GuyAI
מבית GOOL בגרויות

סוג הבחינה: בחינת גמר ממלכתית
מועד הבחינה: קיץ 2021 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. **יש לענות על מבחן אחד בלבד בהתאם למוסד הלימודים:** מבחן ב- Java (עמוד 2) מבחן ב- C# (עמוד 20) בכל מבחן 11 שאלות. **חלק א' - 45 נקודות** שאלות 1-4: יש לענות על **שלוש** שאלות בלבד. ערך כל שאלה - 15 נקודות. **חלק ב' - 30 נקודות** שאלות 5-8: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה - 15 נקודות. **חלק ג' - 25 נקודות** שאלות 9-11: יש לענות על **שתי** שאלות בלבד. ערך כל שאלה - 12 נקודות. נקודה אחת תינתן על הערכה. **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. לשאלון מצורף נספח ממשקים לבחינה ב-JAVA ונספח ממשקים לבחינה ב-C# (ממשקי המחלקות Stack<T>, Node<T> ו-Queue<T>). בשאלון זה 38 עמודי בחינה ו-4 עמודי נספחים.

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

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

בהצלחה!

השאלות

1. בחנות "קח ולך" מוכרים **מחשבי לוח** (טאבלטים) שונים. כדי לעזור ללקוחות לבחור את הדגם המתאים ביותר, הוחלט למחשב את מחסן המלאי. לשם כך הוגדרו שלוש מחלקות:

המחלקה Tablet המייצגת מחשב הלוח. תכונות המחלקה:

- name - שם חברה המייצרת, מסוג מחרוזת, String
- kind - דגם, מסוג מחרוזת, String
- system - מערכת הפעלה מסוג תו, char ('W' - ווינדוס, 'A' - אנדרואיד, 'iOS' - 'I')
- size - גודל, מסוג מספר ממשי, double
- price - מחיר, מסוג מספר ממשי, double

אפשר להניח שבמחלקה הוגדר בנאי (constructor) המקבל פרמטרים לכל התכונות, פעולות get/set ופעולת toString. **אין צורך לממש את הפעולות**.

(בגרסת C# הטיפוס של name ו-kind הוא string.)

א. כתבו פעולה `public boolean isSame(Tablet other)`

(ב-C#: `public bool IsSame(Tablet other)`)

הפעולה מחזירה true אם השם, הדגם והגודל של העצם המפעיל את הפעולה זהים להשם, הדגם והגודל של other. ולא הפעולה תחזיר false.

(3 נקודות)

המחלקה Store מייצגת מחסן. תכונות המחלקה:

- tablets – מערך עצמים מסוג Tablet. גודל המערך 1000.
- systems – מערך מספרים שלמים. בכל תא במערך נשמר מספר סוגי המחשבים שחנות מציעה, לפי מערכת הפעלה: מספר המכשירים בעלי מערכת ווינדוס, מספר המכשירים בעלי מערכת ההפעלה אנדרואיד ומספר המכשירים בעלי מערכת ההפעלה iOS.

```
public class Store {
private Tablet[] tablets;
private int[] systems;
public Store()
{
 tablets = new Tablet[1000];
 systems = new int[3];
}
….
}
```

```
public class Store {
private Tablet[] tablets;
private int[] systems;
public Store()
{
 tablets = new Tablet[1000];
 systems = new int[3];
}
….
}
```

ב. כתבו פעולה `public boolean addTablet(Tablet tab)`
(ב-C#: `public bool AddTablet(Tablet tab)`)
הפעולה מקבלת מחשב לוח tab ומוסיפה אותו למחסן.

- אם אין מקום להוסיף מוצר חדש, הפעולה תחזיר false.
- אם כבר קיים מוצר בעל אותו שם, דגם וגודל, הפעולה תעדכן את המחיר למחיר הגבוה בין המחיר של המוצר הקיים ובין המוצר שמתקבל כפרמטר ותחזיר true.
- אם לא קיים מוצר כמו tab, הפעולה תוסיף אותו למערך מוצרים, תעדכן את מספר המוצרים במחסן ותחזיר true.

שימו לב: **עצמים במערך צריכים להיות ברצף ללא "חורים" (תאים עם ערך null).**

(6 נקודות)

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

ג. כתבו פעולה `public int sortStore()`
(ב-C#: `public int SortStore()`)
הפעולה מסדרת את מערך המוצרים כך שבתחילת המערך יהיו מחשבי לוח בעלי מערכת הפעלה ווינדוס, אחר כך מכשירים בעלי מערכת הפעלה אנדרואיד ולבסוף מכשירים עם מערכת הפעלה iOS. הפעולה תחזיר את מספר המקומות הפנויים במחסן.
(6 נקודות)

2. כתבו פעולה המקבלת תור של מספרים שלמים. האיברים בתור לא ממוינים ויכולים להופיע כמה פעמים.
הפעולה תחזיר תור חדש הכולל רק את האיברים שמופיעים יותר מפעמיים.
לדוגמה:
עבור תור q1 הבא: [**2**,**5**,**5**,7,**2**,4,1,3,**2**,**5**,**5**,1] הפעולה תחזיר תור חדש [2,5].

