GuyAI
מבית GOOL בגרויות

סוג הבחינה: בחינת גמר ממלכתית
מועד הבחינה: קיץ 2023 מועד ב'
סמל השאלון: 97105
שם התלמיד/ה: _____

מבנה נתונים ותכנות מונחה עצמים

הנדסת תוכנה • הנדסאים וטכנאים

הוראות

א. משך הבחינה: ארבע שעות וחצי.

ב. מבנה השאלון ומפתח ההערכה: **בשאלון זה שני מבחנים, עליכם לענות על מבחן אחד בלבד בהתאם למוסד הלימודים:** מבחן ב- Java (עמוד 2) מבחן ב- C# (עמוד 15) בכל מבחן 10 שאלות. **חלק א' – 48 נקודות** שאלות 1-4: יש לענות על **שלוש** שאלות בלבד. ערך כל שאלה 16 נקודות. **חלק ב' – 36 נקודות** שאלות 5-8: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 18 נקודות. **חלק ג' – 16 נקודות** שאלות 9-10: יש לענות על **אחת** בלבד. ערך שאלה 16 נקודות. **בסך הכול: 100 נקודות.**

ג. חומר עזר מותר בשימוש: 1. מחשבון (אין להשתמש במחשב כף יד או במחשבון עם תקשורת חיצונית). 2. קלסר **אחד בלבד** עם **חומר ההרצאות**. אין להוציא דפים מהקלסר. **אין לצרף ספרים או חוברות עם פתרונות.**

ד. הוראות מיוחדות: 1. יש לקרוא בעיון את ההנחיות בדף השער ואת כל שאלות הבחינה, ולוודא שהן מובנות. 2. את התשובות יש לכתוב בצורה מסודרת, בכתב יד ברור ונקי (גם בכך תלויה הערכת הבחינה). 3. יש להשאיר את העמוד הראשון במחברת הבחינה ריק. בסיום המבחן יש לרשום בעמוד זה את מספרי התשובות לבדיקה. התשובות ייבדקו לפי סדר כתיבתן בעמוד זה. לא ייבדקו תשובות עודפות. 4. יש לכתוב את התשובות במחברת הבחינה **בעט בלבד**, בכתב יד ברור. 5. יש להתחיל כל תשובה בעמוד חדש ולציין את מספר השאלה ואת הסעיף. אין צורך להעתיק את השאלה עצמה. 6. טיוטה יש לכתוב במחברת הבחינה בלבד. יש לרשום את המילה "טיוטה" בראש העמוד ולהעביר עליו קו כדי שלא ייבדק. 7. יש להציג פתרון מלא ומנומק, כולל חישובים לפי הצורך. הצגת תשובה סופית ללא שלבי הפתרון לא תזכה בניקוד. 8. יש להסביר בפירוט כל תוכנית שנכתבה, **תוכנית ללא הסבר מפורט לא תזכה בניקוד.** 9. אם לדעתכם חסר בשאלה נתון, יש לציין זאת ולהוסיף נתון מתאים שיאפשר לכם להמשיך בפתרון השאלה, נמקו את בחירתכם.

השאלות בשאלון זה מנוסחות בלשון רבים, אף על פי כן על כל תלמידה וכל תלמיד להשיב עליהן באופן אישי.

בהצלחה!

השאלות

1. א. שני מספרים שלמים וחיוביים `num1` ו- `num2` נקראים "זרים זה לזה" אם אין להם אף מחלק משותף פרט ל- 1.

לדוגמה:

המספרים 8 ו- 9 הם זרים, 10 ו- 21 הם זרים.

המספרים 14 ו- 21 הם לא זרים כי יש להם מחלק משותף 7.

המספרים 6 ו- 12 הם לא זרים כי יש להם מחלקים משותפים 2, 3, 6.

כתבו פעולה המקבלת שני מספרים שלמים חיוביים. הפעולה תבדוק אם הם "זרים". אם כן, הפעולה תחזיר ערך `true`, ולא - הפעולה תחזיר ערך `false`.

כותרת הפעולה:

Java:

```
public static boolean strangers(int num1, int num2)
```

C#:

```
public static bool Strangers(int num1, int num2)
```

(6 נקודות)

ב. כתבו פעולה `change` המקבלת תור `q` ומספר `num`. הפעולה משנה את סדר האיברים בתור כך שכל המספרים ה"זרים" ל- `num` יהיו בתחילת התור וכל האיברים אשר "לא זרים" ל- `num`, יהיו אחריהם.

