Homework 4 – LL Parsing

Due Date: Thursday, November 3, 2005, 2:00

Your Name: _____

Problem 1 (5 points) Programs can contain errors at four different levels (and the compiler can catch errors at only three of these). What are these different four kinds of errors?

_____

_____

_____

_____

Problem 2 (5 points) Consider the following grammar.

S → D C | **f** C
C → C **b** | **a** | **b**
D → **g** D | **g e**

Which symbols are terminals? _____

Which symbols are nonterminals? _____

Problem 3 (5 points) What is the relationship between the set of regular languages and the set of context-free languages?

_____

_____

Problem 4 (2 points)

Does a top-down parser find a left-derivation or a right-derivation? _____

Does a bottom-up parser find a left-derivation or a right-derivation? _____

Problem 5 (5 points) A predictive parser is a top-down parser that never does what?

_____

Problem 6 (15 points, 5 each) Consider the following grammar.

S → **(** L **)** | **a**

L → S **,** L | S

Draw parse trees for the following three sentences:

Problem 6-a: **(a , a)**

Problem 6-b: **(a , (a , a))**

Problem 6-c: **((a, a , a) , (a))**

Please *STAPLE* a separate sheet of paper to your homework. Be sure to label your answer by writing the problem number clearly!

Problem 7 (15 points, 5 each) Show a leftmost derivation for each.

(a , a)

(a , (a , a))

((a, a , a) , (a))

Problem 8 (15 points, 5 each) Show a rightmost derivation for each.

(a , a)

(a , (a , a))

((a, a , a) , (a))

Problem 9 (5 points) Consider the following grammar.

S → **a** S **b** S
→ **b** S **a** S
→ ε

Show that this grammar is ambiguous by constructing two different parse trees for the sentence

a b a b

Please *STAPLE* a separate sheet of paper to your homework. Be sure to label your answer by writing the problem number clearly!

Problem 10 (5 points) Show that this grammar is ambiguous by constructing two different leftmost derivations for

a b a b

Problem 11 (5 points) Show that this grammar is ambiguous by constructing two different rightmost derivations for

a b a b

Problem 12 (5 points) Eliminate left-recursion in this grammar:

S → S **b** | **c**

Problem 13 (10 points) Eliminate left-recursion in this grammar:

S → S **a** | S **b** | S **c** | **d** | **e** | **f**

Problem 14 (10 points) Eliminate left-recursion in this grammar:

S → A | **b** B
A → A **c** | A **d** | B | **b** A
B → **f** A | **f b** | **g**

Problem 15 (5 points) Construct a transition diagram (in the style shown in the lecture notes) for this grammar:

S → **(** L **)** | **a**
L → S **,** L | S

Please *STAPLE* a separate sheet of paper to your homework. Be sure to label your answer by writing the problem number clearly!

Problem 16 (5 points) In the non-recursive predictive parsing algorithm, what two data structures do we use besides the input stream of tokens?

Problem 17 (8 points) In the non-recursive predictive parsing algorithm, what can go onto the stack?

What indexes the columns of the table?

What indexes the rows of the table?

What are the kinds of entries that can go into the table?

Problem 18 (5 points) In the algorithm to construct the table for the non-recursive predictive parser, if the grammar in not LL(1), what will this algorithm try to do?

Problem 19 (2 points) Can a non-recursive predictive parser handle all LL(1) grammars?

Problem 20 (5 points) How would you describe the set of languages that can be recognized by a predictive parser (either recursive or not)? _____

Problem 21 (10 points) Find FIRST(S) and FOLLOW(S) for this grammar:

S → **d** S **a** S
→ **c** S **b** S
→ ε

_____

_____

Problem 22 (10 points) Find the FIRST and FOLLOW sets for this grammar:

S → **a** S **b** B | **g** A
A → C B | **c** A **h**
B → **d** | **f** C | C
C → ε

FIRST (S) = _____

FIRST (A) = _____

FIRST (B) = _____

FIRST (C) = _____

FOLLOW (S) = _____

FOLLOW (A) = _____

FOLLOW (B) = _____

FOLLOW (C) = _____

Problem 23 (10 points, 2 each) Using the PCAT grammar for expressions (including the PCAT rules for associativity and precedence), add parentheses to make the order of evaluation clear. The first one is done for you as an example.

a + b + c + d

ANSWER: ((a + b) + c) + d

x * y / z * w

_____

x * y – a * z – b

_____

a + x < y + b < z * c

_____

a + b * c < d

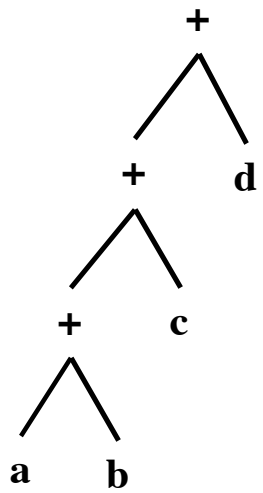
_____

a > b – c * d

_____

Problem 24 (24 points, 3 each) Using the PCAT grammar for expressions (including the PCAT rules for associativity and precedence), draw abstract syntax trees for each of the following expressions. Please *STAPLE* a separate sheet of paper to your homework. Be sure to label your answer by writing the problem number clearly! The first one is done for you as an example.

(24-Example) ((a + b) + c) + d



(24-a) a + (b + (c + d))

(24-b) a – b / c

(24-c) a / b – c

(24-d) a < b < c < d

(24-e) – a + – b

(24-f) not a or not b and not c

(24-g) – (a + b) * c – (d < e) * f

(24-h) x + y * – z * (w – v)

Problem 25 (15 points, 3 each) Using the PCAT rules for associativity and precedence, remove all redundant and unnecessary parentheses from the following expressions. The first one is done for you as an example.

((a + b) + c)+ d
ANSWER: a + b + c + d

(x+ (b * c))

((–(a-b))*c) + d

(x * y) * (z * w)

(x < y) < (z * a)

–(–(–(–(a))) – –(b – –(c – d)))
