מדינת ישראל
משרד החינוך

סוג הבחינה: בגרות
מועד הבחינה: קיץ תשפ"ו, 2026
מספר השאלון: 899271

מדעי המחשב

הוראות

א. משך הבחינה: שלוש שעות.

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

פרק ראשון	–	25×2	50 נקודות
פרק שני	–	25×2	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 יוּרדוּ 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!

/המשך מעבר לדף/

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.

פרק ראשון (50 נקודות)

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

1. נתונה המחלקה **Order** – הזמנה של לקוח, ולה שתי תכונות:

- id – מספר זהות של הלקוח, מטיפוס שלם
- count – כמות המוצרים שהוזמנו, מטיפוס שלם

הניחו שיש פעולות get/Get ו־ set/Set לתכונות המחלקה, ופעולה בונה המקבלת ערכים בעבור תכונות המחלקה.

בחברת המשלוחים "ברק" נבנה תור – qOrder מטיפוס **Order** , השומר את ההזמנות השונות של הלקוחות ביום מסוים, שחברת המשלוחים צריכה לספק.

הערות:

- ייתכן שיהיו בתור כמה הזמנות של אותו הלקוח – id (אם יש ללקוח יותר מהזמנה אחת באותו היום).
- מיקום ההזמנות בתור אינו לפי סדר כלשהו (וגם ההזמנות של אותו הלקוח יכולות להופיע במקומות שונים בתור).

בסוף היום, כדי לייעל את המשלוחים, מבצעים בתור חדש איחוד הזמנות לפי מזהה לקוח (id), כך שלכל לקוח נשמרת בתור הזמנה אחת בלבד, עם סך כל המוצרים שהוא הזמין בכל ההזמנות (count). כך למשל, אם יש בתור qOrder 3 הזמנות של אותו הלקוח: הזמנה של 20 מוצרים, הזמנה של 15 מוצרים והזמנה של 30 מוצרים, לאחר האיחוד יופיע הלקוח רק פעם אחת בתור החדש – עם הזמנה של 65 מוצרים.

א. **(1)** ממשו את הפעולה שלפניכם:

Java – public static Queue<Order> uniteOrders (Queue<Order> qOrder)

C# – public static Queue<Order> UniteOrders (Queue<Order> qOrder)

הפעולה מקבלת תור שיש בו הזמנות – qOrder ובו אותו לקוח יכול להופיע יותר מפעם אחת, ומחזירה תור חדש ובו יש איחוד הזמנות כפי שהוסבר לעיל.

הערות:

- אין חשיבות לסדר ההזמנות לאחר איחוד ההזמנות.
- אין להשתמש בסעיף זה במערך או ברשימה מקושרת (או בשום מבנה נתונים אחר פרט לתור).
  פתרון הכולל שימוש בהם לא יזוכה בנקודות.
- אפשר לשנות את התור שהתקבל.

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

(שימו לב: המשך השאלה בעמוד הבא.)

/המשך בעמוד 3

ב. **(1)** "לקוח רגיל" הוא לקוח שהזמין פחות מ- 10 מוצרים **סך הכול** ביום מסוים, ו"לקוח מועדף" הוא לקוח שהזמין 10 מוצרים ומעלה **סך הכול** ביום מסוים.

ממשו את הפעולה שלפניכם:

Java – public static Queue<Integer> preferredClients (Queue<Order> qOrder)

C# – public static Queue<int> PreferredClients (Queue<Order> qOrder)

הפעולה מקבלת תור הזמנות של יום מסוים – qOrder מטיפוס **Order** (תור "לא מאוחד", שבו אותו לקוח יכול להופיע כמה פעמים), ומחזירה תור מטיפוס **שלם**. בתחילת התור המוחזר יופיעו מספרי הזהות (id) של כל מי שהוא "לקוח מועדף", ואחריהם יופיעו מספרי הזהות (id) של כל מי שהוא "לקוח רגיל" (ללא חשיבות לסדר הלקוחות בכל קבוצה).

אפשר להשתמש בפעולה שכתבתם בסעיף א.

<u>הערות</u>:

- כל לקוח (id) יופיע פעם אחת בלבד בתור המוחזר.
- <u>אין להשתמש בסעיף זה במערך או ברשימה מקושרת</u> (או בשום מבנה נתונים אחר פרט לתור). פתרון הכולל שימוש בהם לא יזוכה בנקודות.
- אפשר לשנות את התור שהתקבל.

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

/המשך בעמוד 4/

2. נתונות שתי הפניות לשרשרות של חוליות:

numHead – הפניה לשרשרת חוליות של מספרים מטיפוס שלם.

charHead – הפניה לשרשרת חוליות מטיפוס תו (char), שבה מיוצגות פעולות החשבון חיבור וחיסור ('+', '−').

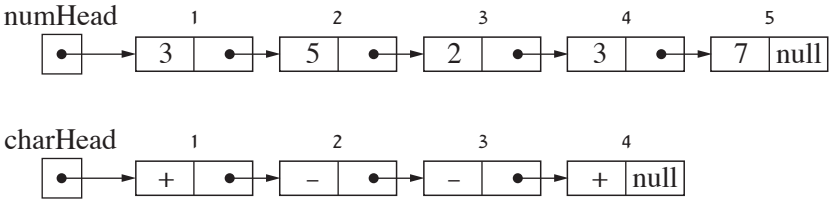
הערה: כמות החוליות בשרשרת המספרים גדולה ב־ 1 מכמות החוליות בשרשרת התווים.

שילוב בין שתי השרשרות מייצג סדרת פעולות חשבון, כמפורט לפניכם:

התו במיקום 1 בשרשרת התווים מתאר את הפעולה שבאה לפני המספר שבמיקום 2 בשרשרת המספרים, התו במיקום 2 מתאר את הפעולה שבאה לפני המספר שבמיקום 3, התו במיקום 3 מתאר את הפעולה שבאה לפני המספר שבמיקום 4, וכן הלאה עד סוף שתי השרשרות.

שימו לב: אין פעולת חשבון לפני המספר הראשון בשרשרת המספרים.

דוגמה: שתי השרשרות שלפניכם מייצגות את סדרת פעולות החשבון: $3+5-2-3+7$.



א. ממשו את הפעולה שלפניכם:

Java – public static int eval (Node<Integer> numHead, Node<Character> charHead, int len)

C# – public static int Eval (Node<int> numHead, Node<char> charHead, int len)

len הוא כמות **המספרים** שנרצה לחשב מתוך שרשרת המספרים (אין פעולת חשבון לפני המספר הראשון ולכן כמות התווים היא len–1 בהתאם).

הפעולה תחזיר את תוצאת החישוב של len מספרים ברצף, החל מתחילת שרשרת המספרים ותחילת שרשרת התווים.

הניחו ש־ len גדול מ־ 0 . אין לשנות את השרשרות שהתקבלו.

הערה: אם len גדול מכמות החוליות בשרשרת המספרים, הפעולה תחזיר את תוצאת החישוב של כל המספרים בשרשרת.

דוגמה: בעבור שתי השרשרות בדוגמה שלעיל ו־ $\text{len} = \mathbf{1}$, הפעולה תחזיר 3 , כיוון שהוא המספר הראשון בשרשרת המספרים (היות שהוא המספר הראשון והיחיד, אין שום פעולת חשבון שצריך לבצע).

