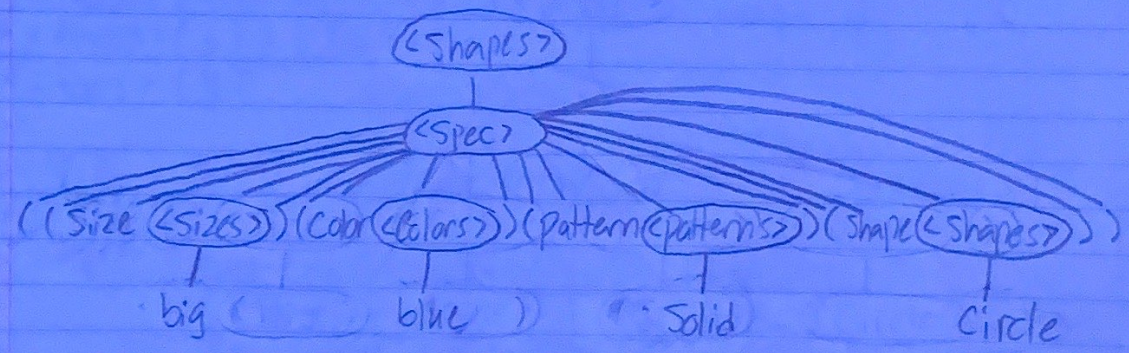
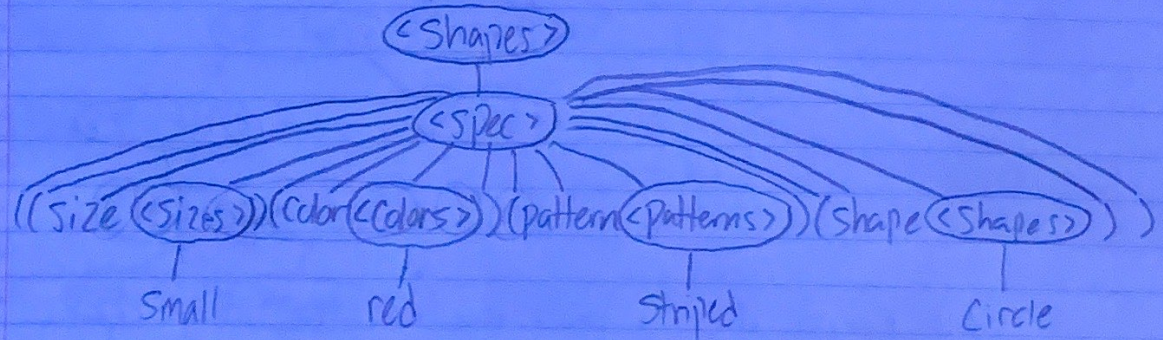
BNF Assignment

Problem 1 - Shapes

<Shapes> ::= <Spec>
<Spec> ::= ((Size <Sizes>)(Color <Colors>)(Pattern <patterns>)(Shape <Shapes>))
<Sizes> ::= Small | medium | big
<Colors> ::= blue | yellow | red
<Patterns> ::= Solid | dotted | Striped
<Shapes> ::= Circle | sCircle | square | triangle

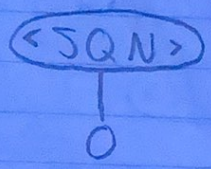




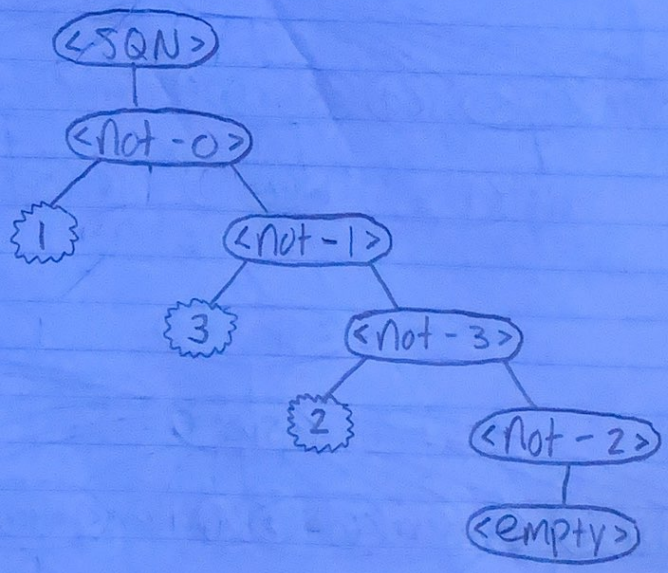
Problem 2 - SQN (Special Quaternary Numbers)

