Basic Python Programs, Software Engineering Tools

Rose-Hulman Institute of Technology

Computer Science and Software Engineering

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Announcements

- Homework timing:

Day assigned	Reading quizzes due (next class day 8 AM)	Other parts of assignments due 8 AM
Monday	Tuesday	Wednesday
Tuesday	Thursday	Thursday
Thursday	Monday	Monday

Who wants to share?

Sample Scenes from HW 1

ROSE-HULMAN
INSTITUTE OF TECHNOLOGY

Outline

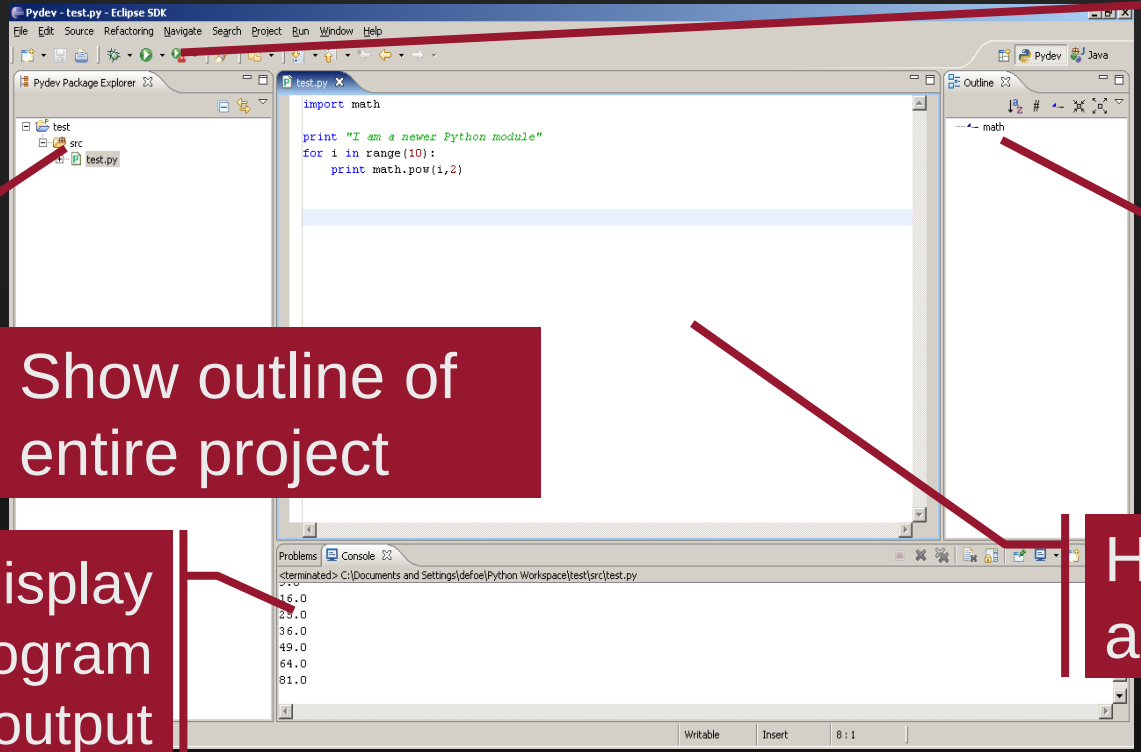
- Goals for today:
  - Get started with Eclipse and Subversion
  - Learn and practice writing small programs
    - Functions
    - The main function
    - The input-compute-output pattern
    - Examples: chaos, temperature, kph

Integrated Development Environments

- What do they do?
- Why use one?
- Our IDE – Eclipse

Details on next slides

IDEs – What do they do?



Let us compile, run, and debug

Show an outline of our module (file)

Show outline of entire project

Display program output

Help us enter and edit code

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

IDEs – Why use one?

- Harness the power of the computer to make us more productive!

IDEs – Why Eclipse?

