<u>הערכה חלופית – מדעי המחשב</u>

הערות כלליות:

עבודה זו מחליפה את הבגרות שהייתה אמורה להתקיים ומשפיעה באופן ישיר על הציון הסופי שלכם במדעי המחשב. אנא קראו היטב את כל ההוראות לפני שאתם מתחילים לעבוד. תיקיית העבודה מכילה שני קבצים: הסבר מקיף על העבודה(זהו קובץ זה) הכולל את העבודה עצמה, וקובץ בויזואל סטודיו בו תכתבו את הקוד שלכם. עליכם "לחלץ" את הקבצים של העבודה כדי שתוכלו לפתוח את קבצי הקוד ללא בעיות. שימו לב, לא ניתן לכתוב את התשובות במקום אחר שאינו הקובץ שמסופק לכם. למרות זאת, **במידת הצורך ועל פי ראות עינכם - ניתן להוסיף פעולות פרטיות משלכם בקוד, יש להוסיף אותן באותה מחלקה בה אתם פותרים את השאלה המתאימה לה.** עבור כל אחד מהסעיפים בעבודה קיים מקום אחד ויחיד בקובץ המיועד לו, השלד עבור כל אחת מהפעולות שעליכם ליצור כבר מוכן(כותרת הפעולה), עליכם רק להשלים את קוד הפעולה עצמה(לאחר שמחקתם את השורה הנוכחית שבתוך הפעולה. שורה זו "זורקת חריגה" ומטרתה לשמש כתחליף זמני לקוד שאתם תכתבו). **אין לשנות דבר בחתימת הפעולות, חוץ משאלה 8, בה ניתן יהיה להוסיף/לערוך את כותרת הפעולה לשם שמירת עקרונות תכנות מונחה עצמים. כמו כן אין לשנות דבר בקבצים Node, Stack, BinNode. בפרט, אסור להוסיף למחלקות אלו פעולות עזר.** העבודה מכילה בין היתר שאלות לא פשוטות, לכן מומלץ להתחיל לעבוד עליה בהקדם. מומלץ לבצע שמירה של השינויים שאתם מבצעים בתדירות גבוהה. העבודה תיבדק אוטומטית על ידי בדיקות אוטומטיות שהוכנו מראש עבור כל סעיף.

אותנטיות:

העבודה ביחידים בלבד. אין לעבוד בזוגות, בשלישיות וכדומה, ללא יוצא מן הכלל. שימו לב, את העבודות של כלל התלמידים(משתי קבוצות הלימוד) נריץ בתוכנה מתקדמת לזיהוי העתקות, שני תלמידים(או יותר) שהקוד שלהם מועתק יקבלו **אפס**. הזהרה מראש: התוכנה מזהה קוד גם עם "שינויים קלים"(כגון שינוי שם משתנים, החלפת סדר השורות וכדומה), ולכן גם במצב זה המעתיק והמועתק יקבלו אפס בעבודה. **אל תגידו שלא הזהרנו**.

ציון:

כפי שנאמר לעיל, העבודה תיבדק באמצעות בדיקות אוטומטיות ובדיקות פרונטליות באופן הבא: רוב העבודה תיבדק באמצעות בדיקות אוטומטיות. כלומר, בדיקות של הקוד שהוכנו מראש. בדיקות אלו יוודאו את נכונות הפעולות שמימשתם, אם מימשתם את הפעולות בצורה נכונה הבדיקות צריכות להצליח, במידה ויש טעויות בפעולות הבדיקות עלולות להיכשל. הציון ייקבע לפי מספר הבדיקות הכולל שהצליחו. שימו לב, במידה ולא תשנו בפעולה מסוימת את התוכן המקורי שלה(זריקת השגיאה), הבדיקות עבור פעולה זו ייכשלו מיידית. הבדיקות האוטומטיות של העבודה יהוו 60% מהציון הסופי עליה. שימו לב כי באחריותכם לוודא שאין שגיאות קומפילציה בקוד. במידה ויהיו כאלו הדבר יגרור הורדה נוספת בציון. במידה ולא הצלחתם סעיף כלשהו, עדיף שתשאירו את השורה שזורקת שגיאה וזאת על מנת שלא ירד לכם ניקוד בעקבות שגיאות קומפילציה בקוד. בנוסף, נבצע בדיקות פרונטליות- כלומר שיחה בזום תלמיד עם מורה אחד על אחד, ובבדיקה זו נשאל אתכם שאלות הנוגעות לעבודה שלכם, היוודאו הבנה של החומר ואוטנטיות. נוכל לשאול שאלות הנוגעות לכל חלק בעבודה, לכן חשוב שתבינו בצורה מוחלטת את העבודה ותפקיד כל חלק בה, כלומר **חשוב שתבצעו אותה בעצמכם.** נוכל לשאול אתכם גם מה צריך לשנות בחלקים מסוימים בקוד עקב שינויים מסוימים שנרצה לבצע, זאת שוב על מנת לוודא הבנה של התשובות של עצמכם. הבדיקות הפרונטליות של העבודה יהוו 40% מהציון הסופי שלה. תאריך ושעה לבדיקות הפרונטליות עבור כל תלמיד ייקבע בהמשך.

הגשה:

תאריך אחרון להגשה הינו 9.5, יום שבת בשעה 11:59 בלילה. לא יתקבלו איחורים, ולא ינתנו הארכות(בשל לחץ למתן ציונים סופיים). אנא נהלו את הזמן שלכם בהתאם. עליכם לכווץ את הקובץ המכיל את כל הפרויקט(קובץ הקוד). שנו את השם שם הקובץ המכווץ, לשמכם המלא באנגלית. נא לפעול על פי ההוראות, וזאת כדי למנוע בלבולים. יש להגיש את הקובץ המכווץ.

בהצלחה לכולם.

הוראות:

יש לפתור את כל השאלות: הניקוד המופיע הינו לחלק של הבדיקות האוטומטית בלבד והוא יתחלק באופן יחסי לאותה חלוקה שהייתה בבגרות.

שימו לב: הניקוד לחלק של הבדיקה הפרונטלית אינו חייב בהכרח לעמוד באותה חלוקה.

חלק א' – חימום (9 נק'):

1. (4 נק') נתון מערך חד מימדי בגודל 62 המכיל מספרים שלמים וחיוביים. כתבו קטע תוכנית שיחשב את סכום כל המספרים התלת ספרתיים במערך, וידפיס סכום זה. בנוסף על התוכנית למנות כמה מספרים גדולים מ-669, ולהחזיר מספר זה.
כתבו פתרון לשאלה זו בקובץ Program.cs בפעולה Q1
2. (5 נק') כתבו פעולה המקבלת מערך מטיפוס בוליאני. הפעולה תבדוק האם המערך הינו מערך בוליאני מתהפך, ותחזיר אמת או שקר בהתאם. הגדרה- מערך בוליאני מתהפך הוא מערך שערכיו הם אמת ושקר לסירוגין. לדוגמא, המערך הבא הוא מערך בוליאני מתהפך: אמת, שקר, אמת, שקר, אמת(האיבר הראשון יכול להיות אמת או שקר), ואילו המערך הבוליאני הבא אינו מתהפך: אמת, שקר, אמת, אמת, שקר.
כתבו פתרון לשאלה זו בקובץ Program.cs בפעולה Q2

חלק ב' – מחלקות (12 נק'):

3. (6 נק') במערכת המחשוב של חנות רהיטים יש מחלקה בשם Sofa שיש לה שלוש תכונות:

model – דגם הספה
country – ארץ הייצור של הספר
price – מחיר הספה כמספר ממשי

א. כתבו בקובץ Sofa.cs את כותרות המחלקה Sofa ואת התכונות שלה, כמו גם פעולה בונה המקבלת ערכים עבור דגם הספה ופעולות Get, ארץ הייצור ומחיר הספה.

