

סוג הבחינה: מבחן מתכונת
מבחן: מתכונת 6
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)

נתונה המחלקה **PrintJob** - עבודת הדפסה שנשלחה למדפסת משותפת במשרד, ולה שתי תכונות:

· **code** - קוד העבודה, מטיפוס שלם (לכל עבודה קוד שונה)

· **pages** - מספר העמודים בעבודה, מטיפוס שלם גדול מ-0

הניחו שלתכונות המחלקה יש פעולות **get/Set** ו- **set/Set**.

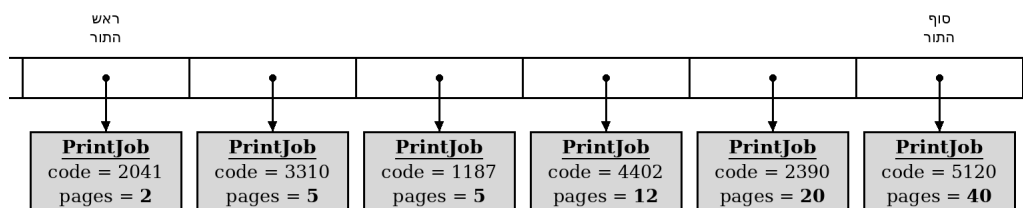
כדי לקצר את זמן ההמתנה, המדפסת מדפיסה את העבודות לפי הסדר שלהן:

עבודה שיש בה פחות עמודים מודפסת מוקדם יותר. כאשר יש יותר מעבודה אחת עם אותו מספר עמודים, העבודה שנשלחה קודם מודפסת מוקדם יותר.

כדי לשמור על סדר ההדפסה נבנתה המחלקה **PrintQueue** - תור הדפסות, ולה תכונה אחת:

· **q** - הפניה לתור, מטיפוס **PrintJob**

דוגמה לתור q:



הניחו שהעבודות שיש בהן אותו מספר עמודים מוצגות בדוגמה לפי סדר שליחתן למדפסת.

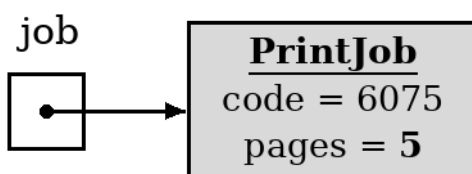
א. ממשו את הפעולה שלפניכם השייכת לממשק המחלקה **PrintQueue**:

Java - `public void addJob (PrintJob job)`

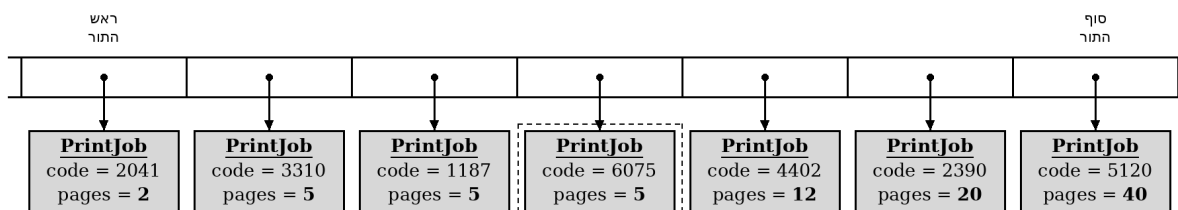
C# - `public void AddJob (PrintJob job)`

הפעולה מקבלת עבודה חדשה - **job** ומכניסה אותה לתור **q** בהתאם לסדר ההדפסה הכתוב לעיל.

לדוגמה: עבור התור המוצג לעיל והעצם שלפניכם:



התור ייראה כך לאחר ההכנסה:



ב. עובד ששלח למדפסת שתי עבודות יכול לבקש לאחד אותן לעבודה אחת.

ממשו את הפעולה שלפניכם השייכת לממשק המחלקה PrintQueue :

Java - public void combine (int code1, int code2)

C# - public void Combine (int code1, int code2)

הפעולה מקבלת קודים של שתי עבודות שונות הנמצאות בתור - code1 ו- code2.

הפעולה תאחד את שתי העבודות לעבודה אחת: לעבודה code1 יתווספו כל העמודים של העבודה

code2, והעבודה code2 תוצא מהתור. העבודה המאוחדת תוצב בתור במקום המתאים לה

(בהתאם למספר העמודים החדש שלה). אם בתור כבר יש עבודות אחרות עם אותו מספר עמודים,

העבודה המאוחדת תוצב אחריהן כאילו נשלחה אחריהן למדפסת.

2. (25 נקודות)

נתונה המחלקה Parcel - חבילה, ולה שתי תכונות:

code - קוד החבילה, מטיפוס מחרוזת.

weight - משקל החבילה בקילוגרמים - מספר הגדול מ-0, מטיפוס שלם.

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

נתונה המחלקה Truck - משאית חלוקה, ולה תכונה אחת:

lst - הפניה לשרשרת חוליות שאינה ריקה, מטיפוס Parcel. כל חוליה בשרשרת מכילה חבילה

המועמסת על המשאית.

הערה: החבילות אינן מסודרות בשרשרת בסדר מסוים, וכל חבילה מופיעה פעם אחת בלבד (ייתכן

שלכמה חבילות יש אותו משקל).

א. ממשו את הפעולה של ממשק המחלקה Truck שלפניכם:

Java - public int removeFrom (int w, int k)

C# - public int RemoveFrom (int w, int k)

הפעולה תשאיר בשרשרת את $k - 1$ החבילות הראשונות שמשקלן w , ותמחק מן השרשרת את כל

שאר החבילות שמשקלן w (כלומר את החבילה ה- k במספר מבין החבילות שמשקלן w , ואת כל

החבילות שמשקלן w שאחריה).

