

CIS 4932 – Special Topics (Software Testing) Summer 2004

Instructor:	Dr. E.L. Jones	Office:	307 TEC A
Class Meeting:	3:00-6:00 Tuesday 024 TEC A	Office Hours:	Tues 4-5:30; Mon 2-3:30
Course Email:	cis4932joe@cis.famu.edu	Phone:	599-3042
Course Website:	www.cis.famu.edu/~cis4932joe		

PREREQUISITE: COP2532 (Program, File & Data Structures).

COURSE DESCRIPTION: The purpose of this course is to build the skills necessary to do software testing at the function, class and application level. Students will be taught concepts of black-box (functional and boundary) and white-box (coverage-based) testing, and will apply these concepts to small program, components (functions and classes) and applications. Students will also be taught evaluative techniques such as coverage and mutation testing (error seeding). This course introduces the software engineering discipline of software quality engineering and to the legal and societal issues of software quality.

TEXTBOOKS:

1. *Teach Yourself Unix in 24 Hours, 3rd Edition*, SAMS Publishing.
2. Course notes will be available on the website.

STUDENT RESPONSIBILITY: The student is responsible for material covered in class lectures and discussion, assigned readings, and course projects. **Class attendance is essential** since this is a lab-based course. The student must keep all graded work should a dispute/appeal arise. The student has **ONE WEEK** after assignments have been returned to resolve any discrepancies. When an excused absence occurs, the student must submit a copy of an official excuse within **ONE WEEK** of returning to class or the Instructor will honor the excuse.

ACADEMIC MISCONDUCT: Unless an assignment is designated a team project, each student is expected to do his or her own work. Any student who submits work done by another student, with or without that student's knowledge, has committed *plagiarism*. Any form of unauthorized assistance during an exam will be treated as *cheating*. Plagiarism and cheating will be dealt with severely, and may result in a failing grade for the work in question or for the entire course.

LABORATORY ASSIGNMENTS: There will be 10-12 in-class supervised lab exercises with a companion out-of-class assignment. All work requires the use of the Unix system. Some require programming in C++, Visual Basic and the Unix C shell scripting language; others require the use of a web browser and Microsoft Excel. One or more projects will be designated as team projects. All work must be submitted to the class repository on the CIS Department's Unix network. Procedures for submitting assignments are posted on the course website. Projects are **due at 2PM** on the assigned due date. **No lates will be accepted.**

EXAMS: There will be three (3) exams, each counting 10% of the overall grade, and a final exam counting 10% of the overall grade. One or more exams will be administered as an on-line test.

COMPUTING FACILITIES: All submitted programs must be written in C++, and must compile and execute on the CIS Department's Unix network. Electronic communication with the Instructor should be directed to cis4932joe@cis.famu.edu.

ATTENDANCE POLICY: The University's attendance policy will be followed.

MAKE UP POLICY: Penalty-free make-ups are given **only for exams** missed due to an officially excused absence. To qualify, the student must: (1) give **prior notification** via e-mail; and (2) present an official excuse from his/her Dean within one week of the exam date. Otherwise, the Instructor reserves the right to deny a make-up, to give an alternative exam, or to deduct up to 50% of the score earned by the student.

INCOMPLETE WORK: A grade of **I** will be assigned **ONLY** in the event of a medical emergency, provided the student has a **PASSING** grade at the time of the emergency.

GRADING POLICY: The course grade is computed as shown below. Grades are based on accumulated points, which you will be able to determine by executing the myGrade.csh utility.

ACTIVITY	POINTS	GRADING SCALE
Exams (3)..... ~30%	300	91-100% : A
Projects (10+)..... ~40%	400	81-90% : B
Quizzes (10)..... ~10%	100	71-80% : C
Homework ~10%	100	60-70% : D
Final Exam ~15%	150	below 60 : F
Total Possible Points	1000	

Conceptual Objectives: Upon completion of this course, the student should understand

- ≈ the software test life cycle
- ≈ the relationship between testing, software quality and other verification techniques
- ≈ the theoretical limits of software testing
- ≈ concepts and techniques for black-box and white-box testing
- ≈ test coverage measures such as statement, branch, and path coverage
- ≈ relationship between testing and reliability, safety and system security
- ≈ the SPRAE (specification-premeditation-repeatability-accountability-economy) framework for testing practice
- ≈ techniques for test automation
- ≈ the challenges of object-oriented testing
- ≈ management procedures for software testing.

Performance Objectives: Upon completion of this course, the student should be able to perform the following tasks

- ≈ Use an advanced Unix text editor (emacs or vi) to create and edit a C++ or C shell program
- ≈ Compile and execute C++ programs in a Unix environment
- ≈ Write simple shell scripts with arguments and inputs
- ≈ Design functional and boundary test cases
- ≈ Devise and document a test strategy (plan)
- ≈ Develop manual test scripts
- ≈ Conduct testing according to a test script and document results

- ⌘ Write test drivers to automate testing a function, object (class) or program (application)
- ⌘ Write test scripts to automate the set-up of a test environment and execute test scripts
- ⌘ Evaluate the results from a test session
- ⌘ Write clear and concise write problem reports (test incidence reports)