בס"ד

מטלה 1 – מונחה עצמים

חלק א'

סעיף 6 – בסעיף זה התבקשנו לפרט אודות מבני הנתונים והאלגוריתמים ואופן השימוש בפרויקט.

בשלב הראשון מימשנו את תבנית העיצוב של Observer:

Observe היא תבנית עיצוב שבה אובייקט הייחוס מחזיק רשימה של אובייקטים (במקרה שלנו Member) שתלויים בו (מקושרים אליו) אשר כל אחד מהם נקרא צופה לגביו.

בקוד שלנו יצרנו Array List של אובייקטים מטיפוס Member ובהמשך הקוד נראה כיצד הם מקושרים לאובייקט הייחוס UndoableStringBuilder.

```
ArrayList<Member> members = new ArrayList<~>();
UndoableStringBuilder s = new UndoableStringBuilder();
```

אובייקט הייחוס "מודיע" לצופים עליו על שינויים המתרחשים בו, כגון שינוי ערך של פרמטר, בדרך כלל על ידי קריאה לאחת מהפונקציות שלהם.

על מנת לממש את ה – observer בנינו פונקציית הוספה והסרה של Member (צופים) מה - Array List.

```
/**
 * added new Member to the members ArrayList
 * @param obj
 */
@Override
public void register(Member obj) {
    members.add(obj);

}
```

```
/**
 * delete Member from the members ArrayList
 * @param obj
 */

@Override
public void unregister(Member obj) {
    members.remove(obj);

}
```

לאחר מכן נרצה לעדכן בכל פעולה שנעשתה באובייקט הייחוס שלנו, למשל אם ב-UndoableStringBuilder נעשתה הכנסה כלשהי, נרצה לעדכן את כל ה-Members באופן הבא :

```
/**
 * Inserts the string into this character sequence.
 * After the insertion we will update all the Members
 * @param offset - int object.
 * @param obj    - String object.
 *
 */

@Override
public void insert(int offset, String obj) {
    s.insert(offset,obj);
    for (int i = 0; i < members.size() ; i++) {
        members.get(i).update(s);
    }

}
```

אנו רצים בלולאה ומעדכנים כל אובייקט מסוג Member שנמצא ב-Array List members שהגדרנו בהתחלה.

העדכון נעשה בעזרת הפונקציה update ששייכת ל-interface Member.

נוכל לחשוב על דוגמא שמסבירה את מימוש תבנית העיצוב במקרה שלנו. למשל, אפליקציית YouTube - כאשר זמר מסויים מעלה סרטון חדש לעמוד שלו ב-YouTube, האנשים שעוקבים אחריו מעודכנים בזאת. כך גם אצלנו, כל שינוי שנעשה ב – UndoableStringBuilder יעדכן את ה-members.