

פתרון הבחינה בתוכנה 1
אוהד ברזילי, סיון טולדו, מיכל עוזרי-פלאטו, ליאור שפירא
מועד א' סמסטר א' תשס"ז, 11 בפברואר 2007
 משך הבחינה שלוש שעות.

יש לענות על כל השאלות. בשאלות שבהן יש צורך לנמק, תשובה ללא נימוק לא תזכה באף נקודה.

יש לענות על כל השאלות בגוף הבחינה במקום המיועד לכך. המקום המיועד מספיק לתשובות מלאות. יש לצרף את טופס המבחן למחברת הבחינה. מחברת ללא טופס עזר תפסל. תשובות במחברת הבחינה לא תיבדקנה.

יש למלא מספר סידורי (מספר מחברת) ומספר ת"ז על כל דף של טופס הבחינה.

אסור השימוש בחומר עזר כלשהו, כולל מחשבוניו או כל מכשיר אחר פרט לעט. בהצלחה!

לשימוש הבודקים

24		4 6	4 5	4 4	4 3	4 2	4 1	1
20				5 ד	5 ג	5 ב	5 א	2
28	4	4 ו	4 ה	4 ד	4 ג	4 ב	4 א	3
28	4	4 ו	4 ה	4 ד	4 ג	4 ב	4 א	4
100	סה"כ							

שאלה 1

להלן הקוד של מחלקה D (במידת הצורך קיימת פקודת import מתאימה למחלקה B).

```
public class D{
    public static void main (String[] args){
        new B();
    }
}
```

בכל חלק של שאלה זו, יש להסביר האם הקוד של שלושת המחלקות A, B ו-D מתקמפל או שהקומפילר מדווח על שגיאה. אם לדעתך הקוד לא מתקמפל, יש להסביר מדוע. אם לדעתך הקוד מתקמפל, תאר/י מה קורה שמחלקה D מורצת.

```
1. public abstract class A{
    private int[] arr = {1,2,3};
    protected A(){ f(arr); }
    protected abstract void f(int[] arr);
}

public class B extends A{
    public B(){ System.out.println("B"); }
    public void f(int[] kuki){
        System.out.println();
        System.out.println(kuki[3]);
    }
}
```

יזרק חריג בזמן ריצה

`ArrayIndexOutOfBoundsException in System.out.println(kuki[3]);`

```
2. package aaa;
    class A{}

    package bbb;
    import aaa.A;
    public class B extends A{
        public B(){ System.out.println("B"); }
    }
```

`aaa.A is not visible to bbb.B` : שגיאת קומפילציה:

```

3. public class A{
    public final String s = "1";
    public A() { System.out.println(g()+s); }
    public static String g(){ return "A";}
}

public class B extends A{
    public String s= "2";
    public static String g(){ return "B";}
}

```

A1 (static methods and variables depend on static type)

```

4. public class B{
    private int i;
    private class BB{
        private void f(){ System.out.println(i); }
    }
    public B(){
        (new BB()).f();
    }
}

```

0 (full visibility between a class and its nested class, default initialization of fields)

```

5. public class A{
    public A(){ f( 1 ); }
    public void f(Integer i){System.out.println("A"+i/2);}
}

public class B extends A{
    public void f(int i){ System.out.println("B"+i/2);}
}

```

A0 (overloading, "/" operator returns an integer)

```

6. public class A{
    private String s = "1";
    public String f(A a){
        a.s ="2";
        a = new A();
        return a.s;
    }
    public String toString(){return s;}
}

public class B{
    public B(){
        final A a = new A();
        System.out.println(a.f(a)+a);
    }
}

```

12 (arguments are passed by value, reference arguments)

שאלה 2

בשאלה הזו יש לענות על הסעיפים מילולית, לא לספק קוד.
 א. בהינתן קוד SWT המצורף, מה צריך להוסיף כדי שיתבצע קטע קוד בכל פעם שהמשתמש ילחץ על הפקודה בתפריט?

```
MenuItem newItem = new MenuItem(fileMenu, SWT.PUSH);
newItem.setText("New &Address Book\tCtrl+N");
```

יש צורך לקרוא ל-`addSelectionListener` ולהעביר לו מופע של מחלקה אנונימית המממשת את `SelectionListener`. הקוד בשירות `widgetSelected` יתבצע בכל פעם שבוחרים את התפריט.

