As you arrive:

1. Start up your computer and plug it in.
2. ***Log into Angel*** and go to CSSE 120.
   Do the ***Attendance Widget*** –
   the PIN is on the board.
3. Go to the ***Course Schedule*** web page.
   Open the ***Slides*** for today if you wish.

The quiz lists its URL, then BOOKMARK it.

Today:

Introduction to the course, Eclipse and Python

Introduction to course:

- Introduce yourself
- Course web site, resources
- Course background
  - What is software development?

Introduction to Eclipse & Python

- **Eclipse** – our Integrated Development Environment
- **Python** – our programming language

Outline of today's session

- Introductions: students, assistants and instructor
- Resources:
  - Course web site, CSSE lab hours, csse120-staff email
- Course background:
  - What is computer science? Software development?
- Hands-on introduction to Eclipse and Python
  - Eclipse – our Integrated Development Environment (IDE)
    - Including Subversion – our version control system, for turning in work
  - Python – our first programming language
    - A whirl-wind tour, plus your first Python program
    - Including a little *zellegraphics*

Robots starting at the *next* session

Roll Call & Introductions

- Name (nickname)
- Hometown
- Where you live on (or off) campus
- Something about you that most people in the room don't know

This means you should be answering Question #1 on the quiz.

Q1

Resources

- **Course web site**:

  `www.rose-hulman.edu/class/csse/csse120`

  then **`Robotics (Fisher and Mutchler)`**

- **Course schedule** – find it now (from course web site)
  - Slides
  - Topics and Reading
  - Homework
- **CSSE lab assistants in:**

```
Moench F-217
7 to 9 p.m.
Sundays thru Thursdays
```

- **Email to:** `csse120-staff@rose-hulman.edu`

Q2-4

What is Computer Science (CS)?

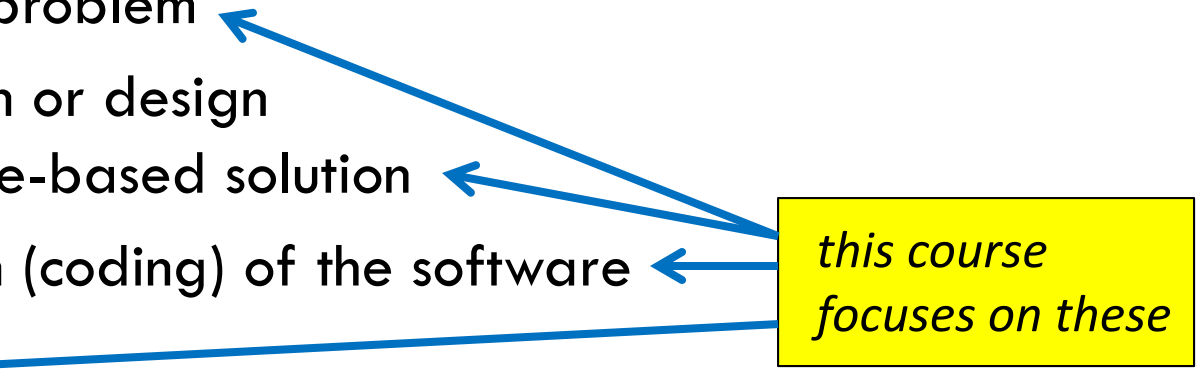
- The work of computer scientists falls into three broad categories:
  - ***designing and building software;*** ← *this course focuses on this*
  - developing effective ways to ***solve computing problems***, such as:
    - storing information in databases,
    - sending data over networks or
    - providing new approaches to security problems; and
  - devising new and better ways of ***using computers*** and addressing particular ***challenges in areas*** such as
    - robotics,
    - computer vision, or
    - digital forensics.

from the Association for Computing Machinery (ACM)

Q5-6

What is software development?

- Software development includes:
  - Market research
  - Gathering requirements for the proposed business solution
  - Analyzing the problem
  - Devising a plan or design for the software-based solution
  - Implementation (coding) of the software
  - Bug fixing
  - Testing the software
  - Maintenance



from Wikipedia, Software Development

What is a *program*? A *programming language*?

- ***Program***
  - Detailed set of instructions
  - Step by step
  - Meant to be executed by a computer
- A ***programming language*** specifies the:
  - ***Syntax*** (form), and
  - ***Semantics*** (meaning)

  of legal statements in the language

There are thousands of computer languages. We use ***Python*** because it:

- Is ***powerful***: strong programming primitives and a huge set of libraries.
- Has a ***gentle learning curve***; you will start using it today!
- ***Plays well with others*** (e.g. COM, .NET, CORBA, Java, C) and ***runs everywhere***.
- Is ***open source***.

- See Wikipedia's History of Programming Languages for a timeline of programming languages.
- Python was introduced in 1991.
- Its predecessors include ABC, Algol 68, Icon and Modula-3.

Q7

Human Languages vs. Programming Languages

- Ambiguous *vs.* very precise
- Syntax (form) must exactly match …
  - CaSe MAtTerS
- Semantics (meaning)
- Translation
  - From a high-level language (Maple, Java, Python, C)
  - To a low-level language (machine language)

A *compiler* or *interpreter* does this translation. A *loader* puts the translated program into memory, and the *CPU* (Central Processing Unit) runs the program under the direction of the *Operating System*.

