# מבחן מסכם מועד ב' ב"מבוא למדעי המחשב" 201-1-101-1

סמסטר א' תשס"א
22.2.2001

פרופ' אורי אברהם
פרופ' דניאל ברנד
ד"ר שמואל ספרוני
ד"ר מיכאל קודיש

משך הבחינה שעתיים וחצי.
חומר עזר אסור.
אין להשתמש במחשבון.

במבחן זה 7 שאלות המאפשרות לצבור עד 100 נקודות.

אנא רשמו את תשובותיכם בדף התשובות בלבד. בשאלות 7-5 רשמו את כל התשובות הנכונות. הקפידו לרשום בדף התשובות גם את מספר הנבחן ומספר החדר שבו אתם נבחנים וכן את מחלקתכם. המחברת שקיבלתם היא **מחברת הטיוטה והיא לא תימסר כלל לבדיקה**. בסיום הבחינה נאסוף אך ורק את דף התשובות.

בהצלחה

**<u>שאלה 1 (15 נקודות)</u>**

המחלקה DNode הנתונה כאן מגדירה חוליה ברשימה דו-כיוונית ובה שדה data מסוג Object ושדות previous ו-next המהווים מצביעים לחוליה הקודמת ולזו העוקבת. ההגדרה של המחלקה DNode נתונה כהרחבה של המחלקה Node המגדירה חוליה ברשימה רגילה:

```
public class DNode extends Node{
    public Node previous;

    DNode(Object data, Node previous, Node next){
        super(data, next);
        this.previous = previous;
    }
    public Node get_previous(){return previous;}
    public void set_previous(Node a){previous = a;}
} // DNode
```

```
public class Node{
   public Object data;
   public Node next;

   Node(Object data, Node next){
     this.data = data;
     this.next = next;
   }
   public Object get_data(){return data;}
   public Node get_next(){return next;}
   public void set_next(Node a){next = a;}
}// Node
```

המחלקה DList (נתונה בהמשך) מגדירה מבנה של רשימה דו-כיוונית של חוליות מסוג DNode ובה שתי חוליות מיוחדות: left ו- right, שנקראות זקיפים. חוליות אלו אינן מכילות data אך נותנות גישה נוחה לחוליה השמאלית והימנית במבנה ומפשטות את הגדרת הפעולות על המבנה.

**<u>המשימה</u>**

ההגדרה של המחלקה DList הנתונה בעמוד הבא פעלה כשורה עד שנשמטו מתוכה כל פעולות ההמרה (casting). יש בקוד <u>חמש</u> שורות שבהן <u>הכרחי</u> להוסיף פעולות המרה על מנת שיעבור קומפילציה ויעבוד כשורה. תקנו בדף התשובות את השורות הבעייתיות (וציינו את מספרי השורות), כך שהתכנית תעבוד כשורה.

<u>אין לרשום יותר מחמש שורות.</u>

```
1.     public class DList{
2.         public DNode left, right;
3.         DList(){
4.             left = new DNode(null,null,null);
5.             right = new DNode(null,left,null);
6.             left.set_next(right);
7.         }
8.         public boolean isEmpty(){
9.             return (left.get_next() == right);
10.         }
11.         public void addAtHead(Object data){
12.             DNode new_node =
13.                 new DNode(data, left, left.get_next());
14.             left.get_next().set_previous(new_node);
15.             left.set_next(new_node);
16.         }
17.         public void addAtTail(Object data){
18.             DNode new_node =
19.                 new DNode(data, right.get_previous(), right);
20.             right.get_previous().set_next( new_node);
21.             right.set_previous(new_node);
22.         }
23.         public void forwards_print(){
24.             DNode i = left.get_next();
25.             while( i != right ){
26.                 System.out.print(i.get_data() + " ");
27.                 i = i.get_next();
28.             }
29.             System.out.println();
30.         }
31.         public void backwards_print(){
32.             Node i =  right.get_previous();
33.             while( i != left ){
34.                 System.out.print(i.get_data() + " ");
35.                 i =  i.get_previous();
36.             }
37.             System.out.println();
38.         }
39.         public void delete(DNode i){//node i must be in the list
40.             i.get_next().set_previous(i.get_previous());
41.             i.get_previous().set_next(i.get_next());
42.         }
43.     }// class DList
```

## שאלה 2 (25 נקודות)

בשאלה זו נשתמש בקוד המצורף של המחלקות DList ו- DNode משאלה מס' 1 על-מנת לכתוב את המחלקה Set, המרחיבה את DList ומייצגת קבוצה של עצמים (שונים זה מזה, כלומר אסור לאותו איבר להופיע יותר מפעם אחת בייצוג הקבוצה).