3. חברת משלוחים מציעה ללקוחותיה שירות חדש: לאחד מוצרים מסוגים שונים במשלוח אחד.
החברה מתמקדת בשלב הראשון במשלוח בגדים וספרים.
לשם כך הוחלט ליצור פרויקט הכולל מחלקות הבאות:

- המחלקה Clothes עבור בגדים
- המחלקה Shirt עבור חולצות
- המחלקה Dress עבור שמלות
- המחלקה Book עבור ספרים

Java:

```
public class Clothes {
   private String fabric;  // סוג בד
   private String color; // צבע
   private double price; // מחיר
   public double getPrice(){ return this.price;  }
}
public class Shirt extends Clothes {
   private String size; // (L, X,XL,XXL)מידה
}
public class Dress extends Clothes {
   private double length; // אורך בסמ'
   private int size; // מידה
}
public class Book {
   private String name; // שם ספר
   private String author; // מחבר
   private double price; // מחיר
   public double getPrice(){return 0.9*this.price;}
}
```

C#:

```
public class Clothes {
   private string fabric;  // סוג בד
   private string color; // צבע
   private double price; // מחיר
   public double GetPrice(){ return this.price;  }
}
public class Shirt : Clothes {
   private string size; // (L, X,XL,XXL)מידה
}
public class Dress : Clothes {
   private double length; // אורך בסמ'
```


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

```
  private int size; // מידה
}
public class Book {
  private string name; // שם ספר
  private string author; // מחבר
  private double price; // מחיר
  public double GetPrice(){return 0.9*this.price;}
}
```

א. המחלקה Shipping מתארת משלוח. למחלקה יש שתי תכונות:

- address - כתובת משלוח מסוג מחרוזת, String
- arr - מערך הפריטים למסירה.

כתבו את כותרת המחלקה ואת התכונות שלה. ציינו מהו סוג המערך שנבחר והסבירו את בחירתכם.

(5 נקודות)

ב. כתבו פעולה פנימית `public double sum()` המחזירה את הסכום הכולל של הפריטים במשלוח.

(ב-C#: `public double Sum()`)

הערה: אין להוסיף מחלקות או פעולות, אין לשנות את יחסי הירושה בין המחלקות.

(10 נקודות)

4. **הערה:** בשאלה זו אין קשר בין הסעיפים!

א. נתונות שתי מחלקות A, B:

Java:

```
public class A{
}
public class B extends A{
 public B(){
  System.out.println("B constructor");
 }
}
```

C#:

```
public class A{
}
public class B : A{
 public B(){
  Console.WriteLine("B constructor");
 }
}
```

לפניכם ארבעה היגדים. קבעו לגבי כל אחד מהם אם הוא נכון או אינו נכון ונמקו את קביעתכם.

(1) הבנאי של B יקרא לבנאי הריק של A (בנאי ללא פרמטרים, בנאי ברירת מחדל, default constructor)

(2) הבנאי הריק של Object לא ייקרא כי ל- A אין זימון מפורש של בנאי זה

(3) הבנאי הריק של Object ייקרא לאחר הדפסת המחרוזת "B constructor"

(4) הבנאי הריק של B יקרא לבנאי הריק של A ואז תודפס המחרוזת

(5 נקודות)

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

ב. נתונות שלוש מחלקות A, B ו- C, כאשר מחלקה B יורשת מהמחלקה A והמחלקה C יורשת מהמחלקה B.

כמו כן נתון שלמחלקה A מוגדרת התכונה value.

לפניכם ארבעה היגדים. קבעו לגבי כל אחד מהם אם הוא נכון או אינו נכון ונמקו את קביעתכם.

(1) מתוך המחלקה C אין גישה לתכונה value של המחלקה A אפילו ש- value מוגדרות כ- protected מכיוון שמדובר בשתי רמות של ירושה.

(2) מתוך המחלקה C יש גישה לתכונה value רק בתנאי שהתכונה value במחלקה A מוגדרת כ- protected.

(3) מתוך המחלקה C יש גישה לתכונה value של המחלקה A באמצעות הפקודה super.super.value.

(4) מתוך המחלקה C יש גישה לתכונה value מכיוון שהמחלקה C יורשת בעקיפין מהמחלקה A, בלי קשר להרשאת הגישה של התכונה.

(5 נקודות)

ג. נתונה הפקודה הבאה העוברת שלב קומפילציה (הידור):

```
((A)b).myFun();
```

(ב-MyFun.((A)b)`:C#();`)

כאשר b הוא מצביע מסוג המחלקה B.

לפניכם ארבעה היגדים. קבעו לגבי כל אחד מהם אם הוא נכון או אינו נכון ונמקו את קביעתכם.

(1) הפעולה myFun מוגדרת במחלקה A.

(2) B היא מחלקה היורשת את המחלקה A.

(3) A היא מחלקה היורשת את המחלקה B.

(4) הפעולה myFun מוגדרת גם במחלקה A וגם במחלקה B.

