

סוג הבחינה: מבחן מתכונת
מבחן: מתכונת 8
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 ppp, המקבלת תור q ובו מספרים הגדולים מ-0, ומספר שלם k - הגדול מ-0.

Java

```
public static boolean ppp (Queue<Integer> q, int k)
{
    q.insert (-1);
    int max = 0;
    int c = 0;
    while (q.head() != -1)
    {
        int x = q.remove();
        if (x > max)
        {
            max = x;
            c++;
        }
        q.insert (x);
    }
    q.remove();
    return c >= k;
}
```

נתון תור q מטיפוס שלם:

ראש התור	סוף התור							
		4	2	7	7	5	9	3

(1) עקבו בעזרת טבלת המעקב שלפניכם אחר הפעולה ppp(q, 4), וכתבו מה הפעולה מחזירה.

התור q	x	x > max	max	c

(2) הסבירו מה הפעולה ppp עושה.

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

ב. (1) לפניכם הפעולה find, המקבלת תור - q ובו מספרים הגדולים מ-0, ומספר שלם - k הגדול מ-0. הפעולה find משתמשת בפעולה ppp מסעיף א.

```
Java
public static int find (Queue<Integer> q, int k)
{
    if (!ppp (q, k))
        return k - 1;
    return find (q, k + 1);
}
```

נתון התור q מסעיף א:

ראש התור	סוף התור							
		4	2	7	7	5	9	3

(1) עקבו אחר הפעולה find (q, 1), וכתבו מה הפעולה מחזירה (אין צורך לעקוב אחר הפעולה ppp).
המעקב יכלול בכל קריאה את הערכים של k, q ואת הערך המוחזר.
לפניכם הצעה לטבלת מעקב (אין חובה להשתמש בטבלה זו).

התור q שמתקבל בפעולה	הערך k שמתקבל בפעולה	ppp (q, k) == true	ערך מוחזר

(2) הסבירו מה הפעולה find עושה כאשר מזמנים אותה עם k = 1.
(3) מהי סיבוכיות זמן הריצה של הפעולה find כאשר מזמנים אותה עם k = 1? 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 Ppp, המקבלת תור - q ובו מספרים הגדולים מ-0, ומספר שלם - k הגדול מ-0.

```
C#
public static bool Ppp (Queue<int> q, int k)
{
    q.Insert (-1);
    int max = 0;
    int c = 0;
    while (q.Head() != -1)
    {
        int x = q.Remove();
        if (x > max)
        {
            max = x;
            c++;
        }
        q.Insert (x);
    }
}
```

```

}
q.Remove();
return c >= k;
}

```

נתון תור q מטיפוס שלם:

ראש התור	סוף התור							
		4	2	7	7	5	9	3

(1) עקבו בעזרת טבלת המעקב שלפניכם אחר הפעולה $Ppp(q, 4)$, וכתבו מה הפעולה מחזירה.

התור q	x	$x > \max$	max	c

(2) הסבירו מה הפעולה Ppp עושה.

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

ב. (1) לפניכם הפעולה $Find$, המקבלת תור q ובו מספרים הגדולים מ-0, ומספר שלם k הגדול מ-0. הפעולה $Find$ משתמשת בפעולה Ppp מסעיף א.

C#

```

public static int Find (Queue<int> q, int k)
{
    if (!Ppp (q, k))
        return k - 1;
    return Find (q, k + 1);
}

```

נתון התור q מסעיף א:

ראש התור	סוף התור							
		4	2	7	7	5	9	3

(1) עקבו אחר הפעולה $Find(q, 1)$, וכתבו מה הפעולה מחזירה (אין צורך לעקוב אחר הפעולה Ppp).

המעקב יכול לכלול בכל קריאה את הערכים של q, k ואת הערך המוחזר. לפניכם הצעה לטבלת מעקב (אין חובה להשתמש בטבלה זו).

התור q שמתקבל בפעולה	הערך k שמתקבל בפעולה	$Ppp(q, k) == true$	ערך מוחזר

(2) הסבירו מה הפעולה $Find$ עושה כאשר מזמנים אותה עם $k = 1$.

(3) מהי סיבוכיות זמן הריצה של הפעולה $Find$ כאשר מזמנים אותה עם $k = 1$? נמקו את תשובתכם.

2. (25 נקודות)

במשחק קפיצות משתמשים בשתי שרשרות של חוליות:

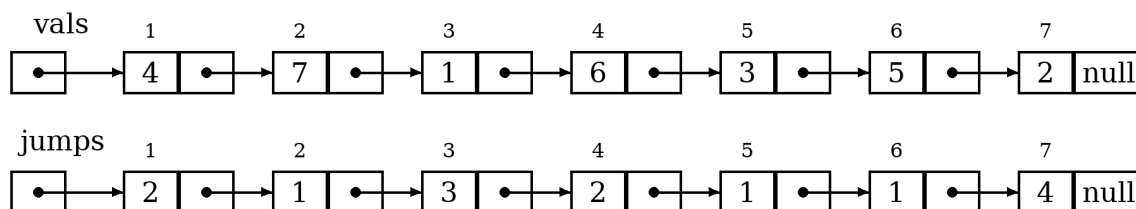
vals - הפניה לשרשרת חוליות של מספרים שלמים - הנקודות שבכל משבצת.

jumps - הפניה לשרשרת חוליות של מספרים שלמים **חיוביים** - אורך הקפיצה מכל משבצת.

הערה: כמות החוליות בשתי השרשרות **שווה**. החוליה במיקום k בשרשרת vals והחוליה במיקום k בשרשרת jumps מתארות את אותה משבצת.

מהלך המשחק: השחקן עומד על משבצת, **אוסף** את הנקודות שבה, ואז קופץ קדימה מספר משבצות השווה לאורך הקפיצה של המשבצת שעליה הוא עומד (קפיצה באורך 1 - למשבצת הבאה). כך הוא ממשיך עד שאסף את מספר האיסופים הנדרש, או עד שקפץ אל מעבר למשבצת האחרונה.

דוגמה: שתי השרשרות שלפניכם. שחקן שמתחיל במשבצת 1 אוסף 4 נקודות וקופץ 2 משבצות - למשבצת 3; שם הוא אוסף 1 וקופץ 3 - למשבצת 6; שם הוא אוסף 5 וקופץ 1 - למשבצת 7; שם הוא אוסף 2 וקופץ 4 - אל מעבר לסוף השרשרת.



א. ממשו את הפעולה שלפניכם:

Java - `public static int collect (Node<Integer> vals, Node<Integer> jumps, int steps)`

#C - `public static int Collect (Node<int> vals, Node<int> jumps, int steps)`

steps הוא מספר האיסופים שהשחקן מבצע.

הפעולה תחזיר את סכום הנקודות שאוסף שחקן שמתחיל **במשבצת הראשונה** (החוליה הראשונה בשתי השרשרות) ומבצע **steps** איסופים.

