GuyAI
מבית GOOL בגרויות

סוג הבחינה: בחינת גמר ממלכתית
מועד הבחינה: אביב 2026 מועד א'
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- C# (עמוד 2) מבחן ב- JAVA (עמוד 12) בכל מבחן תשע שאלות. **חלק א' – 32 נקודות** שאלות 1–3: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 16 נקודות. **חלק ב' – 32 נקודות** שאלות 4–6: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 16 נקודות. **חלק ג' – 36 נקודות** שאלות 7–9: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה 18 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 המבחן, ולוודא שהן מובנות. 2. את התשובות יש לכתוב בצורה מסודרת, בכתב יד ברור ונקי (גם בכך תלויה הערכת המבחן). 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. **חל איסור מוחלט להוציא שאלון או מחברת בחינה מחדר הבחינה!** בנספחי הבחינה מובאים ממשקים (interface) הכוללים פונקציות ומחלקות עזר, יש להקפיד ולהשתמש בהן בשאלות הרלוונטיות בבחינה על מנת להימנע מאיבוד ניקוד. לשאלון מצורפים נספח ממשקים לבחינה JAVA, נספח ממשקים לבחינה C# ומילון עזר.

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

בהצלחה!

השאלות

1. הגדרה:

סדרת מספרים נקראת "עולה - יורדת" אם עד מספר מסוים המספרים הולכים וגדלים וממספר זה ואילך, הם הולכים וקטנים.

לדוגמה, הסדרה שלפניכם היא סדרה "עולה - יורדת":

4, 9, 13, 20, 45, 33, 18, 1

4
9
13
20
45
33
18
1

א. כתבו פעולה מקבלת מחסנית של מספרים שלמים. הפעולה תבדוק אם הערכים שבמחסנית מהווים סדרה "עולה - יורדת". אם כן - הפעולה תחזיר ערך true, ואם לא - הפעולה תחזיר ערך false.

יש לשמור על שלמות המחסנית בסוף הפעולה.

כותרת הפעולה:

Java:

```
public static boolean isUpDown(Stack<Integer> st)
```

C#:

```
public static bool IsUpDown(Stack<int> st)
```

(14 נקודות)

ב. מהי הסיבוכיות של הפעולה שכתבתם בסעיף א'? **הסבירו את תשובתכם.**

(2 נקודות)

2.

א. כתבו פעולה המקבלת הפניה לחוליה הראשונה בשרשרת חוליות של מספרים שלמים ומספר שלם חיובי pos. הפעולה תחזיר איבר שנמצא במקום pos בשרשרת. אם אין מקום pos בשרשרת, הפעולה תחזיר ערך (-1).

יש להניח כי המיקום נספר החל מ-1, כאשר pos = 1 מתייחס לחוליה הראשונה.

Java:

```
public static int valueAt(Node<Integer> ch, int pos)
```

C#:

```
public static int ValueAt(Node<int> ch, int pos)
```

(6 נקודות)

ב. כתבו פעולה המקבלת שתי הפניות לחוליות הראשונות של שתי שרשראות של מספרים שלמים ch1 ו- ch2. הפעולה תיצור שרשרת חדשה לפי הכלל הבא:

- ראשית מכניסים את האיבר הראשון מהשרשרת ch1, ואחריו - את האיבר האחרון מהשרשרת ch2;
- מכניסים את האיבר השני מהשרשרת ch1, ואחריו את האיבר שלפני האחרון מהשרשרת ch2;
- וכך הלאה.

הפעולה תחזיר הפניה לחוליה הראשונה של השרשרת החדשה.

אפשר להניח שאורכן של שתי השרשראות שווה.

