PRELAB 3

In this prelab you will convert a semi-python code into a semi c++ code. The input python file contains only assignment operations with addition as it is seen in the example. There are three types of variables, integer, float and string. In python, you do not need to declare the type of the variable, but in c++ you have to. So your task is to convert code to c++ by putting declarations into the head of output.

- You can assume that no variable will have more than one type. For example in the input file you will not see lines like below.

A = 3
A = “ali”

- There can be type mismatch in the input file.
    - an integer and a string, halt the program and given an error message

A = 3
B = “ali”
C = A + B
	type mismatch in line 3

    - a float and a string, halt the program and given an error message

A = 3.2
B = “ali” + A
	type mismatch in line 2

    - Untyped variable is used, halt the program and given an error message

B = “ali” + A	Untyped variable is used in line 1

    - an integer and a float, assign that variable as float

A = 3.2
B = A + 3
	float A, B;
	A = 3.2;
	B = A + 3;

- There are several input and output files below. Please check them to understand better. If there is any unclear point, do not make assumptions, ask in the coadsys forum.

WARNING:

Zip your lex, yacc and Makefile files (just 3 files) and submit to coadsys. We will use your makefile to grade your prelab, so be sure that your Makefile works properly.

HOW TO CODE

This assignment is not easy if you try to do everything at the same time. But if you seperate the things to do and carry out each of them one by one, you can do it easily. To help you, I give you the list to do. If you follow this order you can easily do your assignment.

1) Syntax analysis
    a) Write your lex code and test it
    b) Write your yacc grammar and test it. It is suggested to use or update Example 6 or example 7 to write the grammar.
    c) In this part, just test if your grammar can accept the input file or not. Do not make any semantic analysis.
2) Semantic analysis part 1
    a) A variable with no type cannot be summed with any variable.
    b) A string variable cannot be summed with a float or int.
    c) The result will be float if an integer and float variable is summed.
    d) In the order to satisfy the constraints in a,b and c we need to store the types of variables in a structure.
    e) In example 7, we used map structure to keep variables and corresponding values. So if you wish you can update example 7 to store types not values. You can use map structure or a different one you are familiar with.
    f) You can store the types in any format you want in your structure, you can directly store it as strings (string, int, float) or you can store it as an integer value, for example (0 for string, 1 for float, 2 for integer).
    g) Now, since you have a table now, you can fill your table by using your grammar and $$ properties of yacc. Pass the type information to the parent and make control in each step if there is a type conflict or not. In the rule, (var = …) set the type of the variable.
    h) Check your grammar with the inputs that contain the conflicts in a and b.

3) Semantic analysis part 2

   a) Now you have a table with the types.
   b) Print the declaration lines for each type. The output should be in format:

```
int variable_list;
float variable_list;
string variable_list;
```

   c) There may not be any int, float or string variables in the input file. Do not forget those cases.

4) Semantic analysis part 3
   a) Finally you will write the remaining of the output.
   b) You cannot use $$ in this part, because you have already used it in part1 to pass types to parents.
   c) You cannot print it immediately. If so, the declaration part (output of part 2) will be in the end. But we want it in the beginning of the output.
   d) Thus, you should use a global string variable as in the example 10, and pushback each line by putting semicolon in the end.
   e) After printing the output in part 2 , print the content of the global variable.
   f) You have finished your assignment Congratulations :)

EXAMPLES

Example input	Example output

a = 3
b = 5
var = “ali”
var2 = “ata”
d = a + b
var = var + var2
c = a + b + 10 + d
f = 3.5
g = 5.2
res = f + g
	int a,b,c,d;
	float f,g,res;
	string var,var2;
	
	a = 3;
	b = 5;
	var = "ali";
	var2 = "ata";
	d = a + b;
	var = var + var2;
	c = a + b + 10 + d;
	f = 3.5;
	g = 5.2;
	res = f + g;

Example input	Example output:

b = 5
a = 3 + b + 8
	int a,b;
	
	b = 5;
	a = 3 + b + 8;

Example input	Example output

a = 3
b = 5
var = "ali"
var2 = "ata"
d = a + b
var = var + var2
c = a + b + 10 + 5.2
f = 3.5
g = 5.2
res = f + g
	int a,b,d;
	float c,f,g,res;
	string var,var2;
	
	a = 3;
	b = 5;
	var = "ali";
	var2 = "ata";
	d = a + b;
	var = var + var2;
	c = a + b + 10 + 5.2;
	f = 3.5;
	g = 5.2;
	res = f + g;

Example input	Example output

a = 3
b = 5
var = "ali"
var2 = "ata"
d = a + b
var = a + var2
c = a + b + 10 + 5.2
f = 3.5
g = 5.2
	type mismatch in line 6