<SQN> ::= 0 | <not-0>
<not-0> ::= 1 <not-1> | 2 <not-2> | 3 <not-3> | <empty>
<not-1> ::= 0 <not-0> | 2 <not-2> | 3 <not-3> | <empty>
<not-2> ::= 0 <not-0> | 1 <not-1> | 3 <not-3> | <empty>
<not-3> ::= 0 <not-0> | 1 <not-1> | 2 <not-2> | <empty>

0



132

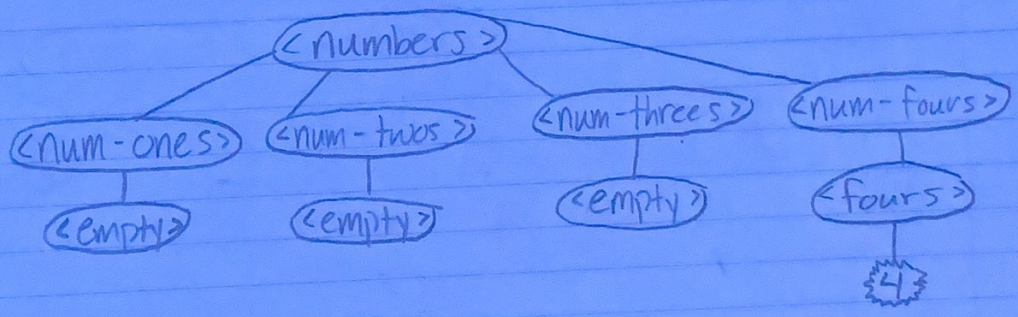


You cannot draw a parse tree for the string 1223 because this language is not defined for the same number twice in a row. <not-2> only allows for a 0, 1, 3, or nothing to come after a 2.

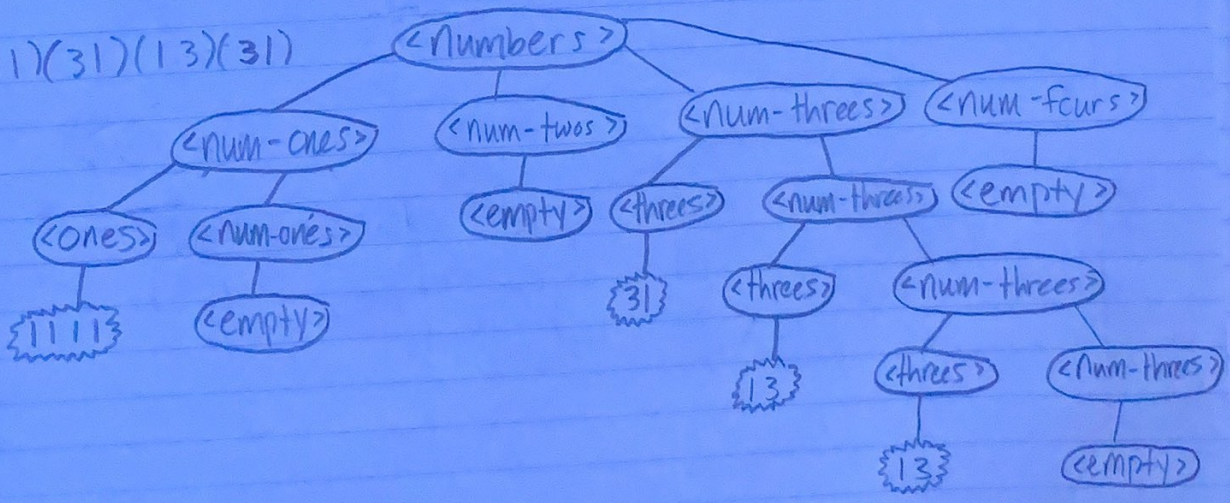
Problem 3 - Fours

<numbers> ::= <num-ones> <num-twos> <num-threes> <num-fours>
<num-ones> ::= <ones> <num-ones> | <empty>
<num-twos> ::= <twos> <num-twos> | <empty>
<num-threes> ::= <threes> <num-threes> | <empty>
<num-fours> ::= <fours> <num-fours> | <empty>
<ones> (1111)
<twos> (112) | (121) | (211)
<threes> (31) | (13)
<fours> (4)

