UNIVERSITY OF DUBLIN

TRINITY COLLEGE

CS3BA71

Faculty of Engineering and Systems Sciences

School of Engineering

BA (Mod) Computer Science
Junior Sophister Examination

Trinity Term, 2002

3BA7 — Software Engineering and Compiler Design

Friday 24 May, 2002 Mansion House 9:30 - 12:30

Dr JA Redmond, Dr DM Abrahamson

Attempt five questions, at least two from each section.
Please use separate answer books for each section.

Section A

1. i. Responsibility-Driven Design with Classes, Responsibilities and Collaborations technique (RDD with CRC—Wirf's Brock et al) can be represented in 26 steps in six phases. Briefly give the six phases with diagrams where necessary and discuss briefly what each phase does.

   ii. Comment on RDD with CRC as an object-oriented design method based on your experience (or not) of using it in the ZIP (ATM) and second (own choice) assignments. If your team did not use RDD with CRC give the reasons why and explain and justify the method the team did use.

   iii. Give four items you would like to see in an automated object-oriented design package for RDD with CRC?

   iv. Give four rules-of-thumb which are useful in RDD with CRC.

2. i. Discuss the term "software maintenance" indicating the contradictions in this term. What are the 3 main areas which are usually meant by maintenance?

   ii. Write a brief note on Rapid Incremental Application Development (RIAD) as discussed and used during the course.

   iii. You have been appointed Manager of a major software project and must decide on how it should be manned. Give four rules of thumb for teams which you would use.

3. During the academic year you used Responsibility Driven Design (RDD) with Classes, and Responsibilities and Collaborations (CRC) as software system design methods. To control the design and implementation process the Coverdale Systematic Approach (CSA) was discussed and suggested. Discuss the Coverdale Systematic Approach by:

   i. Outlining the method and its steps.

   ii. Briefly discussing how useful it was or was not in a team situation.

   iii. Giving its advantages and disadvantages.

   iv. Discussing briefly how the method can be used in both a 2-dimensional and a 3-dimensional fashion.

4. i. Discuss Mantei's three programming teams giving the management structure, communication channels, advantages and disadvantages of each. Summarize the characteristics of each in a table.

   ii. You have been appointed project manager of a project which is estimated to take three years and it is estimated that the final product will consist of 2 million source lines of code. Assuming a productivity rate of 10 source lines of completed code per day per programmer, estimate the number of programming teams needed, show how you would structure them and estimate the amount of administrative staff necessary.

   iii. Which of Mantei's three teams corresponds to the structure of your team during the Lift and Car Rally assignments? What characteristic(s) of this team type did your team obviously exhibit?

Section B

5. Using finite state techniques, design a lexical analyser for processing 16-bit integer constants described by the regular expression ((+|–)? ([0-9]+)) | ([0-9a-fA-F]+"@"16).

   A complete set of test inputs designed to visit all non-error entries in the transition table should be included with the design description.

6. Consider the following arithmetic translation grammar where **Const** is a lexical token representing a numeric constant and the symbol ↑ is the exponentiation operator:

```
<S>  →  <E> {Result}
<E>  →  <E> + <T>
<E>  →  <T>
<T>  →  <T> * <F>
<T>  →  <F>
<F>  →  <P> ↑ <F>
<F>  →  <P>
<P>  →  ( <E> )
<P>  →  Const
```

   By eliminating left recursion etc, find an equivalent LL(1) grammar that generates the same language (sequences of terminal symbols), and add attributes and associated attribute evaluation rules so that the action symbol **{Result}** will inherit the value of the arithmetic expression generated by <E>. Show how to compute the selection sets for the productions in the grammar and draw a fully decorated derivation-tree for the expression 2↑3↑2.

7. i. Explain clearly the relationship between the first, follow and select sets in the context of LL(1) parsing.

   ii. Describe in detail the contents of a run-time activation record, and show by example the differences between the static and dynamic link fields in such a record.

8. i. Design an L-attributed translation grammar for a loop statement of the general form:

```
<statement> → for <control-variable> := <initial-value>
              to <final-value> do <statement>
```

Outline the information represented by the attributes and explain the function of the action symbols. Particular care should be taken to ensure that the resulting object code will function correctly for all possible termination values of the control variable.

ii. State the rules for handling attributes in a recursive descent parser.