CSC 408F – SOFTWARE ENGINEERING FALL 2001/2002

Mid Term Test (25% of course mark) October 31, 2001.

5 questions on **1** page. **100 marks total. 50 minutes total Open Book & Notes**
Answer ALL 5 questions. All questions have equal weight. WRITE LEGIBLY!

1. Discuss the advantages and disadvantages of an organization in which the primary departmentalization is vertical (i.e. by specialty, such as databases, human-computer interfaces, graphics programming) as opposed to an organization in which the primary departmentalization is horizontal (i.e. requirements analysis, software design, implementation, testing).

2. You are managing a large software project (75 people, 3+ years development time), and have allocated 3% of your budget for the effort required to build *traceability* into the software. Your management is questioning the need for that expenditure. Give a justification for your budget allocation

3. The ideal goals of software engineering are

— to produce software that is absolutely correct
— to produce software with a minimum of effort
— to produce software at the lowest possible cost
— to produce software in the least possible time
— to maximize the profitability of the software production effort
— to produce software that can be maintained with minimum effort

In the context of the CSC408F course project (where profitability is measured in terms of marks), give an ordering of these goals from most important to least important. Justify your answer.

4. You have been hired by a company that builds large software packages for PCs running Windows. The company has discovered that many of their customers are moving to Linux on their PCs. They want you to examine their software and advise them of the effort require to port their software to a Linux environment.
What information would you want to gather in order to advise them?
What portability issues would you expect to be most important?
How would you proceed?

5. Several models for the software development process were discussed in lectures

— Waterfall Model
— Prototyping
— Incremental Development
— Spiral Model
— Rapid Application Development

Which of these techniques would be most suitable for a small (25,000 lines of C, 6 months effort) software project ? Which of these techniques would be most suitable for a very large (5,000,000 lines of C++, 5 years effort) software project?
Justify your answer.