

סוג הבחינה: מבחן מתכונת
מבחן: מתכונת 5
מספר השאלון: 899271
שם התלמיד/ה: _____

מדעי המחשב

מבני נתונים ומסלול - 4-5 יחידות לימוד

הוראות

א. משך הבחינה: שלוש שעות.

ב. מבנה השאלון ומפתח ההערכה: בשאלון זה שני פרקים.

פרק ראשון - 2×25 - 50 נקודות

פרק שני - 2×25 - 50 נקודות

סך הכול - 100 נקודות

ג. חומר עזר מותר בשימוש: כל חומר עזר, חוץ ממחשבון שיש בו אפשרות תכנות.

ד. הוראות מיוחדות:

(1) רשמו על הכריכה החיצונית של המחברת את שם המסלול שלמדתם.

המסלול הוא אחד משלושת המסלולים האלה: אלגוריתמים, מודלים חישוביים, תכנות מונחה עצמים.

(2) את כל התוכניות שיש לכתוב בשפת מחשב בפרקים הראשון והשני 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 input = new Scanner(System.in);`

שימו לב: בכל שאלה שנדרש בה מימוש אפשר להשתמש בפעולות של המחלקות: תור, מחסנית, עץ בינרי וחוליה,

בלי לממש אותן. אם משתמשים בפעולות נוספות, יש לממש אותן.

השאלות

פרק ראשון

ענו על שתיים מן השאלות 1-3 (לכל שאלה - 25 נקודות).

1. (25 נקודות)

נוסח Java

לשאלה זו שני נוסחים: בשפת Java ובשפת C#. ענו לפי הנוסח של השפה שבה אתם כותבים.

לפותרים בשפת Java

א. (1) לפניכם הפעולה kkk, המקבלת תור q ובו מספרים הגדולים מ-0, ומספר שלם k שאינו שלילי.

Java

```
public static boolean kkk (Queue<Integer> q, int k)
{
    q.insert (0);
    int a = 0;
    int b = 0;
    int i = 0;
    while (q.head() != 0)
    {
        i++;
        if (i <= k)
        {
            a = a + q.head();
        }
        else
        {
            b = b + q.head();
        }
        q.insert (q.remove());
    }
    q.remove();
    return a == b;
}
```

נתון תור q מטיפוס שלם:

ראש התור		סוף התור				
	3	5	2	4	6	

(1) עקבו בעזרת טבלת המעקב שלפניכם אחר הפעולה kkk(q, 3), וכתבו מה הפעולה מחזירה.

התור q	i	i <= k	a	b

(2) הסבירו מה הפעולה kkk עושה.

(3) מהי סיבוכיות זמן הריצה של הפעולה kkk? נמקו את תשובתכם.

- ב. (1) לפניכם הפעולה `mystery`, המקבלת תור q ובו מספרים הגדולים מ-0, ומספר שלם n שאינו שלילי. הפעולה `mystery` משתמשת בפעולה `kkk` מסעיף א.

Java

```
public static int mystery (Queue<Integer> q, int n)
{
    if (n == 0)
        return 0;
    if (kkk (q, n))
        return n;
    return mystery (q, n - 1);
}
```

נתון התור q מסעיף א:

ראש התור	סוף התור
3	5
2	4
6	

- (1) עקבו אחר הפעולה `mystery (q, 5)`, וכתבו מה הפעולה מחזירה (אין צורך לעקוב אחר הפעולה `kkk`).
המעקב יכלול בכל קריאה את הערכים של n , ואת הערך המוחזר.
לפניכם הצעה לטבלת מעקב (אין חובה להשתמש בטבלה זו).

התור q שמתקבל בפעולה	הערך n שמתקבל בפעולה	<code>kkk (q, n) == true</code>	ערך מוחזר

- (2) הסבירו מה הפעולה `mystery` עושה כאשר n הוא מספר האיברים בתור q .
(3) מהי סיבוכיות זמן הריצה של הפעולה `mystery` כאשר n הוא מספר האיברים בתור q ? נמקו את תשובתכם.

נוסח #C

לשאלה זו שני נוסחים: בשפת `Java` ובשפת `#C`. ענו לפי הנוסח של השפה שבה אתם כותבים.

לפותרים בשפת #C

- א. (1) לפניכם הפעולה `Kkk`, המקבלת תור q ובו מספרים הגדולים מ-0, ומספר שלם k שאינו שלילי.

C#

```
public static bool Kkk (Queue<int> q, int k)
{
    q.Insert (0);
    int a = 0;
    int b = 0;
    int i = 0;
    while (q.Head() != 0)
    {
        i++;
        if (i <= k)
        {
            a = a + q.Head();
        }
    }
}
```

```

    }
    else
    {
        b = b + q.Head();
    }
    q.Insert (q.Remove());
}
q.Remove();
return a == b;
}

```

נתון תור q מטיפוס שלם:

ראש התור		סוף התור			
	3	5	2	4	6

(1) עקבו בעזרת טבלת המעקב שלפניכם אחר הפעולה $Kkk(q, 3)$, וכתבו מה הפעולה מחזירה.

התור q	i	$i \leq k$	a	b

(2) הסבירו מה הפעולה Kkk עושה.

(3) מהי סיבוכיות זמן הריצה של הפעולה Kkk ? נמקו את תשובתכם.

ב. (1) לפניכם הפעולה $Mystery$, המקבלת תור q ובו מספרים הגדולים מ-0, ומספר שלם n שאינו שלילי. הפעולה $Mystery$ משתמשת בפעולה Kkk מסעיף א.

```

C#
public static int Mystery (Queue<int> q, int n)
{
    if (n == 0)
        return 0;
    if (Kkk (q, n))
        return n;
    return Mystery (q, n - 1);
}

```

נתון התור q מסעיף א:

ראש התור		סוף התור			
	3	5	2	4	6

(1) עקבו אחר הפעולה $Mystery(q, 5)$, וכתבו מה הפעולה מחזירה (אין צורך לעקוב אחר הפעולה Kkk).

המעקב יכלול בכל קריאה את הערכים של n , ואת הערך המוחזר. לפניכם הצעה לטבלת מעקב (אין חובה להשתמש בטבלה זו).

התור q שמתקבל בפעולה	הערך n שמתקבל בפעולה	Kkk (q, n) == true	ערך מוחזר

(2) הסבירו מה הפעולה Mystery עושה כאשר n הוא מספר האיברים בתור q.

(3) מהי סיבוכיות זמן הריצה של הפעולה Mystery כאשר n הוא מספר האיברים בתור q? נמקו את תשובתכם.

2. (25 נקודות)

נתונה המחלקה Song - שיר, ולה שתי תכונות:

title - שם השיר, מטיפוס מחרוזת.

length - אורך השיר בשניות - מספר הגדול מ-0, מטיפוס שלם.

הניחו שיש פעולות get/Set ו-set/Set לתכונות המחלקה.

נתונה המחלקה Playlist - רשימת השמעה, ולה תכונה אחת:

lst - הפניה לשרשרת חוליות שאינה ריקה, מטיפוס Song. כל חוליה בשרשרת מכילה שיר מרשימת ההשמעה, לפי סדר ההשמעה.

הערה: השירים אינם מסודרים בשרשרת לפי אורכם, וכל שיר מופיע פעם אחת בלבד.

