

משימה #5 בקורס תוכנה 1

הוראות הגשה:

1. קראו בעיון את קובץ נוהלי הגשת התרגילים אשר נמצא באתר הקורס.
2. הגשת התרגיל תעשה ע"י המערכת VirtualTAU (<http://virtual2002.tau.ac.il/>).
3. הגשת התרגיל תתבצע ע"י יצירת קובץ zip שנושא את שם המשתמש. לדוגמא, עבור המשתמש zvainer יקרא הקובץ zvainer.zip.
קובץ ה zip יכיל:
 - א. קובץ פרטים אישיים בשם details.txt המכיל את שמכם ומספר ת.ז. זהות שלכם.
 - ב. קבצי ה-java של התכניות שהתבקשתם לכתוב.
 - ג. קובץ טקסט עם העתק של כל קבצי ה Java.
 - ד. קובץ טקסט עם פתרונות לשאלות בהן לא נדרשתם לכתוב קוד

חלק א':

מטרת חלק זה לתרגל כתיבת שירותים סטטיים לא טריוויאליים. ס בהינתן החוזה של השירות. בכל סעיף מוגדר שירות למימוש עם שני חוזים אפשריים. עליכם לממש את השירות פעמיים, כך שיתאים לחוזים. ברוב המקרים ניתן לפתור את התרגיל בקלות ע"י הוספת **פונקציות עזר** (אף שהתרגיל לא דורש זאת במפורש).

הערה: במידת הצורך, עבור כל זוג פונקציות תוכלו להשתמש באחת הפונקציות לצורך מימוש השנייה.

1. בהינתן מערך כקלט, החזר מערך המכיל את אותם מספרים, אך מסודר כך שלאחר כל מופע של 4 יש מופע של 5. מותר להזיז את כל אברי המערך חוץ מהאברים שערכם 4. להלן מספר דוגמאות:

`fix45({5,4,9,4,9,5}) → {9,4,5,4,5,9}`

`fix45({1,4,1,5}) → {1,4,5,1}`

`fix45({1,4,1,5,5,4,1}) → {1,4,5,1,1,4,5}`

```
/*
 * @pre occurrences(4,arr) == occurrences(5, arr)
 * @pre arr[arr.length - 1] != 4
 * @pre forall 0 <= i < arr.length-2, arr[i] == 4 ==> arr[i+1] != 4
 * @post forall 0 <= i < arr.length-1, $prev(arr[i]) == arr[i]
 * @post $ret != arr
 * @post $ret.length == arr.length
 * @post forall 0 <= i < $ret.length-2, $ret[i] == 4 => $ret[i+1] == 5
 * @post forall 0 <= i < arr.length-1, arr[i] == 4 ==> $ret[i] == 4
 */
public static int[] fix45A (int[] arr) {

}
```

```

/*
 * @post ((occurrences(4,arr) == occurrences(5, arr)) AND
 *        (arr[arr.length - 1] != 4) AND
 *        (forall 0 <= i < arr.length-2, arr[i] == 4 ==> arr[i+1] != 4))
 *        ==>
 *        ((forall 0 <= i < arr.length-1, $prev(arr[i]) == arr[i]) AND
 *        ($ret != arr) AND
 *        ($ret.length == arr.length) AND
 *        (forall 0 <= i < $ret.length-2, $ret[i]==4 ==> $ret[i+1]==5) AND
 *        (forall 0 <= i < arr.length-1, arr[i] == 4 ==> $ret[i] == 4))
 */
public static int[] fix45B (int[] arr) {

}

```

2. בהינתן מערך של מספרים שלמים (integers), האם ניתן לבחור תת קבוצה מתוך המערך כך שסכום תת הקבוצה שווה למספר מטרה מסוים? ניתן לפתור בעיה זו ע"י רקורסיה. טיפ: במקום להסתכל על כל המערך, נסתכל על המערך החל מאינדקס start ועד לסופו. הקורא לשירות זה יכול לציין שברצונו לפתור עבור כל המערך ע"י נתינת ערך 0 ל-start. להלן כמה דוגמאות:

groupSum(0, {2, 4, 8}, 10) → true

groupSum(0, {2, 4, 8}, 14) → true

groupSum(0, {2, 4, 8}, 9) → false

```

/*
 * @pre 0 <= start <= nums.length - 1
 * @pre nums != null
 * @pre  $\forall 0 \leq i \leq \text{nums.length}-1, \text{nums}[i] \in \mathbf{Z}$ 
 * @post  $(\exists S \subseteq \text{nums}, \sum_{s \in S} s = \text{target}) \Rightarrow \$ret == \text{true}$ 
 * @post  $(\neg \exists S \subseteq \text{nums}, \sum_{s \in S} s == \text{target}) \Rightarrow \$ret == \text{false}$ 
 */
public static boolean groupSumA(int start, double[] nums, int target) {

}

