ECE364
Software Engineering Tools Lab

Lecture 0

Course Introduction

“There are two types of people in the world:
those who cannot extrapolate from incomplete data.”*

**From ThinkGeek*

Outline

- Administrative Issues
- About the Lab
- Version Control: Subversion (SVN)
- Review: Unix File Permissions

Course Staff / Contact Info

▪ Alex Gheith	Lecturer
▪ V.K. Chaithanya Manam	Lab TA
▪ Wachirawit (Mint) Ponghiran	Lab TA
▪ Dr. Mark Johnson	Lab Admin
▪ ULA: Andrew, Anqi, Jiacheng, Zhen	

- Contact the TAs: Post on Piazza, as a private post.
- Also ee364ta@ecn.purdue.edu
  This email reaches all TAs, but not frequently monitored.

Important URLs

Piazza

https://piazza.com/purdue/fall2017/ece364/home

Announcements, Syllabus, Lecture Notes, TA office hours, contact info, useful links, assignments, Discussion, Q & A

Course Website

http://engineering.purdue.edu/ee364

Course Calendar

Blackboard (Learn)

https://mycourses.purdue.edu

Grades

Getting Help

- Office Hours: Check Piazza under the “Staff” Tab.
- All technical questions should be addressed during office hours.
  - *Please do not email us for technical questions.*
- You are encouraged to use the Piazza Q & A Forum among yourselves.
- Please do NOT post your code on Piazza.
  - If someone uses the code, it is plagiarism.
- The TAs will NOT answer questions on Piazza.
  - Only during office hours.

Emergency Preparedness

Emergency preparedness is your personal responsibility. Purdue University is actively preparing for natural disasters or human-caused incidents with the ultimate goal of maintaining a safe and secure campus. Let's review the following procedures:

- For any emergency call 911.
- 300 Emergency Telephone Systems throughout campus connect directly to the Purdue Police Department (PUPD). If you feel threatened or need help, push the button and you will be connected to the PUPD.
- If we hear a fire alarm we will immediately evacuate the building and proceed to **the MSEE atrium in bad weather or front of MSEE facing Northwestern in good weather** (location).
  - **Do not use the elevator.**
  - Proceed to nearest exit. Front doors for EE117. Back door by vending machines for EE215, EE206, and EE207.

Emergency Preparedness

- If notified of a Shelter in Place requirement for a tornado <u>warning</u> we will shelter in the lowest level of this building away from windows and doors. Our preferred location is **<u>the ground floor of EE away from windows</u>**

- If notified of a Shelter in Place requirement for a hazardous materials release we will shelter in our classroom shutting any open doors and windows.

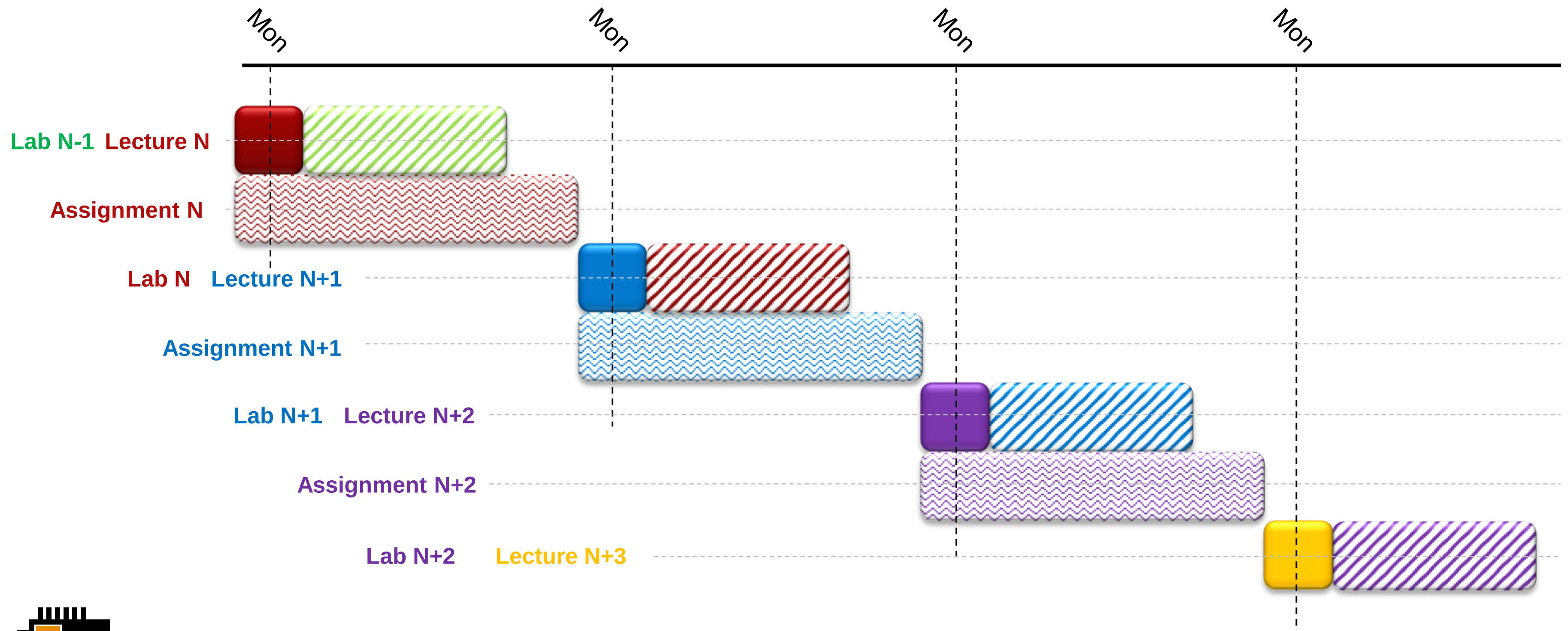
- If notified of a Shelter in Place requirement for a civil disturbance such as a shooting we will shelter in a room that is securable preferably without windows. Our preferred location is **<u>this classroom after securing or barricading doors.</u>**