א. ממשו את הפעולה של ממשק המחלקה Playlist שלפניכם:

Java - public int removeLastLong (int n, int t)

C# - public int RemoveLastLong (int n, int t)

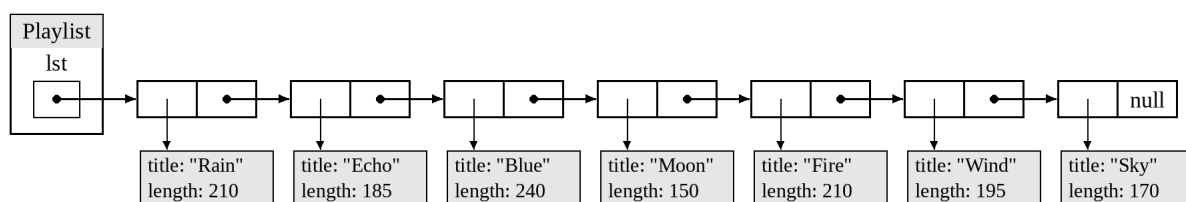
הפעולה תמחק מן השרשרת את n השירים האחרונים בשרשרת שאורך כל אחד מהם גדול מ-t. אם יש בשרשרת פחות מ-n שירים שאורכם גדול מ-t, יימחקו כל השירים האלה.

הפעולה תחזיר את כמות השירים שנמחקו.

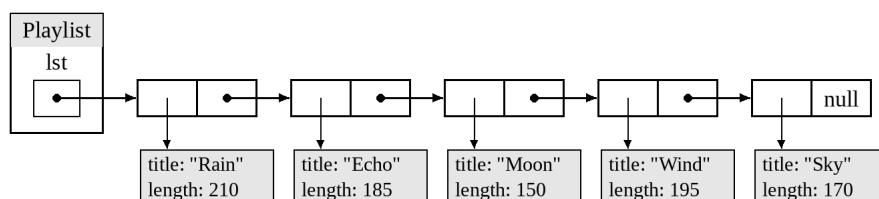
הניחו ש-n גדול מ-0 וקטן מכמות השירים בשרשרת, וש-t אינו שלילי.

הערה: שאר השירים בשרשרת יישארו באותו הסדר. אם אין בשרשרת שום שיר שאורכו גדול מ-t, השרשרת תישאר ללא שום שינוי והפעולה תחזיר 0.

דוגמה: בעבור $n = 2$, $t = 200$ והשרשרת lst שלפניכם:

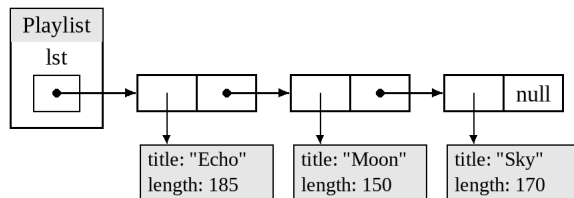


הפעולה תחזיר 2 והשרשרת תיראה כך בתום הפעולה:



הסבר: בשרשרת יש שלושה שירים שאורכם גדול מ-200: "Rain", "Blue" ו-"Fire". מכיוון ש-n = 2, נמחקו שני האחרונים מביניהם - "Blue" ו-"Fire", והשיר "Rain" נשאר.

דוגמה נוספת: בעבור אותה השרשרת מן הדוגמה המקורית שלעיל ו-n = 5, t = 190, הפעולה תחזיר 4 והשרשרת תיראה כך בתום הפעולה:



הסבר: יש בשרשרת רק ארבעה שירים שאורכם גדול מ-190 ("Wind" ו-"Rain", "Blue", "Fire").
לכן נמחקו כל ארבעתם, והפעולה החזירה 4.

ב. ממשו את הפעולה של ממשק המחלקה **Playlist** שלפניכם:

Java - `public int shorten (int maxTotal)`

C# - `public int Shorten (int maxTotal)`

הפעולה תקצר את רשימת ההשמעה כך שסכום האורכים של השירים שישארו בשרשרת יהיה **לכל היותר** `maxTotal`: כל עוד סכום האורכים של השירים בשרשרת גדול מ-`maxTotal`, הפעולה תמחק מן השרשרת את השיר הארוך ביותר. אם יש כמה שירים ארוכים ביותר (באותו אורך), יימחק **האחרון** מביניהם.

הפעולה תחזיר את כמות השירים שנמחקו.

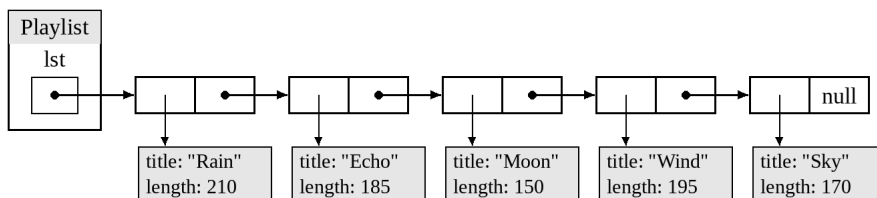
הניחו ש-`maxTotal` גדול או שווה לאורך של כל אחד מן השירים בשרשרת.

הערות:

. השירים בשרשרת שלא נמחקו יישארו באותו הסדר.

. אפשר להשתמש בפעולה שבסעיף א.

דוגמה: בעבור אותה השרשרת מן הדוגמה המקורית בסעיף א ו-`maxTotal = 1000`, הפעולה תחזיר 2, והשרשרת תיראה כך:



הסבר: סכום האורכים בהתחלה הוא 1360. נמחק השיר הארוך ביותר ("Blue" (240), והסכום ירד ל-1120. הסכום עדיין גדול מ-1000, ולכן נמחק שוב השיר הארוך ביותר: יש שני שירים באורך 210 ("Rain" ו-"Fire"), ונמחק האחרון מביניהם - "Fire". הסכום ירד ל-910, שאינו גדול מ-1000, והפעולה החזירה 2.

3. (25 נקודות)

בעץ בינרי, **הרמה** של צומת היא מספר הקשתות במסלול מן השורש אל הצומת: רמת השורש היא 0, רמת הבנים של השורש היא 1, וכן הלאה.

"זוג נגדי" בעץ הוא שני מספרים בעץ שאחד מהם הוא המספר הנגדי של השני (למשל 5 ו-5-). **פער הרמות** של זוג נגדי הוא ההפרש, בערך מוחלט, בין הרמות של שני המספרים.

נתון עץ בינרי `t` מטיפוס שלם, המכיל מספרים חיוביים ושליילים שאינם 0, ו**כולם שונים זה מזה** (כלומר אין שני מספרים זהים בעץ).

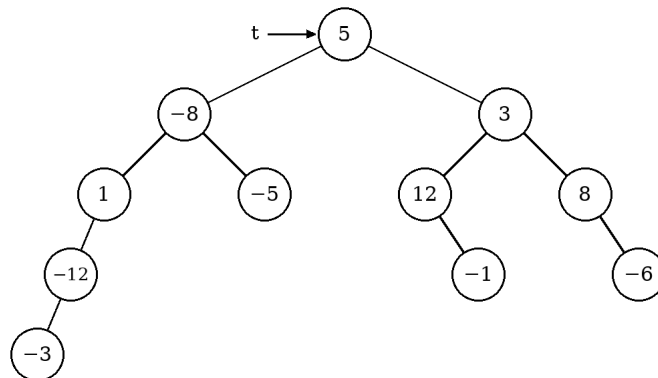
א. כתבו פעולה ששמה levelGap בשפת Java או LevelGap בשפת C המקבלת את העץ t ומספר שלם num (חיובי או שלילי) המופיע בעץ.

הפעולה תחזיר את פער הרמות של הזוג הנגדי ש- num שייך אליו. אם המספר הנגדי ל- num אינו מופיע בעץ, הפעולה תחזיר -1. אפשר לכתוב פעולות עזר.

הערות:

- חובה לשמור על העץ t.

- אין להשתמש בסעיף זה במערך. פתרון הכולל שימוש במערך לא יזוכה בנקודות. **דוגמאות:** עבור העץ t שלפניכם:



עבור $num = 5$ הפעולה תחזיר 2 (5 ברמה 0, ו-5 ברמה 2).

