

בחינה בתוכנה 1

סמסטר א' וב', מועד ב', תש"ע
18/08/2010

דן הלפרין, אוהד ברזילי, אסף זריצקי, מתי שמרת
ליאור וולף, ליאור שפירא, רובי בוים

הוראות (נא לקרוא!)

- משך הבחינה **שלוש שעות**, חלקו את זמנכם ביעילות.
- אסור השימוש בחומר עזר כלשהו, כולל מחשבוניו או כל מכשיר אחר פרט לעט. בסוף הבחינה צורף לנוחותכם נספח ובו תיעוד מחלקות שימושיות.
- 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 תיפסל. תשובות במחברת הבחינה לא תיבדקנה. במידת הצורך ניתן לכתוב בגב טופס הבחינה.
- יש למלא מספר סידורי (מס' מחברת) ומספר ת.ז על כל דף של טופס הבחינה.
- בכל השאלות ניתן להניח שכל החבילות הדרושות יובאו, ואין צורך לכתוב שורות import.
- במקומות בהם תתבקשו לכתוב מתודה (שירות), ניתן לכתוב גם מתודות עזר.
- ניתן להוסיף הנחות לגבי אופן השימוש בשירותים המופיעים בבחינה, ובלבד שאין הן סותרות את תנאי השאלה. יש לתעד הנחות אלו כחווה (תנאי קדם, תנאי בתר) בתחביר המקובל, שיכתב בתחילת השרות.

לשימוש הבודקים בלבד:

שאלה	א	ב	ג	ד	סה"כ
1					
2					
3					
4					

בהצלחה!

כל הזכויות שמורות ©
מבלי לפגוע באמור לעיל, אין להעתיק, לצלם, להקליט, לשדר, לאחסן במאגר מידע, בכל דרך שהיא, בין מכונית ובין אלקטרונית או בכל דרך אחרת כל חלק שהוא מטופס הבחינה.

שאלה 1 (15 נקודות)

ממשו את השרות `maxRun` אשר בהינתן מחרוזת (`string`) מחזיר את הריצה (`run`) הארוכה ביותר במחרוזת. ריצה מוגדרת בתור מספר הפעמים שבו מופיע אותו התו ברצף (השרות מחזיר את אורך הרצף הארוך ביותר במחרוזת).

להלן כמה דוגמאות:

```
maxRun("hoopla") → 2  
maxRun("$") → 1  
maxRun("abbCCCddBBBxx") → 3  
maxRun("") → 0
```

ניתן להגדיר מבני עזר או שרותים חדשים לצורך המימוש:

```
public static int maxRun(String str) {
```

שאלה 2 (40 נקודות)

- א. רב-מפה (multimap) הוא ממשק דומה למפה, אשר בשונה ממפה מאפשר לשמור כמה ערכים לכל מפתח. השימוש ברב מפה שימושי עבור מיפויים שאינם חד ערכיים. לדוגמא:
- "עובד" עם כמה "כלי רכב"
 - "לקוח" עם כמה "חשבונות בנק".

```
import java.util.Set;

public interface MultiMap<K,V> {

    public void put(K key, V value);

    public Set<V> get(K key);

}
```

ממשו מחלקה המממשת את הממשק רב-מפה לעיל (ניתן להיעזר במחלקות המתועדות בנספח):

ב. מפה מתויגת (TaggedMultiMap) הוא מנשק אשר מרחיב את מנשק המפה (MultiMap), המאפשר להכניס לכל זוג עוד עצם מטיפוס כלשהו (tag), הקשור במובן כלשהו לזוג הממופה. מפה מתויגת היא שימושית כאשר רוצים להוסיף מידע שאינו שייך ספציפית לא למפתח (key) ולא לערך (value), וכן כדי להבחין בין מספר ערכים הממופים לאותו מפתח.

לדוגמא: במנוע חיפוש ניתן ליצור מיפוי בין "שאלתה" ובין "עמודי אינטרנט", עם תגיות מטיפוס Double, אשר יהיה מספר בין 0 ל-1 שמייצג את מידת הרלוונטיות של העמוד לשאלתה.