כותרת הפעולה:

Java:

```
public static Node<Integer> merge (Node<Integer>ch1,
                                   Node<Integer>ch2)
```

C#:

```
public static Node<int> Merge (Node<int>ch1, Node<int>ch2)
```

(8 נקודות)

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

(2 נקודות)

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

3. בפרויקט פיתוח רכב אוטונומי הוגדרו שלושה ממשקים ושתי מחלקות:

Java:

```
public interface Steering {
   void turnLeft();
   void turnRight(int degrees);
}
public interface Navigation{ void setDestination(String dest); }
public interface Sensing { void activateLidar(); }
public class AutonomousCar implements Steering, Navigation,
                                      Sensing {
   private String modelName;
   private double batteryLevel;
   // …
}
public class TestDrive {
   public static void main(String[] args) {
      AutonomousCar car1 = new AutonomousCar("Tesla Y", 98.5);
      AutonomousCar car2 = new AutonomousCar("Waymo One");
      // ( ˙˙˙˙˙)
   }
}
```

C#:

```
public interface Steering {
   void TurnLeft();
   void TurnRight(int degrees);
}
public interface Navigation{ void SetDestination(string dest); }
public interface Sensing { void ActivateLidar(); }
public class AutonomousCar : Steering, Navigation, Sensing {
   private string modelName;
   private double batteryLevel;
   // …
}
public class TestDrive {
   public static void Main(string[] args) {
      AutonomousCar car1 = new AutonomousCar("Tesla Y", 98.5);
      AutonomousCar car2 = new AutonomousCar("Waymo One");
      // ( ˙˙˙˙˙)
   }
}
```

א. השלימו את המחלקה AutonomousCar: כתבו כותרות לפעולות החסרות בה, כולל כל הפעולות הבונות (constructors) הנדרשות כדי שהקוד בפעולה main יהיה תקין, וכן את כל הפעולות הנדרשות למימוש הממשקים. **אין צורך לממש את גוף הפעולות.**
(4 נקודות)

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

ב. בכל אחד מסעיפים הבאים השורה (*) תוחלף בשורת קוד משורות הקוד 1-6 מהטבלה הבאה. עבור כל אחד מהסעיפים, כתבו אם הקוד תקין או לא. אם הקוד אינו תקין - יש להסביר למה הוא אינו תקין, יש לציין את סוג השגיאה (קומפילציה או זמן ריצה).

שימו לב: אין קשר בין סעיפים.

(1) **Java:**

```
Steering s1 = car1;
s1.turnRight(45);
```

C#:

```
Steering s1 = car1;
s1.TurnRight(45);
```

(2) **Java:**

```
Navigation n2 = new Navigation();
n2.setDestination("Tel Aviv");
```

C#:

```
Navigation n2 = new Navigation();
n2.SetDestination("Tel Aviv");
```

(3) **Java:**

```
Navigation n3 = new AutonomousCar("Audi A8", 60.0);
Steering s3 = (AutonomousCar)n3;
```

C#:

```
Navigation n3 = new AutonomousCar("Audi A8", 60.0);
Steering s3 = (AutonomousCar)n3;
```


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

(4) Java:

```
Navigation n4 = new AutonomousCar("Mercedes");
n4.activateLidar();
```

C#:

```
Navigation n4 = new AutonomousCar("Mercedes");
n4.ActivateLidar();
```

(5) Java:

```
Steering s5 = new AutonomousCar("Ford");
Sensing sn5 = (Sensing) s5 ;
sn5.activateLidar();
```

C#:

```
Steering s5 = new AutonomousCar("Ford");
Sensing sn5 = (Sensing) s5 ;
sn5.ActivateLidar();
```

(6) Java:

```
Sensing sn6 = new AutonomousCar("Nissan", 70.0);
((AutonomousCar) sn6).turnLeft();
((Navigation) sn6).setDestination("Haifa");
```

C#:

```
Sensing sn6 = new AutonomousCar("Nissan", 70.0);
((AutonomousCar) sn6).TurnLeft();
((Navigation) sn6).SetDestination("Haifa");
```

(12 נקודות)

4. א. הגדרה:

שני מספרים שלמים וחיוביים num1 ו-num2 נקראים "זרים זה לזה" אם אין להם שום ספרה משותפת.

לדוגמה:

המספרים 81224 ו-973656 הם זרים, וגם המספרים 1000 ו-289 הם זרים.

המספרים 14 ו-21 הם לא זרים כי יש להם ספרה משותפת, 1.

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

כותרת הפעולה:

Java:

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

C#:

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

(6 נקודות)

ב. כתבו פעולה המקבלת תור של מספרים שלמים חיוביים. הפעולה תבדוק אם כל הערכים בתור זרים זה לזה. אם כן, הפעולה תחזיר ערך true, ואם לא - הפעולה תחזיר ערך false.