עבור $num = -1$ הפעולה תחזיר 1 (1 ברמה 3, ו-1 ברמה 2).

עבור $num = -6$ הפעולה תחזיר -1 (משום שהמספר הנגדי לו, 6, אינו מופיע בעץ).

ב. כתבו פעולה ששמה maxGap בשפת Java או MaxGap בשפת C המקבלת את העץ t. הפעולה תחזיר את פער הרמות הגדול ביותר של זוג נגדי בעץ. אם אין בעץ שום זוג נגדי, הפעולה תחזיר -1. אפשר להשתמש בפעולה שכתבתם בסעיף א.

הערות:

- בפעולה זו אין חובה לשמור על העץ t.

- אין להשתמש בסעיף זה במערך. פתרון הכולל שימוש במערך לא יזוכה בנקודות.

לדוגמה: עבור העץ t שבדוגמה לעיל הפעולה תחזיר 3.

הסבר: בעץ חמישה זוגות נגדיים: 5 ו-2 (פערי 2), 8 ו-1 (פערי 1), 1 ו-12 (פערי 1), 12 ו-12 (פערי 0), 3 ו-3 (פערי 3). לזוג 3 ו-3 פער הרמות הגדול ביותר.

פרק שני (50 נקודות) - אלגוריתמים

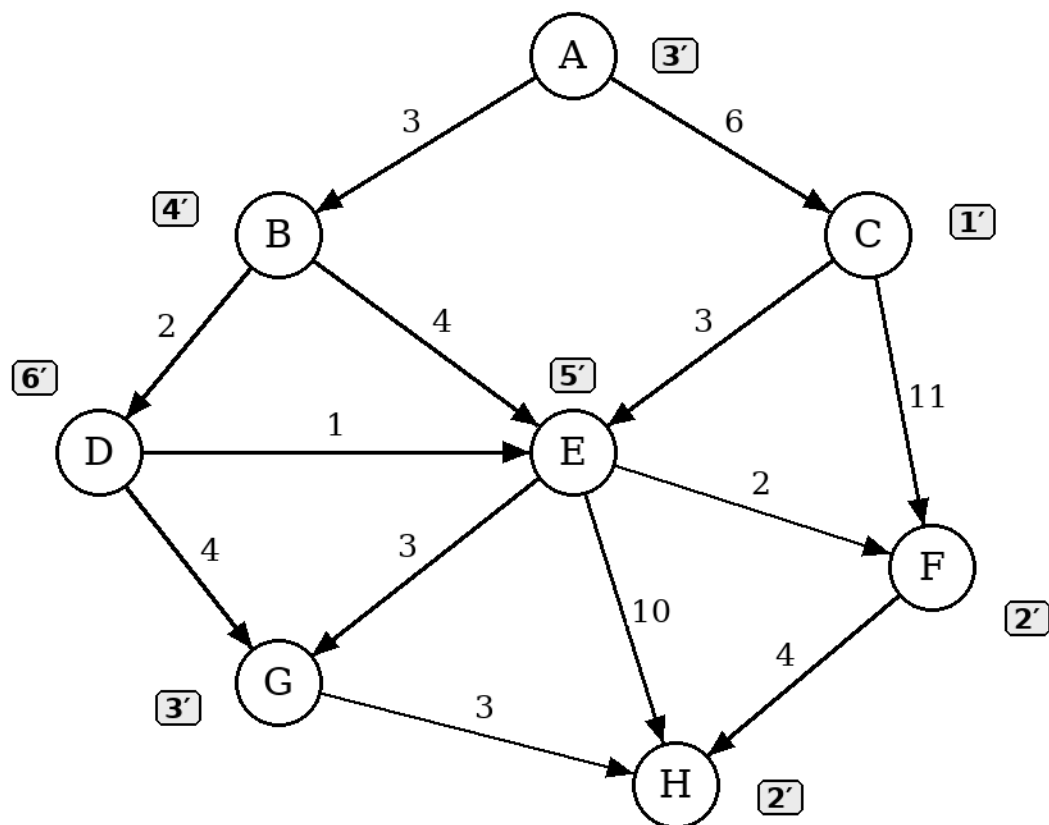
יש לענות על שתי שאלות במסלול שלמדתם (לכל שאלה - 25 נקודות).

4. (25 נקודות)

באתר סקי יש תחנות על צלע ההר, ובין חלק מן התחנות יש **מסלולי גלישה**. כל מסלול גלישה **יורד** מתחנה גבוהה יותר לתחנה נמוכה יותר, ולכן אפשר לגלוש בו רק בכיוון אחד (בכיוון החץ). לכל מסלול גלישה יש **זמן גלישה** בדקות. בכל תחנה יש נקודת ביקורת, וגולש **שעובר דרך** התחנה חייב לעצור בה; לכל תחנה יש **זמן עצירה** בדקות.

הזמן של דרך בין שתי תחנות הוא סכום זמני הגלישה של מסלולי הגלישה בדרך, ועוד סכום זמני העצירה של התחנות שהגולש עובר דרכן. הזמן **אינו כולל** את זמן העצירה בתחנה שבה הגולש מתחיל ואת זמן העצירה בתחנה שבה הוא מסיים. הדרך **המהירה ביותר** בין שתי תחנות היא הדרך שהזמן שלה הקטן ביותר.

לפניכם אתר הסקי SKI. המספר ליד כל חץ הוא זמן הגלישה במסלול, והמספר שבמלבן ליד כל תחנה הוא זמן העצירה בה (בדקות):



דוגמה: הזמן של הדרך $A \rightarrow B \rightarrow D \rightarrow G \rightarrow H$ הוא 25 דקות: זמני הגלישה $3 + 4 + 2 + 3 = 12$, ועוד זמני העצירה בתחנות B, D, G (בלי A ובלי H): $4 + 6 + 3 = 13$.

א. באתר SKI, מהי הדרך המהירה ביותר מתחנה A לתחנה H ? כתבו את שמות התחנות בדרך זו לפי הסדר, ואת הזמן שלה (אין צורך לבצע מעקב).

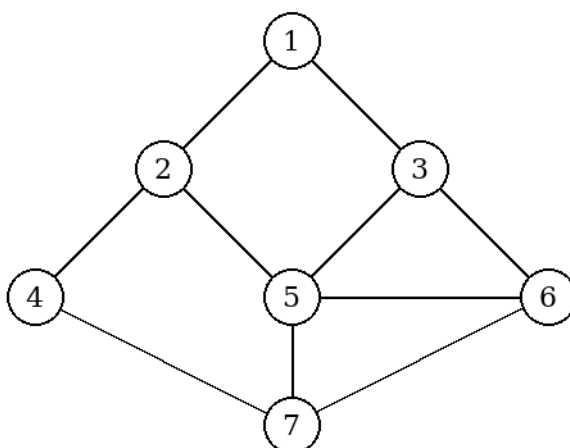
ב. (1) כתבו אלגוריתם **יעיל ככל האפשר** המוצא בכל אתר סקי כזה את הדרך המהירה ביותר מתחנה K_1 לתחנה K_2 .

הערה: יש לכתוב אלגוריתם יעיל שאינו עובר על כל הדרכים האפשריות.

(2) סרטטו את הגרף המייצג את אתר הסקי SKI הנתון לעיל, באופן שיתאים לאלגוריתם שכתבתם.

5. (25 נקודות)

לפניכם גרף לא מכוון $G = (V, E)$.