דוגמה נוספת: בעבור אותן שתי שרשרות ו־ $\text{len} = \mathbf{3}$, הפעולה תחזיר 6 ($3+5-2=6$).

דוגמה נוספת: בעבור אותן שתי שרשרות ו־ $\text{len} = \mathbf{8}$, הפעולה תחזיר 10 ($3+5-2-3+7=10$).

הסבר: מספר החוליות בשרשרת המספרים קטן מ־ 8 , ולכן הפעולה מחזירה את תוצאת החישוב של כל המספרים בשרשרת.

(שימו לב: המשך השאלה בעמוד הבא.)

ב. ממשו את הפעולה שלפניכם:

Java – public static boolean match (Node<Integer> numHead, Node<Character> charHead, int len, int val)

C# – public static bool Match (Node<int> numHead, Node<char> charHead, int len, int val)

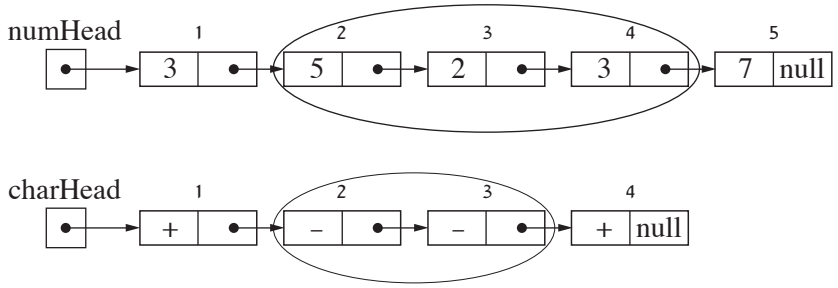
הפעולה תחזיר true אם קיימת חוליה **במיקום כלשהו** בשרשרת המספרים וחוליה **במיקום זהה** בשרשרת התווים, שמהם והלאה יש **len** מספרים ברצף שתוצאת החישוב שלהם היא **val** (כלומר אם מתחילים את החישוב **מן החוליה השנייה** בשרשרת המספרים, גם פעולת החשבון הראשונה תהיה **החוליה השנייה** בשרשרת התווים). אחרת הפעולה תחזיר false .

אפשר להשתמש בפעולה שכתבתם בסעיף א. הניחו ש־ len גדול מ־ 0 . אין לשנות את השרשרות שהתקבלו.

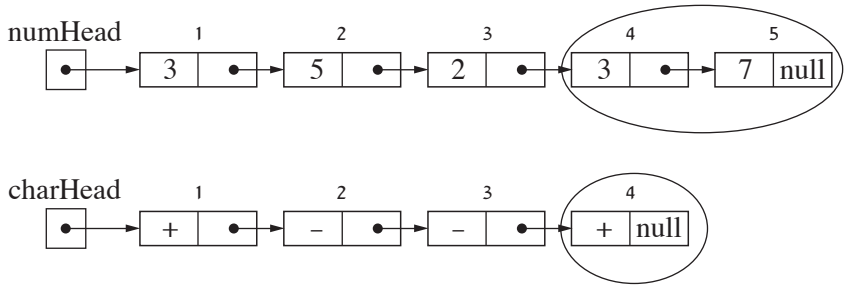
שימו לב: אין פעולת חשבון לפני המספר הראשון בחישוב (מספר התווים הנדרש הוא len–1).

הערה: אם מחוליה כלשהי יש פחות מ־ len מספרים עד סוף השרשרת – החישוב מתבצע עד סוף השרשרת.

דוגמה: בעבור שתי השרשרות המוצגות בדוגמה שלעיל, $len = 3$ ו־ $val = 0$, הפעולה תחזיר **true** , כי החל **מן החוליה השנייה** בשרשרת המספרים ו**מן החוליה השנייה** בשרשרת התווים יש 3 מספרים ברצף שתוצאת החישוב שלהם היא 0 ($0 = 3 - 2 - 5$).



דוגמה נוספת: בעבור שתי השרשרות המוצגות בדוגמה שלעיל, $len = 3$ ו־ $val = 10$, הפעולה תחזיר **true** , כי החל **מן החוליה הרביעית** בשרשרת המספרים ו**מן החוליה הרביעית** בשרשרת התווים ועד סוף שרשרת המספרים תוצאת החישוב היא 10 ($10 = 7 + 3$).



/המשך בעמוד 6/

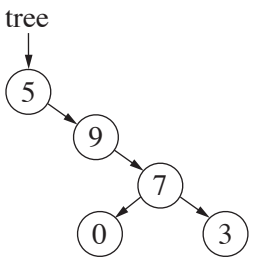
3. שימו לב: לשאלה זו שני נוסחים: בשפת **Java** בעמודים 6–7 ובשפת **C#** בעמודים 8–9.

<u>לפותרים בשפת Java</u>

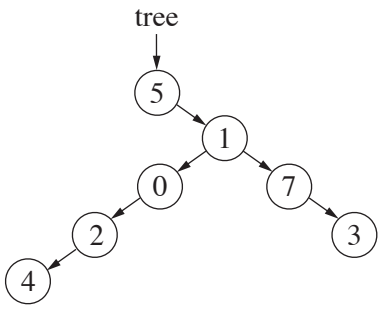
א. נתונה הפעולה foo שלפניכם:

```
public static int foo (BinNode<Integer> tree) {
    if (tree.getLeft() == null && tree.getRight() == null)
        return 0;
    if (tree.getLeft() == null)
        return foo (tree.getRight()) + 1;
    if (tree.getRight() == null)
        return foo (tree.getLeft()) − 1;
    return foo (tree.getLeft()) + foo (tree.getRight());
}
```

(1) בצעו מעקב אחרי הפעולה foo והעץ שלפניכם, וכתבו מה הפעולה מחזירה. יש להציג את המעקב.



(2) בצעו מעקב אחרי הפעולה foo והעץ שלפניכם, וכתבו מה הפעולה מחזירה. יש להציג את המעקב.



(3) בעבור עץ שיש לו 6 צמתים, מהו המספר הכי גבוה שהפעולה foo יכולה להחזיר? הציגו את העץ.

(4) בעבור עץ שיש לו 6 צמתים, מהו המספר הכי נמוך שהפעולה foo יכולה להחזיר? הציגו את העץ.

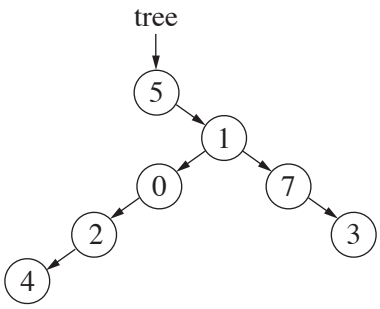
(5) כתבו בקצרה מה הפעולה foo מחזירה בעבור עץ כלשהו שאינו ריק.

(שימו לב: המשך השאלה בעמוד הבא.)

ב. נתונה הפעולה goo שלפניכם:

```
public static boolean goo (BinNode<Integer> tree) {
    if (tree == null)
        return true;
    if (foo (tree) != 0)
        return false;
    return goo (tree.getLeft()) && goo (tree.getRight());
}
```

(1) בצעו מעקב אחרי הפעולה goo והעץ שלפניכם, וכתבו מה הפעולה מחזירה. יש להציג את המעקב.
בסעיף זה אין צורך להציג מעקב אחרי הפעולה foo .