לדוגמא:

עבור התור `q` הבא:

←	1	4	7	3	12	10	2	←

והמספר `num=9`

התור שיתקבל יכול להיות:

←	12	3	1	4	7	10	2	←

כותרת הפעולה:

Java:

```
public static void change(Queue<Integer> q, int num)
```

C#:

```
public static void Change(Queue<int> q, int num)
```

(6 נקודות)

ג. מהי הסיבוכיות של הפעולות שכתבתם בסעיפים א' ו-ב'? **הסבירו את תשובתכם.**

(4 נקודות)

2.

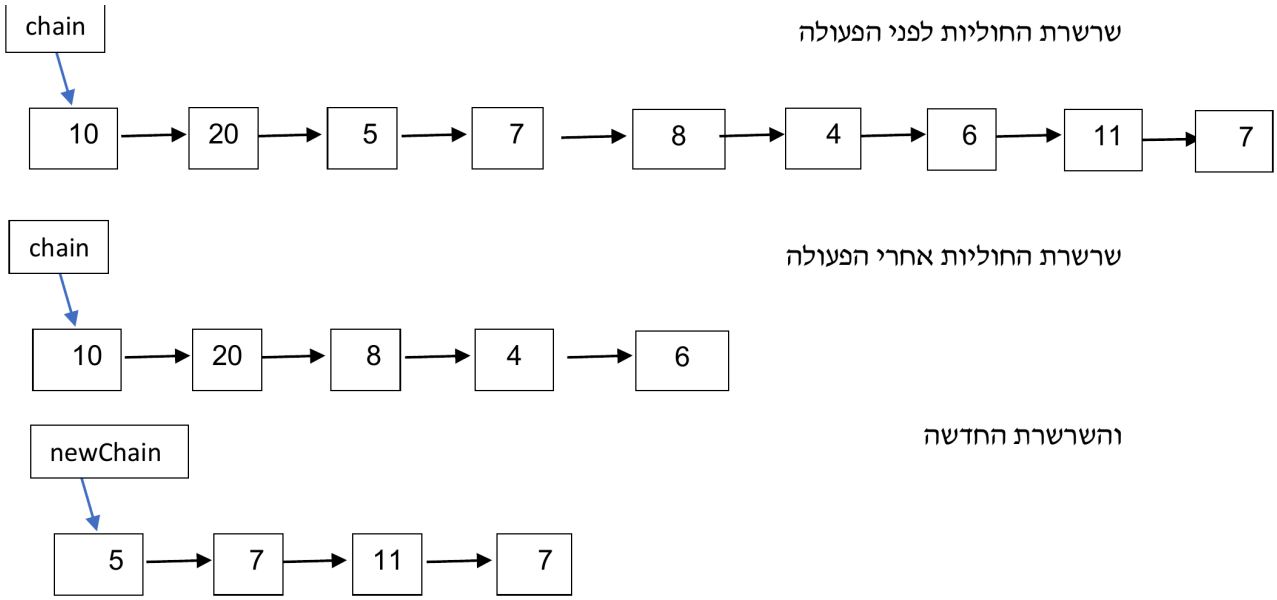
נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

כתבו פעולה המקבלת הפניה לחוליה הראשונה של שרשרת חוליות של מספרים שלמים.
הפעולה תפצל את השרשרת לשניים לפי הכלל הבא:

- אם ערך הראשון בשרשרת הוא זוגי, יש ליצור שרשרת חדשה רק מהערכים האי-זוגיים תוך מחיקתם מהשרשרת המקורית (בשרשרת המקורית יישארו רק ערכים הזוגיים).
- אם ערך הראשון בשרשרת הוא אי-זוגי, יש ליצור שרשרת חדשה רק מהערכים הזוגיים תוך מחיקתם מהשרשרת המקורית (בשרשרת המקורית יישארו רק ערכים האי-זוגיים).

הפעולה תחזיר הפניה לחוליה הראשונה של השרשרת החדשה.

לדוגמה:



א. כותרת הפעולה:

Java:

```
public static Node<Integer> split(Node<Integer> chain)
```

C#:

```
public static Node<int> Split(Node<int> chain)
```

(12 נקודות)

ב. מהי סיבוכיות הפעולה `split` שכתבתם בסעיף א'? **הסבירו את תשובתכם.**

(4 נקודות)

3.