About the Structure

- For a given lecture, you will be given a week to practice and complete the lab assignment.
- The assignment is released on Monday and is due on the following **Sunday**, by Midnight.

Legend

■	Lecture
▒	Assignment
▨	In-Lab



About the Lab

- Each lab consists of two parts: A lab assignment, and an in-lab exercise.
- The lab assignment must be completed by Sunday evening. It is worth 80% of the lab grade.
- The in-lab exercise must be completed in your lab session. It is worth 20% of the lab grade.
- The lab session is **One Hour and 50 Minutes**. Please show up a couple of minutes early and always **Logout** before leaving.
- If you miss a lab, you will need to have a makeup. (Check syllabus for policy.)

About the Lab

- In-lab exercise handouts are collected at the end of the session.
- Always read the instructions to identify what you need to submit to SVN, for both the assignment and the in-lab exercise. We will only grade what’s in SVN.
- In Python labs, verify you are using Python 3.4. You will lose points if you are using a different version.

Passing the Course

- Obtain **60%** or more **AND** Pass all **Course Objectives**.
- If you do not pass any of the objectives, you will obtain an “F” regardless of your cumulative grade.
- Missing labs (without excuse), exams, or not completing the project will also result in a “F”.
- Check the Syllabus for more details.

Controlling Access to Your Code

- Sharing code with your colleagues is considered cheating, the giver and the receiver are equally so.
- Do not **publish** (i.e. make public) your code on the Web (GitHub … etc.) If someone use that without your knowledge, you are <u>still</u> responsible.
- If you use code-hosting, make sure you use private repositories:
  - GitHub has a free educational license for private repositories. **Non-educational accounts are public**.
  - BitBucket is private by default.

The Syllabus

- Read the course syllabus!

- It covers important course information
  - Grading, and lab make-up policy
  - Course Objectives, Academic Honesty
  - etc..

Why Learn Bash and Python?

- Bash is a widely used scripting language for Unix/Linux environments
- Python because is a great general purpose language for scripting and application development
  - (Lots of companies look for Python experience)
- Why not C?
  - It's hard to debug and takes a long time to write
  - You are constantly dealing with unimportant/annoying details like memory allocation and pointers
  - Unless your program is flying an airplane or processing streaming HD video it doesn’t really matter that it takes 90ms or 9ms to execute…

Topics in ECE364

- An introduction to several new programming styles, concepts and tools
  - Bash and Python
  - Software Automation
  - Regular Expressions and Text Processing
  - Programming a GUI
  - Object Oriented Programming

- All stuffed into 1 credit hour!