(2) הציגו עץ שיש לו <u>מעל</u> 3 צמתים שבעבורו הפעולה goo תחזיר תוצאה <u>הפוכה</u> מן התוצאה שהתקבלה בתת־סעיף ב(1).

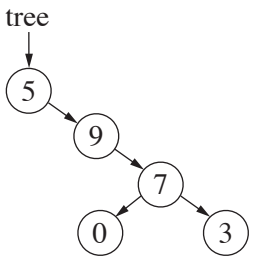
/המשך בעמוד 8/

לפותרים בשפת C#

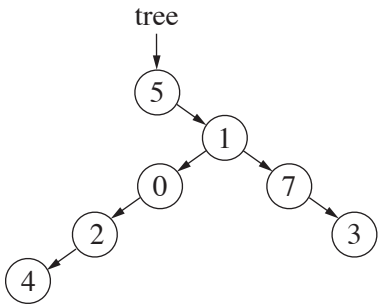
א. נתונה הפעולה Foo שלפניכם:

```
public static int Foo (BinNode<int> tree)
{
    if (tree.GetLeft() == null && tree.GetRight() == null)
        return 0;
    if (tree.GetLeft() == null)
        return Foo (tree.GetRight()) + 1;
    if (tree.GetRight() == null)
        return Foo (tree.GetLeft()) – 1;
    return Foo (tree.GetLeft()) + Foo (tree.GetRight());
}
```

(1) בצעו מעקב אחרי הפעולה Foo והעץ שלפניכם, וכתבו מה הפעולה מחזירה. יש להציג את המעקב.



(2) בצעו מעקב אחרי הפעולה Foo והעץ שלפניכם, וכתבו מה הפעולה מחזירה. יש להציג את המעקב.



(3) בעבור עץ שיש לו 6 צמתים, מהו המספר הכי גבוה שהפעולה Foo יכולה להחזיר? הציגו את העץ.

(4) בעבור עץ שיש לו 6 צמתים, מהו המספר הכי נמוך שהפעולה Foo יכולה להחזיר? הציגו את העץ.

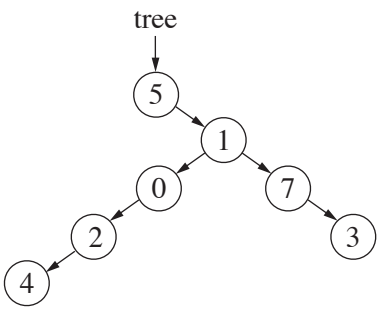
(5) כתבו בקצרה מה הפעולה Foo מחזירה בעבור עץ כלשהו שאינו ריק.

(שימו לב: המשך השאלה בעמוד הבא.)

ב. נתונה הפעולה Goo שלפניכם:

```
public static bool Goo (BinNode<int> tree)
{
    if (tree == null)
        return true;
    if (Foo (tree) != 0)
        return false;
    return Goo (tree.GetLeft()) && Goo (tree.GetRight());
}
```

(1) בצעו מעקב אחרי הפעולה Goo והעץ שלפניכם, וכתבו מה הפעולה מחזירה. יש להציג את המעקב.
בסעיף זה אין צורך להציג מעקב אחרי הפעולה Foo .



(2) הציגו עץ שיש לו <u>מעל</u> 3 צמתים שבעבורו הפעולה Goo תחזיר תוצאה <u>הפוכה</u> מן התוצאה שהתקבלה בתת־סעיף ב(1).

/המשך בעמוד 10/

פרק שני (50 נקודות)

בפרק זה שאלות בשלושה מסלולים:

אלגוריתמים, עמודים 10–14.

מודלים חישוביים, עמודים 15–17.

תכנות מונחה עצמים (שפת Java ושפת C#), עמודים 18–27.

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

אלגוריתמים

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

א. לפניכם שבע טענות (1)–(7). בחרו בחמש מהן, וכתבו את מספריהן. ציינו בנוגע לכל טענה שבחרתם אם היא נכונה או לא נכונה. אם הטענה נכונה – נמקו מדוע, ואם היא לא נכונה – הביאו דוגמה נגדית מגרף שיש בו ארבעה קודקודים לפחות.

(1) נתון גרף G שאינו מכוון. אם בגרף יש קשת בין שני הקודקודים x , y, הרכיבים הקשירים של x ושל y זהים.

(2) נתון גרף G מכוון. אם בגרף יש קשת מקודקוד y לקודקוד x וקשת מקודקוד x לקודקוד y, הרכיבים הקשירים היטב של x ושל y זהים.

(3) בגרף מכוון שיש בו מעגל, יש לפחות שני קודקודים שדרגת הכניסה שלהם גדולה מ־ 0.

(4) אם בגרף ממושקל קשיר לא מכוון יש לכל קשת משקל שונה, אז המסלול הקצר (הקל) בין כל שני קודקודים בגרף הוא מסלול יחיד (כלומר אין יותר ממסלול קצר אחד בין קודקוד לקודקוד).

(5) בגרף מכוון שבו המשקל של כל קשת הוא משקל שונה יש עץ פורש מינימלי יחיד.

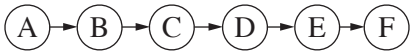
(6) לא קיים גרף מכוון ממושקל שבו האלגוריתם של דייקסטרה מקודקוד y אל קודקוד x ייתן את אותו מסלול קצר (קל) שהאלגוריתם BFS נותן.

(7) אפשר לעשות מיון טופולוגי בעבור כל גרף מכוון.

(שימו לב: המשך השאלה בעמוד הבא.)

ב. עץ פורש "**שרשרת**" הוא עץ שבו לכל צומת יש בן אחד לכל היותר.

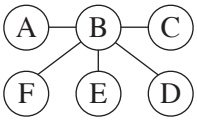
דוגמה לעץ פורש "**שרשרת**":



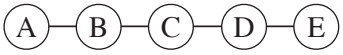
גרף קשיר לא מכוון מכונה "**DFS שרשרתי**" אם בכל סריקת DFS מכל צומת בגרף מתקבל עץ פורש "**שרשרת**".

לפניכם שבעה גרפים $G_1 - G_7$. בחרו בארבעה מהם, וכתבו בנוגע לכל גרף שבחרתם אם הוא גרף "**DFS שרשרתי**" או לא. אם כתבתם שלא – הציגו סריקת DFS שבעבורה מתקבל גרף שאינו עץ פורש "**שרשרת**".

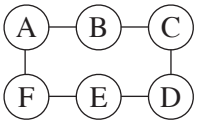
G_1:



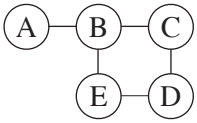
G_2:



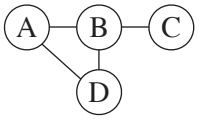
G_3:



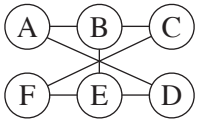
G_4:



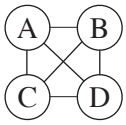
G_5:



G_6:

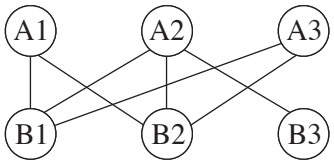


G_7:



/המשך בעמוד 12/

5. לפניכם גרף לא מכוון דו־צדדי $G = (A, B, E)$.