כותרת הפעולה:

Java:

```
public static boolean isStrangersQueue(Queue<Integer> q)
```

C#:

```
public static bool IsStrangersQueue(Queue<int> q)
```

(7 נקודות)

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

(3 נקודות)

5. חברת סטארט-אפ פיתחה פלטפורמה לשירותי רשת. הטבלה שלפניכם מייצגת את היררכיית היישומים (Applications) במערכת.

מחלקה	יורשת מ-	תיאור
Application	Object	מחלקת אב - יישום כללי במערכת
WebService	Application	שירות רשת (Web)
MobileApp	Application	אפליקציה לנייד
BackendService	WebService	שירות צד שרת
FrontendApp	WebService	שירות צד לקוח
DataQuery	MobileApp	כלי לשליפת נתונים באפליקציה ניידת

א. שרטטו את עץ הירושה הכולל של המחלקות (Class Diagrams).
(2 נקודות)

ב. נתון קטע קוד מתוך הפעולה הראשית (main):

```
Application a1 = new Application();
Application a2 = new FrontendApp();
Application a3 = new MobileApp();
WebService ws1 = new BackendService();
MobileApp ma1 = new DataQuery();
```

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).

שימו לב: אין קשר בין הקטעים.

(1)
```
WebService ws2 = new Application();
```

(2)
```
WebService ws3 = a2;
```

(3)
```
WebService ws4 = (WebService)a2;
```

(4)
```
WebService ws5 = (WebService)a3;
```

(5)
```
BackendService bs1 = (BackendService)ws1;
```

(6)
```
MobileApp ma2 = (MobileApp)a1;
```

(7)
```
Application a4 = (MobileApp)(new DataQuery());
```

(8)
```
FrontendApp fa1 = (WebService)(new BackendService());
```

(8 נקודות)

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

ג. לכל שירות במערכת יש עלות תפעול משלו (Cost). המטרה היא לחשב את סכום העלויות של כל היישומים שהם שירותי הרשת (WebService).

(1) באילו מהמחלקות יש להגדיר את הפעולה `public double getCost()` (ב-C#: `public double GetCost()`) המחזירה את עלות התפעול? **הסבירו את תשובתכם.**

(2) כתבו פעולה סטטית בשם sumWebServiceCost המקבלת מערך של יישומים ומחזירה את סכום העלויות של כל **שירותי הרשת**. כותרת הפעולה:

Java:

```
public static double sumWebServiceCost(Application[] arr)
```

C#:

```
public static double SumWebServiceCost(Application[] arr)
```

הניחו שהפעולה getCost() (ב-C#: GetCost()) מומשה כראוי.

(6 נקודות)

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

6. נתונה המחלקה SpecNode:

```
public class SpecNode {
  private int value;
  private int numOfSmallers;
  private SpecNode next;

  public SpecNode (int val) {
    this.value = val;
    this. numOfSmallers = 0;
    this.next = null;
  }
}
```

למחלקה הוגדר בנאי.

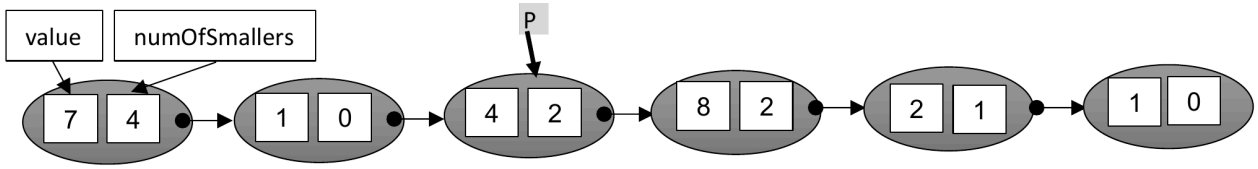
לכל תכונה במחלקה הוגדרו פעולות get/set (ב-C#: Get/Set).