(5 נקודות)

5. נתונות שתי המחלקות הבאות:

Java:

```
class A{
   public void f() {
      System.out.println("A.f");
   }
   public void g() {
      f();
   }
}
public class B extends A{
   public void f(){
      System.out.println("B.f");
   }
   public void g() {
      System.out.println("B.g");
   }
   public void superG() {
      super.g();
   }
}
```

C#:

```
class A{
   public virtual void F() {
      Console.WriteLine("A.F");
   }
   public virtual void G() {
      F();
   }
}
public class B : A{
   public override void F(){
      Console.WriteLine("B.F");
   }
   public override void G() {
      Console.WriteLine("B.G");
   }
   public void SuperG() {
      base.G();
   }
}
```

נתון קטע הקוד הבא:

Java:

```
A a = new A();
B b = new B();
A ab = new B();
a.f();
ab.f();
b.f();
a.g();
ab.g();
b.g();
((B)(ab)).superG();
b.superG();
```

C#:

```
A a = new A();
B b = new B();
A ab = new B();
a.F();
ab.F();
b.F();
a.G();
ab.G();
b.G();
((B)(ab)).SuperG();
b.SuperG();
```

א. עקבו אחרי ביצוע קטע הקוד ורשמו מה יהיה הפלט.

(8 נקודות)

ב. האם אפשר להוסיף לקטע הקוד את שתי הפקודות הבאות?

```
a.superG();
((B)a).superG();
```

(ב-`a.SuperG();` ב-C#: `((B)(a)).SuperG();`)

הסבירו את תשובתכם.

(7 נקודות)

6. נתונות שתי המחלקות הבאות: Apple ו- Banana

Java:

```
public class Apple {
    private int weight;
    public Apple (int w) {
        weight = w;
    }
    public int getWeight () {
        return weight;
    }
    public boolean equals(Apple other) {
        return ((other!=null) &&
            (weight == other.weight));
    }
}

public class Banana {
    private int weight;
    public Banana (int w) {
        weight = w;
    }
    public int getWeight () {
        return weight;
    }
    public boolean equals (Object other) {
        return ((other != null) &&
            (other instanceof Banana) &&
            (weight == ((Banana)other).weight));
    }
}
```

C#:

```
public class Apple {
   private int weight;
   public Apple (int w) {
      weight = w;
   }
   public int GetWeight () {
      return weight;
   }
   public bool Equals(Apple other) {
      return ((other!=null) &&
```


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

```
            (weight == other.weight));
    }
}

public class Banana {
    private int weight;
    public Banana (int w) {
        weight = w;
    }
    public int GetWeight () {
        return weight;
    }
    public override bool Equals (Object other) {
        return ((other != null) &&
            (other is Banana) &&
            (weight == ((Banana)other).weight));
    }
}
```

א. האם קיימת העמסה (Overloading) או דריסה (Overriding) של הפעולה equals במחלקות Apple ו- Banana? הסבירו את תשובתכם.

(5 נקודות)

ב. נתונה המחלקה Program הבאה:

Java:

```
public class Program {
   public static void main (String[] args) {
      System.out.println ("·····");
      Apple a1 = new Apple (10);
      Object a2 = new Apple (10);
      Banana b1 = new Banana (10);
      Object b2 = new Banana (10);
      ·····
 }
}
```

C#:

```
public class Program {
   public static void main (string[] args) {
      Console.WriteLine ("·····");
      Apple a1 = new Apple (10);
      Object a2 = new Apple (10);
      Banana b1 = new Banana (10);
      Object b2 = new Banana (10);
      ·····
 }
}
```

בהתייחס לכל אחת מהשורות הבאות, כתבו מה יקרה בעקבות הוספתה לשיטה main שלעיל, לאחר ההצהרות על האובייקטים (במקום שמסומן בכוכביות *).

Java:

```
1. System.out.println (a1.weight);
2. System.out.println (((Banana)a2).getWeight());
3. System.out.println (a1.equals(a2));
4. System.out.println (a2.equals(a1));
5. System.out.println (b1.equals(b2));
6. System.out.println (b2.equals(b1));
7. System.out.println (a1.equals((Banana)b2));
8. System.out.println (a1.equals((Apple)a2));
9. System.out.println (b1.equals((Apple)a2));
10. System.out.println (b1.equals((Banana)a2));
```


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

C#:

```
1. Console.WriteLine (a1.weight);
2. Console.WriteLine (((Banana)a2).GetWeight());
3. Console.WriteLine (a1.Equals(a2));
4. Console.WriteLine (a2.Equals(a1));
5. Console.WriteLine (b1.Equals(b2));
6. Console.WriteLine (b2.Equals(b1));
7. Console.WriteLine (a1.Equals((Banana)b2));
8. Console.WriteLine (a1.Equals((Apple)a2));
9. Console.WriteLine (b1.Equals((Apple)a2));
10. Console.WriteLine (b1.Equals((Banana)a2));
```

הערה:

אם הוספתם פקודת קוד גורמת לשגיאה יש להסביר מהי השגיאה (שגיאת קומפילציה או שגיאת זמן ריצה).

