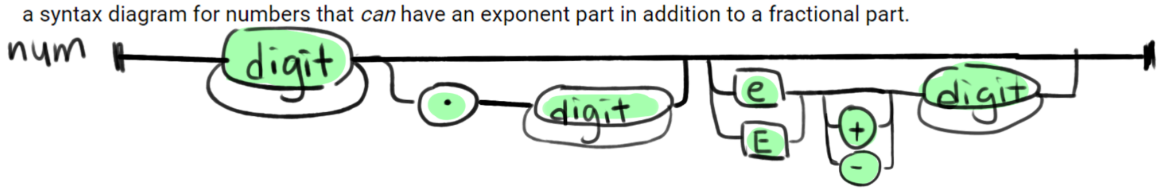
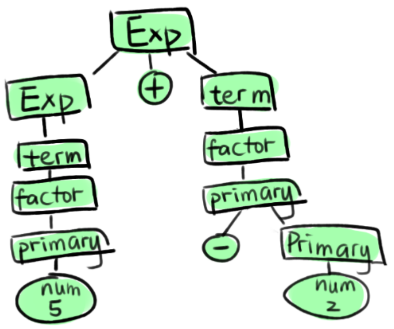
1. The Astro language allows numbers to have a fractional part but no exponent part (like `e5`, `E+7` or `e-12`). Give a syntax diagram for numbers that *can* have an exponent part in addition to a fractional part.



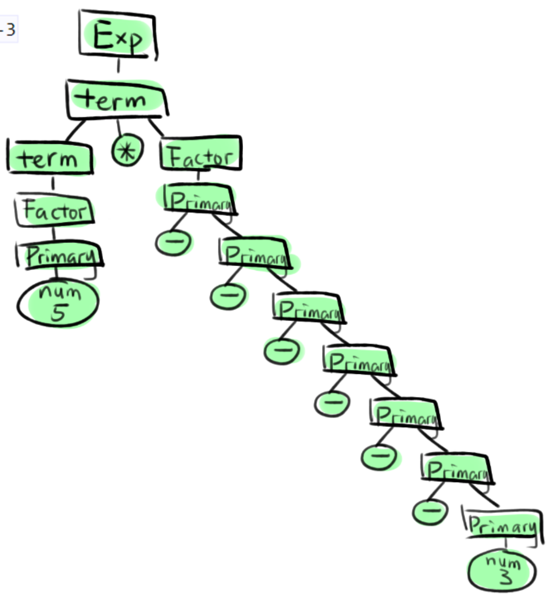
2. Even though we designed the Astro language to have what seems to be reasonable precedence and associativity, it allows some weird constructs, such as:

For each, show the concrete syntax tree (parse tree) rooted at **Exp**.

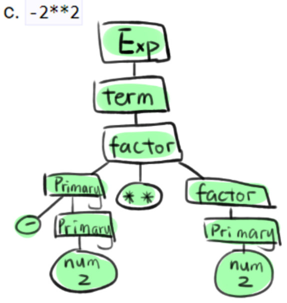
a. `5+-2`



b. `5*------3`



c. `-2**2`

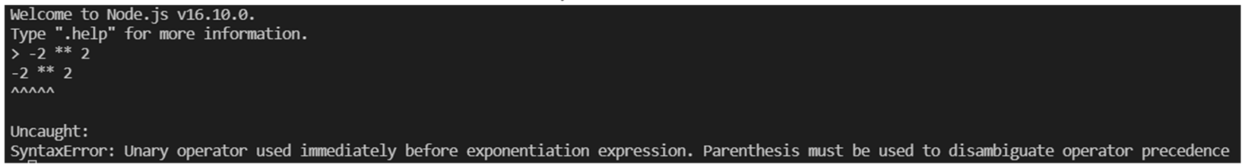


3. In the previous problem, we mentioned that `-2**2` was a "weird" expression. But maybe it is not. Or maybe it is. To investigate, answer the following:

a. What does `-2**2` evaluate to in Python? -4

b. What does `-2**2` evaluate to in Astro? 4

c. What does `-2**2` evaluate to in JavaScript? Syntax error



```
Welcome to Node.js v16.10.0.
Type ".help" for more information.
> -2 ** 2
-2 ** 2
^^^^^

Uncaught:
SyntaxError: Unary operator used immediately before exponentiation expression. Parenthesis must be used to disambiguate operator precedence
```

d. If the previous three answers were not the same, which of the following do you like best and why? In particular, what do you think is most beneficial about JavaScript's approach (from a human user perspective)? I like Astro's answer because it gives the right answer.