ב. המטלה האחרונה בתרגיל 'עולם פראי' הייתה להחליף קבצי חיות עם תלמיד אחר. אילו קבצים הייתם צריכים לקחת מן התלמיד האחר כדי להיות מסוגלים להשתמש בחיות שלו ואילו שינויים בקוד שלכם נדרשו?

קבצי ה-`Class` של מחלקות `Animal` ו-`Species` היו דרושות בכדי להחליף חיות בין הסטודנטים. השינוי היחיד שהיה אמור להידרש בקוד היה יצירת מופע ה-`Species`. כל השאר לא היה אמור להשתנות.

```
public interface IIPAddress {
    public String toString();
    public boolean equals(IIPAddress ip);
    public short getOctet(int index);
    public boolean isPrivateNetwork();
    public void mask(IIPAddress mask);
```

ג. בתרגיל 5 מימשתם את המנשק `IIPAddress` במספר צורות (עם מחרוזות, מערך של `short` ו-`int`). עליכם להגדיר מחלקת אב מופשטת משותפת לכל המימושים. אילו מן השירותים הבאים (שבתיבה משמאל) ניתן לממש במחלקה המשותפת כדי למנוע שכפול קוד? נמקו.

במחלקה המופשטת המשותפת ניתן להשתמש בשירות `getOctet` כדי לממש את `toString`, `equals`, `isPrivateNetwork`, את `getOctet` עצמו יש לממש בכל מימוש מוחשי של המנשק, בהתאם למימוש. מאחר והפקודה `mask` הינה פקודה המשנה את כתובת ה-`IP`, גם אותה יש לממש בכל מימוש מוחשי.

על התשובה שבעזרת `toString` ניתן לממש את שאר השאליות הורדה נקודה

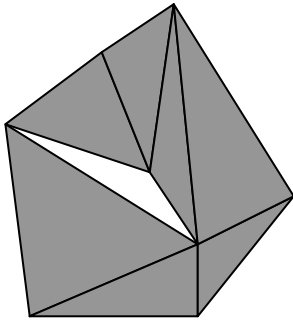
ד. בתרגיל 9 היה עליכם לממש את המנשק `SortedSet`, אוסף של עצמים ממוינים ושונים אחד מן השני. בהינתן מימוש פשוט `SimpleSortedSet` תארו כיצד עובד האיטרטור של `MergedSortedSet` (המכיל את החיתוך של שני אוספים A ו-B אותם הוא מקבל בבנאי).

בהינתן שני איטרטורים של סטים ממוינים נבצע את האלגוריתם הבא: בתחילה שני האיטרטורים מצביעים לתחילת הרשימות הממוינות של A ו-B. בכל קריאה ל-`next` נחפש את האבר הבא המשותף (ז"א נבדוק אם האבר הבא ב-A ו-B זהה, אם לא נקדם את הקטן מביניהם ונשווה שוב, נחזור על כך עד שנמצא אבר משותף).

תשובה שבה לכל איבר בקבוצה A מחפשים אם הוא קיים בקבוצה B , לא קיבלה ניקוד כלל, משום שאלגוריתם הבניה הזה אינו משתמש כלל בתכונות האיטרטורים או מיון האברים.

תשובה שבה בונים רשימה חדשה ממוינת של החיתוך ומחזירים איטרטור פשוט שלה קיבלה ניקוד חלקי רק אם בנית הרשימה החדשה השתמשה באלגוריתם המיזוג לעיל.

שאלה 3



בחבילה מסויימת נתונה המחלקה Point שמייצגת נקודות במישור.

```
// Abstract state: a point in the plane
public class Point {
    public Point(double x, double y) {...}
    public double getX() {...}
    public double getY() {...}
}
```

כמו כן נתון מנשק Mesh שמייצג חלוקה של תחום במישור למשולשים, בדומה לחלוקה שבאיור. המשולשים של Mesh יכולים לשתף קודקודים וצלעות, אבל אסור לשני משולשים לכסות שטח משותף.

```
// Abstract state: a set of triangles T1, T2,...,Tn
// such that no point belongs to the interior of two
// triangles
Public interface Mesh {

    // return the number of triangles in the mesh
    public int size();

    // returns a 3-element array of points that are
    // the nodes of triangle Ti
    public Point[] get(int i);
}
```