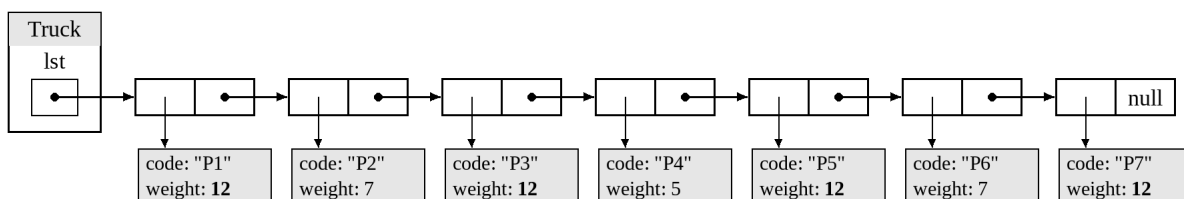
הפעולה תחזיר את כמות החבילות שנמחקו.

הניחו ש- w ו- k גדולים מ-0, ושבתום הפעולה השרשרת לא תהיה ריקה.

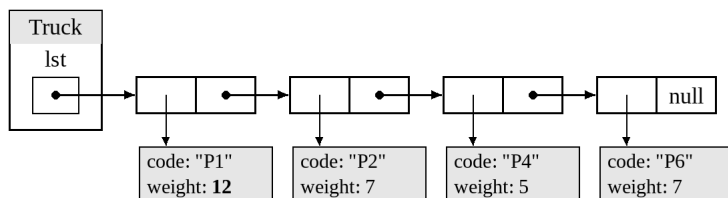
הערה: שאר החבילות בשרשרת יישארו באותו הסדר. אם בשרשרת יש פחות מ- k חבילות שמשקלן

w , השרשרת תישאר ללא שום שינוי והפעולה תחזיר 0.

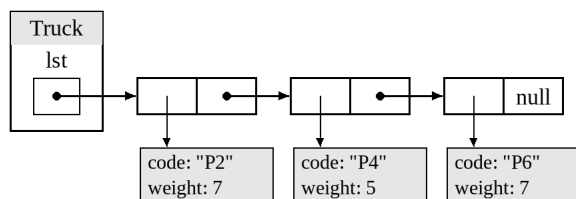
דוגמה: בעבור $w = 12$, $k = 2$ והשרשרת lst שלפניכם:



הפעולה תחזיר 3 והשרשרת תיראה כך בתום הפעולה:



הסבר: בשרשרת יש ארבע חבילות שמשקלן 12. מכיוון ש- $k = 2$, החבילה הראשונה מביניהן ("P1") נשארה בשרשרת, ושלוש האחרות ("P3", "P5", "P7") נמחקו. הפעולה החזירה 3. **דוגמה נוספת:** בעבור אותה השרשרת מן הדוגמה המקורית שלעיל ו- $k = 1$, $w = 12$, הפעולה תחזיר 4 והשרשרת תיראה כך בתום הפעולה:



הסבר: כאשר $k = 1$ לא נשארה בשרשרת אף חבילה שמשקלה 12, ולכן נמחקו כל ארבע החבילות שמשקלן 12 (גם החבילה הראשונה בשרשרת), והפעולה החזירה 4. **ב.** ממשו את הפעולה של ממשק המחלקה **Truck** שלפניכם:

() **Java** - `public int removeRepeats`

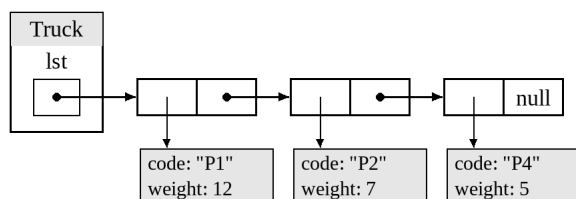
() **C#** - `public int RemoveRepeats`

הפעולה תמחק מן השרשרת חבילות, כך שמכל משקל תישאר בשרשרת רק החבילה הראשונה שזה משקלה. הפעולה תחזיר את סכום המשקלים הכולל של כל החבילות שנמחקו.

הערות:

- החבילות בשרשרת שלא נמחקו יישארו באותו הסדר.
- אם אין בשרשרת שתי חבילות שמשקלן זהה, השרשרת תישאר ללא שינוי והפעולה תחזיר 0.
- אפשר להשתמש בפעולה שבסעיף א.

דוגמה: בעבור אותה השרשרת מן הדוגמה המקורית בסעיף א, הפעולה תחזיר 43, והשרשרת תיראה כך:

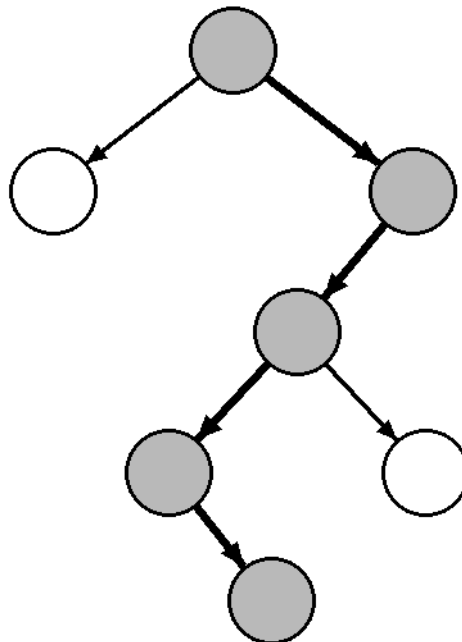


הסבר: מבין ארבע החבילות שמשקלן 12 נשארה רק הראשונה ("P1"), ומבין שתי החבילות שמשקלן 7 נשארה רק הראשונה ("P2"). נמחקו "P3", "P5", "P7", וסכום משקליהן $12 + 12 + 7 = 43$.

3. (25 נקודות)

"עץ K-דליל" הוא עץ בינרי ובו כל מסלול משורש העץ לעלה כלשהו מכיל **לכל היותר** K צמתים שיש להם **בן יחיד** (כלומר צמתים שיש להם בן שמאלי בלבד או בן ימני בלבד).
שימו לב: ב"עץ K-דליל" ייתכן שאין שום מסלול המכיל K צמתים שיש להם בן יחיד (כלומר ייתכן שבכל אחד מן המסלולים יש פחות מ-K צמתים כאלה).

דוגמה:



עץ זה **אינו** "עץ 1-דליל" כי בעץ יש מסלול שבו יותר מצומת אחד שיש לו בן יחיד (במסלול המודגש, שצמתיו צבועים באפור, יש שני צמתים שיש להם בן יחיד).

