Muhammad Owais Bawany -- 7993647

Patrick Langis -- 8196917

Felix Singerman -- 7970742

CSI3120 - Assignment 2

**1. Define syntax and semantics.**

Syntax is the structure or form of the expressions, program units and statements. Semantics are the meaning of the statements, expressions and the program units.

**2. Define lexeme and token.**

A lexeme is the lowest level syntactic unit of a language while a token is a category of lexemes.

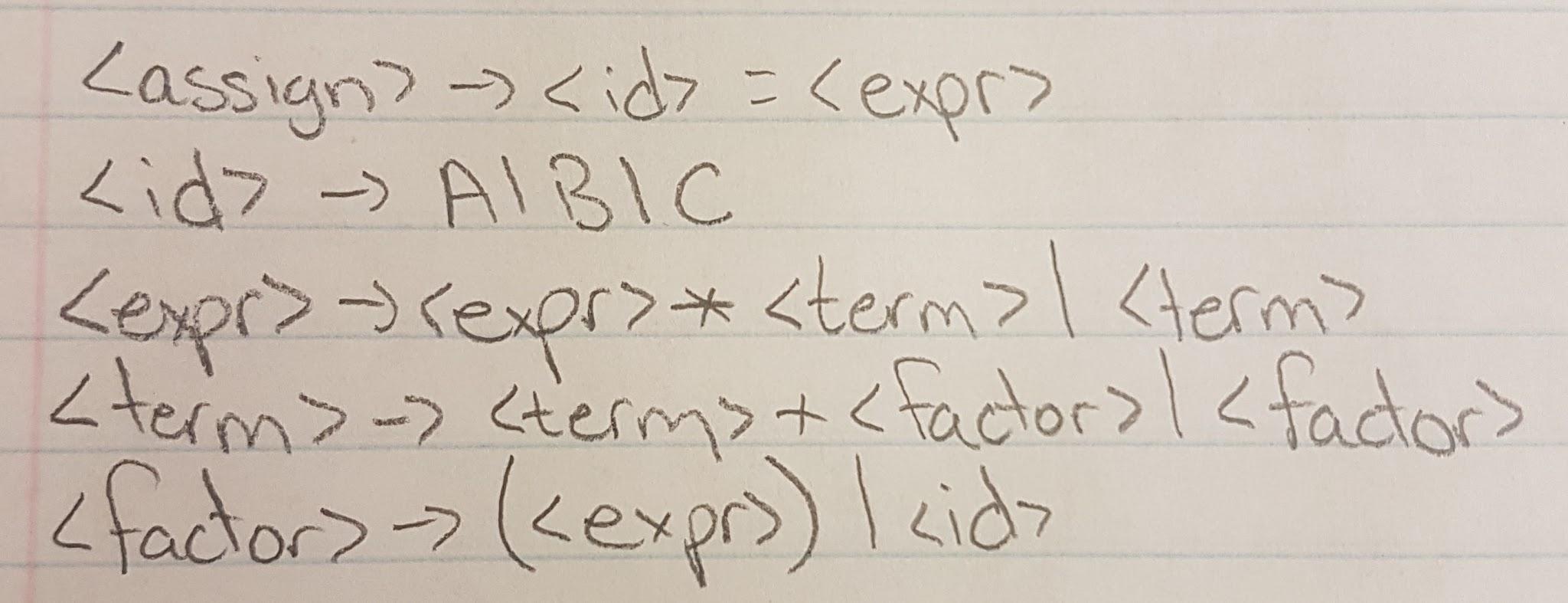
**3. The two mathematical models of language description are generation and recognition. Describe how each can define the syntax of a programming language.**

Generators are devices that generate sentences, a string of characters over an alphabet, of a language. By comparing the structure of the generators to the given sentences, it is possible to determine if the syntax is correct. Recognizers, on the other hand are devices whose purpose to check a string and see if it belongs in the language

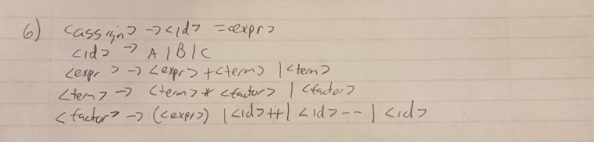
**4. What is the difference between a sentence and a sentential form?**

A sentential form is a string of symbols where a sentence is a sentential form made of only terminal symbols.