צמתים בצד A	$\mathbf{A} = \{A1, A2, A3\}$
צמתים בצד B	$\mathbf{B} = \{B1, B2, B3\}$
קשתות בין A ל־B	$\mathbf{E} = \{(A1-B1), (A1-B2), (A2-B1), (A2-B2), (A2-B3), (A3-B1), (A3-B2)\}$

א. הצומת M מוגדר "שכן" של הצומת U, אם יש קשת ביניהם.

לפניכם טבלת שכנויות המציגה את השכנים של כל צומת בגרף G. הצמתים השכנים של A1 ושל B1 נתונים בטבלה.

השלימו את הטבלה.

U	A1	A2	A3	B1	B2	B3
M	B1 , B2			A1 , A2 , A3		

ב. נתונה רשימה של חלק מן הקשתות בגרף G: $L = \{(A1-B1), (A2-B2)\}$.

"**צומת תפוס**" הוא צומת שמופיע ברשימה L (הצמתים A1 , A2 , B1 , B2) מופיעים ולכן הם "צמתים תפוסים").

"**צומת חופשי**" הוא צומת שאינו מופיע ברשימה L (הצמתים A3 , B3) אינם מופיעים ולכן הם "צמתים חופשיים").

נתון האלגוריתם **Path** (G, L).

<u>שלב אתחול נתונים באלגוריתם</u>:

1. נגדיר תור – q, שאליו נכניס את הצומת A3.

<u>שימו לב</u>: במהלך ריצת האלגוריתם ייכנסו לתור צמתים נוספים השייכים לצד A או לצד B.

2. נגדיר מערך – visited בגודל 6, ובו נסמן את כל הצמתים שבהם ביקרנו במהלך ריצת האלגוריתם (בהתחלה בתא של A3 מופיע הערך כן, ובשאר התאים במערך מופיע הערך לא).

visited:

A1	A2	A3	B1	B2	B3
לא	לא	כן	לא	לא	לא

3. נגדיר מערך – parent בגודל 6, ובו נשמור בעבור כל צומת את ה"הורה" שלו, כלומר את הצומת שממנו הגענו אליו במהלך ריצת האלגוריתם (בהתחלה כל התאים במערך ריקים).

parent:

A1	A2	A3	B1	B2	B3

(שימו לב: המשך השאלה בעמוד הבא.)

שלב הרצת האלגוריתם:

1. כל עוד התור – q אינו ריק:
   - נוציא את הצומת הנמצא בראש התור, ונקרא לו U .
   - נבדוק אם U נמצא בצד B והוא גם "צומת חופשי":
     - **אם כן** – האלגוריתם מחזיר רשימת צמתים שהם: U , ההורה של U וההורה של ההורה של U וכן הלאה, עד הצומת שאין לו הורה (כלומר לצומת שהתא שלו ריק במערך parent), **והאלגוריתם מסתיים**.
     - **אם לא** (כלומר אם U נמצא בצד A או אם U הוא "צומת תפוס") – בעבור כל צומת שכן – M של הצומת U :

אם **עדיין לא ביקרנו** בצומת M (לא visited[M] =), וגם **אחד מן התנאים** שלהלן מתקיים:

– U בצד **A** והקשת (U–M) **אינה נמצאת** ברשימה L

– U בצד **B** והקשת (U–M) **נמצאת** ברשימה L

אז:

א. נכניס את M לתור.

ב. נכתוב את הערך **כן** בתא M במערך visited .

ג. נעדכן שהצומת U הוא ההורה של M במערך parent .

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

עקבו בטבלת מעקב אחרי ריצת האלגוריתם Path על הגרף הנתון G , עם הרשימה L .

המעקב צריך לכלול בכל איטרציה את הפריטים שלהלן:

- מצב התור
- הצומת U שהוצאנו מן התור
- מערך visited
- מערך parent

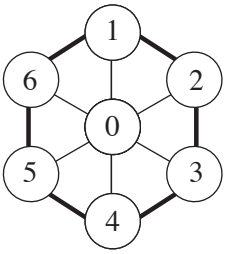
התור	U	מערך visited						מערך parent					
		A1	A2	A3	B1	B2	B3	A1	A2	A3	B1	B2	B3
← [ ]													
← [ ]													

ג. כתבו את **רשימת הצמתים** שהאלגוריתם מחזיר.

/המשך בעמוד 14/

6. גרף "**כוכב סגור**" – G_n הוא גרף לא מכוון שמורכב ממעגל של n קודקודים (מ־ 1 עד n), המכונים "**קודקודי המעגל**", ומקודקוד נוסף – 0 , המכונה "**קודקוד מרכז**", שנמצא במרכז המעגל ומחובר לכל אחד מ"קודקודי המעגל" (כך שסך הכול יש $\mathbf{n+1}$ קודקודים בגרף). מספר "קודקודי המעגל" בגרף גדול מ־ 2 (כלומר $n > 2$).

<u>דוגמה</u>: הגרף שלפניכם הוא גרף "**כוכב סגור**" – G_6 , כי הקודקודים 1 עד 6 יוצרים מעגל, והקודקוד הנוסף, 0 , נמצא במרכז ומחובר לכל אחד מהם.



"**קשתות המעגל**" הן הקשתות המחברות את "קודקודי המעגל" זה לזה (למשל בדוגמה לעיל, הקשת שמחברת את הקודקודים 1 ו־ 2 זה לזה, הקשת שמחברת את הקודקודים 2 ו־ 3 זה לזה וכן הלאה).

"**קשתות מרכזיות**" הן הקשתות המחברות את "קודקוד מרכז" לכל אחד מ"קודקודי המעגל" (למשל בדוגמה לעיל, הקשת שמחברת את קודקוד 0 לקודקוד 1 , הקשת שמחברת את קודקוד 0 לקודקוד 2 וכן הלאה).

א. נתון גרף "**כוכב סגור**" – G_n .

(1) מה הן דרגות "קודקודי המעגל" בגרף, ומהי דרגת "קודקוד מרכז"? נמקו את תשובתכם.

(2) כמה "קשתות מעגל" יש בגרף, וכמה "קשתות מרכזיות" יש בגרף? נמקו את תשובתכם.

(3) האם הגרף הוא דו־צדדי? נמקו את תשובתכם.

ב. נתון גרף ממושקל שהוא "**כוכב סגור**" – G_n , שבו לכל "קשתות המעגל" משקל 2 , ולכל "הקשתות המרכזיות" משקל 1 . כמה עפ"מים (עצים פורשים מינימליים) שונים יש בגרף? נמקו את תשובתכם.

ג. נתון גרף ממושקל שהוא "**כוכב סגור**" – G_n , שבו לכל "קשתות המעגל" משקל b (גדול מ־ 0), ולכל "הקשתות המרכזיות" משקל m (גדול מ־ 0).

(1) הציבו ערכים ל־ b ו־ m , שעבורם <u>כל</u> מסלול קצר (קל) בין שני "קודקודי מעגל" יעבור דרך "קודקוד מרכז". נמקו את תשובתכם.

(2) נתון כי $b = 10$. מהו הערך הגדול ביותר של m שבעבורו <u>כל</u> מסלול קצר (קל) בין שני "קודקודי מעגל" עדיין יעבור דרך "קודקוד מרכז"? נמקו את תשובתכם.

/המשך בעמוד 15/

<u>מודלים חישוביים</u>

7. לפניכם שני סעיפים א–ב, שאין קשר ביניהם. ענו על <u>שני</u> הסעיפים.

