

# 236319 - Programming Languages - HW02

---

Submitted by Itay Segev and Arad Reder



# Question 1

---

1.



Start Symbol: `<statements>` .

Terminals:  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  
`+` , `-` , `×` , `÷` .




Non-Terminals: `<statements>` , `<statement>` , `<expression>` , `<variable>` ,  
`<term>` , `<operation>` .

2.

a. No

Every `<statements>` starts with a `<statement>` , which, in turn, starts with either a  or  . This series of terminals doesn't begin with either, so it doesn't belong in this grammar.

b. No

Every notebook emoji comes from a variable, and every variable needs either a  or a  before it according to the grammar. On line 3 there's a notebook with a  before it, which can't be in this grammar.

c. Yes

`<statements>`

↓

`<statement>`  
`<statements>`

↓






`<statement>`  
`<statement>`  
`<statements>`

↓








`<statement>`  
`<statement>`

<statement>









↓

 <variable>  <expression>  
 <expression>  
 <variable>  <expression>














↓

   <term>  
 <expression> <operation> <expression>  
   <term> - <term>
















↓

   <term>  
 <term>  <expression> <operation> <expression>  
   <term> - <term>

↓

     
    <variable> - <term>  
    - 

↓

     
     -   
    - 

d. Yes

<statements>

↓

<statement>

<statements>

↓

<statement>

<statement>

<statements>

↓

<statement>

<statement>

<statement>

↓


 <variable>  <expression>


 <expression>

 <expression>

↓

   <term>

 <expression> <operation> <expression>

 <expression> <operation> <expression>

↓

  <variable>   <variable>

  <variable>   <variable>





↓

### 3. Yes


For example: "  +  +  " is ambiguous. Because this can come to be in 2 different ways:

<statements>


↓

<statement>

↓

 <expression>



 `<expression> <operation> <expression>`

From here we can break either the left or right `<expression>` to "`<expression> <operation> <expression>`", which will create 2 different trees (with possibly 2 different meanings).

## Question 2

---

1. The `"#"` function in SML casts a `string` to a `char`, but the `"^"` function expects 2 strings.
2. The `"/"` function expects 2 `real`s, but both 84 and 2 are `int`s.
3. Comparing `x` and `0` leads us to believe `x` is of type `int`, but the function can either return `x` or `false` (depending on the value of `x`), which is forbidden in SML since a function can only have a single return type (`x` is `int`, `false` is `bool`).
4. Comparing `x` and `#"a"` leads us to believe `x` is of type `char`, but further on we try to use `^` on it, which only accepts `string` types as arguments.
5. The `-` in SML means subtraction, not negation (negation is `~`), and so `(-3)` is not a valid expression.
6. The function `Math.sqrt` expects a `real` as an argument, but `9` is of type `int`.
7. `sin` is not a defined function in SML.
8. `if` is a reserved word in SML, and cannot be used as a value name.
9. The function `String.sub` returns the `char` at the specified index of a string. The index of the last character in `"hello"` is 4, and so trying to take the 5th character isn't allowed.
10. The `Math.sqrt` function's return type is `real`, although the function `sqrt_of_int` is set to return an `int`.

## Rejected Memes

---