להלן מנשק הטיפוס TaggedMultiMap:

```
import java.util.Set;

public interface TaggedMultiMap<K,V,T> extends MultiMap<K,V> {

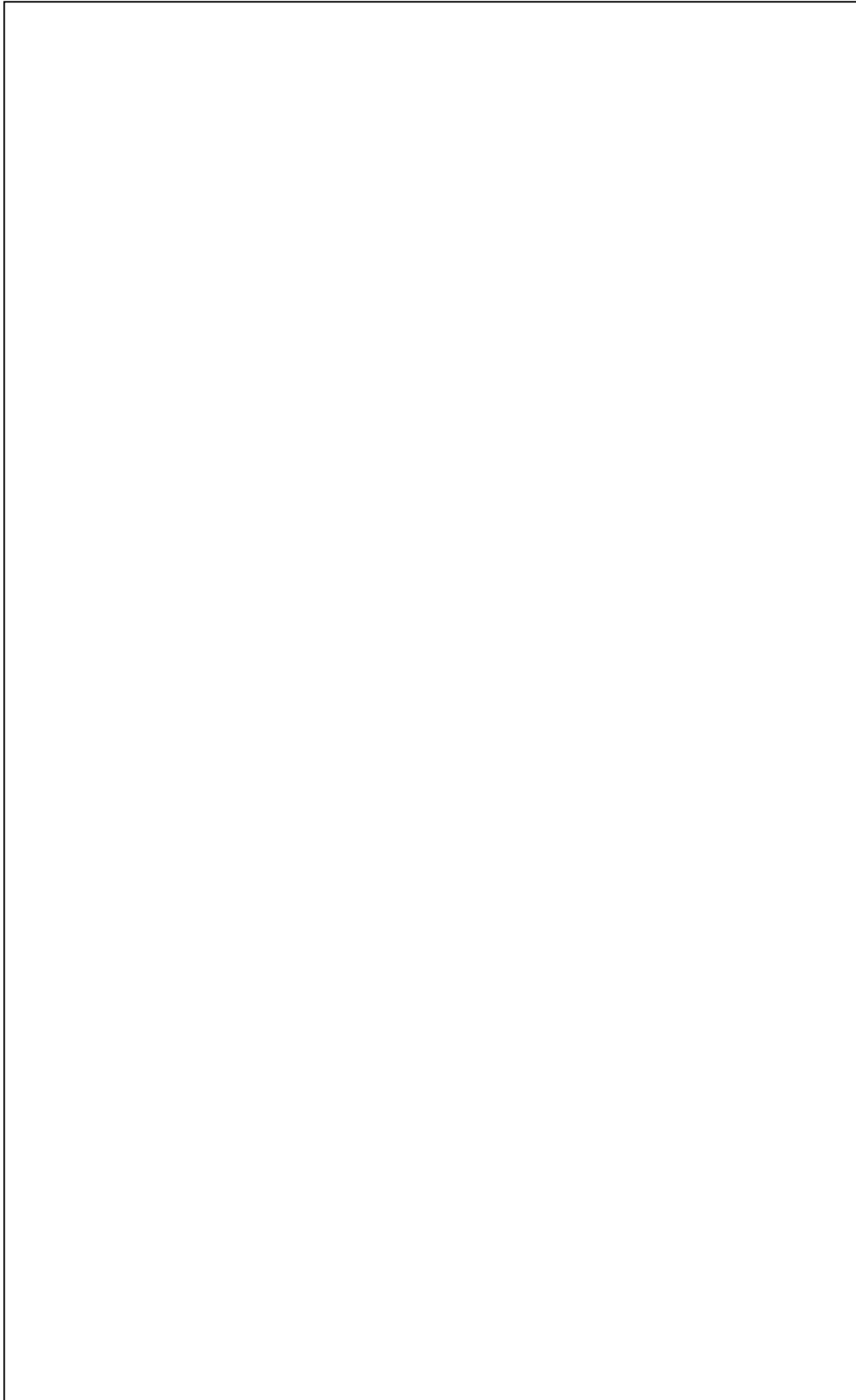
    public void put(K key, V value, T tag);

    public T getTag(K key, V val);
    public T getTag(Entry<K, V> entry);
    public Set<T> getTags(K key);

    public interface Entry<K,V> {
        K getKey();
        V getValue();
    }
}
```

שימו לב כי ניתן להשתמש בשרות put של הטיפוס TaggedMultiMap גם עבור key ו-value שהוכנסו כבר ל map ללא tag (למשל ע"י שרות ה-put של MultiMap). במקרה כזה, השרות מוסיף למיפוי את התגית (או, אם כבר הייתה תגית למיפוי, מחליף אותה). עבור ערכים (values) ללא תגית מחזיר השרות getTag את הערך null. אם עבור מפתח (key) מסוים לא הוכנסו תגיות כלל, מחזיר השרות getTags את הקבוצה הריקה.

ממשו מחלקה המממשת את המנשק TaggedMultiMap:



ג. בחברת מכשירי הטלפון הסלולרי Mokia, מתכננים את ספר הטלפונים של מכשירי הדור הבא. בגרסה הראשונה של המכשיר ניתן יהיה להזין אנשי קשר ולהם כמה מספרי טלפון. בעלת המכשיר הנייד י/תוכל לציין עבור כל מספר טלפון בעת הזנתו האם הוא טלפון בבית, בעבודה, נייד או אחר. אם מספר טלפון הזן ללא ציון סוגו, ניתן אחר כך להוסיף לו סוג ע"י הזנתו מחדש תוך ציון סוגו.

