

Introduction

The mad squirrels of Tralla are nuts about nuts. They live for only one thing: to accumulate a massive hoard of nutritious, delicious, life-sustaining, awe-inspiring, world-shaking, fortune-making magical hazelnuts. Tralla produces a limited supply of these nuts and some trees produce more than others. Of course, it would never do for more than one squirrel to share the same tree, so Tralla has something of a real estate problem. Squirrels with less productive trees desperately desire to move into trees with higher yields of nutritious, delicious, life-sustaining, awe-inspiring, world-shaking, fortune-making magical hazelnuts.

Since Tralla has no national currency, there is only one way for "poorer" squirrels to move from the slums and ghettos to the mansions of glory — physical violence. Dueling between Trallan squirrels is very simple and predictable. Larger squirrels always beat smaller squirrels and ties go to the defender. How do squirrels become large or small? In order to grow, a squirrel must dip into his hoard of nutritious, delicious, life-sustaining, awe-inspiring, world-shaking, fortune-making magical hazelnuts. For each nut he (only male squirrels are mad in Tralla, female squirrels are too sensible to participate in such nonsense. Besides -- someone has to keep some order on the island!) consumes, he grows one inch.

This means that squirrels spend their entire lives engaged in two activities: hoarding nuts and assaulting each other. Strangely enough, the squirrels are pathologically methodical in these two activities. Every squirrel has a well-defined strategy for accumulating more nuts than his neighbors from which he never deviates. This strategy can be expressed in MSBPL ("Mad Squirrel Battle Plan Language"), which has the following BNF grammar:

```

statements := statements statement | statement

statement := declaration | whileLoop | ifStatement | eatStatement |
            pushStatement | intAssign | treeAssign | dirAssign

declaration := type varName ";"

type := INT | TREE | DIRECTION

VarName := [A-z] [A-z0-9]*

whileLoop := WHILE boolExpr "{" statements "}"

ifStatement := IF boolExpr "{" statements "}" ELSE "{" statements "}"

eatStatement := EAT intExpr ";"

pushStatement := PUSH <dirExpr> ";"

intAssign := VarName "=" intExpr ";"

treeAssign := VarName "=" treeExpr ";"

dirAssign := VarName "=" dirExpr ";"

boolExpr := "(" boolExpr ")" | boolExpr "&&" boolExpr | boolExpr "||" boolExpr |
            intExpr "<" intExpr | intExpr ">" intExpr | intExpr "==" intExpr | dirExpr "=="
            dirExpr | treeExpr "==" treeExpr

intExpr := "(" intExpr ")" | [0-9]+ | VarName | treeExpr "[X]" | treeExpr "[Y]" |
            intExpr "+" intExpr | intExpr "-" intExpr | intExpr "*" intExpr |
            intExpr "/" intExpr | intExpr "%" intExpr | NUTS treeExpr

treeExpr := "(" treeExpr ")" | "[" intExpr "," intExpr "]" | VarName |
            TREE dirExpr | NULL

dirExpr := "(" dirExpr ")" | NORTH | SOUTH | WEST | EAST | HERE | VarName

```

Some notes on MSBPL

MSBPL has three variable types: integers (mad squirrels can't do fractions), trees (simple structures consisting of integer coordinates X and Y), and directions (NORTH, SOUTH, WEST, EAST, and HERE). It supports primitive looping (with the "while" statement) and an "if" statement.

The "NUTS" keyword can be used to obtain the number of nuts produced by a particular tree each day. Two special keywords, "EAT" and "PUSH", indicate that the squirrel should eat a certain number of hazelnuts or try to occupy one of the trees nearest to his current location. Once a squirrel has eaten or "pushed" the plan immediately terminates (like a "return" statement, kinda) and will begin from the top at the start of the next day. All variables are static global, so they are preserved from day to day.

Your Mission

This assignment has three parts:

A. On paper, write regular expressions for all the possible lexemes defined in this grammar.
(20 pts)

B. On paper, create an NFA for each of the regular expressions you used in part A.
(30 pts)

C. Create a scanner for MSBPL using flex.
(40 pts)

Your scanner should consume whitespace and print each token using the syntax "<id>" or the syntax "<id, value>" if there is a value associated with that token.

D. Glitter
(10 pts)

Glitter points are assigned for extending the project assignment in a creative way.

You will earn more points for ideas that are unique, creative, and clever. I will also factor in the amount of time I believe was necessary to implement your idea.

Some suggestions for ways to earn glitter points:

Write a partial MSBPL scanner in C++ that can process some portion of the language.

Extend the language by adding new keywords (for instance a BOMB keyword, that uses all of the squirrels nuts, but allows him to reduce one of his opponents to size 0).

Generate your NFAs using "graphviz" (a GPL'd graph generating program).

Submitting

Create a tarball of your project and upload it using the web form located at <http://narnia.homeunix.com/~robert/submit/>