SYNTAX
שפה מייצרת חשיבה

Lesson 6 – Functions & String

הנושאים הנלמדים:

פונקציות וספריית String

פונקציות

- קוד ללא פונקציות הוא ארוך, מסורבל ועם הרבה חזרות. מדוע?
- האם הפעולה sin עצמה משתנה כאשר מתבצעת על 30 ° או על 50 ° ?
- הפונקציות מאפשרות לנו להגדיר את sin כפעולה עצמאית שכל פעם מקבלת נתון אחר ומחזירה ערך אחר.
- מה שמשתנה זה הפרמטר המועבר והערך המוחזר, הפעולה עצמה לא משתנה ומכאן החסכון האדיר שבפונקציות!

פונקציות - מבנה

```
public static <טיפוס מוחזר> <שם> (<פרמטרים>)
{
        //הוראות
}
```

- פונקציה מגדירה "עולם" - קטע קוד עצמאי לחלוטין החי ברשות עצמו.
- כל המשתנים שהיא מגדירה בפנים כמו גם הפרמטרים משתחררים בסיומה.
- הפרמטרים אופציונלים ומופרדים בפסיק.
- פונקציה יכולה גם שלא להחזיר ערך ואז בטיפוס המוחזר רושמים void.
- return מלה שמורה לצורך החזרת ערך (צריך להתאים לטיפוס בכותרת).
- return מסיימת את הפונקציה (גם פונקציית void).

פונקציות - דוגמה

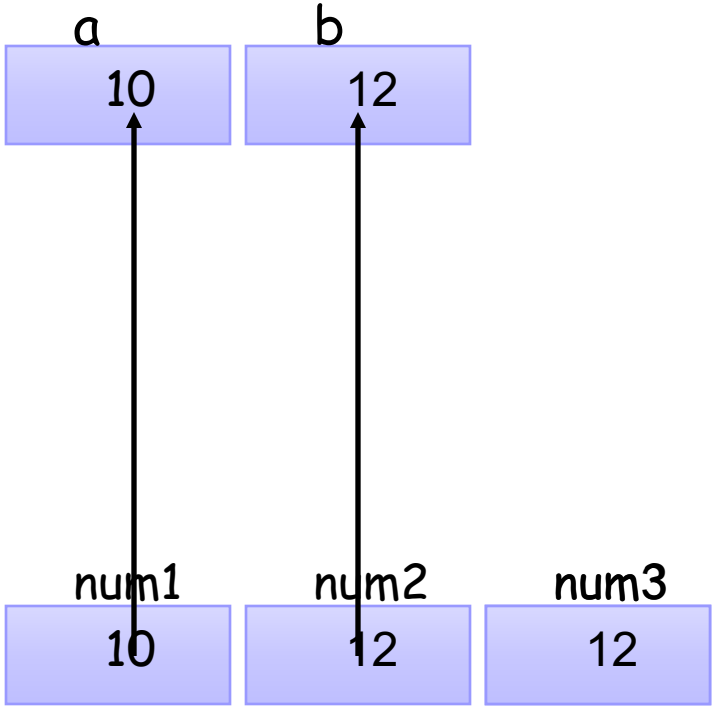
טיפוס מוחזר | שם | פרמטרים

```
public static int bigger(int a,int b)
{
        if(a>b)
                return a;
        return b;  // no need for else cause return finish the function
}
public static void main(String[] args)
{
        int num1=10;
        int num2=12;
        int num3=bigger(num1,num2);
        System.out.println("The bigger is " + num3);
}
```

בקריאה לפונקציה הערכים שמאחורי num1 ו num2 מועתקים לתוך a ו b והערך המוחזר נשמר ב num3

פונקציות - דוגמה

```
public static int bigger(int a,int b)
{
        if(a>b)
                return a;
        return b;
}
public static void main(String[] args)
{
        int num1=10;
        int num2=12;
        int num3=bigger(num1,num2);
        System.out.println("The bigger is " + num3);
}
```



פונקציה שאינה מחזירה ערכים

```
public static void print(int x,String str) {

    if(str==null || str.length()==0) return;

    for(int i=0;i<x;i++)
        System.out.println(str);
}

public static void main(String[] args) {

    print(10,"Hello world!");
}
```

פונקציית void
יכולה לסיים את
הפונקציה עם
return ריק

פונקציה מוכנה מראש של
המחלקה String המחזירה
את אורך המחרוזת

מצא את השגיאה

```
public static int bigger(int a,int b) {
    if(a>b) return a;
    if(b>a) return b;
}
```

- הפונקציה הבאה נותנת שגיאת קומפילציה " missing return statement"
- מדוע?
- במידה והפונקציה מחזירה ערך אנחנו צריכים שכל נתיבי הקוד יחזירו ערך
- מה קורה כאשר a==b?

