Lesson 01: Course Introduction (W01D1)

Balboa High School

Michael Ferraro

<mferraro@balstaff.org>

Do Now #1

- Begin by turning on your workstation and starting **Ubuntu Linux** — not Windows!
- Using Chrome or Firefox, open Lesson #1: `http://feromax.com/apcs/lessons/`

proceed to next page →

Do Now #2

- Create a user account at GitHub
  - username **must** be
    - in this format: `lastname-firstname`
    - lowercase — e.g., `Ferraro-Michael` is NOT OK!
  - create a free account using this link
  - username taken? add a digit (e.g., `ferraro-michael4`)
- Complete the form here
- Visit this registration link to create an account on the course website ← **use same username as GitHub**

Aim

Students will. . .

- be welcomed to the course
- read the course syllabus
- complete their first graded exercise
- register on the course website
- complete an online information form
- obtain accounts on `apcs02` (our server) and GitHub

Definition of Computer Science

"Computer science (CS) is the study of computers
and algorithmic processes, including
their principles,
their hardware and software designs,
their applications, and
their impact on society."

From *A Model Curriculum for K-12 Computer Science: Final Report of the ACM K-12 Task Force Curriculum Committee*, Tucker et al, October 2003

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language

  Let's see an example of a process/algorithm:

  - download the Scratch project here

    (right-click, save target/link as. . . , save to Desktop)

  - start Scratch, open saved project file

    press `F12`, type "`scratch &`"

  - things to try:
    - click the green flag (top right)
    - edit→start single stepping→flash blocks (slow)
  - can you explain how this program works?
  - changing the code, changing the output, and getting the same output. . .

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language

Now the same example in Java:

```
int x = 1;

while( x <= 15 ) {

    if( x % 3 == 0 ) {
        System.out.println(x);
    }

    x = x + 1;

}
```

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language
- Writing programs using Java

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language
- Writing programs using Java
  - Java has a written syntax (c.f. the blocks in Scratch)

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language
- Writing programs using Java
  - Java has a written syntax (c.f. the blocks in Scratch)
  - Java is a richer programming language than Scratch and can be used to write very complex and large programs

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language
- Writing programs using Java
  - Java has a written syntax (c.f. the blocks in Scratch)
  - Java is a richer programming language than Scratch and can be used to write very complex and large programs
- Analyzing what given Java code will do when run on a computer

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language
- Writing programs using Java
  - Java has a written syntax (c.f. the blocks in Scratch)
  - Java is a richer programming language than Scratch and can be used to write very complex and large programs
- Analyzing what given Java code will do when run on a computer
- Debugging code so that it behaves as intended

What We Learn in APCS

- Designing processes/algorithms using building blocks provided by a programming language
- Writing programs using Java
  - Java has a written syntax (c.f. the blocks in Scratch)
  - Java is a richer programming language than Scratch and can be used to write very complex and large programs
- Analyzing what given Java code will do when run on a computer
- Debugging code so that it behaves as intended
- Working in teams to tackle a complex technical project (Mine Sweeper!)

