CSCI 334: Principles of Programming Languages

Lecture 22: Inheritance in OO

Instructor: Dan Barowy

Williams

1

Topics

SCS

OOP Inheritance

How to give a good talk

2

Your to-dos

1. **Mostly working** project, **due Monday 5/13**
2. Want to talk about your project?
   Office hours today 2-3pm.
3. Optional self-scheduled project meetings **next week**. See course website to schedule.

3

Final project timeline

1. ~~Minimally working version (Lab 9), **due Mon 4/29**~~
2. Mostly working version (Lab 10), **due Mon 5/13**
3. Project + video presentation, **due Mon 5/20** (last day of exams)

4

Announcements

Ward Prize Talks
Friday, May 10 @ 2:35pm
Wege Auditorium

Presentations by students nominated for the 2024 Rich Ward Prize for Best Student Project in Computer Science. CS colloquium credit for attendance!

End-of-year celebration to follow in Science Quad

5

Evaluation Forms

(all of these are anonymous)

6

We **listen carefully** to what you say in these forms. Please take your time and write thoughtful responses.

Your feedback is **valuable** to **me**!

7

Purpose of SCS Forms

“[T]he SCS provides instructors with feedback regarding their courses and teaching. The faculty legislation governing the SCS provides that SCS results are made available to the appropriate department chair, the Dean of the Faculty, and at appropriate times, to members of the Committee on Appointments and Promotions (CAP). The results are considered in matters of faculty reappointment, tenure, and promotion.”

—Office of the Provost, Williams College

8

Purpose of “Blue Sheets”

Student comments on the blue sheets […] are solely for [instructor] benefit. They are not made available to department or program chairs, the Dean of the Faculty, or the CAP for evaluation purposes.

—Office of the Provost, Williams College

9

<u>Blue sheet prompts:</u>

* What **course topic** did you **enjoy the most**?

* What **course topic** did you **least enjoy**? Do you think that it was valuable to learn anyway?

* Please comment on the following:
  * Did you find **weekly quizzes** helpful or not helpful?
  * Would you like more **mentoring** or less mentoring?
  * Would you prefer TA hours or a **dedicated lab meeting**?

* Did you **look forward to coming to class**?

10

Optional Quiz

11

“Object-oriented programming is a **solution** to **complexity**”

12

Dynamic Dispatch

- Suppose we have:

```
class Number {
  int value;

  public Number(int v) {
    value = v;
  }

  public int getValue() {
    return value;
  }

  public String squee() {
    return "squee!";
  }
}
```

13

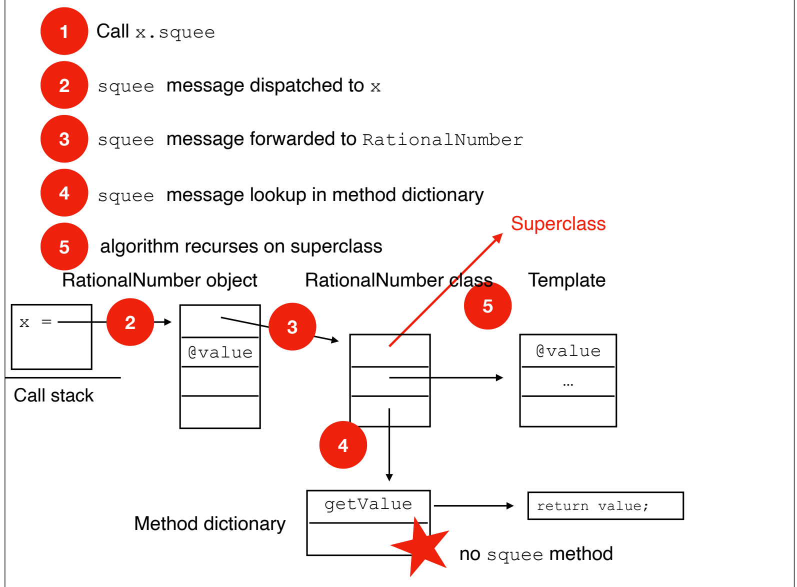
Dynamic Dispatch

- And we add

```
class RationalNumber extends Number {
    protected double numerator;
    protected double denominator;

    public RationalNumber(double n, double d) { … }

    public double getValue() { … }

    public String toString() { … }
}
```

- Observe: no `squee` method.
- Does `RationalNumber` have a `squee` method?