Q8

Integrated Development Environments (IDEs) – Outline

- What are they?
- Why use one?
- Our IDE – Eclipse
  - Why we chose it
  - Basic concepts in Eclipse
    - Workspace, Workbench
    - Files, folders, projects
    - Views, editors, perspectives

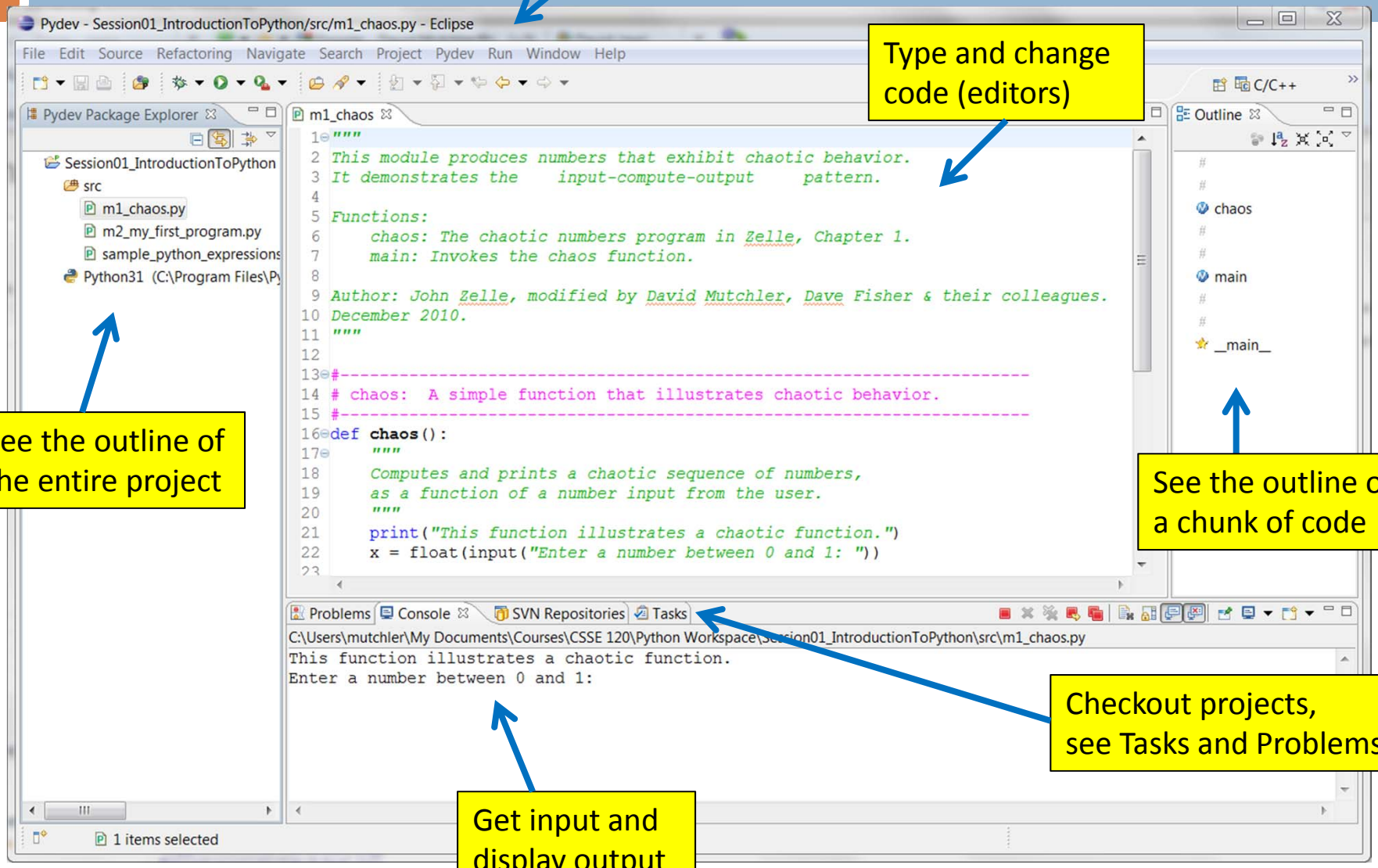
The next slides address these points about IDEs.

IDEs – What are they?

An IDE is an application that makes it easier to develop software.

They try to make it easy to:

Compile, run, debug, document, and more



Type and change code (editors)

See the outline of the entire project

See the outline of a chunk of code

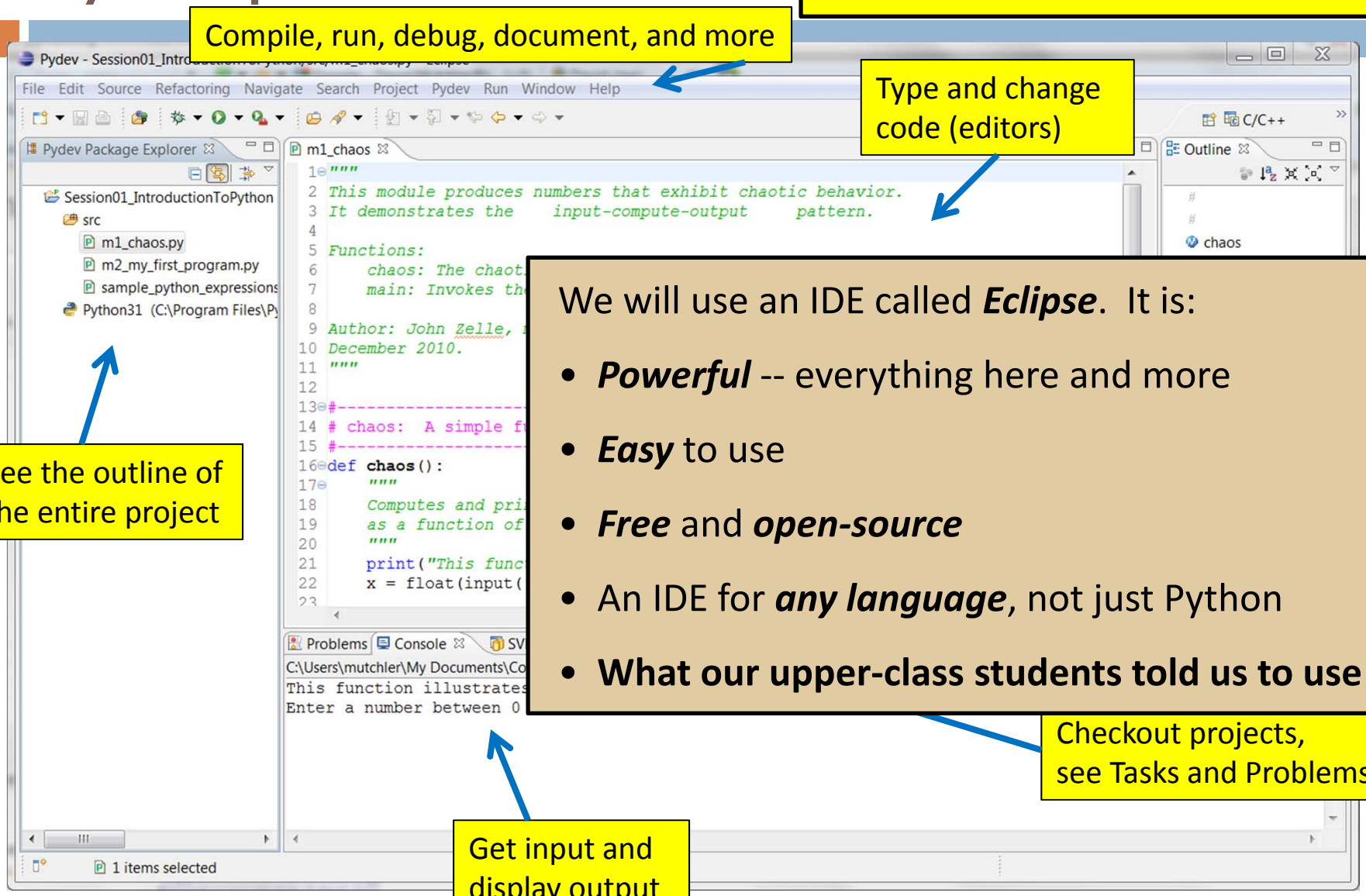
Checkout projects, see Tasks and Problems

