CPSC-224 Software Development

Encapsulation

Yu Wang

wangy2@gonzaga.edu

Feb 5&7, 2025

Announcement

- Homework0 - Career Development 01 – First Resume Draft
  - Deadline: March 3rd
- Homework1 (Check the updates on Canvas)
  - CPSC 224 - GitHub Classroom Assignment Instructions
  - Deadline for submission: Feb. 12
- Homework2 will upload to Canvas on Friday

Daily Attendance (01)

- Scan the QR Code for yourself

Daily Attendance (02)

- Scan the QR Code for yourself

Review - Last Class

- ✓ We reviewed common mistakes in quiz#01
- ✓ We discussed about career development, talked about how to improve the interview, etc.

Encapsulation

- This section we will be looking at the first of the four main principles of object-oriented programming:
  - **Encapsulation**
  - Abstraction
  - Inheritance
  - Polymorphism

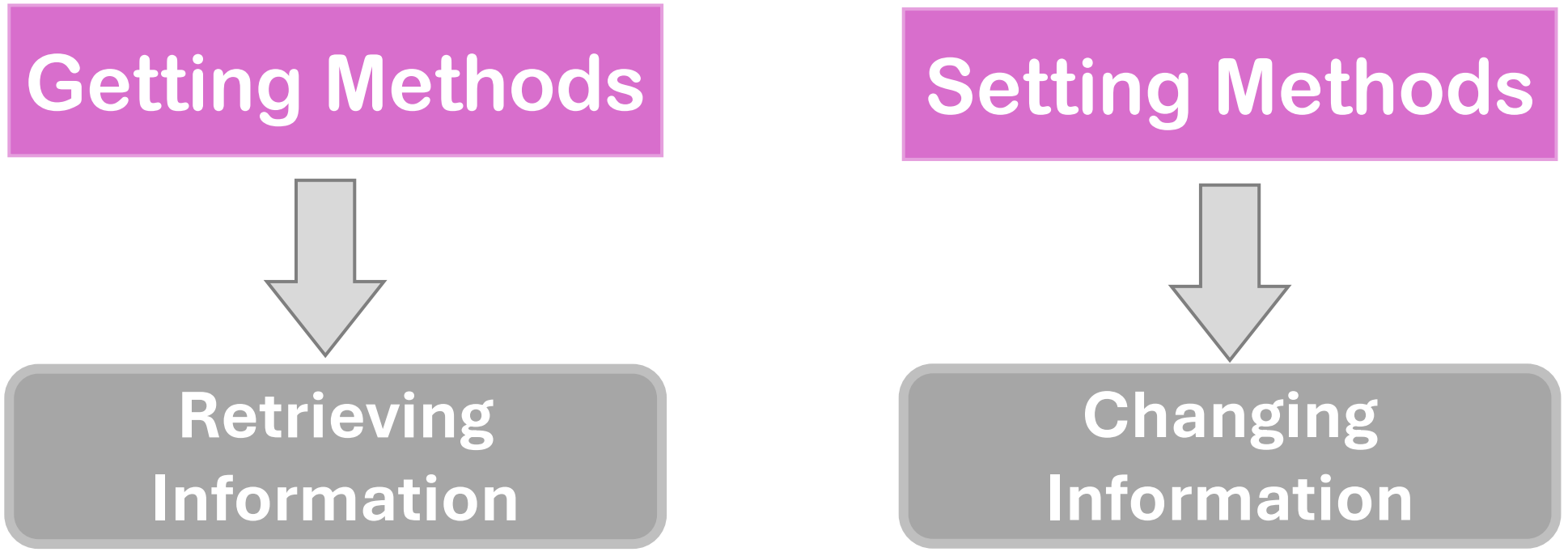
Encapsulation

- Encapsulation refers to **bundling data with methods** that can operate on that data within a class
- Essentially, it is the idea of **hiding data** within a class, **preventing anything outside that class from directly interacting with it**

