Question 1 [25 Marks]

This question is based on the material of Units 9 and 10.

(i) Enter the OU Prolog system, and edit the file party by adding the following clause to those clauses already in the file:

\[ \text{newinvite}(X) :\text{- enjoytalking}(X, Z), \text{atparty}(X). \]

Exit from the Editor, and use the listing directive to view the clauses in the database.

(a) Describe, in your own words, using conventional English, the meaning of the new relation newinvite.

[1 mark]

(b) Obtain a printer listing of the database. Copy onto this listing the query

\[ ?\text{- newinvite}(\text{Guest}). \]

Indicate the following on the listing, and send it to your tutor:

a property relation; a goal; a condition; and a constant which is not an atom.

If any of these do not exist; say so explicitly on the listing.

[4 marks]

(c) If the query given in part (b) is input to our Prolog system, when would the first instantiation take place, and what would that instantiation be?

[3 marks]

(d) Submit a printer listing of the result of executing the query given in (b) using the trace facility. The trace should also show the result of attempting to resatisfy the query. Indicate on the listing all instances where backtracking occurs.

[4 marks]

(e) Draw a box diagram, similar to that shown in Figure 8.4 of Unit 10, showing how the following query is satisfied.

\[ ?\text{- newinvite}(<\text{name}>). \]

[5 marks]

(ii) The biological database is to be amended so that it can respond to the query

\[ ?\text{- special_form} \]

which repeatedly asks the user to input the name of a class (e.g., organism, bovidae, etc.) and outputs the more
specialised forms (if any) of that class. The specialised forms are given by instances of the special relationship. The input end is to be used to terminate the session and produce the message Session completed. The dialogue might proceed thus.

Enter the name of a class:
horse.
pony is a special form of horse
arabian is a special form of horse
appaloosa is a special form of horse

Enter the name of a class:
.
.
.

Enter the name of a class:
End.
Session completed.

Submit the following:

(a) a listing of the special_form procedure (including any subsidiary procedures); [5 marks]

(b) a printout showing the results from the query

|? - special_form.

When the following sequence is entered:

cattle, insect, organism, end. [3 marks]

**Question 2 [25 Marks]**

*This question is based on Units 9 and 10, and uses an example from Unit 5.*

(i) Remind yourself of the Towers of Hanoi problem described in Example 14.5 of Unit 5. In this question you are asked to write a solution to the problem using Prolog. The program towers is to output the sequences of moves, using the algorithm quoted on page 45 of Unit 5.

Move N discs from the 'left' pole to the 'right', using the 'centre' pole as an auxiliary, by completing the following:

\[
towers(n) :- move(N, left, right, centre). \\
moves( ) :- /* rule 1 */ \\
moves( ) :- /* rule 2 */
\]

Rule 1 is to be used to implement step 1 of the algorithm to move a disc from A to C and to output the move. Rule 2 is to be used to implement step 2 of the algorithm by moving N (N > 1) discs from pole A to pole C using pole B.

Thus the output from towers(1) would be:

Move disc from left pole to right pole
Submit the following:

(a) a listing of your completed towers program; [8 marks]

(b) a printout showing the results from the queries towers(2) and towers(3). [3 marks]

(ii) Consider the following specification.

SETS

L the set of lists
N the set of positive integers and zero

SYNTAX

CARDINALITY: L → N

SEMANTICS

If a is a member of the set L, n is a member of the set N, and the underlying model is that of a list,

pre-CARDINALITY(a) ::= true
post-CARDINALITY(a; n) ::= 
  IF isemptylist(a) 
  THEN 
  n = 0 
  ELSE 
  n = 1 + CARDINALITY(trailer(a))

(a) Write down a Prolog procedure for a relation, cardinality, which implements the specification given above. [4 marks]

(b) Submit a listing which shows the solution that your procedure in (a) gives to the following queries:

|? - cardinality([1, 2, 3, 4, 5], X).
|? - cardinality([7, 3, 5], 4).
|? - cardinality([sara, pete, jane], X). [3 marks]

(iii) The following Pascal program (on the main part is shown) uses an implementation of the CARDINALITY operation specified in (ii). The function cardinality is of type integer, and getlist obtains q, a list of integers.

begin {main}
getlist(q);
writeln('Cardinality of list is ', cardinality(q))
end.

(a) Which of the queries given in (ii) (b), if any, can the Pascal program deal with? Give reasons. [3 marks]

(b) Draw conclusions from (a) to describe the differences between parameters in Prolog and Pascal. [3 marks]
Question 3  [25 Marks]

This question is based on the material of Units 11.

Here is a BNF definition of a grammar with <expression> as the start symbol.

```plaintext
<expression> ::= <statement>
<statement> ::= <identifier> | <statement> <operator> <statement> | <negator> <statement>
<operator> ::= and | or
<negator> ::= not
<identifier> ::= a | b | c
```

(i) (a) Write down the derivation of the <expression> a or not b and c. Construct a parse tree to show the derivation. [5 marks]

(b) Show that the language is ambiguous, by finding an alternative derivation. Construct the corresponding parse tree. [5 marks]

(c) Use the trees constructed in (a) and (b) to evaluate the <expression> a or not b and c for a = false, b = true and c = false in each case. [4 marks]

(ii) Here is a revised version of the grammar.

```plaintext
<expression> ::= <statement>
<statement> ::= <identifier> | <statement> <operator> <identifier>
<operator> ::= and | or
<identifier> ::= a | b | c | not <identifier>
```

(a) Show that the <expression> a or not b and c is no longer ambiguous. [4 marks]

(b) In the revised grammar, what is the relative precedence of the operators and, or and not? Give reasons for your answer. [3 marks]

(c) Is the revised grammar an LL(1) grammar? Give a reason for you answer. [4 marks]
Question 4 [25 Marks]

This question is based on Unit 12. You are advised to complete Practical Exercise 9.1 before attempting it.

A `<while>` statement is to be added to SL. This new statement is to be identified using the `%` symbol, and should be capable of repeating precisely two statements while the condition (an `<expression>`) is true. Like the `<loop>` statement, the `<while>` statement should be written on one line. The syntax of `<while>` is

```latex
<while> ::= % <expression>; <statement>; <statement>
```

(i) Write a program in SL. Using the `<while>` statement to read in a series of numbers from the keyboard. The input is terminated by a number less than one. The output is the largest of the numbers. [4 marks]

(ii) Define the rules for the translation of the `<while>` statement into IC. Write your answers in the style of Figure 9.5. You may omit the syntax, but include the other headings. [6 marks]

(iii) Save a copy of SLIC3 and SLIC3B. In the editor, amend SLIC3B to implement the `<while>` statement according to the rules devised in (ii). Submit the following:

(a) a partial listing of SLIC3B, showing the changes from SLIC3; [9 marks]

(b) the output from SLIC3B using the SL program from (i) as input. Show the symbol table, the program store and the output when the sequence 12, 9, 2, 15, 7, 0 is entered from the keyboard. [6 marks]