בשרשרת חוליות מסוג SpecNode:

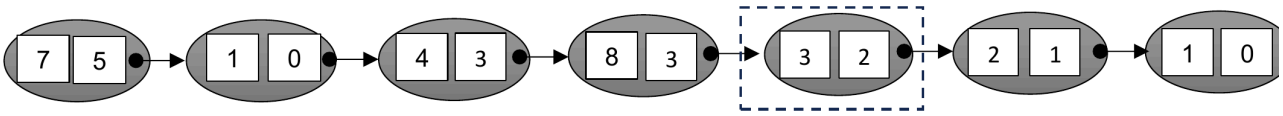
- value - מספר שלם.
- numOfSmallers - מספר החוליות המופיעות **אחרי** החוליה הנוכחית בשרשרת שערך ה-value שלהן **קטן** מערך ה-value של החוליה הנוכחית.
- next - מצביע לחוליה הבאה.

לדוגמה:

- ערך ה-value של החוליה הראשונה הוא 7, ואחריה יש ארבע חוליות שערכי ה-value שלהן קטן מ-7 ובגלל זה ערך התכונה numOfSmallers של החוליה הראשונה הוא 4.
- P מצביע על חוליה עם value=4, והתכונה numOfSmallers=2 מציינת כי יש שתי חוליות אחריה שערכי ה-value שלהן קטנים מ-4 (כלומר 1 ו-2).



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



א. כתבו פעולה חיצונית המכניסה חוליה חדשה לשרשרת במיקום pos ועם ערך val. הפעולה מקבלת הפניה לחוליה הראשונה של שרשרת חוליות מסוג SpecNode, מספר שלם val ומספר שלם חיובי pos. (המיקום הראשון בשרשרת הוא 1). **אפשר להניח כי מיקום pos חוקי (אינו עולה על אורך רשימה+1).**

לאחר ההוספה, יש לעדכן את כל ערכי numOfSmallers בהתאם.

כותרת הפעולה:

Java:

```
public static SpecNode addNum(SpecNode list, int val, int pos)
```

C#:

```
public static SpecNode AddNum (SpecNode list, int val, int pos)
```

לדוגמה:

אחרי זימון הפעולה addNum(list, 3, 5) השרשרת שתתקבל תהיה:

(14 נקודות)

ב. מהי סיבוכיות הפעולה? **הסבירו את תשובתכם.**

(2 נקודות)

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

7. נתון פרויקט שהוגדרו בו שלוש מחלקות Employee, Manager, Intern ומחלקה Driver, כך:

Java:

```
public class Employee {
  private int seniority; // ותק

  public Employee(int seniority) {
    this.seniority = seniority;
  }
  public int getSeniority() {
    return this.seniority;
  }
  public boolean check(Employee e) {
    System.out.println("EmpCheck");
    return this.seniority > e.seniority;
  }
}
public class Manager extends Employee {
  public Manager(int seniority) {
    super(seniority);
  }
  public boolean check(Manager m) {
    System.out.println("MgrCheck");
    return this.getSeniority() == m.getSeniority();
  }
}
public class Intern extends Employee {
  private int mentorId;

  public Intern(int seniority, int mentorId) {
    super(seniority);
    this.mentorId = mentorId;
  }
  public boolean check(Employee e) {
    System.out.println("IntCheck");
    if (!(e instanceof Intern)) {
      return true;
    }
    Intern other = (Intern) e;
    return this.mentorId == other.mentorId;
  }
}

public class Driver
{
  public static void main(String[] args)
```

```
  {
    Employee e1 = new Manager(15);
    Manager  m1 = new Manager(15);
    Employee e2 = new Employee(10);
    Employee i1 = new Intern(2, 800);
    Intern   i2 = new Intern(2, 900);
    Intern   i3 = new Intern(5, 800);


(1)   System.out.println(e1.check(m1));
(2)   System.out.println(m1.check(e1));
(3)   System.out.println(m1.check(m1));
(4)   System.out.println(i1.check(e2));
(5)   System.out.println(i1.check(i2));
(6)   System.out.println(i1.check(i3));
(7)   System.out.println(e2.check(i1));

  }
}
```