כחלק משיפוץ הנערך בביתכם עליכם לקנות 3 ספות חדשות. החלטתם כי אתם מעוניינים לקנות 3 ספות שונות מ-3 דגמים שונים. התקציב שהוקצה לקניית הספות הינו מוגבל, ואתם מחפשים ספות בתקציב הזה בדיוק.

ב. כתבו פעולה חיצונית ThreeSofas המקבלת מערך של עצמים שונים מסוג sofa, ומספר – budget כמספר ממשי.

על הפעולה למצוא ולהחזיר מערך ספות בגודל 3 כך שמערך זה מכיל את שלושת הספות שסכום המחירים שלהם הוא בדיוק budget. במידה ולא נמצא אחד כזה, החזירו null.
כתבו פתרון לשאלה זו בקובץ Program.cs בפעולה ThreeSofas

4. (6 נק') כינרת המאמנת מבצעת מעקב הדוק אחר המשקולות בחדר-הכושר. המשקולות של כנרת מאפשרות כיוונון על-פי תכנון המתאמן או המתאמנת, כך שניתן להוסיף משקלים מצד ימין ו/או מצד שמאל של המשקולת. משקולת לדוגמא:

כינרת תכננה את המחלקה HandWeight (משקולת יד) ולה שני שדות: השדה leftWeight מסוג double המייצג את המשקל הצבור בצד השמאלי של המשקולת ואת השדה rightWeight מסוג double המייצג את המשקל הצבור בצד הימני של המשקולת .

נתון קטע הקוד הבא:

```
HandWeight hw1 = new HandWeight();
HandWeight hw2 = new HandWeight(2.5);
HandWeight hw3 = new HandWeight(3.14, 2);
Console.WriteLine("hw1: " + hw1.ToString());
Console.WriteLine("hw2: " + hw2.ToString());
Console.WriteLine("hw3: " + hw3.ToString());
Console.WriteLine("Is hw1 balanced? " + hw1.IsBalanced());
Console.WriteLine("Is hw2 balanced?: " + hw2.IsBalanced());
Console.WriteLine("Is hw3 balanced?: " + hw3.IsBalanced());
HandWeight.MoveWeight(hw1, hw2);
Console.WriteLine("hw1: " + hw1.ToString());
Console.WriteLine("hw2: " + hw2.ToString());
```

עבורו **נשאף** שיתקבל הפלט הבא:

```
hw1: <Left: 0, Right: 0>
hw2: <Left: 2.5, Right: 2.5>
hw3: <Left: 3.14, Right: 2>
Is hw1 balanced? true
Is hw2 balanced?: true
Is hw3 balanced?: false
hw1: <Left: 2.5, Right: 2.5>
hw2: <Left: 0, Right: 0>
```

יש לענות על הסעיפים הבאים כך שבהפעלת הקוד הנתון יתקבל הפלט אותו אנו שואפים. **יש להשלים את הקוד לכלל הסעיפים בקובץ HandWeight.cs**

a. כתבו את גוף בנאי המחלקה
```
public HandWeight(double leftWeight, double rightWeight)
```
b. כתבו את גוף בנאי המחלקה
```
public HandWeight(double weight)
```
c. כתבו את גוף בנאי המחלקה
```
public HandWeight()
```
d. כתבו את גוף השיטה שחתימתה `public bool IsBalanced()`
e. כתבו את גוף השיטה הסטטית שחתימתה:
```
public static void MoveWeight(HandWeight handWeight1, HandWeight
handWeight2)
```
f. השלימו את גוף השיטה ()ToString.