Get input and display output

IDEs – Why use one? Why Eclipse?

An IDE is an application that makes it easier to develop software.

They try to make it easy to:

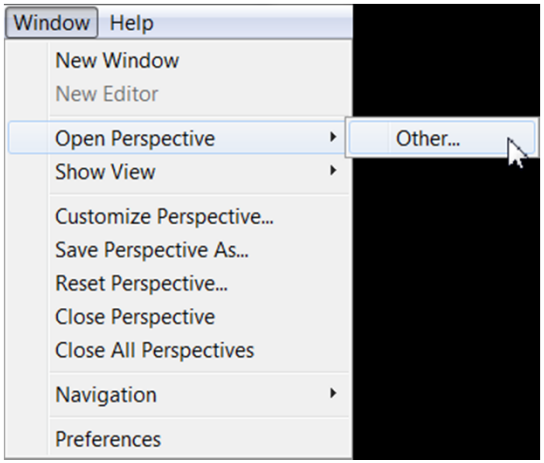


We will use an IDE called ***Eclipse***. It is:

- ***Powerful*** -- everything here and more
- ***Easy*** to use
- ***Free*** and ***open-source***
- An IDE for ***any language***, not just Python
- **What our upper-class students told us to use!**

Open Eclipse

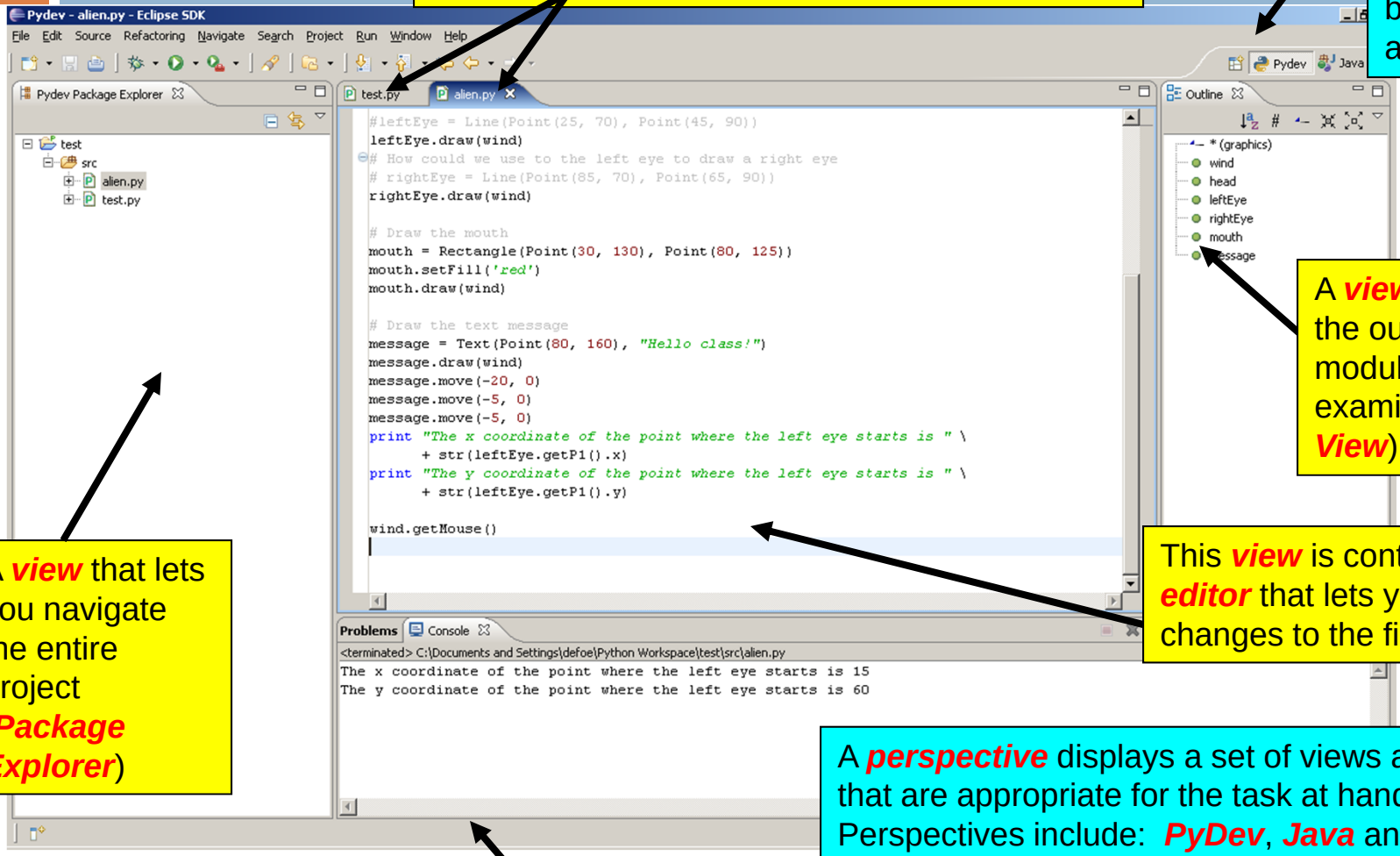
- Your instructor will show you're the highlights.
  - They are summarized on the next several slides.