לעומת זאת, עץ זה הוא "עץ 2-דליל" כי אין בו מסלול שבו יותר משני צמתים שיש להם בן יחיד (ובאותה מידה הוא גם "עץ 3-דליל", וגם "עץ 4-דליל" וכן הלאה).

- א.** סרטטו "עץ 1-דליל" (כלומר שבכל מסלול של העץ יש לכל היותר צומת אחד שיש לו בן יחיד), ובו 10 צמתים, **מקסימום** 4 רמות, **ולפחות** שני צמתים שיש להם בן יחיד.
- ב.** כתבו פעולה חיצונית ששמה isSparseK בשפת Java או IsSparseK בשפת C#, המקבלת עץ בינרי - root מטיפוס שלם שאינו ריק, וערך K שאינו שלילי, ומחזירה true אם העץ הוא "עץ K-דליל", ואם לא - היא מחזירה false.

פרק שני (50 נקודות) - אלגוריתמים

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

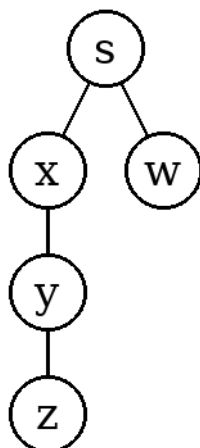
4. (25 נקודות)

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

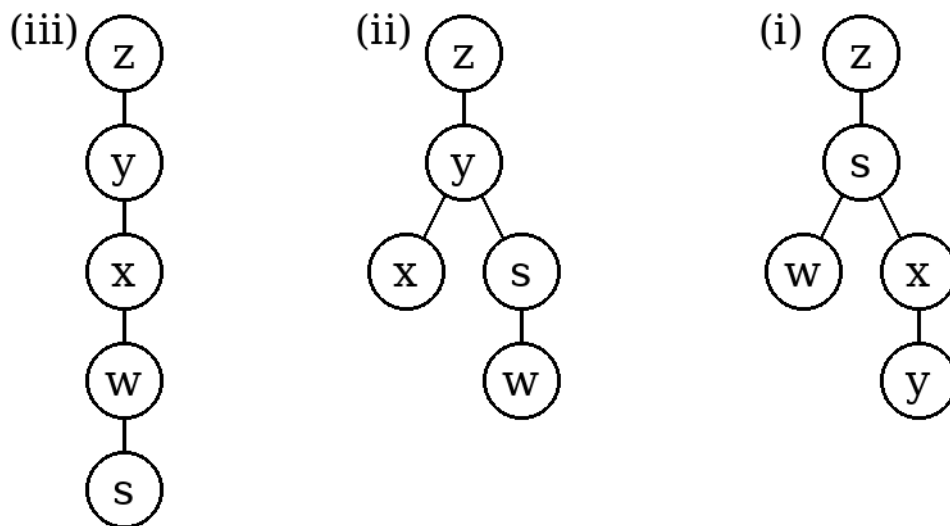
- א.** (1) לפניכם שש טענות 1-6. בחרו **בארבע** מהן, כתבו את מספר הטענה, וציינו בנוגע לכל טענה שבחרתם אם היא נכונה או לא נכונה. אם הטענה נכונה - נמקו מדוע, ואם היא אינה נכונה - הביאו דוגמה נגדית מגרף שיש בו 4 **קודקודים לפחות**.

- (1) נתון גרף **ממושקל** (אי-שלילי), **קשיר ולא מכוון**. בין כל שני קודקודים u ו- v , המסלול שביניהם בעץ הפורש המינימלי של הגרף הוא בהכרח מסלול קל ביותר (קצר ביותר) בין u ל- v בגרף.
- (2) נתון גרף **מכוון** ללא מעגלים ובו n קודקודים (אין בגרף קשתות כפולות). בהכרח יש בגרף **לכל היותר** $n(n-1)/2$ קשתות.
- (3) נתון גרף **מכוון**. אם סריקת DFS מקודקוד s מגלה את **כל** קודקודי הגרף, אז הגרף בהכרח **קשיר היטב** (יש מסלול מכוון מכל קודקוד לכל קודקוד).
- (4) נתון גרף **לא מכוון** ובו n קודקודים, שבו לכל קודקוד יש **לפחות שכן אחד**. בהכרח יש בגרף לכל היותר $n/2$ רכיבי קשירות.
- (5) נתון גרף **מכוון** ללא מעגלים, שבו יש קודקוד **אחד בלבד** שדרגת הכניסה שלו 0. לגרף זה יש בהכרח מיון טופולוגי **אחד בלבד**.
- (6) נתון גרף **לא מכוון**. הרצנו סריקת DFS מקודקוד s , ולא נמצאה בה אף **קשת אחורה**. לכן בגרף אין מעגלים.

ב. נתון גרף **לא מכוון** G ובו 5 קודקודים: s, w, x, y, z . לאחר הרצת DFS מקודקוד התחלה s התקבל העץ שבסרטוט הנתון:

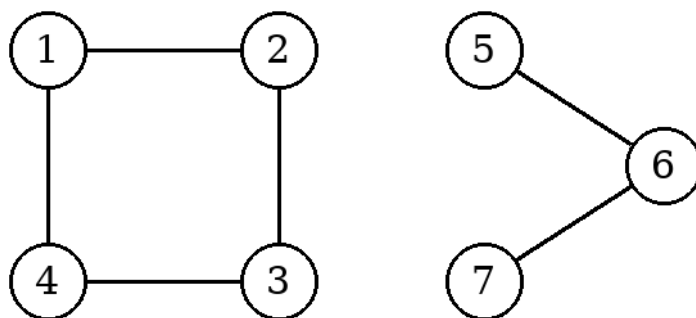


בנוגע לכל אחד משלושת העצים שלפניכם, קבעו אם ייתכן שהוא התקבל מהרצת DFS על אותו הגרף G , מקודקוד התחלה z . נמקו את קביעותיכם.



5. (25 נקודות)

בשאלה זו שני סעיפים, א-ב, שאין ביניהם קשר. יש לענות על שני הסעיפים.
א. להלן גרף $G = (V, E)$ שאינו מכוון:



א. (1) האם הגרף קשיר? אם לא - כמה רכיבי קשירות יש בו? נמקו.