לפניכם כותרות המחלקות `Second`, `First` ותכונות שלהן ומחלקה `Test` הכוללת פעולה ראשית:

Java:

```
public class First
{
 protected int num;
  ….
}
public class Second extends First
{
   private double x;
   private First f;
  …
}
public class Test
{
public static void main(String[] args)
{
    First[] arr = new First[5];
    arr[0] = new First (13);
    arr[1] = new First ();
    arr[2] = new Second();
    arr[3] = new Second(5, arr[0]);
    arr[4] = new Second(2, 3.7, arr[2]);
}
}
```

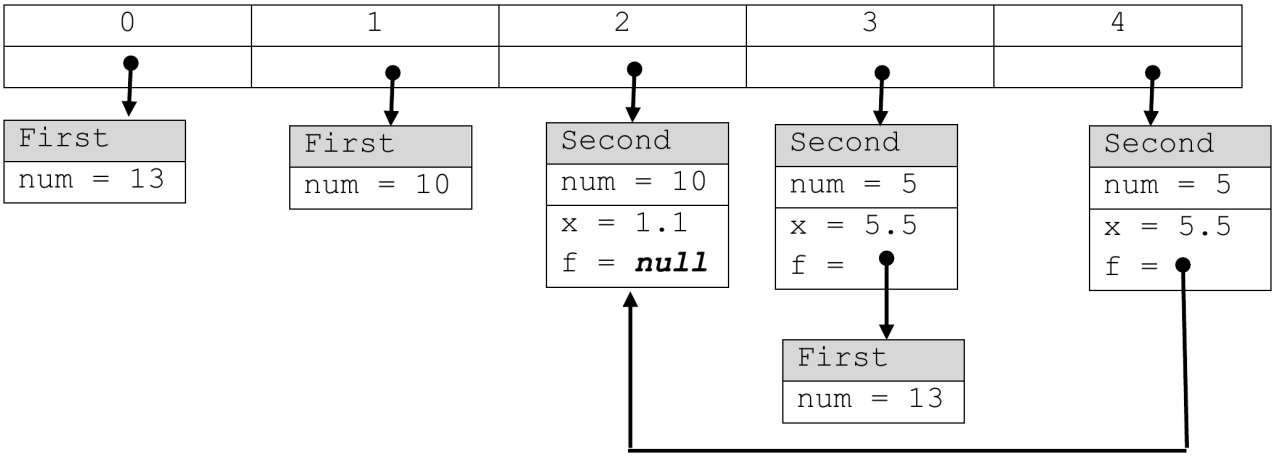
C#:

```
public class First
{
 protected int num;
  ….
}
public class Second : First
{
   private double x;
   private First f;
  …
}
public class Test
{
public static void Main(string[] args)
```


נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

```
{
   First[] arr = new First[5];
   arr[0] = new First (13);
   arr[1] = new First ();
   arr[2] = new Second();
   arr[3] = new Second(5, arr[0]);
   arr[4] = new Second(2, 3.7, arr[2]);
}
}
```

הרצת הפעולה יצרה את העצמים הבאים:

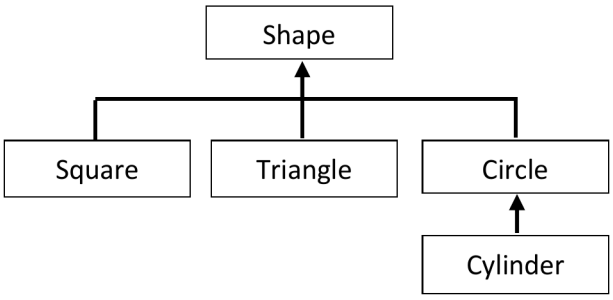


א. כתבו במחלקות `First` ו- `Second` את הפעולות הבונות (constructors) הנדרשות לשם הרצת הפעולה הראשית כך שיתקבלו העצמים המתוארים.

(8 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

נתון עץ הירושה הבא:



ב. לפניכם קטע קוד מתוך פעולה הראשית:

```
Shape s1 = new Shape();
Shape s2 = new Circle();
Shape s3 = new Cylinder();
Circle c = new Cylinder();
```

לפניכם שמונה קטעי קוד. עבור כל אחד מהסעיפים יש לציין, אם הקוד תקין או לא.
אם הקוד אינו תקין - יש להסביר למה הוא אינו תקין ולציין את סוג השגיאה (שגיאת קומפילציה/הידור או שגיאת זמן ריצה).