הערה: גם אם לא עניתם על חלק מהסעיפים, תוכלו להשתמש בשיטות שהתבקשתם להגדיר בסעיפים האחרים. כמוכן, תוכלו להתיחס לשיטות המצורפות לשאלה 1, גם אם לא תיקנתם את בעיות ההמרה.

בסעיפים א'- ד' תמצאו הנחיות על-מנת להשלים את ההגדרות הבאות:

```
Public class Set extends DList {
    Public Set() {super();}

    public boolean isMember(Object b) {
    /***
                                    א. השלימו בדף התשובות
    ***/
    }
    public void add(Object b) {
    /***
                                    ב. השלימו בדף התשובות
    ***/
    }
    public Set sameClass(String type) {
    /***
                                    ג. השלימו בדף התשובות
    ***/
    }
    public Set setsOfClasses(){
    /***
                                    ד. השלימו בדף התשובות
    ***/
    }
} // class Set
```

## סעיף א (5 נק')

השלימו את השיטה isMember(Object b) של המחלקה Set אשר בודקת אם העצם b הינו איבר בקבוצת איברי עצם המפתח. הבדיקה נעשת ע"י השיטה הבוליאנית equals(Object c) שתהיה מוגדרת עבור מחלקות האיברים שבקבוצה.

```
    public boolean isMember(Object b) {
    /***
                                    השלימו בדף התשובות
    ***/
    }
```

## סעיף ב (5 נק')

השלימו את השיטה add(Object b) של המחלקה Set אשר מוסיפה איבר b לקבוצת האיברים בעצם המפתח.

```
    public void add(Object b) {
    /***
                                    השלימו בדף התשובות
    ***/
    }
```

## סעיף ג (8 נק')

הנח כי הוספנו את השיטה getType() למחלקה Node והיא מחזירה כמחרוזת את שם המחלקה של שדה ה-data בעצם המפתח. לדוגמה: אם נבצע את שתי הפקודות

```
Node s = new Node(new Set());
System.out.print(s.getType());
```

אזי יודפס "Set".

העזרו בשיטה getType() כדי להשלים את השיטה sameClass(String type) של המחלקה Set אשר משמיטה מעצם המפתח את כל האיברים מטיפוס type, ומחזירה אותם כקבוצה חדשה של איברים.

```
public Set sameClass(String type){
/***
                                        השלימו בדף התשובות
***/
}
```

לדוגמה, נניח שניתנו הפקודות:

```
Set s1 = new Set();
s1.add("123"));
s1.add(new Double(4.2));
s1.add(new Integer(3));
s1.add(new Double(1.2));
```

אם בשלב זה s1.left.next מצביע על חוליה ששדה ה- data שלה הוא מטיפוס Double ואם נבצע עתה את הפקודה הנוספת

```
Set s2 = s1.sameClass(s1.left.next.getType());
```

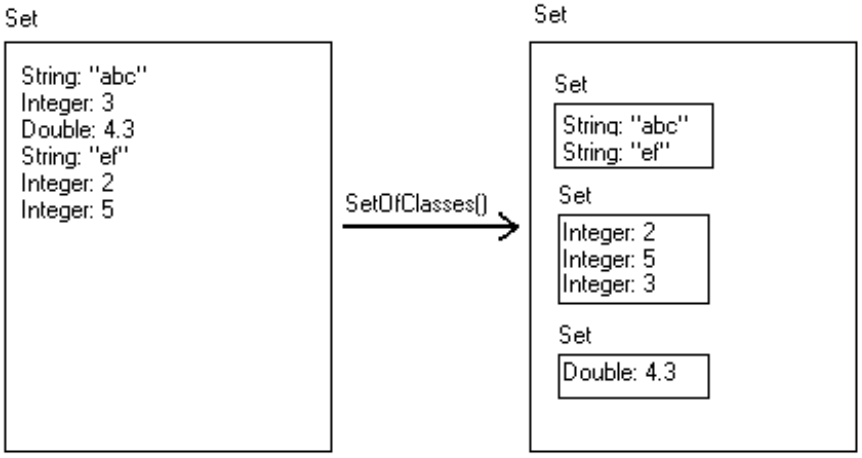
אזי הקבוצה s1 תכיל את המחרוזת "123" ואת השלם 3, ואילו s2 תכיל את הממשיים 1.2 ו- 4.2 .