**5. Rewrite the following BNF to give + precedence over \* and force + to be right associative.**



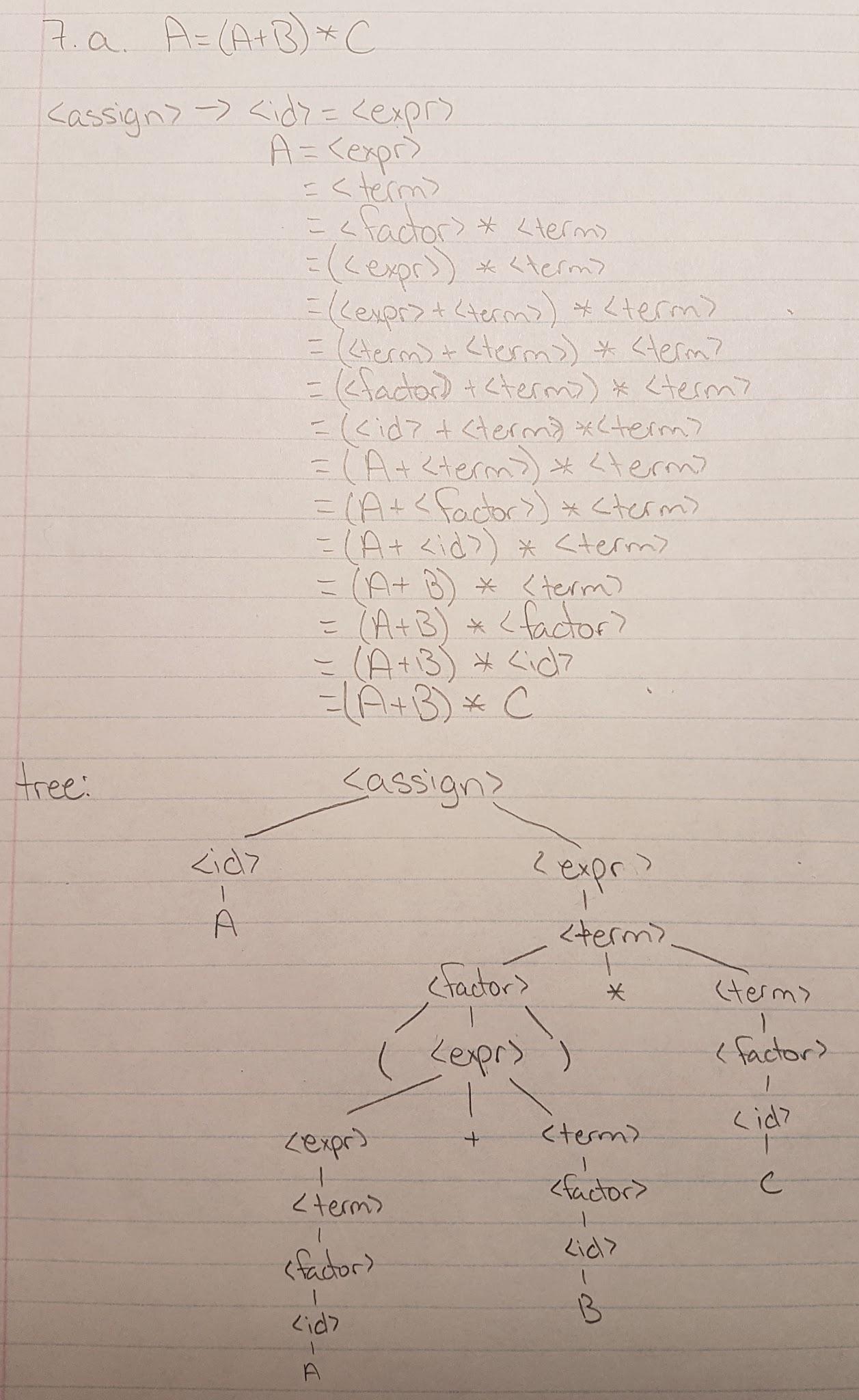
**6. Rewrite the BNF of problem 5 to add the ++ and -- unary operators of Java.**



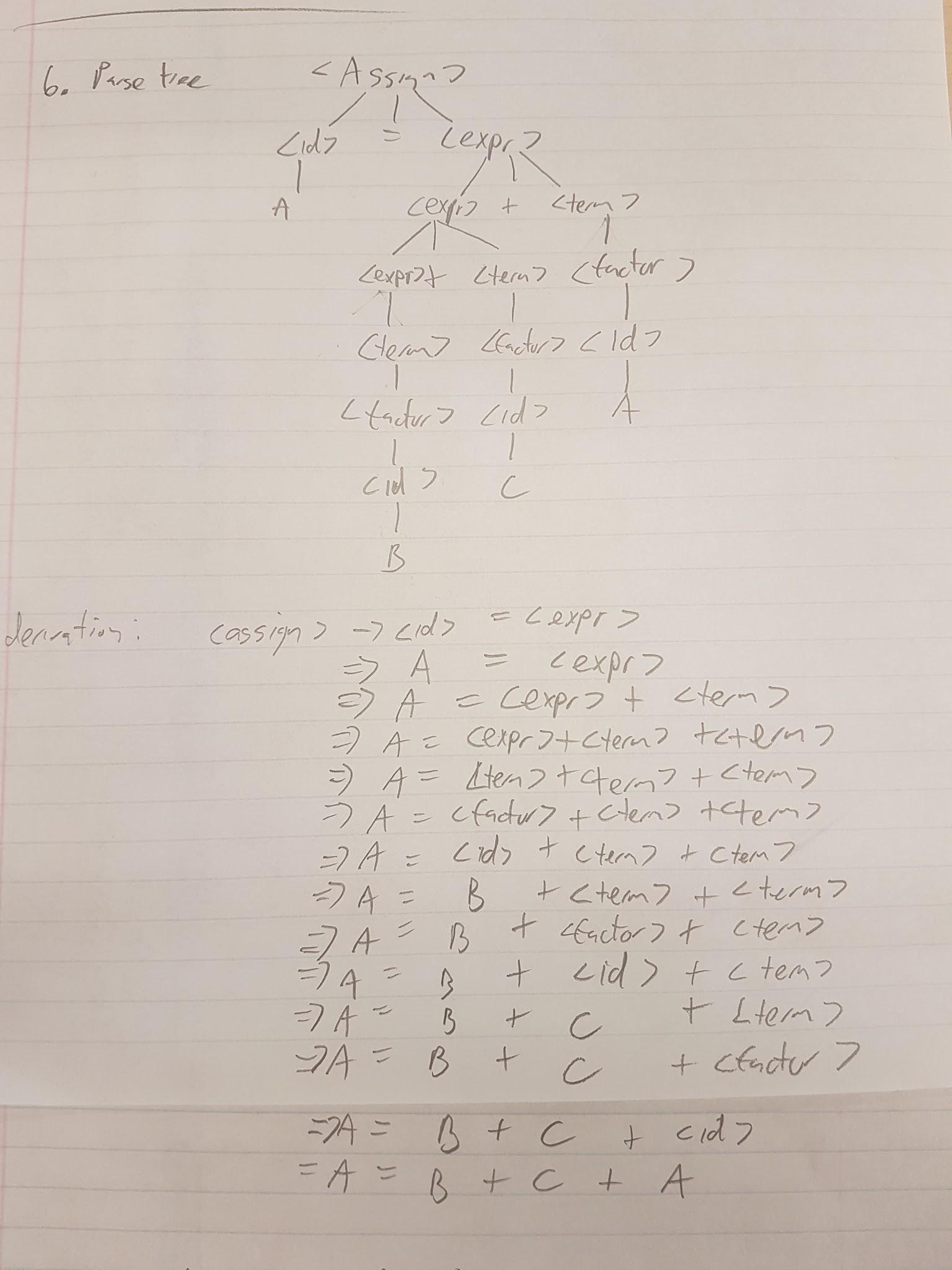
**7. Using the grammar in problem 5, show a parse tree and a leftmost derivation for each of the following statements:**