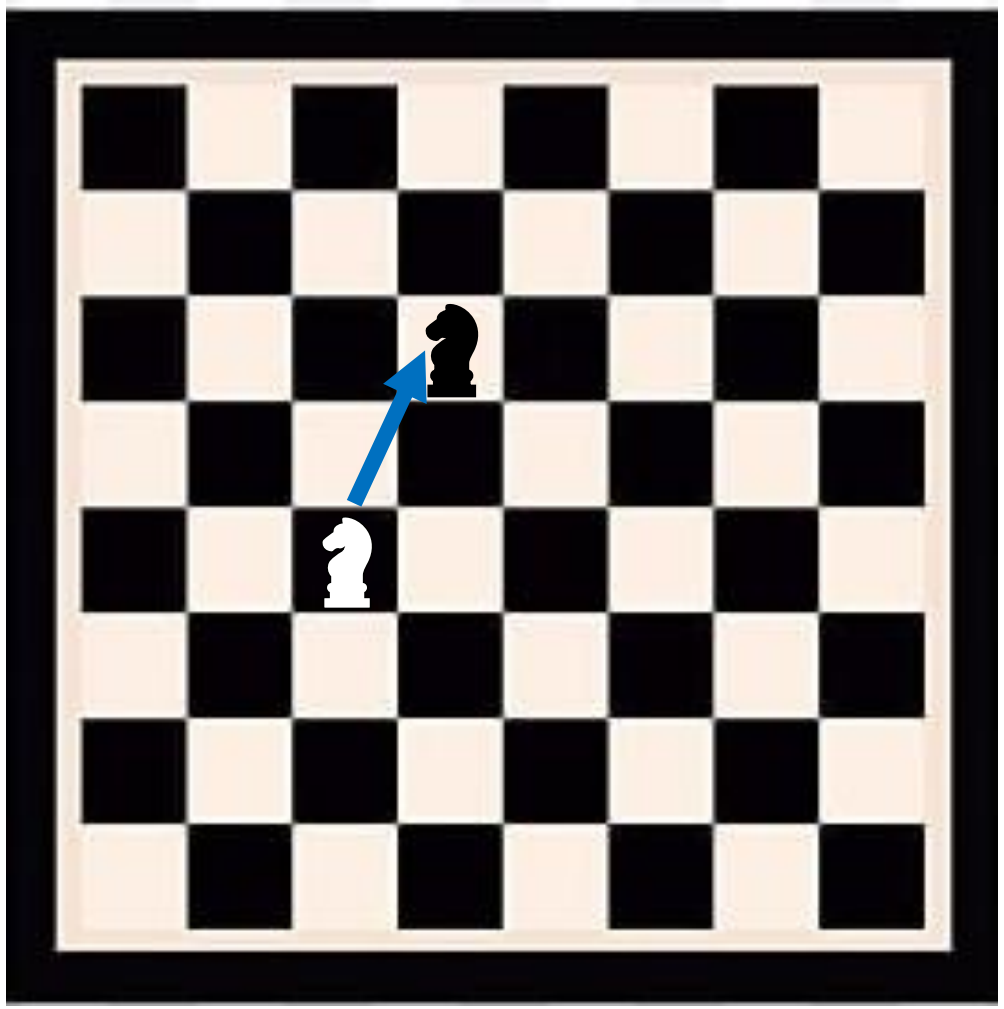
Encapsulation

- This does not mean that members of other classes cannot interact at all with the attributes of another object.
- However, instead, members of other classes looking to interact with the attributes of an object should only be able to do so through that classes methods.
- Remember, methods are the functions defined within the class.

Encapsulation - Methods



Encapsulation – Example (Chess)

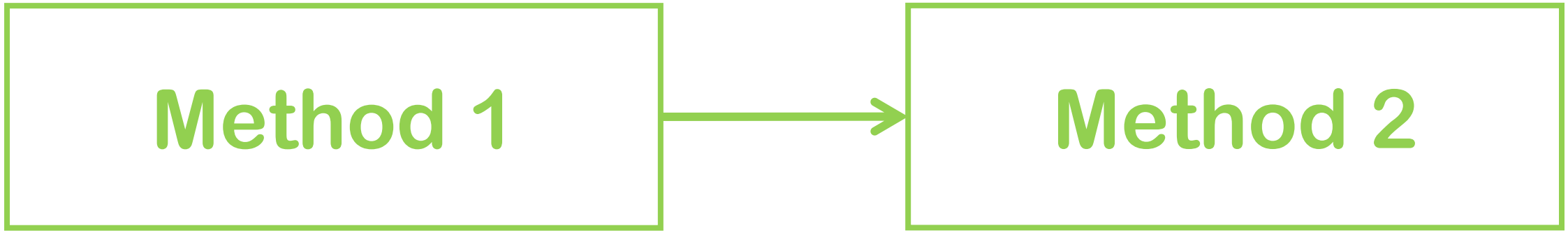


piece.getColor()

