CSC 408S – Software Engineering   Winter 1999/2000

Mid Term Test (25% of course mark)   February 29, 2000.

7 questions on **1** page. **100 marks total. 60 minutes total**
Answer ANY 5 questions. All questions have equal weight. WRITE LEGIBLY!

1. Assume that you are designing a buffer manager similar to the course project but with the additional requirement that the buffer manager software should be *highly portable*, i.e. really easy to move between different hardware platforms and different operating systems. Describe how you would design a buffer manager to achieve high portability.

2. The string concatenation function `strcat` in the C string library has the prototype

```
char * strcat( char *s1 , const char * s2 ) ;
```

The function appends characters from the string pointed at by `s2` to the string pointed at by `s1`. Devise a set of test cases that will thoroughly test the `strcat` function.

3. The software project you are managing is in trouble. In order to achieve your schedule and performance goals, your technical experts tell you that you **must** do one of

1. Increase the amount of coupling between the modules in the system

or 2. Decrease the amount of cohesion within many of the modules in the system

or 3. Plead with management for more time and budget for the project. This alternative will probably negatively impact your chances for advancement in the company and also carries a significant risk that your project will be cancelled.

Which alternative would you select? Justify your choice.

4. In selecting Phase B software to use in Phase C of the course project, your team had to face the risk that the software you selected might contain serious errors or inefficiencies. Describe the steps that your team took *during the selection process* to try and minimize this risk. What further steps should you take during Phase C to deal with this risk?

5. One of Stenning's Project Hygiene Principles is

> Both people and products should be insulated from the effects of changes that are not (currently) of relevance.

Explain why this is a worthwhile principle. How would you implement this principle in a large software project?

6. A desirable property of a requirements is that it should be *testable*, i.e. it should be possible to determine if an entity or activity in a system satisfies a requirement or does not satisfy the requirement. This determination should be algorithmic and repeatable (i.e. it should not depend on the judgment of the person making the determination).
Explain why testable requirements add in the production of high quality software. What steps should be taken to make requirements testable?

7. Define *traceability.* Describe how traceability can be used to improve the quality of a software system.