

בחינה בתוכנה 1 (0368 - 2157)
ובתוכנה 1א (0368 - 2004)

סיון טולדו, אוהד ברזילי, מיכל עוזרי-פלאטו, ליאור שפירא

סמסטר א' תשס"ז
מועד ג', 21 בדצמבר 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:

					ג	ב	א	1
				ד	ג	ב	א	2
		ו	ה	ד	ג	ב	א	3
								4
	סה"כ							

במבחן נדון במחלקות שמייצגות מספרים רציונליים, כגון $1/3$ או $99/100$. תזכורת מתמטית קצרה: כדי לצמצם שברים צריך לחלק את המונה והמכנה בגורם המשותף הגדול ביותר (GCD באנגלית). למשל, הגורם המשותף הגדול ביותר של 42 ו-56 הוא 14, ולכן

$$\frac{42}{56} = \frac{3 \cdot 14}{4 \cdot 14} = \frac{3}{4}.$$

כדי לחבר או לחסר שברים, צריך למצוא את המכנה המשותף (LCM באנגלית). למשל, המכנה המשותף של 21 ו-6 הוא 42, ולכן

$$\frac{2}{21} + \frac{1}{6} = \frac{4}{42} + \frac{7}{42} = \frac{11}{42},$$

שאלה 1, 31 נקודות

א. בין המתכנתים ביל ואומה התפתח ויכוח האם **טרם** לכתיבת המחלקה המייצגת מספרים רציונליים יש לכתוב **מנשק** המתאר את הפונקציונליות של המחלקה. ביל טען כי המחלקה פשוטה דיה וכי אין הצדקה לתאר את אותו הדבר גם ע"י מנשק וגם ע"י מחלקה. ציינו 3 טיעונים **נגדיים** המצדדים בשימוש במנשק. נמקו או הדגימו בקצרה את טיעוניכם:

_____ .1

_____ .2

_____ .3

עמוד 3 מתוך 11 מספר סידורי: _____ מספר ת"ז: _____

ב. לאחר שביל השתכנע בנחיצות המנשק, החל דיון בינו ובין אומה על נחיצות השרותים הבאים. עבור כל אחד מהם ציינו האם יש לו מקום במנשק, במחלקה, בשניהם או באף אחד מהם. נמקו בקצרה:

1. add - לחיבור שני מספרים רציונליים

2. subtract – לחיסור שני מספרים רציונליים

3. multiply - להכפלת שני מספרים רציונליים

4. divide – לחלוקת שני מספרים רציונליים

5. equals - להשוואת שני מספרים רציונליים

6. gcd – לחישוב הגורם המשותף הגדול ביותר של שני שלמים

7. lcm – לחישוב מכנה משותף של שני שלמים

8. normalize – להבאת שבר פשוט למצב מצומצם

שאלה 2, 31 נקודות

ביל ואומה מתלבטים לגבי הקבעון (mutability) של הטיפוס החדש. טיפוס מקובע (immutable) הוא טיפוס שלא ניתן לשנות את ערכי העצמים שלו לאחר שנוצרו. ראינו בתרגול את הטיפוסים String ו-StringBuffer המממשים גרסא מקובעת ושאינה מקובעת למחרוזות.

א. ציינו את היתרונות שיש לטיפוס רציונלי מקובע ואת היתרונות שיש לטיפוס רציונלי שאינו מקובע:

יתרונות הטיפוס הרציונלי המקובע (immutable):

יתרונות הטיפוס הרציונלי שאינו מקובע (mutable):

ב. באיזה מקרה יהיה ההבדל בין מימוש השרות equals במחלקות מקובעות לעומת מימוש במחלקות שאינן מקובעות? הסבירו במילים, אין צורך לכתוב קוד (רמז: המחלקה (String).

שאלה 3, 18 נקודות

המתכנת סטיב מימש את המחלקה SimpleRational לייצוג טיפוס רציונלי ובה נכלל קוד ההשוואה הבא:

```
public class SimpleRational implements ImmutableRational {

    private int num;
    private int denom;

    public SimpleRational(int num, int denom) {
        this.num = num;
        this.denom = denom;
    }

    public boolean equals(SimpleRational other) {
        return num==other.num && denom==other.denom;
    }

    // the rest of the class...
}
```