צמתי הגרף	$V = \{1, 2, 3, 4, 5, 6, 7\}$
קשתות הגרף	$E = \{(1-2), (1-3), (2-4), (2-5), (3-5), (3-6), (4-7), (5-6), (5-7), (6-7)\}$

- א. הצומת M מוגדר "שכן" של הצומת U, אם יש קשת ביניהם. לפניכם טבלת שכנויות המציגה את השכנים של כל צומת בגרף G, מסודרים מן הקטן אל הגדול. הצמתים השכנים של 1 ושל 4 נתונים בטבלה. השלימו את הטבלה.

U	1	2	3	4	5	6	7
M	3, 2			7, 2			

- ב. נתון האלגוריתם $\text{TwoColors}(G, s)$, שמנסה לצבוע את צמתי הגרף בשני צבעים, 0 ו-1, כך שלכל שני צמתים שכנים יהיו צבעים שונים. נריץ את האלגוריתם מן הצומת $s = 1$. שלב אתחול נתונים באלגוריתם:

- נגדיר תור - q, שאליו נכניס את הצומת 1.
- נגדיר מערך color בגודל 7, ובו נשמור את הצבע של כל צומת (בהתחלה בתא של הצומת 1 מופיע הערך 0, ושאר התאים במערך ריקים).

color:	1	2	3	4	5	6	7
	0						

- נגדיר מערך parent בגודל 7, ובו נשמור בעבור כל צומת את ה"הורה" שלו, כלומר את הצומת שממנו הגענו אליו במהלך ריצת האלגוריתם (בהתחלה כל התאים במערך ריקים).

parent:	1	2	3	4	5	6	7

שלב הרצת האלגוריתם:

- כל עוד התור - q אינו ריק:
 - נוציא את הצומת הנמצא בראש התור, ונקרא לו U.
 - בעבור כל צומת שכן - M של הצומת U, לפי הסדר שבטבלת השכנויות מסעיף א:

. אם התא של M במערך color ריק -

- נכתוב בתא של M במערך color את הצבע ההפוך לצבע של U (כלומר את הערך $1 - \text{color}[U]$).
- נעדכן שהצומת U הוא ההורה של M במערך parent.
- נכניס את M לתור.

. **אחרת, אם** $\text{color}[M] = \text{color}[U]$ - האלגוריתם מחזיר רשימת צמתים שהם: M, U,

ההורה של U, ההורה של ההורה של U וכן הלאה, עד הצומת שאין לו הורה (כלומר

לצומת שהתא שלו ריק במערך parent), **והאלגוריתם מסתיים**.

. **אחרת** (הצבעים של M ושל U שונים) - לא עושים דבר.

2. אם התור ריק, נחזיר null, **והאלגוריתם מסתיים**.

עקבו בטבלת מעקב אחרי ריצת האלגוריתם TwoColors על הגרף הנתון G, מן הצומת 1. המעקב צריך לכלול בכל איטרציה את הפריטים שלהלן:

. מצב התור

. הצומת U שהוצאנו מן התור

. מערך color

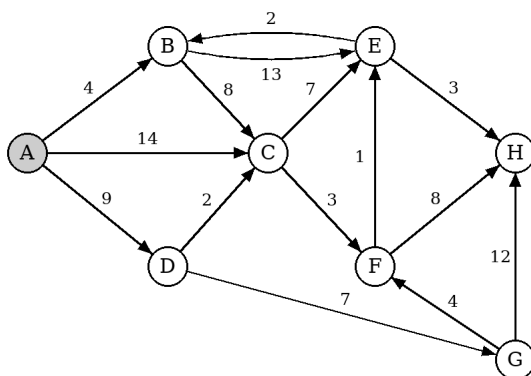
. מערך parent

התור	U	color	1	2	3	4	5	6	7	parent	1	2	3	4	5	6	7
← []																	
← []																	

ג. כתבו את רשימת הצמתים שהאלגוריתם מחזיר.

6. (25 נקודות)

לפניכם הגרף המכוון הממושקל G:



- כתבו אלגוריתם המוצא בגרף מכוון ממושקל (אי-שלילי) כלשהו, שבו n קודקודים מ- v_0 עד v_{n-1} , וקודקוד בגרף v_j , את המסלולים הקלים ביותר מקודקוד v_j אל שאר הקודקודים שבגרף. בנוסף, לכל קודקוד v האלגוריתם יחזיר את מספר הקשתות שבמסלול הקל ביותר שנמצא מ- v_j אל v.
- מהי סיבוכיות זמן הריצה של האלגוריתם? נמקו את תשובתכם.

- ב. (1) בעבור גרף G הנתון, מצאו בעזרת האלגוריתם שכתבתם את המסלול הקל ביותר מקודקוד A לכל אחד מן הקודקודים, וסרטטו טבלת מעקב כמפורט:
- המעקב יכול לכל איטרציה את קבוצת הקודקודים הקבועים (שכבר ביקרנו בהם) - P ואת קבוצת הקודקודים הזמניים (שבהם עדיין לא ביקרנו) - T . נוסף על כך, בעבור כל קודקוד יצוין אורך המסלול עד אליו וזהות הקודקוד הקודם לו ("הורה" שלו).
- (2) בעבור גרף G הנתון, סרטטו את עץ המסלולים הקלים ביותר (מקודקוד A).

פרק שני (50 נקודות) - מודלים חישוביים

יש לענות על שתי שאלות במסלול שלמדתם (לכל שאלה - 25 נקודות).

7. (25 נקודות)

לפניכם שני סעיפים א-ב, שאין קשר ביניהם. ענו על שני הסעיפים.

- א. (1) לפניכם שלוש שפות $L_1 - L_3$ מעל הא"ב $\{a, b\}$ (בכל השפות n ו- m הם מספרים שלמים, $n, m \geq 0$):

$$L_1 = \{a^n b a^m \mid m < 30, n \geq m\}$$

$$L_2 = \{a^n b a^m \mid n \geq m > 30\}$$

$$L_3 = \{a^n b a^m \mid 0 < n \cdot m < 100\}$$

בנוגע לכל אחת מן השפות $L_1 - L_3$, כתבו אם היא רגולרית או לא. נמקו את תשובתכם (די בהסבר מילולי; אין צורך באוטומט).

- (2) לפניכם שלוש שפות $L_4 - L_6$ מעל הא"ב $\{a, b\}$:

$$L_4 = L_1 \cup L_2$$

$$L_5 = L_2 \cap R(L_2)$$

$$L_6 = L_2 \cap L_3$$

בנוגע לכל אחת מן השפות $L_4 - L_6$, כתבו אם היא רגולרית או לא. נמקו את תשובתכם (די בהסבר מילולי; אין צורך באוטומט).

- ב. לפניכם השפה L מעל הא"ב $\{a, b, c\}$:

$$L = \{a^n b^m c^k w \mid n, m, k > 0\}$$

<

w היא מילה מעל הא"ב $\{a, b\}$ המאופיינת באופן שלהלן:

- אם $n + 2m$ מתחלק ב-3 בלי שארית, w היא מילה שמתחילה באות a .

- אם $n + 2m$ מתחלק ב-3 עם שארית 1, w היא מילה שאורכה 2 לכל היותר (גם המילה הריקה מתאימה).

- אם $n + 2m$ מתחלק ב-3 עם שארית 2, w היא מילה ריקה.

דוגמה למילה השייכת לשפה L ($a^n = aa, b^m = b, c^k = cc, w = ba$): $L : aa b cc ba$

הסבר: $n = 2, m = 1, n + 2m = 4$. $n + 2m$ מתחלק ב-3 עם שארית 1, ו- w היא מילה שאורכה 2.

בנו אוטומט סופי דטרמיניסטי שאינו מלא המקבל את השפה L .