C#:

```
public class Employee {
  private int seniority; // ותק

  public Employee(int seniority) {
    this.seniority = seniority;
  }
  public int GetSeniority() {
    return this.seniority;
  }
  public virtual bool Check(Employee e) {
    Console.WriteLine("EmpCheck");
    return this.seniority > e.seniority;
  }
}
public class Manager : Employee {
  public Manager(int seniority): base(seniority) {
  }
  public bool Check(Manager m) {
    Console.WriteLine("MgrCheck");
    return this.GetSeniority() == m.GetSeniority();
  }
}
public class Intern : Employee {
```


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

```
  private int mentorId;

  public Intern(int seniority, int mentorId): base(seniority){
      this.mentorId = mentorId;
  }
  public override bool Check(Employee e) {
     Console.WriteLine("IntCheck");
     if (!(e is Intern)) {
        return true;
     }
     Intern other = (Intern) e;
     return this.mentorId == other.mentorId;
  }
}

public class Driver
{
   public static void Main(string[] args)
   {
      Employee e1 = new Manager(15);
      Manager  m1 = new Manager(15);
      Employee e2 = new Employee(10);
      Employee i1 = new Intern(2, 800);
      Intern   i2 = new Intern(2, 900);
      Intern   i3 = new Intern(5, 800);


  (1)   Console.WriteLine(e1.Check(m1));
  (2)   Console.WriteLine(m1.Check(e1));
  (3)   Console.WriteLine(m1.Check(m1));
  (4)   Console.WriteLine(i1.Check(e2));
  (5)   Console.WriteLine(i1.Check(i2));
  (6)   Console.WriteLine(i1.Check(i3));
  (7)   Console.WriteLine(e2.Check(i1));

   }
  }
```

א. עבור כל אחת מהפקודות 1-7 כתבו מה יהיה הפלט של הפקודה.

(14 נקודות)

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

ב. נתונה הפקודה: `System.out.println(i1.check(XXX))` (ב-:C#) ``Console.WriteLine(i1.Check(XXX))`` כאשר **XXX** הוא משתנה שהוגדר בפעולה הראשית main במחלקה Driver.

כתבו את כל המשתנים שאפשר לשים במקום XXX כדי שלאחר ביצוע הפקודה יודפס הפלט "IntCheck false".

(4 נקודות)

8. **א.** הפעולה **first** שלפניכם מקבלת הפניה ch לחוליה הראשונה של שרשרת החוליות של מספרים שלמים ומספר שלם x.

:Java

```
public static Node<Integer> first(Node<Integer> ch, int x){
  if(ch == null)
    return new Node<Integer>(x);
  ch.setNext(first(ch.getNext(),x));
  return ch;
}
```

:#C

```
public static Node<int> First(Node<int> ch, int x){
  if(ch == null)
    return new Node<int>(x);
  ch.SetNext(First(ch.GetNext(),x));
  return ch;
}
```

(1) נתונה שרשרת החוליות: **ch ->8→7→9→15→4**

עקבו אחרי זימון הפעולה ch = first(ch, 3) ורשמו איך תיראה השרשרת אחרי ביצוע הפעולה.

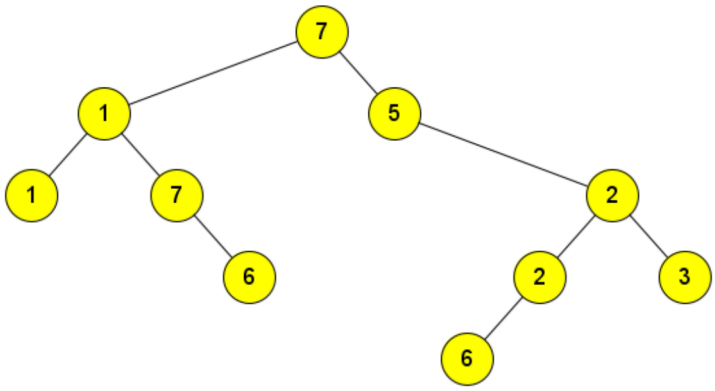
(2) מה מבצעת הפעולה first באופן כללי?

(4 נקודות)

ב. כתבו את הפעולה first בצורה הלא רקורסיבית.

(4 נקודות)

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



ג. הפעולה **second** שלפניכם מקבלת הפניה לשורש של עץ בינארי ומספר שלם וחיובי k.