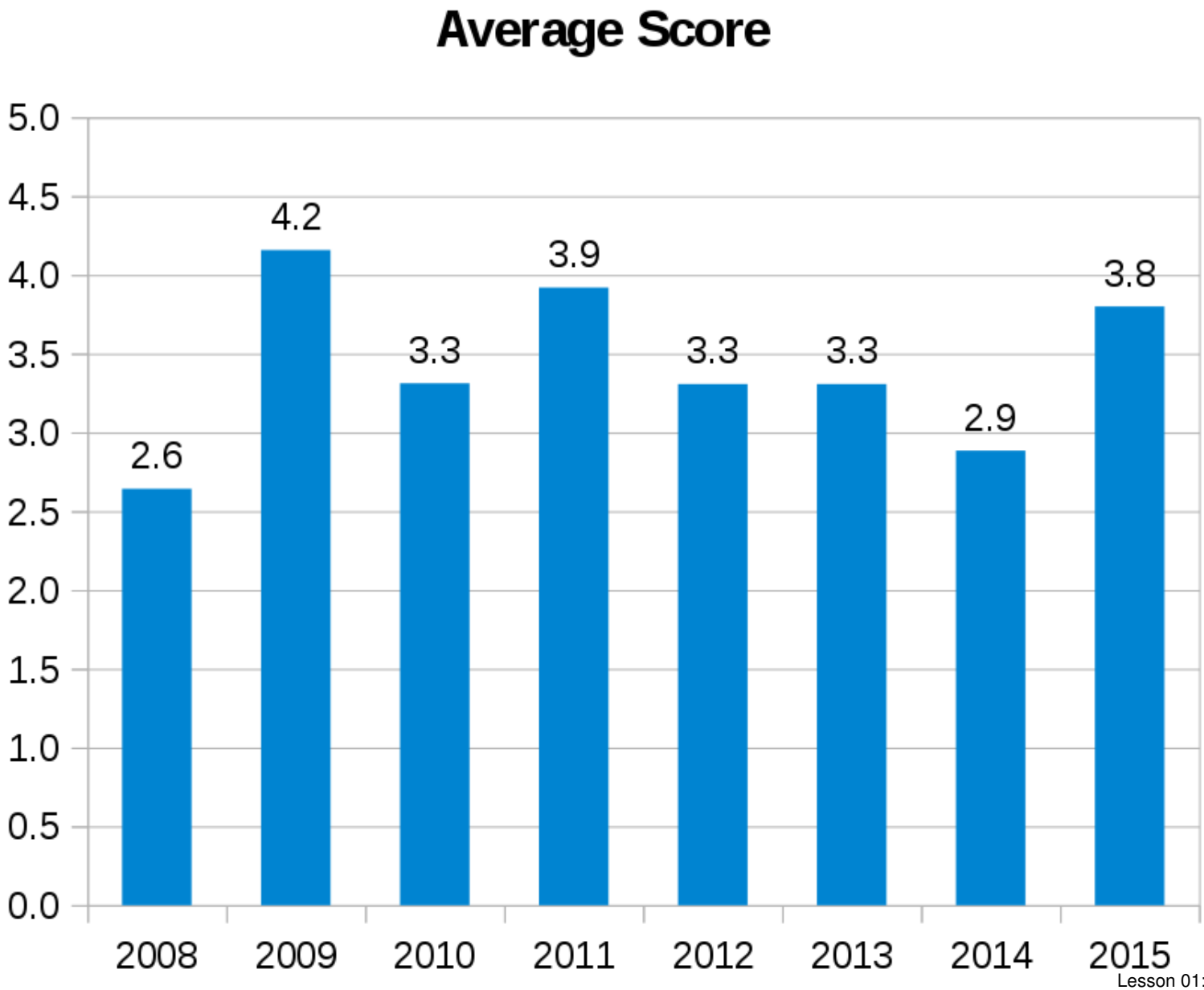
Who am I?

- Degree in EECS
- Industry experience
  - systems administrator (IT)
  - hosted web applications
  - corporate IT consultant
- Math teacher