Basic concepts in Eclipse

- ***Workspace*** – where your ***projects*** are stored on your computer
- ***Project*** – a collection of files, organized in folders, that includes:
  - ***Source code*** (the code that you write)
  - ***Compiled code*** (what your source code is translated into, for the machine to run)
  - ***Design documents***
  - ***Documentation***
  - ***Tests***
    - And more that you will learn about over time
- ***Workbench*** – what we saw on the previous slide, that is, the tool in which you do your software development

Views, editors, perspectives



This is the ***PyDev perspective*** but just a button click brings us to another

Tabbed ***views*** of the source code of this project

A ***view*** that shows the outline of the module being examined (***Outline View***)

A ***view*** that lets you navigate the entire project (***Package Explorer***)

This ***view*** is controlled by an ***editor*** that lets you make changes to the file

A ***perspective*** displays a set of views and editors that are appropriate for the task at hand. Perspectives include: ***PyDev***, ***Java*** and lots more

Tabbed ***views*** (***Problems***, ***Console***)

Eclipse in a Nutshell

- ***Workspace*** – where your ***projects*** are stored on your computer
- ***Project*** – a collection of files, organized in folders, that includes:
  - ***Source code*** and ***Compiled code*** and more
- ***Workbench*** – the tool in which to work
  - It has ***perspectives*** which organize the ***views*** and ***editors*** that you use
- ***View*** – a "window within the window"
  - displays code, output, project contents, debugging info, etc.

Introduction to Python

- We will see a quick view of Python programming today, but we will examine all of today's ideas in more detail in forthcoming sessions.
- Follow me as I demonstrate how to program in a Python Shell:
  - ***Follow me. You'll get a summary and transcript later.***
- ***Get an assistant to help if you have any troubles during ANY of this "live coding" session.***

Key ideas from live coding session:

evaluation in the interpreter, variables (case matters!), assignment

- In the interactive Python shell (at the >>> prompt), try:
  - `3 + 4`
  - `3 + 4 * 2`

The interpreter evaluates the expression that it is given and shows the result. Note the use of "precedence".

  - `width = 4`
  - `height = 5`
  - `width`
  - `width, height`

Assignment: read it as "width GETS 4"

  - `width = width + 2`
  - `width`
  - `Width`

Terrible mathematics, but common programming paradigm: increment width by 2

Case matters. Try to decipher the error message.

Key ideas from live coding session:
importing modules

- In the interactive Python shell (at the >>> prompt), try:
  - `abs(-7)`
  - `sin(pi/3)`

    You'll get an error message from the above

Some functions are built-in.

Some aren't. Importing module X lets you use **`X.name`** *to refer to things defined in module X*

  - `import math`
  - `math.sin(math.pi / 3)`

There are other ways to do import, but they should be used with caution.

Key ideas from live coding session:

strings and comments

- In the interactive Python shell (at the >>> prompt), try:
  - `"hello"`
  - `'hello'`
  - `width + height`
  - `"width" + "height"`
  - `"width" * height`
  - `"width" * "height"`
  - `# This is a comment.`
  - `# It is ignored by the interpreter,`
  - `# but is important help to human readers.`

Double-quotes …

… are the same in Python as single-quotes (not typical of other languages)

Do you see the difference between variable names and string constants?

This one is cool! Can you guess what will happen? Note that **`height`** *is NOT in quotes.*

The same thing with **`height`** *is quotes yields an error. Do you see why?*

Key ideas from live coding session:

Loops! and *range!*

- Back in the interpreter (at the >>> prompt), try:
  - `list(range(12))`
  - `list(range(2, 12))`
  - `list(range(2, 12, 3))`

Note that this yields 0 to 11 (not 12)

Note the colon and subsequent indentation

```
for k in range(6):
    print(k, k * k)
```

Your turn: Write a ***for*** *loop that prints:*

```
0, 8
1, 7
2, 6
3, 5
4, 4
5, 3
6, 2
7, 1
```

Software Engineering Tools

- The computer is a powerful tool
- We can use it to make software development easier and less error prone!
- Some software engineering tools:
  - IDEs, like Eclipse and IDLE
  - Version Control Systems, like Subversion
  - Testing frameworks, like JUnit
  - Diagramming applications, like UMLet, Violet and Visio
  - Modeling languages, like Alloy, Z, and JML
  - Task management trackers like TRAC

