

פתרון הבחינה בתוכנה 1

סמטר ב', מועד א', תשע"ב
09/07/2012

יעל אמסטרדמר, אוהד ברזילי, ליאור וולף, אלכסיי זגלסקי

הוראות (נא לקרוא!)

- משך הבחינה **שלוש שעות**, חלקו את זמנכם ביעילות.
- אסור השימוש בחומר עזר כלשהו, כולל מחשבוני 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.
- בעמוד הבא (עמוד 2) מופיע שאלון קצר לגבי הרקע שלכם בתכנות. השאלון ניתן כחלק מהמעקב אחר השפעת קורס ההכנה לקראת התואר הראשון (קורס הקיץ שלפני שנה א'). מילוי השאלון לא ישפיע על ציונכם בשום צורה.
- ניתן להניח לאורך הבחינה שכל החבילות הדרושות יובאו, ואין צורך לכתוב שורות import.
- במקומות בהם תתבקשו לכתוב מתודה (שירות), ניתן לכתוב גם מתודות עזר, אלא אם צוין במפורש אחרת.
- ניתן להוסיף הנחות לגבי אופן השימוש בשורות המופיעים בבחינה, ובלבד שאין הן סותרות את תנאי השאלה. יש לתעד הנחות אלו כחווה (תנאי קדם, תנאי בתר) בתחביר המקובל, שייכתב בתחילת השרות.

לשימוש הבודקים בלבד:

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

בהצלחה!

שאלה 1 (15 נקודות)

מחרוזת תקרא **מחרוזת-חוקית-סוגריים** אם מספר הסוגריים השמאליים בה תואם למספר הסוגריים הימניים באופן שלכל סוגר שמאלי '(' קיים מימינו סוגר ימיני ')' הסוגר אותו (ורק אותו). לצורך השאלה נדון רק בסוג אחד של סוגריים (כלומר לא נחשיב סוגרים מרובעים או מסולסלים כסוגריים).

נרצה לממש את המתודה `isLegalParentheses` המחזירה `true` אם מחרוזת הקלט שלה, `text`, היא מחרוזת-חוקית-סוגריים כפי שתואר לעיל.

להלן כמה דוגמאות ריצה של הפונקציה:

```
isLegalParentheses(" (hello world) ") → true
isLegalParentheses(" )warning(" ) → false
isLegalParentheses(" ((too much open)") → false
isLegalParentheses(" also legal ") → true
isLegalParentheses("") → true
isLegalParentheses(" ((1 (2,(3),4))) ") → true
isLegalParentheses(" dangling((left) ") → false
```

ניתן להגדיר מבני עזר או שרתים חדשים לצורך המימוש. בסעיף זה אין התייחסות לסיבוכיות זמן הריצה של האלגוריתם. אם יש לכם הנחות לגבי הקלט של הפונקציה ציינו אותן במפורש בתחביר פורמלי ככל האפשר ע"י שימוש בטענות עיצוב בעזרת חוזה, בראש הפונקציה:

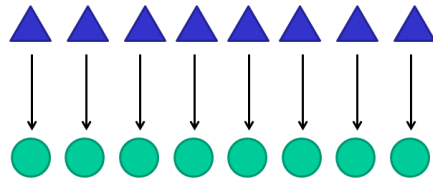
```

/ **
 * @pre: text != null
 */
public static boolean isLegalParentheses (String text) {
    int leftParentheses = 0;
    char[] array = text.toCharArray();
    for (char character:array){
        if (character=='('){
            leftParentheses++;
        }
        if (character==')'){
            if (leftParentheses==0){
                return false;
            }
            leftParentheses--;
        }
    }
    if (leftParentheses==0){
        return true;
    }
    return false;
}

```

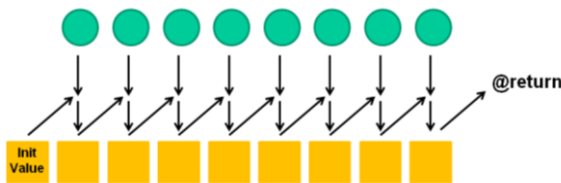
שאלה 2 (40 נקודות)

בשאלה זו נדון בעצוב ובמימוש Map-Reduce - חבילת תוכנה המאפשרת לבטא בקלות משימות תוכנה שונות כהרכבה של פעולות Map ו-Reduce (כפי שיוסבר בהמשך). שימו לב כי על אף השם אין קשר (לפחות לא ישיר) ל-`java.util.Map`.