8. (25 נקודות)

נתונה השפה L מעל הא"ב $\{a, b, c\}$:

$$L = \{a^n b^m c^k \mid n, m, k \geq 0\}$$

$$n \% 2 = 0: m = n + k, n, k > 0$$

$$n \% 2 = 1: k = n + m, m \geq 0$$

{

דוגמה למילה בשפה L בעבור n זוגי: $(n=2, m=5, k=3)$: aa bbbbbb ccc

הסבר: $n \% 2 = 0$, ומתקיים $3 + 2 = 5$, וגם $n, k > 0$.

דוגמה למילה בשפה L בעבור n אי-זוגי: $(n=3, m=1, k=4)$: aaa b cccc

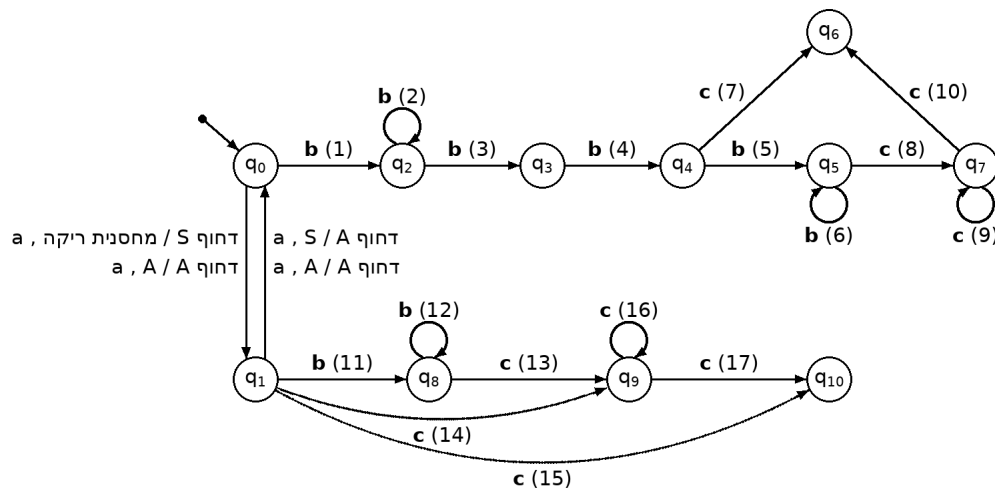
הסבר: $n \% 2 = 1$, ומתקיים $1 + 3 = 4$.

א. לפניכם חמש מילים. בעבור כל אחת מהן, ציינו אם המילה שייכת לשפה L . נמקו את תשובתכם.

abcc, aabb, aaaccc, aabbbbcc, aaabcc

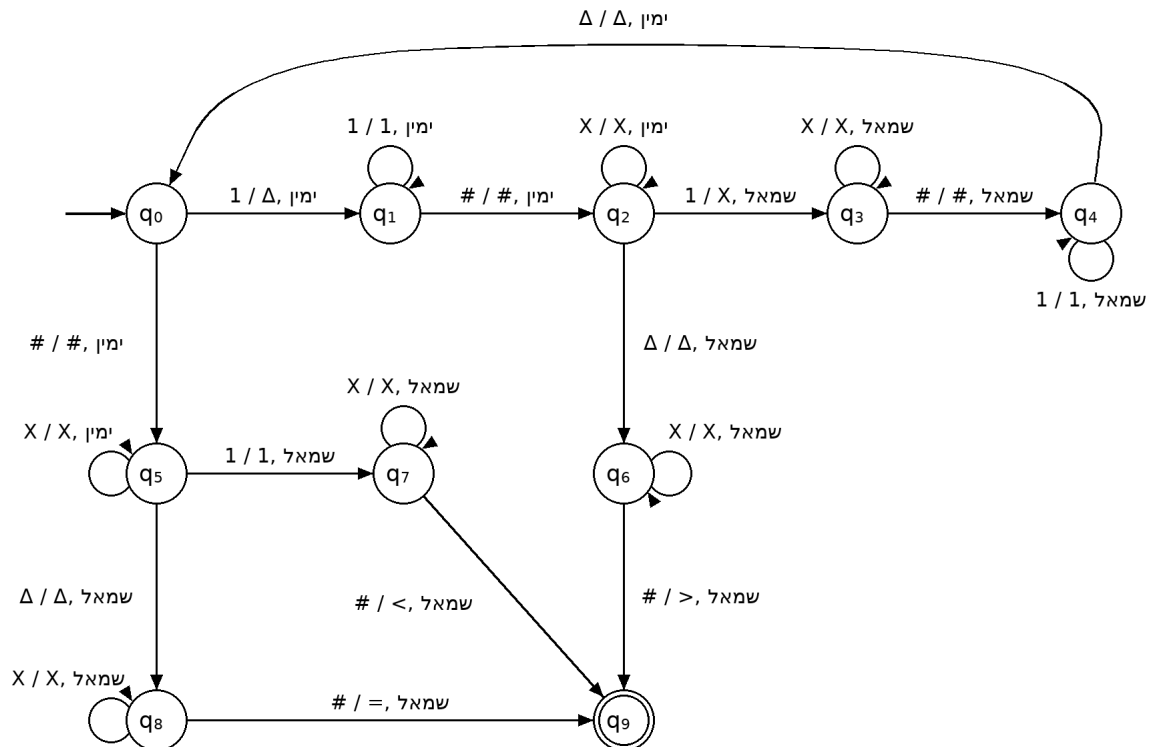
ב. כתבו את המילה הקצרה ביותר בשפה L בעבור n זוגי, ואת המילה הקצרה ביותר בשפה L בעבור n אי-זוגי.

ג. לפניכם אוטומט מחסנית דטרמיניסטי המקבל את השפה L . באוטומט קיימים כל המצבים, המעברים וסימני הקלט הנדרשים. אולם מלבד המעברים בין q_0 ל- q_1 , חסרות במעברים האות שבראש המחסנית והפעולה על המחסנית (מעברים אלה ממוספרים מ-1 עד 17). נוסף על כך, המצבים המקבלים באוטומט לא סומנו. סימני המחסנית שבהם משתמש האוטומט הם S, A, B . בעבור כל מעבר ממוספר, ציינו את מספרו והשלימו את הסימון החסר: האות שבראש המחסנית והפעולה שעל המחסנית (דחוף/pop או שלוף/push או ללא שינוי). נוסף על כך, כתבו מה הם המצבים המקבלים באוטומט (אין צורך להעתיק את האוטומט למחברתכם).



9. (25 נקודות)

לפניכם מכונת טיורינג:



על כל מעבר רשום: הסימן הנקרא / הסימן הנכתב, כיוון התזוזה. q_0 הוא המצב ההתחלתי, ו- q_9 הוא מצב העצירה.

המכונה מקבלת בתחילת הסרט שני מספרים אונריים שביניהם הסימן $\#$. בתחילת הפעולה הראש נמצא על התא הראשון שמימין ל- $\neg$.

א. עקבו אחר המכונה עבור הסרט שלהלן:

...	Δ	Δ	1	#	1	1	$\neg$
-----	----------	----------	---	---	---	---	--------

כתבו מה יתקבל על הסרט בסיום הפעולה.

ב. כתבו מה יתקבל על הסרט שלהלן בסיום הפעולה:

...	Δ	Δ	1	1	#	1	1	$\neg$
-----	----------	----------	---	---	---	---	---	--------

הערה: בסעיף זה אין צורך להראות מעקב.

ג. מה המכונה מבצעת עבור שני מספרים אונריים הגדולים מאפס?

ד. לפניכם שלושה מקרים, 1-3. בעבור כל מקרה ציינו אם המכונה פועלת נכון או לא נכון.