א. הגדר/הגדירי מימוש של המנשק הזה. המימוש חייב להשתמש במחלקה פנימית פרטית Triangle שמייצגת משולש בודד. פרט לכך, המימוש חייב להיות פשוט ככל האפשר. יש להגדיר את השדות, ואת השירותים size ו-get, אבל אין צורך להגדיר בנאים או שירותים אחרים.

```
public class SimpleMesh Implements Mesh {
    private static class Triangle {
        private Point[] nodes;
        private Triangle(Point[] nodes) { this.nodes = nodes; }
    }

    private List<Triangle> triangles = new List<triangle>;
    ... // constructors
    public int size() { return triangle.size(); }
    public Point[] get(int i) {
        return triangles.get(i).nodes;
    }
}
```

ב. מהי פונקציית ההפשטה (abstraction function) של המחלקה שלך? התשובה צריכה להיות פורמאלית, לא תיאור מילולי.

AF: SimpleMesh \rightarrow (T1, T2, ..., Tn)

AF(this) = (triangles.get(1), triangles.get(2), ..., triangles.get(triangles.size()-1))

where

AF: Triangle \rightarrow (P1, P2, P3)

AF(this) = (nodes[0], nodes[1], nodes[2])

where

AF: Point \rightarrow <x,y>

AF(this) = < getX(), getY() >

ג. נתון ממשק שמגדיר איטרטור של משולשים. הגדר/הגדירי שירות שמחזיר איטרטור כזה עבור המחלקה שהגדרת בסעיפים הקודמים. בתשובה הזו מותר להגדיר קוד אך ורק בתוך השירות החדש.

```
public interface TriangleIterator ()
// returns the next triangle in a set of triangles, or
// null if there are no more triangles.
public Point[] next();
}
```

```
public class SimpleMesh ... {
...
public TriangleIterator triangleIterator {
return new TriangleIterator() {
private int i = 0;
public Point[] next() {
if (i==triangles.size()) return null;
return triangles.get(i++).nodes;
}
};
}
}
```

ד. נניח שהשירות triangleIterator מוגדר בממשק Mesh. ממש מחלקה ששירות מחלקה שלה מחשב את השטח שמכוסה על ידי Mesh. מותר להגדיר במחלקה הזו שירות עזר שיחשב את השטח של משולש נתון, אבל אין צורך לקודד את הנוסחה לחישוב שטח משולש (כלומר את/ה יכול/ה לממש את גוף שירות העזר כ-"return ...;").

```
public class MeshAreaCalculator {

private static triangleArea(triangle) { return ...; }

public static double area(Mesh m) {
double a = 0;
TriangleIterator i = m.triangleIterator();
for (Point[] triangle = i.next();
triangle != null;
triangle = i.next())
```

```
    a += triangleArea(triangle);  
    return a;  
  }  
}
```


ה. הוכח/הוכיחי שהשירות שהגדרת בסעיף הקודם נכון. ציין/ציני בבירור את ההנחות שעליהן ההוכחה מסתמכת.

הוכחה: המשתנה `triangle` מקבל את כל הערכים האפשריים שהאיטרטור מחזיר עד שהוא מחזיר `null`. כמו כן, איננו משנים את ה-`Mesh` במהלך השימוש באיטרטור, מה שמבטיח שהאיטרטור פועל נכון. (אנו משתמשים כאן בהנחה שהאיטרטור מחזיר את כל המשולשים ב-`Mesh` ואז `null` אם לא משנים את ה-`Mesh` מרגע שהאיטרטור נוצר ועד שמפסיקים להשתמש בו). מכיון שהפנים של המשולשים זר, השטח של כל ה-`Mesh` הוא סכום השטחים של המשולשים, והמשתנה `a` אכן מכיל בסוף השגרה את סכום השטחים הללו. אנו גם מניחים כמובן ששגרת העזר מחשבת שטח משולש בצורה נכונה.

