

Programming Languages and Compilers

General Examination

Fall, 2004

Friday, October 22, 2004

There are four questions from programming languages and six questions from compilers. Answer **ALL** of questions from the compiler section and answer **three** of the four questions from the programming language section. Please only answer three questions from the programming language section; otherwise, we will choose the first three for grading.

COMPILERS

(Answer all Questions)

I. Define the following terms:

Handle
Regular Grammar
Syntax Scan
Token

II. Describe what is done during a two pass Assembler.

III. Describe the code generation operations FX and SX for optimizing code.

IV. Give an example of each of the following optimizations:

Invariant Loop Code
Common Subexpressions

V. Write a BNF grammar to describe the real numbers (for example: -3.14 or .691 etc).

VI. Give an example of left factoring. Give an example involving left recursion.

PROGRAMMING LANGUAGES

(Answer three of the four questions)

Answer each of the following:

a) Consider the program segment below. What values will be stored in the variable y and the array b after execution of this program segment if we are using:

- i) call by value
- ii) call by value-result
- iii) call by reference (or variable or address)
- iv) call by name
- v) call by result

```
void main () {
    int b[2], j, y;
    void s(int x)
    {
        b[0] = 4;
        j = 1;
        y = 2;
        x = x + 4;
    }

```

```
    j = 0;
    b[0] = 3;
    b[1] = 2;
    s(b[j]);
}
```

b) What would the output for the following program be under static scoping? What would it be under dynamic scoping?

```
void main () {
    int y;
    void p1 () {
        void p2 () {
            int y;
            y = 0;
            p1;

            while (y < 2) do
                print(y);
                y = y + 1;
                p2;
        }
        y = 0;
        p1;
    }
}
```

c) Consider the statement $a = b + 10$; What attributes are bound and at what times?

d) Consider the statements $a = 10$; $b = a + f(c)$; . Explain how side effects (in the function f) could make the addition operator in the second statement noncommutative.

b per Dr Kraft 10/22/04

II. Answer each of the following:

a) Why wouldn't it be a good idea to view C or Java as a functional programming language like Haskell?

b) Explain how a recursive function can be implemented.

c) Regarding type checking, what is strong type checking? Distinguish between name equivalence and structural equivalence when it comes to type checking. Try to illustrate your point with an example.

d) What are the basic design issues in adding concurrency features to a programming language? What does it mean to short circuit a Boolean expression?

e) What does it mean for a function or operator to be over-loaded? What does it mean for a function to be generic. Try to use a small example to illustrate your points.

f) What does it mean for an operator to be right associative? Give an example to illustrate your point.

g) If one wants to be certain that after execution of the statement

if (c > d) y = c;

the value of y and the value of c are the same, what is the weakest (least restrictive) condition that must necessarily hold before execution of that statement?

III. Given the following attribute grammar where Val, the set of positive integers, is a synthesized attribute for num and dig and Size, the set of positive integers, is an inherited attribute for str.:

```
<literal> ::= <num>@<str>
                Size (<str>) <- Val (<num>)
                Condition: Val (<num>) > 0
<num> ::= <dig>
                Val (<num>) <- Val (<dig>)
        <num> <dig>
                Val (<num>) <- 10 X Val (<num>) + Val (<dig>)
<str> ::= <char>
                Condition: Size (<str>) = 1
        | <str> <char>
                Size (<str>) <- Size (<str>) + 1
<char> ::= <dig> | A | B | C ..... | Z
<dig> ::= 0
                Val (<dig>) <- 0
        | 1
                Val (<dig>) <- 1
        | 9
                Val (<dig>) <- 9
```

- a) Describe the set of strings generated by the grammar?
- b) Can you modify the grammar so that Size is only an inherited attribute? If yes, modify the grammar. If no, explain why.
- c) Can you modify the grammar so that Size is only a synthesized attribute? If yes, modify the grammar. If no, explain why.

IV. Answer each of the following:

- a) Describe the basic concepts, including all key components, of denotational semantics?
- b) Describe the basic concepts, including all key components, of axiomatic semantics?

How do operational and denotational semantics differ? Be specific.
Compare and contrast attribute grammars and BNF grammars?
- d) Which of the three approaches to semantics is the least abstract?
- e) True or false - "The axiomatic approach is the most useful for a compiler writer". Justify your answer.