UNIVERSITY OF DUBLIN

TRINITY COLLEGE

CS3BA71

Faculty of Engineering and Systems Sciences

School of Engineering

BA (Mod) Computer Science
Junior Sophister Examination

Trinity Term, 2003

3BA7 — Software Engineering and Compiler Design

Tuesday 20 May, 2003 Samuel Beckett Room 9:30 - 12:30

Dr JA Redmond, Dr DM Abrahamson

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

Section A

1. Write brief notes on four of the following topics:

   i. Design walkthroughs.

   ii. CRC cards.

   iii. Rapid Incremental Application Development (RIAD).

   iv. The *tangibility effect* in the context of requirements acquisition.

   v. The waterfall software lifecycle.

2. i. Write a note on the Level-5 Object Knowledge-based System describing its major components.

   ii. Give its advantages and disadvantages.

   iii. Briefly discuss Level-5 Object as a vehicle for implementing Object-oriented specifications (such as CRC with RDD) with particular reference to the projects which you implemented. Comment on how the design, programming and implementation task was divided among the team members and how this actually worked out.

   iv. What features are desirable in a future version of Level-5 Object and why?

3. 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.

4. i. Write a note on the Rational Unified Process (RUP).

   ii. Write a brief note on OPEN (Object-oriented Process, Environment and Notation) comparing and contrasting it with RUP.

   iii. Write a very brief note on SAP.

Section B

5. Using finite state techniques, design a lexical analyser for processing decimal and hexadecimal constants described by the regular expression ([0-9a-fA-F]+ [hH]?)

   Note: A complete set of test inputs designed to visit all non-error entries in the transition table should be included with the design description. [20 Marks]

6. Consider the following translation grammar for arithmetic expressions with starting non-terminal symbol <E>:

```
<E> → <E> + <T> {add}
<E> → <T>
<T> → <T> * <F> {mult}
<T> → <F>
<F> → <P> ↑ <F> {power}
<F> → <P>
<P> → ( <E> )
<P> → Ident
```

   where **Ident** is a lexical token representing a scalar variable, the ↑ symbol is the exponentiation operator, and the action symbols represent calls to the code generator.

   By eliminating left recursion, find an equivalent LL(1) grammar that generates the same language (sequences of terminal symbols). Compute the selection set for each production in the grammar, and finally add attributes and attribute evaluation rules so that the starting non-terminal has a synthesized attribute describing the value of the arithmetic expression generated by the grammar. [20 Marks]

7. i. In relation to context-free grammars, define the terms **ambiguity**, **prefix-property** and **selection set**, and show by example the difference between left-factoring and the elimination of left recursion. [10 Marks]

   ii. Describe the structure of the procedure **skip_to** and explain its use in global error recovery during recursive descent parsing. [10 Marks]

8. i. Design an L-attributed translation for a Java conditional assignment of the form:

```
<conditional assignment> → <variable> = <expression1> ?
                           <expression2> : <expression3>
```

Outline the information represented by the attributes and explain the function of the action symbols. [10 Marks]

ii. Demonstrate how a well-structured symbol table can cater for the overloading of identifiers in a block structured language. [10 Marks]