JavaScript stopped human from making mistakes, which will avoid time spent on debugging.

4. Show alternative grammar rules for Exp, Term, Factor, and Primary that would make the code fragments of Problem 2 illegal (not derivable by the grammar)?

4 Exp → (Exp("+"|"-"))? Term

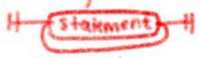
Term → (Term("*"|"/"|"%"))? Factor

Factor → Primary ("**" Factor)?

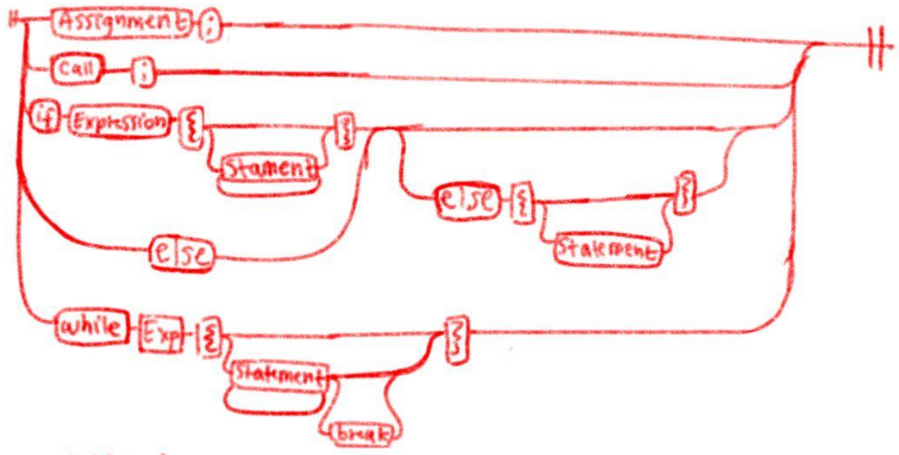
Primary → num
| id
| Call
| "(-" Primary ")"
| "(" Exp ")"

5. Astro is a really tiny language, so we'd like to make Astro++. This new language has an if-statement, a while statement, a break statement, and relational operators. The while statement should start with the keyword `while`, followed by a test expression, followed by a block (a curly-brace delimited sequence of statements). The if-statement should start with the keyword `if`, followed by a test expression, then a block, then an optional else-part which is the keyword `else` followed by either a block or another if-statement. Neither the while statement nor the if statement should end with a semicolon. The break statement is only allowed to appear in a while statement's block. The relational operators are the same as those in Python and are to be NON-associative. All of the relational operators are on the same precedence level, lower than all other