(1)
```
Circle c0 = new Shape();
```

(2)
```
Circle c1 = s1;
```

(3)
```
Circle c2 = (Circle)s2;
```

(4)
```
Circle c3 = (Circle)s3;
```

(5)
```
Circle c4 = (Circle) s1;
```

(6)
```
Triangle t = new Triangle();
Shape s5 = t;
Circle c5 = (Circle)s5;
```

(7)
```
Shape s = (Circle)(new Cylinder());
```

(8)
```
Circle c = (Shape)(new Cylinder());
```

(8 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

4. בפרויקט מסוים הוגדרו שני ממשקים ושלוש מחלקות:

Java:

```
public interface Int1{ void doIt1(); }
public interface Int2{ void doIt2(); }

class One implements Int1
{
   private int a;
   public One()
   {
    System.out.println("ctor One");
    this.a = 0;
   }
   public  void doIt1()
   {
    System.out.println("One:DoIt1()");
    this.a += 5;
   }
   public String toString()
   {
    return "One: a=" + this.a;
   }
}
class Two extends One
{
   private int b;
   public Two()
   {
       System.out.println("ctor Two");
       this.b = 0;
   }
   public void doIt1()
   {
       System.out.println("Two: doIt1()");
       super.doIt1();
       this.b += 5;
   }
   public  String toString()
   {
       return super.toString() + " Two: b=" + this.b;
    }
 }
class Three extends One implements Int2
{
   private int c;
```

```
  public Three()
  {
     System.out.println("ctor1 Three");
     this.c = 0;
  }

public void doIt1()
  {
     System.out.println("Three: doIt1()");
     super.doIt1();
     this.c += 5;
  }

  public void doIt2()
  {
     System.out.println("Three: doIt2()");
     super.doIt1();
     this.c += 10;
  }

  public String toString()
  {
    return super.toString() + " Three: c=" + this.c;
   }
}
```

C#:

```
public interface Int1 { void DoIt1(); }
public interface Int2 { void DoIt2(); }

class One : Int1
 {
    private int a;
    public One()
    {
     Console.WriteLine("ctor One");
     this.a = 0;
    }
    public virtual void DoIt1()
    {
     Console.WriteLine("One:DoIt1()");
     this.a += 5;
    }
    public override string ToString()
```


נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

```
    {
     return "One: a=" + this.a;
    }
 }
class Two : One
 {
    private int b;
    public Two()
    {
     Console.WriteLine("ctor Two");
     this.b = 0;
    }
    public override void DoIt1()
    {
     Console.WriteLine("Two: doIt1()");
     base.DoIt1();
     this.b += 5;
    }
    public override string ToString()
    {
    return base.ToString() + " Two: b=" + this.b;
    }
}
class Three : One, Int2
{
    private int c;
    public Three()
    {
     Console.WriteLine("ctor1 Three");
     this.c = 0;
    }
    public override void DoIt1()
    {
     Console.WriteLine("Three: doIt1()");
     base.DoIt1();
     this.c += 5;
    }
    public void DoIt2()
    {
     Console.WriteLine("Three: doIt2()");
     base.DoIt1();
     this.c += 10;
    }
    public override string ToString()
    {
     return base.ToString() + " Three: c=" + this.c;
```


נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

```
    }
}
```

5-1 לפניכם חמישה קטעי קוד מתוך פעולה הראשית במחלקה אחרת. עבור כל אחד מהסעיפים יש לציין, אם הקוד תקין או לא. אם הקוד אינו תקין - יש להסביר למה הוא אינו תקין ולציין את סוג השגיאה (קומפילציה או זמן ריצה). אם הקטע תקין - מהו הפלט (אם יש)?

שימו לב שאין קשר בין סעיפים!

(1) **Java:**

```
Int2 x1 = new Int2();
x1.doIt2();
System.out.println(x1);
```

C#:

```
Int2 x1 = new Int2();
x1.DoIt2();
Console.WriteLine(x1);
```

(2) **Java:**

```
Int1 x2 = new Three();
Int2 y2 = (Int2)x2;
y2.doIt2();
System.out.println(x2);
System.out.println(y2);
```

C#:

```
Int1 x2 = new Three();
Int2 y2 = (Int2)x2;
y2.DoIt2();
Console.WriteLine(x2);
Console.WriteLine(y2);
```

(3) **Java:**

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

```
Three x3 = new One();
x3.doIt1();
System.out.println(x3);
```

C#:

```
Three x3 = new One();
x3.DoIt1();
Console.WriteLine(x3);
```

(4) Java:

```
One x4 = new Two();
Three y4 = (Three)x4;
y4.doIt2();
System.out.println(x4);
System.out.println(y4);
```

C#:

```
One x4 = new Two();
Three y4 = (Three)x4;
y4.DoIt2();
Console.WriteLine(x4);
Console.WriteLine(y4);
```

(5) Java:

```
Int2 x5 = new Three();
((One)x5).doIt1();
System.out.println(x5);
```

C#:

```
Int2 x5 = new Three();
((One)x5).DoIt1();
```


נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

```
Console.WriteLine(x5);
```

5. הערה כללית לשאלה:

- אין להשתמש במבני נתונים נוספים פרט למחסנית או מחסניות אחרות.
- יש לשמור את המחסנית במצב מקורי בסיום הפעולה.