הניחו ש- **steps** גדול מ- 0. אין לשנות את השרשרות שהתקבלו.

הערה: אם השחקן קופץ אל מעבר למשבצת האחרונה לפני שביצע **steps** איסופים, הפעולה תחזיר את סכום הנקודות שאסף עד אז.

דוגמה: בעבור שתי השרשרות בדוגמה שלעיל ו- **steps = 1**, הפעולה תחזיר 4.

דוגמה נוספת: בעבור אותן שתי שרשרות ו- **steps = 3**, הפעולה תחזיר 10 ($10 = 5 + 1 + 4$).

דוגמה נוספת: בעבור אותן שתי שרשרות ו- **steps = 10**, הפעולה תחזיר 12 ($12 = 2 + 5 + 1 + 4$), כי אחרי 4 איסופים השחקן קופץ אל מעבר לסוף השרשרת.

ב. ממשו את הפעולה שלפניכם:

Java - `public static boolean reach (Node<Integer> vals, Node<Integer> jumps, int steps, int target)`

#C - `public static bool Reach (Node<int> vals, Node<int> jumps, int steps, int target)`

הפעולה תחזיר true אם קיימת משבצת **כלשהי** (חוליה במיקום כלשהו בשרשרת vals והחוליה **במיקום זה** בשרשרת jumps), כך ששחקן שמתחיל **ממנה** ומבצע steps איסופים (כמתואר בסעיף א) אוסף בדיוק target נקודות. אחרת הפעולה תחזיר false . אפשר להשתמש בפעולה שכתבתם בסעיף א. הניחו ש- steps גדול מ- 0 . אין לשנות את השרשרות שהתקבלו.

דוגמה: בעבור שתי השרשרות בדוגמה שלעיל, $steps = 2$ ו- $target = 11$, הפעולה תחזיר true , כי שחקן שמתחיל **במשבצת 4** אוסף 6 , קופץ 2 משבצות - למשבצת 6 , ואוסף 5 : $11 = 5 + 6$.
דוגמה נוספת: בעבור אותן שתי שרשרות, $steps = 2$ ו- $target = 9$, הפעולה תחזיר false .

3. (25 נקודות)

בשאלה זו אפשר להשתמש בפעולה החיצונית sumTree / SumTree שלהלן בלי לממש אותה.

דוגמאות	תיאור הפעולה	כותרת הפעולה
- בעבור עץ שבו צומת יחיד עם הערך 6, הפעולה תחזיר 6. -
 בעבור tree = null, הפעולה תחזיר 0.	הפעולה מקבלת הפניה לעץ בינארי של מספרים שלמים - tree, ומחזירה את סכום הערכים בכל הצמתים של העץ. אם tree הוא null, הפעולה מחזירה 0.	בשפת Java:
public static int sumTree
C#:
public static int SumTree
(BinNode<int> tree)

צאצאים של צומת בעץ בינארי הם כל הצמתים שבתת-העץ השמאלי שלו ובתת-העץ הימני שלו (הבנים שלו, הבנים של הבנים שלו, וכן הלאה).

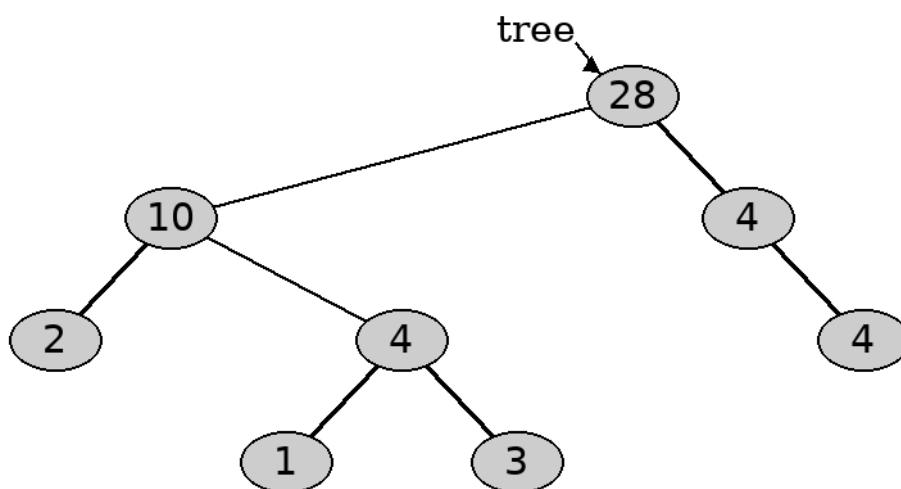
עץ בינארי של מספרים שלמים הוא "**עץ סכומים**" אם הערך של כל צומת שאינו עלה שווה לסכום הערכים של כל הצאצאים שלו. (על העלים אין שום דרישה). ממשו את הפעולה החיצונית שלהלן:

Java - public static boolean isSumTree (BinNode<Integer> tree)

C# - public static bool IsSumTree (BinNode<int> tree)

הפעולה מקבלת הפניה לעץ בינארי של מספרים שלמים - tree שאינו null. הפעולה תחזיר true אם tree הוא "עץ סכומים", אחרת הפעולה תחזיר false. דוגמה:

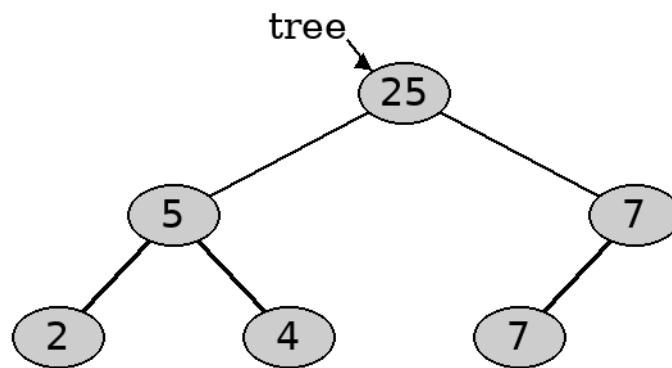
בעבור העץ הנתון הפעולה תחזיר true.



הסבר: $28 = 10 + 4 + 3 + 1 + 4 + 2 + 4$ (כל הצאצאים של השורש), $10 = 2 + 4 + 3 + 1$, וגם בצומת 4 שמימין לשורש: $4 = 4$.

דוגמה:

בעבור העץ הנתון הפעולה תחזיר false.



הסבר: אף על פי שבשורש מתקיים $25 = 5 + 2 + 4 + 7 + 7$, בצומת 5 לא מתקיים התנאי: סכום הצאצאים שלו הוא $6 = 4 + 2$.

הערה: עץ שיש בו צומת יחיד הוא "עץ סכומים".
ממשו את הפעולה isSumTree / IsSumTree המתוארת לעיל.