א. **(1)** לפניכם שלוש שפות $L_3 - L_1$ מעל הא"ב $\{a, b, c\}$:

$$L_1 = \{a^n b^k c^n \mid 0 < n < 1000 < k\}$$

$$L_2 = \{a^n b^k c^n \mid 0 < k < 1000 < n\}$$

$$L_3 = \{a^n b^k c^n \mid 0 < k < 1000 < n < 2000\}$$

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

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

$$L_4 = L_2 \cap R(L_2)$$

$$L_5 = L_1 \cup L_2$$

$$L_6 = \overline{L_1} \cup \overline{L_3}$$

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

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

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

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

- אם $n + k$ מתחלק ב־ 3 בלי שארית, w היא מילה ריקה.
- אם $n + k$ מתחלק ב־ 3 עם שארית 1, w היא מילה שאינה מכילה את האות b כלל.
- אם $n + k$ מתחלק ב־ 3 עם שארית 2, w היא מילה שאינה מכילה את האות c כלל.

<u>דוגמה למילה השייכת לשפה L</u>: $\overbrace{a}^{a^n}\overbrace{b\,b}^{b^m}\overbrace{a\,a\,a}^{a^k}\overbrace{c\,c\,c}^{w}$

<u>הסבר</u>: $n = 1$, $k = 3$. $n + k$ מתחלק ב־ 3 עם שארית 1, ו־ w היא מילה שאינה מכילה את האות b.

בנו אוטומט סופי דטרמיניסטי שאינו מלא המקבל את השפה L.

/המשך בעמוד 16/

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

$$L = \{a^n b^m c^k \mid \begin{cases} n\%2 = 0: & m\%2 = 0,\ n = m/2 + k,\ n,m,k > 0 \\ n\%2 = 1: & n \le m + k - 2 \end{cases}\}$$

<u>דוגמה למילה בשפה L בעבור n **זוגי**</u>: $\overbrace{a\,a\,a\,a}^{n=4}\overbrace{b\,b}^{m=2}\overbrace{c\,c\,c}^{k=3}$.

הסבר: $n\%2 = 0$, ובהתאם לכך גם $m\%2 = 0$, ומתקיים $4 = (2 / 2) + 3$ וגם $n, m, k > 0$.

<u>דוגמה למילה בשפה L בעבור n **אי־זוגי**</u>: $\overbrace{a\,a\,a}^{n=3}\overbrace{b\,b\,b}^{m=3}\overbrace{c\,c\,c\,c}^{k=4}$.

הסבר: $n\%2 = 1$, ובהתאם לכך מתקיים $3 \le 3 + 4 - 2$.

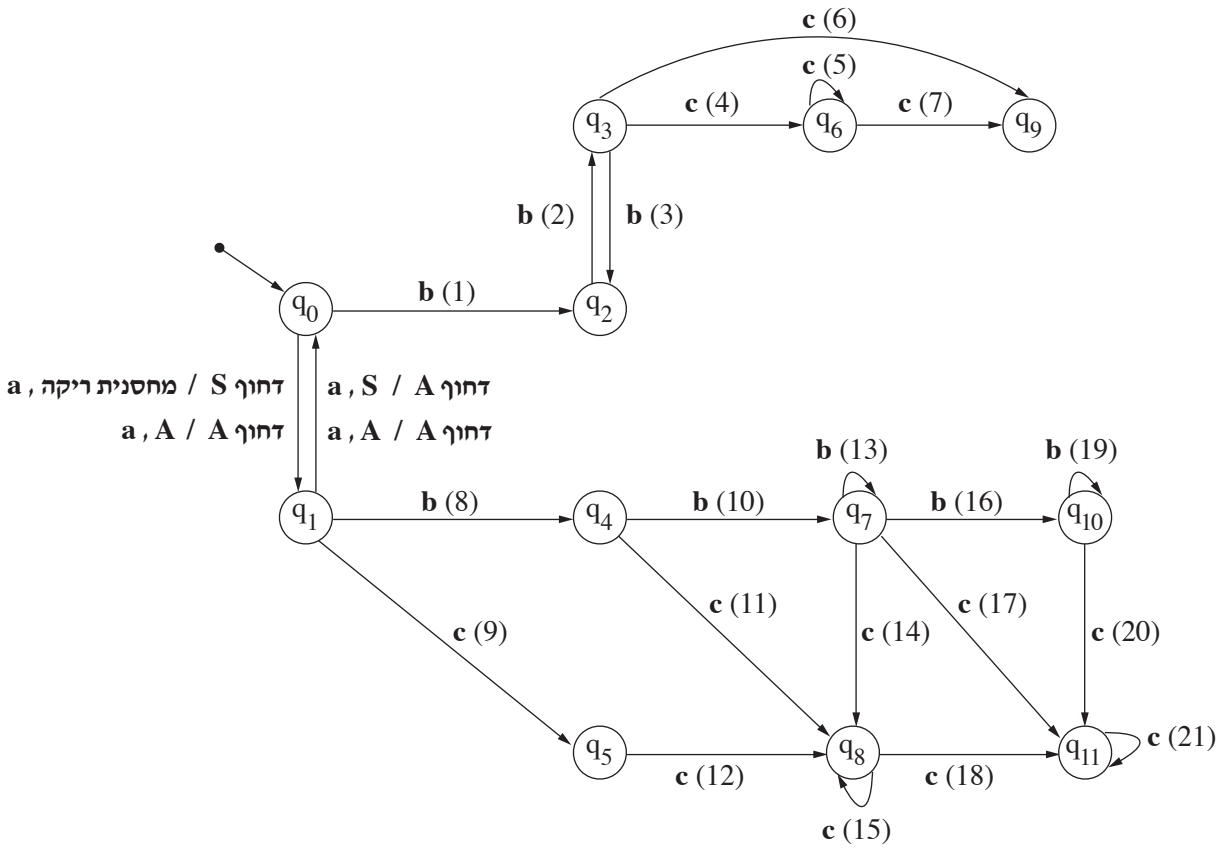
א. לפניכם חמש מילים. בעבור <u>כל אחת</u> מהן, ציינו אם המילה שייכת לשפה L. נמקו את תשובתכם.

abcc, acccc, aabbc, aabbbb, aaabbc.

ב. כתבו את המילה הקצרה ביותר בשפה L בעבור n זוגי, ואת אחת מן המילים הקצרות ביותר בעבור n אי־זוגי (יש יותר מאפשרות אחת).

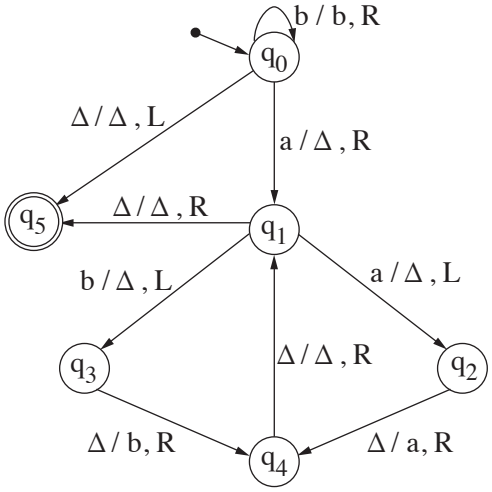
ג. לפניכם אוטומט **מחסנית דטרמיניסטי** המקבל את השפה L. באוטומט קיימים כל המצבים, המעברים וסימני הקלט הנדרשים. אולם מלבד המעברים בין q_0 ל־ q_1, חסרות במעברים האות שבראש המחסנית והפעולה על המחסנית (מעברים אלה ממוספרים מ־ 1 עד 21). נוסף על כך, המצבים המקבלים באוטומט לא סומנו.