א. כתבו פעולה המקבלת מחסנית של מספרים שלמים `stk` ומספר שלם `num`.
הפעולה תחזיר מיקומו **הראשון** של `num` בתוך המחסנית. אם `num` לא מופיע במחסנית, הפעולה תחזיר ערך 1-.

לדוגמה:

עבור המחסנית `stk` הבאה והמספר `num=3`

⇅	18	3	15	13	3	12	21	12	10	3	7

הפעולה תחזיר 2.

כותרת הפעולה:

Java:

```
public static int firstPlace(Stack<Integer>stk, int num)
```

C#:

```
public static int FirstPlace(Stack<int> stk, int num)
```

(5 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

ב. כתבו פעולה המקבלת מחסנית של מספרים שלמים `stk` ומספר שלם `num`. הפעולה תחזיר את מיקומו **האחרון** של `num` בתוך המחסנית. אם `num` לא מופיע במחסנית, הפעולה תחזיר ערך 1-.

לדוגמה:

עבור המחסנית `stk` הבאה והמספר `num=3`

⇅	18	3	15	13	3	12	21	12	10	3	7

הפעולה תחזיר 10.

כותרת הפעולה:

Java:

```
public static int lastPlace(Stack<Integer>stk, int num)
```

C#:

```
public static int LastPlace(Stack<int> stk, int num)
```

(5 נקודות)

ג. כתבו פעולה המקבלת מחסנית של מספרים שלמים, כל מספר במחסנית מופיע **פעמיים בדיוק**. הפעולה תבדוק מהו המרחק במחסנית עבור כל שני מספרים זהים ותחזיר את **המרחק הקטן ביותר** בין שני מספרים זהים. המרחק בין שני מספרים זהים מוגדר כמות המספרים הנמצאים ביניהם.

לדוגמה:

עבור המחסנית הבאה הפעולה תחזיר 1 כי מרחק הקטן ביותר בין שני מספרים זהים (11 ו- 11) הוא 1.

⇅	4	8	2-	11	4	11	1	2-	8	1

(5 נקודות)

ד. מהן סיבוכיות של הפעולות שכתבתם בסעיפים א'-ג'? **הסבירו את תשובתכם.**

(3 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

6. בחנות הנעליים "קח - נעל - לך" יש מאות זוגות נעליים. כדי לאפשר מחשוב של מחסן החנות הוחלט לחלק את מגוון הסוגים השונים של הנעלים לשלוש קטגוריות: נעלי ילדים, נעלי נשים, נעלי גברים.

בנעלי ילדים עבור כל זוג נשמרים הפרטים הבאים:

- יצרן (`producer`).
- מידה (`size`).
- צבע (`color`).
- מחיר בסיסי (`price`).
- האם נעל אורתופדי? (`orth`).

בנעלי נשים עבור כל זוג נשמרים הפרטים הבאים:

- יצרן (`producer`).
- מידה (`size`).
- צבע (`color`).
- מחיר בסיסי (`price`).
- שנת ייצור (`year`).
- גובה עקב (`heel`).

בנעלי גברים עבור כל זוג נשמרים הפרטים הבאים:

- יצרן (`producer`).
- מידה (`size`).
- צבע (`color`).
- מחיר בסיסי (`price`).
- שנת ייצור (`year`).
- אם זו נעל עבודה (`work`).

מחיר הנעליים נקבע לפי מספר קריטריונים:

- מחיר בסיסי עבור נעלי ילדים הוא 100 ₪.
- מחיר בסיסי עבור נעליים למבוגרים הוא 200 ₪.
- עבור מידה מעל 40 המחיר הוא 300 ₪.
- נעל ילדים אורתופדית מוסיפה 25 ₪ למחיר.
- גובה עקב של שלושה ס"מ לפחות מוסיף 30 ₪ למחיר, גובה עקב של חמישה ס"מ לפחות מוסיף 60 ₪ למחיר.
- עבור נעל שנוצרה השנה, 2023, יש תוספת של 50% למחיר.
- נעל עבודה מוזילה את מחיר הנעל ב- 50 ₪.

א. הגדירו את המחלקות הדרושות לניהול המחסן באופן המתאים ביותר לעקרונות של תכנות מונחה עצמים. סרטטו תרשים UML המתאר את הקשר בין המחלקות.

(2 נקודות)

ב. לכל אחת מהמחלקות כתבו את כותרת המחלקה ואת התכונות שלה. כתבו לכל מחלקה את הפעולה הבונה.