Checks the color of any given piece from anywhere in the program

Encapsulation

- Setting methods also allow the programmer to easily keep track of attributes that depend on one another



Encapsulation – Example (Game)

GONZAGA UNIVERSITY

Class Player

Player.maxHealth

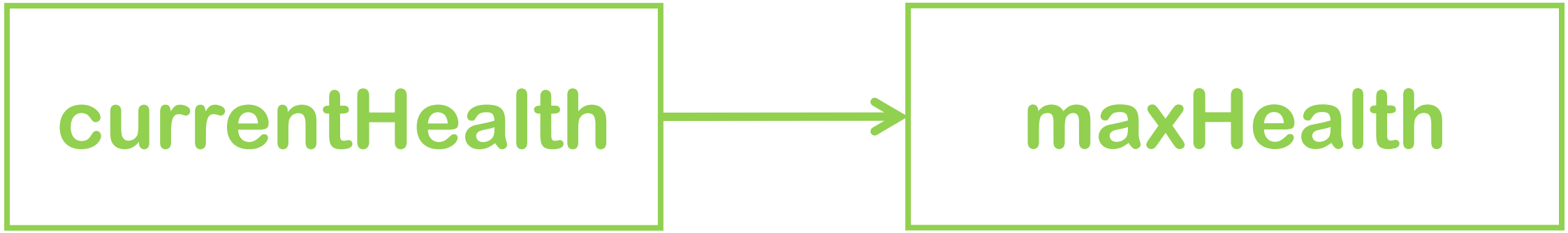
Player.currentHealth

Player.levelUp()

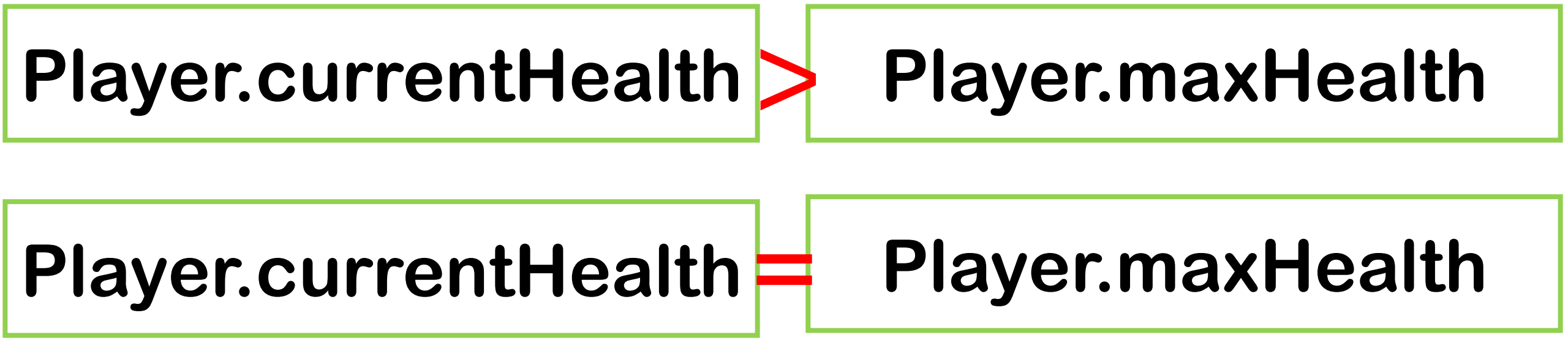
Setting Method

Encapsulation

- The setting method allows both attributes to be changed as they should, rather than requiring you to individually change them.