הפעולה `map` (בעברית: מיפוי, הפעלת פונקציה) פועלת על רשימה של אברים (`List<T1>`) ומייצרת רשימה אחרת של אברים (`List<T2>`) שהיא תוצאה של הפעלת פעולה כלשהי על כל אחד מאברי הרשימה המקורית. הפעולה `map` אינה משנה את רשימה שעליה היא פועלת.

הפעולה `reduce` (בעברית: צמצום, "סיכום") פועלת על רשימה של אברים (`List<T2>`) ומחזירה



ערך מטיפוס `T3` שהוא תוצאה של הפעלת פעולה כלשהי על כל אחד מאברי הרשימה המקורית. שימו לב כי בשונה מפעולת `Map`, פעולת `Reduce` מצמצמת את הרשימה לכדי איבר אחד מטיפוס `T3`. גם הפעולה `reduce` אינה משנה את הרשימה שעליה היא פועלת.

לדוגמא: ניתן לתאר את **סכום הריבועים של רשימת מספרים כלשהי** כפעולת Map-Reduce. שלב ה-Map יעלה כל איבר בריבוע. שלב ה-Reduce יחזיר את סכום אברי הרשימה החדשה. בדוגמא הזו הטיפוסים `T1`, `T2` ו-`T3` הם כולם `Integer`, אבל אפשר לחשוב גם על פעולות Map ו-Reduce שבהן `T1`, `T2` ו-`T3` שונים זה מזה, למשל: פעולת Map אשר מעגלת מספרים (ל-`Double` Integer) או פעולת Reduce המחשבת ממוצע של רשימת שלמים (ל-`Integer` Double).

ביל הדביל מציע לעצב מערכת Map-Reduce בעזרת מנשקים והורשה באופן הבא: הוא מגדיר את המנשק הגנרי `MapReduceTask`:

```
public interface MapReduceTask<T1,T2,T3> {
    T2 mapOperation(T1 element);
    T3 reduceOperation(T3 partial, T2 element);
    T3 initialValue();
    T3 apply();
}
```

השרות `mapOperation` יופעל על כל אחד מאברי רשימת הקלט בשלב ה-`map`.

שלב ה-`reduce` מתבצע על ידי הפעלת השרות `reduceOperation` על כל אחד מאברי הרשימה שנוצרה בשלב ה-`map`, באופן הבא: בתחילה `partial` הוא ערך ברירת המחדל המוחזר על ידי השרות `initialValue`, ו-`element` הוא האיבר הראשון ברשימה. בהפעלה השנייה ואילך `partial` הוא הערך שהוחזר בהפעלה הקודמת של `reduceOperation` ו-`element` הוא האיבר הבא ברשימה. תוצאת שלב ה-`reduce` היא בעצם תוצאת הפעלה האחרונה של `reduceOperation`.

הפעלת השרות `apply` על מחלקה המממשת את המנשק, תבצע ברצף את שלבי ה-`map` וה-`reduce` שתוארו לעיל ותחזיר את תוצאתם.

א. (12 נקודות) השלימו את מימוש המחלקה המופשטת `AbstMapReduceTask`, אשר תשמש מחלקת בסיס למשימות Map-Reduce עתידיות. במקרה הצורך ניתן להוסיף שרותים, בנאים, שדות וטיפוסי עזר. בפרט, על המחלקה לספק בנאי אשר יקבל כארגומנט את הרשימה `(List<T1>)` שעליה תבוצע פעולת ה Map-Reduce (בקריאה ל `apply`).