ו. כעת אנו מעוניינים לממש את `Mesh` בצורה יעילה וחסכונית בזיכרון ככל האפשר. במחלקה הבאה הגדרנו את השדות. תאר/י פונקציית הפשטה (`abstraction function`) שממפה את השדות הללו למצב המופשט של `Mesh`. יש גם להשלים את ההכרזה של המחלקה. בסעיף הבא תתבקש/י להגדיר את השירותים של המחלקה הזו. לסעיף הזה יש יותר מפתרון אחד אפשרי.

```
public class EfficientMesh Implements Mesh {
    private double[] a ;
    private int[]    b ;
    ...
}
```

פונקציית ההפשטה היא:

AF: `EfficientMesh` \rightarrow (`T1`, `T2`, ..., `Tn`)

```
AF(this) = (
    [<a[b[0]], a[b[0]+1]>, // one point (X and Y coordinates)
     <a[b[1]], a[b[1]+1]>,
     <a[b[2]], a[b[2]+1]> // first triangle T1 ends here
    ],
    [<a[b[3]], a[b[3]+1]>,
     <a[b[4]], a[b[4]+1]>,
     <a[b[5]], a[b[5]+1]> // T2 ends here
    ],
    ...
    [<a[b[3n-3]], a[b[3n-3]+1]>,
     <a[b[3n-2]], a[b[3n-2]+1]>,
     <a[b[3n-1]], a[b[3n-1]+1]> // Tn ends here
    ]
)
```

`b.length == 3*n`

אפשר גם לשמור את הקודקודים של כל משולש בשייטת מספרים עוקבים ב-`a` ולא להשתמש ב-`b` בכלל. זה גורם לשכפול קודקודים ב-`a`.

ז. הגדר/הגדירי את השירותים `size` ו-`get` עבור המחלקה מהסעיף הקודם. הגדרת השירותים צריכה להתאים לפונקציית ההפשטה שנתת.

```
public class EfficientMesh implements Mesh {
    private double[] a ;
    private int[]    b ;

    public int size() { return b.length / 3; }
```

```
public Point[] get(int i) {  
    Point[] nodes = new Point[3];  
    nodes[0] = new Point(a[b[3*i]], a[b[3*i]+1]);  
    nodes[1] = new Point(a[b[3*i+1]], a[b[3*i+1]+1]);  
    nodes[2] = new Point(a[b[3*i+2]], a[b[3*i+2]+1]);  
    return nodes;  
}  
}
```

שאלה 4

נתונים המנשקים הבאים המתארים קורס וסטודנט במערכת מרשם קורסים:

```
public interface Course {
    public int units();
    מספר נקודות (שעות) – (שלם בין 1 ל 5)

    public int level();
    רמת הקורס (שלם בין 1 ל 3)

    public int numOfStudents();
    מספר הסטודנטים הרשומים כרגע לקורס

    public int maxNumStudents();
    המספר המירבי המותר של סטודנטים רשומים לקורס

    public boolean registered(Student s);
    החזר true אם ורק אם הסטודנט s רשום לקורס.

    public void register(Student s);
    רשום את הסטודנט s לקורס. הפעולה חוקית רק אם s לא רשום לקורס, והקורס לא מלא

    public void drop(Student s);
    בטל את הרישום של s לקורס. הפעולה חוקית רק אם s רשום לקורס
}

```

```
public interface Student {
    public void register(Course c);
    הרשם לקורס c . הפעולה חוקית אם הסטודנט לא רשום לקורס c , הקורס לא מלא,
    ומספר הנקודות הכולל של הסטודנט לאחר הרישום יהיה לכל היותר 10.

    public void drop(Course c);
    בטל את הרישום לקורס c . הפעולה חוקית אם הסטודנט רשום לקורס c .

    public int totalUnits();
    מספר הנקודות הכולל של הקורסים שהסטודנט רשום להם

    public String name();
    שם הסטודנט
}

```

להלן דוגמת שימוש במנשקים:

```
Course c1 = new ... ; // a course of 3 units, and max number of students 40
Course c2 = new ... ; // a course with 4 units, and max number of students 40
Student s1 = new ... ;
Student s2 = new ... ;
System.out.println(s1.totalUnits());
s1.register(c1);
s2.register(c1);
s1.register(c2);
System.out.println(s1.totalUnits());
s1.drop(c1);
System.out.println(s1.totalUnits());
```

הפלט של סדרת הפעולות יהיה:

```
0
7
4
```

המנשקים האלה וכל שאר המחלקות והמנשקים בשאלה הזו הם חלק מהחבילה `com.university.students`. החבילה אינה מכילה מנשקים וחבילות אחרים פרט לאלה המוזכרים בשאלה (בפרט, שאר הקוד של מערכת המחשב של האוניברסיטה נמצא בחבילות אחרות).