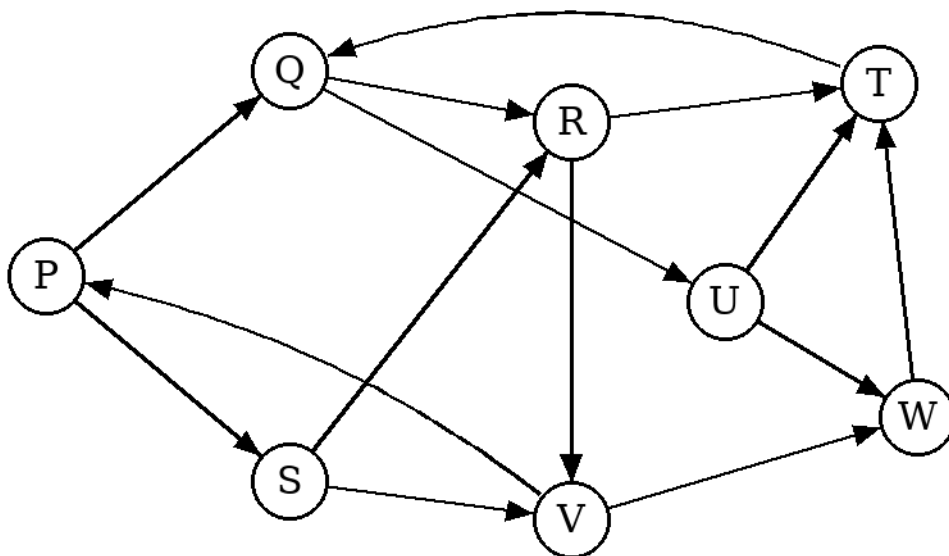
(2) האם הגרף דו-צדדי? אם כן - כתבו חלוקה מתאימה של הקודקודים לשתי קבוצות. נמקו.

(3) האם יש בגרף מעגל? נמקו.

(4) האם הגרף הוא עץ? נמקו.

ב. (1) חברת תחבורה מפעילה קווי אוטובוס ישירים בין שמונה תחנות: P, Q, R, S, T, U, V, W .

בסרטוט שלהלן, חץ מתחנה X לתחנה Y מציינ קו ישיר שנוסע מ-X ל-Y (בכיוון זה בלבד):



(1) סרטטו את רשימת הסמיכויות המייצגת את קווי האוטובוס בין התחנות.

(2) בעקבות עבודות בכביש הוחלט:

• תחנה U - ביטול כל הקווים היוצאים ממנה וכל הקווים הנכנסים אליה

• תחנה S - ביטול הקו היוצא לתחנה R

• תחנה R - ביטול הקו היוצא לתחנה T

(2) סרטטו את רשימת הסמיכויות המתקבלת לאחר השינויים המצוינים לעיל.

(3) I. דני רוצה לנסוע מתחנה מסוימת לתחנה אחרת. הוא רוצה לבדוק אם הוא יכול להגיע

לאותה תחנה, ואם הוא אכן יכול - הוא רוצה למצוא מסלול נסיעה שבו מספר הקווים

שהוא נוסע בהם הוא **הקטן ביותר**.

מהו האלגוריתם שעליו להשתמש בו למטרות אלה? הסבירו.

(3) II. הפעילו את האלגוריתם שכתבתם עליו בסעיף I(3) על הגרף שהתקבל לאחר השינויים (סעיף

(2)), כדי למצוא מסלול נסיעה מתחנה P לתחנה T. כתבו את מסלול הנסיעה שמצאתם.

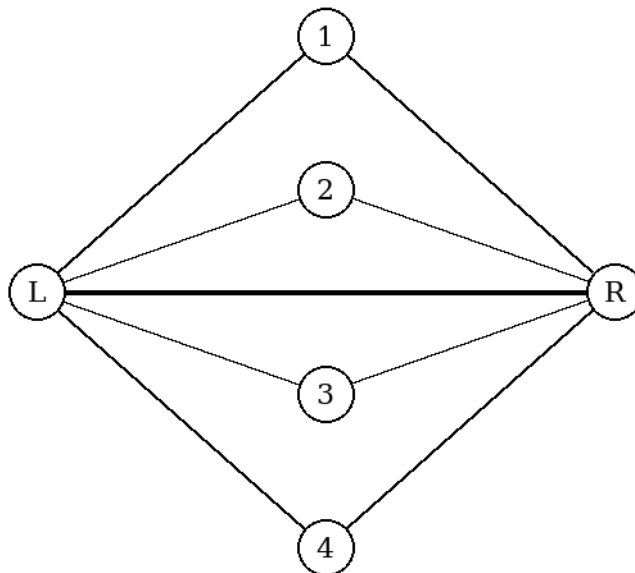
יש לבצע מעקב מפורט בכל שלב בהתאם לאלגוריתם שהפעלתם. (הניחו שהשכנים של כל

תחנה נסרקים לפי סדר אלפביתי.)

6. (25 נקודות)

גרף "גשר" - B_n הוא גרף לא מכוון שמורכב משני "קודקודי קצה" - L ו- R , ומ- n "קודקודי עמוד" שמסומנים 1 עד n (כך שסך הכול יש $n + 2$ קודקודים בגרף). כל "קודקוד עמוד" מחובר בקשת ל- L וגם ל- R , ובנוסף יש קשת אחת שמחברת את L ל- R . מספר "קודקודי העמוד" בגרף גדול מ-1 (כלומר $n > 1$).

דוגמה: הגרף שלפניכם הוא גרף "גשר" - B_4 (הקשת $L-R$ מודגשת).



"קשתות שמאליות" הן הקשתות שמחברות את L ל"קודקודי העמוד" (למשל בדוגמה לעיל, הקשת שמחברת את L לקודקוד 1, הקשת שמחברת את L לקודקוד 2 וכן הלאה).

"קשתות ימניות" הן הקשתות שמחברות את R ל"קודקודי העמוד" (למשל בדוגמה לעיל, הקשת שמחברת את R לקודקוד 3).

