PRINCE OF SONGKLA UNIVERSITY

FACULTY OF ENGINEERING

Department of Computer Engineering

**Final Examination**: Semester 1 **Academic Year**: 2007-2008

**Date**: Monday, 8th October, 2007 **Time**: 13:30 – 16:30 **(3 hours)**

**Subject Number**: 240-340 **Room**: Hua Hun (Robot)

**Subject Title**: Compiler Structures

**Lecturer**: Aj. Andrew Davison

---

**Exam Duration**: 3 hours

**This paper has 3 pages.**

**Authorized Materials**:

- Writing instruments (e.g. pens, pencils).
- Books (e.g. dictionaries) and calculators are **not** permitted.

**Instructions to Students:**

- *Answer questions in English.* Perfect English is **not** required.
- Attempt all questions.
- Write your answers in an answer book.
- Start your answer to each question on a new page
- Clearly number your answers.
- Any unreadable parts will be considered wrong.
- When writing programs, use good layout, and short comments; marks will not be deducted for minor syntax errors.
- The marks for each part of a question are given in brackets (...).

A more operation increases a component's value by 0.1, a less operation decreases it by 0.1

Initially the colour has (red, green, blue) component values of (0.5, 0.5, 0.5) before the operations in a colour sentence modify them.

d) Write a yacc grammar which implements your attribute grammar of part (c). Explain in words what data types you have defined.
*Note*: do **not** write a lex grammar. (20)

## Question 1 (80 minutes; 80 marks)

a) Use shift-reduce parsing to evaluate the string "adedefb" against the grammar:

S → a A b
A → d e A | f

S and A are non-terminals, and a, b, d, e, and f are terminals. S is the starting non-terminal. (10)

b) **Briefly** explain the following LR table generation techniques:

- LR(0) items (5)
- the closure() function (5)
- the goto() function (5)

c) Produce a LR parse table for the grammar in part (a) using the techniques of part (b). Show **all** your working. (40)

d) Evaluate the string "adedefb" using your LR parse table from part (c). (15)

## Question 2 (55 minutes; 55 marks)

a) Explain yacc by specifying the main sections of a typical yacc program. (10)

b) What is an attribute grammar? (10)

c) Define an attribute grammar for the context free grammar given below: (15)

Colour → Ops
Ops → Ops Op | ε
Op → (red | green | blue) (more | less)

Colour, Ops, and Op are non-terminals, while `red`, `green`, `blue`, `more`, and `less` are terminals. Colour is the starting non-terminal.

A Colour sentence specifies a value for a colour in terms of its red, green, and blue components. Each component can range between 0.0 and 1.0.

A `more` operation increases a component's value by 0.1, a `less` operation decreases it by 0.1

Initially the colour has (red, green, blue) component values of (0.5, 0.5, 0.5) before the operations in a colour sentence modify them.

d) Write a yacc grammar which implements your attribute grammar of part (c). Explain in words what data types you have defined.
*Note*: do **not** write a lex grammar. (20)

**Question 3 on the next page.**

## Question 3 (45 minutes; 45 marks)

a) What is intermediate code? Give some **brief** examples of the different kinds. (15)

b) Describe the stack-based intermediate code used by the expressions language. Do **not** include any parser code, but include diagrams where possible. (10)

c) Translate the expressions program:

```
let w = 3 + x
w / 2
```

into intermediate code. Explain the translation in words. Do **not** include any parser code, but include diagrams where possible. (10)

d) Evaluate the intermediate code of part (c). Show all your working. Do **not** include any parser code, but include diagrams where possible. (10)

*--- End of Examination ---*