Prerequisite Knowledge

- Everything from ECE264!
  - Variables, data types, I/O, functions (recursion), scope, call stack, pointers, arrays, structures, lists, queues, trees …
- Basics on using the command line
  - Running commands, navigating directories …

About the Lab

- This lab is NOT about the syntax. It is about:
  1) software engineering process.
  2) Tools: Design, Problem Solving, Debugging, etc.
- Set aside C-efficiency, and learn the software process.
- Lab Workload:
  - Lecture.
  - Prelab.
  - Practice.

About the Lab

- Do NOT translate the syntax. Learn the language.
- Consider finding the average of an array:

C

```c
// You have some array 'n'.
int length = sizeof(n) / sizeof(n[0]);
int total = 0;

for (int i = 0; i < length; i++) {
    total += n[i];
}

double avg = total / length;
```

Python

```python
# You have some list 'n'.
length = len(n)
total = 0

for i in range(length):
    total += n[i]


avg = total / length
```

- In Python, this is how this is done:

```python
avg = sum(n) / len(n)
```

Version Control System (VCS)

The Need For Version Control

- Source code for programs may contain multiple directories with many files
  - A small program may have 1 file
  - A large program may have ***thousands*** of files
- How would you track changes between files?
  - Make backups or copies after every change?
  - Maintain a single “CHANGES” file that lists what was done?
- What would happen if you made a mistake last week and now just found out?

Subversion (SVN)

- Is a version control system
  - Manages changes to files and directories
  - Can handle multiple users concurrently
  - Supports local or remote storage of repository data

- SVN is REQUIRED for ALL assignments
  - This is how you will upload code for grading
  - No code often results in a zero!

- Lab 0 will cover the basics of SVN

SVN Terminology

- Repository - a central store that contains all distinct copies (revisions) of your work.

- Revision – a unique snapshot of the repository contents at a specific point in time
  - A revision is identified by a unique number
  - A revision represents the state of all files at a point in time

- Working Copy – a copy of what is stored in the repository.
  - Modifications to file and directories are made to a working copy
  - Usually the working copy is located on your local machine

SVN Terminology (2)

- Checkout - the act of creating a new working copy
  - Downloads whatever is stored in the repository
  - You can checkout specific revision, not just most recent
- Committing - the act of uploading changes from your working copy to the repository
  - Creates a new revision in the repository
- Updating - the act of downloading changes from the repository to your working copy
  - Your working copy could be several revisions old
  - Synchronizes the working copy to the repository

SVN Basic Commands

- `svn checkout <repo-url> <working-dir>`
  - Creates a new working copy of the specified repository
  - Making changes to a file or set of files in the working copy does not change the contents of the repository

- `svn commit`
  - When you are satisfied with your changes you must commit them to the repository
  - All changes to your working copy will be uploaded as a new revision of your repository
  - A text message is usually added to indicate what changes were made for this commit

SVN Basic Commands (2)

- `svn update`
  - Synchronizes your working copy with the most recent revision of the repository
  - Other programmers may have made changes to other files and committed them to the repository
- What happens if two programmers made a change to the same line of the same file?
  - Called a conflict
  - We will not worry about this but SVN has tools for resolving conflicts

SVN Basic Commands (3)

- `svn add <file/directory>`
  - Stages a file or directory for addition into the SVN repository
  - **Important:** Files are not added until the next `svn commit`

- `svn rm <file/directory>`
  - Stages a file or directory for deletion from the SVN repository
  - **Important:** Files are not deleted until the next `svn commit`

- ***** Changes to the working copy are not reflected in the repository until you commit! *****

SVN Basic Commands (4)

- `svn cp <source> <destination>`
  - Copies a file or directory from a source to destination
  - **Important:** New files are not added until the next `svn commit`

- `svn mv <source> <destination>`
  - Moves a file or directory from a source to destination
  - **Important:** New files are not added and old files are not removed until the next `svn commit`

- ***** Always use SVN commands to perform basic directory operations *****

SVN Revisions

- SVN creates a new revision each time you commit changes

- You can revert individual files or entire directories to a specific revision using the update command

- `svn update -r<revision> <file/directory>`
  - `<revision>` as a number: `1234`
  - `<revision>` as a date: `'{2012-01-10 23:49}'`

SVN Keywords

- Special keywords expand at commit time to provide additional versioning information

`$Author$`	Username of the person committing
`$Date$`	Date and time this file was committed
`$Revision$`	The revision number of this file
`$HeadURL$`	The URL where this file lives in the repository
`$Id$`	A combination of the above keywords