(10 נקודות)

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

7. אפשר לייצג את הזיכרון במחשב באמצעות רשימה (שרשרת חוליות), כאשר כל איבר ברשימה מכיל את גודל קטע הזיכרון (בבתים) ומציין אם הקטע פנוי או תפוס. כל מקטע מיוצג באמצעות המחלקה Data:

Java:

```
public class Data{
 private boolean free;
 private int size;
 //constructor
 public Data (int size) {
   this.free = true;
   this.size = size;
  }
 public boolean isFree(){ return free; }
 public int getSize()    { return size; }
 public void setFree(boolean free){this.free = free; }
 public void setSize(int size){ this.size = size;}
}
```

C#:

```
public class Data{
 private bool free;
 private int size;
 //constructor
 public Data (int size) {
   this.free = true;
   this.size = size;
  }
 public bool IsFree()   { return free; }
 public int GetSize()    { return size; }
 public void SetFree(bool free)   {this.free = free; }
 public void SetSize(int size){ this.size = size;}
}
```

הזיכרון כולו מיוצג באמצעות האובייקט הבא:

```
public class Memory {
 private Node<Data> start;
 public Memory(int totalSize) {
    this.start = new Node<Data>(new Data(totalSize));
  }
}
```

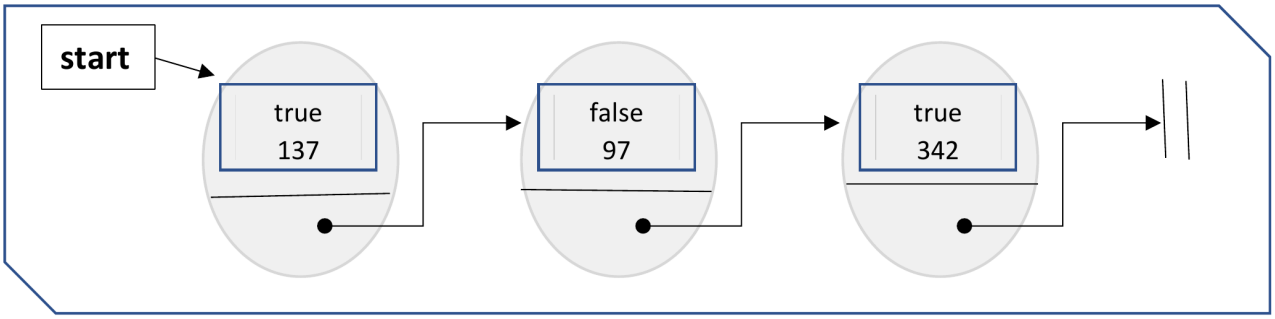

נכתב ונבדק על ידי צוותי GOOL בגרויות ו-GuyAI · www.guyai.co.il

(בגרסת C# קוד המחלקה Memory זהה.)

להלן דוגמה לזיכרון שיש בו מקטע פנוי בגודל 137, אחריו מקטע תפוס בגודל 97 ולבסוף מקטע פנוי בגודל 342 (איור לשאלה 7).

הערה: לא ייתכן שיהיו שני איברים סמוכים במצב true. אבל זה כן ייתכן לגבי מצב false.

זיכרון מחשב נמצא ב"מצב מסוכן" (Dangerous State) אם כמות הזיכרון הפנוי יורדת מתחת 10% מכמות זיכרון הכללי.

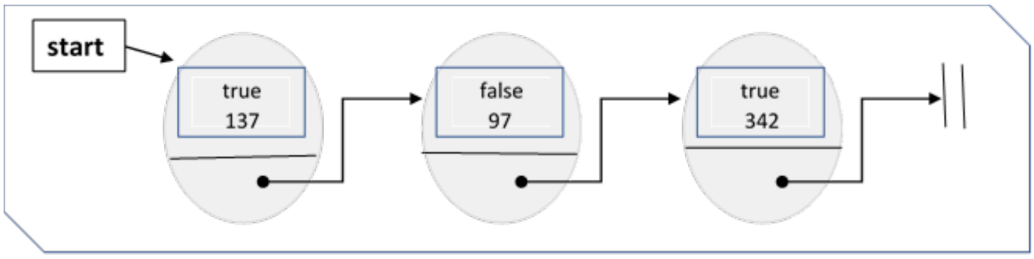


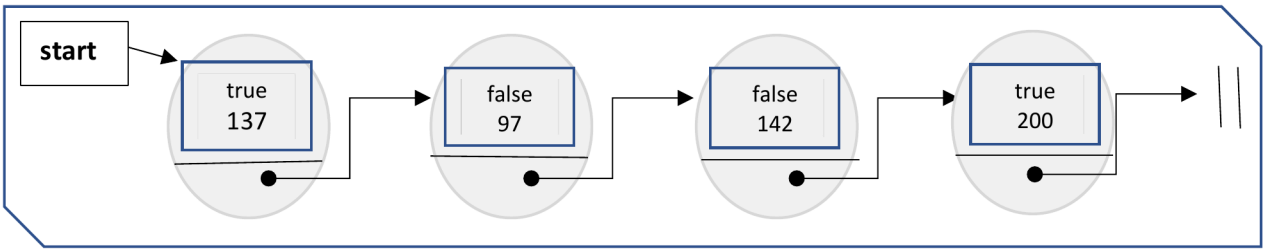
א. כתבו פעולה פנימית במחלקה Memory, הבודקת את מצב הזיכרון והמחזירה true, אם הוא ב"מצב מסוכן", ולא הפעולה מחזירה false.

(5 נקודות)

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

כאשר המעבד צריך להקצות זיכרון בגודל מסוים (num) הוא יכול להשתמש באלגוריתם First Fit.
האלגוריתם מחפש את מקטע הזיכרון הפנוי הראשון שיכול להכיל את num (כלומר, שגודלו הוא לפחות num), ומקצה לו מקום בזיכרון וקובע שמקטע הזיכרון במצב תפוס (false).
אם הזיכרון נמצא ב"מצב מסוכן" האלגוריתם אינו מבצע דבר.