## סעיף ד (7 נק')

השלימו את השיטה setsOfClasses() של המחלקה Set אשר מחלקת את קבוצת האיברים בעצם המפתח לקבוצות של איברים על-פי טיפוסיהם. הערך המוחזר הינו קבוצה של קבוצות, שבכל אחת מהן האיברים הם מאותו טיפוס.

```
public Set setsOfClasses(){
/***
                                        השלימו בדף התשובות
***/
}
```

לדוגמה, אם נפעיל את השיטה על הקבוצה שמצוירת בצד שמאל, אזי הערך המוחזר יהיה קבוצת הקבוצות שמצוירת מצד ימין:



**שאלה 3 (20 נקודות)**

השלימו את השיטה `subsetsOfSize(int n,int k)`, אשר מקבלת שני מספרים שלמים k ו- n (כך ש- $0 \le k \le n$ ו- $1 \le n$) ומדפיסה את כל תתי-הקבוצות בגודל k של הקבוצה **{1,2,...,n}** (הדפסת תת-קבוצה פירושה הדפסת כל איבריה כמחרוזת). למשל, השיטה main הנתונה קוראת ל- `subsetsOfSize(n,k)` כאשר n = 5 ו- k=3, ואז התכנית תדפיס את המחרוזות הבאות (אחת לשורה בסדר כלשהו):

123 124 125 134 135 145 234 235 245 345

```
public class subsets{
   static void subsetsOfSize(int n, int k){
        subsetsOfSize(n,k,"");
   }

   static void subsetsOfSize(int n, int k, String s){
   /***
                                          השלימו בדף התשובות
   ***/

   public static void main(String[] args){
      subsetsOfSize(5,3);
   }
 }
```

**שאלה 4 (10 נקודות)**

**סעיף א** ציין בדף התשובות האם התוכנית המוגדרת במחלקה what:

א. אינה עוברת קומפילציה.
ב. עוברת קומפילציה אך נתקלת בשגיאת הרצה.
ג. רצה והפלט (משמאל לימין) הוא.

**סעיף ב** חזור על סעיף א' בהנחה ששתי השורות המסומנות (*** 1 *** ו- *** 2 ***) בשיטה what הוחלפו זו בזו.

שימו לב שהמחלקה Node נתונה בשאלה מס' 1 ואילו המחלקה List מובאת בהמשך כתזכורת.

```
class what{
    public static void what(List x){
        if (x.first != null && x.first.get_next() != null){
            Node n = x.first;
            x.first = x.first.get_next();  // *** 1 *** //
            n.set_next(null);              // *** 2 *** //
            what(x);
            Node node = x.first;
            while (node.get_next()!=null)
                node = node.get_next();
            node.set_next(n);
        }
    }
    public static void main(String[] args){
        List x = new List();
        x.addAtHead(new Integer(5)); x.addAtHead(new Integer(4));
        x.addAtHead(new Integer(3)); x.addAtHead(new Integer(2));
        x.addAtHead(new Integer(1));
        what(x); x.print();
    }
}
```

זה המימוש של רשימה משורשרת (להזכירכם):

```
public class List{
      public Node first;
            List(){first = null;}
      public void addAtHead(Object data){
            first = new Node(data, first);
      }
      public void print(){
            Node link = first;
            while( link != null ){
                  System.out.print(link.get_data() + " ");
                  link = link.get_next();
            }
                System.out.println();
      }
}// class List
```

**שאלה 5 (10 נקודות)**

בהתייחס למחלקות MyInt ,Exam:

```
class MyInt {
  private int _n;
  MyInt (int i) {_n = i;}
  int intValue() {return _n;}
  static void inc(MyInt n) {
    if (n!=null) n = new MyInt (n.intValue() + 1);
  }
}

public class Exam {
  public static void main(String[] a) {
    MyInt i2 = new MyInt(2);
    i2.inc(i2);
    System.out.println(i2.intValue());
  }
}
```

סמנו את כל התשובות הנכונות.