Version Control Systems

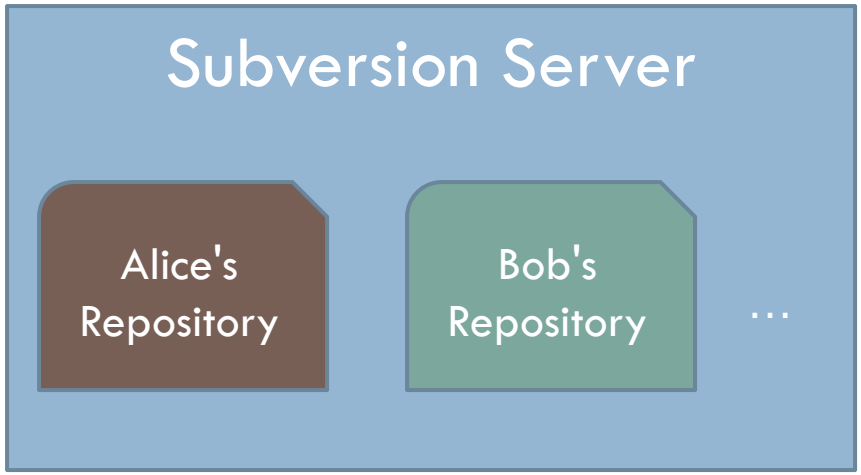
- Store "snapshots" of all the changes to a project over time
- Benefits:
  - Multiple users
    - Multiple users can share work on a project
    - Record who made what changes to a project
    - Provide help in resolving conflicts between what the multiple users do
    - Maintain multiple different versions of a project simultaneously
  - Logging and Backups
    - Act as a "global undo" to whatever version you want to go back to
    - Maintain a log of the changes made
    - Can simplify debugging
  - Drop boxes are history!
    - Turn in programming projects
    - Get it back with comments from the grader embedded in the code

Our Version Control System

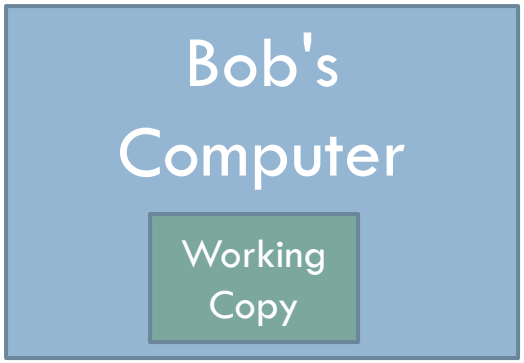
- Subversion, sometimes called SVN
- A free, open-source application
- Lots of tool support available
  - Works on all major computing platforms
  - **TortoiseSVN** for version control in Windows Explorer
  - **Subclipse** for version control inside Eclipse

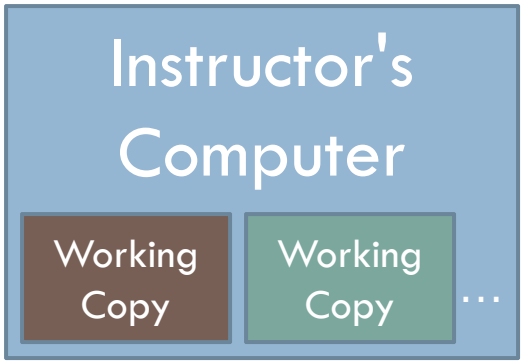
Version Control Terms

Repository: the copy of your data on the server, includes ***all*** past versions