חלק ג' – מבני נתונים (24 נק')

5. (8 נק') נתון הממשק של המחלקה TwinList:

TwinList(Node<int> chain)	פעולה בונה המקבלת רשימה chain
Node<int> GetEven()	פעולה המחזירה רשימה המכילה את האיברים במקומות הזוגיים שברשימה chain
Node<int> GetOdd	פעולה המחזירה רשימה המכילה את האיברים במקומות האי-זוגיים שברשימה chain
Node<int> GetChain()	פעולה המחזירה את הרשימה chain
Node<int> SwitchChain()	פעולה המחזירה את השרשרת הנוצרת מהחלפת התאים האי-זוגיים בתאים הזוגיים שברשימה chain. כך לדוגמה עבור הרשימה: $12 \rightarrow 21 \rightarrow 300 \rightarrow 41 \rightarrow 50$ תוחזר הרשימה: $21 \rightarrow 12 \rightarrow 41 \rightarrow 300 \rightarrow 50$ הערה: במקרה של מספר חוליות אי זוגי, האיבר האחרון ברשימה ישאר במקומו

ממשו את המחלקה כרצונכם בקובץ TwinList.cs. שימו לב: המקום הראשון ברשימה הוא המקום ה-0.

6. (8 נק') מחסנית חשבונית הינה מחסנית של תווים Stack<char> דרכה ניתן לבנות תרגיל חשבוני. מבנה תקין של המחסנית ייראה כך:

המחסנית מכילה אך ורק תווים מסוג ספרות, וכן את התווים '+', '-', '*', '/', '='. תחילה נשלוף ספרות מהמחסנית (קיימת לפחות ספרה אחת במחסנית במבנה תקין) עד אשר נקבל את אחד מהסימנים '+', '-', '*', '/', המייצגים את הפעולה החשבונית המתאימה. הספרות ששלפנו עד לאותה נקודה ירכיבו מספר, כך שספרת האחדות תהיה האחרונה שנשלפה לפני התו של הפעולה החשבונית, ספרת העשות היא זו שלפניה, וכן הלאה עד לספרה הראשונה. לאחר מכן נשלוף מהמחסנית שוב פעם ספרות (גם כאן, במבנה תקין תהיה לפחות ספרה אחת) עד אשר נקבל את התו '=', או עד אשר המחסנית תתרוקן. גם כאן נרכיב מהספרות שיצאו מספר באותו האופן, ולאחר שנקבל את התו '=' או שנרוקן את המחסנית, נבצע את התרגיל החשבוני.

'7'
'2'
'5'
'-'
'2'
'4'
'6'
'='
'5'
'3'
'1'

לדוגמה, במחסנית החשבונית הבאה המוצגת מימין, נשלוף את '5', '2', '7' ונרכיב ממנו את המספר 725, לאחר מכן נשלוף את התו '-', ולאחריו את התווים '2', '4', '6', המרכיבים לנו את המספר 246. התרגיל שיבוצע הוא התרגיל 725-246. לא נתייחס לתווים הנמצאים לאחר התו '=', במידה וקיימים כאלו.

א. כתבו פעולה חיצונית int StringToInt(string num) בקובץ Program.cs המקבלת כקלט מחרוזת המורכבת מספרות בלבד, ותחזיר את המספר המיוצג במחרוזת בתור מספר שלם. לדוגמה, עבור המחרוזת "567", הפעולה תחזיר את המספר 567. על הפעולה להיות רקורסיבית, או להיעזר בפעולת עזר רקורסיבית. אין צורך לבדוק את תקינות הקלט. **אין להשתמש בפעולה int.Parse().**

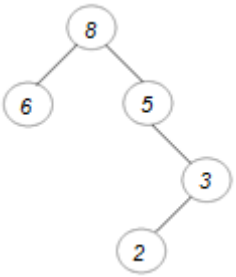
ב. כתבו פעולה חיצונית int CalcStr(string num1, string num2, char op) בקובץ Program.cs המקבלת כקלט 2 מחרוזות ותו, שתי המחרוזות הראשונות מייצגות מספרים

במחרוזות, והתו op מייצג את פעולת החשבון '+', '-', '*' או '/'. הפעולה תחשב את תרגיל החשבון בסדר הבא <num1> <op> <num2>.
לדוגמה: עבור op='-', num2='327', num1='567', הפעולה תחשב 567-327 ותחזיר את תוצאת החיסור 240.
אין צורך לבדוק את תקינות הקלט. חובה להיעזר בפעולה של סעיף א'. במקרה של חילוק, יש להחזיר את תוצאת החילוק השלמה.

