1. (a) Parsers which have been produced by using the Yacc, Bison or CUP parser generators maintain a semantic stack while parsing. Describe the operation of such parsers as they parse input tokens and show how they would manipulate the stack as they process the input expression `1 + 3 * 3`.

[10 marks]

(b) A modification to the Tiger language is being considered whereby conditional statements can be chained together using the newly introduced keyword `elsif`, but the statement is otherwise unchanged. Thus the following examples would all be legal.

```
if x < 0
then print "neg."

if x < 0
then print "neg."
else print "non-neg."

if x < 0
then print "neg."
elsif x = 0
   then print "zero"
   else print "pos."
```

```
if x < 0
then print "neg."
elsif x = 0
   then print "zero"
   elsif x = 1
      then print "one"
      else print "pos."
```

(i) Give a grammar for this fragment of the language in the form of the input to the Yacc/Bison parser generator *or* in the form of the input to the CUP parser generator. You may omit the semantic actions for construction of the abstract syntax representation of the input.

[10 marks]

(ii) It has been suggested that this modification will correct the familiar "dangling `else`" problem for this grammar. Describe the problem carefully and then explain whether or not this modification will help.

[5 marks]

2. (a) The *semantic analysis* phase of compilation requires well-designed and efficient symbol table data structures in which to store information which is derived from declarations. Give a description of the interface to such a symbol table in the form of *either* a C header file *or* a Java package interface.

[8 marks]

(b) A compiler for the Tiger language should make use of both a *type environment* and a *value environment*. Using an example program, explain why two environments would be needed.

[5 marks]

(c) What is the information which would be recorded in the value environment for a Tiger language function?

[2 marks]

(d) Describe informally the type-checking which is applied to a Tiger language function of the following form.

`function f(a: ta, b: tb) : rt =` *body*

[5 marks]

(e) Is the following program fragment legal or illegal in the Tiger language? Justify your answer.

```
let type a = {x: int, y: int}
    type b = {x: int, y: int}
    val i : a := b { x = 1, y = 2 }
in  i
end
```

[5 marks]

3. (a) Certain optimisations which can be applied to intermediate languages used in compilation are only valid when expressions *commute*. Explain this term, using examples to illustrate your answer.

[5 marks]

(b) A compiler cannot always determine whether or not expressions commute, in which case it *conservatively estimates* the decision. Using the `Tree` intermediate language from the Tiger compiler give examples where it is necessary to conservatively estimate and examples where it is not.

[5 marks]

(c) The `Tree` language contains the following instruction forms.

BINOP (op, e_1, e_2) The application of operator op to e_1 and e_2, evaluated in that order.

CALL (f, l) The application of function f to argument list l, evaluated in that order.

ESEQ (s, e) The statement s is evaluated for its side effect then e for its result.

MOVE (TEMP $t, e)$ Evaluate e and move it to temporary t.

CJUMP (op, e_1, e_2, t, f) Evaluate e_1, then e_2. Compare the results using the relational operator op. If the result is `true`, jump to t; otherwise jump to f.

SEQ (s_1, s_2) The statement s_1 followed by s_2.

Supply identities for the following `Tree` language statements which could be used to optimize their evaluation. In each case where commuting parts of the statements would make a difference to the outcome give both identities which could be used.

(i) BINOP $(op, \mathrm{ESEQ}(s, e_1), e_2)$

(ii) BINOP $(op, e_1, \mathrm{ESEQ}(s, e_2))$

(iii) CJUMP $(op, \mathrm{ESEQ}(s, e_1), e_2, t, f)$

(iv) CJUMP $(op, e_1, \mathrm{ESEQ}(s, e_2), t, f)$

[10 marks]

(d) A term of the form BINOP $(op, \mathrm{CALL}(\ldots), \mathrm{CALL}(\ldots))$ requires special handling when optimising. Explain why, and give the corresponding re-arranged term.

[5 marks]