פונקציות ו stack

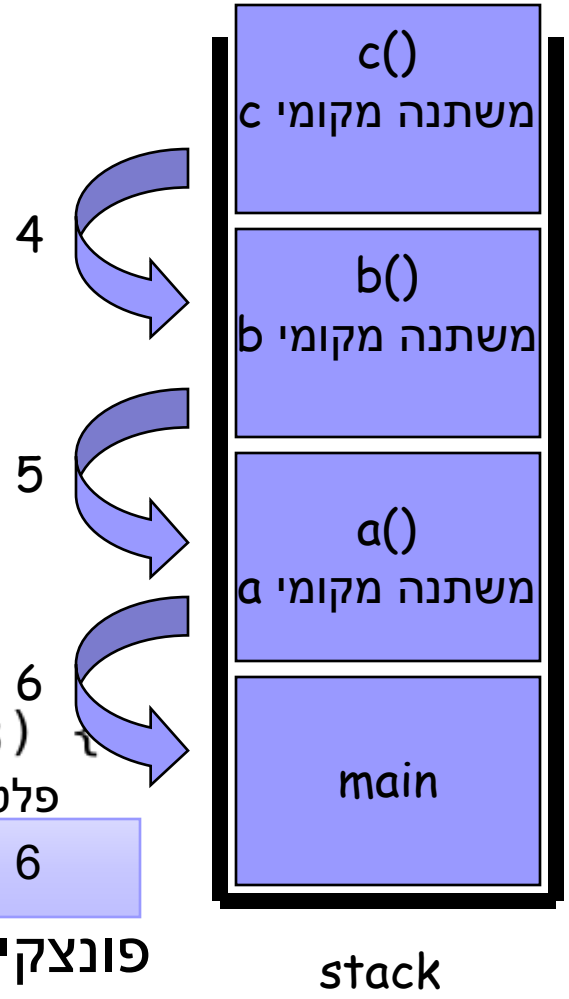
```
public static int a(int a) {
    return b(a) + 1;
}

public static int b(int b) {
    return c(b) + 1;
}

public static int c(int c) {
    return c+1;
}

public static void main(String[] args) {

    System.out.println(a(3));
}
```



פונצקיית ה main היא פונקציה מיוחדת - היא נקודת הכניסה - ממנה הכל מתחיל

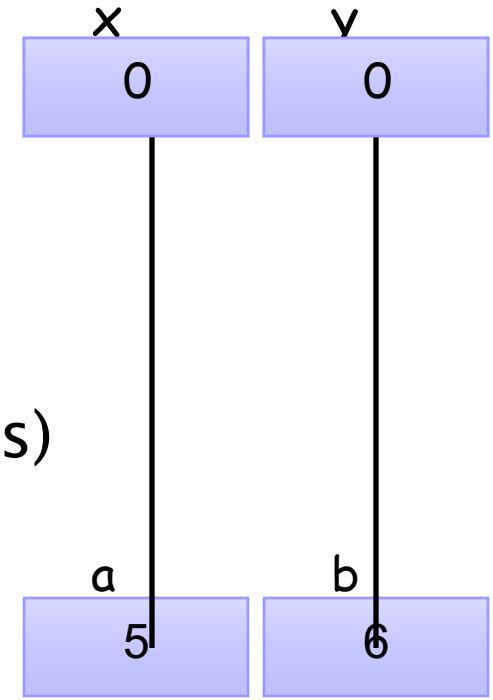
פונקציות ומערכים

- בהעברת פרמטרים פרימטיבים התוכן מועתק.
- הפונקציה עובדת על עותק משלה של המשתנים.
- העברה זו נקרת העברה **by value** **כלומר הערך הוא זה שמועבר וזה נכון לכל המשתנים הפרימיטיבים.**
- **מערך** מועבר לפונקציה **by reference** - התוכן המועתק למשתנה הלוקלי הוא **הכתובת של המערך.**
- שתי הכתובות **מצביעות לאותו מערך**, והפונקציה עובדת על אותו המערך המקורי.