**Assumption: use the initial grammar of 5**

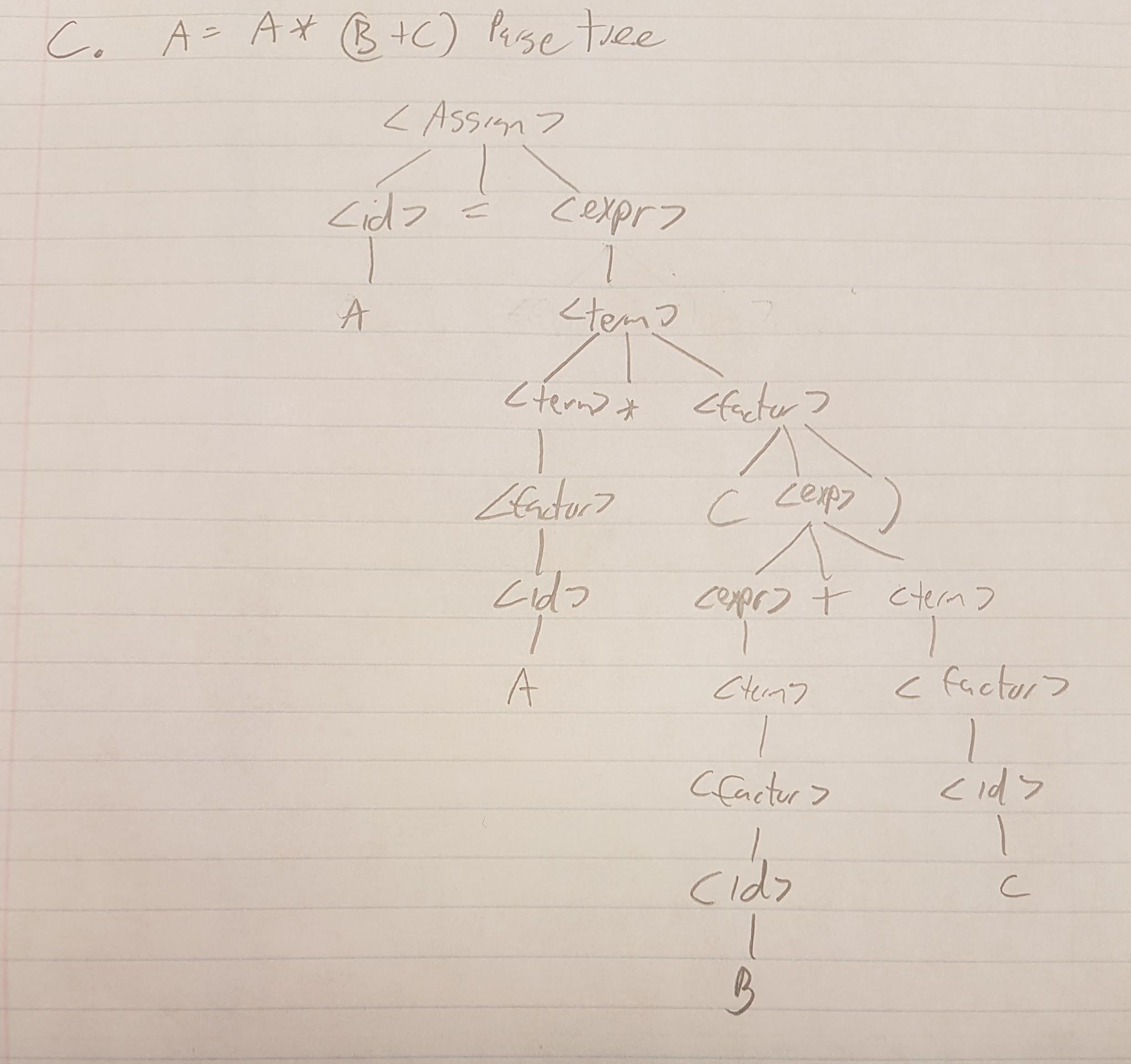
**a. A = ( A + B ) \* C**

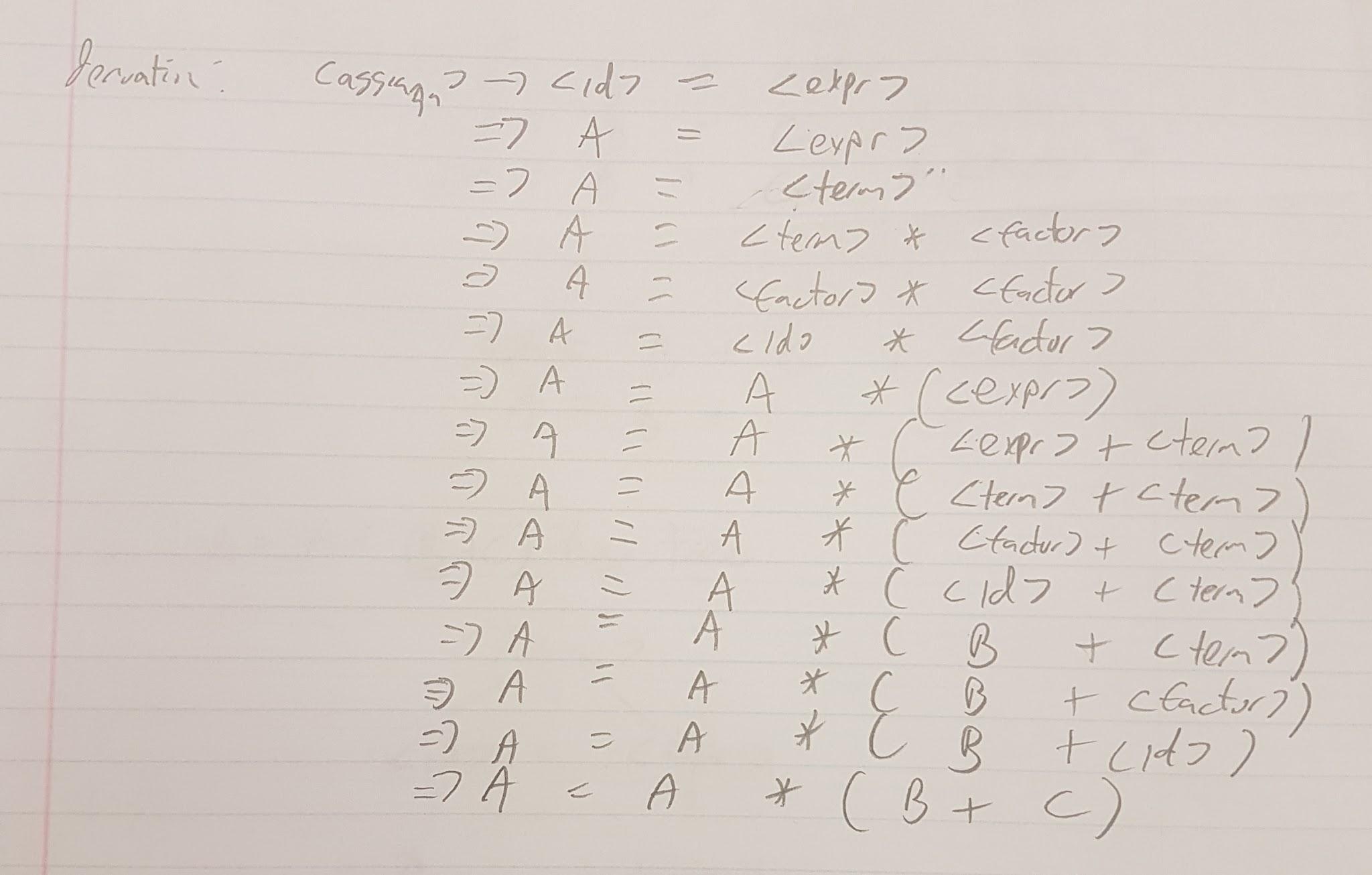


**b. A = B + C + A**

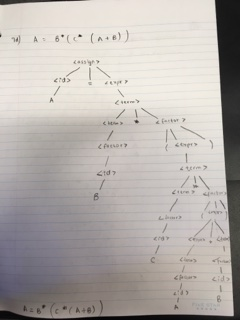
****

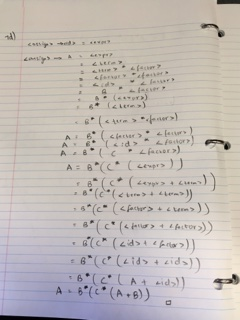
**c. A = A \* (B + C)**

****

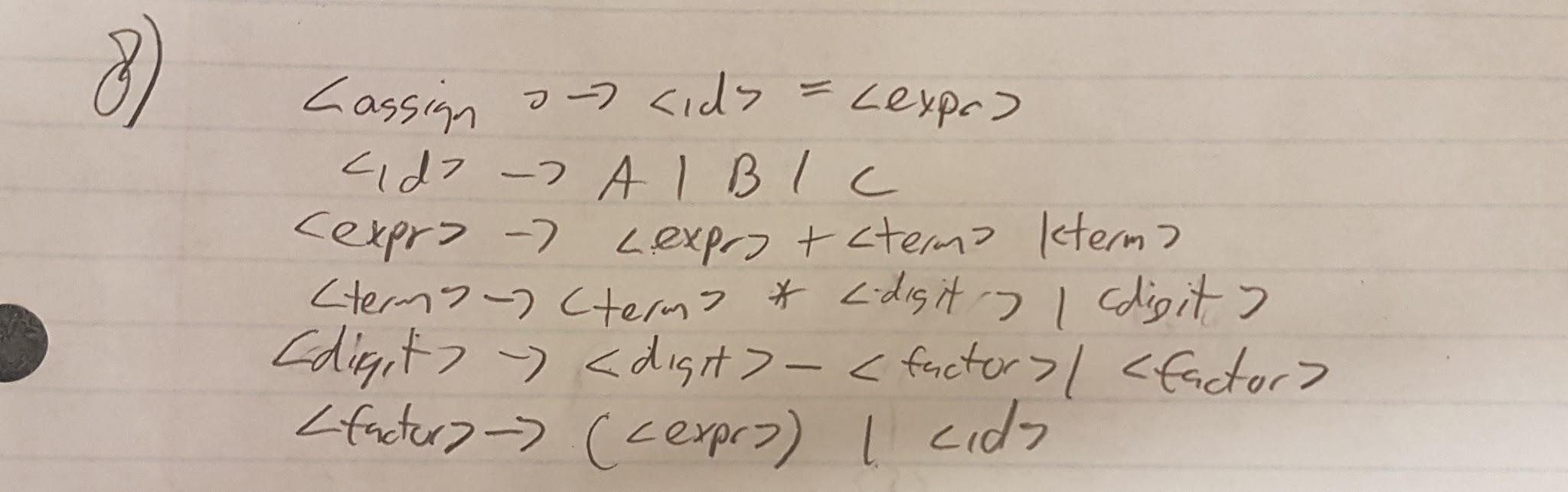