המחלקה אינה מכילה מימושים נוספים של השרות equals. כדי לבדוק את נכונות המימוש כתב סטיב את מחלקת הבדיקה הבאה:

```
public class TestRationalEquality {

    public static void main(String[] args) {
        SimpleRational r1 = new SimpleRational (1,2);
        SimpleRational r2 = new SimpleRational (1,2);
        ImmutableRational ir1 = new SimpleRational (1,2);
        ImmutableRational ir2 = new SimpleRational (1,2);

        // HERE
    }
}
```

עבור כל אחת משורות הקוד הבאות ציינו האם ניתן להגדירה בפונקציה ה-main של המחלקה TestRationalEquality במקום ההערה // HERE. סמנו את האפשרות המתאימה: האם תהיה שגיאת קומפילציה (אם כן, ציינו מה השגיאה), או שתהיה תעופה בזמן ריצה (אם כן, ציינו מאיזה סיבה), או שהקוד ירוץ בצורה תקינה (אם כן, ציינו מה יהיה פלט ההדפסה) או שהקוד ירוץ אך יחשוף באג במימוש השרות equals (אם כן, ציינו מה יהיה פלט ההדפסה, מה אמור היה להיות פלט ההדפסה ומה סיבת הבאג).

א. System.out.println(r1.equals(r2));

שגיאת קומפילציה (מהי?) / תעופה בזמן ריצה (מה הסיבה) / קוד תקין (מה מודפס) / קוד תקין החושף באג (מה מודפס? מה אמור להיות מודפס? איפה הבאג?)

System.out.println(ir1.equals(ir2)); .ב.

שגיאת קומפילציה (מהי?) / תעופה בזמן ריצה (מה הסיבה) / קוד תקין (מה מודפס) / קוד תקין החושף באג (מה מודפס? מה אמור להיות מודפס? איפה הבאג?)

System.out.println(r1.equals(ir1)); .ג.

שגיאת קומפילציה (מהי?) / תעופה בזמן ריצה (מה הסיבה) / קוד תקין (מה מודפס) / קוד תקין החושף באג (מה מודפס? מה אמור להיות מודפס? איפה הבאג?)

System.out.println(ir1.equals(r1)); .ד.

שגיאת קומפילציה (מהי?) / תעופה בזמן ריצה (מה הסיבה) / קוד תקין (מה מודפס) / קוד תקין החושף באג (מה מודפס? מה אמור להיות מודפס? איפה הבאג?)

r1 = ir1;
System.out.println(r1.equals(ir1)) .ה.

שגיאת קומפילציה (מהי?) / תעופה בזמן ריצה (מה הסיבה) / קוד תקין (מה מודפס) / קוד תקין החושף באג (מה מודפס? מה אמור להיות מודפס? איפה הבאג?)

ir1 = r1;
System.out.println(r1.equals(ir1)); .ו.

שגיאת קומפילציה (מהי?) / תעופה בזמן ריצה (מה הסיבה) / קוד תקין (מה מודפס) / קוד תקין החושף באג (מה מודפס? מה אמור להיות מודפס? איפה הבאג?)

שאלה 4, 20 נקודות

בשאלה זו נכתוב איטרטור מסוג `UniqueIterator<T>`, אשר מחזיר מכל רצף של איברים זהים רק איבר בודד.

למשל:

עבור הרצף 4 1 1 1 5 5 5 3 3 2 2 2 1 1 1
הוא יחזיר ערכים כאילו הרצף הוא 4 1 5 3 2 1

האיטרטור יוגדר כמחלקה **פנימית** גנרית בתוך המחלקה `MyList<T>` אשר מממשת את הממשק `MyList.Iterable<T>`. היא רשימה מקושרת של אברים מהטיפוס `Cell<T>` כפי שראינו בהרצאה.

השלימו את הקוד החסר במחלקות `MyList` ו- `UniqueIterator` שבעמוד הבא. יש להשאיר את מימוש השרות `remove` של האיטרטור ריק.

```
public class Cell <T> {
    private T      cont;
    private Cell <T> next;

    public Cell (T cont, Cell <T> next) {
        this.cont = cont;
        this.next = next;
    }

    public T cont() {
        return cont;
    }

    public Cell <T> next() {
        return next;
    }

    public void setNext(Cell <T> next) {
        this.next = next;
    }
}
```

```
public class MyList<T> implements Iterable<T> {  
    private Cell<T> head;  
    public MyList(T... elements) {  
        this.head = null;  
        for (int i = elements.length - 1; i >= 0; i--) {  
            head = new Cell<T>(elements[i], head);  
        }  
    }  
}
```

```
class UniqueIterator implements Iterator<T> {
```

```
    }  
}
```

ושוב, בהצלחה!