"קשת הגשר" היא הקשת שמחברת את L ל- R .

א. (1) נתון גרף "גשר" - B_n .

(1) מה הן הדרגות של "קודקודי הקצה" בגרף, ומהי הדרגה של כל "קודקוד עמוד"? נמקו את תשובתכם.

(2) כמה "קשתות שמאליות", כמה "קשתות ימניות" וכמה קשתות יש בגרף בסך הכול? נמקו את תשובתכם.

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

ב. נתון גרף ממושקל שהוא "גשר" - B_n , שבו לכל "הקשתות השמאליות" משקל 1, לכל "הקשתות הימניות" משקל 2, ול"קשת הגשר" משקל 3. כמה עפ"מים (עצים פורשים מינימליים) שונים יש בגרף? נמקו את תשובתכם.

ג. (1) נתון גרף ממושקל שהוא "גשר" - B_n , שבו לכל "הקשתות השמאליות" ולכל "הקשתות הימניות" משקל w (גדול מ-0), ול"קשת הגשר" משקל g (גדול מ-0).

(1) הציבו ערכים ל- w ו- g , שבעבורם כל מסלול קצר (קל) בין L ל- R יעבור דרך "קודקוד עמוד". נמקו את תשובתכם.

(2) נתון כי $g = 12$. מהו הערך הגדול ביותר של w שבעבורו כל מסלול קצר (קל) בין L ל- R עדיין יעבור דרך "קודקוד עמוד"? נמקו את תשובתכם.

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 $\{a, b, c\}$:

$$L_1 = \{b^m a^n c^m \mid 0 < n < 50 < m\}$$

$$L_2 = \{b^m a^n c^m \mid 0 < m < 50 < n\}$$

$$L_3 = \{b^m a^n c^m \mid 0 < n < 50 < m < 2n\}$$

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 $\{a, b, c\}$:

$$L_4 = L_1 \cap L_2$$

$$L_5 = L_1 \cup L_3$$

$$L_6 = \bar{L}_1 \cap L_2$$

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 $\{a, b, c\}$:

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

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

< - אם המכפלה $n \cdot m$ היא מספר זוגי, w היא מילה שמספר האותיות b בה זוגי (גם מילה ללא b כלל, וגם המילה הריקה).

< - אם המכפלה $n \cdot m$ היא מספר אי-זוגי, w היא מילה שיש בה בדיוק a אחת.

דוגמה למילה השייכת לשפה L ($b^n = bbb$, $c^m = c$, $w = bab$): $L : bbb c bab$

הסבר: $m = 1$, $n = 3$. המכפלה $n \cdot m = 3$ היא מספר אי-זוגי, ו- w היא מילה שיש בה בדיוק a אחת.

בנו אוטומט סופי דטרמיניסטי שאינו מלא המקבל את השפה L .

8. (25 נקודות)

מרחק בין שתי מילים שאורכן זהה, המורכבות אך ורק מן הספרות 0 ו-1, הוא מספר המקומות שבהם הספרות בשתי המילים שונות זו מזו.

דוגמה: המרחק בין $x = 10110$ ובין $y = 11100$ הוא 2, כי הספרות שונות רק במקומות 1 ו-3:

4	3	2	1	0	
0	1	1	0	1	x
0	0	1	1	1	y
	✓		✓		שונות?

כתבו מכונת טיורינג המקבלת בתחילת הסרט קלט של שתי מילים שאורכן זהה, המורכבות אך ורק מן הספרות 0 ו-1. המכונה מחזירה את המרחק בין שתי המילים כמספר אונרי, כלומר כרצף של אחדות שמספרן שווה למרחק.

הנחיות:

- שתי המילים בקלט מופרדות ב-#, ואורך כל אחת מהן 1 לפחות.
- אין צורך לשמור על הקלט (אפשר לשנות את הקלט לסימנים שונים).
- המספר האונרי המוחזר יופיע במקום כלשהו בסרט בין שני סימני \$. אם המרחק הוא 0, יופיעו על הסרט שני סימני \$ צמודים.
- אין צורך לבדוק את תקינות הקלט.

דוגמה:

הסרט לפני ההרצה:

...	Δ	Δ	0	0	1	1	1	#	0	1	1	0	1	1
-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---

הסרט לאחר ההרצה:

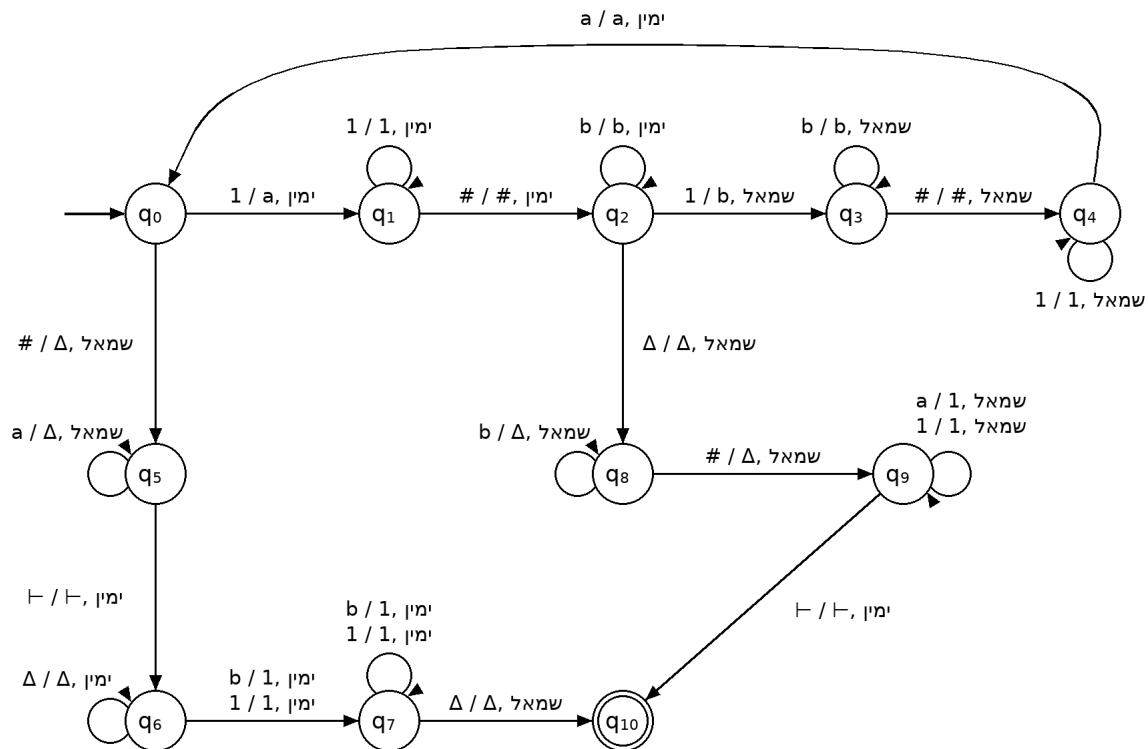
...	\$	1	1	\$	...
-----	----	---	---	----	-----