Java:

```
public static Node<Integer> second(BinNode<Integer> bt, int k){
  Node<Integer> list = third(bt, null, k);
  return list;
 }
public static Node<Integer> third(BinNode<Integer> bt,
                    Node<Integer> list, int k) {
  if(bt!=null)
   {
    if(k>0)
     {
      list = third(bt.getLeft(), list, k-1);
      list = third(bt.getRight(), list, k-1);
     }
    else
      list = first(list, bt.getValue());
   }
   return list;
}
```

C#:

```
public static Node<int> Second(BinNode<int> bt, int k){
  Node<int> list = Third(bt, null, k);
  return list;
 }
public static Node<int> Third(BinNode<int> bt, Node<int> list,
                    int k) {
  if(bt!=null)
   {
    if(k>0)
     {
```


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

```
    list = Third(bt.GetLeft(), list, k-1);
    list = Third(bt.GetRight(), list, k-1);
   }
  else
    list = First(list, bt.GetValue());
 }
 return list;
}
```

(1) עקבו אחרי זימון הפעולה second(bt,3) ורשמו מה תחזיר הפעולה עבור העץ שלפניכם:

(2) מה מבצעת הפעולה second באופן כללי?

(10 נקודות)

9. מערכת הניהול של תהליך הרישום לכנס מדעי כוללת את שתי המחלקות הבאות: המחלקה "מושב" (Session) והמחלקה "ניהול כנס" (ConferenceManager).

למחלקה "מושב" (Session) התכונות הבאות:

- שם המושב – nameSession
- תור שמות המשתתפים – participants
- קיבולת מקסימלית – maxCapacity
- מספר המשתתפים בפועל – numParticipants
- סימן אם המושב הוא "קריטי" – isCritical. המושב הוא קריטי, אם נשארו פחות מחמישה מקומות פנויים.

למחלקה "ניהול כנס" ConferenceManager שתי תכונות: שם הכנס ורשימת מושבים:

Java:

```
public class ConferenceManager {
  private String nameConference;
  private Node<Session> sessions;
}
```

C#:

```
public class ConferenceManager {
  private string nameConference;
  private Node<Session> sessions;
}
```

אפשר להניח שבשתי המחלקות הוגדרו פעולות get לכל התכונות.

א. כתבו במחלקה Session פעולה בונה המקבלת את שם המושב ואת הקיבולת המקסימלית.
הפעולה מאתחלת מושב ללא משתתפים:

Java:

```
public Session (String nameSession, int maxCapacity)
```

C#:

```
public Session (string nameSession, int maxCapacity)
```

(3 נקודות)

ב. ניתן להוסיף משתתף למושב **רק** אם יש מקום פנוי והמשתתף עדיין לא קיים ברשימת המשתתפים הרשומים במושב זה.
כתבו במחלקה Session פעולה המקבלת שם משתתף ומנסה לרשום אותו למושב. אם הרישום אפשרי, הפעולה תוסיף את שם המשתתף לסוף התור, תעדכן את התכונות הרלוונטיות ותחזיר ערך true. אם הרישום אינו אפשרי, הפעולה תדפיס הודעה מתאימה ותחזיר ערך false.
כותרת הפעולה:

Java:

```
public boolean registerParticipant(String name)
```

C#:

```
public bool RegisterParticipant(string name)
```

(7 נקודות)

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

ג. עקב ביקוש רב, הוחלט להחמיר את כללי הרישום למושבים "קריטיים": משתתף יכול להירשם למושב קריטי **רק** אם הוא אינו רשום בשום מושב קריטי אחר.

כתבו במחלקה ConferenceManager פעולה המקבלת שם משתתף ושם מושב. הפעולה תבדוק אם ניתן לרשום את המשתתף למושב. אם כן - הפעולה תבצע רישום ותחזיר ערך true. אם אין מושב רצוי או שהרישום אינו אפשרי, הפעולה תדפיס הודעה מתאימה ותחזיר ערך false.

כותרת הפעולה:

Java:

```
public boolean tryRegister(String nameSession, String name)
```

C#:

```
public bool TryRegister(string nameSession, string name)
```

חובה להשתמש בפעולה registerParticipant.

(8 נקודות)

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