פרק שני (50 נקודות) - אלגוריתמים

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

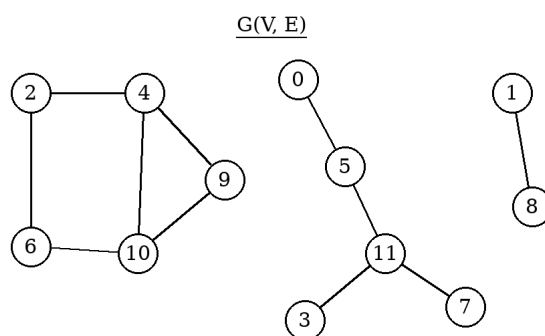
4. (25 נקודות)

נתון גרף לא מכוון ולא קשיר $G(V, E)$ שבו n קודקודים, מ- v_0 עד v_{n-1} . הגרף מתאר מפה של כפרים: כל קודקוד הוא כפר, וכל קשת היא כביש המחבר שני כפרים.

א. כתבו אלגוריתם שמקבל שני קודקודים שונים בגרף v_s ו- v_t , ומחזיר מסלול מ- v_s אל v_t שמספר הקשתות בו הוא הקטן ביותר (את רשימת הקודקודים שבמסלול, לפי הסדר מ- v_s אל v_t). אם אין מסלול בין שני הקודקודים, האלגוריתם יחזיר רשימה ריקה.

הערה: יש לכתוב אלגוריתם יעיל שאינו עובר על כל המסלולים האפשריים.

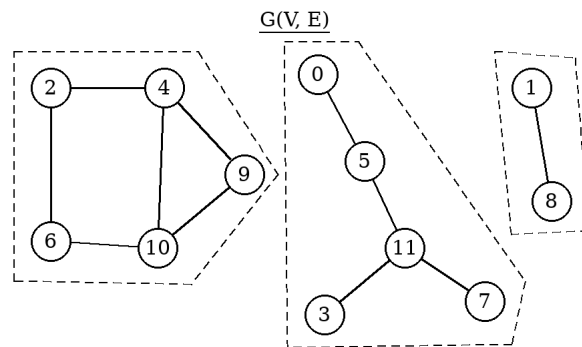
דוגמה: עבור הגרף שלפניכם והקודקודים 2 ו-9, האלגוריתם יחזיר את המסלול 2, 4, 9.



הסבר: גם 2, 6, 10, 9 הוא מסלול מקודקוד 2 אל קודקוד 9, אבל יש בו 3 קשתות, ובמסלול 2, 4, 9 יש רק 2 קשתות. עבור הקודקודים 0 ו-9 האלגוריתם יחזיר רשימה ריקה.

ב. "רכיב קשירות" בגרף לא מכוון $G(V, E)$ הוא קבוצת קודקודים שבה בין כל שני קודקודים יש מסלול, ואין שום קשת היוצאת מקודקוד בקבוצה לקודקוד שאינו בקבוצה. קודקוד שאין קשת בינו ובין שום קודקוד אחר יהיה בקבוצה משלו.

דוגמה: עבור הגרף שבדוגמה לעיל, שלושת רכיבי הקשירות מסומנים בקו מקווקו.

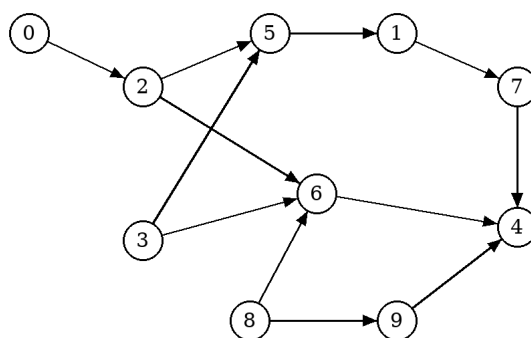


כתבו אלגוריתם המוצא ומחזיר את רכיב הקשירות שבו מספר **הקשתות** הוא הגדול ביותר. למשל עבור הדוגמה שלעיל, האלגוריתם יחזיר את הקודקודים 2, 4, 6, 9, 10 (ברכיב זה יש 6 קשתות, ברכיב {0, 3, 5, 7, 11} יש 4 קשתות, וברכיב {1, 8} יש קשת אחת). הניחו שיש רק רכיב קשירות **אחד** שבו מספר הקשתות הוא הגדול ביותר. **הערה:** יש לכתוב אלגוריתם יעיל שאינו עובר על כל המסלולים האפשריים.

5. (25 נקודות)

נתון גרף **מכוון ללא מעגלים** $G(V, E)$ שבו n קודקודים, מ- v_0 עד v_{n-1} . הגרף מייצג משימות בפרויקט: קשת מ- v_i אל v_k פירושה שאת המשימה v_i יש לסיים לפני שמתחילים את המשימה v_k . הגרף מיוצג ברשימות סמיכויות: לכל קודקוד יש רשימה של הקודקודים שהקשתות היוצאות ממנו מגיעות אליהם.

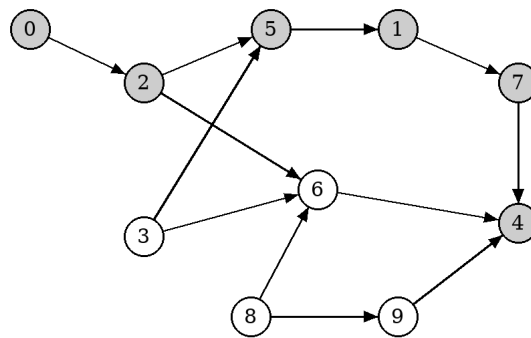
א. כתבו אלגוריתם המוצא ומחזיר את כל הקודקודים שיש מהם מסלול **אל** הקודקוד v_j (כלומר, את כל המשימות שיש לסיים לפני שמתחילים את המשימה v_j). **הערה:** יש לכתוב אלגוריתם יעיל שאינו עובר על כל המסלולים האפשריים. **דוגמה:** עבור הגרף שלפניכם, וקודקוד 6, האלגוריתם יחזיר את הקודקודים 0, 2, 3, 8.



הסבר: יש קשת ישירה אל קודקוד 6 מכל אחד מן הקודקודים 2, 3, 8, ויש מסלול $0 \rightarrow 2 \rightarrow 6$. מאף קודקוד אחר אין מסלול אל 6.

ב. "שרשרת משימות" היא מסלול מכוון בגרף. **אורך** השרשרת הוא מספר הקשתות שבה. כתבו אלגוריתם המוצא ומחזיר את השרשרת **הארוכה ביותר** בגרף $G(V, E)$ - את הקודקודים שבה, לפי הסדר.

דוגמה: עבור הגרף שבדוגמה לעיל, האלגוריתם יחזיר את הקודקודים 0, 2, 5, 1, 7, 4 (שרשרת באורך 5; הקודקודים שלה צבועים באפור):



הניחו שיש רק שרשרת אחת שהיא הארוכה ביותר.
הערה: יש לכתוב אלגוריתם יעיל שאינו עובר על כל המסלולים האפשריים.