אפשר להניח שבכל מחלקה הוגדרו פעולות `set/get`.

(4 נקודות)

ג. המחלקה מחסן (`Storage`) מיועדת לשמירת נתונים עבור כל זוגות הנעליים שהחנות מציעה. כתבו את כותרת המחלקה ואת התכונה או התכונות שלה והוסיפו אותה ל-UML. **יש לתעד את המחלקה!**

(2 נקודות)

ד. כתבו פעולה או פעולות `calcPrice` לחישוב המחיר הסופי של נעליים לפי הקריטריונים. המימוש חייב להיות בהתאם לעקרונות תכנות מונחה עצמים. ציינו באיזו מחלקה כתבתם את הפעולה.

(8 נקודות)

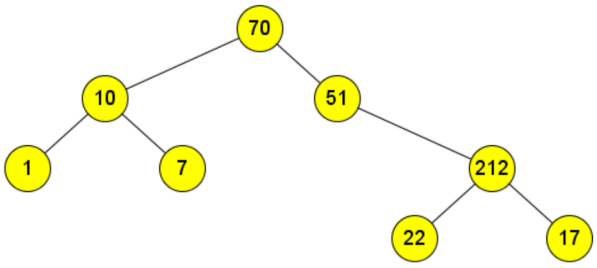
ה. כתבו במחלקה `Storage` פעולה המחזירה את מספר זוגות הנעלים האורתופדיות שהחנות מציעה.

(2 נקודות)

7. עץ בינארי של מספרים שלמים נקרא "עץ מיוחד" אם הוא עונה על כלל הבא: עבור כל צומת בעץ כל הערכים הנמצאים בתת- העץ השמאלי שלו שונים מכל הערכים הנמצאים בתת-העץ הימני שלו.

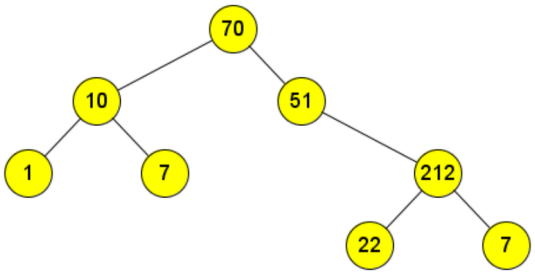
לדוגמה:

העץ הבא הוא "עץ מיוחד":



נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

העץ הבא הוא לא "עץ מיוחד" כי הערך 7 מופיע גם בתת-העץ השמאלי של הצומת 70 וגם בתת-העץ הימני שלו.



א. האם קיים "עץ מיוחד" שיש בו ערכים זהים? אם כן - ציירו את העץ ואם לא - תסבירו מדוע עץ כזה לא קיים.

(3 נקודות)

ב. האם קיים **עץ חיפוש בינרי** שהוא לא "עץ מיוחד"? אם כן - ציירו את העץ ואם לא - הסבירו מדוע עץ כזה לא קיים.

(3 נקודות)

ג. כתבו פעולה המקבלת הפניה לשורש של עץ בינרי ובודקת אם הוא "עץ מיוחד". אם כן - הפעולה תחזיר ערך `true`, ולא - הפעולה תחזיר ערך `false`.

(9 נקודות)

ד. מהי סיבוכיות הפעולה שכתבתם בסעיף ג'? הסבירו את תשובתכם.

(3 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

8. נתונות המחלקות `A, B, C` הבאות:

Java:

```
public class A
{
   private static int countA = 0;
   protected int x;
   public A()
    {
       countA++;
       this.x = 0;
   }
   public A(int x)
    {
       countA++;
       this.x = x;
   }
   public String toString()
    {
       return "A:"+this.x;
    }
   public int doIt(int z)
    {
       return this.x * z;
    }
   public static int getCount(){ return countA;  }
}// end of class A

public class B extends A
{
   protected int y;
   private static int countB = 0;
   public B(int y)
   {
       this.y = y;
       countB++;
   }
   public B(int x, int y)
   {
       super(x);
       countB++;
       this.y = y;
   }
   public String toString()
   {
     return "B:"+super.toString()+":"+this.y;
```

```
  }
  public static int getCount(){return countB;}
  public int doIt(int z)
  {
     return super.doIt(z) + this.y;
  }
} //end of class B
public class C extends B
{
  private A myA;
  private static int countC = 0;
  public C(int x, int y)
  {
    super(x, y);
    this.myA = new A(x+y);
    countC++;
  }
  public static int getCount(){return countC;}
  public String toString()
  {
    return "C:"+super.toString()+" ["+this.myA+"]";
  }
}// end of class C
```