להלן הממשק Phonebook המתאר את הפונקציונליות לעיל:

```
import java.util.Set;

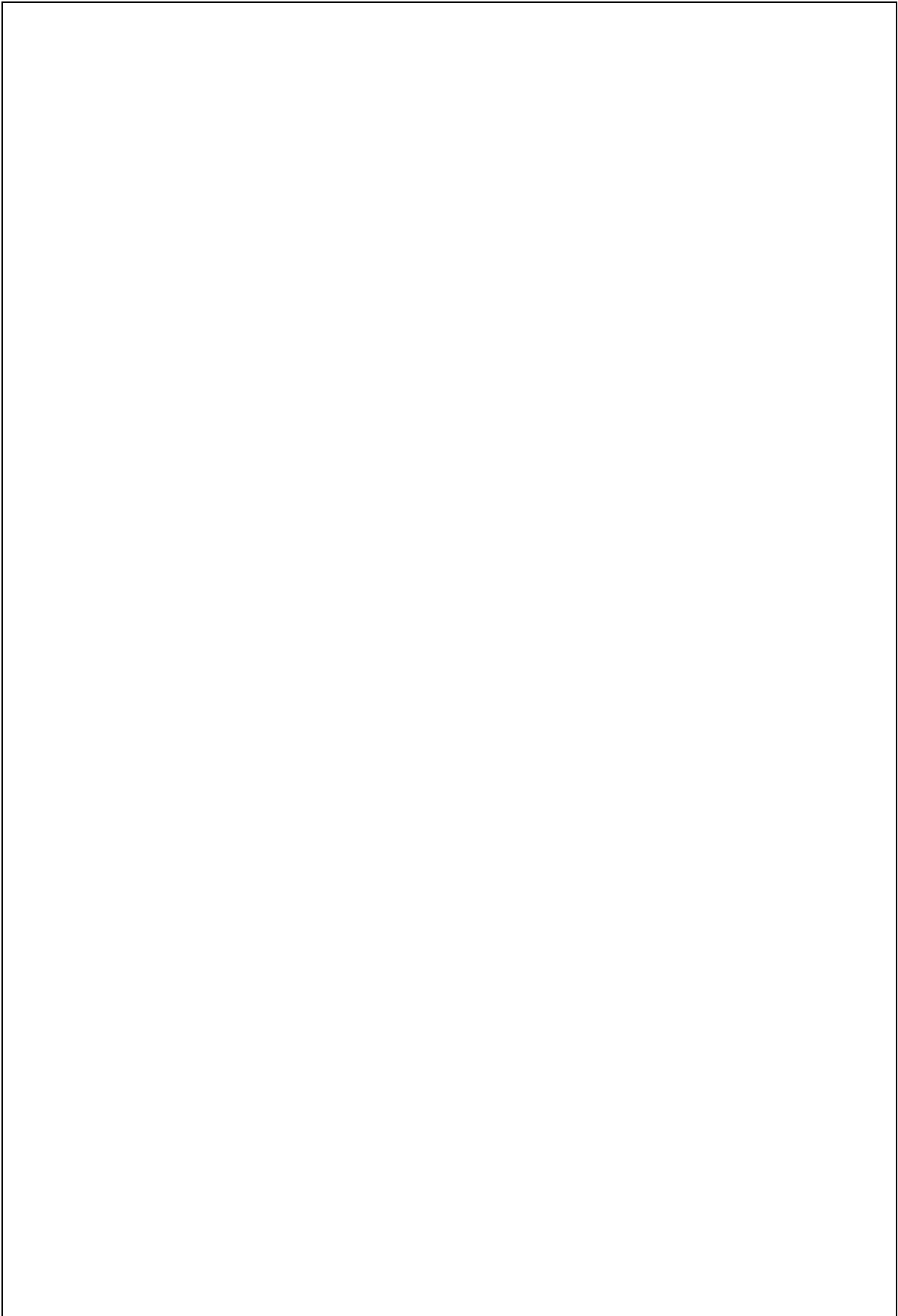
public interface Phonebook {

    Set<String> getContactPhoneNubers(String name);
    String getContactSpecificPhoneNuber(String name, NumberType type);

    void insert(String name, String number);
    void insert(String name, String number, NumberType type);

    public enum NumberType {
        Home, Work, Mobile, Other;
    }
}
```

ממשו מחלקה המממשת את הממשק Phonebook. אם ברצונכם להוסיף הנחות נוספות על השימוש במחלקה ציינו אותן כהערות מעל מימושי השרותים בתחביר פורמאלי ככל הניתן.



