# מבוא לתיכנות מונחה עצמים – ירושה

### ירושה – למה צריך את זה?

כדי שהתוכנה תשקף את המציאות.

```
למשל: במציאות, מנהל הוא סוג של עובד.
לשניהם יש שם ומשכורת. אבל למנהל יש גם בונוס.
לכן בתוכנה:
```

public class Manager extends Employee {

```
... שדות נוספים ...
מתודות נוספות או מוחלפות ...
```

מושגים: העובד הוא supertype והמנהל הוא

### polymorphism – ריבוי צורה

- מתודות עם אותו שם ומימוש שונה.
- בג'אבה יש שני סוגים של ריבוי-צורה:
  - :Overriding החלפה
- בחירת המתודה מתבצעת לפי סוג העצם.
- . ע"י **המכונה** late binding ע"י **המכונה** 
  - Overloading העמסה
- בחירת המתודה מתבצעת לפי סוג **המשתנה**.
- . ע"י **הקומפיילר** early binding ע"י **הקומפיילר**

#### סוגי שגיאות

שגיאת קומפילציה
שגיאת ריצה
שגיאה לוגית

#### סוגי שגיאות

זולה	שגיאת קומפילציה
בינונית	שגיאת ריצה
יקרה	שגיאה לוגית

#### הגבלת ירושה

- ;בחלקה final אי אפשר להרחיב
  - דוגמה String
  - ;סתודה final אי אפשר להחליף
    - דוגמה getClass.
- מחלקה abstract חייבים להרחיב כל עצם חייב
   להשתייך למחלקה יורשת.
  - מתודה abstract חייבים להחליף כל עצם חייב
     להשתייך למחלקה יורשת שמממשת את המתודה.

#### מתי **לא** משתמשים בירושה?

- Liskov Substitution) עקרון ההחלפה של ליסקוב (Principle):
  - אם מחלקה ב יורשת את מחלקה א, \*
- \* **אז** בכל מקום שבו יש עצם ממחלקה א אפשר לשים במקומו עצם ממחלקה ב והתוכנית תמשיך לעבוד בצורה תקינה.
  - דוגמה נגדית: מחסנית ווקטור.

## Object המחלקה

- . כל מחלקה יורשת מ-Object מאחרי הקלעים.
  - :Object מתודות של
    - toString -
    - getClass -
  - equals, hashCode
    - clone -
  - wait, notify, notifyAll -

## equals - hashCode

a.equals(b)	a.hashCode() =b.hashCode()	
true	true	טוב
false	false	טוב
true	false	רע
false	true	?

### equals - hashCode

a.equals(b)	a.hashCode() =b.hashCode()	
true	true	טוב
false	false	טוב
true	false	רע
false	true	*בסדר

<sup>\*</sup> תקין, אבל עלול לגרום אי-יעילות