Encapsulation – Example (Game)



Encapsulation - Methods

- You may also want some attributes to be "read only" from the outside
- To do this, you would define a getter method but not a setter method
- The variable could only be referenced, not changed

Encapsulation – Overview

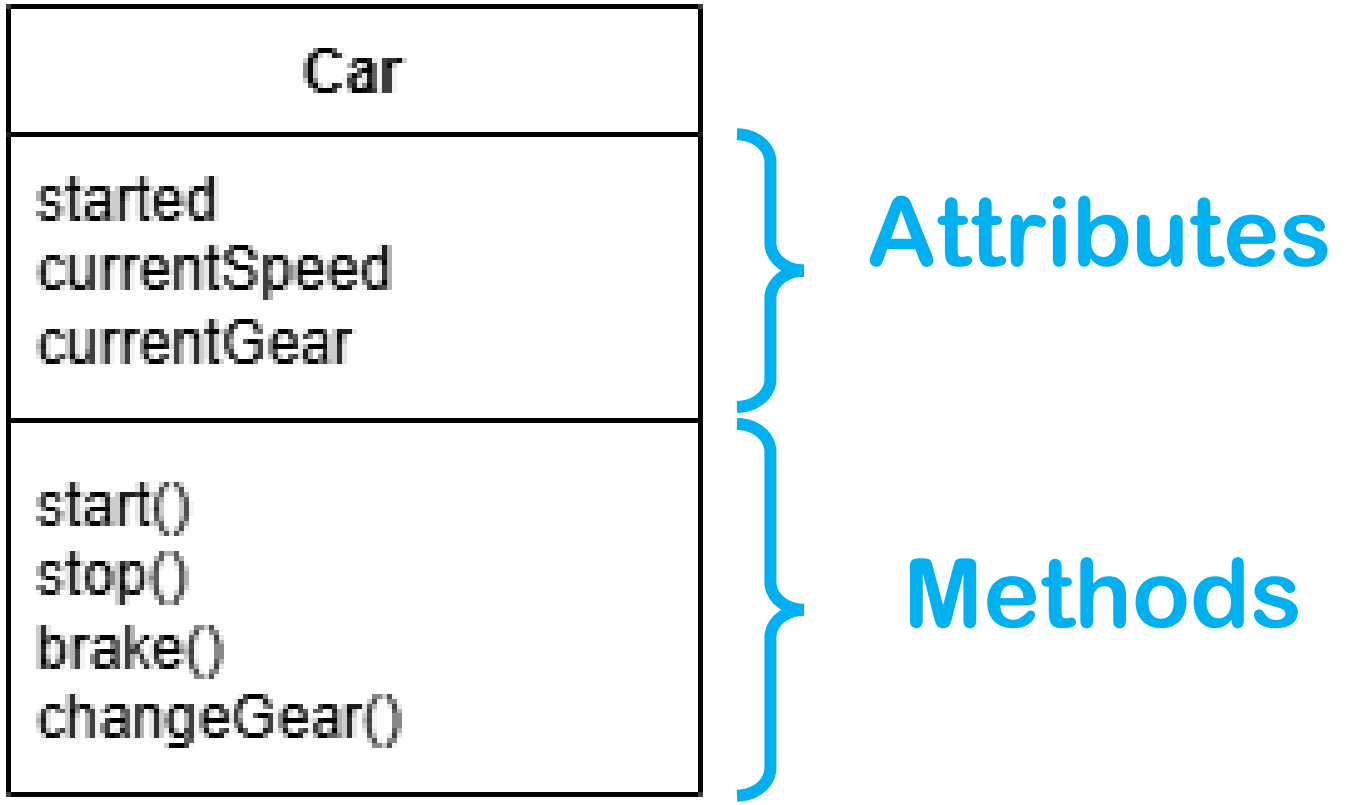
- Encapsulation is a vital principle in Object Oriented Programming.
- Encapsulation:
  - Keeps the programmer in control of access to data
  - Prevents the program from ending up in any strange or unwanted states.