כתבו מכונת טיורינג המקבלת בתחילת הסרט שתי מילים כמתואר לעיל, ומחזירה בין שני סימני \$ את המרחק ביניהן כמספר אונרי.

(25 נקודות)

9. (25 נקודות)

לפניכם מכונת טיורינג:



על כל מעבר רשום: הסימן הנקרא / הסימן הנכתב, כיוון התזוזה. q_0 הוא המצב ההתחלתי, ו- q_{10} הוא מצב העצירה.

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

א. עקבו אחר המכונה עבור הסרט שלהלן:

...	Δ	Δ	1	1	#	1	$\vdash$
-----	----------	----------	---	---	---	---	----------

כתבו מה יתקבל על הסרט בסיום הפעולה.

ב. כתבו מה יתקבל על הסרט שלהלן בסיום הפעולה:

...	Δ	Δ	1	#	1	1	1	$\vdash$
-----	----------	----------	---	---	---	---	---	----------

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).

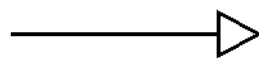
10. (25 נקודות)

נוסח Java

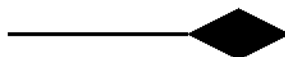
בחברה להשכרת מחסנים "שמור ובטוח" פותחה מערכת ממוחשבת שבה המחלקות האלה:
Lease - חוזה אחסון, **Unit** - יחידת אחסון, **Locker** - תא אחסון, **ColdRoom** - חדר קירור,
Container - מכולה.

להלן פירוט תכונות המחלקות:

- למחלקה חוזה אחסון (Lease) שלוש תכונות: tenant - שם השוכר (מחרוזת), months - מספר חודשי השכירות (מטיפוס שלם),
 - boxes - מספר הארגזים שהשוכר מאחסן (מטיפוס שלם).
 - למחלקה יחידת אחסון (Unit) שתי תכונות: קוד היחידה - code (מחרוזת), חוזה אחסון - lease (Lease).
 - למחלקה תא אחסון (Locker) שלוש תכונות: קוד היחידה - code (מחרוזת), חוזה אחסון - lease (Lease),
 - מספר הקומה שבה נמצא התא - floor (מטיפוס שלם).
 - למחלקה חדר קירור (ColdRoom) שלוש תכונות: קוד היחידה - code (מחרוזת), חוזה אחסון - lease (Lease),
 - הטמפרטורה בחדר - temp (מטיפוס שלם).
 - למחלקה מכולה (Container) שלוש תכונות: קוד היחידה - code (מחרוזת), חוזה אחסון - lease (Lease),
 - מכולה מקורה - roofed (בוליאני. אם המכולה עומדת תחת גג התכונה היא אמת ואם לא, היא שקר).
- 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 Lease:

Java

```
public Lease (String tenant, int months, int boxes)
```

אין צורך לממש את הפעולה.

(1) לפניכם כותרת הפעולה הבונה של המחלקה Unit . הפעולה מקבלת שם שוכר, מספר חודשי שכירות, מספר ארגזים וקוד יחידה.

Java

```
public Unit (String tenant, int months, int boxes, String code)
```

ממשו את הפעולה הבונה.

(2) לפניכם כותרת הפעולה הבונה של המחלקה ColdRoom . הפעולה מקבלת שם שוכר, מספר חודשי שכירות, מספר ארגזים, קוד יחידה וטמפרטורה.

Java

```
public ColdRoom (String tenant, int months, int boxes, String code, int temp)
```

ממשו את הפעולה הבונה.

ג. (1) התעריף הבסיסי שהחברה גובה בעבור השכרת יחידת אחסון - Unit הוא 100 שקלים לכל חודש שכירות ו-4 שקלים לכל ארגז.

מחיר השכירות של תא אחסון (Locker) הוא לפי התעריף הבסיסי.

מחיר השכירות של חדר קירור (ColdRoom) הוא לפי התעריף הבסיסי ונוסף על כך 35 שקלים לכל חודש שכירות.

מחיר השכירות של מכולה (Container) הוא 80% מן התעריף הבסיסי.

ג. הפעולה price מחזירה מספר ממשי השווה לסכום שהשוכר נדרש לשלם בעבור כל אחד מסוגי היחידות שהחברה

משכירה (בהתאם לעצם שזימן את הפעולה).

(1) כתבו את הפעולה price במחלקה Unit (כאמור לעיל, הסכום לתשלום הוא לפי התעריף הבסיסי).

(2) הוסיפו את הפעולה price 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.

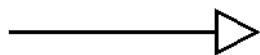
פתרון הכולל שימושים כאלה לא יזוכה בנקודות.

נוסח C#

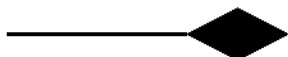
בחברה להשכרת מחסנים "שמור ובטוח" פותחה מערכת ממוחשבת שבה המחלקות האלה:
Lease - חוזה אחסון, **Unit** - יחידת אחסון, **Locker** - תא אחסון, **ColdRoom** - חדר קירור,
Container - מכולה.