****

**d. A = B \* (C \* (A + B))**

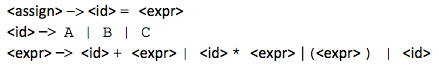
****

****

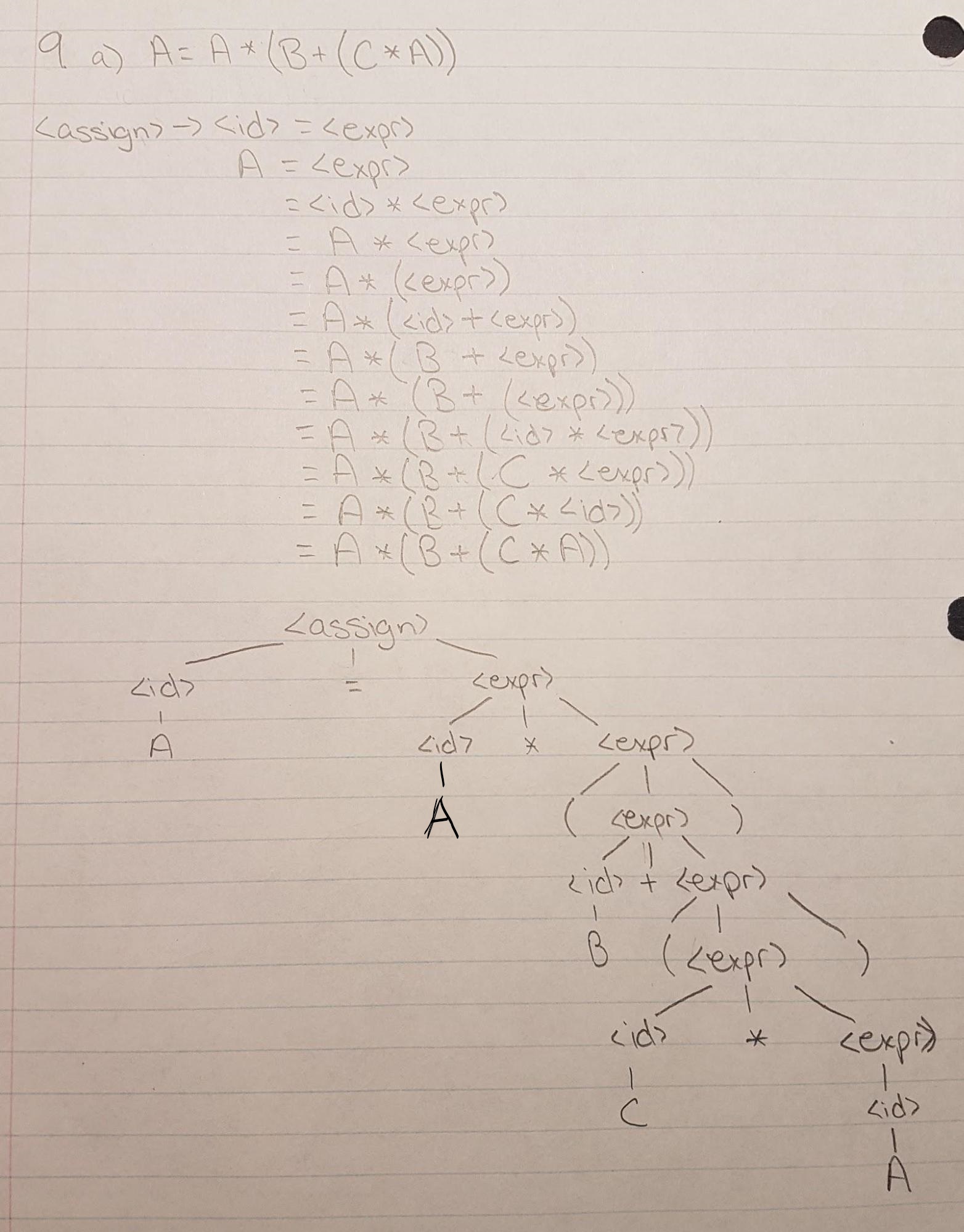
**8. Modify the grammar of problem 5 to add a unary minus operator that has higher precedence than either + or \* .**



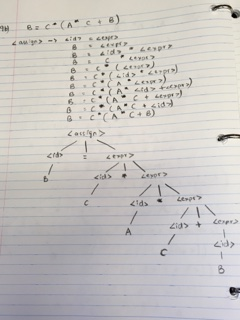
**9. Using the following grammar, show a parse tree and a leftmost derivation for each of the statements:**



