Formal Languages and Compilers
Proff. Breveglieri, Morzenti
Written exam[1]: laboratory question
03/07/2018

SURNAME: .....
NAME: ..... Student ID: .....
Course: ○ Laurea Specialistica ○ V. O. ○ Laurea Triennale ○ Other: .....
Instructor: ○ Prof. Breveglieri ○ Prof Morzenti

The laboratory question must be answered taking into account the implementation of the `Acse` compiler given with the exam text.

Modify the specification of the lexical analyser (`flex` input) and the syntactic analyser (`bison` input) and any other source file required to extend the `Lance` language with the new `alias` construct.

The `alias` construct defines a code block where a variable acts as an alias for another one. Consider the following example.

```
int a,b,c;

a=3;
b=5;
c=7;

alias a b {
  a=a+1-c;
}

write(a);
write(b);
```

The example at hand employs the `alias` construct to define `a` as an alias for `b` within the code block following `b`. This implies that any action (definition or use) of `a` in the code block indeed ends up acting on `b`. In the example the two `write` statements will cause the printout of 3 and -1 respectively, as `a` is not modified, while `b` is as an effect of the aliasing.

The `alias` construct *cannot* be nested; such a syntax error should prevent the compilation of the program.

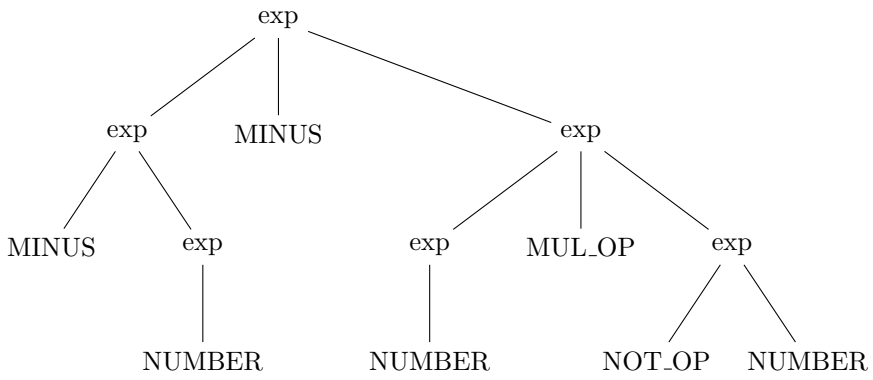
[1] Time 60'. Textbooks and notes can be used.
Pencil writing is allowed. Write your name on any additional sheet.

1. Define the tokens (and the related declarations in **Acse.lex** and **Acse.y**). (3 points)
2. Define the syntactic rules or the modifications required to the existing ones. (4 points)
3. Define the semantic actions needed to implement the required functionality. (18 points)
   The solution is in the attached patch.

4. Given the following `Lance` code snippet:

```
-3 - 3 * !3
```

write down the syntactic tree generated during the parsing with the Bison grammar described in Acse.y *starting from the exp nonterminal*. (5 points)



5. (**Bonus**) Describe how to modify the implementation you provided to support nested alias constructs.

开栈 alias struct (stack)

没 current_alias

current_aliased