C#:

```
public class A
{
 private static int countA = 0;
 protected int x;
 public A()
 {
   countA++;
   this.x = 0;
 }
 public A(int x)
 {
   countA++;
   this.x = x;
 }
 public override string ToString()
 {
   return "A:" + this.x;
 }
 public virtual int DoIt(int z)
```

```
 {
   return this.x * z;
 }
 public static int GetCount() { return countA; }
}// end of class A


public class B : A
{
 protected int y;
 private static int countB = 0;
 public B(int y)
 {
   this.y = y;
   countB++;
 }
 public B(int x, int y):base(x)
 {
   countB++;
   this.y = y;
 }
 public override string ToString()
 {
   return "B:" + base.ToString() + ":" + this.y;
 }
 public static new   int GetCount() { return countB; }
 public override int DoIt(int z)
 {
   return base.DoIt(z) + this.y;
 }
} //end of class B
public class C : B
{
 private A myA;
 private static int countC = 0;
 public C(int x, int y):base(x,y)
 {
   this.myA = new A(x + y);
   countC++;
 }
 public static new int GetCount() { return countC; }
 public override string ToString()
 {
   return "C:" + base.ToString() + " [" + this.myA + "]";
 }
}// end of class C
```

א. ציירו תרשים UML שמייצג את הקשרים בין המחלקות `A, B, C`.

יש לסמן ירושה באמצעות החץ ⟶ והכלה באמצעות הסימן ◆——

(1 נקודות)

ב. נתון קטע קוד בפעולה הראשית במחלקה אחרת:

```
A[] arr = new A[5];
arr[0] = new A(13);
arr[1] = new B(5, 10);
arr[2] = new C(2, 7);
arr[3] = new C(3, 3);
arr[4] = new B(5);
```

עקבו אחרי ביצוע קטע קוד. יש לצייר את כל העצמים שנוצרו ולציין ערכי התכונות שלהם.

(5 נקודות)

ג. לקוד של סעיף ב' התווסף קוד הבא:

Java:

```
System.out.println("A objects: " + A.getCount());
System.out.println("B objects: " + B.getCount());
System.out.println("C objects: " + C.getCount());
```

C#:

```
Console.WriteLine("A objects: " + A.GetCount());
Console.WriteLine("B objects: " + B.GetCount());
Console.WriteLine("C objects: " + C.GetCount());
```

רשמו מה יהיה הפלט.

(3 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

ד. לקוד של סעיף ב' התווסף קוד הבא:

Java:

```
for (int k=0; k <arr.length; k++)
 System.out.println(arr[k]);
```

C#:

```
for (int k=0; k <arr.length; k++)
  Console.WriteLine(arr[k]);
```

רשמו מה יהיה הפלט.

(5 נקודות)

ה. לקוד של סעיף ב' התווסף קוד הבא:

Java:

```
for (int k=0; k <arr.length; k++)
 if(arr[k] instanceof B)
    System.out.println(arr[k].doIt(k+1));
```

C#:

```
for (int k=0; k <arr.length; k++)
  if(arr[k] instanceof B)
    Console.WriteLine(arr[k].doIt(k+1));
```

רשמו מה יהיה הפלט.

(4 נקודות)

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

9. נתונה הפעולה `sod` המקבלת תור `q` של מספרים שליליים וחיוביים (ללא אפסים). הפעולה משתמשת בפעולה רקורסיבית `what`

Java:

```
public static void sod(Queue<Integer> q)
{
  q.insert(0);
  what(q);
  q.remove();
}

public static void what(Queue<Integer> q)
{
 if (q.head() != 0)
 {
  int x = q.remove();
  if (x > 0)
    q.insert(x);
  what(q);
  if (x < 0)
    q.insert(x);
 }
}
```

C#:

```
public static void Sod(Queue<int> q)
{
  q.Insert(0);
  What(q);
  q.Remove();
}

public static void What(Queue<int> q)
{
 if (q.Head() != 0)
 {
  int x = q.Remove();
  if (x > 0)
    q.Insert(x);
  What(q);
  if (x < 0)
    q.Insert(x);
```


נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

```
    }
}
```

א. מה יהיה התוכן של התור `q` אחרי זימון הפעולה `sod(q)` עבור התור הבא:

←	3	-1	7	2	4	-9	←

יש להראות מעקב אחרי ביצוע הפעולה.

(6 נקודות)

ב. אחרי זימון הפעולה `sod(q)` עבור התור `q` לא ריק, התור `q` **נשאר ללא שינוי**.
לפניכם ארבע טענות. עבור כל אחד מהטענות ציינו אם היא נכונה או לא ותנו דוגמה של תור מתאימה.