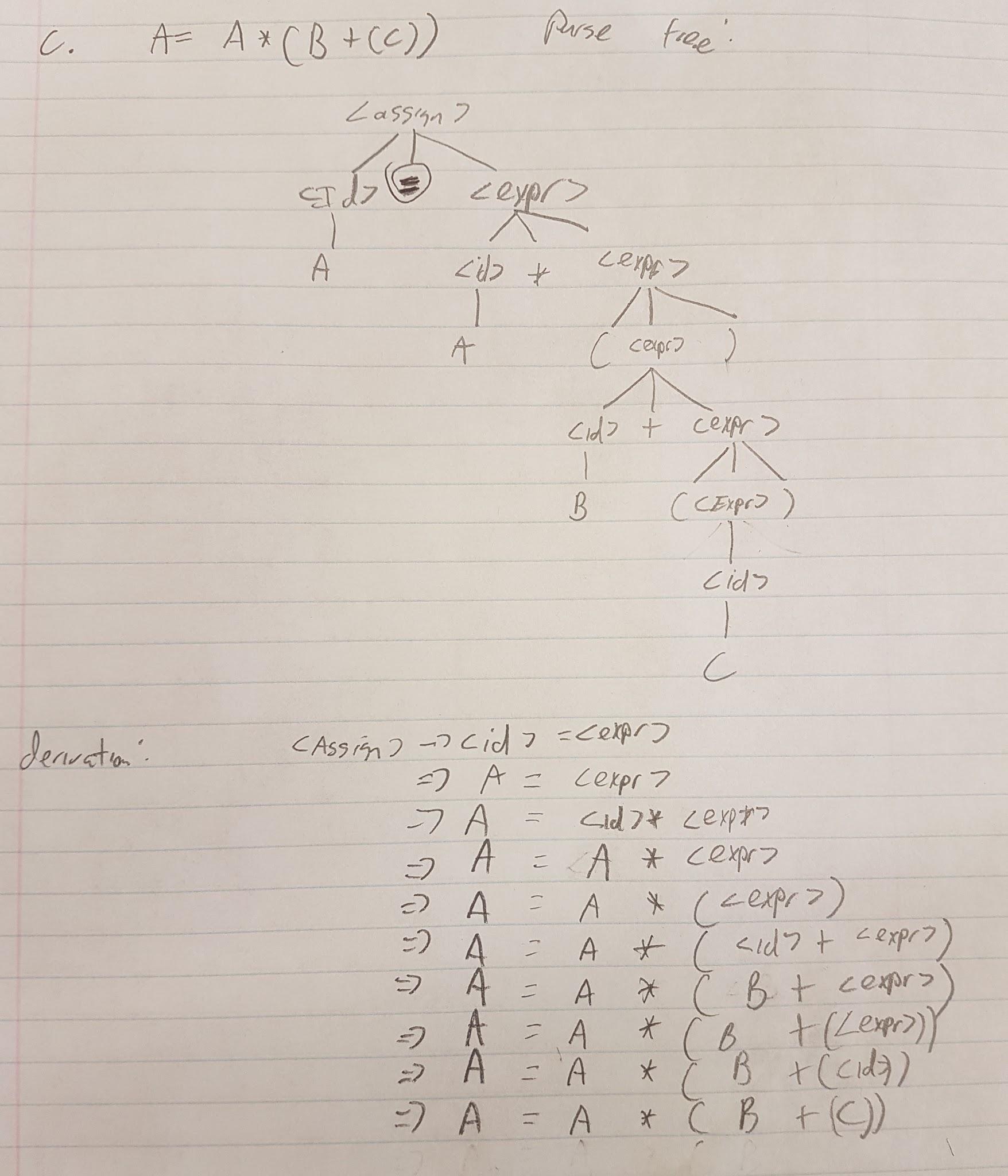
**a. A = A \* (B + (C \* A))**

****

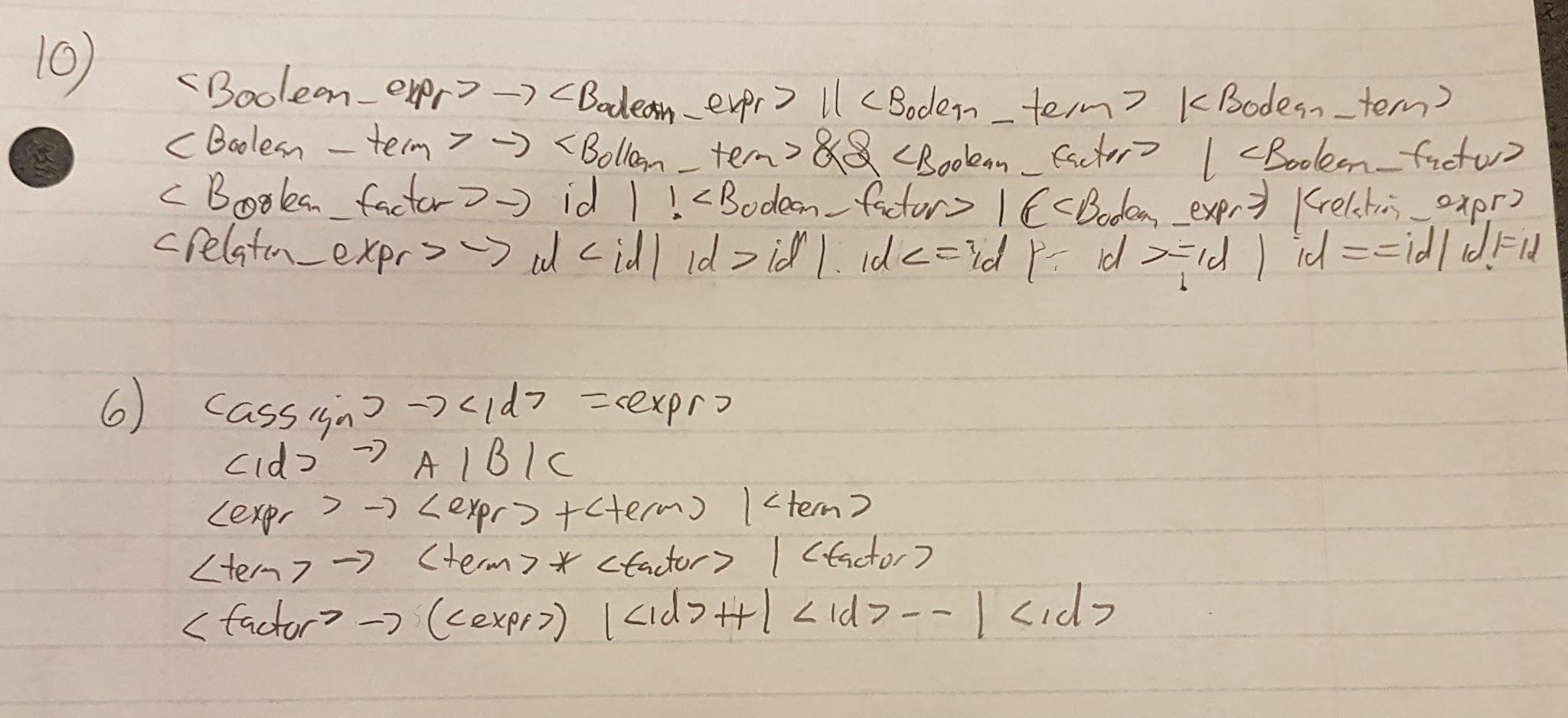
**b. B = C \* (A \* C + B)**

****

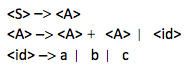
**c. A = A \* (B + (C))**

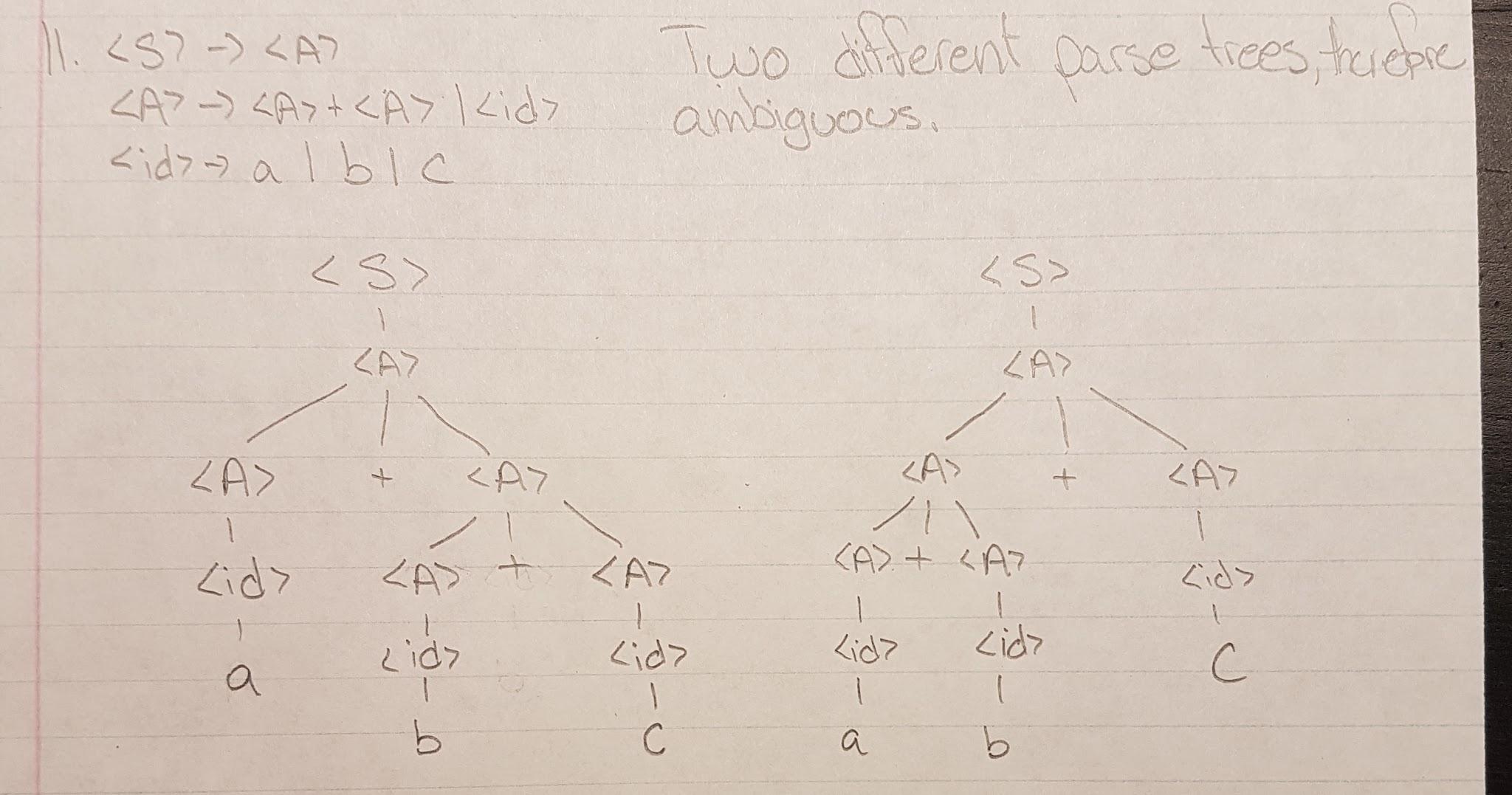
****

**10. Write a BNF description of the Boolean expressions of Java, including the three operators &&, ||, and ! and the relational expressions .**

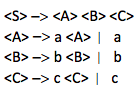
****

**11. Prove that the following grammar is ambiguous:**

****

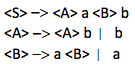
****

**12. Describe, in English, the language defined by the following grammar:**



The language has to