ב. כתבו במחלקה Memory פעולה הממששת את האלגוריתם. כותרת הפעולה:

`boolean firstFit(int num)`

(ב-C#: `bool FirstFit(int num)`)

פעולה זו מקבלת כפרמטר את num שהוא גודל הזיכרון הנדרש, מאתרת את המקטע הראשון הפנוי שגדול או שווה ל- num, משנה את הרשימה באמצעות הכנסת חוליה המייצגת מקטע תפוס בגודל num ומעדכנת את גודל המקטע הפנוי. הפעולה מחזירה true אם נמצא מקום כזה ו- false אם לא נמצא מקום. אם הזיכרון נמצא ב"מצב מסוכן" הפעולה מחזירה false.

לדוגמה:

אם לפני הזימון של הפעולה מצב הזיכרון הוא: מקטע פנוי (true) בגודל 137, מקטע תפוס (false) בגודל 97, מקטע פנוי (true) בגודל 342,

אז אחרי הזימון firstFit(142) מצב הזיכרון ישתנה למצב הבא: פנוי 137, תפוס 97, תפוס 142, פנוי 200, והפעולה תחזיר true.

(שני המצבים, לפני הזימון ואחריו, מוצגים באיורים לסעיף ב.)

(10 נקודות)

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

8. נתונה מחלקה IntList הבאה:

Java:

```
public class IntList {
  private Node<Integer> head;
  public IntList( ) {
    head = null;
  }
  public void add(int a) {
    head = new Node<Integer> (a, head);
  }
  public String toString(){
    String s = "{";
    Node<Integer> h = head;
    while(h.getNext()!= null){
      s += h.getValue()+",";
      h = h.getNext();
    }
    return s + h.getValue()+"}";
  }
  public  boolean what1 (IntList list) {
    Node<Integer> h1 = head;
    Node<Integer> h2 = list.head;
    while ((h1 != null) && (h2 != null))  {
      if (h1.getValue()!= h2.getValue())
        return false;
      h1 = h1.getNext();
      h2 = h2.getNext();
    }
    return true;
  }
  public boolean what2 (IntList list) {
    Node<Integer> h1 = head;
    while (h1 != null) {
      boolean found = false;
      Node<Integer> h2 = list.head;
      while ((h2 != null) && (!found)) {
        if (h1.getValue()== h2.getValue())
          found = true;
        h2 = h2.getNext();
      }
      if (!found)
        return false;
      h1 = h1.getNext();
    }
    return true;
```


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

```
  }
}
```

C#:

```
public class IntList {
  private Node<int> head;
  public IntList( ) {
    head = null;
  }
  public void Add(int a) {
   head = new Node<int> (a, head);
  }
  public override string ToString(){
    string s = "{";
    Node<int> h = head;
    while(h.GetNext()!= null){
      s += h.GetValue()+",";
      h = h.GetNext();
    }
    return s + h.GetValue()+"}";
  }
  public  bool What1 (IntList list) {
    Node<int> h1 = head;
    Node<int> h2 = list.head;
    while ((h1 != null) && (h2 != null))  {
      if (h1.GetValue()!= h2.GetValue())
        return false;
      h1 = h1.GetNext();
      h2 = h2.GetNext();
    }
    return true;
  }
  public bool What2 (IntList list) {
    Node<int> h1 = head;
    while (h1 != null) {
      bool found = false;
      Node<int> h2 = list.head;
      while ((h2 != null) && (!found)) {
        if (h1.GetValue()== h2.GetValue())
          found = true;
        h2 = h2.GetNext();
      }
      if (!found)
        return false;
```

```
        h1 = h1.GetNext();
      }
      return true;
    }
}
```

נתון קטע קוד של הפעולה main במחלקה Program:

Java:

```
IntList testList = new IntList();
testList.add(2); testList.add(3); testList.add(6);
testList.add(1); testList.add(4);
System.out.println(testList);
```

C#:

```
IntList testList = new IntList();
testList.Add(2); testList.Add(3); testList.Add(6);
testList.Add(1); testList.Add(4);
Console.WriteLine(testList);
```

א. מהו הפלט של קטע הקוד (אין צורך במעקב מפורט)?

(3 נקודות)

ב. בעבור עצם שנוצר בסעיף א' הביאו דוגמה של עצם אחר מטיפוס IntList בשם **list** הכולל ארבעה איברים לפחות, כך ש:

(1) זימון testList.what1(list) יחזיר **true** והזימון testList.what2 (list) יחזיר **true**
אם אי אפשר להביא דוגמה כזו, ציינו זאת והסבירו מדוע.

(2) זימון testList.what1(list) יחזיר **true** והזימון testList.what2 (list) יחזיר **false**
אם אי אפשר להביא דוגמה כזו, ציינו זאת והסבירו מדוע.

(3) זימון testList.what1(list) יחזיר **false** והזימון testList.what2 (list) יחזיר **true**
אם אי אפשר להביא דוגמה כזו, ציינו זאת והסבירו מדוע.