- Powerful
- Easy to use
- Free and open-source
- An IDE for any language, not just Python

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
  - Compiled code
  - Design documents
  - Documentation
  - Tests
  - And more…

Setting up Eclipse

- Just need to do this the **first time**!
- Follow along with me
  - We will:
    - Open Eclipse
    - Set the workspace
    - Switch to the **PyDev120** perspective
  - If your setup is different, see
    - http://www.rose-hulman.edu/class/csse/resources/Eclipse/installation.htm

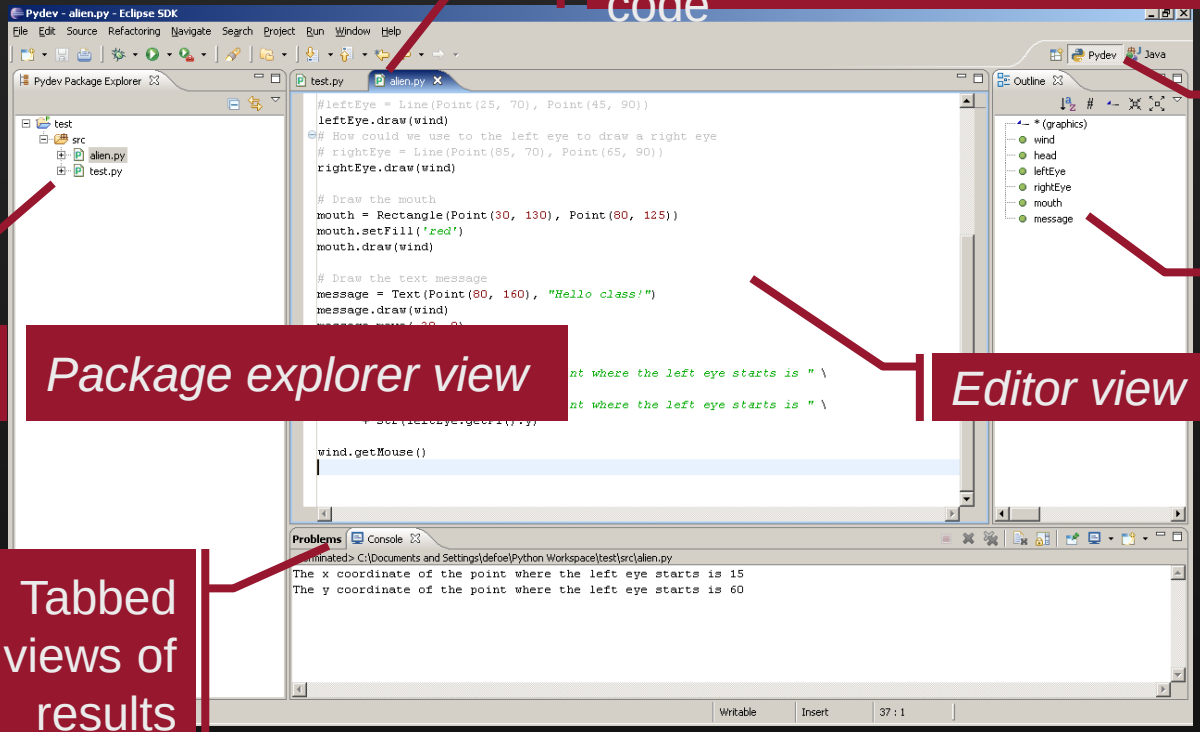
Follow Along…

1. *File ➢ New ➢ Pydev **Project***
2. *File ➢ New ➢ Pydev **Module***
3. Type `print("hello world")`
4. Run the program
5. Type a few more *print* statements, including one that is wrong.
   - See where the error message appe… …ng on it brings you to the offending line

Help us enter and edit code

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Views, Editors, PyDev Perspectives



Tabbed views of source code

Perspective switcher

Outline view

Package explorer view

Editor view

Tabbed views of results

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Eclipse in a Nutshell

- *Workspace* – where your projects are stored on your computer
- *Project* – a collection of files, organized in folders
- *Perspective* – a set of views and editors
- *View* – an area of a window showing focused information (code, outline, results, etc.)

