

## התקדמות לינארית

- בשבוע שעבר דנו בביטויים בשפה וראינו תוכניות פשוטות שמבצעות פעולות (בעיקר השמה) על ביטויים ומשתנים

- התוכנית התקדמה באופן קווי (לינארי) – החל בשורה הראשונה ב main והתוכנית התבצעה שורה אחר שורה, עד שהגיעה לסוף main והסתיימה

- השבוע נראה כיצד ניתן להתקדם באופן לא לינארי

```
public class StringExample {
    public static void main(String[] args) {
        String str = "SupercaliFragalistic";
        int len = str.length();
        String upper = str.toUpperCase();
        System.out.println("String length is " + len);
        System.out.println("String in UPPERCASE is '" + upper + "'");
    }
}
```

2

## תוכנה 1

תרגול 2: מערכים ומבני בקרה  
נעמה מאיר ומתי שמרת

1

## משפט if/else

- אם נרצה שאשר התנאי מתקיים יתבצע משפטים מסוימים וכאשר הוא אינו מתקיים יתבצע משפטים אחרים נשתמש במשפט else

```
...
if (grade>60) {
    System.out.println("You passed the test!");
    System.out.println("Hip Hip Hooray !");
} else {
    System.out.println("You failed!");
    System.out.println("It is so sad...");
}
System.out.println("Your grade is: " + grade);
```

- אם התנאי אינו מתקיים התוכנית מדלגת על ביצוע פסוקי-אז וקופצת לשורת ה else משפט else יכול להכיל משפט אחד, בלוק או לא להופיע כלל
- משפט else הוא בעצם פסוקי (פסוקי-אחרת, else-clause) - הוא יכול להופיע רק אחרי פסוקי-אז

- מה יודפס עבור grade שהוא 740
- מה יודפס עבור grade שהוא 100

4

## משפט if

- ביצוע מותנה של משפט (מזכיר ביטוי ב-scheme)

```
if ( <boolean_expression> )
    <statement>
```

דוגמא:

```
public class IfExample {
    public static void main(String[] args) {
        int grade = Integer.parseInt(args[0]);

        if ( grade > 60 )
            System.out.println("You passed the test!");
        System.out.println("Your grade is: " + grade);
    }
}
```

- ליבטוי בסוגריים חייב להיות ערך בולאני
- אם ערכו הוא true יתבצע המשפט או הבלוק המופיע מיד אחרי הסוגריים (פסוקי-אז, then-clause), ואחר כך תמשיך התוכנית בצורה דרדית אחרת, התוכנית תדלג על משפט (או בלוק) זה

3

## לולאות

- בקורס המבוא למדנו על תהליכים איטרטיביים ורקורסיביים

- שניהם נכתבו בתחביר של רקורסיה (האינטרפרטר ממיר רקורסית זנב לאיטרציה)

- בג'אווה איטרציה כותבים במפורש בעזרת משפטים מיוחדים שנקראים לולאות

- ג'אווה תומכת בשלושה סוגים של לולאות: משפט

- while, משפט do ומשפט for

- ג'אווה מאפשרת גם רקורסיה (נראה בהמשך)

6

## ריבוי תנאים (else-if)

```
if (exp1) {
    // code for case when exp1 holds
}
else if (exp2) {
    // when exp1 doesn't hold and exp2 does
}
// more...
else {
    // when exp1, exp2, ... do not hold
}
```

- למבנה else if אין סמנטיקה מיוחדת, נשתמש בו במקרה שמתוך אוסף מקרים אמור להתקיים מקרה אחד לכל היותר

5



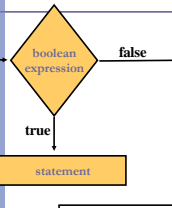
## "אין מדברים בזמן השיעור"

```
int counter = 0;
while (counter < 1000) {
    System.out.println("No talking during class");
    counter++;
}
```

- אף על פי שהדוגמא פשוטה, נמצאים בה כל מרכיבי הלולאה:
  1. הגדרת משתנה עזר ואתחולו
  2. בדיקת תנאי עצירה (שנמשך?)
  3. ביצוע איטרציה נוספת
  4. קידום משתנה העזר

- מוסכמות:
  - משתני עזר מאותחלים ל-0
  - בדיקת הסיום היא בעזרת האופרטור < על מספר האיטרציות המבוקש

## משפט while



```
while ( <boolean_expression> )
  <statement>
```

- ביצוע משפט ה while נעשה כך:
  1. הביטוי <boolean\_expression> מחושב:
    - אם ערכו false מדלגים על <statement> (גוף הלולאה - משפט או בלוק משפטים)
    - אם ערכו true מבצעים את גוף הלולאה וחוזרים ל- (1)

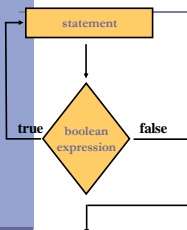
■ לדוגמא: נשתמש בלולאת while כדי להדפיס 1000 פעמים את המחרוזת "אין מדברים בזמן השיעור"

## משפט do

```
public class DoExample
{
    public static void main(String[] args) {
        int counter = 0;
        do {
            System.out.println("No talking during class");
            counter++;
        } while (counter < 1000);
    }
}
```

- הבחירה בין השימוש במשפט do לשימוש במשפט while במקרים שבהם ידוע בוודאות שיהיה לפחות מחזור אחד של הלולאה) היא עניין של טעם אישי