6. (25 נקודות)

גרף לא מכוון נקרא "גרף-עץ-שבור" אם יש בו שני קודקודים שאין ביניהם קשת, וכאשר מוסיפים קשת ביניהם מקבלים עץ.

א. סרטטו גרף שיש בו 6 קודקודים והוא "גרף-עץ-שבור".

ב. (1) נתון "גרף-עץ-שבור" ובו n קודקודים.

(1) האם אפשר לדעת את מספר הקשתות בגרף זה? נמקו את תשובתכם. אם עניתם שאפשר, ציינו את מספר הקשתות.

(2) האם אפשר לדעת את מספר רכיבי הקשירות בגרף זה? נמקו את תשובתכם. אם עניתם שאפשר, ציינו את מספר רכיבי הקשירות.

ג. (1) נתון גרף לא מכוון G ובו n קודקודים ($n \geq 2$), המיוצג על ידי רשימת סמיכויות.

(1) כתבו אלגוריתם יעיל המחזיר "אמת" אם הגרף הוא "גרף-עץ-שבור" ואחרת הוא מחזיר "שקר".

(2) מהי סיבוכיות זמן הריצה של האלגוריתם שכתבתם? נמקו את תשובתכם.

פרק שני (50 נקודות) - מודלים חישוביים

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

7. (25 נקודות)

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

א. (1) לפניכם שלוש שפות מעל הא"ב $\{a, b\}$. הסימון $n_a(w)$ הוא מספר המופעים של a במילה w , הסימון $n_b(w)$ הוא מספר המופעים של b במילה w , והסימון $|w|$ הוא אורך המילה w .

$$L_1 = \{ w \mid n_a(w) = n_b(w) \}$$

$$\{ w \mid n_a(w) - n_b(w) = 3 \text{ ללא שארית} \} = L_2$$

$$L_3 = \{ w \mid |w| \geq 12 \}$$

לפניכם ארבעה סעיפים (1)-(4). ענו על כולם (בנימוק מספיק הסבר מילולי, אין צורך באוטומט).

(1) בנוגע לכל אחת מן השפות L_1-L_3 , כתבו אם היא רגולרית או לא רגולרית. נמקו את תשובתכם.

(2) האם השפה $L_1 \cap L_3$ רגולרית? נמקו את תשובתכם.

(3) האם השפה $L_1 \cap \bar{L}_3$ רגולרית? נמקו את תשובתכם.

(4) האם השפה $L_2 \cap \bar{L}_1$ רגולרית? נמקו את תשובתכם.

ב. (1) נתונה השפה L מעל הא"ב $\{a, b, \#\}$:

$$L = \{ u \# v \mid u, v \in \{a, b\}^* \text{ וגם } u \text{ מסתיימת באות } a \text{ או } (v \% 3 = 0) \}$$

הסבר: כל מילה בשפה L מורכבת משני חלקים לא ריקים u ו- v , שבהם רק האותיות a ו- b , ובין שני החלקים סימן $\#$ אחד. כמו כן, החלק u מסתיים באות a , או שאורך החלק v מתחלק ב-3.

דוגמה למילה בשפה L היא $bba\#ab$, כי החלק $u = bba$ מסתיים באות a .

דוגמה נוספת למילה בשפה L היא $ab\#bab$, כי אורך החלק $v = bab$ הוא 3, ומתחלק ב-3.

דוגמה למילה שאינה בשפה L היא $abb\#ab$. החלק $u = abb$ אינו מסתיים באות a , ואורך החלק $v = ab$ הוא 2 - לא מתחלק ב-3.

(1) לפניכם חמש מילים: $ba\#b$, $ab\#aab$, $ab\#ab$, $a\#$, $aa\#b\#a$.

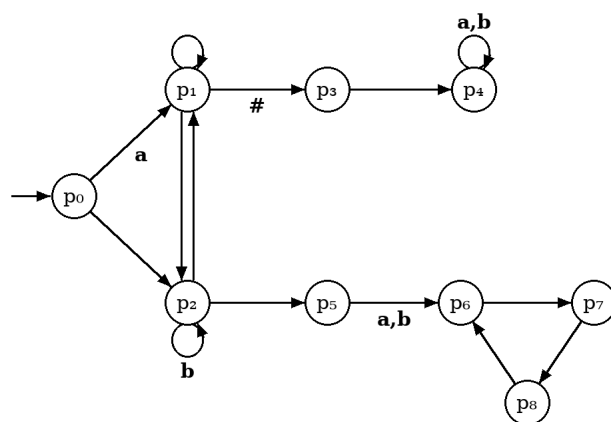
העתיקו כל אחת מן המילים למחברתכם, וקבעו אם היא שייכת לשפה L או לא שייכת לשפה L . נמקו את קביעותיכם.

(2) נתון לפניכם אוטומט סופי דטרמיניסטי שאינו מלא המקבל את השפה L . באוטומט קיימים

כל המצבים וכל המעברים הנדרשים, אך בכמה מן המעברים חסרים סימני הקלט (התווים במעברים), והמצבים המקבלים באוטומט אינם מסומנים כלל. מצב ההתחלה הוא p_0 . שימו לב: במעבר אחד יכולים להופיע כמה סימני קלט (למשל a, b).

העתיקו את האוטומט למחברתכם, הוסיפו את סימני הקלט החסרים, וסמנו את המצבים המקבלים.

הערה: האוטומט צריך להישאר סופי דטרמיניסטי שאינו מלא. אין להוסיף בו מצבים או מעברים, ואין לשנות את סימני הקלט המופיעים בו. **תשובה שיהיה בה שינוי באוטומט הנתון לא תזכה בנקודות.**



8. (25 נקודות)

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

א. (1) לפניכם שש שפות מעל הא"ב $\{x, y\}$:

Σ^* - מציין את שפת כל המילים מעל הא"ב $\{x, y\}$.

L^R - מציין את השפה שמילותיה הן היפוכי המילים של L .