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

Version Control Systems

- Benefits for individual work:
  - 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

Version Control Systems

- Benefits for group work:
  - 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

Version Control Systems

- Benefits for CSSE 120 programming work:
  - Three-click turn in for programming projects
  - Get back comments from the grader right in the code
  - No need for ANGEL drop boxes

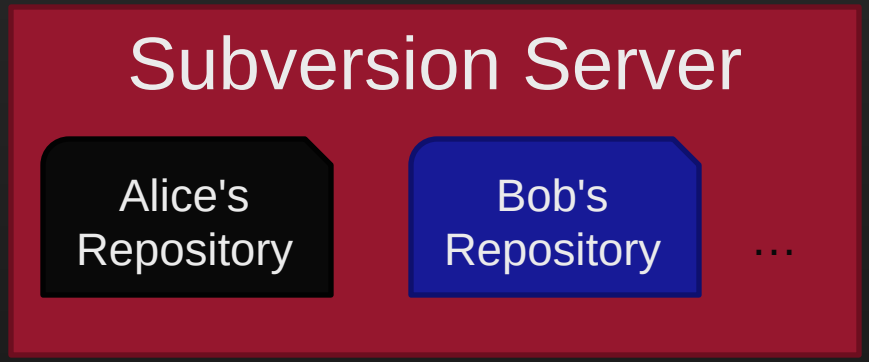
Our Version Control System

- 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

Q1a

Version Control Terms

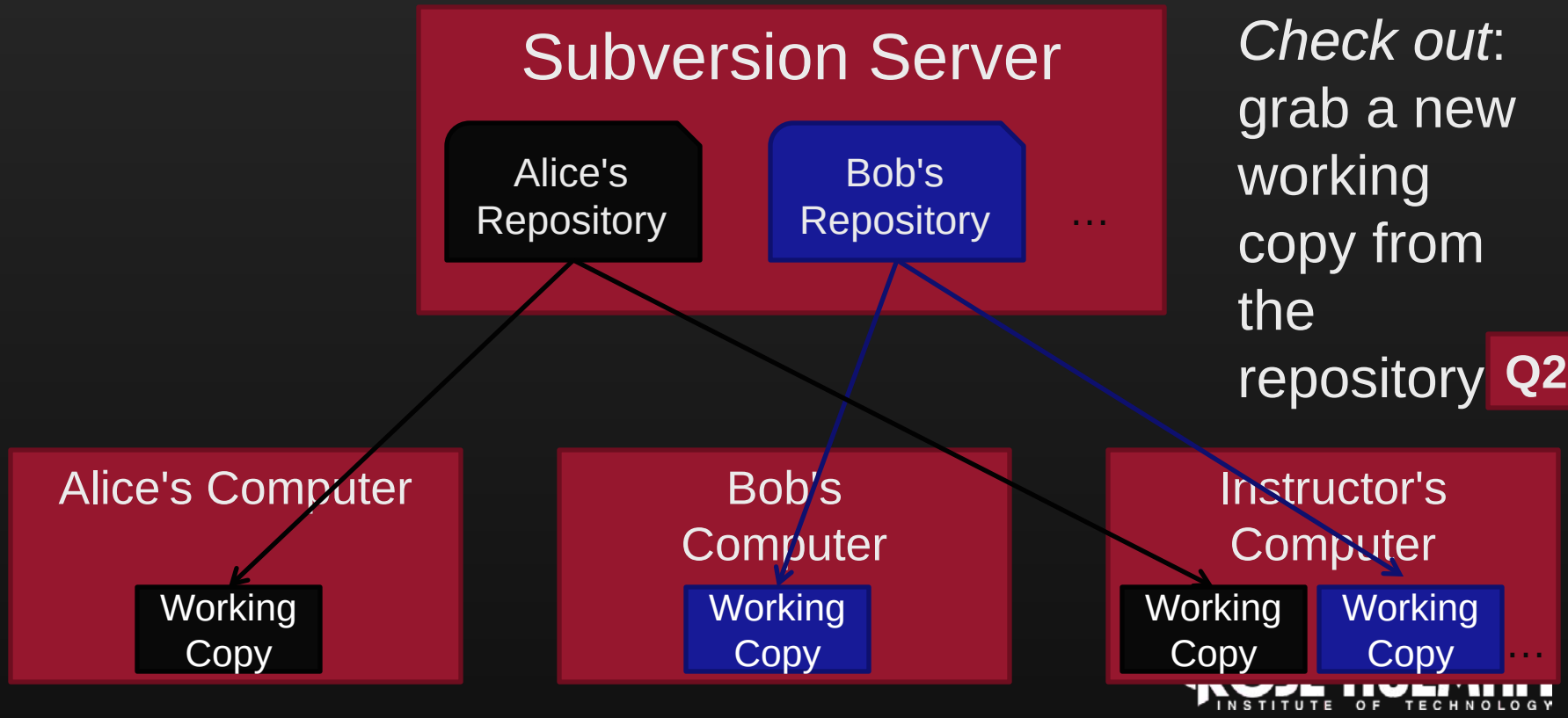
Repository: the copy of your data on the server, includes **all** past versions