(4) זימון testList.what1(list) יחזיר **false** והזימון testList.what2 (list) יחזיר **false**
אם אי אפשר להביא דוגמה כזו, ציינו זאת והסבירו מדוע.

(8 נקודות)

ג. מה מבצעות הפעולות what1 ו- what2 באופן כללי?

(4 נקודות)

9. בבית חולים מסוים פותח פרויקט לניהול הצוות הרפואי. לכל עובד (**Employee**) בביה"ח יש שם ומספר עובד. מספר העובד הוא **מספר ייחודי הניתן באופן אוטומטי** עם הוספת עובד חדש למאגר העובדים כך שהעובד הראשון יקבל את המספר 1, העובד השני 2 וכך הלאה.

לאחות (**Nurse**) יש את התכונות:

- num - מספר עובד, מטיפוס שלם, int.
- name - שם, מטיפוס מחרוזת, String.
- type - סוג (מעשית, מוסמכת), מטיפוס מחרוזת, String.

לרופא (**Doctor**) יש את התכונות:

- num - מספר עובד, מטיפוס שלם, int.
- name - שם, מטיפוס מחרוזת, String.
- specialization - התמחות (רופא לב, רופא מרדים, רופא מנתח וכו'), מטיפוס מחרוזת, String.

ראש צוות (**Supervisor**) - ראש צוות הוא רופא שאחראי על צוות של __עד עשרה__ רופאים ואחיות:

- arr - מערך של אנשי הצוות הכפופים לראש הצוות.
- current - מספר אנשי הצוות בפועל, מטיפוס שלם, int.

(בגרסת C# הטיפוס של המחרוזות הוא string.)

א. שרטטו תרשים UML המתאר את הקשר בין המחלקות **Employee, Nurse, Doctor, Supervisor** באופן המתאים ביותר לעקרונות של תכנות מונחה עצמים.

(2 נקודות)

ב. לכל אחת מהמחלקות **Employee, Nurse, Doctor, Supervisor** כתבו:

- כותרת המחלקה.
- תכונות.
- פעולה בונה – הפעולה הבונה של כל מחלקה מקבלת את כל הפרמטרים הנדרשים.

הערה:

יש לשים לב שמספר העובד הוא מספר ייחודי, שנוצר באופן אוטומטי, ואין להעבירו כפרמטר לפעולה בונה!

(4 נקודות)

המחלקה **AllEmployees** מיועדת לנהל את כל העובדים בבית החולים. תכונות המחלקה:

- arr - מערך של כל אנשי הצוות. בבית חולים יכולים לעבוד לא יותר מ- 200 אנשי צוות.
- current - מספר אנשי הצוות בפועל, מטיפוס שלם, int.

ג. כתבו את כותרת המחלקה ואת התכונות של המחלקה **AllEmployees** והוסיפו את הפעולות הפנימיות הבאות:

- פעולה בשם numSupervisor המחזירה את מספר ראשי כל הצוותים.
- פעולה בשם getNewNurse המקבלת את סוג האחות ומחזירה את האחות מהסוג הנדרש שהצטרפה אחרונה למאגר העובדים (מספר העובד שלה מקסימלי). אם אין אחות מהסוג הנדרש, הפעולה תחזיר null.

(ב-NumSupervisor: C# ו-GetNewNurse.)

הערה: הניחו שהפעולות get ו- set מוגדרות בעבור כל תכונה בכל אחת מהמחלקות.

(6 נקודות)

10. במכללה החליטו להוסיף למערכת הממוחשבת אפשרות לשמור את הציונים הסופיים (הממוצע המשוקלל של כל שנות הלימודים) של כל הסטודנטים הלומדים ולמדו במוסד.

לשם כך הוגדרו המחלקות Student ו- GradesFile.

כל סטודנט מאופיין באמצעות מספר סטודנט (studentId) שמורכב משמונה ספרות וציון (grade).

מכיוון שמרחב מספרי הסטודנטים גדול, הוחלט שהציונים יישמרו ב- 100 רשימות, כדי לאפשר חיפוש מהיר.

מאגר ציוני הסטודנטים (GradesFile) מאופיין באמצעות מערך בגודל 100 של **אוספים** סטודנטים.

תהליך האחסון במאגר ייעשה באופן הבא:

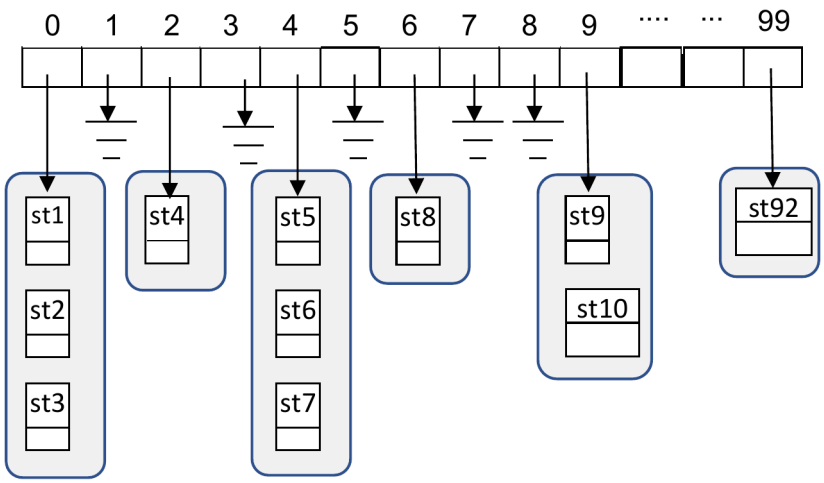
לוקחים את שתי __הספרות האמצעיות__ של מספר הסטודנט. אלה יוצרות מספר K בין 0 ל- 99.