אם המכונה פועלת נכון, הציגו באמצעות דוגמה את סרט הקלט ואת סרט הפלט. אם המכונה פועלת לא נכון, הסבירו מדוע.

1. המספר הראשון (משמאל) שווה לאפס והמספר השני גדול מאפס.

2. המספר השני שווה לאפס והמספר הראשון גדול מאפס.

3. שני המספרים שווים לאפס.

תכנות מונחה עצמים

יש לענות על שתי שאלות במסלול שלמדתם (לכל שאלה - 25 נקודות).

10. (25 נקודות)

נוסח Java

כדי לנהל את המשלוחים שלה בנתה חברת השליחויות "מדלת לדלת" את המחלקה **Address** – כתובת, שלה שלוש תכונות: עיר – city (מטיפוס מחרוזת), רחוב – street (מטיפוס מחרוזת), מספר בית – number (מטיפוס שלם).

נוסף על כך, לצורך ניהול המשלוחים נבנתה המחלקה **Shipment** – משלוח.

לחברה יש שלושה סוגי משלוחים: חבילות, מכתבים ומשטחים (משטח שעליו ארגזים). כדי לשמור את המידע שבכל סוג משלוח, בנתה החברה את המחלקות האלה: חבילה – **Parcel**, מכתב – **Letter**, משטח – **Pallet**.

התכונות של המחלקות הן בהתאם לדרישות המידע שלפניכם:

המידע ששומרים על חבילה – **Parcel**:

- code – קוד המשלוח, מטיפוס מחרוזת
- weight – משקל המשלוח בק"ג, מטיפוס ממשי
- address – כתובת היעד, מטיפוס Address
- fragile – האם החבילה שבירה, מטיפוס בוליאני

המידע ששומרים על מכתב – **Letter**:

- code – קוד המשלוח, מטיפוס מחרוזת
- weight – משקל המשלוח בק"ג, מטיפוס ממשי
- address – כתובת היעד, מטיפוס Address
- registered – האם המכתב רשום, מטיפוס בוליאני

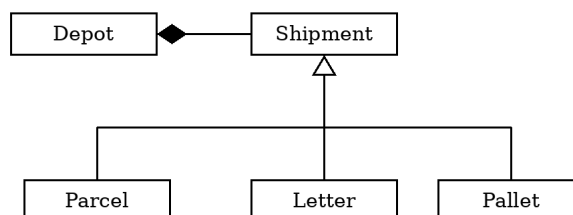
המידע ששומרים על משטח – **Pallet**:

- code – קוד המשלוח, מטיפוס מחרוזת
- weight – משקל המשלוח בק"ג, מטיפוס ממשי
- address – כתובת היעד, מטיפוס Address
- boxWeights – המשקלים (בק"ג) של הארגזים שעל המשטח, במערך מטיפוס ממשי (גודל המערך הוא לפי מספר הארגזים)

המידע ששומרים על משלוח – **Shipment**:

אותן דרישות המידע המפורטות בסוגי המשלוחים שלעיל ושצריכות להישמר במחלקה **Shipment** בהתאם לעקרונות תכנות מונחה עצמים.

נוסף על המחלקות האמורות, נבנתה המחלקה **Depot** – מחסן, השומרת עד 3,000 משלוחים – **Shipment**, במערך – arr בגודל 3,000 מטיפוס **Shipment**. לפניהם תרשים חלקי של המחלקות:



שימו לב: ירושה מסומנת באמצעות החץ <--- (המשולש הריק בצד מחלקת-העל), והכללה מסומנת באמצעות הסימן ◆--- (המעוין המלא בצד המחלקה המכילה).

על הסעיפים שלפניכם יש לענות לפי עקרונות של תכנות מונחה עצמים.

- א. העתיקו את התרשים למחברת והוסיפו את המחלקה Address לתרשים. כתבו את כותרת המחלקה משלוח – Shipment ואת התכונות שלה.
- הניחו שבשאר המחלקות כתובות התכונות המתאימות, ושכל המחלקות יש פעולות בונות ופעולות get ו-set.
- ב. כתבו במחלקה Depot פעולה ששמה registeredLetters המקבלת עצם – adr מטיפוס Address, ומחזירה מערך מטיפוס Letter בגודל 60, המכיל את כל המכתבים הרשומים הנשלחים לעיר שבכתובת adr.

שימו לב:

הניחו שקיימת הפעולה שלפניכם במחלקה Address :

Java

```
public boolean sameCity (Address other)
```

הפעולה מחזירה true אם העיר בכתובת הנוכחית זהה לעיר בכתובת other, אחרת היא מחזירה false. אפשר להשתמש בפעולה בלי לממש אותה.

הניחו שיש בדיוק 60 מכתבים רשומים הנשלחים לעיר שבכתובת adr.

הניחו כי בעת הקריאה לפעולה זו, מערך המשלוחים – arr במחלקה Depot מלא בעצמים (ללא null).

- ג. לפניכם כותרת הפעולה הפנימית – needsCare במחלקה Shipment :

Java

```
public boolean needsCare()
```

זימון הפעולה יחזיר true אם המשלוח (Shipment) הוא חבילה (Parcel) שבירה, או אם המשלוח הוא משטח (Pallet) שיש עליו לפחות ארגז אחד שמשקלו גדול מ-30 ק"ג, אחרת יוחזר false.

ממשו את הפעולה, והוסיפו פעולות למחלקות המתאימות כדי לבצע את הנדרש. ציינו בעבור כל פעולה באיזו מחלקה היא מוספת.

הערה: בסעיף זה אין להשתמש בפעולה instanceof ובפעולות של המחלקה Object.

נוסח #C

כדי לנהל את המשלוחים שלה בנתה חברת השליחויות "מדלת לדלת" את המחלקה **Address** – כתובת, שלה שלוש תכונות: עיר – city (מטיפוס מחרוזת), רחוב – street (מטיפוס מחרוזת), מספר בית – number (מטיפוס שלם).

נוסף על כך, לצורך ניהול המשלוחים נבנתה המחלקה **Shipment** – משלוח.

לחברה יש שלושה סוגי משלוחים: חבילות, מכתבים ומשטחים (משטח שעליו ארגזים). כדי לשמור את המידע שבכל סוג משלוח, בנתה החברה את המחלקות האלה: חבילה – **Parcel**, מכתב – **Letter**, משטח – **Pallet**.

התכונות של המחלקות הן בהתאם לדרישות המידע שלפניכם:

המידע ששומרים על חבילה – **Parcel**:

- code – קוד המשלוח, מטיפוס מחרוזת
- weight – משקל המשלוח בק"ג, מטיפוס ממשי
- address – כתובת היעד, מטיפוס Address
- fragile – האם החבילה שבירה, מטיפוס בוליאני

המידע ששומרים על מכתב – **Letter**:

- code – קוד המשלוח, מטיפוס מחרוזת
- weight – משקל המשלוח בק"ג, מטיפוס ממשי
- address – כתובת היעד, מטיפוס Address
- registered – האם המכתב רשום, מטיפוס בוליאני

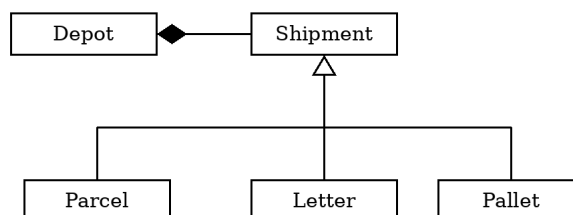
המידע ששומרים על משטח – **Pallet**:

- code – קוד המשלוח, מטיפוס מחרוזת
- weight – משקל המשלוח בק"ג, מטיפוס ממשי
- address – כתובת היעד, מטיפוס Address
- boxWeights – המשקלים (בק"ג) של הארגזים שעל המשטח, במערך מטיפוס ממשי (גודל המערך הוא לפי מספר הארגזים)

המידע ששומרים על משלוח – **Shipment**:

אותן דרישות המידע המפורטות בסוגי המשלוחים שלעיל ושצריכות להישמר במחלקה **Shipment** בהתאם לעקרונות תכנות מונחה עצמים.

