<table>
<thead>
<tr>
<th>HW</th>
<th>Assigned</th>
<th>See Page</th>
<th>HW</th>
<th>Assigned</th>
<th>See Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>01/24</td>
<td>2</td>
<td>2</td>
<td>02/08</td>
<td>2, 3</td>
</tr>
<tr>
<td>3</td>
<td>02/14</td>
<td>4, 5</td>
<td>4</td>
<td>03/01</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>03/09</td>
<td>7</td>
<td>6</td>
<td>03/26</td>
<td>8, 9</td>
</tr>
<tr>
<td>6</td>
<td>03/25</td>
<td></td>
<td>7</td>
<td>03/30</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>04/06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>11/11/05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
HW 1 Assigned: 01/24 Due: 01/31 at recitation or in class

Read:
Problems: Book Exercise, pp 42-43 Hand in answers to Problems 4 and 6

HW 2-21/2 Assigned: 02/08 Due: 02/14 by in class or at recitation

Read: Chapt Text Chapt 3 and We covered Notes 1 and 2a. and 2b.
Problems: You should try all the Problems-Hand in answers for 1a, 2b 3, 4, and 6

1 Given grammars G1 and G2 in EBNF below Upper case letters are NTSs, All others including lower case letters are TSs

<table>
<thead>
<tr>
<th>G1:</th>
<th>G2:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X ---&gt; Y { ( +</td>
<td>- ) ( a</td>
</tr>
<tr>
<td>Y ---&gt; (a</td>
<td>b)</td>
</tr>
<tr>
<td>X is the start symbol</td>
<td>O ---&gt; 7</td>
</tr>
<tr>
<td></td>
<td>E is the start symbol</td>
</tr>
</tbody>
</table>

Give an equivalent CFG (Simple Productions only) for:
(a) G1:
(b) G2:

1. For G1
- Y ---> a
- Y ---> b
- Y ---> aX
- Y ---> bX
- X ---> + a X
- X ---> + b X
- X ---> - a X
- X ---> - b X
- X is the start symbol

2 Given CFG Precedence Grammar G3 a Precedence Expression Grammar in which <xxx>s are NTs, All others including lower case letters are TSs

<table>
<thead>
<tr>
<th>G3</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;op&gt; ---&gt; n</td>
</tr>
<tr>
<td>&lt;fac&gt; ---&gt; &lt;op&gt;</td>
</tr>
<tr>
<td>&lt;exp&gt; ---&gt; &lt;fac&gt;</td>
</tr>
<tr>
<td>&lt;exp&gt; is the start symbol</td>
</tr>
</tbody>
</table>

Show the Tree version of a parse for:
(a) i + n
(b) i + n * n
(c) (i - n) + i * n
3) In G4 Upper case letters are NTSs, All others including Lower case letters and are TSs

G4:
1 S → / | x=y | x!=y
2 I ----> if C then S
3 I ----> if C then S else S
4 C ---> (a>b)

S is the starting symbol,

4) Define a recursive function, in a C like language (with no whiles, fors, or repeats) that is equivalent to the following program for counting the number of digits in an integer

```c
int numbdigits (int x)
{
    if (t >= 10) { numbdigits(n+1, t/10) ; else return n; }
    return n;
}
```

5) Give a non-recursive C-like procedure definition using a while that produces the same output as definition (a) when called with sum(n,0). and also the same output as definition (b) with sum(n,0).

Definition (a) int sum(int n, int s)
{
    if (n > 0) sum(n -1, s+n) else return s; }

Definition (b): int sum(int n)
{
    if (n> 0) return n + sum(n -1) else return 0; }

6) A regular grammar (RG) cannot generate the language ai b i (a string of a’s followed by an equal number of b’s). However it can produce aa*bb*. The following incomplete RG is intended to do so. Complete the grammar by putting an NTS where the underlines are and adding any rule(s) you need

Q → aX
X → a__ | b__ | b

b Show an FSA for the same language.
Some Questions From F03 MidTerm On Material We Have Covered. More next week.

