Version Control and Git

July 10, 2017

Midterms

Midterm Stats

- Mean: 63.57
- Median: 65.5
- Max: 70
- Standard Deviation: 6.572
- Count: 14

Question Walkthrough

Version Control

Why version control in the course?

- Practical
  - **needed** for contributing to others' projects
  - **strongly suggested** for managing your own
- Structural
  - Projects aren't static, projects are their history too
- Group Development
  - Good version control enables working in teams

The Many Flavors of Version Control

- CSV
- Subversion (SVN)
- Mercural
- Perforce
- BitKeeper
- Team Foundation
- Git

Why Git?

- **Distributed**
  git, mercurial, Perforce

- **Fast**
  git, Perforce, BitKeeper, Team Foundation

- **Free and Open Source**
  git, cvs, svn, mercurial

- **Popular**
  git, mercurial, svn

Git Overview

- Distributed system
- Tree based depiction of changes
- Nodes are "diffs"
- Edges are transitions between states in the project

Git Terms

- staging area
- commit
- HEAD
- branch
- merge
- rebase
- clone

Git Demo –>

Git Terms

- staging area
- commit
- HEAD
- branch
- merge
- rebase
- clone

Using Git in Teams

- Git is distributed
- Distribution allows large teams to work together
- Common git practices make team work possible

Common Simple Git System

- Three types of branches
  - master <– rolling collection of all changes
  - release <– each deployed release
  - feature <– small branches, each representing a single change