(1) זה יכול לקרות רק כאשר בתור יש ערך אחד בלבד.

(2) זה יכול לקרות רק כאשר התור מלא בערכים חיוביים בלבד.

(3) זה יכול לקרות רק כאשר התור מלא בערכים שליליים בלבד.

(4) לא ייתכן שבתור יש גם ערכים חיוביים וגם ערכים שליליים.

(4 נקודות)

ג. מה מבצעת הפעולה `sod` באופן כללי עבור התור של מספרים שליליים וחיוביים (ללא אפסים).

(2 נקודות)

ד. כתבו את הפעולה `sod` בלי שימוש ברקורסיה.

(4 נקודות)

10. חברת "בית הבא" מפרסמת באינטרנט דירות להשכרה. כל מי שרוצה לקבל שירות חייב להירשם כלקוח. כל לקוח יכול לבחור לצפות בצילומי הדירות ממאגר הדירות שהחברה מציגה. אם לקוח רוצה לצפות בפרטי דירה חדשה (דירה שהוא עדיין לא צפה בה), הוא יחויב בשקל אחד. אם לקוח רוצה לחזור לדירה שהוא כבר צפה בה - הצפייה תהיה ללא תשלום נוסף. מערכת ניהול החברה כוללת את המחלקות הבאות:

המחלקה `Apartment` מייצגת דירה להשכרה. למחלקה יש את התכונות הבאות:

- `code` - מספר ייחודי של הדירה, מטיפוס שלם, `int`.
- `city` - העיר שבה נמצאת הדירה, מטיפוס מחרוזת, `String`.
- `numRooms` - מספר החדרים, מטיפוס שלם, `int`.
- `numPictures` - מספר תמונות הדירה, מטיפוס שלם, `int`.

במחלקה הוגדרו פעולה בונה ופעולות `set/get` לכל התכונות. המחלקה `Client` מייצגת לקוח. למחלקה יש את התכונות הבאות:

- `id` - מספר תעודת הזהות של הלקוח, מטיפוס מחרוזת, `String`.
- `codeList` - רשימה (שרשרת חוליות) של קודי הדירות שהלקוח כבר צפה בהן, מטיפוס `Node<Integer>`.
- `totalSum` - סכום תשלום, מטיפוס שלם, `int`.

במחלקה הוגדרו פעולות `set/get` לכל התכונות והפעולה הבונה הבאה:

Java:

```
public Client(String id)
{
    this.id = id;
    this.codeList = null;
    this.totalSum = 0;
}
```

C#:

```
public Client(string id)
{
  this.id = id;
  this.codeList = null;
  this.totalSum = 0;
}
```

(בנוסח C#: הטיפוסים `string` ו-`Node<int>`, ופעולות `Set/Get`.)

א. כתבו במחלקה `Client` פעולה `addApCode`. הפעולה מקבלת קוד דירה שהלקוח מעוניין לצפות בה. אם זו דירה חדשה, הפעולה תוסיף קוד דירה לרשימת דירות של הלקוח ותעדכן סכום לתשלום.

(4 נקודות)

המחלקה `Rentings` מייצגת נתונים של החברה. למחלקה שתי תכונות:

- `aparts` - רשימה (שרשרת חוליות) של דירות מטיפוס `Node<Apartment>`.
- `clients` - רשימה (שרשרת חוליות) של לקוחות מטיפוס `Node<Client>`.

במחלקה הוגדרו פעולה בונה ופעולות `set/get` לכל התכונות.

ב. כתבו במחלקה `Rentings` פעולה המקבלת את מספר תעודת הזהות של הלקוח. הפעולה תוסיף את הלקוח לרשימת הלקוחות. אם הלקוח כבר קיים, הפעולה לא מבצעת כלום. כותרת הפעולה:

Java:

```
public void addClient(String id)
```

C#:

```
public void AddClient(string id)
```

(6 נקודות)

ג. כדי לייעל חיפוש דירה מתאימה לדרישות הלקוח הוחלט להוסיף למחלקה `Rentings` פעולה המדפיסה רשימת קודים של דירות הנמצאות בעיר מסוימת `city`, שיש להן מספר חדרים מסוים `numRooms` ויש לפחות `numPictures` תמונות.

אם אין במאגר נתוני החברה דירות מתאימות הפעולה תדפיס הודעה "No match".

כותרת הפעולה:

Java:

```
public void printMatch(String city, int numRooms, int numPictures)
```

C#:

```
public void PrintMatch(string city, int numRooms, int numPictures)
```

(6 נקודות)

כל מבחני מה"ט עם פתרונות מלאים: www.guyai.co.il/mahat