מוסיפים את הסטודנט למאגר לאוסף הנמצא במקום K במערך.

לדוגמה:

אם מספר הסטודנט הוא **12345678** הסטודנט ייכנס למאגר לרשימה במקום **45** במערך.

המבנה ייראה כך (איור לשאלה 10):



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

א. כתבו את כותרת המחלקות Student ו- GradesFile ואת התכונות שלהן.

הערה: יש לבחור מבנה נתונים מתאים לשמירת אוסף נתונים לא מוגבל (תור, מחסנית, שרשרת חוליות).

(2 נקודות)

ב. ממשו במחלקה Student את הפעולה `int getCode()`.

(ב-C#: `int GetCode()`)

הפעולה מחזירה מספר המורכב משתי ספרות האמצעיות של מספר סטודנט (studentID).

__לדוגמה__: עבור המספר **12345678** הפעולה תחזיר **45**.

(2 נקודות)

לפניכם חלק מהפעולות במחלקה GradesFile. **אין צורך לממש את הפעולות!**

פעולה	תיאור
`Student getStudent(int k)`	הפעולה מחזירה את הסטודנט הראשון באוסף במקום ה- k במערך. אם אוסף במקום k ריק או k לא נמצא בגבולות המערך, הפעולה תחזיר null.
`boolean isEmpty(int k)`	הפעולה מחזירה "אמת" אם אוסף הסטודנטים במקום ה- k ריק, ו"שקר" אם לא. אם k לא נמצא בגבולות המערך, הפעולה מחזירה "אמת".
`boolean listIsGood(int k)`	הפעולה מחזירה "אמת" אם כל הסטודנטים הנמצאים באוסף במקום ה- k במערך, מתאימים למקום זה על פי ה- studentId, ו"שקר" אם לא. אם k לא נמצא בגבולות המערך או האוסף במקום ה- k ריק, הפעולה מחזירה "אמת".

(ב-C#: GetStudent, IsEmpty, ListIsGood, ו-bool במקום boolean.)

ג. ממשו את הפעולה `void moveStudent(int k, int j)`.

(ב-C#: `void MoveStudent(int k, int j)`)

פעולה	תיאור
`void moveStudent(int k, int j)`	הפעולה מעבירה את הסטודנט הראשון באוסף שמיקומו k במערך להיות סטודנט אחרון באוסף שמיקומו j במערך. אם האוסף במקום ה- k ריק או k או j לא נמצאים בגבולות המערך, הפעולה לא מבצעת דבר.

(4 נקודות)

ד. במהלך הכנסת הנתונים למערכת קרתה תקלה, ובעקבות כך חל בלבול ולא כל הסטודנטים הוכנסו למקומות המתאימים לפי מספר הסטודנט שלהם.

כתבו __פעולה חיצונית__ המקבלת את מאגר המידע (הפניה לעצם מסוג GradesFile) ומעדכנת אותו כך שבכל תא במערך יהיה אוסף סטודנטים בעלי מספרי סטודנט המתאימים למספר התא במערך (על פי שיטת האחסון שתוארה בתחילה).

הערה:

בפתרון של סעיף ד' יש להשתמש בפעולות הנתונות של המחלקות Student ו-GradesFile בלבד!

אין להשתמש בפעולות של מבנים אחרים ואין להניח על קיומן של הפעולות האחרות במחלקות Student ו- GradesFile.

(4 נקודות)

11. נתונות שלוש מחלקות הבאות: One, Two, Driver

Java:

```
public class One {
   private int num;
   private char ch;
   public One() {num = 2;  ch = 'G'; }
   public One(int n) {  num = n;  ch = 'M'; }
   public One(int n, char c) {  num = n; ch = c;  }
   public One(One other){
      num = other.num;
      ch = other.ch;
   }
   public int getNum()   { return num;  }
   public char getCh()   { return ch; }
   public void inc(){ num ++; ch ++; }
   public String toString(){
      String s = "";
      for (int i = 0; i < num; i++)
         s += ch;
      return s;
   }
}// end of a class One

public class Two extends One {
 private One first;
 public Two()   {  super();  first = new One();}
 public Two(int n) {  super(n);  first = new One(); }
 public Two(One other) {
  super(); first = new One(other);
 }
 public Two(One other, int n) {
  super(other); first = new One(n);
 }
 public void inc(){ first.inc(); }
 private int what (int n, int m) {
  if(n > m )return n;
  return m;
 }
 private char what (char ch1, char ch2){
 if(ch1<ch2) return ch1;
 return ch2;
 }
 public One makeOne(){
 return
   new One(what(first.getNum(),getNum()),
```