```

```

/*
 * @pre 0 <= start <= nums.length - 1
 * @post  $(\exists S \subseteq [0..nums.length-1], \sum_{s \in S} nums[s] = target) \Rightarrow \$ret == true$ 
 * @post  $(\neg \exists S \subseteq [0..nums.length-1], \sum_{s \in S} nums[s] == target) \Rightarrow \$ret == false$ 
 */
public static boolean groupSumB(int start, int[] nums, int target) {

}

```

3. בהינתן מחרוזת, החזירו true אם מספר המופעים של תת המחרוזת "is" במקום כלשהו במחרוזת שווה למספר ההופעות של תת המחרוזת "not" במחרוזת (case sensitive). להלן כמה דוגמאות:

```

equalIsNot("This is not") → false
equalIsNot("This is notnot") → true
equalIsNot("noisxxnotyynotxisi") → true

```

```

/*
 * @pre str != null
 * @post $ret == (occurrences("is") == occurrences("not"))
 */
public static boolean equalIsNotA(String str) {

}

```

```

/*
 * @pre (str != null) AND (occurrences("is") == occurrences("not"))
 * @post $ret == (occurrences("is") == occurrences("not"))
 */
public static boolean equalIsNotB(String str) {

}

```

4. ניתנים כקלט שני מערכים של מחרוזות, a ו- b ללא כפילויות. החזירו את מספר המחרוזות המופיעות בשני המערכים. הפתרון צריך להיות יעיל ככל האפשר, "לינארי" אם המחרוזות ממוינות, ז"א עובר פעם אחת על המערכים. להלן כמה דוגמאות:

`sharedStr({"Call", "me", "Ishmael"}, {"Call", "me", "Jonha"}) → 2`

2

`sharedStr({"a", "c", "x"}, {"z", "b", "c", "x", "a"}) → 3`

`sharedStr({"a", "b", "c"}, {"a", "b", "c"}) → 3`

```

/*
 * @pre a != null AND b != null
 * @pre forall i, j; (0 <= i <= a.length-1 AND 0 <= j <= a.length-1) ==>
 *   ((i+1 == j) ==> (a[i] <= a[j]))
 * @pre forall i, j; (0 <= i <= b.length-1 AND 0 <= j <= b.length-1) ==>
 *   ((i+1 == j) ==> (b[i] <= b[j]))
 * @pre not exists i, j; i != j ==> a[i].equals(a[j])
 * @pre not exists i, j; i != j ==> b[i].equals(b[j])
 * @post |S| where S ≡ {s | ∃ i, j; s.equals(a[i]) ∧ s.equals(b[j])}
 */
public static int sharedStrA(String[] a, String[] b) {

}

```

```

/*
 * @pre not exists i, j; i != j ==> a[i].equals(a[j])
 * @pre not exists i, j; i != j ==> b[i].equals(b[j])
 * @post |S| where S ≡ {s | ∃ i, j; s.equals(a[i]) ∧ s.equals(b[j])}
 */
public static int sharedStrB(String[] a, String[] b) {

}

```

חלק ב':

חלק זה נועד לתרגל כתיבת חוזים עבור שירותים קיימים. בכל סעיף נתונה פונקציה, עליכם כתוב את החוזה (תנאי קדם ואחר) עבור הפונקציה הנתונה.

1. שורש

```
public static double sqrt(double d) {...}
```

2. ערך מוחלט

```
public static double abs(double d) {...}
```

3. עצרת

```
public static int factorial(int n) {  
    int fact = 1;  
    for (int i = 1; i <= n; ++i) {  
        fact *= i;  
    }  
    return fact;  
}
```

4. מיון

```
public static void sort(int[] arr) {  
    int first, location, temp;  
  
    for (first = 1; first < arr.length; first++) {  
        if (arr[first] < arr[first - 1]) {  
            temp = arr[first];  
            location = first;  
  
            do {  
                arr[location] = arr[location-1];  
                location--;  
            }  
            while (location > 0 && arr[location-1] > temp);  
  
            arr[location] = temp;  
        }  
    }  
}
```

חלק ג':

נתונה המחלקה Rational המייצגת מספר רציונאלי (ללא מימוש). יש לציין את המצב המופשט (abstract state) של הטיפוס. כמו כן, עבור כל אחד מהשירותים יש לציין את משמעותו בעזרת המצב המופשט.

אין צורך לממש את השירותים!

```
public class Rational {  
    public int getNumerator() {...}  
    public int getDenominator() {...}  
    public Rational add(Rational other) {...}  
    public Rational subtract(Rational other) {...}  
    public Rational multiply(Rational other) {...}  
    public Rational divide(Rational other) {...}  
    public boolean equals(Rational other) {...}  
    public String toString() {...}  
}
```

חלק ד':

בחלק זה נתרגל כתיבת מחלקות בהינתן תיאור של השירותים. עליכם לממש מחלקה בשם DisjointSets המייצגת קבוצה S של disjoint sets $S = \{S_1, S_2, \dots, S_n\}$ כך שכל S_i הינו קבוצה של מספרים שלמים אי-שליליים. המחלקה תתמוך בשירותים הציבוריים הבאים:

```
/**
 * Create a set containing x (i.e. {x}) and add it to
 * this object. The given integer x should not be a
 * member of any other set in self.
 */
public void makeSet(int x);

/**
 * Return true if and only if x and y belong to the same
 * set in this object.
 */
public boolean equiv(int x, int y);

/**
 * Find the different sets that x and y belong to. Remove
 * them from this object and add their union
 */
public void joinSets(int x, int y);

/**
 * Return true if and only if x is in some set of this
 * object
 */
public boolean inASet(int x);
```

