UNIVERSITY OF DUBLIN

TRINITY COLLEGE

CS3BA71

Faculty of Engineering and Systems Sciences

School of Engineering

BA (Mod) Computer Science
Junior Sophister Examination

Trinity Term, 2004

3BA7 — Software Engineering and Compiler Design

Tuesday 18 May, 2004 Goldsmith Hall 9:30 – 12:30

Dr R Meier, Dr DM Abrahamson

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

Section A (Software Engineering)

1. a) A number of practices that can be used as part of a software lifecycle model have been identified in the context of the extreme programming model. Describe what you understand by the practises below and discuss their trade-offs:

   i. Pair programming.
   ii. Stand-up meetings.
   iii. CRC cards. [5 marks each]

   b) Describe the project team approach that has been identified as naturally suited for the extreme programming model and outline why it is naturally suited. Discuss the approach. [5 marks]

2. Consider the following fragment of a requirements document for an elevator system:

A product is to be installed to control lifts in a building with n floors. The problem concerns the logic required to move lifts between floors according to the following constraints:

- Each lift has a set of n buttons, one for each floor. These illuminate when pressed and cause the lift to visit the corresponding floor. The illumination is cancelled when the lift visits the corresponding floor and the current floor is indicated.
- Each floor, except the ground floor and the top floor has two buttons, one to request an up-lift and one to request a down-lift. These buttons illuminate when pressed. The illumination is cancelled when a lift visits the floor and then moves in the desired direction.
- A lift that has no requests remains at its current location with its doors closed.

a) Generate the following UML diagrams:
   i. A use case diagram for the user's main actions.
   ii. A sequence diagram for users pressing lift buttons.
   iii. A domain-level class diagram for the key objects in the system and their relationships. [5 marks each]

b) There are several aspects of these requirements that are ambiguous and would need to be clarified. Identify some of these ambiguities and briefly discuss the implications of ambiguities in requirements. [5 marks]

3. a) What do you understand of the following concepts:
   i. Top-down design.
   ii. Program validation.
   iii. Unit testing.
   iv. Version management.
   v. Debugging. [3 marks each]

b) Valuable data tend to live longer than the applications that generated such data. Describe the properties of binary and of text-based data formats and discuss the trade-offs involved in choosing between these formats. [5 marks]

4. a) Suppose you are the manager of a project that has 8 tasks with the following estimated duration and dependencies:

Task	Duration (person-months)	Precedents
A	3	
B	4	
C	2	B
D	5	B
E	6	A
F	2	C,G
G	8	
H	3	D,E

Draw a Gantt chart for your project outlining the constraints of the individual activities. [5 marks]

b) Describe two different types of software contract, including the main items mandated by those software contracts. Describe the different payment models and penalty clauses that can be included. [8 marks]

c) According to Schach, two thirds of the costs associated with the traditional software life cycle are incurred during the maintenance phase. Describe some specific actions that can be undertaken during earlier life cycle phases in order to reduce these maintenance costs. [7 marks]

