רקורסיה במחסנית

תרגיל 1

```
public static int what (Stack<Integer> stk)
{
        if (stk.isEmpty())
                return 0;
        int x = stk.pop();
        int sod = what (stk);
        stk.push (x);
        return sod + 1;
}
```

טענת כניסה: stk מחסנית של מספרים שלמים.

עקוב אחר ביצוע הפעולה בטבלת מעקב ורשום את טענת היציאה של הפעולה.

דוגמת מעקב: עבור המחסנית stk: [3, 9, 12, 5, 4, 6]

המחסנית בחזרה	ערך מוחזר sod	x	מחסנית ריקה?	המחסנית בזימון
[3, 9, 12, 5, 4, 6]	__5_ + 1 = 6	3	לא	[~~3~~, 9, 12, 5, 4, 6]
[9, 12, 5, 4, 6]	__4_ + 1 = 5	9	לא	[~~9~~, 12, 5, 4, 6]
[12, 5, 4, 6]	__3_ + 1 = 4	12	לא	[~~12~~, 5, 4, 6]
[5, 4, 6]	__2_ + 1 = 3	5	לא	[~~5~~, 4, 6]
[4, 6]	__1_ + 1 = 2	4	לא	[~~4~~, 6]
[6]	__0_ + 1 = 1	6	לא	[~~6~~]
[ ]	0		כן	[ ]

הערך המוחזר: 6

הפעולה סופרת ומחזירה את מספר האיברים במחסנית.

שימו לב! בסיום המחסנית נשארת ללא שינוי

תרגיל 2

לפניכם פעולה רקורסיבית:

```
public static void what (Stack<Integer>s1, Stack<Integer> s2)
{
        if (! s1.isEmpty())
        {
                int x = s1.pop();
                what (s1, s2);
                s1.push(x);
                s2.push(x);
        }
}
```

טענת כניסה: s2 מחסנית ריקה.

עקוב אחר ביצוע הפעולה בטבלת מעקב ורשום את טענת היציאה של הפעולה.

תרגיל 3

לפניכם פעולה רקורסיבית:

```
public static boolean mystery (Stack<Integer> stk, int x)
{
        if (stk.isEmpty())
                return false;
        int y = stk.pop();
        if (x == y)
        {
                stk.push (y);
                return true;
        }
        boolean sod = mystery (stk, x);
        stk.push (y);
        return sod;
}
```

טענת כניסה: stk מחסנית של מספרים שלמים. x מספר שלם.

עקוב אחר ביצוע הפעולה בטבלת מעקב ורשום את טענת היציאה של הפעולה.

תרגיל 4

לפניכם פעולה רקורסיבית:

```
public static boolean mystery (Stack<Integer>s1, Stack<Integer>s2)
{
        if (s1.isEmpty() && s2.isEmpty())
                return true;

        if (s1.isEmpty() || s2.isEmpty())
                return false;

        int x = s1.pop();
        int y = s2.pop();

        boolean sod = (x == y)  &&  Mystery (s1, s2);

        s1.push (x);
        s2.push (y);

        return sod;
}
```

טענת כניסה: s1 ו- s2 מחסניות של מספרים שלמים

עקוב אחר ביצוע הפעולה בטבלת מעקב ורשום את טענת היציאה של הפעולה.

boolean sod = (x == y) && mystery (s1, s2);

שים לב, זהו ביטוי לוגי.
תוצאת הביטוי הלוגי תושם במשתנה בוליאני sod

בטבלת המעקב רשום עמודה עבור x == y,
עמודה עבור הערך המוחזר מהזימון הרקורסיבי
ועמודה עבור sod שיכיל את התוצאה של שתי העמודות הקודמות

תרגיל 5

לפניכם פעולה רקורסיבית המשתמשת בפעולה sod:

```
public static int what (Stack<Integer>stk)
{
	if (stk.isEmpty())
		return 0;
	int x = stk.pop();
	int whoAmI =  sod (x, (what (stk));
	stk.push (x);
	return whoAmI;
}

public static int sod (int x, int y)
{
	if (x > y)
		return x;
	return y;
}
```

טענת כניסה: stk מחסנית של מספרים שלמים

עקוב אחר ביצוע הפעולה בטבלת מעקב ורשום את טענת היציאה של הפעולה.

תרגיל 6

```
public static int What (Stack<Integer>stk)
{
	if (! stk.isEmpty())
	{
		int x = stk.pop();
		if (stk.isEmpty())
			return x;

		int b = what (stk);
		stk.push (x);
		return b;
	}
	return -1;
}
```

(המשך התרגיל בעמוד הבא)

```
public static boolean mystery (Stack<Integer>stk, int t)
{
	if (t > 0)
	{
		int a = what (stk);
		if (a == -1)
			return true;
		int b = stk.pop();
		if (a != b)
		{
			stk.push (b);
			return false;
		}
		return  mystery (stk, t-1);
	}
	return true;
}
```

א. נתונה המחסנית הבאה: stk: [12, 11, 10, 9, 8, 10, 11, 12]
עקוב אחר הפעולה what והמחסנית stk ורשום את טענת היציאה. התייחס למצב המחסנית בסיום הפעולה.

ב. עקוב אחר משפט הזימון: mystery (stk, 3) ורשום מה יוחזר. (אין צורך להראות מעקב נוסף אחר הזימון של what)

ג. מה יחזיר הזימון: mystery (stk, 4) (אין צורך להראות מעקב).

ד. רשום את טענת היציאה של הפעולה Mystery (stk, t). התייחס ל- t ולמצב המחסנית בסיום הפעולה.