Working copy: the **current** version of the files, on your computer

Q1b

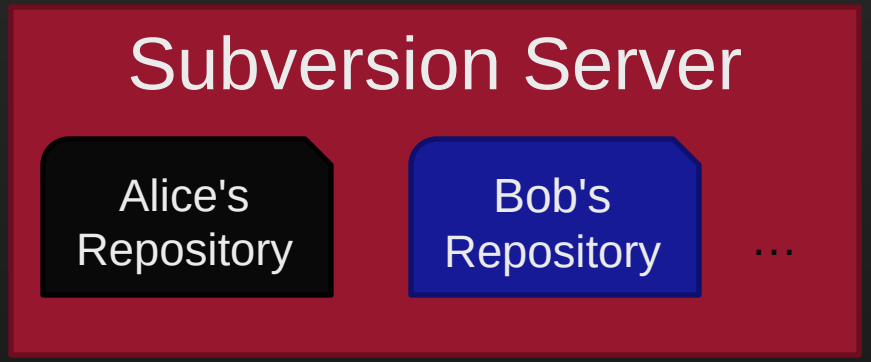
Version Control Steps—Check Out



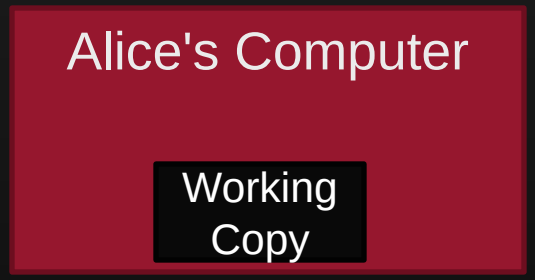
Check out: grab a new working copy from the repository