נוסף על המחלקות האמורות, נבנתה המחלקה **Depot** – מחסן, השומרת עד 3,000 משלוחים – **Shipment**, במערך – arr בגודל 3,000 מטיפוס **Shipment**. לפניהם תרשים חלקי של המחלקות:



שימו לב: ירושה מסומנת באמצעות החץ <--- (המשולש הריק בצד מחלקת-העל), והכלה מסומנת באמצעות הסימן ---◆ (המעוין המלא בצד המחלקה המכילה).

על הסעיפים שלפניכם יש לענות לפי עקרונות של תכנות מונחה עצמים.

- א. העתיקו את התרשים למחברת והוסיפו את המחלקה Address לתרשים. כתבו את כותרת המחלקה משלוח – Shipment ואת התכונות שלה.
- הניחו שבשאר המחלקות כתובות התכונות המתאימות, ושכל המחלקות יש פעולות בונות ופעולות Get ו- Set.
- ב. כתבו במחלקה Depot פעולה ששמה RegisteredLetters המקבלת עצם – adr מטיפוס Address, ומחזירה מערך מטיפוס Letter בגודל 60, המכיל את כל המכתבים הרשומים הנשלחים לעיר שבכתובת adr.

שימו לב:

הניחו שקיימת הפעולה שלפניכם במחלקה Address :

C#

```
public bool SameCity (Address other)
```

- הפעולה מחזירה true אם העיר בכתובת הנוכחית זהה לעיר בכתובת other, אחרת היא מחזירה false. אפשר להשתמש בפעולה בלי לממש אותה.
- הניחו שיש בדיוק 60 מכתבים רשומים הנשלחים לעיר שבכתובת adr.
- הניחו כי בעת הקריאה לפעולה זו, מערך המשלוחים – arr במחלקה Depot מלא בעצמים (ללא null).

- ג. לפניכם כותרת הפעולה הפנימית – NeedsCare במחלקה Shipment :

C#

```
public virtual bool NeedsCare()
```

- זימון הפעולה יחזיר true אם המשלוח (Shipment) הוא חבילה (Parcel) שבירה, או אם המשלוח הוא משטח (Pallet) שיש עליו לפחות ארגז אחד שמשקלו גדול מ- 30 ק"ג, אחרת יוחזר false.
- ממשו את הפעולה, והוסיפו פעולות למחלקות המתאימות כדי לבצע את הנדרש. ציינו בעבור כל פעולה באיזו מחלקה היא מוספת.
- הערה:** בסעיף זה אין להשתמש בפעולות is ו- as ובפעולות של המחלקה object.

11. (25 נקודות)

נוסח Java

לשאלה זו שני נוסחים: בשפת Java ובשפת #C .

לפותרים בשפת Java

נתונות המחלקות Card , Gift , Wish :

Java

```
public class Card {
    public static int made = 0;
    private int code;
    public Card() {
        made++;
        code = made * 10;
    }
    public Card (int c) {
        code = c;
    }
    public String toString() {
        return code + " of " + made;
    }
}

-----

public class Gift extends Card {
    private String text;
    public Gift() {
        text = "Hi";
    }
    public Gift (String t) {
        super (7);
        text = t;
    }
    public String toString() {
        return text + ": " + super.toString();
    }
}

-----

public class Wish extends Gift {
    private int stars;
    public Wish (int s) {
        super ("Wow");
        stars = s;
    }
    public String toString() {
        return super.toString() + " " + stars + "*";
    }
}
```

א. נתונה המחלקה Test :

Java

```
public class Test
{
    public static void main (String[] args)
    {
        Card[] arr = new Card[6];
        arr[0] = ... // (1)
        arr[1] = ... // (2)
        arr[2] = ... // (3)
    }
}
```

```

arr[3] = ... // (4)
arr[4] = ... // (5)
arr[5] = ... // (6)
for (int i = 0; i < arr.length; i++) {
    System.out.println (arr[i]);
}
}
}

```

השלימו את השורות (1)-(6) כך שהפלט יהיה:

```

Hi: 10 of 3
Wow: 7 of 3 3*
20 of 3
Bye: 7 of 3
Hi: 30 of 3
Wow: 7 of 3 5*

```

ב. הכותרת של המחלקה Wish השתנתה לכוותרת:

Java

```
public class Wish extends Card
```

האם שינוי זה יגרום לשגיאה? אם כן - הסבירו את השגיאה, אם לא - רשמו מה יהיה הפלט של הפעולה main אחרי השינוי (השורות החסרות הן כפי שכתבתם בסעיף א).
הערה: אין שינויים נוספים במחלקה.

נוסח #C

לשאלה זו שני נוסחים: בשפת Java ובשפת #C.

לפותרים בשפת #C

נתונות המחלקות Card, Gift, Wish:

```

public class Card {
    public static int made = 0;
    private int code;
    public Card() {
        made++;
        code = made * 10;
    }
    public Card (int c) {
        code = c;
    }
    public override string ToString() {
        return code + " of " + made;
    }
}

-----

public class Gift : Card {
    private string text;
    public Gift() {
        text = "Hi";
    }
    public Gift (string t) : base (7) {
        text = t;
    }
    public override string ToString() {
        return text + ": " + base.ToString();
    }
}

-----

public class Wish : Gift {
    private int stars;
    public Wish (int s) : base ("Wow") {
        stars = s;
    }
    public override string ToString() {
        return base.ToString() + " " + stars + "*";
    }
}

```

א. נתונה המחלקה Test:

```

public class Test
{
    public static void Main (string[] args)
    {
        Card[] arr = new Card[6];
        arr[0] = ... // (1)
        arr[1] = ... // (2)
        arr[2] = ... // (3)
        arr[3] = ... // (4)
        arr[4] = ... // (5)
    }
}

```

```
arr[5] = ... // (6)
for (int i = 0; i < arr.Length; i++) {
    Console.WriteLine (arr[i]);
}
}
```

השלימו את השורות (1)-(6) כך שהפלט יהיה:

```
Hi: 10 of 3
Wow: 7 of 3 3*
20 of 3
Bye: 7 of 3
Hi: 30 of 3
Wow: 7 of 3 5*
```

ב. הכותרת של המחלקה Wish השתנתה לכותרת:

C#

```
public class Wish : Card
```

האם שינוי זה יגרום לשגיאה? אם כן - הסבירו את השגיאה, אם לא - רשמו מה יהיה הפלט של הפעולה Main אחרי השינוי (השורות החסרות הן כפי שכתבתם בסעיף א).
הערה: אין שינויים נוספים במחלקה.

12. (25 נקודות)

נוסח Java

חנות כלי הנגינה "צליל" מעוניינת למכור את מוצריה גם באופן מקוון (אונליין). בשלב הראשון החליט בעל החנות למכור באופן מקוון רק גיטרות ופסנתרים.
לשם כך נבנו המחלקות האלה:
- המחלקה **Instrument**, המייצגת כלי נגינה
- המחלקה **Guitar**, המייצגת גיטרה
- המחלקה **Piano**, המייצגת פסנתר

Java

```
public class Instrument {
    private int code;           // קוד המוצר
    private String brand;       // שם היצרן
    private int year;           // שנת הייצור
    private double price;       // מחיר
}

public class Guitar extends Instrument {
    private int strings;        // מספר המיתרים
}

public class Piano extends Instrument {
    private boolean electric;   // האם הפסנתר חשמלי
}
```

הניחו שיש פעולות get ו-set בעבור התכונות בכל המחלקות.

