CS 352 HW 1: Parsing

Due Friday, January 31, 10pm

Question 2:

[2] Completely parenthesize the expression below according to the operator table from Question 1.

```
(((1 + 2) - 3) < (4 * (5 ^ 6)))
```

[4] Draw the parse tree for this expression according to your `expr` rule from Question 1. You may omit whitespace from your parse tree.

|- int_val - $

Expr – eq_less – “<”

|- int_val - %

|- mult – expo – atom - int_val – “3”

|

% - Expr – add_sub – “-“ |- mult – expo – atom - int_val – “2”

|- expr – add_sub – “+”

|- mult – expo – atom - int_val – “1”

|- atom – int_val – “6”

|- expo – “^”

$ - Expr – mult – “*” |- atom – int_val – “5”

|

|- expo – atom – int_val – “4”

Question 3

[6] Describe an operator that could be added to the expression language in question 1.

The operand I would add would be the greater than symbol.

- Symbol: “>”
- Arity: binary
- Precedence: at the bottom with “<”
- Associativity: Non-associative
- Type: Integer
- Semantics: Would return a Boolean value of 1 if the left operand is less than the right operand. Otherwise, return a Boolean value of 0.

[2] Explain the approach you would take to add this operator to your grammar.

To implement this, I would change my ‘eq_less’ nonterminal to be named ‘eq_less_greater’ and implement the functionality therein. There would also need to be a new terminal to recognize the symbol that would then be appropriately implemented in our renamed function.