בעבור כל מעבר ממוספר, ציינו את מספרו והשלימו את הסימון החסר: האות שבראש המחסנית והפעולה שעל המחסנית (דחוף/push או שלוף/pop או ללא שינוי). נוסף על כך, כתבו מה הם המצבים המקבלים באוטומט (אין צורך להעתיק את האוטומט למחברתכם).



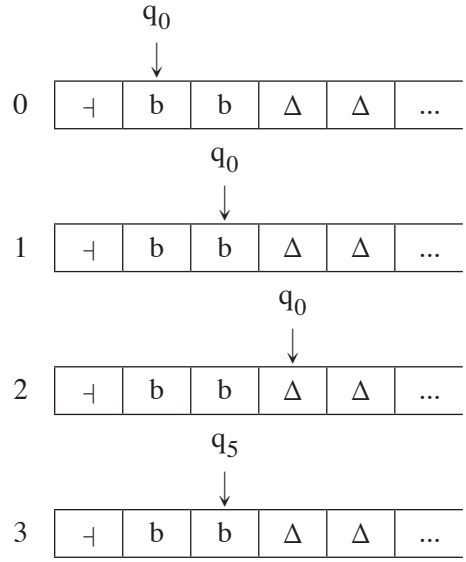
/המשך בעמוד 17/

9. 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 $q_5 - q_0$ נמצאים.

להלן דוגמה לאופן שבו יש לבצע מעקב בעבור המכונה הנתונה והמילה bb.



א. **(1)** עקבו אחרי פעולת המכונה בעבור המילה a . יש להציג את המעקב.

(2) עקבו אחרי פעולת המכונה בעבור המילה aba . יש להציג את המעקב.

(3) הציגו את הסרט המתקבל בגמר הפעולה בעבור המילה baba . אין צורך להציג את המעקב.

ב. **(1)** האם קיימת מילה המכילה גם את a וגם את b שאינה משתנה לאחר הרצת המכונה עליה? אם כן – כתבו דוגמה למילה כזו. אם לא – נמקו את תשובתכם.

(2) כתבו בקצרה מה המכונה עושה.

/המשך בעמוד 18/

תכנות מונחה עצמים

שימו לב: השאלות 10–12 הן גם בעבור הפותרים בשפת Java וגם בעבור הפותרים בשפת C# .
לכל שאלה נוסח בשפת Java ונוסח בשפת C# .

10. לשאלה זו שני נוסחים: בשפת **Java** בעמוד 18 ובשפת **C#** בעמוד 19.

לפותרים בשפת Java

באקדמיה למוזיקה "**שיר חדש**" פותחה מערכת ממוחשבת ובה המחלקות האלה:

Instrum – כלי נגינה, **Bow** – כלי קשת , **Wind** – כלי נשיפה, **Perc** – כלי הקשה, **Musician** – נגן בתזמורת, **Orchestra** – תזמורת.

להלן פירוט תכונות המחלקות:

למחלקה **Instrum** (כלי נגינה) שתי תכונות: name – שם הכלי (מטיפוס מחרוזת), material – החומר שממנו עשוי הכלי (מטיפוס מחרוזת).

למחלקה **Bow** (כלי קשת) שלוש תכונות: name – שם הכלי (מטיפוס מחרוזת), material – החומר שממנו עשוי הכלי (מטיפוס מחרוזת), bowLength – אורך הקשת (מטיפוס שלם).

למחלקה **Wind** (כלי נשיפה) שלוש תכונות: name – שם הכלי (מטיפוס מחרוזת), material – החומר שממנו עשוי הכלי (מטיפוס מחרוזת), numberOfValves – מספר השסתומים בכלי (מטיפוס שלם).

למחלקה **Perc** (כלי הקשה) שלוש תכונות: name – שם הכלי (מטיפוס מחרוזת), material – החומר שממנו עשוי הכלי (מטיפוס מחרוזת), hasSticks – האם יש שימוש במקלות הקשה (מטיפוס בוליאני. אם יש שימוש במקלות – true , ואם לא – false).

למחלקה **Musician** (נגן בתזמורת) שתי תכונות: name – שם הנגן (מטיפוס מחרוזת), instrum – כלי הנגינה שלו (מטיפוס **Instrum**). כל נגן מנגן על כלי אחד בלבד.

למחלקה **Orchestra** (תזמורת) שתי תכונות: arrM – מערך של נגנים (מטיפוס **Musician**), count – כמות הנגנים השמורים במערך (מטיפוס שלם).

הנגנים מופיעים ברצף מתחילת המערך ללא null ביניהם, אך ייתכן שהמערך אינו מלא עד הסוף.

א. **(1)** סרטטו תרשים היירכייה בין כל המחלקות.

יש לסמן ירושה באמצעות החץ ◁—— והכלה באמצעות הסימן ——◆ .

(2) כתבו את כותרות המחלקות ואת התכונות שלהן. הניחו שבכל מחלקה הוגדרו פעולות get ו־ set ופעולות בונות.

ב. כתבו במחלקה **Orchestra** פעולה ששמה withSticks , המחזירה את מספר הנגנים בתזמורת המנגנים בכלי הקשה שיש בהם שימוש במקלות.

ג. מערך arrM "מסודר" הוא מערך שבו מופיעים קודם נגני כלי הקשת, אחר כך נגני כלי הנשיפה, ואחר כך נגני כלי ההקשה.
כתבו במחלקה **Orchestra** פעולה ששמה insert , המקבלת נגן – musician שאינו מופיע במערך arrM .
הניחו שלפני תחילת הפעולה המערך "מסודר", ושיש בתזמורת לפחות נגן אחד בכל סוג של כלי.
אם אין מקום לנגן נוסף במערך, הפעולה תחזיר false . אחרת, הפעולה תכניס את הנגן למערך במקום המתאים באופן שבו המערך ימשיך להיות "מסודר" (והנגנים יופיעו ברצף בתחילת המערך), ותחזיר true .
הערה: אפשר לשנות את סדר הנגנים, ובלבד שהמערך יהיה "מסודר" בסוף הפעולה.

/המשך בעמוד 19

לפותרים בשפת C#

באקדמיה למוזיקה "**שיר חדש**" פותחה מערכת ממוחשבת ובה המחלקות האלה:

Instrum – כלי נגינה, **Bow** – כלי קשת , **Wind** – כלי נשיפה, **Perc** – כלי הקשה, **Musician** – נגן בתזמורת, **Orchestra** – תזמורת.

להלן פירוט תכונות המחלקות:

למחלקה **Instrum** (כלי נגינה) שתי תכונות: name – שם הכלי (מטיפוס מחרוזת), material – החומר שממנו עשוי הכלי (מטיפוס מחרוזת).

למחלקה **Bow** (כלי קשת) שלוש תכונות: name – שם הכלי (מטיפוס מחרוזת), material – החומר שממנו עשוי הכלי (מטיפוס מחרוזת), bowLength – אורך הקשת (מטיפוס שלם).

למחלקה **Wind** (כלי נשיפה) שלוש תכונות: name – שם הכלי (מטיפוס מחרוזת), material – החומר שממנו עשוי הכלי (מטיפוס מחרוזת), numberOfValves – מספר השסתומים בכלי (מטיפוס שלם).