שאלה 3 (20 נקודות)

א. נתונה המחלקה TwoDPoint הבאה:

```
1. public class TwoDPoint {
2.     int x,y;

3.     public TwoDPoint(int x, int y) {
4.         this.x = x;
5.         this.y = y;
6.     }
7. }
```

אומה הסתומה, שמצאה את המחלקה באינטרנט החליטה להשתעשע איתה:

```
1. TwoDPoint p1 = new TwoDPoint(1,2);
2. TwoDPoint p2 = new TwoDPoint(1,2);
3. System.out.println(p1.equals(p2));
```

מה רבה הייתה הפתעתה כאשר גילתה שהפלט אינו כפי שציפתה, אלא: **false**

האם אתם, ילדים, תוכלו לעזור לאומה להבין את פשר הדבר? אילו שינויים יש לעשות בקוד כדי שיודפס true? ממשו בקוד את כל השינויים הדרושים:

ב. ביל הדביל מצא את המחלקה הבאה באינטרנט:

```

1. public class ThreeDPoint extends TwoDPoint {
2.     int z;

3.     public ThreeDPoint(int x, int y, int z) {
4.         super(x, y);
5.         this.z = z;
6.     }
7. }

```

והחליט להשתמש איתה קצת. אומה, שבינתיים הספיקה לתקן את TwoDPoint, נתנה לו את המימוש שלה (שגורם לקוד בסעיף א' להדפיס true) והם שניהם הריצו את הקוד הבא:

```

1. TwoDPoint p1 = new ThreeDPoint(1,2,3);
2. TwoDPoint p2 = new ThreeDPoint(1,2,3);
3. TwoDPoint p3 = new ThreeDPoint(1,2,4);

4. Set<TwoDPoint> set = new HashSet<TwoDPoint>();
5. set.add(p1);
6. System.out.println(set.contains(p1));
7. System.out.println(set.contains(p2));
8. System.out.println(set.contains(p3));

```

מה רבה הייתה הפתעתם כאשר גילו שהפלט אינו כפי שציפו...
מה מדפיסה התוכנית? הסבירו את הסיבה לכל אחת מההדפסות:

ג. אילו שינויים יש לעשות בקוד כדי שפלט התוכנית יהיה:

```

true
true
false

```

שאלה 4 (25 נקודות)

א. בנבכי קוד המקור של הרשת החברתית Fakebook מצאה המפתחת אלכסנדרה את השרות הבא:

```
private static void postCommentsToFriendsWall(Entry entry, Contact [] friends){
    for(int i = 0; i < MAX_NUMBER_OF_FRIENDS; i++){
        friends[i].postCommentToWall(entry.getComments());
    }
}
```

`MAX_NUMBER_OF_FRIENDS` הוא משתנה ציבורי (public) שהוגדר קודם לכן.

למרות שהשרות מתקפל ורץ עבור קלטים רבים, אלכסנדרה מבינה מקריאת הקוד שכותב השרות מימש אותו עבור הקשר מסוים ותחת הנחות מסוימות, והיא מחליטה לנסח אותו בעזרת חוזה.

עזרו לאלכסנדרה לנסח את החוזה של השרות, השתמשו בתחביר פורמאלי חוקי במידת האפשר:

ב. מארק, מפתח ותיק בחברה, ממליץ לאלכסנדרה לשכתב את הקוד כך שלא יהיו לשרות תנאי קדם כלל (precondition: true) אולם יהיו לו תנאי צד. מארק ממליץ לעשות שימוש בחריגים שאינם נבדקים (unchecked exceptions) שימסכו מקרי קצה (או חריגים אחרים) אשר עשויים להפתיע את לקוחות הפונקציה וייתנו חיווי טוב יותר על אופי התקלות שקרו. עזרו לאלכסנדרה לשכתב את מימוש הפונקציה כפי שהמליץ לה מארק. במקרה הצורך, ניתן להגדיר מבני עזר חדשים לצורך המימוש:

ג. אלכסנדרה לא מרוצה מהמימוש הקודם ומחליטה לשכתב את מימוש השרות כך שלא יהיו לו תנאי קדם או תנאי צד כלל (אינה זורקת חריגים, precondition: true) – ובמקום זאת יהיה לו **ערך מוחזר משמעותי**. עיזרו לאלכסנדרה לשכתב את הפונקציה והשלימו את החוזה שלה (יש יותר מפתרון אחד אפשרי). השתמשו בתחביר פורמאלי חוקי במידת האפשר. במקרה הצורך, ניתן להגדיר מבני עזר חדשים לצורך המימוש:

```
/**  
 * @pre: true  
 *  
 * @post:  
 *  
 *  
 *  
 *  
 */
```

ד. שכתוב חתימה של שרות הוא צעד נדיר עבור קוד הנמצא כבר בשימוש. מדוע במקרה זה מותר היה לאלכסנדרה לשכתב גם את חתימת השרות? ציינו 2 סיבות:

ושוב, בהצלחה!