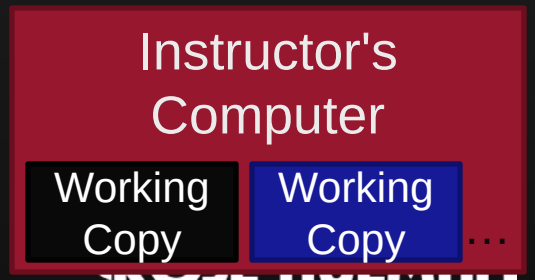
Q2a

Version Control Steps—Edit

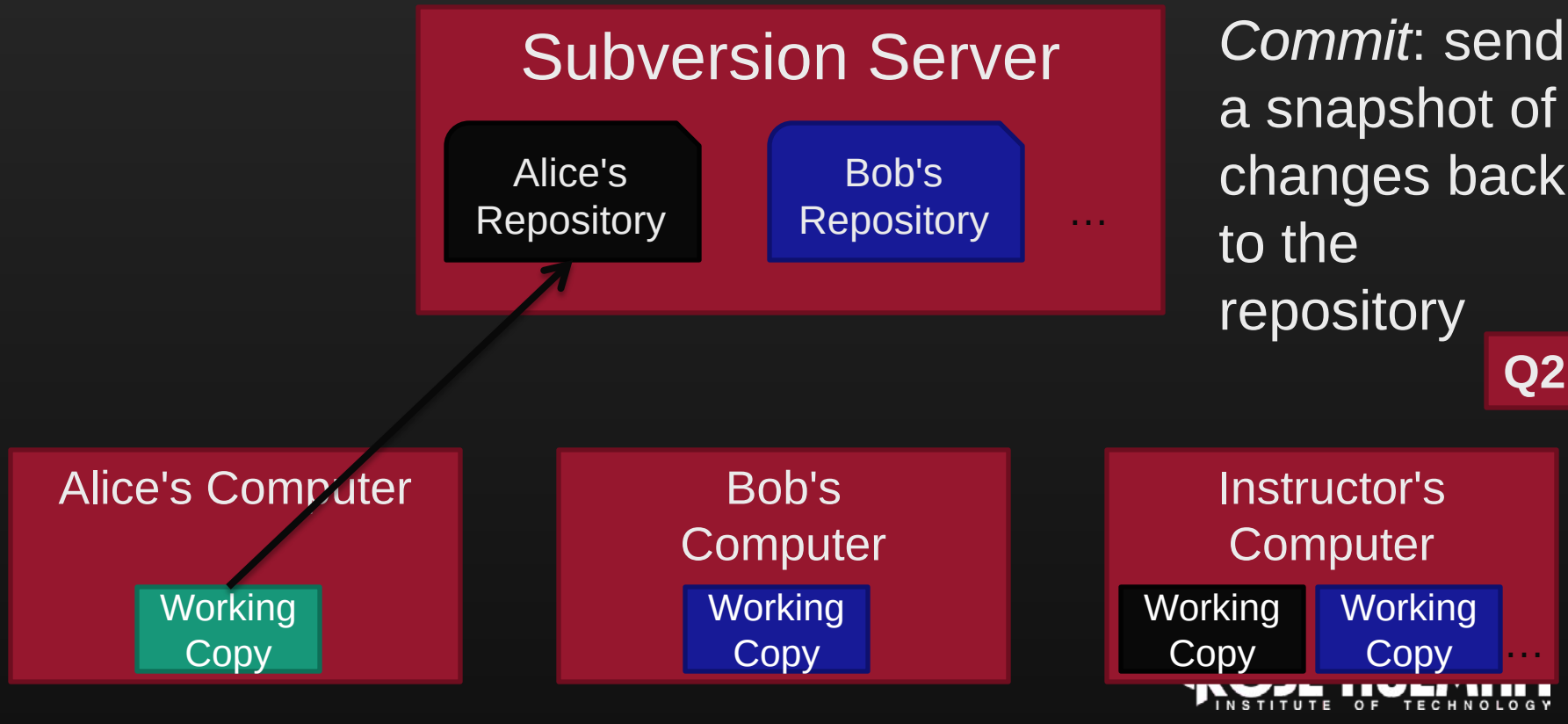


Edit: make ***independent*** changes to a working copy





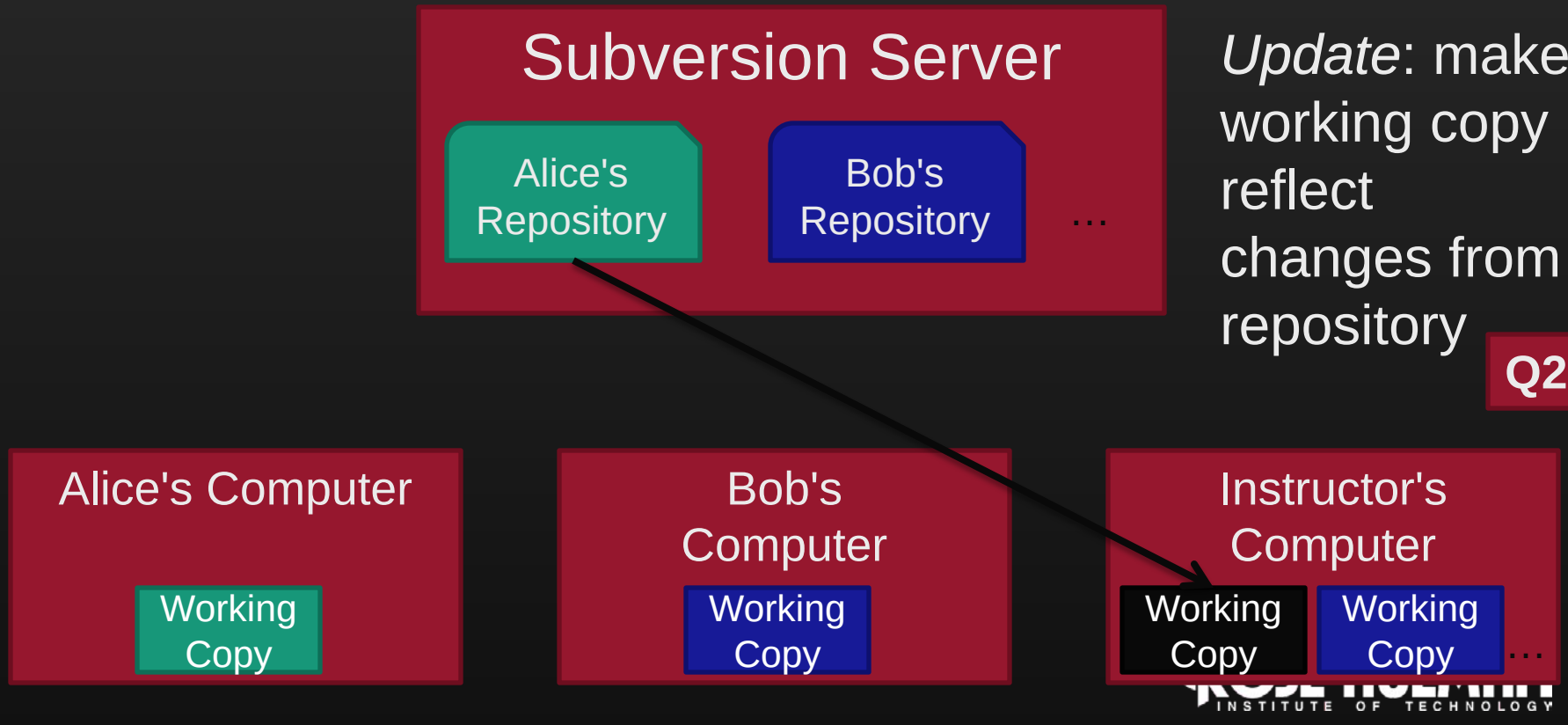
Version Control Steps—Commit



Commit: send a snapshot of changes back to the repository

Q2b

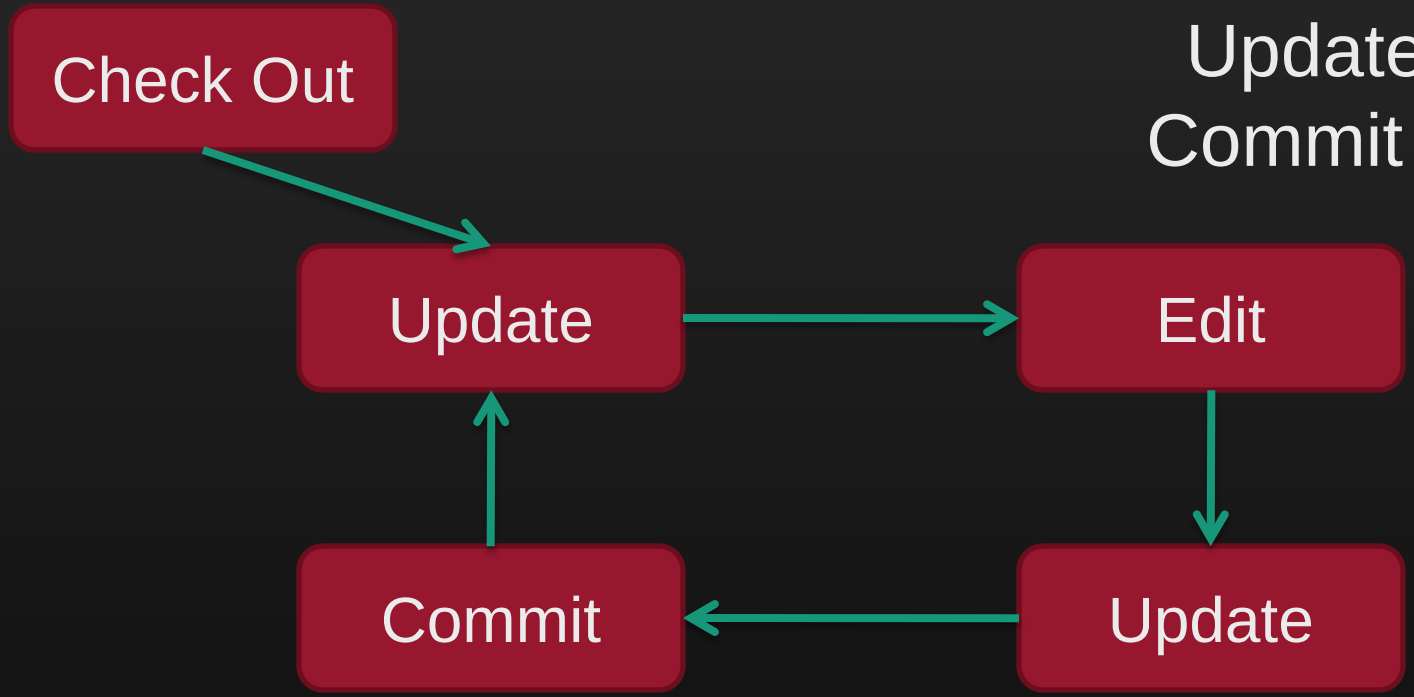
Version Control Steps—Update



Update: make working copy reflect changes from repository

Q2c

The Version Control Cycle



Update and
Commit often!

Check out today's project

- Follow along
- You'll need:
  - Subversion URL from Homework 2
  - SVN password

Functions

- Examine the **hello.py** module in your Eclipse project.
- *Functions*
  - Named sequences of statements
  - We can *invoke* them—make them run
  - They can take *parameters—changeable* parts