שימו לב כי אין לממש את השרותים `mapOperation`, `reduceOperation` ו- `initValue`. שרותים אלו ימומשו במחלקות יורשות.

טיפ: מומלץ לחזור לסעיף זה שוב לאחר פתרון הסעיף הבא, בו אתם מתבקשים לממש מחלקה אשר יורשת מ `AbstMapReduceTask`, וזאת כדי לוודא ששני הסעיפים עקביים זה עם זה.

```
public abstract class AbstMapReduceTask<T1, T2, T3>
    implements MapReduceTask<T1, T2, T3> {

    private List<T1> originalList;

    public AbstMapReduceTask(List<T1> l){
        this.originalList = l;
    }

    @Override public T3 apply() {
        List<T2> mappedList = new ArrayList<T2>();
        for (int i = 0; i < originalList.size(); i++) {
            mappedList.add(i,mapOperation(originalList.get(i)));
        }

        T3 result = initValue();
        for (int i = 0; i < mappedList.size(); i++) {
            result=reduceOperation(result, mappedList.get(i));
        }
        return result;
    }

    /* שימוש ב- for each בפתרון הנ"ל אפשרי גם כן */

    abstract public T2 mapOperation(T1 element);
    abstract public T3 reduceOperation(T3 partial, T2 element);
    abstract public T3 initValue();
}
```

ב. (8 נקודות) המחלקה `SumOfSquares` יורשת מהמחלקה `AbstMapReduceTask` שהוגדרה בסעיף הקודם. הפעלת השרות `apply` על עצם מהמחלקה `SumOfSquares` מחזירה את סכום ריבועי רשימת המספרים שהועברה בבנאי. להלן דוגמת שימוש במחלקה `SumOfSquares`:

```
List<Integer> input = Arrays.asList(1,2,3,4);
MapReduceTask<Integer, Integer, Integer> task = new SumOfSquares(input);
System.out.println(task.apply()); // prints 30
```

ממשו את המחלקה `SumOfSquares` לפי התאור לעיל:

```
public class SumOfSquares
    extends AbstMapReduceTask<Integer, Integer, Integer>{

    public SumOfSquares(List<Integer> l) {
        super(l);
    }

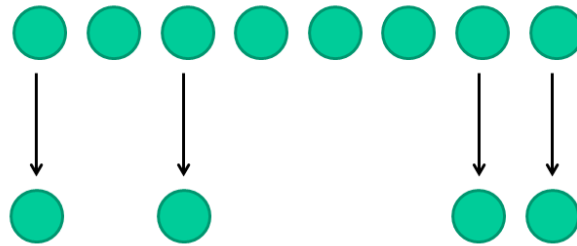
    @Override
    public Integer mapOperation(Integer element) {
        return element * element;
    }

    @Override
    public Integer reduceOperation(Integer partial, Integer e2) {
        return partial + e2;
    }

    @Override
    public Integer initialValue() {
        return 0;
    }
}
```

מסן (FilterTask) הוא טיפוס אשר פועל על רשימה כלשהי ומייצר **תת-רשימה** שלה לפי קריטריון כלשהו. המתכנתת אולגה מציעה להשתמש במחלקה `AbstMapReduceTask` כדי לממש את המחלקה המופשטת `FilterTask` באופן הבא: המסן יעבור על כל אברי הרשימה הנתונה בזה אחר זה, ויפעיל על כל אחד מהם את הפונקציה המופשטת הבולאנית `test`. רק אם הפונקציה מחזירה `true` האיבר יכלל ברשימה המסוננת.

ניתן להניח כי האיברים ברשימה המועברת כארגומנט לבנאי אינם `null`.