א. כתבו את החוזה של המנשק `Course`: לכל שרות כתבו תנאי קדם (precondition) ותנאי אחר (postcondition) באופן המקובל (ביטויים בוליאניים שיכולים להשתמש בשאילות). במידת הצורך, הוסיפו במילים תנאים שלא ניתנים לביטוי בצורה הרגילה.

```
@post 1 ≤ $ret ≤ 5
```

```
public int units()
```

```
@post 1 ≤ $ret ≤ 3
```

```
public int level()
```

```
@post 0 ≤ $ret ≤ maxNumStudents()
```

```
public int numOfStudents()
```

```

    @post 0 ≤ $ret

public int maxNumStudents()

    הערה: אם צוין תנאי קדם s != null אזי אסור לאף מימוש עתידי לבדוק תנאי זה

public boolean registered(Student s)

    @pre numStudents() < maxNumStudents()
    @pre !registered(s)
    @post registered(s)
    @post numOfStudents() == $prev(numOfStudents()) + 1

public void register(Student s)

    @pre registered(s)
    @post !registered(s)
    @post numOfStudents() == $prev(numOfStudents()) - 1

public void drop(Student s)

    ב. כתבו את החווה של המנשק Student (ההנחיות זהות לסעיף הקודם).

    @pre !c.registered(this)
    @pre c.numStudents() < c.maxNumStudents()
    @pre totalUnits() + c.units() ≤ 10
    @post c.registered(this)
    @post totalUnits() == $prev(totalUnits()) + c.units()

public void register(Course c)

    @pre c.registered(this)
    @post !c.registered(this)
    @post $prev(c.registered(this)) $implies
        totalUnits() == $prev(totalUnits()) - c.units()
    @post $prev(c.registered(this)) $implies
        c.numStudents() < c.maxNumStudents()

public void drop(Course c)

    @post 0 ≤ $ret
public int totalUnits()

public String name()

```

המחלקה SimpleCourse אמורה לממש את הממשק Course בצורה פשוטה, תוך שימוש במערך לייצוג הסטודנטים הרשומים לקורס. נתון חלק מהקוד – כל השדות, הבנאי, ומימוש של שרות אחד.

```
public class SimpleCourse implements Course {

    private int MaxNumStudents;
    private int top;
    private int units;
    private int level;
    private Student[] students;

    public SimpleCourse(int MaxNumStudents, int units, int level) {
        this.MaxNumStudents = MaxNumStudents;
        students = new Student [MaxNumStudents];
        this.units = units;
        this.level = level;
        top = -1;
    }
    public void register(Student s) {
        students[++top] = s;
    }

    // more code omitted
}
```

ג. הגדירו את משתמר הייצוג (representation invariant) של המחלקה SimpleCourse

```
1 ≤ level == level() ≤ 3
1 ≤ units == units() ≤ 5
top+1 == numOfStudents() ≤ maxNumOfStudents()
MaxNumStudents == maxNumOfStudents() == students.length
foreach 1=0..top : students[i] != null
```

השלימו את הקוד של המחלקה SimpleCourse.

```
public int units() { return units; }

public int level(){ return level; }

public int numOfStudents(){ return top+1 ; }

public int maxNumStudents(){ return maxNumStudents; }

public boolean registered(Student s) {
    for (Student ss : students)
        if (ss == s)
            return true;
    return false;
}
```

```
public void drop(Student s) {  
    for (int i=0; i<top+1; i++)  
        if (students[i] == s){  
            students[i] = students[top];  
            top--;  
            return;  
        }  
}
```

נתונה מחלקה SimpleStudent שמממשת את הממשק Student בצורה פשוטה:

```
public class SimpleStudent implements Student {

    private String name;
    private int totalUnits = 0;

    public SimpleStudent(String name) {
        this.name = name;
    }

    public void register(Course c) {
        c.register(this);
        totalUnits += c.units();
    }

    public void drop(Course c) {
        c.drop(this);
        totalUnits -= c.units();
    }

    public int totalUnits() {
        return totalUnits;
    }

    public String name() {
        return name;
    }
}
```

ד. מפתחי התוכנה רוצים להקל על כותבי קוד הלקוח, ומציעים להגדיר מחלקה OtherStudent שתממש גם היא את הממשק Student אבל מותר יהיה לקרוא לפעולה drop גם אם הסטודנט לא רשום לקורס (ובמקרה זה השרות drop לא יגרום לכל שינוי). האם זה תקין מבחינת כללי תיכון בעזרת חוזה? יש לנמק בקיצור.

