RYERSON POLYTECHNIC UNIVERSITY

DEPARTMENT OF MATH, PHYSICS, AND COMPUTER SCIENCE

CPS 710

FINAL EXAM

FALL 95

STUDENT ID: _____

INSTRUCTIONS

Please write your student ID on this page. Do not write it or your name on any other pages.

Please answer directly on this exam.

This exam has 4 questions and is worth 30% of the course mark.

NO AIDS ARE ALLOWED.

A - General Concepts	40
B - Error Management	15
C - Evaluation	20
D - Grammars	25

This exam has 7 pages including the front page

This exam has 7 pages including the front page

Part A - General Concepts - 40 marks

A1 (4 marks)

What is the primary difference between an interpreter and a compiler?

A2 (4 marks)

What is a token?

A3 (6 marks)

Draw a transition diagram which accepts the r.e: 01(1*|0*)10

A4 (4 marks)

What type of languages can be parsed with recursive descent parsers with 2 lookaheads?

A5 (6 marks)

Explain the difference between top-down parsing and bottom-up parsing

A6 (6 marks)

Explain the difference between lexical and syntax errors. Give an example of each type of error.

A7 (10 marks)

Assuming that you are writing an interpreter for a dynamically scoped language, you could organise your symbol table either as a single table or as multiple tables. Explain how these two options would work (with diagrams if you wish) and what are the pros and cons of each.

Part B - Error Management (15 marks)

For each error described below, state which type of error it is by marking the box or boxes which describes it. If you mark more than 1 box, explain the conditions under which the error would occur in each category.

function called with the wrong number of parameters	Syntax Error	Semantic error detected at compile time	Semantic error detected at run time
array index out of range	Syntax Error	Semantic error detected at compile time	Semantic error detected at run time
break not inside a loop	Syntax Error	Semantic error detected at compile time	Semantic error detected at run time
missing semi-colon at the end of a statement	Syntax Error	Semantic error detected at compile time	Semantic error detected at run time
Division by 0	Syntax Error	Semantic error detected at compile time	Semantic error detected at run time

function called with parameters of the wrong type	Syntax Error	Semantic error detected at compile time	Semantic error detected at run time

Part C - Evaluation - 20 points - (this has short answers)

- A syntax tree node has the following structure:
  typedef struct node {
  int type;
  struct node *v1, *v2, *v3, *v4;
  } node;
- a FOR structure is organised as follows:
  - type = FOR
  - v1 points to a set of statements that initialise some variables
  - v2 points to a condition tree that must evaluate to TRUE for the FOR loop to continue
  - v3 points to a set of statements that are executed inside the loop (if v2 evaluates to TRUE)
  - v4 points to an expression evaluated at the end of the FOR loop. The value of this expression will be the value of the whole FOR loop.
- You are writing int eval(node *tree), a function which evaluates a syntax tree. Eval returns an int which is either the value of the syntax tree, or NOVALUE when a syntax tree has no value. Condition trees evaluate to TRUE or FALSE. Eval is completely written except for the part which handles FOR structures.
- Write C code for the section of eval which handles FOR structures.
- You do not need to do any memory management in your code. You can assume that the FOR expression you are evaluating is error-free.

Part D - Grammars - 25 marks

In this question, non-terminals are in UPPER-CASE and terminals are underlined**.**

D1 (6 marks)

What is an **left-factored** grammar? Why would you want a grammar to be left-factored?

D2 (6 marks)

Left-factor the following set of productions fully. You may need to introduce new non-terminals.

E ∅ FUN | VAR | o

FUN ∅ ident (E)

VAR ∅ ident

VAR ∅ ident [E]

D3 (7 marks)

Remove the **left recursion** from the following set of productions.

C ∅ C or T | C and T | T

T ∅ not C | t | f

D4 (6 marks)

Why is the following set of productions **ambiguous**?

DISPLAY ∅ display (EXPRS)

EXPRS ∅ EXPR $EXPRS^0$

EXPR ∅ TERM | TERM + TERM | TERM - TERM

TERM ∅ - INTEGER | INTEGER

INTEGER ∅ { 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 }$^+$