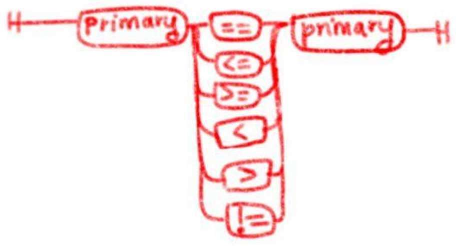
5 Program



Stament



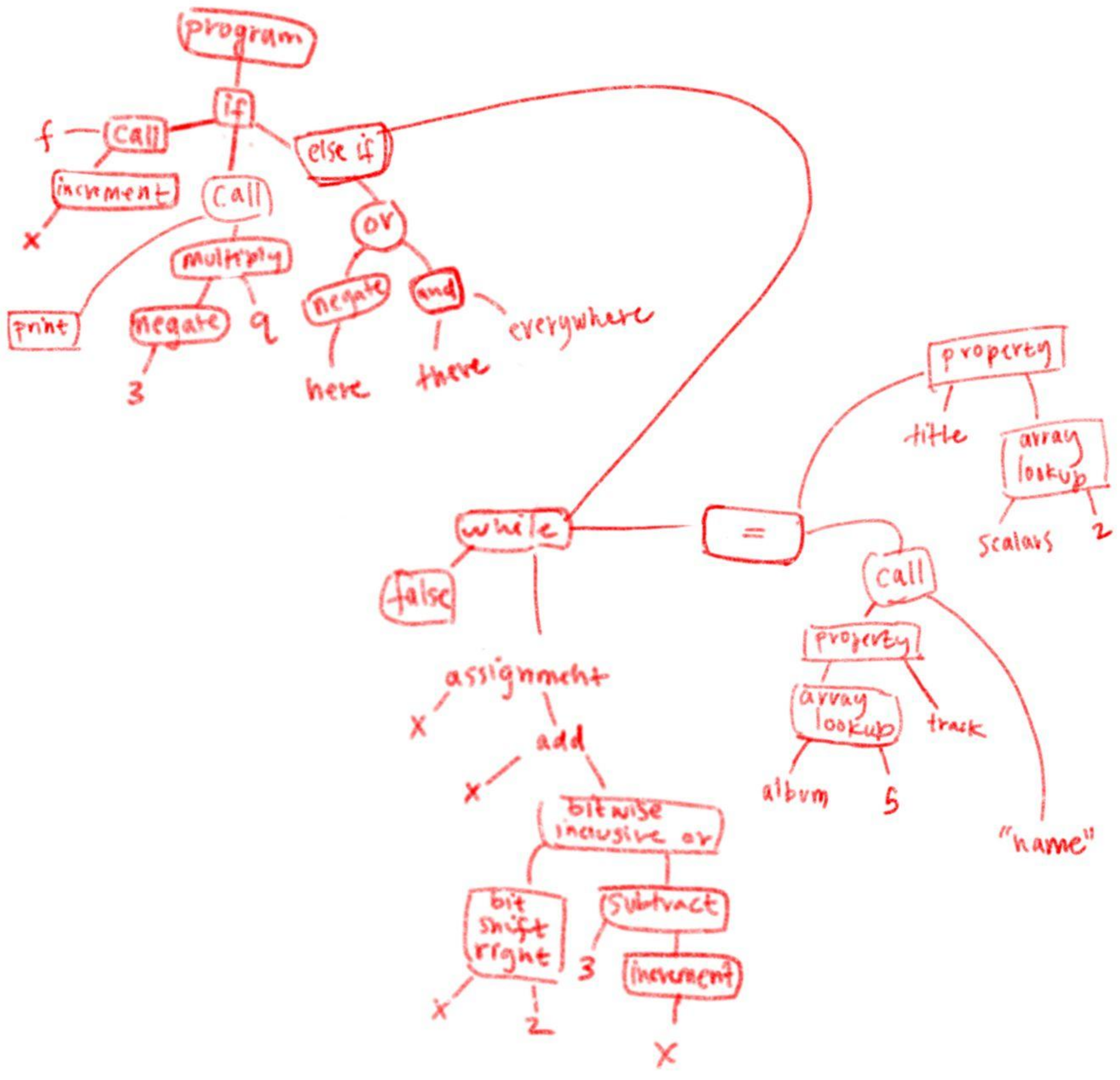
relational



6. Give an abstract syntax tree for the following Java code fragment. (This problem gives you the opportunity to really think through the difference between concrete and abstract syntax).

```
if (((f(--x)))) {
    System.out.println(- 3*q);
} else if (!here||there    &&everywhere) {
    while (false) {
        x += x >> 2 | 3  -++  x;
    }
    album[5].track("name") = title.scalars[2];
}
```

Please note that the question asks for **an abstract syntax tree** and not a parse tree. Please understand the difference before you tackle this problem.


program
if
f
call
increment
x
call
print
multiply
negate
3
q
else if
or
negate
and
everywhere
here
there
while
false
assignment
x
add
x
bitwise inclusive or
bit shift right
x
2
subtract
3
increment
x
=
property
title
array lookup
scalars
2
call
property
array lookup
track
album
5
"name"