תלוי בצורה שבה ניסחנו את תנאי האחר.

השרות במחלקה המוצעת מחליש את תנאי הקדם וזה תקין.

אם תנאי האחר נוסחו כך:

```
@post !c.registered(this)
@post totalUnits() == $prev(totalUnits()) - c.units()
@post c.numStudents() < c.maxNumStudents()
```

אזי השרות במחלקה המוצעת סותר את התנאים השני והשלישי וזה אינו תקין.

אולם, אם תנאי האחר נוסחו כך:

```
@post !c.registered(this)
```

```
@post $prev(c.registered(this)) $implies  
totalUnits() == $prev(totalUnits()) - c.units()
```

```
@post $prev(c.registered(this)) $implies  
c.numStudents() < c.maxNumStudents()
```

אזי הם ממשיכים להתקיים גם עבור המימוש החדש.

ה. מוצע להגדיר מחלקה UndergraduateStudent שתייצג סטודנטים לתואר ראשון, תירש מהמחלקה SimpleStudent, ותוסיף את הדרישה שפעולת register חוקית רק אם רמת הקורס (level) אליו הסטודנט נרשם קטנה מ 3. האם זה תקין מבחינת כללי תיכון בעזרת חוזה? יש לנמק בקיצור.

לא.

השרות במחלקה המוצעת מחזק את תנאי הקדם וזה אינו תקין.

ו. מהנדס התוכנה שגיא טוען שבמערכת שהוגדרה יש ליקוי חמור. לטענתו, אם קוד הלקוח מכיל קריאה כמו c.register(s) או c.drop(s) כאשר הטיפוס הדינמי של c הוא SimpleCourse ושל s הוא SimpleStudent בהתאמה, תתעורר בעיה. הסבירו מהי הבעיה, ואיך ניתן לפתור אותה. הפתרון יכול לכלול שינויים במנשקים ו/או במחלקות הממשות. אין צורך לכתוב קוד, אלא להסביר במילים את השינויים הנדרשים.

הבעיה:

פעולות ה register וה drop דורשות את עדכון מחלקת הסטודנט ועדכון מחלקת הקורס. שרותי ה register וה drop של המחלקה SimpleStudent כוללים גם את עדכון הקורס המתאים, אולם אם מתבצעת קריאה ישירות לשרותי ה register וה drop של המחלקה SimpleCourse לא מובטח כי עצם הסטודנט המתאים יתעדכן (הדבר פוגע, למשל, בעדכון היחידות של הסטודנט).

פתרונות אפשריים:

- א. הסרת השרותים register ו drop של אחת מהמחלקות ע"י הסרתם מהמנשק ומתן נראות חבילה לשרותים במחלקה המתאימה. כך מובטח שרק שרותים מאותה החבילה יגשו אליהם, ונוכל לאכוף את עקביות העדכון בקורס. בשאלה צוין במפורש כי רק שתי המחלקות האלה נמצאות בחבילה com.university.students כך שנראות החבילה אינה פומבית מדי.
- ב. ניתן להסיר את השרותים משני המנשקים ולהוסיף מחלקה חדשה באותה חבילה עם שרותי מחלקה (סטטיים) מתאימים.
- ג. אפשרות פחות טובה היא קינון המחלקות זו בזו (הפיכת אחת המחלקות למחלקה פנימית של האחרת). על אף שהדבר פותר את הבעיה, היחס בין המחלקות אינו יחס של מחלקה פנימית-חיצונית. בפרט לכל אחד מהעצמים של שתי המחלקות יש זכות קיום עצמאית ולא רק כמחלקת עזר של המחלקה האחרת.
- ד. פתרון בעייתי נוסף הוא להוסיף קריאה בגוף השרותים register ו drop של המחלקה SimpleCourse לשרותים המתאימים של המחלקה SimpleStudent. הפתרון עלול ליצור לולאות אינסופיות. הסרת הקריאות מהמחלקה SimpleStudent תחזיר אותנו לבעיה דואלית לבעיה המקורית. ניתן לפתור את בעיית הלולאות ע"י הוספת דגלים (משתני עזר) שיאוחלו בראשית פעולת register ו drop בין אם של SimpleCourse ובין אם של SimpleStudent.

ושוב, בהצלחה!