א. (1) המחלקה **Order** מייצגת הזמנה. למחלקה שלוש תכונות:

• customer - שם הלקוח

• items - מערך המכיל גם גיטרות וגם פסנתרים

• num - מספר הכלים הנוכחי בהזמנה

(1) כתבו את כותרת המחלקה **Order** ואת התכונות שלה.

(2) סרטטו תרשים הייררכייה המתאר את הקשר בין המחלקות: **Instrument**, **Guitar**, **Piano**, **Order**.

יש לסמן ירושה באמצעות החץ <--- והכלה באמצעות הסימן ◆---.

ב. (1) בשלב השני החליט בעל החנות למכור באופן מקוון גם תווים (חוברות תווים).

לפניכם המחלקה **SheetMusic** המייצגת חוברת תווים:

```

Java
public class SheetMusic {
    private int code;           // קוד המוצר
    private String title;       // שם היצירה
    private String composer;    // שם המלחין
    private double price;       // מחיר
}
```

הניחו שיש פעולות **get** ו-**set** בעבור תכונות המחלקה.

כדי שהמערך **items** במחלקה **Order** יוכל להכיל גם חוברות תווים, נעשו הפעולות האלה:

- הוגדרה המחלקה **Item** - פריט.

- טיפוס המשתנה **items** במחלקה **Order** שונה ל-**Item**.

- נכתבו במחלקות המתאימות (כולל במחלקה **SheetMusic**) יחסי הורשה חדשים בינן ובין המחלקה **Item**.

- חלק מן התכונות שהופיעו עד עכשיו בכמה מן המחלקות האחרות (כולל המחלקה **SheetMusic**) הועברו למחלקה **Item**, בהתאם לעקרונות תכנות מונחה עצמים (מעבר לתכונות אלו אין תכונות נוספות במחלקה **Item**).

(1) סרטטו תרשים הייררכייה מעודכן המתאר את הקשר בין המחלקות: **Instrument**, **Guitar**, **Piano**, **Order**, **Item**, **SheetMusic**.

יש לסמן ירושה באמצעות החץ <--- והכלה באמצעות הסימן ◆---.

(2) כתבו את כותרת המחלקה **Item** ותכונותיה.

(3) כתבו מחדש את המחלקות שחל בהן שינוי בעקבות ההוספה של המחלקה **Item**.

שימו לב: יש להתייחס גם ליחסי הורשה חדשים וגם לשינוי שחל בתכונות המחלקה.

ג. בשלב האחרון החליט בעל החנות להכריז על מבצע: ברכישה מקוונת כל הפריטים יימכרו בהנחה של 20%, ובעבור רכישת פסנתר יתווספו למחיר שאחרי ההנחה דמי הובלה של 150 שקלים. נתונה הפעולה הפנימית במחלקה **Item**:

```

Java
public double getFinalPrice()
```

זימון הפעולה מחזיר את המחיר שהלקוח ישלם בעבור הפריט (גיטרה/פסנתר/חוברת תווים) - בהתאם לכל פריט.

ממשו את הפעולה במחלקה Item , והוסיפו פעולה/ות במחלקה/ות המתאימה/ות כדי לבצע את הנדרש. ציינו בעבור כל פעולה שהוספתם באיזו מחלקה היא נוספה.
אין להשתמש בפעולה instanceof ובפעולות המחלקה Object **ואין** לשנות את תכונות המחלקות.

נוסח C#

חנות כלי הנגינה "צליל" מעוניינת למכור את מוצריה גם באופן מקוון (אונליין). בשלב הראשון החליט בעל החנות למכור באופן מקוון רק גיטרות ופסנתרים.
 לשם כך נבנו המחלקות האלה:
 - המחלקה **Instrument** , המייצגת כלי נגינה
 - המחלקה **Guitar** , המייצגת גיטרה
 - המחלקה **Piano** , המייצגת פסנתר

```
C#
public class Instrument
{
    private int code;           // קוד המוצר
    private string brand;       // שם היצרן
    private int year;           // שנת הייצור
    private double price;       // מחיר
}

public class Guitar : Instrument
{
    private int strings;        // מספר המיתרים
}

public class Piano : Instrument
{
    private bool electric;      // האם הפסנתר חשמלי
}
```

הניחו שיש פעולות Get ו-Set בעבור התכונות בכל המחלקות.

א. (1) המחלקה Order מייצגת הזמנה. למחלקה שלוש תכונות:

• customer - שם הלקוח

• items - מערך המכיל גם גיטרות וגם פסנתרים

• num - מספר הכלים הנוכחי בהזמנה

(1) כתבו את כותרת המחלקה Order ואת התכונות שלה.

(2) סרטטו תרשים הייררכייה המתאר את הקשר בין המחלקות: Instrument , Guitar , Piano , Order .

יש לסמן ירושה באמצעות החץ <--- והכלה באמצעות הסימן ◆--- .

ב. (1) בשלב השני החליט בעל החנות למכור באופן מקוון גם תווים (חוברות תווים).

לפניכם המחלקה **SheetMusic** המייצגת חוברת תווים:

```
C#
public class SheetMusic
{
    private int code;           // קוד המוצר
}
```

```
private string title;    // שם היצירה
private string composer; // שם המלחין
private double price;    // מחיר
}
```

הניחו שיש פעולות Get ו-Set בעבור תכונות המחלקה.

כדי שהמערך items במחלקה Order יוכל להכיל גם חוברות תווים, נעשו הפעולות האלה:

- הוגדרה המחלקה **Item** - פריט.

- טיפוס המשתנה items במחלקה Order שונה ל- **Item**.

- נכתבו במחלקות המתאימות (כולל במחלקה SheetMusic) יחסי הורשה חדשים בינן ובין המחלקה **Item**.

- חלק מן התכונות שהופיעו עד עכשיו בכמה מן המחלקות האחרות (כולל המחלקה SheetMusic) הועברו למחלקה **Item**, בהתאם לעקרונות תכנות מונחה עצמים (מעבר לתכונות אלו אין תכונות נוספות במחלקה **Item**).

(1) סרטטו תרשים הייררכיה מעודכן המתאר את הקשר בין המחלקות: Instrument, Guitar, Piano, SheetMusic, Item, Order.

יש לסמן ירושה באמצעות החץ <--- והכלה באמצעות הסימן ◆---.

(2) כתבו את כותרת המחלקה **Item** ותכונותיה.

(3) כתבו מחדש את המחלקות שחל בהן שינוי בעקבות ההוספה של המחלקה **Item**.

שימו לב: יש להתייחס גם ליחסי הורשה חדשים וגם לשינוי שחל בתכונות המחלקה.

ג. בשלב האחרון החליט בעל החנות להכריז על מבצע: ברכישה מקוונת כל הפריטים יימכרו בהנחה של 20%, ובעבור רכישת פסנתר יתווספו למחיר שאחרי ההנחה דמי הובלה של 150 שקלים. נתונה הפעולה הפנימית במחלקה **Item**:

C#

```
public virtual double GetFinalPrice()
```

זימון הפעולה מחזיר את המחיר שהלקוח ישלם בעבור הפריט (גיטרה/פסנתר/חוברת תווים) - בהתאם לכל פריט.

ממשו את הפעולה במחלקה **Item**, והוסיפו פעולה/ות במחלקה/ות המתאימה/ות כדי לבצע את הנדרש. ציינו בעבור כל פעולה שהוספתם באיזו מחלקה היא נוספה.

אין להשתמש בפעולות is ו- as ובפעולות המחלקה **object** **ואין** לשנות את תכונות המחלקות.

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