להלן דוגמת שימוש במחלקה `Evens` אשר יורשת מהמחלקה `FilterTask` ומייצרת את רשימת המספרים הזוגיים מתוך רשימת מספרים שלמים כלשהי:

```
List<Integer> input = Arrays.asList(1,2,3,4);
FilterTask<Integer> task = new Evens(input);
System.out.println(task.apply()); // prints [2,4]
```

ממשו את המחלקות `FilterTask` ו-`Evens` לפי התאור לעיל. (בשני העמודים הבאים)

ג. (15 נקודות) השלימו את קוד המחלקה FilterTask:

```
public abstract class FilterTask<T1> extends AbstMapReduceTask<T1,
T1, List<T1>>{

    public FilterTask(List<T1> l) {
        super(l);
    }

    public abstract boolean test(T1 element);

    @Override
    public T1 mapOperation(T1 element) {

        if(test(element))
            return element;
        return null;
    }

    @Override
    public List<T1> reduceOperation(List<T1> partial, T1 element) {

        if(element != null)
            partial.add(element);
        return partial;
    }

    @Override
    public List<T1> initialValue() {

        return new ArrayList<T1>();
    }

}
```

ד. (5 נקודות) ממשו את המחלקה Evens:

```
public class Evens extends FilterTask<Integer>{  
  
    public Evens(List<Integer> l){  
        super(l);  
    }  
  
    @Override  
    public boolean test(Integer element){  
        return element % 2 == 0;  
    }  
}
```


שאלה 3 (20 נקודות)

הסעיפים בשאלה זו מתייחסים לשלוש המחלקות הבאות:

```
public abstract class A {
    private static int value = 5;

    public A(){
        System.out.println(bar());
    }

    public abstract int foo(int a);

    *****
}
```

```
public class B extends A {
    private static int value = 7;

    public int foo(int b) {
        return b;
    }
    public int bar(){
        return foo(value);
    }
}
```

```
public class C {

    public static void main (String[] args){
        A a = new B();
    }
}
```

בכל אחד מהסעיפים הבאים מוחלפת שורת המכוביות בקטע קוד. הינכם מתבקשים לציין מהו הפלט של הרצת פונקציית ה main של המחלקה C בכל אחד מהמקרים. אם לדעתכם אין פלט לתכנית מכיוון שאינה עוברת קומפילציה או מכיוון שקיימת שגיאה בזמן ריצה (זריקת חריג), הסבירו מה השגיאה. פתרון ללא הסבר לא יזכה בנקודות.

א. (4 נקודות)

```
public abstract int bar();
```

Solution: 7

The constructor will call for the abstract method "bar" which calls the method "foo" and it returns the field "value" of B.

ב. (4 נקודות)

```
public int bar() { return 2; }
```

Solution: 7
The overridden method "bar" of B is called.

ג. (4 נקודות)

```
public final int bar() { return 2; }
```

Solution: compilation error
Cannot override the final method from A.

ד. (4 נקודות)

```
public int bar() throws RuntimeException { return 2; }
```

Solution: 7
Overridden A.bar with B.bar and it was called. The overriding is legal since RuntimeException is an unchecked Exception.

ה. (4 נקודות)

```
public int bar(int a) { return 2; }
```

Solution: Compilation error
The Method bar(int) in the type A is not applicable for the arguments(). A cannot assume that it's inheriting class will have a method called "bar()".

שאלה 4 (25 נקודות)

טוויטר היא רשת חברתית מקוונת המאפשרת לשלוח ולקרוא מסרים קצרים של עד 140 תווים. כל משתמש בטוויטר יכול לכתוב מסר (tweet). כל העוקבים אחר הפרופיל של אותו משתמש יוכלו לראות את המסר בדף הבית שלהם.

בשאלה זו נכתוב את האפליקציה TwitterStalker המציגה על המסך את המסרים (Tweets) שפורסמו על ידי משתמשי הטוויטר שאחריהם עוקב המשתמש (הרשימה שמורה בקובץ), תוך כדי שימוש במנשק גרפי.

