ביום מסוים בכל חניוני העירייה. המערך אינו מסודר לפי סדר כלשהו ואין בו ערכי null . כתבו פעולה חיצונית ששמה incomePerLot בשפת Java או IncomePerLot בשפת #C , המקבלת את המערך dayParkings . הפעולה מחזירה מערך חדש מטיפוס מספר ממשי בגודל 30 , שבכל תא מופיעה ההכנסה הכוללת של העירייה באותו יום מן החניון **שמספרו** זהה לאינדקס של התא. ההכנסה מכל חנייה היא הסכום הסופי שהנהג שילם. **למשל** , אם בחניון מספר 3 היו באותו יום שלוש חניות, והסכומים הסופיים שלהן הם 20.00 ש"ח, 60.00 ש"ח ו- 45.00 ש"ח - יופיע בתא של אינדקס 3 המספר 125.0 .

5. בעיר מסוימת הרחובות ממוספרים מדרום לצפון (רחוב 1 הוא הדרומי ביותר), והשדרות ממוספרות ממערב למזרח. המחלקה **Location** מייצגת מקום בעיר (צומת של רחוב ושדרה), ולה שתי תכונות: street - מספר שלם בין 1 ל- 120 (כולל), המייצג את מספר הרחוב . avenue - מספר שלם בין 1 ל- 12 (כולל), המייצג את מספר השדרה . הניחו שקיימות פעולות get/Get ו- set/Set עבור תכונות המחלקה, ופעולה בונה המקבלת ערכים עבור תכונות המחלקה. לפניהם ממשק המחלקה. אפשר להשתמש בפעולות המחלקה בלי לממש אותן.

תיאור הפעולה	כותרת הפעולה
הפעולה מחזירה true אם המקום הנוכחי נמצא צפונית למקום other (כלומר מספר הרחוב שלו גדול יותר), אחרת הפעולה מחזירה false .	Java - public boolean northOf (Location other) #C - public bool NorthOf (Location other)
הפעולה מחזירה את מספר ה בלוקים בין המקום הנוכחי ובין המקום other (ההפרש בערך מוחלט בין מספרי הרחובות ועוד ההפרש בערך מוחלט בין מספרי השדרות).	Java - public int blocks (Location other) #C - public int Blocks (Location other)

המחלקה **Delivery** מייצגת משלוח של שליח אופניים, ולה שלוש תכונות: orderId - קוד ההזמנה, מטיפוס מחרוזת . pickup - המקום שממנו נאסף המשלוח, מטיפוס Location . dropoff - המקום שאליו נמסר המשלוח, מטיפוס Location . הניחו שקיימות פעולות get/Get ו- set/Set עבור תכונות המחלקה, ושתכונות המחלקה אינן null . נתון מערך חד-ממדי orders מטיפוס **Delivery** המכיל מידע על כל המשלוחים שביצע שליח מסוים ביום אחד. **המערך אינו ממוין** ואינו מכיל ערכי null . א. כתבו פעולה חיצונית ששמה northmost בשפת Java או Northmost בשפת #C , המקבלת את המערך orders , ומדפיסה את קוד ההזמנה של המשלוח שנאסף מהמקום הצפוני ביותר. הניחו שאין שני משלוחים שנאספו מאותו רחוב. ב. (1) השליח מקבל 15 שקלים על כל משלוח, ובנוסף 2 שקלים על כל בלוק **החל מהבלוק ה- 11** של המשלוח (על 10 הבלוקים הראשונים אין תוספת). כתבו פעולה פנימית במחלקה Delivery ששמה distance בשפת Java או Distance בשפת #C , המחזירה את מספר הבלוקים בין מקום האיסוף ובין מקום המסירה של המשלוח.

(2) כתבו פעולה חיצונית ששמה totalPay בשפת Java או TotalPay בשפת #C, המקבלת את המערך orders, ומחזירה את הסכום הכולל שהשליח מקבל על כל המשלוחים. אפשר להשתמש בפעולה שכתבתם בסעיף ב(1).

6. (30 נקודות)

מספר שלם num הגדול מ-1 ייקרא "מספר ראשוני" אם אין לו מחלקים חיוביים ושלמים חוץ מ-1 ומעצמו.

שני מספרים ראשוניים שההפרש ביניהם הוא 2 נקראים "זוג ראשוניים תאומים".

דוגמה למספר ראשוני: המספר 13, כי המחלקים החיוביים שלו הם רק 1 ו-13.

דוגמה למספר שאינו ראשוני: המספר 15, כי הוא מתחלק גם ב-3 וגם ב-5.

דוגמה לזוג ראשוניים תאומים: 17, 19 – שניהם ראשוניים, וההפרש ביניהם הוא 2.

א. (1) כתבו פעולה חיצונית ששמה isPrime בשפת Java או IsPrime בשפת #C, המקבלת מספר שלם -

num הגדול מ-1. הפעולה **מחזירה** true אם הוא מספר ראשוני, אחרת היא מחזירה false.

(2) כתבו פעולה חיצונית ששמה printTwins בשפת Java או PrintTwins בשפת #C, המקבלת שני

מספרים שלמים הגדולים מ-1: low, high. הפעולה **תדפיס** את כל זוגות הראשוניים

התאומים ששני המספרים שבהם נמצאים בין low ל-high (כולל). כל זוג יודפס בשורה נפרדת, המספר הקטן תחילה.

הניחו ש-low קטן מ-high או שווה לו.

הערה: אפשר להיעזר בפעולה שכתבתם בתת-סעיף (1).

דוגמה: בעבור low=10 ו-high=45 הפעולה תדפיס:

13 11

19 17

31 29

43 41

ב. יש השערה שכל מספר זוגי הגדול מ-2 הוא סכום של שני מספרים ראשוניים (לא בהכרח שונים זה

מזה). למשל: $2 + 2 = 4$, $7 + 3 = 10$, $23 + 5 = 28$.

כתבו פעולה חיצונית ששמה goldbach בשפת Java או Goldbach בשפת #C, **המחזירה** true אם

ההשערה מתקיימת בעבור כל המספרים הזוגיים 4 עד 10,000 (כולל). אחרת היא **מחזירה** false.

הערה: אפשר להיעזר בפעולה שכתבתם בתת-סעיף א(1).

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