א. התוכנית תצלח הידור (תעבור קומפילציה), תרוץ ותדפיס 2.
ב. התוכנית תצלח הידור, תרוץ ותדפיס 3.
ג. התוכנית תצלח הידור אך תגרום לשגיאה בזמן ריצה.
ד. התוכנית לא תצלח הידור, שכן הפונקציה הסטטית `inc` מפעילה שיטות של `MyInt`.
ה. התוכנית לא תצלח הידור, שכן בשיטה `main`, הפונקציה הסטטית `inc` אינה מופעלת על שם המחלקה `MyInt`, אלא על אובייקט מטיפוס `MyInt`.
ו. אם נשנה את כותרת השיטה `inc` להיות: `private void inc(MyInt n)` אז התוכנית תצלח הידור.

**<u>שאלה 6 (10 נקודות)</u>**

למחלקה Point, מעבודת בית מס' 3, הוספנו שיטה Point midpoint(Point Q) אשר מחזירה את הנקודה (Point) שהיא אמצע הקטע המחבר את עצם המפתח עם הנקודה הנתונה Q.

המחלקה Triangle מייצגת משולשים. היא מכילה בונה Triangle(Point P1, Point P2, Point P3) אשר מקבל את קודקודי המשולש, וכן מכילה שיטה double area() אשר מחזירה את שטחו של המשולש ושיטה Point[ ] getVertices() אשר מחזירה מערך ובו שלושת קודקודי המשולש. השיטה הבאה נמצאת אף היא במחלקה Triangle:

```
double strangeFunction(){

  double result  = 0;
  if (area() >= 0.001){
      Point Q = getVertices()[0].midpoint(getVertices()[1]);
      Point R = getVertices()[1].midpoint(getVertices()[2]);
      Point S = getVertices()[2].midpoint(getVertices()[0]);
      Triangle T = new Triangle (Q, R, S);
      result = area() + 2 * T.strangeFunction();
  }
  return result;
}
```

נניח עתה כי בתוכנית מסויימת, העצם T מציין משולש במחלקה Triangle אשר קדקודיו נמצאים בנקודות (-1,0), (0,1), (1,0) במישור. מהו ערך המשתנה x לאחר ביצוע ההוראה הבאה:

```
double x = T.strangeFunction();
```

א. 0
ב. 31/16
ג. 1023/512
ד. 5/4
ה. 3/2
ו. אף תשובה לא נכונה.

**<u>שאלה 7 (10 נקודות)</u>**

במחלקה MyIntegers נמצא משתנה עצם n מסוג int, ובונה

```
public MyIntegers(int m){n = m;}
```

כמו כן נמצאת במחלקה זו שיטה int randomPrimeDivisor() אשר מחזירה מחלק ראשוני אקראי של המשתנה n כאשר $n \geq 2$ ומחזירה 1 עבור $n = 1$ (לא נפעילה עם $n \leq 0$). לדוגמה, אם ערך השדה n של העצם x במחלקה MyIntegers הוא 17, אזי x.randomPrimeDivisor() יחזיר 17, ואילו אם $n = 60$ אזי יוחזר ערך מבין 2,3,5 (כל אחד בהסתברות שווה).

עוד נוסיף למחלקה זו שתי שיטות int random1() וכן int random2(). גוף השיטה int random1() הוא:

```
MyIntegers y = new MyIntegers (n / randomPrimeDivisor());
return randomPrimeDivisor() * y.random1();
```

ואילו גוף השיטה int random2() הוא:

```
int l = randomPrimeDivisor();
MyIntegers y = new MyIntegers (n / l);
return l * y.random2();
```

עתה בחן את קטע הקוד הבא, בו m הוא משתנה מטפוס int.

```
MyIntegers x = new MyIntegers (m);
int a = x.random1(), b = x.random2();
```

לאחר ביצוע קוד זה:

א. המשתנים a ו- b שווים בהכרח.
ב. יש (בתיאוריה, אם מתעלמים מכך ש int מוגבל לתחום סופי של מספרים) אינסוף ערכי m עבורם $a = b$ בהכרח, וכן יש אינסוף ערכי m עבורם יתכן ש- $a \neq b$.
ג. עבור $m = 30$ יש 8 תוצאות אפשריות עבור a, ותוצאה אחת בלבד עבור b.
ד. עבור $m = 42$ יש 10 תוצאות אפשריות עבור a, ותוצאה אחת בלבד עבור b.
ה. עבור $m = 109$ יש 5 תוצאות אפשריות עבור a.
ו. אף תשובה לא נכונה.