א. (7 נקודות) השלימו את מימוש המתודה `getNamesFromFile`, אשר קוראת שמות של משתמשים מתוך קובץ, אשר שמו מועבר כארגומנט לפונקציה. המתודה מחזירה רשימה המכילה את שמות המשתמשים (`List<String>`). קובץ הקלט בנוי כך שבכל שורה מופיע שם משתמש בודד בלבד. לדוגמא:

```
ladygaga
katyperry
telavivuniv
```

```
public class HelperUtils {

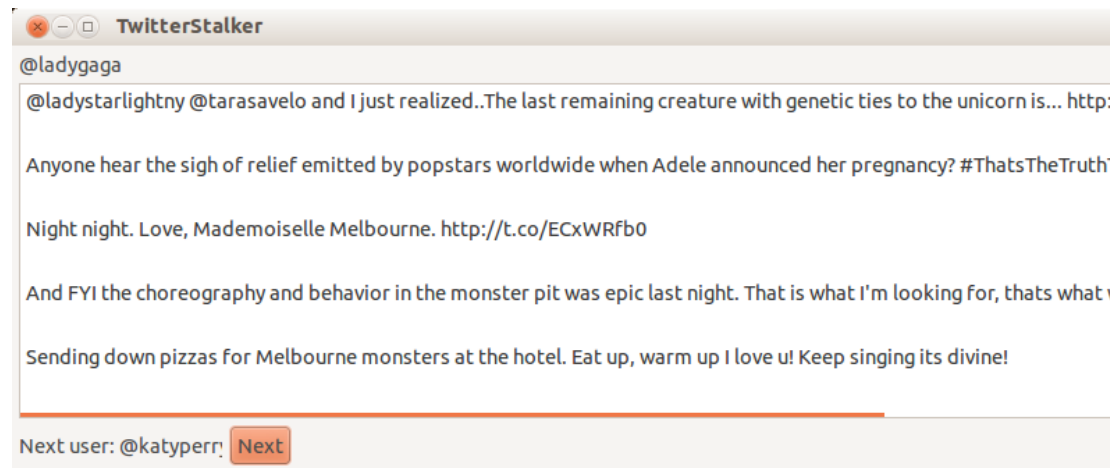
    /**
     * @param fullpath full path to the data file.
     * @return list of String names from the file fullpath. Each line in the
     *         file includes one name.
     */
    public static List<String> getNamesFromFile(String fullpath){

        Scanner s = null;
        List<String> usernames = new ArrayList<String>();
        if (fullpath == null){
            return usernames;
        }
        try {
            s = new Scanner(new File(fullpath));
            while (s.hasNextLine()) {
                String name = s.nextLine();
                usernames.add(name);
            }
        } catch (FileNotFoundException e) {
            return null;
        }finally{
            s.close();
        }
        return usernames;
    }

    /**
     * @return a list of tweets (as strings) from the given user's profile.
     */
    public static List<String> getRecentTweets(String user){
        // Some twitter stuff... No need to implement
        // To be used in later sections!
    }
}
```

ב. (18 נקודות) ביל מימש ממשק משתמש גרפי (GUI) עבור היישום TwitterStalker, אשר מציג את רשימת כל המסרים שפרסם כל אחד מהמשתמשים שאחריהם הוא עוקב.

בשורה העליונה מופיע שם המשתמש הנוכחי (למשל, בדוגמא למטה: ladygaga), לאחר מכן, מוצגים רשימת המסרים (Tweets) שלו, ולבסוף בשורה התחתונה מוצג שם המשתמש הבא (בדוגמא למטה: katyperry), שאליו ניתן לעבור בלחיצה על כפתור Next.



- תפקיד המתודה `advanceUser` הוא לקדם את האינדקס של המשתמש הנוכחי אל המשתמש הבא.
- תפקיד המתודה `updateGUI` הוא לבצע את השינויים הדרושים ב- `widgets` של הממשק הגרפי על מנת להציג את המידע המתאים למשתמש הנוכחי באופן הבא:
 - אם מנסים להתקדם מעבר למשתמש האחרון המתודה לא תבצע אף שינוי בממשק הגרפי.
 - אם המשתמש הנוכחי הוא המשתמש האחרון אזי יודפס ל- `BottomLabel` `Next user: "NONE"`, ובנוסף יש לבטל את פעולת הכפתור `Next` בעזרת המתודה `setEnabled(Boolean arg0)`.