דרך אחת לממש את המחלקה היא ע"י ייצוג כל סט S_i של S בתור עץ. כל צומת (הקשורה למספר שלם אי שלילי x) מצביעה להורה. בדרך זו, שורש של עץ מייצג בצורה ייחודית את הסט S_i . מערך בשם parent יכול להחזיק את כל המצביעים האלה. ספציפית, לכל מספר x , הערך של $parent[x]$ הוא מספר צומת האב בעץ או 1- אם x לא שייך לאף סט S_i . שורש כל עץ מצביע לעצמו. אורך המערך צריך להיות ארוך יותר מכל מספר x שנמצא כרגע בסט S_i כלשהוא. לפיכך, אם נוסף מספר גדול יותר מהמקסימלי עד כה, נצטרך להחליף את המערך במערך חדש גדול יותר.

כל אובייקט של המחלקה DisjointSets מייצג קבוצה של קבוצות זרות, של שלמים אי-שליליים

$\{S_1, \dots, S_n\}$. ייצוג זה מוגדר פורמלית ע"י פונקצית העזר $r(x)$:

```
If parent[x] = -1 then r(x) = -1
Else if parent[x] == x then r(x) = x
Else r(x) = r(parent[x])
```

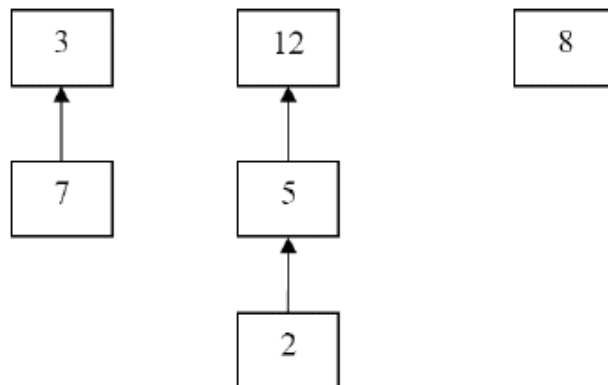
כעת, נוכל להגדיר את המיפוי מ-parent, שדה של האובייקט, לסט של סטים:
 $S(\text{this}) = \{S_1, S_2, \dots, S_n\}$ כך ש:

For all x, [for all i, $1 \leq i \leq n$, $x \notin S_i$] iff $[x \geq \text{parent.length or parent}[x] = -1]$
 For all $0 \leq x, y \leq \text{parent.length}$, $x, y \in S_i$ (x and y are in the same set) iff $r(x) = r(y) \neq -1$

למשל, לאובייקט של DisjointSets כאשר מערך ה parent הוא:

| | | | | | | | | | | | | |
|----|----|---|---|----|----|----|---|---|----|----|----|----|
| -1 | -1 | 5 | 3 | -1 | 12 | -1 | 3 | 8 | -1 | -1 | -1 | 12 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

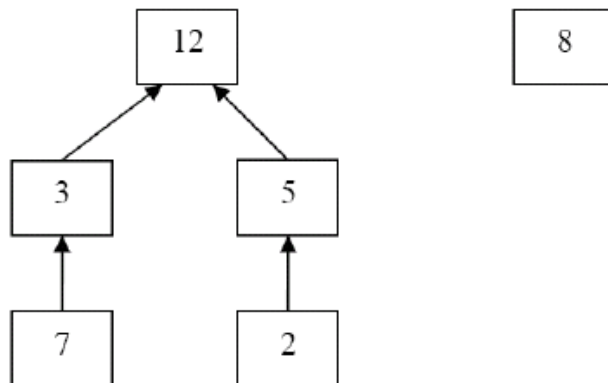
העצים יהיו:



ו- $S = \{ \{3,7\} \{12,5,2\} \{8\} \}$. אם נקרא ל- $\text{equiv}(3,5)$ נקבל false. אם נפעיל $\text{joinSets}(7,5)$ נקבל:

| | | | | | | | | | | | | |
|----|----|---|----|----|----|----|---|---|----|----|----|----|
| -1 | -1 | 5 | 12 | -1 | 12 | -1 | 3 | 8 | -1 | -1 | -1 | 12 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

העצים יהיו:



ו- $S = \{\{3,7,12,5,2\}, \{8\}\}$. אם נפעיל כעת $\text{makeSet}(4)$ נקבל ש- $\{4\}$ $S = \{\{3,7,12,5,2\}, \{8\}, \{4\}\}$

המשימה:

ממשו את המחלקה DisjointSets בצורה יעילה תוך שימוש במערך parent (ניתן להוריד תכנית חלקית מאתר הקורס). שימו לב שניתן להגדיר שירותי עזר. לכל שירות הגדירו תנאי קדם ואחר.