$L_4 = \{xy, yxy\}$	$L_1 = \Sigma$
$L_5 = \{\varepsilon, x, xy\}$	$L_2 = \{\varepsilon\}$
$L_6 = \{y, yx\}$	$L_3 = \emptyset$

כתבו את השפה המתקבלת מכל אחת מחמש הפעולות שלהלן:

$$L_4 \cap L_5 \quad (1)$$

$$L_6 \cdot L_6 \quad (2)$$

$$L_1 \cdot L_2 \quad (3)$$

$$L_3^R \quad (4)$$

$$L_5 \cdot L_6 \quad (5)$$

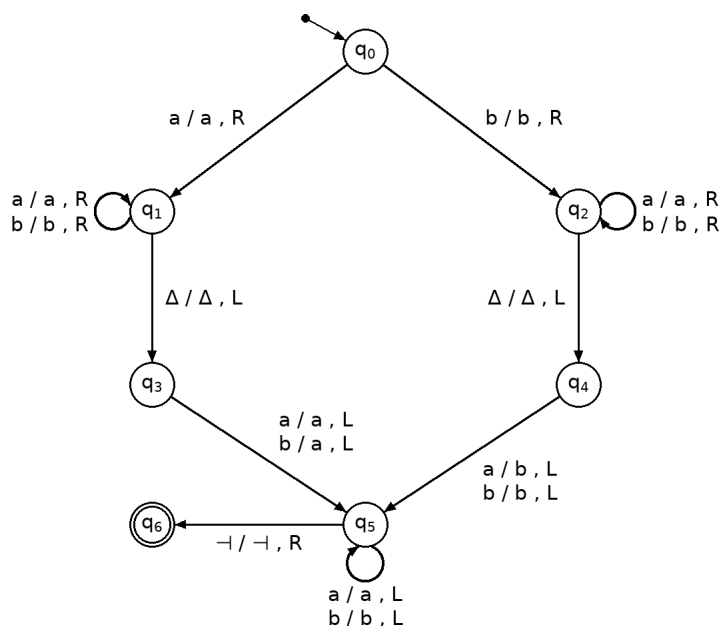
ב. נתונה השפה L מעל הא"ב $\{a, b, c\}$:

$$L = \{a^n b^m c^m a^n \mid n \geq 0, m \geq 1\}$$

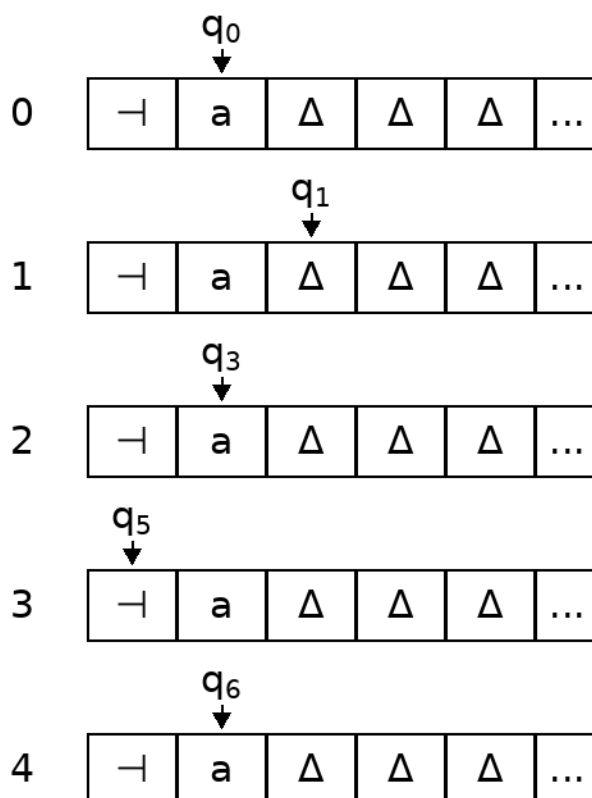
בנו אוטומט מחסנית דטרמיניסטי המקבל את השפה L .

9. (25 נקודות)

לפניכם מכונת טיורינג המקבלת מילים מעל הא"ב $\{a, b\}$ (מקרא: R - ימין, L - שמאל).



בסעיף א שלפניכם תתבקשו לבצע מעקב אחר פעולת המכונה. במעקב יש להראות בכל שלב את הסרט, מעל איזה תא ממוקם הראש הקורא, ובאיזה מצב q_0 - q_6 נמצאים. להלן דוגמה לאופן שבו יש לבצע מעקב בעבור המכונה הנתונה והמילה a.



- א. (1) עקבו אחרי פעולת המכונה בעבור המילה ba. יש להציג את המעקב.
 (2) עקבו אחרי פעולת המכונה בעבור המילה abb. יש להציג את המעקב.
 (3) הציגו את הסרט המתקבל בגמר הפעולה בעבור המילה babaa. אין צורך להציג את המעקב.

- ב. (1) כמה מילים באורך 4 מעל הא"ב {a, b} אינן משתנות לאחר הרצת המכונה עליהן? נמקו את תשובתכם.
- (2) כתבו בקצרה מה המכונה עושה.

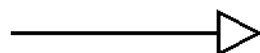
תכנות מונחה עצמים

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

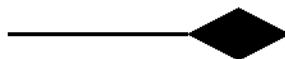
10. (25 נקודות)

נוסח Java

- בפנסיון לחיות מחמד "זנב שמח" פותחה מערכת ממוחשבת שבה המחלקות האלה:
- Stay - שהייה, Pet - חיית מחמד, Cat - חתול, Dog - כלב, Rabbit - ארנב.
- להלן פירוט תכונות המחלקות:
- למחלקה שהייה (Stay) שלוש תכונות: owner - שם הבעלים (מחרוזת), nights - מספר הלילות בפנסיון (מטיפוס שלם), meals - מספר הארוחות המיוחדות שהוזמנו (מטיפוס שלם).
 - למחלקה חיית מחמד (Pet) שתי תכונות: מספר השבב - chip (מחרוזת), שהייה - stay (Stay).
 - למחלקה חתול (Cat) שלוש תכונות: מספר השבב - chip (מחרוזת), שהייה - stay (Stay), חתול בית - indoor (בוליאני). אם החתול אינו יוצא מהבית התכונה היא אמת ואם לא, היא שקר).
 - למחלקה כלב (Dog) שלוש תכונות: מספר השבב - chip (מחרוזת), שהייה - stay (Stay), משקל הכלב בקילוגרמים - weight (מטיפוס שלם).
 - למחלקה ארנב (Rabbit) שלוש תכונות: מספר השבב - chip (מחרוזת), שהייה - stay (Stay), גיל הארנב בשנים - age (מטיפוס שלם).
- א. (1) סרטטו תרשים הייררכייה המתאר את הקשר בין המחלקות של המערכת הממוחשבת. יש לסמן ירושה באמצעות החץ



והכלה באמצעות הסימן



- (2) כתבו את כותרות המחלקות ואת התכונות שלהן. הניחו שהפעולות get ו-set קיימות בכל התכונות של המחלקות, ואין צורך לממש אותן.

- ב. (1) נתונה הפעולה הבונה של המחלקה Stay :

Java

```
public Stay (String owner, int nights, int meals)
```

אין צורך לממש את הפעולה.

- (1) לפניכם כותרת הפעולה הבונה של המחלקה Pet . הפעולה מקבלת שם בעלים, מספר לילות, מספר ארוחות מיוחדות ומספר שבב.

Java

```
public Pet (String owner, int nights, int meals, String chip)
```

ממשו את הפעולה הבונה.

- (2) לפניכם כותרת הפעולה הבונה של המחלקה Dog . הפעולה מקבלת שם בעלים, מספר לילות, מספר ארוחות מיוחדות, מספר שבב ומשקל.

