PURDUE UNIVERSITY®

CS 307: Software Engineering

Exam Review

Prof. Jeff Turkstra

Midterm Exam

- Combination of 45-50 true/false and multiple choice questions
  - Scantron, #2 Pencil
- Assigned seats
- Be sure to bring your student ID

Midterm Exam

- You should only have your ID and a pencil or two at your seat
- Sit in your assigned seat
- Make sure that the color of your scantron matches the color specified on your exam coversheet
- Do not open your exam until you are told to begin
- The exam will end promptly at 12:18 PM
  - When you are told to stop, put your pencils down. Continuing to write will result in a score of 0

Exam topics

- Software, what is it?
- Types of software
- How do engineers approach development?
- What is software engineering?
  - Commonalities and differences with other disciplines

- Code of ethics and professional practice
- Software development
  - Stakeholders
  - Quality
    - Attributes, etc
- Software Life Cycles
  - Models
  - Documents

- Scrum
  - Documents
  - Meetings
  - Team structure
- Revision control
  - Models, snapshots, deltas, etc

- Git
  - General workflow
  - Branches
  - Working with remotes
  - Merging
  - Rebasing
  - Cherry picking
  - Bugs

- Domain analysis
- Requirements analysis
  - Greenfield/brownfield
  - Gathering requirements
  - User stories
  - Use cases
    - Diagrams
      - Extensions, inclusions, generalizations
  - Exploring/organizing
  - Managing/reviewing

- Reusability
  - Why?
  - Why not?
  - Frameworks
    - Slots, hooks
  - Product lines

- Client-server architecture
  - Distributed system (eg, attributes)
  - Terminology
  - Basic sequence
  - Tradeoffs

- Unified Modeling Language (UML)
  - What is it? When should you use it?
  - What diagrams are there?
    - Are they static or structural?
  - Associations/multiplicity
  - Generalization/discrimination
  - System domain model vs system model

- Design patterns
  - What are they?
  - When is each one appropriate?
- Interactions and behavior
  - Measuring class independence
  - Cohesion, coupling
    - What forms of each are there?
    - Which are most/least beneficial

- Sequence diagrams
  - General layout, usage
  - How to draw
- State diagrams
  - Same
- Activity diagrams
  - Forks, joins, rendezvous, swimlanes

- Architecting and designing software
  - Design quality
  - What is design?
  - Terminology
    - Component, module, system, etc
  - Good design
  - Principles
    - Divide and conquer, cohesion, coupling, abstraction, etc

- Architectural decisions
  - What makes a good model?
  - Stability
  - Patterns
- Users, usability
  - How? Why? UI design
  - Usability, likeability, utility, etc
  - Principles
  -

- Inspecting
  - Common causes of defects
  - Terminology
    - Failure, error, defect, etc
  - Fault feedback ratio (FFR)
  - Inspection
    - When can it be done?
    - Steps, roles, logging
  - Inspecting vs testing

- Software testing
  - What is it?
  - vs. Debugging
  - Types of faults
  - Functional vs structural
  - Types (correctness, performance, parts and statement, etc)
  - Testing strategies
    - Big bang, sandwich, etc
    - Stubs and drivers

- Blackbox vs whitebox
- Common defects
- Formal test cases
- Integration, unit, regression tests
- Product release phases

- Project management
  - What is it?
  - Re-engineering
  - Refactoring
  - Cost estimation
    - Principles
    - Scrum poker
  - Teams
    - Types
    - Skills

- Scheduling/tracking
  - PERT, Gantt charts
- Risk Analysis
  - What is it?
  - Types
  - How do they happen?
  - Identification, estimation, and evaluation
  - When should you do it?
  - Risk table
  - Precision vs accuracy

- Bias
- System failure probability
- Classic mistakes
- Lowering risks

- Peopleware
  - Hierarchy of needs
  - Social styles
    - How they might conflict
  - Stress
  - Unmet needs
    - Different behaviors
  - Good team environments vs teamicide
  - Lizard logic rules
  - Good manager attributes

Exam topics
2-5 questions each

- Software and software engineering
- Software life cycles
- Version control
- Requirements analysis
- Reusability
- Unified modeling language (UML)
- Design patterns
- Interactions and behavior

- Architecting and designing software
- Users, usability, and inspection
- Software testing
- Project management
- Risk analysis
- Peopleware

Questions?