## משפט do



```
do
  <statement>
while ( <boolean_expression> );
```

- כאן התנאי מחושב לאחר ביצוע גוף הלולאה
- לכן הלולאה מתבצעת לפחות פעם אחת
- לפעמים מאפשר לחסוך כתיבת שורה לפני הלולאה

■ נתרם את לולאת ה-while מהשקף הקודם ללולאת do-while

## משפט for

```
for (int counter = 0; counter<1000; counter++) {
    System.out.println("No talking during class");
}
```

- הגדרת משתנה עזר ואתחולו – משתנה זה מוגדר אך ורק בתחום לולאת ה for ואינו נגיש לאחריה
- בדיקת תנאי עצירה (שנמשך?)
- גוף הלולאה - ביצוע איטרציה נוספת
- קידום משתנה העזר



## משפט for

- במשפט ה while ראינו את ארבעת יסודות הלולאה
- אולם תחביר הלולאה כפי שמופיע ב while אינו תומך ישירות בהגדרת משתנה עזר, באתחולו ובקידומו
- המתכנתת צריכה להוסיף קוד זה לפני הלולאה או בתוכה
- תחביר משפט for כולל את ארבעת יסודות הלולאה:
 

```
for (<initialize> ; <boolean-expression> ; <increment> )
  <statement>
```

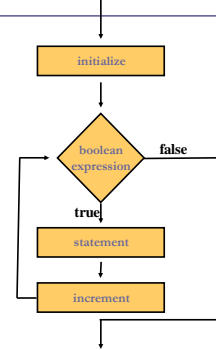
## לולאות מקוננות

- גוף הלולאה יכול להיות לולאה בעצמו
- נדפיס את לוח הכפל:

```
for ( int i = 1; i <= 10; i++ ) {
    for ( int j = 1; j <= 10; j++ )
        System.out.print(i*j + "\t");
    System.out.println();
}
```

14

for (<initialize>; <boolean\_expression>; <increment>)  
<statement>

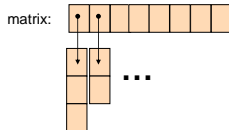


13

## הצהרה

- נסמן מערך בעזרת []
- לדוגמה:

- int[] odds;
- int odds[]; // legal but discouraged
- String[] names;
- int[][] matrix; // an array of arrays

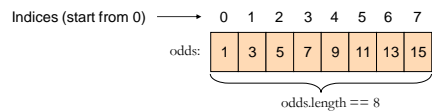


16

## מערכים - תזכורת

- מבנה נתונים בגודל קבוע לשמירת מספר משתנים מאותו טיפוס.

- דוגמא - מערך של מספרים אי-זוגיים:



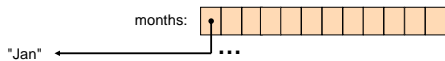
- הטיפוס של כל איבר הוא int
- הערך של האיבר באינדקס 4 הוא 9 `odds[4] == 9`

15

## יצירה ואיתחול

- יצירה ואיתחול של מערך עם מספר קטן וידוע מראש של ערכים.

- int[] odds = {1,3,5,7,9,11,13,15};
- String[] months = {"Jan", "Feb", "Mar", "Apr", "May", "Jun", "July", "Aug", "Sep", "Oct", "Nov", "Dec"};



18

## יצירה וגישה

`Type[] Identifier = new Type[Expression];`

- Examples:

- int[] odds = new int[8];
- float[] vector = new float[j];
- String[] strings = new String[j \* j + 58];

- To access an element of the array

- int val = odds[0];
- float f = vector[vector.length - 1];
- String str = strings[k];

17

## יצירת מערכים

מה הפלט של הקוד הבא:

```
int[] odds = new int[8];
for (int i = 0; i < odds.length; i++) {
    System.out.print(odds[i] + " ");
    odds[i] = 2 * i + 1;
    System.out.print(odds[i] + " ");
}
```

Array creation: all elements get the default value for their type (0 for int)

Output:

0 1 0 3 0 5 0 7 0 9 0 11 0 13 0 15

20

## לולאות ומערכים

- לולאות שימושיות ביותר בעבודה עם מערכים
- להלן קטע קוד שמחשב את סכום אברי המערך arr:

```
double [] arr = {1.0, 2.0, 3.0, 4.0, 5.0};
double sum = 0.0;
for (int i=0; i<arr.length; i++){
    sum += arr[i];
}
System.out.println("arr sum is: " + sum);
```

- הגדרת משתנה עזר שיהיה אינדקס המערך ואתחולו לאפס
- בדיקת תנאי עצירה – האם משתנה העזר עדיין קטן מגודל המערך
- קידום משתנה העזר באחד
- גוף הלולאה - ביצוע פעולה המשתמשת באיבר במקום ה- i

19

## foreach

- ביצוע פעולה מסוימת על כל אברי מערך היא פעולה שכיחה כל כך עד שהחל מ-Java 5 ניתן לה תחביר מיוחד המכונה foreach (או for/in)
- בתחביר זה הקומפיילר מייצר את העבודה עם משתנה העזר בצורה אוטומטית מאחורי הקלעים
- קטעי הקוד הבאים שקולים:

```
for (int i = 0; i < arr.length; i++){
    sum += arr[i];
}
```

שקול ל-

```
for (double d : arr) {
    sum += d;
}
```

קראו זכר "לכל איבר d מטיפוס double שבמערך arr בעצמם..."

21