דוגמה להעברה By value

```
public static void change(int x,int y)
{
        x=0;
        y=0;
}


public static void main(String[] args)
{
        int a=5;
        int b=6;
        change(a,b)
        System.out.println("a="+a+",b="+b);
}
```



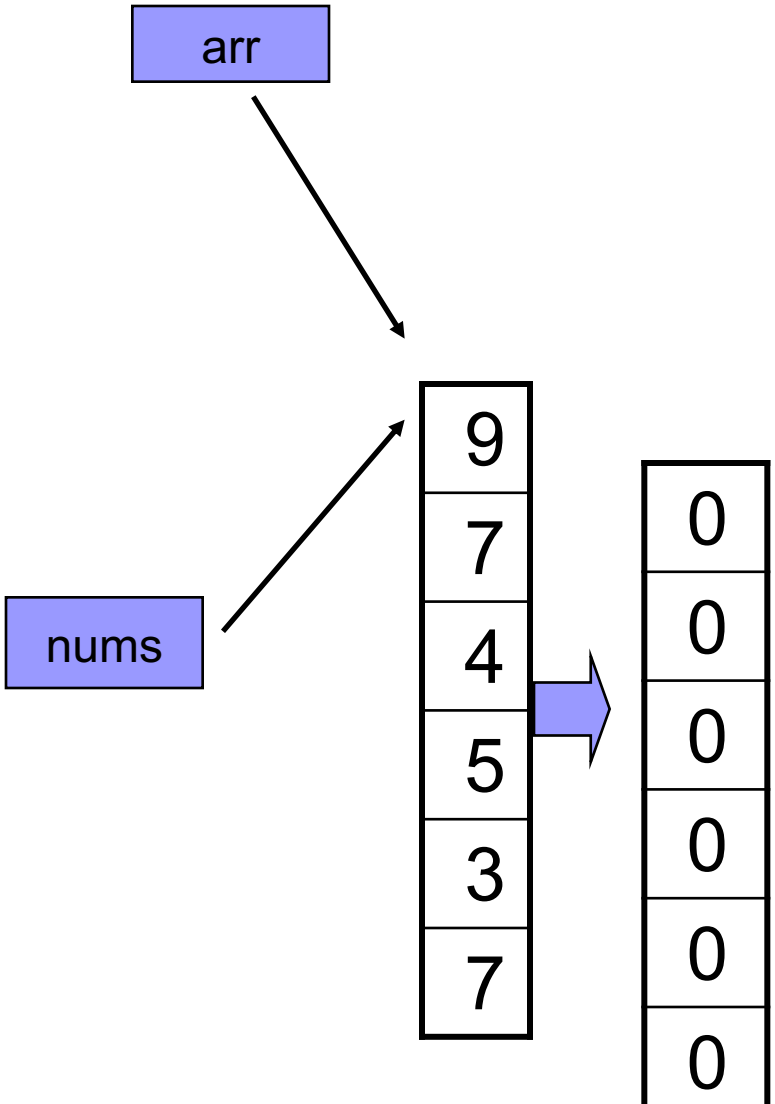
פלט a=5,b=6

העברה by reference דוגמה

```
public static void zeroArray(int[] arr)
{
        for(int i=0;i<arr.length;i++)
                arr[i] = 0;
}

public static void main(String[] args)
{
        int[] nums = {9,7,4,5,3,7}
        zeroArray(nums)

}
```



מחרוזות String

- String הוא אובייקט ונשמר כמו מערך על ה Heap
- String str = “moshe”;
- str הוא מצביע! אם לא ניתן לו ערך הוא יאותחל כמו מערך ל null.
- אם כן, מדוע לא עשינו:
- String str = new String("moshe”);
- קיצור של java עבורנו (עקב נפיצות).
- החשיבות היא למשל בבדיקת שוויון - לא להשתמש באופרטור == כמו פרימיטיבים (ישווה כתובות) - צריך להשתמש ב ()equals

מחרוזות String

- מחרוזת היא כמו מערך סטטית ואינה יכולה להשתנות!
- str = str + “ cohen”
- המערכת יצרה אובייקט חדש.
- המחלקה String היא חלק מה API של ג'אווה(הרבה מאוד פונקציות שימושיות לטיפול במחרוזות).
- את כל הפונקציות נפעיל באמצעות אורפטור הנקודה על המחרוזת שנרצה:

(פרמטרים)פונקציה.<שם המחרוזת>

מחרוזות – פונקציות שימושיות

- String str1 = “Moshe”;
- String str2 = “Cohen”;
- str1.length() -> 5
- str1.equals(str2) - > false
- str1.compareTo(str2) -> positive int
- str1.charAt(2) -> ‘s’ (מתחילים מ 0)
- str1.toLowerCase() -> “moshe”
- str1.toUpperCase() -> “MOSHE”
- str1.subString(1,3) -> “os” (לא כולל ה 3!)
- str1.indexOf(‘s’) -> 2

הפונקציה compareTo מחזירה מספר חיובי אם המחרוזת שהועברה כפרמטר באה במילון לפני המחרוזת עליה הפעלנו את הפונקציה, מספר שלילי במקרה ההפוך ואפס אם הן שוות לחלוטין