Ben Smith
Email: bxs566@case.edu
Course: CSDS 337 - Compiler Design
Instructor: Dr. Vipin Chaudhary

Homework - #1
ID: 3559750
Term: Spring 2024
Due Date: 5^{th} Feb, 2024

Number of hours delay for this Problem Set: 0
Cumulative number of hours delay so far: 0

I discussed this homework with: No one.

Problem 1
What language is generated by the following grammars?

a $S \longrightarrow 0S1|01$

b $S \longrightarrow \mathbf{a}S\mathbf{b}S|\mathbf{b}S\mathbf{a}S|\epsilon$

c $S \longrightarrow S+S|S*S|(S)|\mathbf{id}$

Solution: Your solutions go here

a All strings of length greater than or equal to four of an equal number of zeroes and ones, starting with all zeroes first and ending with the ones.

b All strings of an equal number of a's and b's.

c All mathematical expressions using id's with only addition, multiplication, and parentheses.

Problem 2
Which of the following grammars are ambiguous? If ambiguous, show parse trees to substantiate.

a $S \longrightarrow 0S1|01$

b $S \longrightarrow \mathbf{a}S\mathbf{b}S|\mathbf{b}S\mathbf{a}S|\epsilon$

c $S \longrightarrow S+S|S*S|(S)|\mathbf{id}$

Solution: Your solutions go here

a $S \longrightarrow 0S1|01$ is not ambiguous.

b $S \longrightarrow \mathbf{a}S\mathbf{b}S|\mathbf{b}S\mathbf{a}S|\epsilon$ is ambiguous.

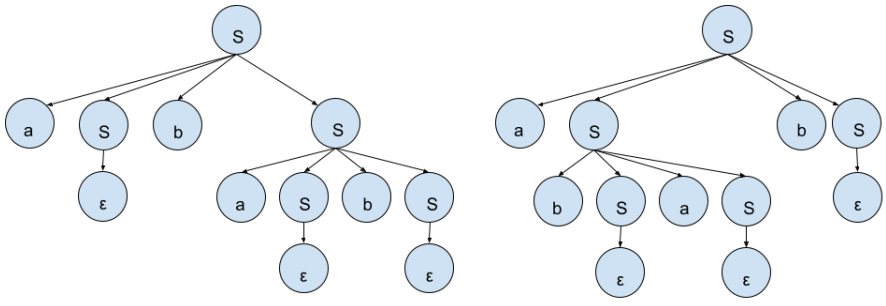

Figure 1: Two parse trees that generate the string "abab".

c $S \longrightarrow S + S|S * S|(S)|\mathbf{id}$ is ambiguous.

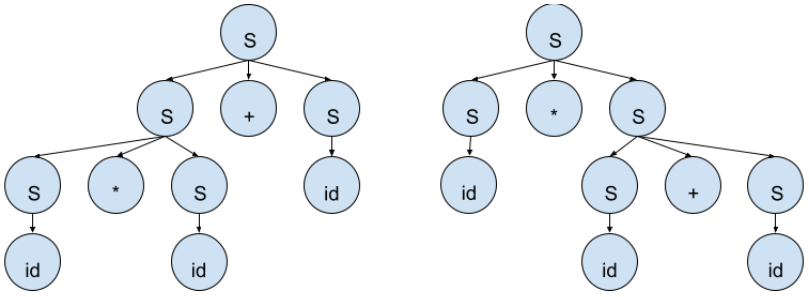

Figure 2: Two parse trees that generate the string "$\mathbf{id} * \mathbf{id} + \mathbf{id}$".

Problem 3

a For the predictive parser provided to you, execute the following program and submit the output.

```
{
    int i; int j; float[10][10] a;
    i = 0;
    while ( i < 10 ) {
            j = 0;
            while ( j < 10 ) {
                a[i][j][j] = 0;
                j = j+1;
            }
            i = i+1;
    }
    i = 0;
    while ( i < 10 ) {
        a[i][i] = 1;
        i = i+1;
    }
}
```

b For the predictive parser provided to you, execute the following program and submit the output.

```
{
    int i; int j; float[20][20] a;
    i = 0;
    while ( i < 20 ) {
            j = 0;
            while ( j < 20 ) {
                a[i][j] = i + j;
                j = j+1;
            }
            i = i+1;
    }
    i = 0;
```

```
        while ( i < 20 ) {
            a[i][i] = 1;
            i = i+1;
        }
    }
```

Solution:

a There was an error while parsing this code, which makes sense. At line 7, 3 array indices are used to get an element in a two-dimensional array. The output from the terminal is below:

```
Exception in thread "main" java.lang.ClassCastException: class symbols.
    Type cannot be cast to class symbols.Array
    (symbols.Type and symbols.Array are in unnamed module of loader 'app')
        at parser.Parser.offset(Parser.java:229)
        at parser.Parser.assign(Parser.java:124)
        at parser.Parser.stmt(Parser.java:110)
        at parser.Parser.stmts(Parser.java:62)
        at parser.Parser.block(Parser.java:30)
        at parser.Parser.stmt(Parser.java:107)
        at parser.Parser.stmt(Parser.java:87)
        at parser.Parser.stmts(Parser.java:62)
        at parser.Parser.stmts(Parser.java:62)
        at parser.Parser.block(Parser.java:30)
        at parser.Parser.stmt(Parser.java:107)
        at parser.Parser.stmt(Parser.java:87)
        at parser.Parser.stmts(Parser.java:62)
        at parser.Parser.stmts(Parser.java:62)
        at parser.Parser.block(Parser.java:30)
        at parser.Parser.program(Parser.java:23)
        at main.Main.main(Main.java:9)
```

b This example produced no errors and successfully generated intermediate code. There were no messages in the terminal. The output is below:

```
L1: i = 0
L3: iffalse i < 20 goto L4
L5: j = 0
L6: iffalse j < 20 goto L7
L8: t1 = i * 160
    t2 = j * 8
    t3 = t1 + t2
    t4 = i + j
    a [ t3 ] = t4
L9: j = j + 1
    goto L6
L7: i = i + 1
    goto L3
L4: i = 0
L10:    iffalse i < 20 goto L2
L11:    t5 = i * 160
    t6 = i * 8
    t7 = t5 + t6
    a [ t7 ] = 1
L12:    i = i + 1
    goto L10
L2:
```