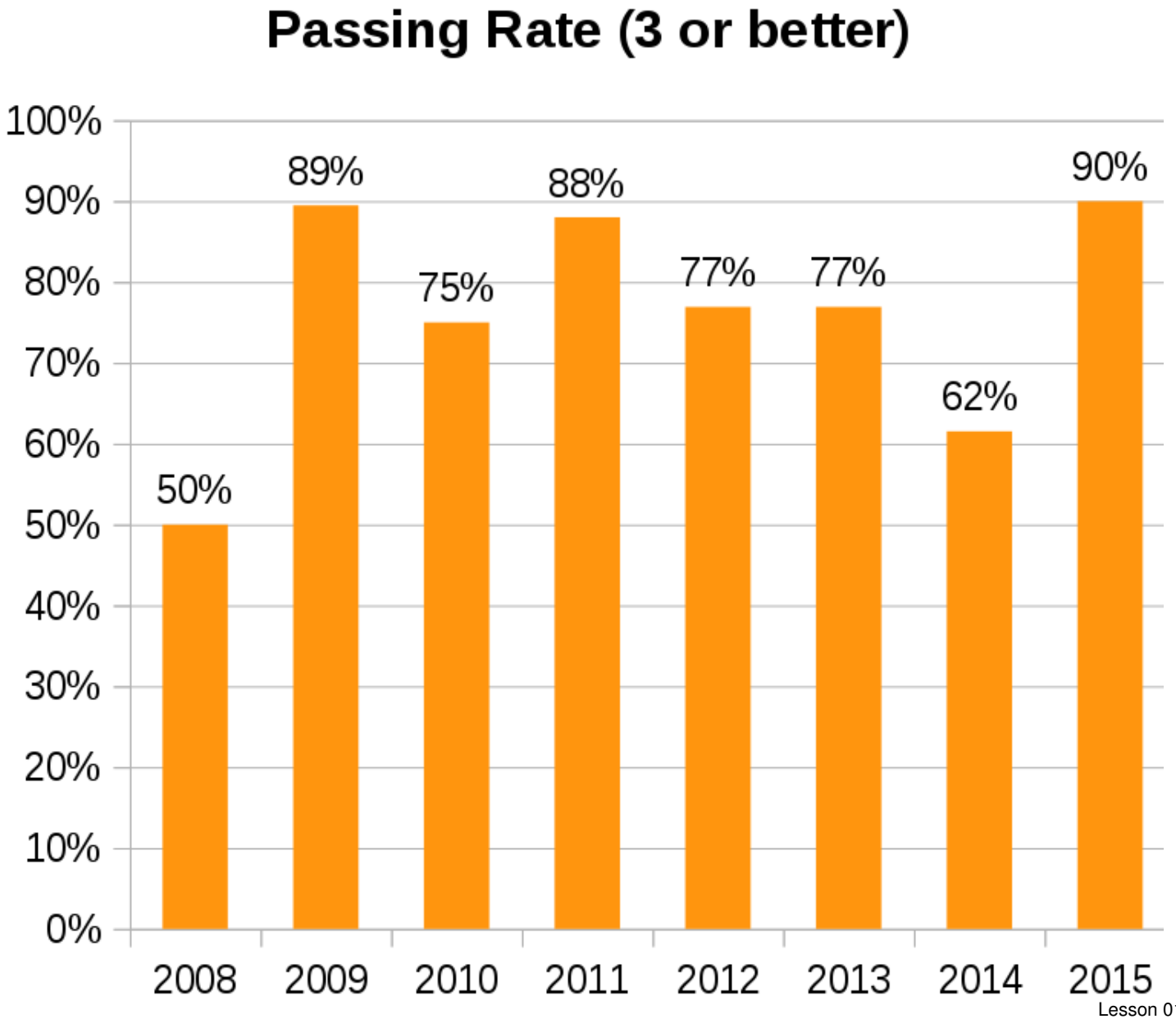
Computer Lab: Room 124

- 35 dual-boot PC workstations
  - Windows 7 for GameDev/AoIT classes
  - Ubuntu Linux for APCS
- Be good lab citizens
  - keep work area clean
  - log out (5th) or shut down (6th) at the end of class
  - report issues like broken peripherals

Those Who Came Before…



Those Who Came Before…



First Assignment

While I'm setting up your accounts, you are to read over the first assignment and complete it, finishing it for HW.

- Read over the directions and questions here.
- Find the syllabus on the course website and begin reading it.
- Afterward, once I've provisioned your server account, you'll be resetting your password and mounting your new "locker" folder

Reset `apcs02` Password, Mount Locker

- Press F12 to start a terminal shell
- Type this command to reset your `apcs02` account password:

  `smbpasswd -r apcs02 -U your_username`

  - your current password: last 4 digits of phone number
  - new password: your choice and don't forget it!
- Use *mount locker* shortcut on desktop

HW

The homework will always be listed on the last slide of the lesson.[a] Also keep an eye out for emails that give additional details, hints, or corrections (errata!).

HW: Finish syllabus questions, due next class

[a] There is a link to the lesson slides on the right side of the course site's homepage at http://feromax.com/.