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 **subtype**.

ריבוי צורה – polymorphism

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

סוגי שגיאות

שגיאת קומפילציה	
שגיאת ריצה	
שגיאה לוגית	

סוגי שגיאות

שגיאת קומפילציה	זולה
שגיאת ריצה	בינונית
שגיאה לוגית	יקרה

הגבלת ירושה

- מחלקה final – אי אפשר להרחיב;
  - דוגמה – String.
- מתודה 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 שממשת את המתודה.

מתי **לא** משתמשים בירושה?

- עקרון ההחלפה של ליסקוב (Liskov Substitution 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.

המחלקה Object

- כל מחלקה יורשת מ-Object מאחרי הקלעים.
- מתודות של Object:
  - 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*

* תקין, אבל עלול לגרום אי-יעילות