למחלקה **Perc** (כלי הקשה) שלוש תכונות: name – שם הכלי (מטיפוס מחרוזת), material – החומר שממנו עשוי הכלי (מטיפוס מחרוזת), hasSticks – האם יש שימוש במקלות הקשה (מטיפוס בוליאני. אם יש שימוש במקלות – true , ואם לא – false).

למחלקה **Musician** (נגן בתזמורת) שתי תכונות: name – שם הנגן (מטיפוס מחרוזת), instrum – כלי הנגינה שלו (מטיפוס **Instrum**). כל נגן מנגן על כלי אחד בלבד.

למחלקה **Orchestra** (תזמורת) שתי תכונות: arrM – מערך של נגנים (מטיפוס **Musician**), count – כמות הנגנים השמורים במערך (מטיפוס שלם).

הנגנים מופיעים ברצף מתחילת המערך ללא null ביניהם, אך ייתכן שהמערך אינו מלא עד הסוף.

א. **(1)** סרטטו תרשים היירכייה בין כל המחלקות.

יש לסמן ירושה באמצעות החץ ◁—— והכלה באמצעות הסימן ——◆ .

(2) כתבו את כותרות המחלקות ואת התכונות שלהן. הניחו שבכל מחלקה הוגדרו פעולות Get ו־ Set ופעולות בונות.

ב. כתבו במחלקה **Orchestra** פעולה ששמה WithSticks , המחזירה את מספר הנגנים בתזמורת המנגנים בכלי הקשה שיש בהם שימוש במקלות.

ג. מערך arrM "מסודר" הוא מערך שבו מופיעים קודם נגני כלי הקשת, אחר כך נגני כלי הנשיפה, ואחר כך נגני כלי ההקשה.

כתבו במחלקה **Orchestra** פעולה ששמה Insert , המקבלת נגן – musician שאינו מופיע במערך arrM .

הניחו שלפני תחילת הפעולה המערך "מסודר", ושיש בתזמורת לפחות נגן אחד בכל סוג של כלי.

אם אין מקום לנגן נוסף במערך, הפעולה תחזיר false . אחרת, הפעולה תכניס את הנגן למערך במקום המתאים באופן שבו המערך ימשיך להיות "מסודר" (והנגנים יופיעו ברצף בתחילת המערך), ותחזיר true .

הערה: אפשר לשנות את סדר הנגנים, ובלבד שהמערך יהיה "מסודר" בסוף הפעולה.

/המשך בעמוד 20/

11. לשאלה זו שני נוסחים: בשפת **Java** בעמודים 20–21 ובשפת **C#** בעמודים 22–23.

<u>לפותרים בשפת Java</u>

לפניכם שלושה סעיפים א–ג שאין קשר ביניהם. <u>ענו על שלושת הסעיפים</u>.

א. לפניכם המחלקה **Father** ללא מימוש הפעולות הבונות, והמחלקה **Son** ללא מימוש הפעולות הבונות ופעולת toString .

```
public class Father {
    private static int st = 0;
    protected int num;
    public Father () {…}
    public Father (int num) {…}
    public String toString() {return "st=" + st + ", num=" + num;}
}
-----------------------------------------------------------------
public class Son extends Father {
    private char ch;
    public Son (int num, char ch){…}
    public Son (int num) {…}
    public String toString() {... }
}
```

לפניכם קטע קוד והפלט שהתקבל בעקבות הרצת קטע הקוד:

הקוד		הפלט	
Father x1 = new Father ();			
System.out.println (x1);	//1	st=1, num=9	//1
Father x2 = new Father (3);			
System.out.println (x2);	//2	st=1, num=3	//2
Father x3 = new Son (5, 'A');			
System.out.println (x3);	//3	st=2, num=5, ch=A	//3
Father x4 = new Son (7);			
System.out.println (x4);	//4	st=2, num=7, ch=Z	//4

ממשו את הפעולות הבונות במחלקה Father ובמחלקה Son , ואת הפעולה toString במחלקה Son , בהתאם לעקרונות תכנות מונחה עצמים, כך שיתקבל הפלט הנתון.

אין להוסיף פעולות נוספות למחלקות. תשובה הכוללת פעולות נוספות במחלקות, לא תזוכה בנקודות.

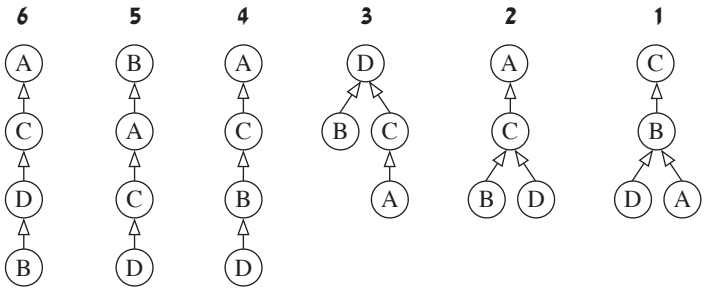
(שימו לב: המשך השאלה בעמוד הבא.)

/המשך בעמוד 21/

ב. נתונות המחלקות **A**, **B**, **C**, **D** והפעולה main במחלקה **Tester**, שרצה באופן תקין.

```
public class Tester {
    public static void main(String[] args) {
        C c1 = new B();
        A a1 = new C();
        A a2 = new B();
        C c2 = new D();
    }
}
```

(1) לפניכם שישה תרשימי היירכייה 1–6 (החץ מסמן יחס ירושה). בנוגע לכל אחד מהם, ציינו אם ייתכן או לא ייתכן שהוא מתאר את יחס ההיירכייה בין המחלקות **A**, **B**, **C**, **D**. אם ציינתם שלא ייתכן, נמקו מדוע.



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

```
A a3 = new B();
D d1 = (D)a3;
```

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

ג. נתונה המחלקה **BBB**, היורשת ממחלקה **AAA**:

```
public class BBB extends AAA {
    private char ch;
    public BBB (int num, char ch) {
        super (num);
        this.ch = ch;
    }
    public BBB (char ch) {this.ch = ch; }
    public int gg (int val) {return super.gg(val) + num; }
    public int mm () {return AAA.c + 2; }
    public boolean nn () {return this.mm (10); }
}
```

כתבו במחלקה AAA את **תכונות** המחלקה (כולל הרשאות הגישה) ואת ה**כותרות** של הפעולות הנדרשות כדי שהמחלקה BBB תהיה תקינה, בהתאם לעקרונות תכנות מונחה עצמים.

<u>אין צורך לממש את הפעולות שאת הכותרות שלהן הוספתם במחלקה AAA. אין לשנות את הקוד של המחלקה BBB</u>.

/המשך בעמוד 22/

לפותרים בשפת C#

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

א. לפניכם המחלקה **Father** ללא מימוש הפעולות הבונות, והמחלקה **Son** ללא מימוש הפעולות הבונות ופעולת ToString .

```
public class Father {
    private static int st = 0;
    protected int num;
    public Father() {…}
    public Father (int num) {…}
    public override string ToString() { return "st=" + st + ", num=" + num; }
}
public class Son : Father {
    private char ch;
    public Son (int num, char ch) ...{…}
    public Son (int num) ...{…}
    public override string ToString() {...}
}
```

לפניכם קטע קוד והפלט שהתקבל בעקבות הרצת קטע הקוד:

הקוד		הפלט	
Father x1 = new Father();			
Console.WriteLine (x1);	//1	st=1, num=9	//1
Father x2 = new Father (3);			
Console.WriteLine (x2);	//2	st=1, num=3	//2
Father x3 = new Son (5, 'A');			
Console.WriteLine (x3);	//3	st=2, num=5, ch=A	//3
Father x4 = new Son (7);			
Console.WriteLine (x4);	//4	st=2, num=7, ch=Z	//4

ממשו את הפעולות הבונות במחלקה Father ובמחלקה Son , ואת הפעולה ToString במחלקה Son , בהתאם לעקרונות תכנות מונחה עצמים, כך שיתקבל הפלט הנתון.