- Keywords are typically placed in comments at the beginning of a source file

Unix File Permissions

Unix File Permissions

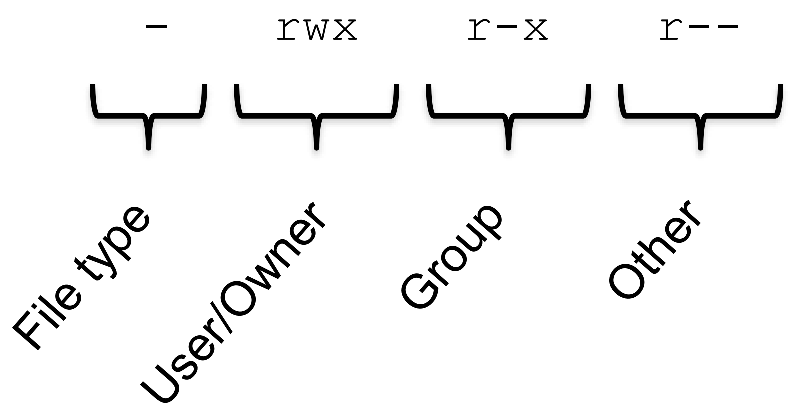
- Unix type operating systems control a users access to files and directories through a set of permissions
  - Read (r) – the ability read the contents of a file or list the contents of a directory
  - Write (w) – the ability to change the contents of a file or directory
  - Execute (x) – the ability to execute the contents of a file or access the contents of known files in a directory

Unix File Permissions (2)

- Every file (and directory) contains three sets of permission classes that control access for different users
  - User/Owner (u) – the individual user who owns the file
  - Group (g) – the group of users associated with the file
  - Other (o) – everyone else
- By combining permissions for the user, group and other classes the ability to access a file is controlled

Unix File Permissions (3)

- A set of permissions can be represented as a string



- In this example
  - User/Owner has all permissions
  - Group has read and execute permissions
  - Other only has read permissions

Unix File Permissions (4)

- The file type field does not represent a permission, it indicates what the file is

File Type	Description
`-`	A regular (ordinary) file
`d`	A directory
`l`	A symbolic link
`c`	A character device file
`b`	A block device file
`p`	A named pipe (FIFO)

Unix File Permissions (5)

- File permissions are typically represented using octal (base-8) values rather than strings
  - Each permission (r, w, x) is represented as one bit
  - If a permission is set the bit value is 1

Binary Value	Octal Value	Permissions
`000`	`0`	No permissions set
`001`	`1`	Execute only
`010`	`2`	Write only
`011`	`3`	Write + Execute
`100`	`4`	Read only
`101`	`5`	Read + Execute
`110`	`6`	Read + Write
`111`	`7`	Read + Write + Execute

Unix File Permissions (6)

- A three digit octal value is used to represent the complete set of permissions (user, group, other)
  - Any octal number between 000 and 777 can be used

Octal Value	Permissions
`000`	No permissions set
`644`	User: r+w, Group: r, Other: r
`755`	User: r+w+x, Group: r+x, Other: r+x
`777`	User: r+w+x, Group: r+w+x, Other: r+w+x
`124`	User: x, Group: w, Other: r
`555`	User: r+x, Group: r+x, Other: r+x
`666`	User: r+w, Group: r+w, Other: r+w
`111`	User: x, Group: x, Other: x
`321`	User: w+x, Group: w, Other: x

Changing File Permissions

- `chmod <permissions> <file/directory>`
  - `<permissions>` the file permissions to set
  - `<file/directory>` the path to the file or directory

- The `<permissions>` argument can accept the octal representation of file permissions
  - `chmod 755 my_script.sh`
  - `chmod 644 /home/ee364ta/roster.txt`

Changing File Permissions (2)

- Permission bits can be set and cleared using a special string: `<class><op><flags>`
  - `<class>` is “u” (user), “g” (group), “o” (other) or “a” (all)
  - `<op>` is “+” (set) or “-” (unset)
  - `<flags>` is “r” (read), “w” (write), “x” (exec)

Example	**Result**
`chmod u+rwx a.out`	Set r, w and x for user/owner
`chmod g-rw bar.txt`	Unsets r and w for group
`chmod u+rwx,g-w,o-w /work/data`	Sets r, w and x for user/owner, unsets w for group and other
`chmod a+rw foo.bin`	Set the r and w for user/owner, group and other