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

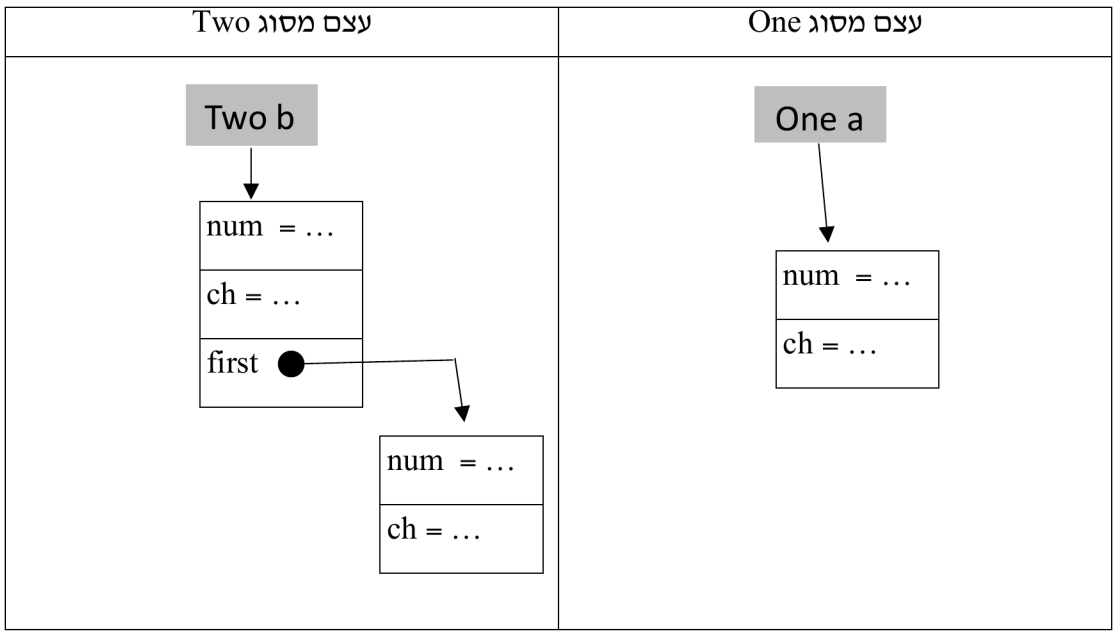
```
        what(first.getCh(), getCh()));
 }
 public String toString(){ return first.toString();}
}// end of a class Two
public class Driver{
  public static void main(String[] args){
    One x1 = new One(4, 'E');
    One x2 = new One(3);
    Two y1 = new Two(x1);
    One x3 = new Two(5);
    System.out.println("x1 before "+x1);
    x1.inc();
    System.out.println("x1 after "+x1);
    System.out.println("x2 " + x2);
    System.out.println("y1 " + y1);
    System.out.println("x3 " + x3);
    Two y2 = new Two(y1, 1);
    System.out.println("y2 " + y2);
    One x4 = y2.makeOne();
    System.out.println("x4 " + x4);
  }
}// end of a class Driver
```

C#:

```
public class One {
  private int num;
  private char ch;
  public One() {num = 2;  ch = 'G'; }
  public One(int n) {  num = n;  ch = 'M'; }
  public One(int n, char c) {  num = n; ch = c;  }
  public One(One other){
    num = other.num;
    ch = other.ch;
  }
  public int GetNum()  { return num;  }
  public char GetCh()  { return ch; }
  public virtual void Inc(){ num ++; ch ++; }
  public override string ToString(){
    string s = "";
    for (int i = 0; i < num; i++)
      s += ch;
    return s;
  }
}// end of a class One
```

```
public class Two : One {
 private One first;
 public Two(): base()   { first = new One(); }
 public Two(int n): base(n) { first = new One(); }
 public Two(One other): base() {
  first = new One(other);
 }
 public Two(One other, int n): base(other) {
  first = new One(n);
 }
 public override void Inc(){ first.Inc(); }
 private int What (int n, int m) {
  if(n > m )return n;
  return m;
 }
 private char What (char ch1, char ch2){
 if(ch1<ch2) return ch1;
 return ch2;
 }
 public One MakeOne(){
 return
    new One(What(first.GetNum(),GetNum()),
         What(first.GetCh(), GetCh()));
 }
 public override string ToString(){ return first.ToString();}
}// end of a class Two
public class Driver{
  public static void Main(string[] args){
     One x1 = new One(4, 'E');
     One x2 = new One(3);
     Two y1 = new Two(x1);
     One x3 = new Two(5);
     Console.WriteLine("x1 before "+x1);
     x1.Inc();
     Console.WriteLine("x1 after "+x1);
     Console.WriteLine("x2 " + x2);
     Console.WriteLine("y1 " + y1);
     Console.WriteLine("x3 " + x3);
     Two y2 = new Two(y1, 1);
     Console.WriteLine("y2 " + y2);
     One x4 = y2.MakeOne();
     Console.WriteLine("x4 " + x4);
  }
}// end of a class Driver
```


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



א. עקבו אחרי הביצוע של הפעולה main ורשמו מה יהיו ערכי התכונות של כל עצם שנוצר במהלך הביצוע.

יש להשתמש באיורים הבאים להצגת העצמים (איור לשאלה 11):

(5 נקודות)

ב. מהו הפלט של הפעולה main?

(ב-Main: C#)

(7 נקודות)

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

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