להלן פירוט תכונות המחלקות:

- למחלקה חוזה אחסון (Lease) שלוש תכונות: **tenant** - שם השוכר (מחרוזת), **months** - מספר חודשי השכירות (מטיפוס שלם),
 - boxes** - מספר הארגזים שהשוכר מאחסן (מטיפוס שלם).
 - למחלקה יחידת אחסון (Unit) שתי תכונות: קוד היחידה - **code** (מחרוזת), חוזה אחסון - **lease** (Lease).
 - למחלקה תא אחסון (Locker) שלוש תכונות: קוד היחידה - **code** (מחרוזת), חוזה אחסון - **lease** (Lease),
 - מספר הקומה שבה נמצא התא - **floor** (מטיפוס שלם).
 - למחלקה חדר קירור (ColdRoom) שלוש תכונות: קוד היחידה - **code** (מחרוזת), חוזה אחסון - **lease** (Lease),
 - הטמפרטורה בחדר - **temp** (מטיפוס שלם).
 - למחלקה מכולה (Container) שלוש תכונות: קוד היחידה - **code** (מחרוזת), חוזה אחסון - **lease** (Lease),
 - מכולה מקורה - **roofed** (בוליאני. אם המכולה עומדת תחת גג התכונה היא אמת ואם לא, היא שקר).
- 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 **Lease**:

C#

```
public Lease (string tenant, int months, int boxes)
```

אין צורך לממש את הפעולה.

- (1) לפניכם כותרת הפעולה הבונה של המחלקה **Unit**. הפעולה מקבלת שם שוכר, מספר חודשי שכירות, מספר ארגזים וקוד יחידה.

C#

```
public Unit (string tenant, int months, int boxes, string code)
```

ממשו את הפעולה הבונה.

- (2) לפניכם כותרת הפעולה הבונה של המחלקה ColdRoom. הפעולה מקבלת שם שוכר, מספר חודשי שכירות, מספר ארגזים, קוד יחידה וטמפרטורה.

C#

```
public ColdRoom (string tenant, int months, int boxes, string code, int temp)
```

ממשו את הפעולה הבונה.

- ג. (1) התעריף הבסיסי שהחברה גובה בעבור השכרת יחידת אחסון - Unit הוא 100 שקלים לכל חודש שכירות ו- 4 שקלים לכל ארגז.
- מחיר השכירות של תא אחסון (Locker) הוא לפי התעריף הבסיסי.
- מחיר השכירות של חדר קירור (ColdRoom) הוא לפי התעריף הבסיסי ונוסף על כך 35 שקלים לכל חודש שכירות.
- מחיר השכירות של מכולה (Container) הוא 80% מן התעריף הבסיסי.
- ג. הפעולה Price מחזירה מספר ממשי השווה לסכום שהשוכר נדרש לשלם בעבור כל אחד מסוגי היחידות שהחברה משכירה (בהתאם לעצם שזימן את הפעולה).
- (1) כתבו את הפעולה Price במחלקה Unit (כאמור לעיל, הסכום לתשלום הוא לפי התעריף הבסיסי).
- (2) הוסיפו את הפעולה Price 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 כותרות המחלקות TT, UU והתכונות שלהן.
הפעולות הבונות של המחלקות מסומנות ב- * (תיתכן יותר מפעולה בונה אחת למחלקה).

Java

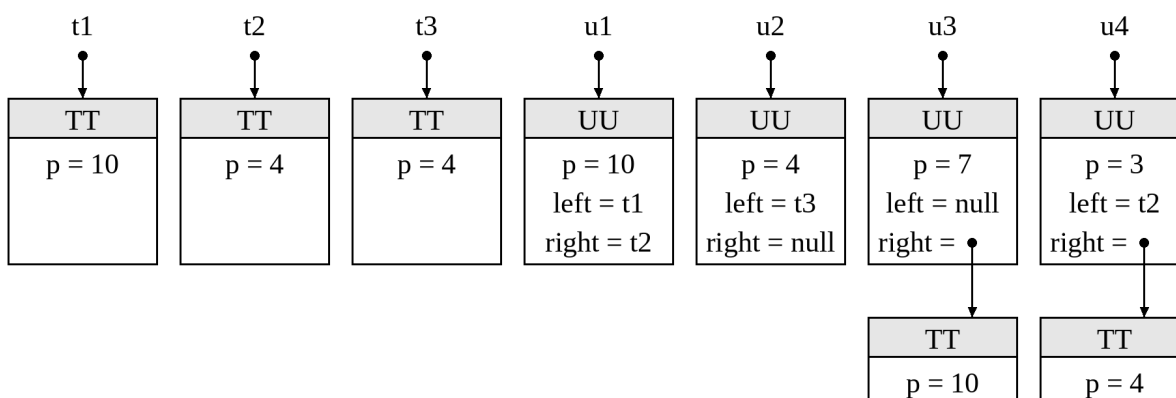
```
public class TT
{
    private int p;
    ***
}
public class UU extends TT
{
    private TT left;
    private TT right;
    ***
}
```

הערה: שימו לב - לשתי המחלקות אין פעולות get ו- set.
לפניכם המחלקה Test, הכוללת פעולה ראשית:

Java

```
public class Test {
    public static void main(String[] args) {
        TT t1 = new TT ();
        TT t2 = new TT (4);
        TT t3 = new TT (t2);
        UU u1 = new UU (t1, t2);
        UU u2 = new UU (t3);
        UU u3 = new UU (7);
        UU u4 = new UU (3, t2);
    }
}
```

לפניכם תרשים של עצמים שנוצרו בעקבות הרצת קטע הקוד.
כתבו במחלקות TT ו- UU את הפעולות הבונות הנדרשות כדי לקבל את העצמים שבתרשים.
ציינו בעבור כל אחד מן העצמים שנוצרו את הפעולה הבונה המתאימה לו.
הערה: אין להוסיף פעולות שאינן בונות במחלקות TT ו- UU. פתרון המוסיף פעולות שאינן בונות לא יזוכה בנקודות.