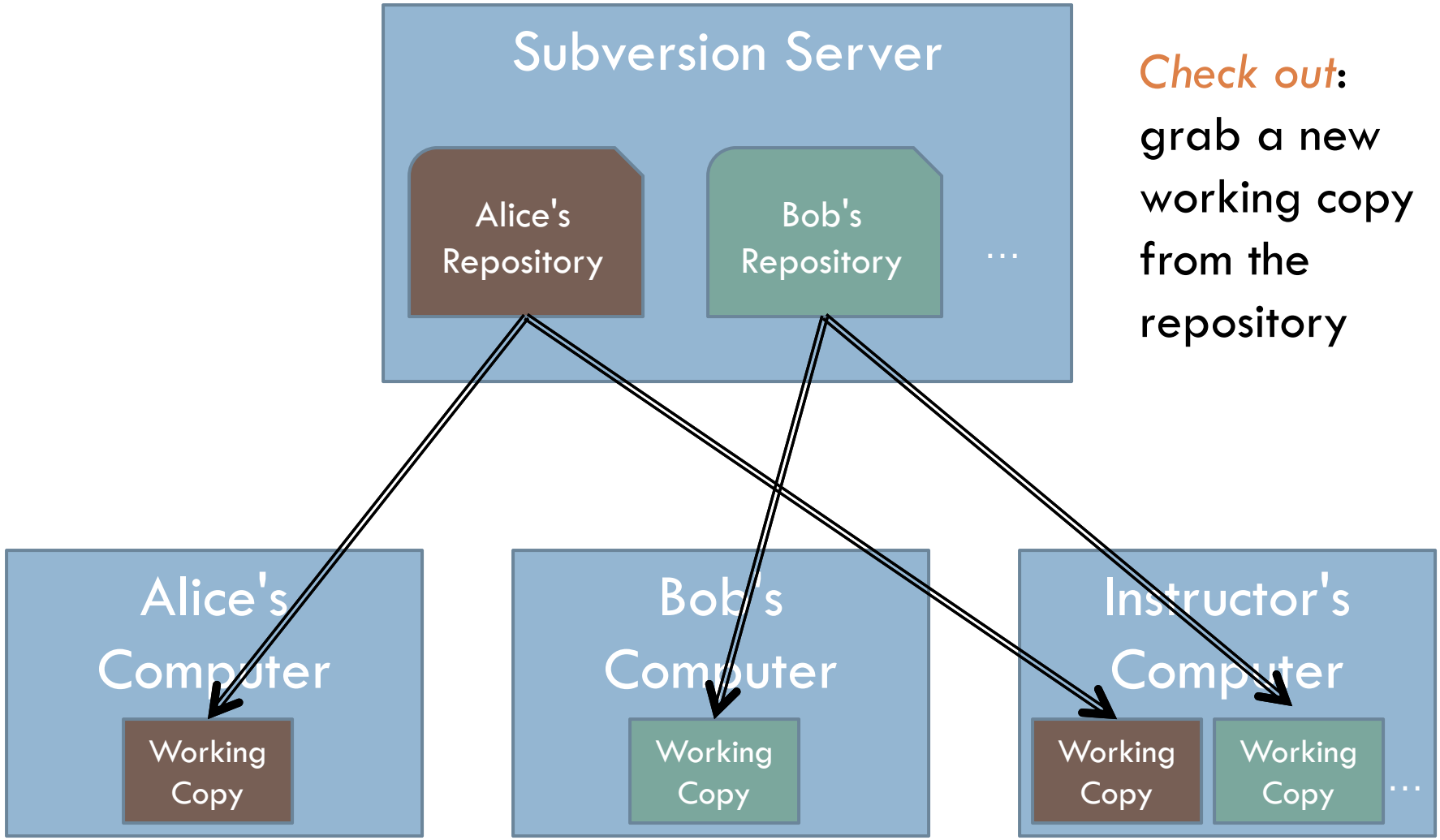


Working copy: the ***current*** version of your data on your computer



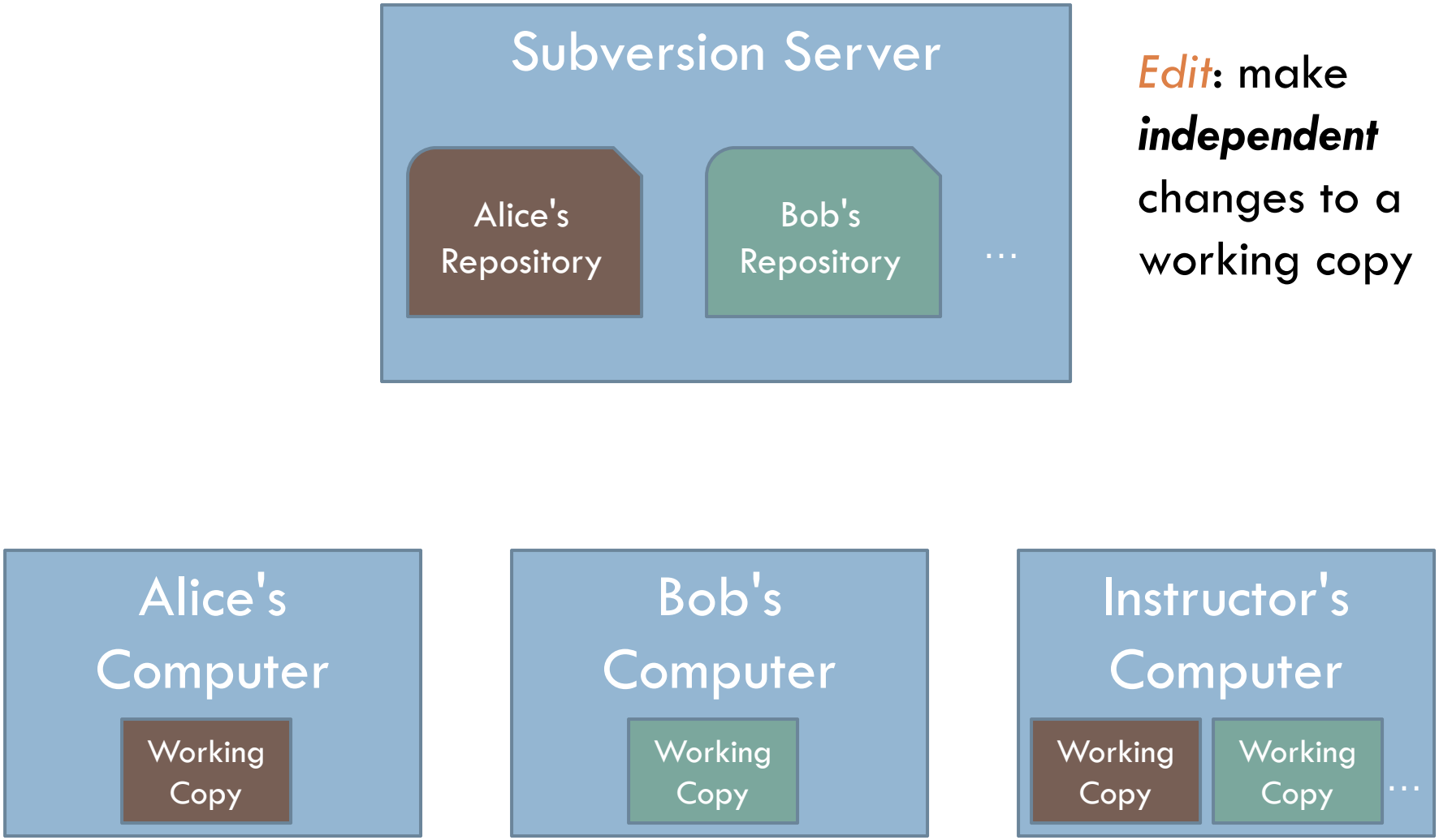


Version Control Steps—Check Out