Encapsulation – Overview

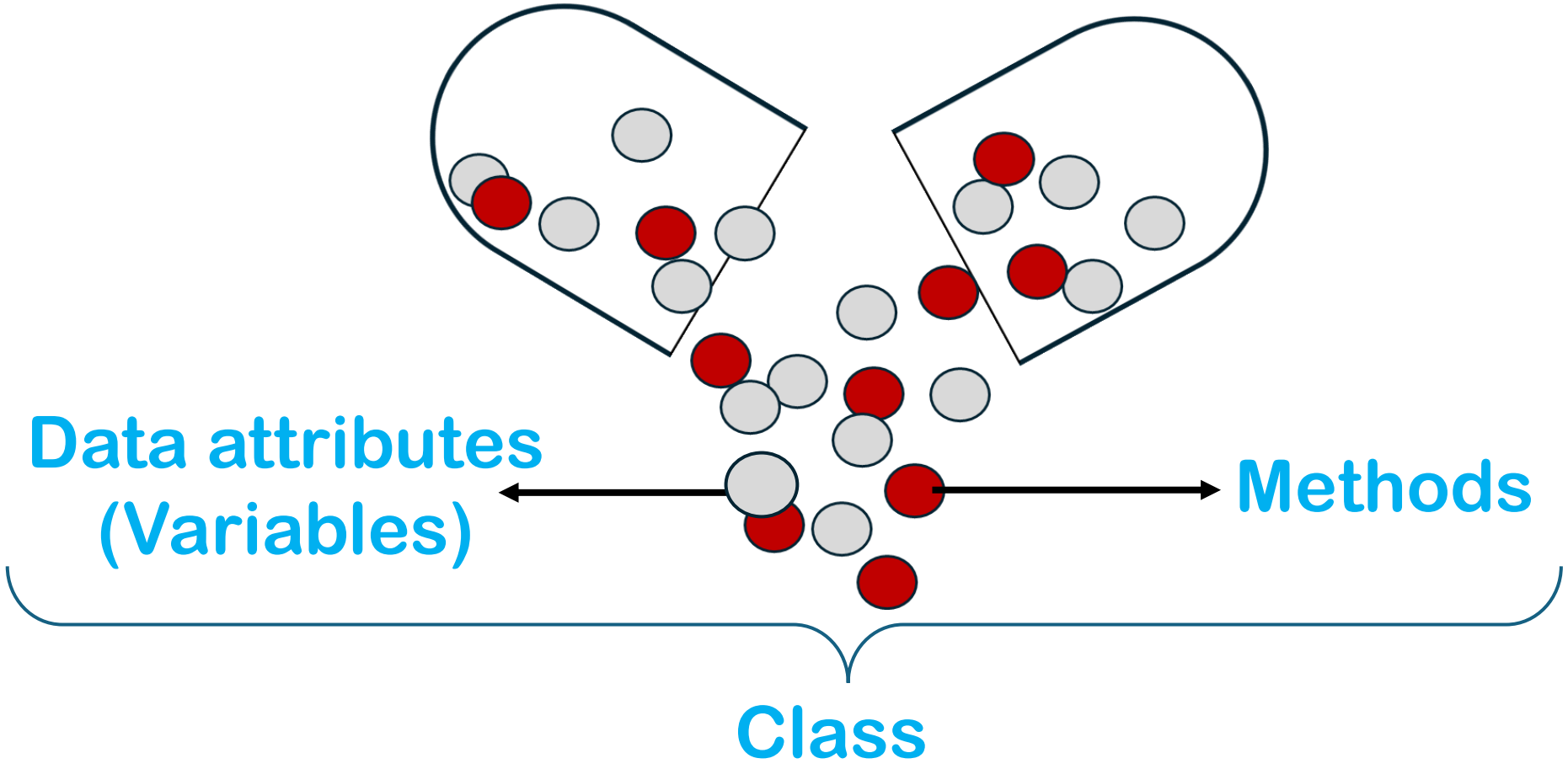
GONZAGA UNIVERSITY

Bundle the data and methods that operate the data in a single unit

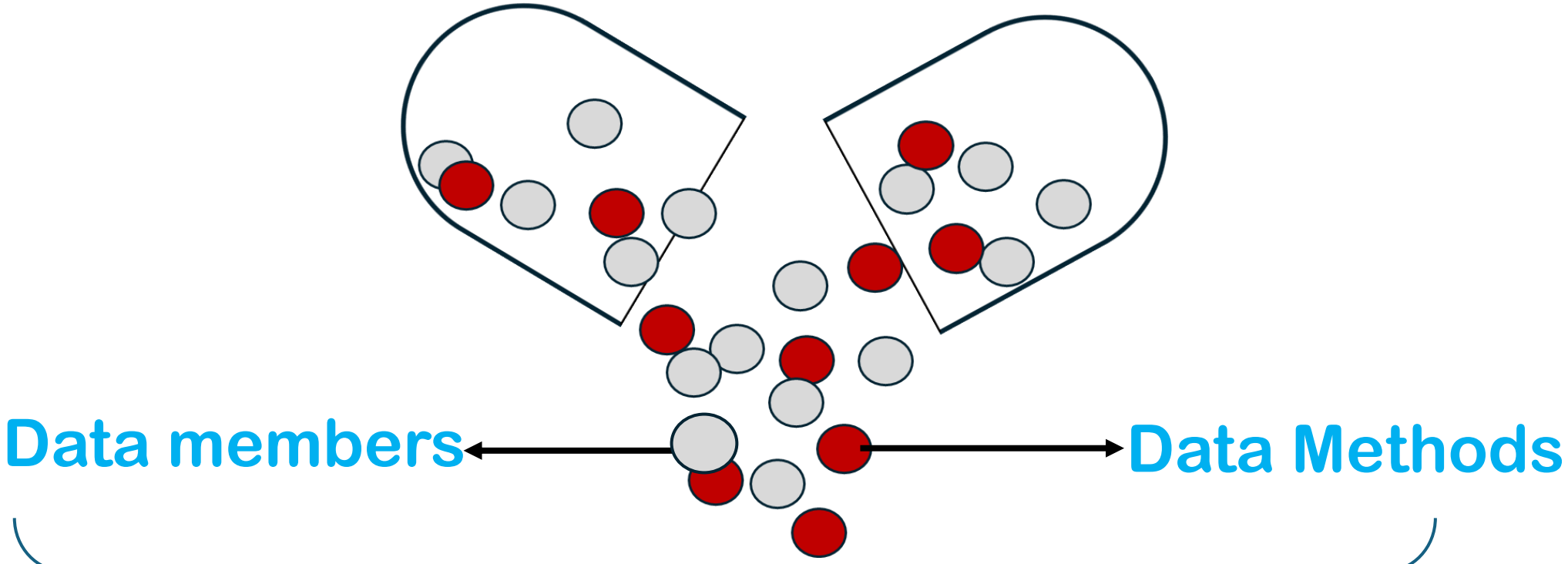
Class and Object



Encapsulation – Overview

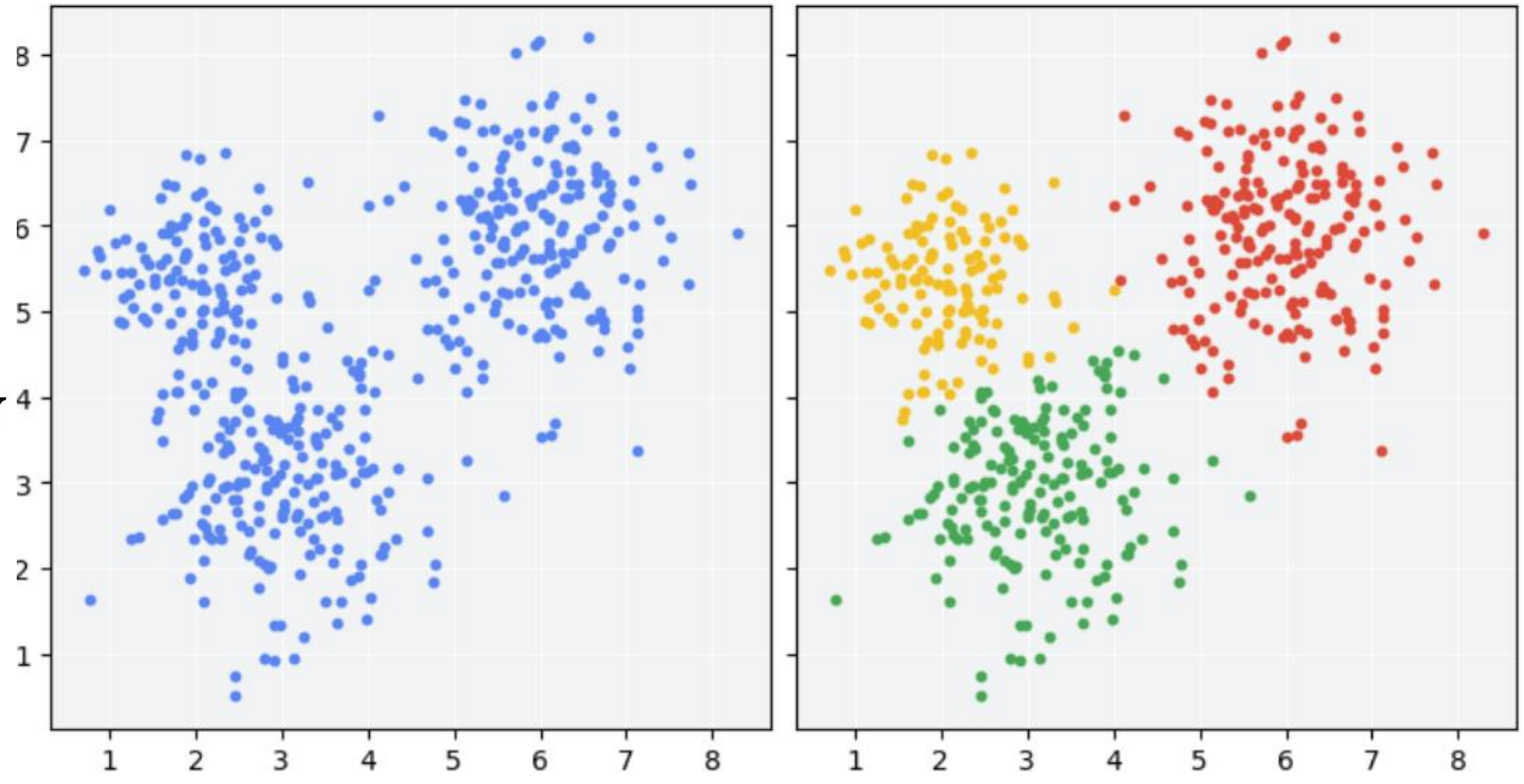


Encapsulation – Overview

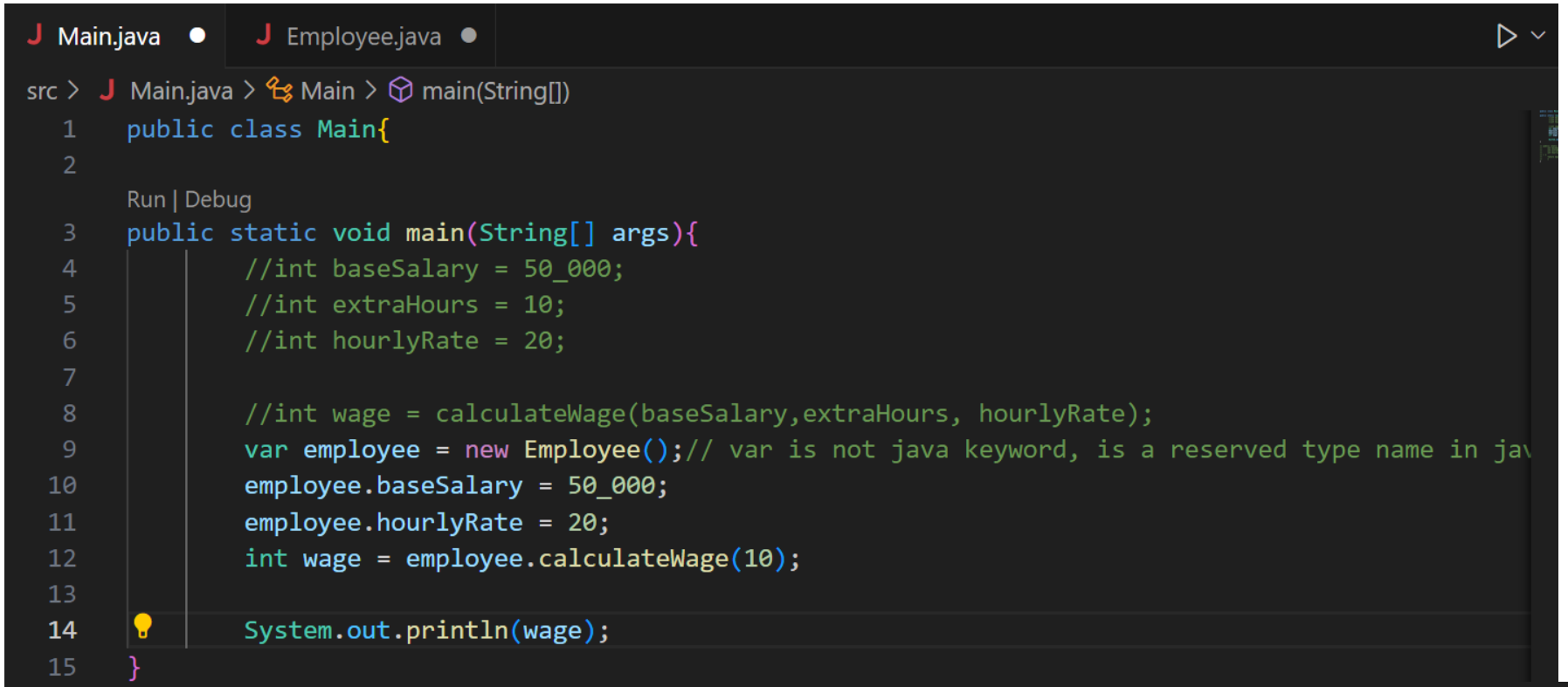


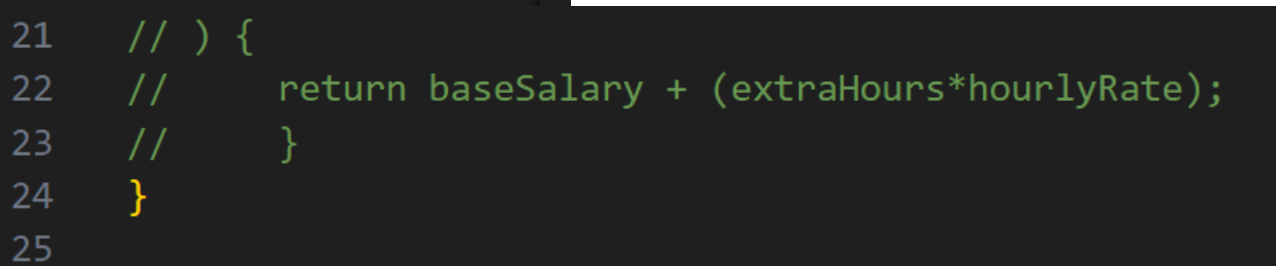
Supplement – What is clustering?

- **Clustering** is an unsupervised machine learning technique designed to group unlabeled examples based on their similarity to each other. (If the examples are labeled, this kind of grouping is called classification.)



Encapsulation – Practice





Encapsulation – Practice

Encapsulation – Getters and Setters

src > Employee.java > Employee

```
1   public class Employee {
7       public int calculateWage(int extraHours){
8           return baseSalary +(hourlyRate*extraHours);
9       }
10
11      // public void setBaseSalary(int baseSalary){
12      //     if (baseSalary <= 0)
13      //     throw new IllegalArgumentException("Salary can not be 0 or less");
14      //     this.baseSalary = baseSalary;
15      // }
16
17      // public int getBaseSalary(){
18      //     return baseSalary;
19      // }
20
21      // public int getHourlyRate() {
22      //     return hourlyRate;
23      // }
24
25      // public void setHourlyRate(int hourlyRate) {
26      // if (hourlyRate <= 0)
27      //     throw new IllegalArgumentException("Hourly Rate can not be 0 or negative");
28      //     this.hourlyRate = hourlyRate;
```

Any Questions?