2) G2:

\[ A \rightarrow ( A | ( B \ B \rightarrow ) B | ) \]

A is the start NTS

a) Give an RE and an FSA for \( \text{L}(G2) \). [Use \( x^+ = xx^* \)]
b) A C-like program to implement that FSA for acceptance-not parse

1) A regular grammar (RG) cannot generate the language \( a^i b^i \) (a string of \( a \)'s followed by an equal number of \( b \)'s). However it can produce \( a a^* b b^* \). The following incomplete RG is intended to do so. Complete the grammar by putting an NTS where the underlines are and adding any rule(s) you need

\[
\begin{align*}
Q & \rightarrow aX \\
X & \rightarrow a_\_ | b_\_ | b
\end{align*}
\]

b) Show an FSA for the same language.

2) G2:

\[
\begin{align*}
A & \rightarrow ( A | ( B \\
B & \rightarrow ) B | ) \\
A & \text{ is the start NTS}
\end{align*}
\]

a) Give an RE and an FSA for \( \text{L}(G2) \). [Use \( x^+ = xx^* \)]
b) A C-like program to implement that FSA for acceptance-not parse

3) A CFG can be defined in which the same terminal string can be parsed two different ways (Ambiguity) (examples are shown in the Notes). Can such a definition be given for a RG? If so give such a (simple) definition
4) Design an Attribute Grammar that accepts strings of representing additions of digits, ex. 1+2+5+2 and 7+9+5 and outputs the sum-for the two examples 10 and 21. The CFG should be unambiguous.

X --> X+Z \( s(X) = s(X) + s(Y) \)

X --> Z \( s(X) = s(Z) \)

Z --> 1 \( s(Z) = 1 \)

Z --> 2

X --> aX \( s(X1) = s(X2) + 1 \)

X --> bX \( s(X1) = s(X2) - 1 \)

X --> a \( s(X) = 1 \)

X --> b \( s(X) = -1 \)

5) Design an Attribute Grammar that accepts any string of a_s and b_s in which the number of a_s is 1 more than the number of b_s. Use 1 Attribute, namely cnt

S --> X accept if cnt = 1

X --> aX \( cnt(X1) = cnt(X2) + 1 \)

X --> bX \( cnt(X1) = cnt(X2) - 1 \)

X --> a \( cnt(X) = 1 \)

X --> b \( cnt(X) = -1 \)

6) Program: if \((A > B)\) then \(X = A\) else \(X = B\);

Prove that after this Program runs \(X = \text{maximum}(A, B)\) by getting the appropriate precondition which is true by definition of maximum.
HW 4 Assigned 03/01 Due 03/06 in class or at recitation. 

Read:  Notes Material is also in Chapt 4 of Text) 

Problems: You should Hand-In answers for all the Problems-

1) Informal functional prefix notation with Scheme like built-ins. Write each in Scheme notation 

   a) \( g? \) \((a, T) \) where \( T \) is a list with two members, say \([x \ y]\), and \( a \) is an atom. 
      \( g?(a, T) \) returns “true” if \( a \) equals \( x \), otherwise returns “false” 
      \( g?(a, T) \) if( \( a == \) car(T) ) return #t 
      else return #f

   b) \( f2(a, L) \) where \( a \) is an atom and \( L \) is a list of pairs say, \([ [x_1 \ y_1] \ [x_2 \ y_2] \ ... \ [x_n \ y_n] ] \) 
      \( f2(a, L) \) returns \( y_j \) if \( a \) equals \( x_j \) and \( a != x_k \) for \( k < j \). /*if \( j \) is the lowest index in \( L \) with \( a = x_j \) 
      it is assumed that this is true one \( x_j \) */
      \( f2(a, L) \): if( q?( a, car(L) ) return car( car(L) ) 
      else return f2( a , cdr(L) )

   c) \( f3(L) \), where \( L \) is a list say \([ x_1 \ x_2 \ ... \ x_n ] \), \( n \) is odd 
      \( f3(L) \) returns \([ x_1 \ x_3 \ ... \ x_{n-2} \ x_n ] \), i.e., the 1st, 3rd, 5th etc member of \( L \). 
      \( f3(L) \): if ( cdr(L) == '() ) return car(L) 
      else return cons( car(L) , f3( cdr( cdr(L) ) ) )