נוסח C#

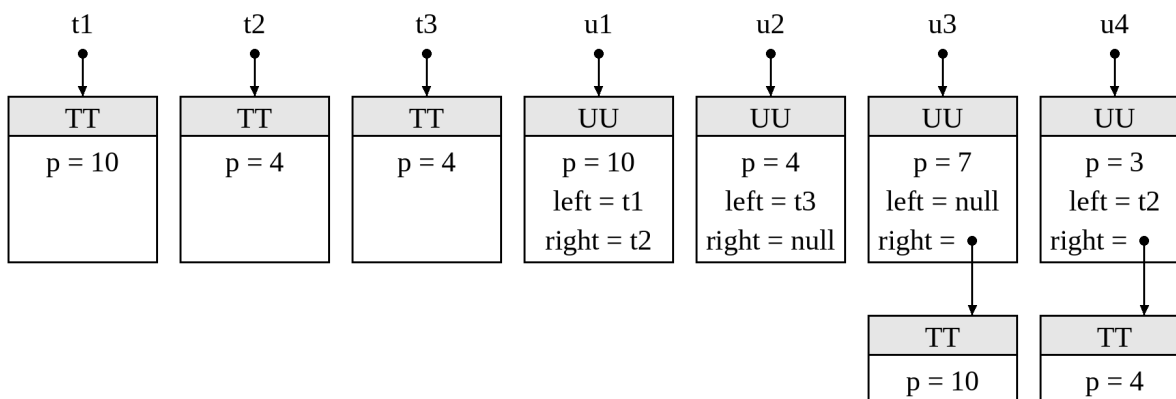
לפניכם כותרות המחלקות TT, UU והתכונות שלהן.
הפעולות הבונות של המחלקות מסומנות ב- * (תיתכן יותר מפעולה בונה אחת למחלקה).

```
C#
public class TT
{
    private int p;
    ***
}
public class UU : TT
{
    private TT left;
    private TT right;
    ***
}
```

הערה: שימו לב - לשתי המחלקות אין פעולות Get ו- Set.
לפניכם המחלקה Test, הכוללת פעולה ראשית:

```
C#
public class Test {
    public static void Main(string[] args) {
        TT t1 = new TT ();
        TT t2 = new TT (4);
        TT t3 = new TT (t2);
        UU u1 = new UU (t1, t2);
        UU u2 = new UU (t3);
        UU u3 = new UU (7);
        UU u4 = new UU (3, t2);
    }
}
```

לפניכם תרשים של עצמים שנוצרו בעקבות הרצת קטע הקוד.
כתבו במחלקות TT ו- UU את הפעולות הבונות הנדרשות כדי לקבל את העצמים שבתרשים.
ציינו בעבור כל אחד מן העצמים שנוצרו את הפעולה הבונה המתאימה לו.
הערה: אין להוסיף פעולות שאינן בונות במחלקות TT ו- UU. פתרון המוסיף פעולות שאינן בונות לא יזוכה בנקודות.



12. (25 נקודות)

נוסח Java

לפניכם המחלקות X, Y, Z :

Java

```
public class X {
    public int val;

    public X() {
        this.val = 3;
    }
    public X(int val) {
        this.val = val;
    }
    public int calc(int k) {
        return k + this.val;
    }
    public int calc(Y other) {
        return this.val * other.val;
    }
}
```

Java

```
public class Y extends X {
    public Y() {
        super(4);
    }
    public Y(int k) {
        this.val = this.val + calc(k);
    }
    public int calc(X other) {
        return other.val - this.val;
    }
    public int calc(Y other) {
        return this.val + 2 * other.val;
    }
}
```

Java

```
public class Z extends Y {
    public Z(int k) {
        super(k);
        this.val = this.val * 2;
    }
    public int calc(Y other) {
        return other.val - 1;
    }
    public int calc(Z other) {
        return this.val + other.val + 10;
    }
}
```

א. לפניכם קטע קוד שרץ ללא שגיאות:

Java

```
X x = new X();
X xy = new Y();
X xz = new Z(0);
```

```
Y y = new Y(2);
Y yz = new Z(1);
Z z = new Z(-3);
```

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

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

ציינו את מספר שורת ההדפסה, וכתבו מה הקוד מדפיס.

Java

```
1. System.out.println(x.calc(y));
2. System.out.println(xy.calc(y));
3. System.out.println(y.calc(x));
4. System.out.println(xz.calc(z));
5. System.out.println(((Y)xz).calc(xy));
6. System.out.println(z.calc(5));
7. System.out.println(xy.calc(xy.calc(2)));
8. System.out.println(yz.calc(x));
9. System.out.println(xy.calc(z));
10. System.out.println(z.calc(z));
```

נוסח #C

לפניכם המחלקות X, Y, Z :

C#

```

public class X {
    public int val;

    public X()
    {
        this.val = 3;
    }
    public X(int val)
    {
        this.val = val;
    }
    public int Calc(int k)
    {
        return k + this.val;
    }
    public virtual int Calc(Y other)
    {
        return this.val * other.val;
    }
}

```

C#

```

public class Y : X {
    public Y() : base(4) { }
    public Y(int k)
    {
        this.val = this.val + Calc(k);
    }
    public int Calc(X other)
    {
        return other.val - this.val;
    }
    public override int Calc(Y other)
    {
        return this.val + 2 * other.val;
    }
}

```

C#

```

public class Z : Y {
    public Z(int k) : base(k)
    {
        this.val = this.val * 2;
    }
    public override int Calc(Y other)
    {
        return other.val - 1;
    }
    public int Calc(Z other)
    {
        return this.val + other.val + 10;
    }
}

```

א. לפניכם קטע קוד שרץ ללא שגיאות:

C#

```
X x = new X();
X xy = new Y();
X xz = new Z(0);
Y y = new Y(2);
Y yz = new Z(1);
Z z = new Z(-3);
```

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

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

ציינו את מספר שורת ההדפסה, וכתבו מה הקוד מדפיס.

C#

```
1. Console.WriteLine(x.Calc(y));
2. Console.WriteLine(xy.Calc(y));
3. Console.WriteLine(y.Calc(x));
4. Console.WriteLine(xz.Calc(z));
5. Console.WriteLine(((Y)xz).Calc(xy));
6. Console.WriteLine(z.Calc(5));
7. Console.WriteLine(xy.Calc(xy.Calc(2)));
8. Console.WriteLine(yz.Calc(x));
9. Console.WriteLine(xy.Calc(z));
10. Console.WriteLine(z.Calc(z));
```

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