ביל הדפיס את קוד המחלקה שמימש כדי להראות לאומה, אך אבוי, בדרך עצר במטבחון שם הניח את כוס הקפה על התדפיס. השלימו את קוד המחלקה (במקומות המסומנים בכתמי קפה).

```
public class GUI {

    private final static String filepath = "data.txt";

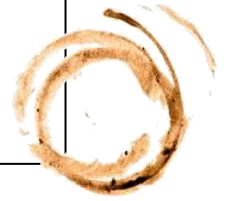
    private int currentUserIndex;

    private Display display;
    private Shell shell;
    private Label topLabel;
    private Label bottomLabel;
    private Button nextButton;
    private org.eclipse.swt.widgets.List listWidget;
    private List<String> userNameList;
```

```
public GUI() {
    userNameList = HelperUtils.getNamesFromFile(filepath);

    display = Display.getDefault();

    shell = new Shell(display);
}
```



```
public static void main(String[] args){
    GUI view = new GUI();
    view.showGUI();
}
```

```
public void showGUI() {
    createShell();
    while (!shell.isDisposed()) {
        if (!display.readAndDispatch())
            display.sleep();
    }
    display.dispose();
}
```

```
// this method builds the GUI including the listeners
private void createShell() {
    shell.setText("TwitterStalker");
    shell.setLayout(new GridLayout(2, false));

    topLabel = new Label(shell, SWT.NONE);
    GridData gridData = new GridData();
    gridData.horizontalSpan = 2;
    topLabel.setLayoutData(gridData);

    listWidget = new org.eclipse.swt.widgets.List (shell, SWT.BORDER |
                                                    SWT.MULTI | SWT.V_SCROLL | SWT.H_SCROLL);
    gridData = new GridData();
    gridData.horizontalSpan = 2;
    listWidget.setLayoutData(gridData);

    bottomLabel = new Label(shell, SWT.NONE);
    gridData = new GridData();
    gridData.horizontalSpan = 1;
    bottomLabel.setLayoutData(gridData);
}
```

המשך בעמוד הבא!

```

nextButton = new Button (shell, SWT.PUSH);
nextButton.setText ("Next");
nextButton.setLocation(0, 0);
nextButton.setSize(100, 30);
nextButton.addSelectionListener(

```

```

    new SelectionAdapter() {
        public void widgetSelected(SelectionEvent e){
            updateGUI();
            advanceUser();
        }
    }

```

```

);

```

```

updateGUI(); // updates the GUI widgets for the first time
advanceUser(); // advance the index to the next user
shell.pack(); // causes the layout manager to lay out the shell
shell.open(); // opens the shell on the screen
}

```

```

/** Updates the UI to show the tweets of the current user. Also updates the
 * names of the current and next users
 */

```

```

private void updateGUI() {

```

```

    if (currentUserIndex >= userNameList.size()){
        return;
    }
    String currentUser = userNameList.get(currentUserIndex);
    String nextUser;
    if (currentUserIndex >= userNameList.size()-1){
        nextButton.setEnabled(false);
        nextUser = "NONE";
    }else{
        nextUser = userNameList.get(currentUserIndex+1);
    }
    topLabel.setText("@"+currentUser);
    bottomLabel.setText("Next user: "+nextUser);
    List<String> tweets = HelperUtils.getRecentTweets(currentUser);
    listWidget.removeAll();
    for (String tweet: tweets){
        listWidget.add (tweet);
    }
}

```

```

}

```

```

/**
 * Advances the index to indicate the next user.
 */

```

```

private void advanceUser(){

```

```

    currentUserIndex++;
}

```

```

}

```

```

}

```

ושוב, בהצלחה!