have at least one of all three “letters”, however, each letter can be repeated as many times as desired, in the order required ie: aaabccccccccc. a has to be before b which is before c.

**13. Consider the following grammar:**

****

**Which of the following sentences are in the language generated by this grammar?**

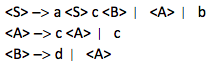
**a. baab**

**b. bbbab**

**c. bbaaaaa**

**d. bbaab**

**14. Consider the following grammar:**

****

**Which of the following sentences are in the language generated by this grammar?**

**a. abcd**

**b. acccbd**

**c. acccbcc**

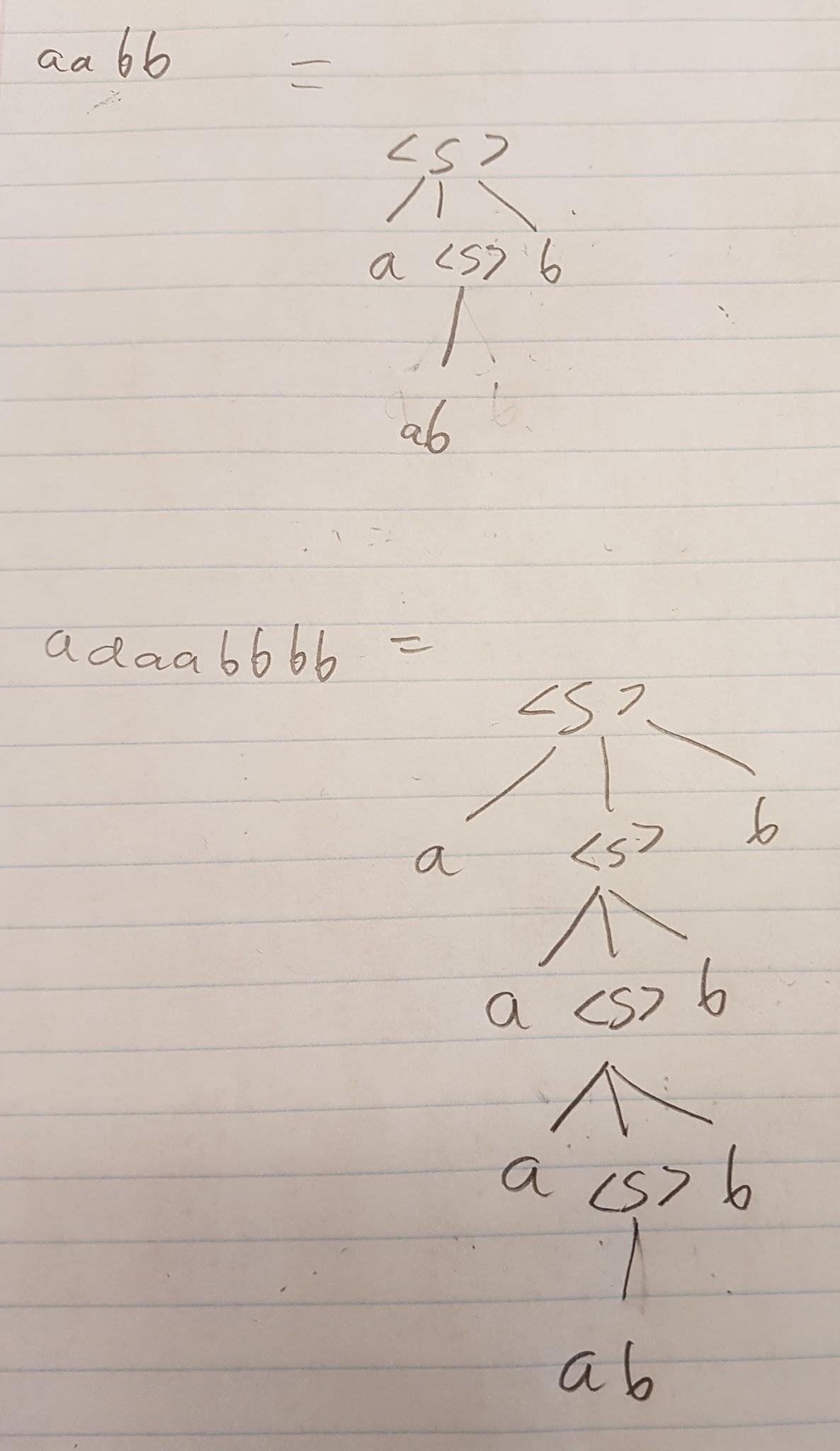
**d. acd**

**e. accc**

**15. Write a grammar for the language consisting of strings that have St of the letter a followed by the same number of copies of the letter b, where n > 0. For example, the strings ab, aaaabbbb, and aaaaaaaabbbbbbbb are in the language but a, abb, ba, and aaabb are not.**

<S> -> a<S>b | ab

**16. Draw parse trees for the sentences aabb and aaaabbbb, as derived from the grammar of previous problem (prob. 15).**



**17. Distinguish between static and dynamic semantics.**

Static semantics: Has to do with the legal forms of the programs. The analysis required to check the specifications is done at compile time. May static semantic rules of a language state its type constraint.