ג. השלימו את הפעולה החיצונית int MathStack(Stack<char> st) בקובץ Program.cs המקבלת מחסנית חשבונית במבנה תקין, מחשבת את התרגיל החשבוני הרלוונטי, ומחזירה את תוצאתו. אין צורך לבדוק את תקינות המחסנית. ניתן להיעזר בפעולה של סעיף ב' בלבד (כלומר אין לקרוא מפעולה זו ישירות לפעולה של סעיף א'). אין צורך לשמור על תוכנה המקורי של המחסנית.

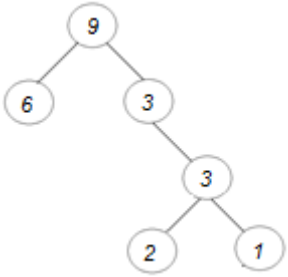
7. (8 נק')

א. עץ בינארי של מספרים שלמים יקרא "סטטי" אם סכום העלים בו שווה לשורש. כך העץ הבא הוא עץ-סטטי:



כתבו פעולה חיצונית בקובץ Program.cs המקבלת עץ-בינארי של שלמים ומחזירה TRUE במקרה והוא עץ סטטי ו FALSE אחרת. במידת הצורך ניתן לכתוב פעולת עזר

ב. עץ בינארי של מספרים שלמים יקרא "סטטי-מאד" אם כל תת-עץ בו הוא סטטי. כך העץ הבא הוא עץ-סטטי-מאד:



כתבו פעולה חיצונית בקובץ Program.cs המקבלת עץ-בינארי של שלמים ומחזירה TRUE במקרה והוא עץ סטטי-מאד ו FALSE אחרת.

חלק ד' – מונחה עצמים:

8. (15 נק') בעולם שלנו ישנם 3 סוגי כלי רכב: אופניים חשמליים, רכבים פרטיים ואוטובוסים.
כלי הרכב מתחלקים באופן המתואר בטבלה לעיל:
יש להשלים בקבצים Vehicle, Bus, Car, Bicycle את כל המתודות כך שימלאו אחר ההוראות המפורטות. תוך שמירה על עקרונות תכנות מונחה עצמים בצורה המיטבית ביותר כפי שלמדנו בשיעור. ניקוד מלא יינתן למימוש מלא ומיטבי של העקרונות הללו.
בכל קובץ הוכנסו כלל השיטות המתוארות בטבלה ולא הוכנסו כלל יחסי הורשה בקוד, את כל אלו עליכם להוסיף בעצמכם, ולמחוק מהמחלקות שיטות שאינן רלוונטיות בעקבות יחסי ההורשה.
תזכורת: רק בשאלה זו מותר לגעת בחתימות הקבצים על מנת להגיע למימוש מיטבי, וזאת רק ע"פ ההוראות המפורטות לעיל. בכל מקרה אין לשנות את שמות השיטות, שכן הדבר יוביל לשגיאות קומפילציה בבדיקות שלנו.
שימו לב כי הטבלה פרוסה על פני שני עמודים, וכן תחילת עמוד 7 שייכת עדיין למחלקה של רכב פרטי