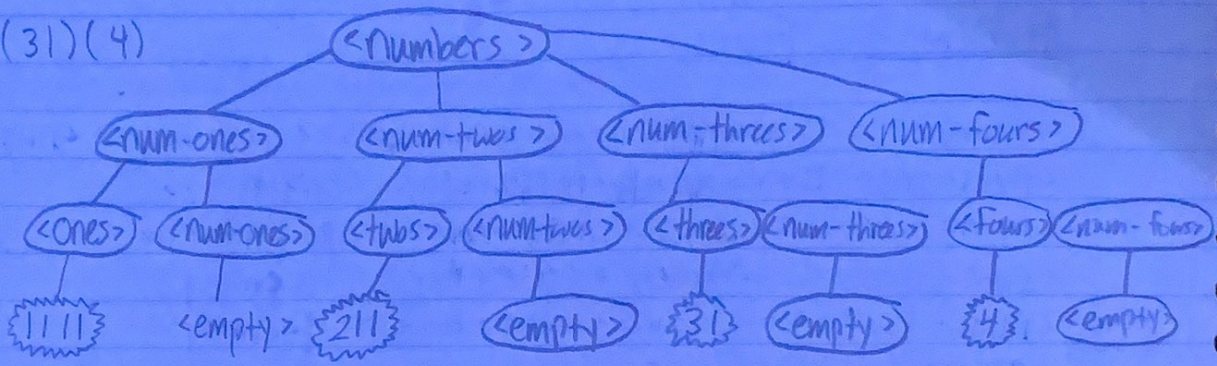
4



(1111)(31)(13)(31)

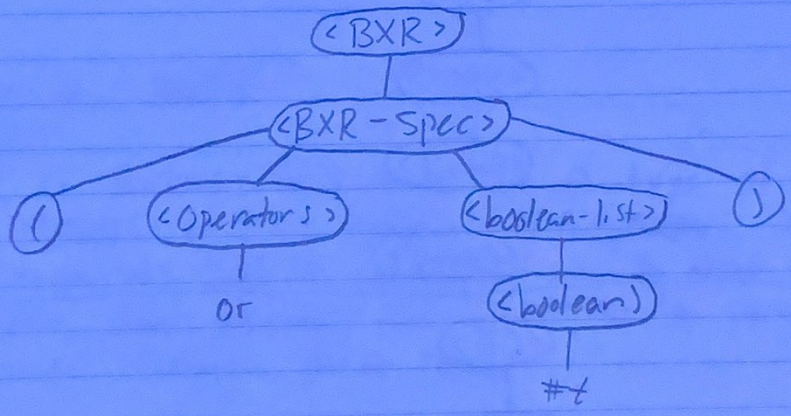


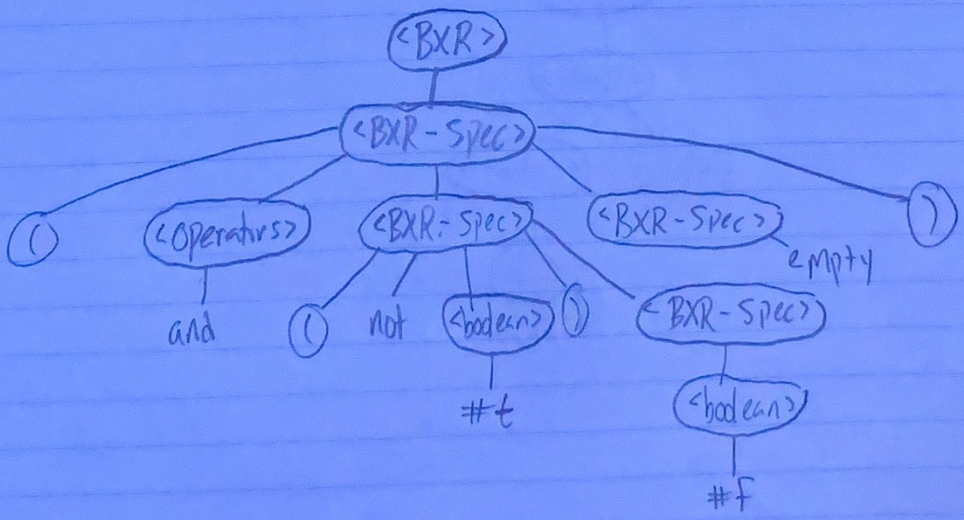
(1111)(211)(31)(4)