Dynamic semantics: Is the meaning of the expression, statements, and program units of a programming language. No universally accepted notation or approach has been devised.

**18. What purpose do predicates serve in an attribute grammar?**

The predicate is used to determine whether a statement in the language is correct or not. For example a false predicate value indicates that there was a violation of the static semantic rules, or of the syntax of the language.

**19. What is the difference between a synthesized and an inherited attribute?**

Synthesized attributes are used to pass semantic information up a parse tree, while inherited attributes pass semantic information down and across a tree.

**20. What is the primary use of attribute grammars?**

An attribute grammar is an extension to a context-free grammar. This allows certain language rules to be described ie: type compatibility. They describe and check the correctness of static semantic rules in a programming language.

**21. Describe the two levels of uses of operational semantics.**

There is natural operational semantics, which is at the highest level, the interest is in the final result of the execution of a complete program.

Structural operational semantics, this is at the lowest level, these semantics can be used to determine the precise meaning of a program through an examination of the complete sequence of the state changes that occur when the program is executed.

**22. In denotational semantics, what are the syntactic and semantic domains?**

The mapping functions of a denotational semantics programming language have a domain and range.

Syntactic domain: The syntactic structures that are mapped.

Semantic domain: The range.

**23. What is stored in the state of a program for denotational semantics?**

The meanings of programs and program constructs and some language constructs for example expression. Expressions are mapped to values and not states. State changes are defined by mathematical functions. Denotational semantics uses the state of the program to describe meaning. The state of the program for the denotational semantics is the value of all its current variables.

**24. Which part of an inference rule is the antecedent?**

The top part of the inference rule is the antecedent.

**25. Explain what the preconditions and postconditions of a given statement mean in axiomatic semantics.**

An assertion before a statement that is a precondition states the relationships and constraints among the variables that are true at that point in the execution of the program. An assertion following the statement is the post-condition. A given assignment statement with both pre and post conditions can be considered a logical statement. If the assignment axiom when applied to the post condition and the assignment statement produces the given precondition the theorem is proven. An assertion immediately preceding a program statement describes the constraints on the program variables at that point in the program. An assertion immediately following a program statement describes the new constraints on those variables and possibly others after execution of the statement.

**26. What is the difference between an intrinsic attribute and a non-intrinsic synthesized attribute?**

An intrinsic attribute is an inherent characteristic of a terminal symbol in the grammar (e.g., an identifier in a program). So the value of the attribute is determined solely from the terminal symbol. A non-intrinsic synthesized attribute is an attribute of a non-terminal symbol in the grammar. It’s value depends on the values of the attributes in the children of that non-terminal symbols nod in the parse tree.

**27. Compute the weakest precondition for each of the following assignment statements and postconditions:**

**a. a = 2 \* (b - 1) - 1 {a > 0}**

Weakest precondition: b>=2

**b. b = (c + 10) / 3 {b > 6}**

Weakest precondition: c > 8

**c. a = a + 2 \* b - 1 {a > 1}**

Weakest precondition: b = 0.5

**d. x = 2 \* y + x - 1 {x > 11}**

Weakest precondition: y>0

**28. Compute the weakest precondition for each of the following sequences of assignment statements and their postconditions:**

**a. a = 2 \* b + 1;**

**b = a - 3**

**{b<0}**

Substitute the right hand side of the second assignment in the postcondition to get a postcondition for the first assignment, then substitute the right hand side of the first assignment in that postcondition to get the precondition.

a-3 < 0

a < 3 weakest precondition for first assignment

2\*b+1 < 3

2b < 2

b < 1

}

**b.**

**a = 3 \* (2 \* b + a);**

**b = 2 \* a – 1**

**{b > 5}**

Substitute the right hand side of the second assignment in the postcondition to get a postcondition for the first assignment, then substitute the right hand side of the first assignment in that postcondition to get the precondition.

b > 2a - 1

Since postcondition is b>5,

2a - 1 > 5

2a > 6

Hence for the second one **a > 3**

And now putting this into the first equation, a = 6b + 3a

6b + 3a > 3, or simply

**2b + a > 3**

**29. Compute the weakest precondition for each of the following selection constructs and their postconditions:**

**a. if (a == b)**

**b = 2 \* a + 1**

**else**

**b = 2 \* a;**

**{b > 1}**

First considering the if part of the statement, b = 2a + 1, and according to postcondition b > 1,

2a + 1 > 1, hence 2a > 0, hence a > 0 from if part

From the else portion, b = 2a following our postcondition b>1, 2a > 1, and hence a > 0.5

Combining the a > 0 from if part and a > 0.5 from else part, we make our weakest precondition to be a > 0.5 that works for both if and else part.

**b. if (x < y)**

**x = x + 1**

**else**

**x = 3 \* x**

**{x < 0}**

For the if part of the statement, x = x +1, with our postcondition x < 0, our x on the RHS would have to be **x < -1**

For the else part x = 3x, with our postcondition x < 0, our x on the RHS would have to be x<0 as well.

Hence x < -1 is our weakest precondition since it satsifies both x < 0 for else part and x < -1 for if part

**c. if (x > y)**

**y = 2 \* x + 1**

**else**

**y = 3 \* x - 1;**

**{y > 3}**

For the first part the if part to satisfy y > 3 for y = 2x +1, 2x + 1 > 3, hence 2x > 2 and x > 1

For the else part y = 3x - 1, 3x -1 > 3, 3x > 4, so x > 4/3.

So the weakest precondition that satisfies for both is x > 4/3.