שם כלי רכב	תכונות	שיטות	הסבר שיטות
אופניים חשמליים	לוחית רישוי	Bicycle	פעולה בונה המאתחלת לפי הערכים המתקבלים וקובעת את האחוז הנוכחי להיות 50
	מספר מושבים	GetLicensePlate	פעולה המחזירה את מספר לוחית הרישוי
	מספר גלגלים	GetNumOfSeats	פעולה המחזירה את מספר המושבים
	בעלים	GetNumOfWheels	פעולה המחזירה את מספר הגלגלים
	אחוז סוללה נוכחי	SellVehicle	פעולה המוכרת את הרכב לידי בעלים חדש בשם המתקבל כפרמטר
		GetOwner	פעולה המחזירה מחרוזת עם שם הבעלים
		Ride	פעולה המתארת רכיבה של מספר הקילומטרים במספר המתקבל כפרמטר, או עד שנגמרת הסוללה. יש לעדכן את אחוז הסוללה כך שעל כל קילומטר ירדו 5 אחוזים, עד למינימום של 0 אחוז
		Charge	פעולה המתארת טעינה של הסוללה במשך מספר הדקות המתקבל כפרמטר, יש לעדכן את אחוז הסוללה כך שעל כל דקת טעינה יתמלאו 3 אחוזים עד לסוף הטעינה או עד לקיבולת מקסימלית של 100 אחוז
		ToString	פעולה המחזירה את תכונות הרכב
רכב פרטי	לוחית רישוי	Car	פעולה בונה המאתחלת לפי הערכים המתקבלים וקובעת את כמות הדלק ההתחלתית להיות 0
	מספר מושבים	GetLicensePlate	פעולה המחזירה את מספר לוחית הרישוי
	מספר גלגלים	GetNumOfSeats	פעולה המחזירה את מספר המושבים

	בעלים	GetNumOfWheels	פעולה המחזירה את מספר הגלגלים
	כמות דלק מקסימלית	SellVehicle	פעולה המוכרת את הרכב לידי בעלים חדש בשם המתקבל כפרמטר
	כמות דלק נוכחית	GetOwner	פעולה המחזירה מחרוזת עם שם הבעלים
		SetAmount	פעולה הקובעת את הכמות הנוכחית של הדלק, כאשר עליה להיות בין 0 לבין כמות הדלק המקסימלית. אם מתקבל מספר מחוץ לטווח אין לשנות את הכמות
		GetAmount	פעולה המחזירה את כמות הדלק הנוכחית ברכב
		GetCapacity	פעולה המחזירה את תכולת מיכל הדלק
		FuelUp	פעולה המתדלקת את הרכב ומחזירה את המחיר שעלה התדלוק ע"פ הנוסחה: (כמות מקסימלית פחות הכמות הנוכחית) כפול המחיר לליטר שהינו 4.9. בסוף הפעולה צריך להתקיים כי amount = capacity
		ToString	פעולה המחזירה את תכונות הרכב
אוטובוס	לוחית רישוי	Bus	פעולה בונה המאתחלת לפי הערכים המתקבלים וקובעת את כמות הדלק ההתחלתית להיות 0
	מספר מושבים	GetLicensePlate	פעולה המחזירה את מספר לוחית הרישוי
	מספר גלגלים	GetNumOfSeats	פעולה המחזירה את מספר המושבים
	בעלים	GetNumOfWheels	פעולה המחזירה את מספר הגלגלים
	כמות דלק מקסימלית	SellVehicle	פעולה המוכרת את הרכב לידי בעלים חדש בשם המתקבל כפרמטר
	כמות דלק נוכחית	GetOwner	פעולה המחזירה מחרוזת עם שם הבעלים
	מספר קו	SetAmount	פעולה הקובעת את הכמות הנוכחית של הדלק, כאשר עליה להיות בין 0 לבין כמות הדלק המקסימלית. אם מתקבל מספר מחוץ לטווח אין לשנות את הכמות
		GetAmount	פעולה המחזירה את כמות הדלק הנוכחית ברכב
		GetCapacity	פעולה המחזירה את תכולת מיכל הדלק
		FuelUp	פעולה המתדלקת את הרכב ומחזירה את המחיר שעלה התדלוק ע"פ הנוסחה: (כמות מקסימלית פחות הכמות הנוכחית) כפול המחיר לליטר שהינו 4.9. בסוף הפעולה צריך להתקיים כי amount = capacity
		GetBusLine	פעולה המקבלת את מספר הקו של האוטובוס הנוכחי
		ChangeBusLine	פעולה המשנה את מספר הקו של האוטובוס הנוכחי
		ToString	פעולה המחזירה את תכונות הרכב