Java

```
public Dog (String owner, int nights, int meals, String chip, int weight)
```

ממשו את הפעולה הבונה.

- ג. (1) העלות הבסיסית של שהיית חיות מחמד - Pet בפנסיון היא 90 שקלים ללילה ו- 12 שקלים לכל ארוחה מיוחדת.
 העלות של שהיית חתול (Cat) היא לפי העלות הבסיסית.
 העלות של שהיית כלב (Dog) שמשקלו יותר מ- 20 קילוגרמים היא העלות הבסיסית ונוסף על כך 15 שקלים לכל לילה.
 העלות של שהיית כלב שמשקלו 20 קילוגרמים או פחות היא לפי העלות הבסיסית.
 העלות של שהיית ארנב (Rabbit) היא 75% מן העלות הבסיסית.
 ג. הפעולה cost מחזירה מספר ממשי השווה לסכום שהבעלים נדרשים לשלם בעבור כל אחד מסוגי החיות שבפנסיון
 (בהתאם לעצם שזימן את הפעולה).
 (1) כתבו את הפעולה cost במחלקה Pet (כאמור לעיל, הסכום לתשלום הוא לפי העלות הבסיסית).
 (2) הוסיפו את הפעולה cost במחלקה/ות האחרות כדי לבצע את הנדרש (רק במחלקות שיש בהן צורך), לפי העקרונות של תכנות מונחה עצמים.
הערה: אין להשתמש בפעולה instanceof בסעיף זה ובפעולות של המחלקה Object ואין לשנות את תכונות המחלקות.
 פתרון הכולל שימושים כאלה לא יזוכה בנקודות.

נוסח #C

בפנסיון לחיות מחמד "זנב שמח" פותחה מערכת ממוחשבת שבה המחלקות האלה:

Stay - שהייה, **Pet** - חיית מחמד, **Cat** - חתול, **Dog** - כלב, **Rabbit** - ארנב.

להלן פירוט תכונות המחלקות:

. למחלקה שהייה (Stay) שלוש תכונות: owner - שם הבעלים (מחרוזת), nights - מספר הלילות בפנסיון (מטיפוס שלם),

meals - מספר הארוחות המיוחדות שהוזמנו (מטיפוס שלם).

. למחלקה חיית מחמד (Pet) שתי תכונות: מספר השבב - chip (מחרוזת), שהייה - stay (Stay).

. למחלקה חתול (Cat) שלוש תכונות: מספר השבב - chip (מחרוזת), שהייה - stay (Stay),

חתול בית - indoor (בוליאני). אם החתול אינו יוצא מהבית התכונה היא אמת ואם לא, היא שקר).

. למחלקה כלב (Dog) שלוש תכונות: מספר השבב - chip (מחרוזת), שהייה - stay (Stay),

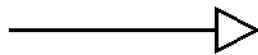
משקל הכלב בקילוגרמים - weight (מטיפוס שלם).

. למחלקה ארנב (Rabbit) שלוש תכונות: מספר השבב - chip (מחרוזת), שהייה - stay (Stay),

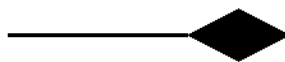
גיל הארנב בשנים - age (מטיפוס שלם).

א. (1) סרטטו תרשים הייררכייה המתאר את הקשר בין המחלקות של המערכת הממוחשבת.

יש לסמן ירושה באמצעות החץ



והכלה באמצעות הסימן



(2) כתבו את כותרות המחלקות ואת התכונות שלהן. הניחו שהפעולות Get ו-Set קיימות בכל

התכונות של המחלקות,

ואין צורך לממש אותן.

ב. (1) נתונה הפעולה הבונה של המחלקה Stay :

C#

```
public Stay (string owner, int nights, int meals)
```

אין צורך לממש את הפעולה.

(1) לפניכם כותרת הפעולה הבונה של המחלקה Pet. הפעולה מקבלת שם בעלים, מספר

לילות, מספר ארוחות

מיוחדות ומספר שבב.

C#

```
public Pet (string owner, int nights, int meals, string chip)
```

ממשו את הפעולה הבונה.

(2) לפניכם כותרת הפעולה הבונה של המחלקה Dog. הפעולה מקבלת שם בעלים, מספר לילות,

מספר ארוחות מיוחדות,

מספר שבב ומשקל.

C#

```
public Dog (string owner, int nights, int meals, string chip, int weight)
```

ממשו את הפעולה הבונה.

ג. (1) העלות הבסיסית של שהיית חיות מחמד - Pet בפנסיון היא 90 שקלים ללילה ו-12 שקלים לכל ארוחה מיוחדת.

העלות של שהיית חתול (Cat) היא לפי העלות הבסיסית.

העלות של שהיית כלב (Dog) שמשקלו יותר מ-20 קילוגרמים היא העלות הבסיסית ונוסף על כך 15 שקלים לכל לילה.

העלות של שהיית כלב שמשקלו 20 קילוגרמים או פחות היא לפי העלות הבסיסית.

העלות של שהיית ארנב (Rabbit) היא 75% מן העלות הבסיסית.

ג. הפעולה Cost מחזירה מספר ממשי השווה לסכום שהבעלים נדרשים לשלם בעבור כל אחד מסוגי החיות שבפנסיון

(בהתאם לעצם שזימן את הפעולה).

(1) כתבו את הפעולה Cost במחלקה Pet (כאמור לעיל, הסכום לתשלום הוא לפי העלות הבסיסית).

(2) הוסיפו את הפעולה Cost במחלקה/ות האחרות כדי לבצע את הנדרש (רק במחלקות שיש בהן צורך), לפי

העקרונות של תכנות מונחה עצמים.

הערה: אין להשתמש בפעולות is ו- as בסעיף זה ובפעולות של המחלקה Object ואין לשנות את תכונות המחלקות.

פתרון הכולל שימושים כאלה לא יזוכה בנקודות.