14



15

Ingalls Test for Extensibility

i.e., the "rectangle test"

- The test is about the ability to extend software *after* it has already been designed and written.

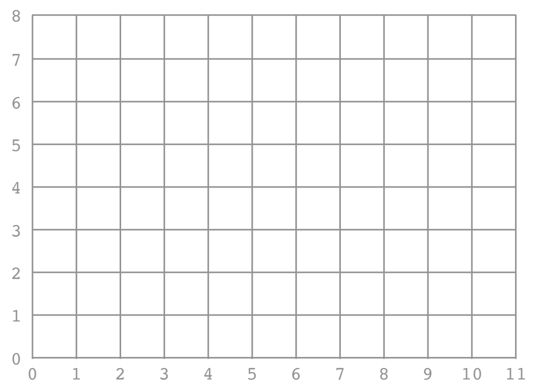
16

Ingalls Test for Extensibility

- The test is about the ability to **extend** software **after** it has already been written.
- E.g., suppose you have a class for a `ColoredRectangle`.
- Can you define a new kind of number (e.g., fractions), use your new numbers to redefine (subtype) rectangle, and then ask the system to color the rectangle?
- If so, you have an OO system.
- You can make the change only by **adding new code**.

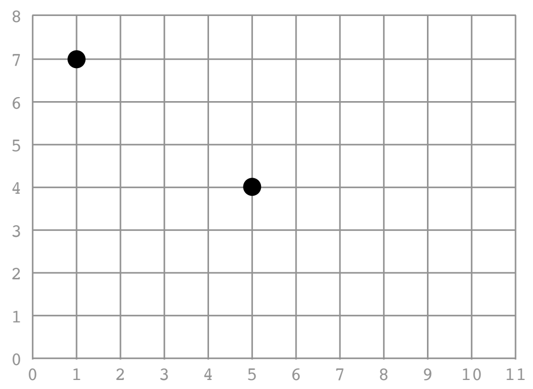
17

Ingalls Test for Extensibility



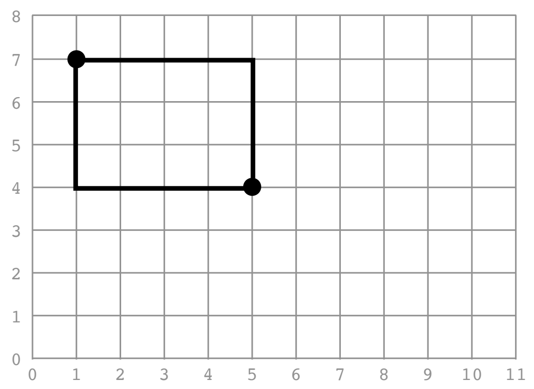
18

Ingalls Test for Extensibility



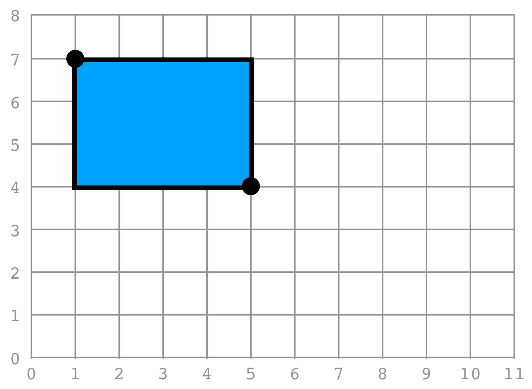
19

Ingalls Test for Extensibility



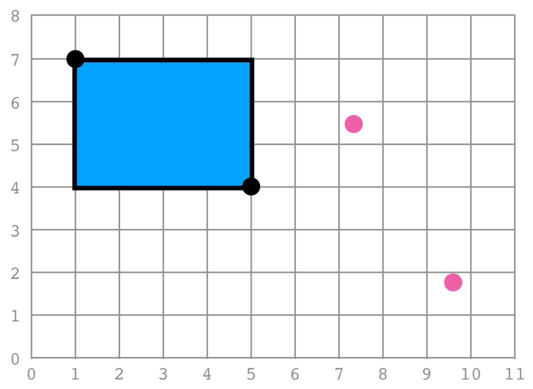
20

Ingalls Test for Extensibility



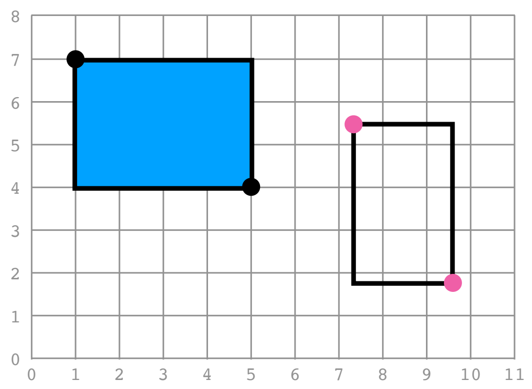
21

Ingalls Test for Extensibility



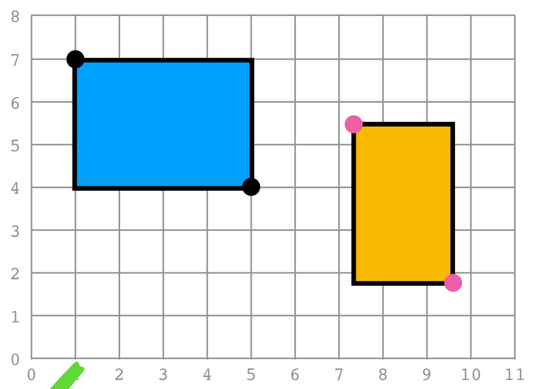
22

Ingalls Test for Extensibility



23

Ingalls Test for Extensibility



✓ yes, it’s object-oriented!

24

OO vs Functional Tradeoff

- OO offers a different kind of extensibility than functional (or function-oriented) languages.
- Suppose you're modeling a hospital.

Operation	Doctor	Nurse	Orderly
`Print`	`Print Doctor`	`Print Nurse`	`Print Orderly`