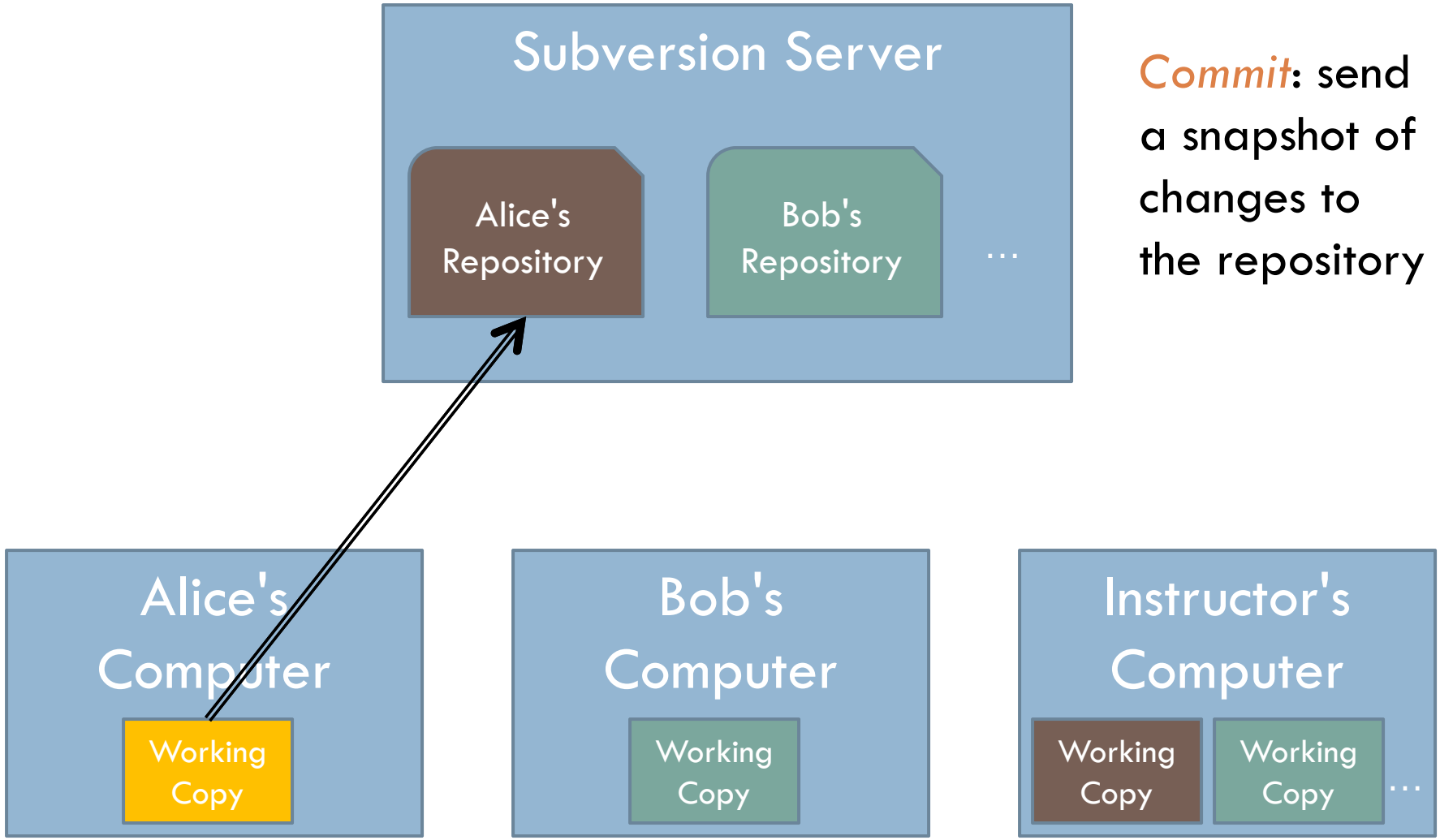
Check out: grab a new working copy from the repository

Version Control Steps—Edit



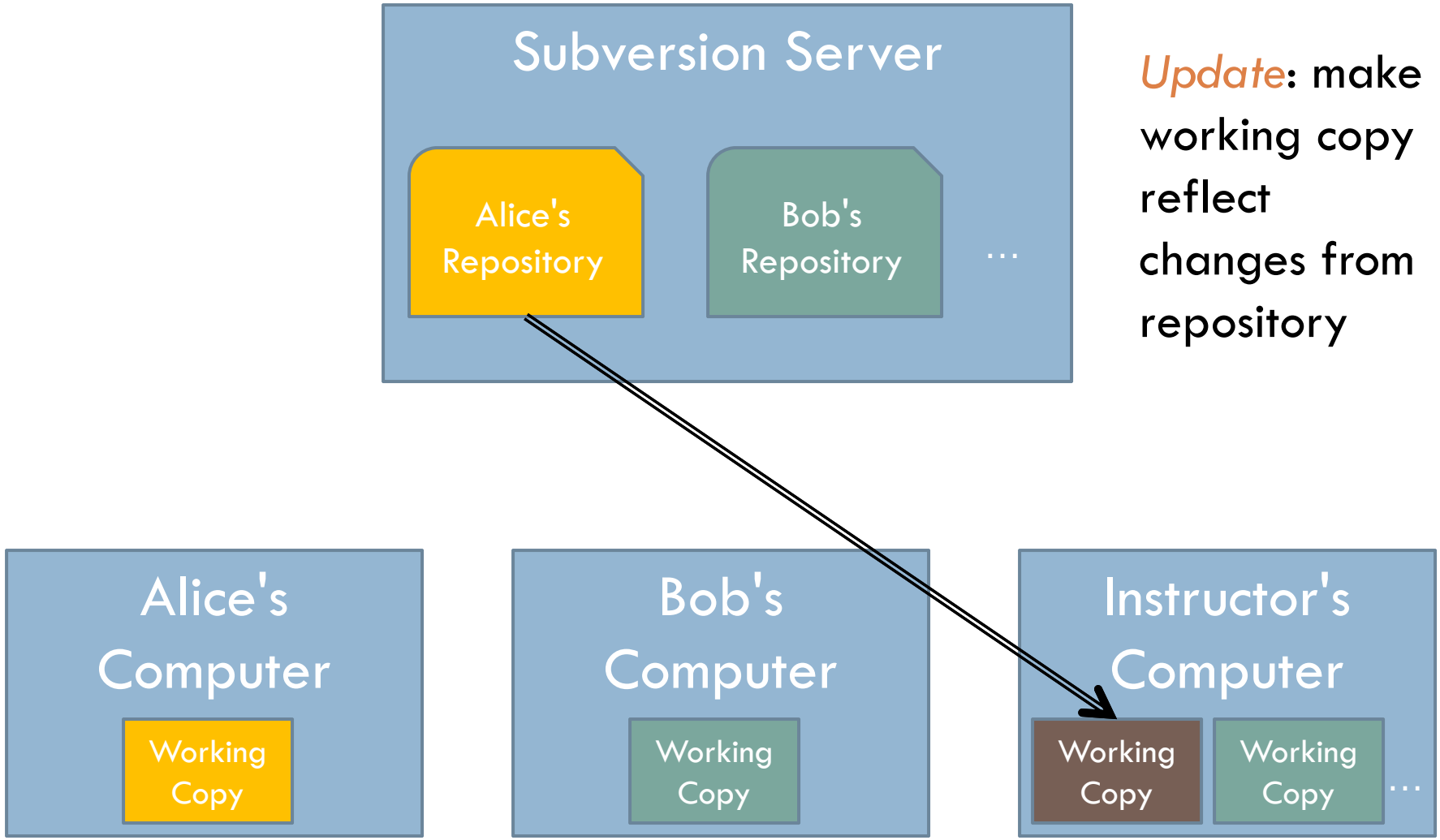
Edit: make ***independent*** changes to a working copy