אין להוסיף פעולות נוספות למחלקות. תשובה הכוללת פעולות נוספות במחלקות, לא תזוכה בנקודות.

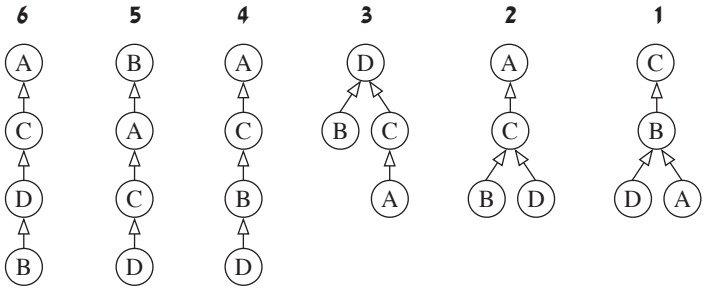
(שימו לב: המשך השאלה בעמוד הבא.)

/המשך בעמוד 23

ב. נתונות המחלקות **A**, **B**, **C**, **D** והפעולה Main במחלקה **Tester**, שרצה באופן תקין.

```
public class Tester {
    public static void Main (string[] args) {
        C c1 = new B();
        A a1 = new C();
        A a2 = new B();
        C c2 = new D();
    }
}
```

(1) לפניכם שישה תרשימי היירכייה 1–6 (החץ מסמן יחס ירושה). בנוגע לכל אחד מהם, ציינו אם ייתכן או לא ייתכן שהוא מתאר את יחס ההיירכייה בין המחלקות **A**, **B**, **C**, **D**. אם ציינתם שלא ייתכן, נמקו מדוע.



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

```
A a3 = new B();
D d1 = (D)a3;
```

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

ג. נתונה המחלקה **BBB**, היורשת ממחלקה **AAA**:

```
public class BBB : AAA {
    private char ch;
    public BBB (int num, char ch) : base (num) {
        this.ch = ch;
    }
    public BBB (char ch) {this.ch = ch;}
    public override int GG (int val) {return base.GG (val) + num;}
    public int MM() {return AAA.c + 2;}
    public bool NN() {return this.MM (10);}
}
```

כתבו במחלקה AAA את **תכונות** המחלקה (כולל הרשאות הגישה) ואת ה**כותרות** של הפעולות הנדרשות כדי שהמחלקה BBB תהיה תקינה, בהתאם לעקרונות תכנות מונחה עצמים.

<u>אין צורך לממש את הפעולות שאת הכותרות שלהן הוספתם במחלקה AAA. אין לשנות את הקוד של המחלקה BBB.</u>

/המשך בעמוד 24/

12. לשאלה זו שני נוסחים: בשפת **Java** בעמודים 24–25 ובשפת **C#** בעמודים 26–27.

<u>לפותרים בשפת Java</u>

לפניכם המחלקות **A**, **B**, **C** :

```
public class A {
    public int num;

    public A() {
        this.num = 1;
    }
    public A(int num) {
        this.num = num;
    }
    public int func (int val) {
        return val * 2;
    }
    public int func (B other) {
        return this.num – other.num;
    }
}
```

```
public class B extends A {
    public B() {
        super(2);
    }
    public B(int val) {
        super(val);
        this.num = this.num + func (val);
    }
    public int func (A other) {
        return this.num + (other.num * 10);
    }
    public int func (B other) {
        return this.num + other.num;
    }
}
```

```
public class C extends B {
    public C(int val)  {
        this.num = this.num – val;
    }
    public int func (B other)  {
        return this.num * other.num;
    }
    public int func (C c)  {
        return this.num * 4;
    }
}
```

(שימו לב: המשך השאלה בעמוד הבא.)

א. לפניכם קטע קוד שרץ ללא שגיאות:

```
A a = new A();
A ab = new B();
A ac = new C(0);
B b = new B(1);
B bc = new C(1);
C c = new C(1);
```

הציגו את העצמים שנוצרו (סוג העצם, סוג ההפניה, וערך התכונה num).

ב. לפניכם המשך קטע הקוד (ממוספר) שרץ ללא שגיאות.

ציינו את מספר שורת ההדפסה, וכתבו מה הקוד מדפיס.

```
1.  System.out.println (a.func (b));
2.  System.out.println (a.func (c));
3.  System.out.println (ab.func ((B)ab));
4.  System.out.println (((B)ab).func (ab));
5.  System.out.println (ab.func (10));
6.  System.out.println (a.func (a.func (2)));
7.  System.out.println (ac.func (c));
8.  System.out.println (c.func (ac));
9.  System.out.println (c.func (a));
10. System.out.println (((C)bc).func ((C)bc));
```

/המשך בעמוד 26/

<u>לפותרים בשפת C#</u>

לפניכם המחלקות **A**, **B**, **C** :

```
public class A {
    public int num;

    public A()
    {
        this.num = 1;
    }
    public A (int num)
    {
        this.num = num;
    }
    public int Func (int val)
    {
        return val * 2;
    }
    public virtual int Func (B other)
    {
        return this.num - other.num;
    }
}
```

```
public class B : A {
    public B() : base (2) { }
    public B(int val) : base (val)
    {
        this.num = this.num + Func (val);
    }
    public int Func (A other)
    {
        return this.num + (other.num * 10);
    }
    public override int Func (B other)
    {
        return this.num + other.num;
    }
}
```

```
public class C : B {
    public C (int val)
    {
        this.num = this.num - val;
    }
    public override int Func (B other)
    {
        return this.num * other.num;
    }
    public int Func (C c)
    {
        return this.num * 4;
    }
}
```

(שימו לב: המשך השאלה בעמוד הבא.)

/המשך בעמוד 27/

א. לפניכם קטע קוד שרץ ללא שגיאות:

```
A a = new A();
A ab = new B();
A ac = new C(0);
B b = new B(1);
B bc = new C(1);
C c = new C(1);
```

הציגו את העצמים שנוצרו (סוג העצם, סוג ההפניה, וערך התכונה num).

ב. לפניכם המשך קטע הקוד (ממוספר) שרץ ללא שגיאות.

ציינו את מספר שורת ההדפסה, וכתבו מה הקוד מדפיס.

```
1. Console.WriteLine (a.Func (b));
2. Console.WriteLine (a.Func (c));
3. Console.WriteLine (ab.Func ((B)ab));
4. Console.WriteLine (((B)ab).Func (ab));
5. Console.WriteLine (ab.Func (10));
6. Console.WriteLine (a.Func (a.Func (2)));
7. Console.WriteLine (ac.Func (c));
8. Console.WriteLine (c.Func (ac));
9. Console.WriteLine (c.Func (a));
10. Console.WriteLine (((C)bc).Func ((C)bc));
```