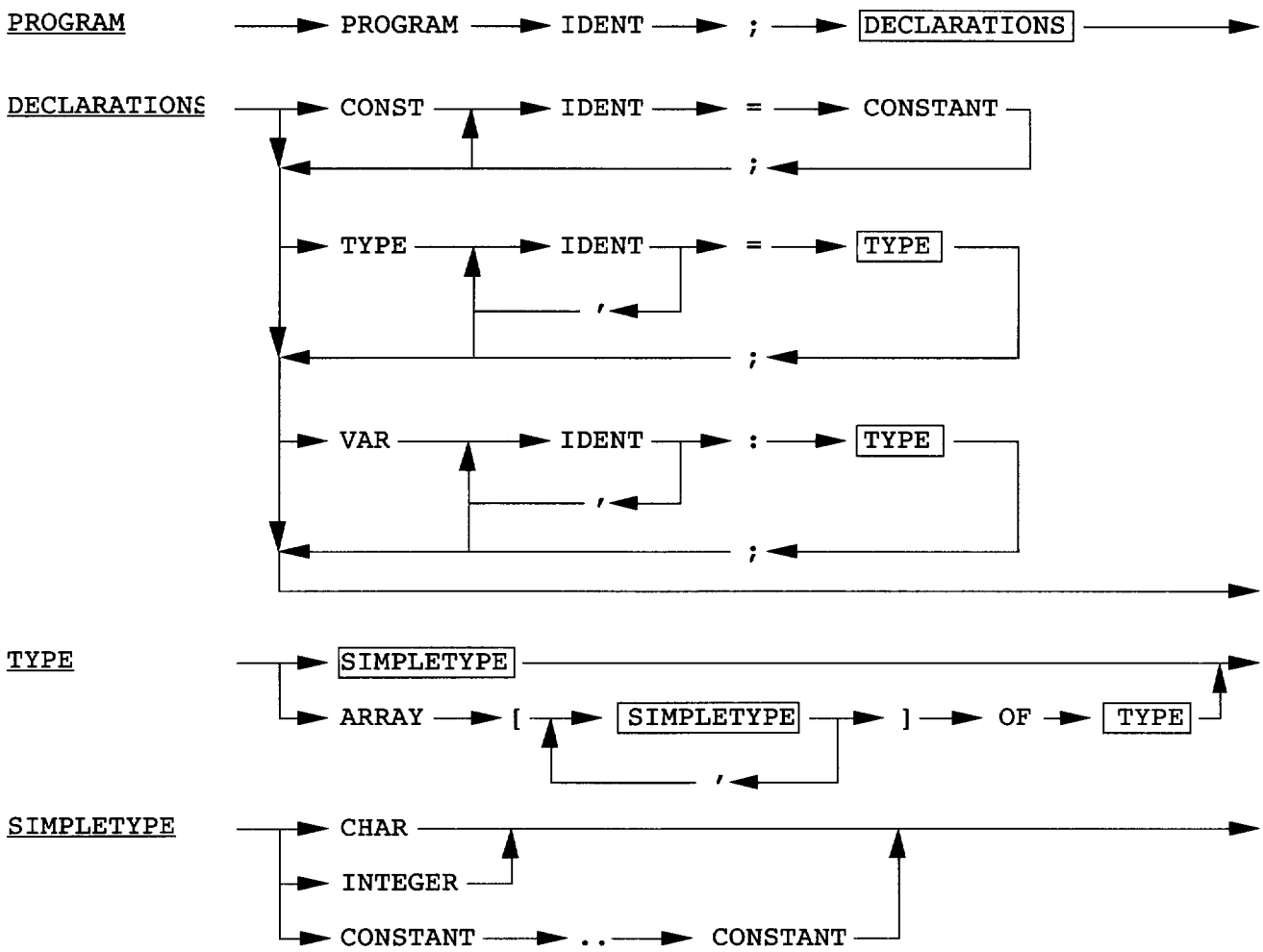
Section B (Compiler Design)

5. Using finite state techniques, design a lexical analyser to process integer constants in the range $-2^{15} .. 2^{15}-1$.

Note: i. Great care should be taken to avoid arithmetic overflow when processing values around either end of the permissible range.

ii. A comprehensive 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. Design an L-attributed translation grammar for the declarative section of a programming language as defined by the following set of syntax diagrams:



Describe in detail the information represented by the attributes and the function of the action symbols, outline the contents of the symbol table and demonstrate the actions that would be performed while parsing the declarations:

```
PROGRAM TEST;
CONST
   MAX = 4096;
TYPE
   ROWS = 1..16;
   COLUMNS = 0..255;
VAR
   I,J: INTEGER;
   TABLE: ARRAY [ROWS,COLUMNS] OF CHAR;
```

[20 Marks]

7. i. In relation to translation grammars, show by example the differences between inherited and synthesized attributes of non-terminal symbols. How are these two different types of attribute handled in a recursive descent parser? [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. By adding attributes and action symbols to the following prefix grammar, design an interpreter that will assign the value of an expression to a list of identifiers. Note, in executing a statement described by the grammar, the expression is to be evaluated first and the assignments are then to be performed from right to left.

```
<assignment statement>  -->  ident = <rest of statement>
<rest of statement>     -->  ident = <rest of statement>
<rest of statement>     -->  <expression>
<expression>            -->  + <expression> <expression>
<expression>            -->  * <expression> <expression>
<expression>            -->  ident
```

   Draw the derivation tree for the expression

```
i = j = x + y * z
```

   and show how to construct a recursive-descent parser for this grammar. [20 Marks]