Version Control Steps—Commit



Commit: send a snapshot of changes to the repository

Version Control Steps—Update



Update: make working copy reflect changes from repository

Checkout today's project: `Session01_Introduction`

Are you in the ***Pydev*** ***perspective? If not:***

`Window ~ Open Perspective ~ Other` then `Pydev`

Messed up views? If so:

`Window ~ Reset Perspective`

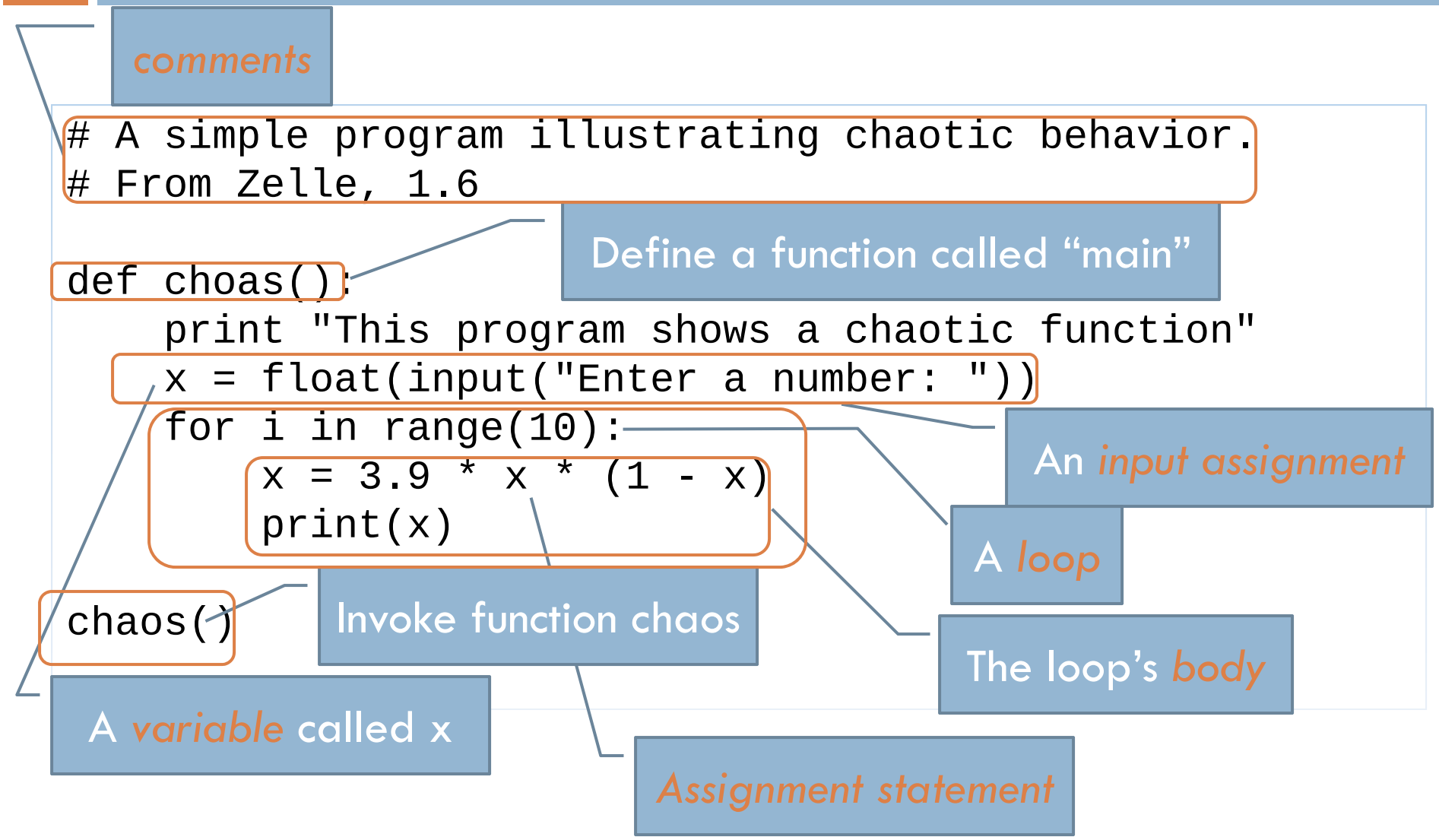
Troubles getting today's project? If so:

No ***SVN repositories*** ***view (tab)? If it is not there:***

`Window ~ Show View ~ Other`
then `SVN ~ SVN Repositories`

1. ***In your*** ***SVN repositories*** ***view (tab),*** ***expand your repository*** ***(****the top-level item) if not already expanded.*
   - If no repository, perhaps you are in the wrong Workspace. Get help.
2. ***Right-click on today's project,*** *then select* ***`Checkout`***.
   Press ***OK*** *as needed.* The project shows up in the
   `Pydev Package Explorer`
   to the right. Expand and browse the modules under `src` as desired.

A simple program that defines and invokes a function called main()



Rest of Session

- ***Check your Quiz answers*** versus the solution
  - An assistant may check your Quiz to ensure you are using the Quizzes appropriately
- ***Work on today's homework***
  - Ask questions as needed!
- **Sources of help after class:**
  - **Assistants in the CSSE lab**

    ```
    Moench F-217
    7 to 9 p.m.
    Sundays thru Thursdays
    ```

    - And other times as well (see link on the course home page)
  - **Email** `csse120-staff@rose-hulman.edu`
    - You get faster response from the above than from just your instructor