`Pay`	`Pay Doctor`	`Pay Nurse`	`Pay Orderly`

- FP makes it easy to add operations (rows).
- OOP makes it easy to add data (columns).

25

CSCI 334: Principles of Programming Languages

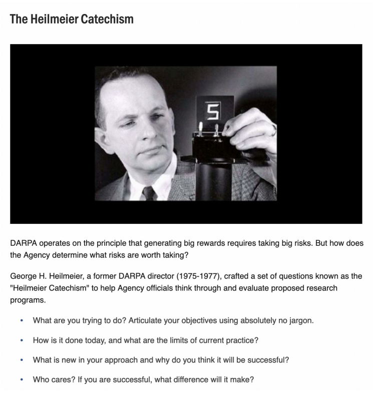
Lecture 22-2: How to give a **good** talk

Instructor: Dan Barowy

Williams

26

Video presentation



The Heilmeier Catechism

DARPA operates on the principle that generating big rewards requires taking big risks. But how does the Agency determine what risks are worth taking?

George H. Heilmeier, a former DARPA director (1975-1977), crafted a set of questions known as the "Heilmeier Catechism" to help Agency officials think through and evaluate proposed research programs.

- What are you trying to do? Articulate your objectives using absolutely no jargon.
- How is it done today, and what are the limits of current practice?
- What is new in your approach and why do you think it will be successful?
- Who cares? If you are successful, what difference will it make?

https://www.darpa.mil/work-with-us/heilmeier-catechism

27

How to give a good talk

Five tips

28

One: Have a story

29

Two: Don't "bury the lede"

THE EXCITING PARTS
"We forgot to tell everyone."
2023-2023

30

Three: Don't make your audience read

31

Four: Show by example

32

Five: Stay on script

33

Six (oops!): Finish on time

34

Recap & Next Class

This lecture:

OOP Inheritance

How to give a good talk

Next lecture:

No next lecture!

35