11. (25 נקודות)

נוסח Java

לפניכם שלושה סעיפים א-ג שאין קשר ביניהם. **ענו על שלושת הסעיפים.**

א. לפניכם המחלקה Box (קופסה) ללא מימוש הפעולות הבונות, והמחלקה GiftBox (קופסת מתנה) ללא מימוש הפעולות הבונות ופעולת toString.

```
public class Box {
    private static int made = 0;
    protected int size;
    public Box () {...}
    public Box (int size) {...}
    public String toString() {return "made=" + made + ", size=" + size;}
}

-----

public class GiftBox extends Box {
    private String color;
    public GiftBox (String color) {...}
    public GiftBox (int size, String color) {...}
    public String toString() {...}
}
```

לפניכם קטע קוד והפלט שהתקבל בעקבות הרצת קטע הקוד:

הקוד

Java

```
Box b1 = new Box ();
System.out.println (b1);           //1
Box b2 = new Box (8);
System.out.println (b2);           //2
Box b3 = new GiftBox ("red");
System.out.println (b3);           //3
Box b4 = new GiftBox (6, "green");
System.out.println (b4);           //4
Box b5 = new GiftBox ("yellow");
System.out.println (b5);           //5
```

הפלט

```
made=0, size=1           //1
made=1, size=8           //2
made=2, size=3, color=red //3
made=2, size=6, color=green //4
made=3, size=6, color=yellow //5
```

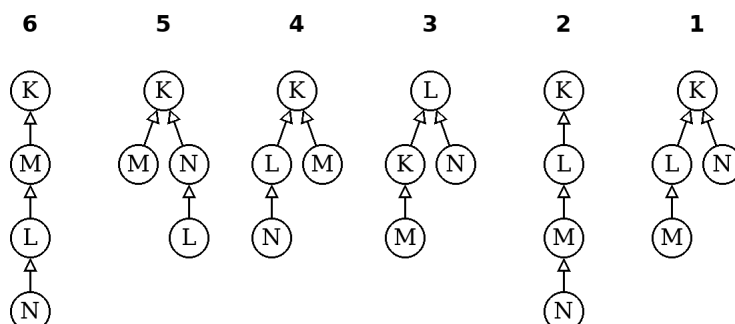
ממשו את הפעולות הבונות במחלקה Box ובמחלקה GiftBox, ואת הפעולה toString במחלקה GiftBox, בהתאם לעקרונות תכנות מונחה עצמים, כך שיתקבל הפלט הנתון. אין להוסיף פעולות נוספות למחלקות. תשובה הכוללת פעולות נוספות במחלקות, לא תזוכה בנקודות.

ב. (1) נתונות המחלקות K, L, M, N והפעולה main במחלקה Tester, **שרצה באופן תקין**.

Java

```
public class Tester {
    public static void main(String[] args) {
        L l1 = new N();
        K k1 = new M();
        K k2 = new L();
    }
}
```

לפניכם שישה תרשימי הייררכייה 1-6 (החץ מסמן יחס ירושה). בנוגע לכל אחד מהם, ציינו אם ייתכן או לא ייתכן שהוא מתאר את יחס ההייררכייה בין המחלקות K, L, M, N. אם ציינתם שלא ייתכן, נמקו מדוע.



(2) בסוף הפעולה main נוספו שורות הקוד שלהלן, **שרצות באופן תקין**:

Java

```
K k3 = new M();
L l2 = (L)k3;
```

בעקבות זאת, האם תשתנה תשובתכם בתת-סעיף ב(1)? נמקו את תשובתכם.

ג. נתונה המחלקה **Lamp** (מנורה), היורשת מהמחלקה **Device** (מכשיר):

Java

```
public class Lamp extends Device {
    private int watts;
    public Lamp (String code, int watts) {
        super (code, watts * 2);
        this.watts = watts;
    }
    public Lamp (int watts) {this.watts = watts; }
    public String info () {return super.info() + "/" + this.watts; }
    public double cost (int hours) {return hours * this.watts * Device.rate; }
    public boolean isOld () {return this.isOld (2020) || this.broken; }
}
```

כתבו במחלקה **Device** את **תכונות** המחלקה (כולל הרשאות הגישה) ואת **הכותרות** של הפעולות הנדרשות כדי שהמחלקה **Lamp** תהיה תקינה, בהתאם לעקרונות תכנות מונחה עצמים.

אין צורך לממש את הפעולות שאת הכותרות שלהן הוספתם במחלקה **Device. אין לשנות את הקוד של המחלקה **Lamp**.**

נוסח #C

לפניכם שלושה סעיפים א-ג שאין קשר ביניהם. **ענו על שלושת הסעיפים.**

א. לפניכם המחלקה **Box** (קופסה) ללא מימוש הפעולות הבונות, והמחלקה **GiftBox** (קופסת מתנה) ללא מימוש הפעולות הבונות ופעולת **ToString**.

C#

```
public class Box {
    private static int made = 0;
    protected int size;
    public Box () {...}
    public Box (int size) {...}
    public override string ToString() {return "made=" + made + ", size=" + size;}
}

-----

public class GiftBox : Box {
    private string color;
    public GiftBox (string color) ...{...}
    public GiftBox (int size, string color) ...{...}
    public override string ToString() {...}
}
```

לפניכם קטע קוד והפלט שהתקבל בעקבות הרצת קטע הקוד:

הקוד

C#

```
Box b1 = new Box ();
Console.WriteLine (b1);           //1
Box b2 = new Box (8);
Console.WriteLine (b2);           //2
```

```
Box b3 = new GiftBox ("red");
Console.WriteLine (b3);           //3
Box b4 = new GiftBox (6, "green");
Console.WriteLine (b4);           //4
Box b5 = new GiftBox ("yellow");
Console.WriteLine (b5);           //5
```

הפלט

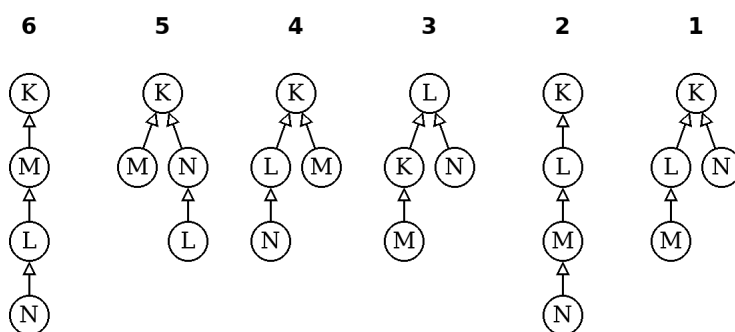
```
made=0, size=1           //1
made=1, size=8           //2
made=2, size=3, color=red //3
made=2, size=6, color=green //4
made=3, size=6, color=yellow //5
```

ממשו את הפעולות הבונות במחלקה Box ובמחלקה GiftBox, ואת הפעולה ToString במחלקה GiftBox, בהתאם לעקרונות תכנות מונחה עצמים, כך שיתקבל הפלט הנתון. אין להוסיף פעולות נוספות למחלקות. תשובה הכוללת פעולות נוספות במחלקות, לא תזוכה בנקודות.

ב. (1) נתונות המחלקות K, L, M, N והפעולה Main במחלקה Tester, **שרצה באופן תקין**.

```
C#
public class Tester {
    public static void Main (string[] args) {
        L l1 = new N();
        K k1 = new M();
        K k2 = new L();
    }
}
```

לפניכם שישה תרשימי הייררכייה 1-6 (החץ מסמן יחס ירודה). בנוגע **לכל אחד** מהם, ציינו אם ייתכן או לא ייתכן שהוא מתאר את יחס ההיררכייה בין המחלקות K, L, M, N. אם ציינתם שלא ייתכן, נמקו מדוע.