<numbers>
<num-ones>
<num-twos>
<num-threes>
<num-fours>
<ones>
<num-ones>
<twos>
<num-twos>
<threes>
<num-threes>
<fours>
<num-fours>
1111
<empty>
211
<empty>
31
<empty>
4
<empty>

Problem 4 - BXR

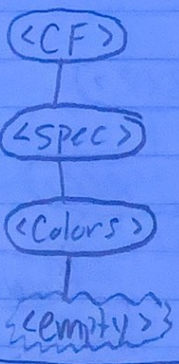
<BXR> ::= <empty> | <BXR-Spec>
<BXR-Spec> ::= (<Operators> <BXR-Spec>) <BXR-Spec> |
(<Operators> <boolean-list>) | (not <boolean>) <BXR-Spec> |
<boolean> | <empty>
<boolean-list> ::= <boolean> <boolean-list> | <boolean>
<Operators> ::= and | or
<boolean> ::= #t | #f

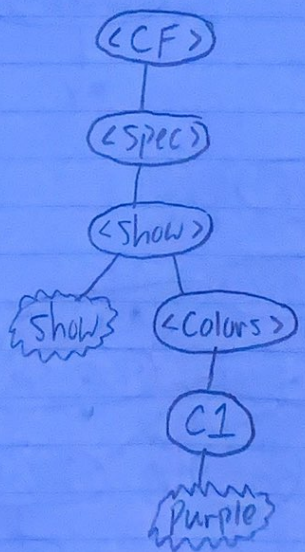


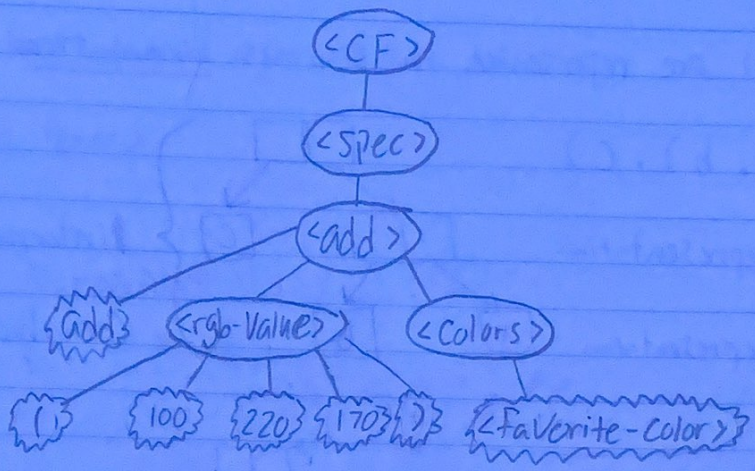


Problem 5 - CF

<Color-fun> ::= <spec>
<Spec> ::= <add> | <Colors> | <Show> | <describe>
<add> ::= add <rgb-value> <Colors> | add <Colors>
<Colors> ::= <C1> | <C2> | <C3> | <C4> | <favorite-Color> | Colors | <empty>
<Show> ::= Show <Colors>
<rgb-value> ::= (<rgb> <rgb> <rgb>) | (<rgb> <rgb> <rgb> <rgb>)







Problem 6 - BNF?

BNF, or Backus-Naur form, is a type of notation that includes a start symbol, tokens, terminals, and non-terminals. BNF is an easier way to break down the grammar that describes the syntax of a programming language through parse trees.