Formal Languages and Compilers
Proff. Breveglieri, Morzenti
Written exam[1]: laboratory question
09/02/2015

SURNAME: ....
NAME: .... Student ID: ....
Course: ○ Laurea Specialistica ○ V. O. ○ Laurea Triennale ○ Other: ...
Instructor: ○ Prof. Breveglieri ○ Prof. Crespi ○ 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 factorial operator ! of an expression and with the absolute value of an expression $|\cdot|$. An example is following provided. Define the syntactic rules so that the precedence of the operators is correctly considered.

```
int x, y;

read(x);
read(y);

// factorial of x
write(x!);

// factorial of x!
write(x! !);

// abs of x
write(|x|);

// abs of |x| - y!
write(| |x| - y! |);
```

Figura 1: Example

[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)

4. Given the following `Lance` code snippet:

```
y = -!x + 2;
```

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

5. (**Bonus**) We are asked to extend the absolute value operator to the arrays. Describe how to implement the assignment of the absolute value $|\cdot|$ of an array (in the argument). For instance,

```
int x[10], y[10];

...

y = |x|;
```

Figura 2: Example