Parts of a Function Definition

Defining a function called “hello”

```
>>> def hello():
        print("Hello")
        print("I'd like to complain about this parrot")
```

Indenting tells interpreter that these lines are part of the **hello** function

Entering a blank line tells interpreter that we’re done defining the hello function

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Defining vs. Invoking

- *Defining* a function says what it should do
- *Invoking* (calling) a function makes that happen
  - Parentheses tell the interpreter to invoke the function

```
>>> hello()
Hello World
```

  - Later we'll define functions with *parameters*

Q3

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

The *input-compute-output* pattern

- Examine the **chaos.py** module in your Eclipse project.
- Do the TODO item in it.
  - I'll demo this with you.
- Commit your code:
  - Right-click project → Team → Commit…

Q4-7

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Breaking Down Chaos

comments

```
# A simple program illustrating chaotic behavior.
# From Zelle, 1.6

def main():
    print "This program shows a chaotic function"
    x = input("Enter a number: ")
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print x

main()
```

Define a function called “main”

An *input statement*

A *loop*

The loop’s *body*

Invoke function main

A *variable* called x

Assignment statement

Homework

- Hand in quiz
- Begin homework
  - Find it from the Schedule page
  - When is each part due? (See the schedule page if you forgot)

Q9-10

ROSE-HULMAN INSTITUTE OF TECHNOLOGY