(2) בסוף הפעולה Main נוספו שורות הקוד שלהלן, **שרצות באופן תקין**:

```
C#
K k3 = new M();
L l2 = (L)k3;
```

בעקבות זאת, האם תשתנה תשובתכם בתת-סעיף ב(1)? נמקו את תשובתכם.

ג. נתונה המחלקה **Lamp** (מנורה), היורשת מהמחלקה **Device** (מכשיר):

C#

```
public class Lamp : Device {
    private int watts;
    public Lamp (string code, int watts) : base (code, watts * 2) {
        this.watts = watts;
    }
    public Lamp (int watts) {this.watts = watts;}
    public override string Info () {return base.Info() + "/" + this.watts;}
    public double Cost (int hours) {return hours * this.watts * Device.rate;}
    public bool IsOld () {return this.IsOld (2020) || this.broken;}
}
```

כתבו במחלקה Device את **תכונות** המחלקה (כולל הרשאות הגישה) ואת **הכותרות** של הפעולות הנדרשות כדי שהמחלקה Lamp תהיה תקינה, בהתאם לעקרונות תכנות מונחה עצמים.
אין צורך לממש את הפעולות שאת הכותרות שלהן הוספתם במחלקה Device . אין לשנות את הקוד של המחלקה Lamp.

12. (25 נקודות)

נוסח Java

לפניכם המחלקות Robot (רובוט) ו־Cleaner (רובוט ניקוי):

Java

```

public class Robot
{
    private static int made = 0;
    protected String name;
    protected int energy;

    public Robot (String name) {
        this.name = name;
        this.energy = 100;
        made++;
        System.out.println ("Robot " + name);
    }
    public static int getMade() {
        return made;
    }
    public int getEnergy() {
        return energy;
    }
    public void work (int hours) {
        this.energy -= 10 * hours;
        System.out.println (this.name + " worked, energy = " + this.energy);
    }
    public String describe() {
        return this.name + "(" + this.energy + ")";
    }
}

public class Cleaner extends Robot
{
    private int rooms;

    public Cleaner (String name) {
        super (name);
        this.rooms = 0;
        System.out.println ("Cleaner");
    }
    public Cleaner (String name, int energy) {
        super (name);
        this.energy = energy;
        this.rooms = 0;
        System.out.println ("Cleaner " + energy);
    }
    public void work (int hours) {
        super.work (hours);
        this.rooms += 2 * hours;
    }
    public void work (int hours, int extra) {
        work (hours + extra);
        System.out.println ("rooms = " + this.rooms);
    }
    public String describe() {
        return super.describe() + " rooms: " + this.rooms;
    }
}

```

נתונה המחלקה Tester :

Java

```
public class Tester
{
    public static void main(String[] args)
    {
        Robot r1 = new Robot ("Alfa");
        Robot r2 = new Cleaner ("Beta");
        Cleaner c1 = new Cleaner ("Gama", 60);
        Cleaner c2 = new Cleaner ("Delta");
        // ***
    }
}
```

א. ציירו את העצמים שנוצרו בפעולה main , וכתבו את הפלט של הפעולה.

ב. הציבו כל אחת מן הפקודות 1-10 שלהלן בפעולה main במקום המצוין לעיל ב- ' .

כתבו במחברת את מספר הפקודה וציינו אם הקוד תקין או לא תקין. אם הקוד תקין - כתבו את הפלט (אין צורך לחזור על הפלט של הפעולות הבונות מסעיף א), ואם הוא אינו תקין - הסבירו מדוע.

הערה: אין קשר בין הפקודות. כלומר, יש להתייחס לכל פקודה כאילו היא היחידה בפעולה main .

1. System.out.println (r1.describe() + " " + Robot.getMade());
2. r2.work (1);
3. r2.work (1, 2);
4. c1.work (1, 2);
5. System.out.println (r2.describe());
6. c2 = r2;
7. ((Cleaner)r1).work (1, 1);
8. Robot r3 = new Cleaner ("Eps", 50, 2);
9. Cleaner c3 = (Cleaner)r2; c3.work (0, 1);
10. ((Robot)c2).work (2);

נוסח #C

לפניכם המחלקות Robot (רובוט) ו־Cleaner (רובוט ניקוי):

C#

```

public class Robot
{
    private static int made = 0;
    protected string name;
    protected int energy;

    public Robot (string name) {
        this.name = name;
        this.energy = 100;
        made++;
        Console.WriteLine ("Robot " + name);
    }
    public static int GetMade() {
        return made;
    }
    public int GetEnergy() {
        return energy;
    }
    public virtual void Work (int hours) {
        this.energy -= 10 * hours;
        Console.WriteLine (this.name + " worked, energy = " + this.energy);
    }
    public virtual string Describe() {
        return this.name + "(" + this.energy + ")";
    }
}

public class Cleaner : Robot
{
    private int rooms;

    public Cleaner (string name) : base (name) {
        this.rooms = 0;
        Console.WriteLine ("Cleaner");
    }
    public Cleaner (string name, int energy) : base (name) {
        this.energy = energy;
        this.rooms = 0;
        Console.WriteLine ("Cleaner " + energy);
    }
    public override void Work (int hours) {
        base.Work (hours);
        this.rooms += 2 * hours;
    }
    public void Work (int hours, int extra) {
        Work (hours + extra);
        Console.WriteLine ("rooms = " + this.rooms);
    }
    public override string Describe() {
        return base.Describe() + " rooms: " + this.rooms;
    }
}

```

נתונה המחלקה Tester :

```
public class Tester
{
    public static void Main(string[] args)
    {
        Robot r1 = new Robot ("Alfa");
        Robot r2 = new Cleaner ("Beta");
        Cleaner c1 = new Cleaner ("Gama", 60);
        Cleaner c2 = new Cleaner ("Delta");
        // ***
    }
}
```

א. ציירו את העצמים שנוצרו בפעולה Main, וכתבו את הפלט של הפעולה.

ב. הציבו כל אחת מן הפקודות 1-10 שלהלן בפעולה Main במקום המצוין לעיל ב- '.

כתבו במחברת את מספר הפקודה וציינו אם הקוד תקין או לא תקין. אם הקוד תקין - כתבו את הפלט (אין צורך לחזור על הפלט של הפעולות הבונות מסעיף א), ואם הוא אינו תקין - הסבירו מדוע.

הערה: אין קשר בין הפקודות. כלומר, יש להתייחס לכל פקודה כאילו היא היחידה בפעולה Main.

```
1. Console.WriteLine (r1.Describe() + " " + Robot.GetMade());
2. r2.Work (1);
3. r2.Work (1, 2);
4. c1.Work (1, 2);
5. Console.WriteLine (r2.Describe());
6. c2 = r2;
7. ((Cleaner)r1).Work (1, 1);
8. Robot r3 = new Cleaner ("Eps", 50, 2);
9. Cleaner c3 = (Cleaner)r2; c3.Work (0, 1);
10. ((Robot)c2).Work (2);
```

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