Problem 1 (7.5pts). Consider the following Pascal-like program:

```pascal
program x(n: integer)
x := n
end

procedure print_x
write_integer(x)
end

procedure first
set_x(1)
print_x
end

procedure second
set_x(2)
print_x
end

set_x(0)
first()
print_x
second()
print_x
```

What does the program print if the language uses static scoping? What does it print with dynamic scoping? Why?
Problem 2 (10pts). [From Scott] Consider the following pseudocode, assuming nested subroutines and static scope:

```plaintext
procedure main
    g : integer

    procedure B(a : integer)
        x : integer

        procedure A(n : integer)
            g := n

        procedure R(m : integer)
            write_integer(x)
            x /:= 2 -- integer division
            if x > 1
                R(m + 1)
            else
                A(m)

        -- body of B
        x := a * a
        R(1)

    -- body of main
    B(3)
    write_integer(g)
```

a) What does this program print?
b) Show the frames on the stack when A has just been called. For each frame, show the static and dynamic links.
c) Explain how A finds g.
Problem 3 (7.5pts). Below is the grammar for a simple declaration $D$ consisting of a basic type $T$ followed by a list $L$ of identifiers. $T$ can be $\text{int}$ or $\text{float}$.

$$
D \rightarrow TL \\
T \rightarrow \text{int} \\
T \rightarrow \text{float} \\
L \rightarrow L, \text{id} \\
L \rightarrow \text{id}
$$

Write an attribute grammar that enters the type of the identifier into the symbol table. You may assume there is a function $\text{addType}(\text{id}.\text{entry}, \text{type})$, where $\text{id}.\text{entry}$ points to the symbol table object for $\text{id}$, and $\text{type}$ is the type entered.

Problem 4 (12.5pts). [Modified from Aho, Sethi, Ullman and Scott] Let synthesized attribute $val$ give the accumulated value of the number generated by $S$ in the following grammar. The accumulated value should be the numerical value of the constant, not the string value (e.g., the value of constant 123.4 is accumulated as $1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 4 \times 10^{-1} = 123.4$, not as "1" + "2" + "3" + "." + "4" = "123.4").

$$
S \rightarrow D.D \\
D \rightarrow B M \\
M \rightarrow D \\
M \rightarrow \epsilon \\
B \rightarrow 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
$$

a) Use synthesized attributes only to determine $S.val$.

b) Determine $S.val$ with an attribute grammar in which the only synthesized attribute of $B$ is $c$, giving the contribution of the digit generated by $B$ to the final value. For example, the contribution of the first digit in 123.4 is 100, and the contribution of the last digit is 0.4.

Problem 5 (12.5pts). [Modified from Aho, Sethi, Ullman] Now, give an attribute grammar to translate infix expressions into infix expressions without redundant parentheses. For example, since + and * associate to the left, $((a*(b+c))*(d))$ can be rewritten as $a*(b+c)*d$.

Use the LR grammar as the underlying CFG:

$$
E \rightarrow E + T \\
E \rightarrow T \\
T \rightarrow T * F \\
T \rightarrow F \\
F \rightarrow \text{id} \\
F \rightarrow (E)
$$