2) For each of the following write the definition of the described function in Scheme 

   a) \( attach(a, b) \) where \( a \) and \( b \) are atoms 
      \( attach(a, b) \) returns the list \([a \ b] \)

   b) \( f4(L J) \), where \( L \) is a list say \([ x_1 \ x_2 \ ... \ x_m ] \), \( J \) is an integer 
      \( f4(L,J) \) returns \([ x_1 \ J+1 \ [x_2 \ J+2] \ ... \ [x_m \ J+m ] \] 
      \( f4(L J) \): if ( cdr(L) == '() ) return ( attach( car(L) , +(J,1)) ) 
      else return append( attach( car(L) , +(J,1)) , f4( cdr(L) , +(J, 1) ) )

3) To write a functional program in Scheme to find the shortest path in a graph, one needs to represent the graph using lists. Describe how this could be done.
HW 5 Assigned 03/09 Due 03/14 in class or at recitation.

Read: Notes Material is also in Chapt 4 of Text

Problems: You should Hand-In answers for all the Problems-

1 For the following Procedure

procedure F(A:copy-in real, C:copy-out real, D: copy-inout real)
    if(A > D) then {C = D - A;} else if (A > C ) { C = D + A;} else C = D;
end F

a) Assume it is correct show the Prefix and Suffix that would be added by the compiler.
b) But it does have a problem (Associated with Call Types). What is it?
c) Is it possible for the Compiler to catch such Problems?
d) If the last 2 formal parameters, C and D were by reference rather as above would the same problem as in b still exist?.

2 Show the code that would be produced for the following procedure

a) Refer to actual parameters as “ap-k”, Include prefix and Suffix.

    procedure funct(int Q copy-inout, int U reference, int V reference)
        int X;
        X = Q + 10;
        U = X;
        V = 5X;
        Q = V /U + Q;
    end funct

b) What, if anything is wrong with the following optimization of the Formal Body? Explain

    int X
    X = Q + 10
    U = X
    V = 5X
    Q = 5 + Q;

3 Complete the following:

In block structured language or, in fact, in virtually all languages with procedures which can be recursive one keeps track of the a sequence of Procedures calling each other in Runtime Stack. Each time a Procedure is called from within another Procedure an Activation Record is added on top of the Runtime Stack. When a local variable, say x, is assigned a value in the currently active Procedure, P, but x is not declared in P there are two major schemes for searching for which x to make the assignment to. One of the schemes is called Static Allocation. It is based on the languages’ static Block Structure. The other is called Dynamic Allocation and is based simply on looking for a declaration of x in the Procedure which called the currently active Procedure if x is not declared within the currently active Procedure.
1 How are blocks defined in C?

2 There are no parameters nor values returned in the following program. **SHOW** the evolution of the Run Time stack for this Program. As in the notes, each Activation Record should contain, the identity of the Procedure, the Dynamic and Static pointers, the return address and the value of local variables.

```
M: Main
    var x=5, y=10
    procedure A();
        var y;
        begin
            writeln (y)
        end
    procedure B();
        begin
            x=7;
            A()
            L2: writeln(x)
        end;
    begin
        -B();
        L1:
    end M
```


2. Give an equivalent C statement to the following Modula-2-like statement.
   ```
   type p = pointer to integer;
   ```
4 This is the layout for an array in Modula, which, as you see, starts at location 100 with an initial index of 2. Give the formula for \( A[J] \), where \( J \) ranges from 2 to 100. (This should have a term independent of \( J \) plus a term dependent on \( J \).)

\[
\text{var } A : \text{ array } [2 .. 100] \text{ of integer}
\]

\[
0 \quad 100
\]

\[
\begin{array}{ccccccc}
2 & 3 & \cdots & \cdots & \cdots & 100
\end{array}
\]

\[
2 \text{ Bytes}
\]

© M C Paull 2008