Formal Languages and Compilers
Proff. Breveglieri, Crespi Reghizzi, Morzenti
Written exam[1]: laboratory question
15/01/2019

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 *scalar product* operator. The operator, indicated by the `@` symbol, performs the scalar product between two conformant integer arrays. The precedence of the scalar product operator is higher than any other operator. An attempt to employ it to multiply scalars or non conformant arrays must result in a compile-time error. The following code snippet provides an example of operator use.

```
int u[3],v[3],a;

v[0]=1; v[1]=2; v[2]=-1;
u[0]=8; u[1]=1; u[2]=2;

a = v@v + 1; /* a == 7 after this statement */

if (u@v) {
  write(1); /* this statement is executed */
} else {
  write(0);
}
```

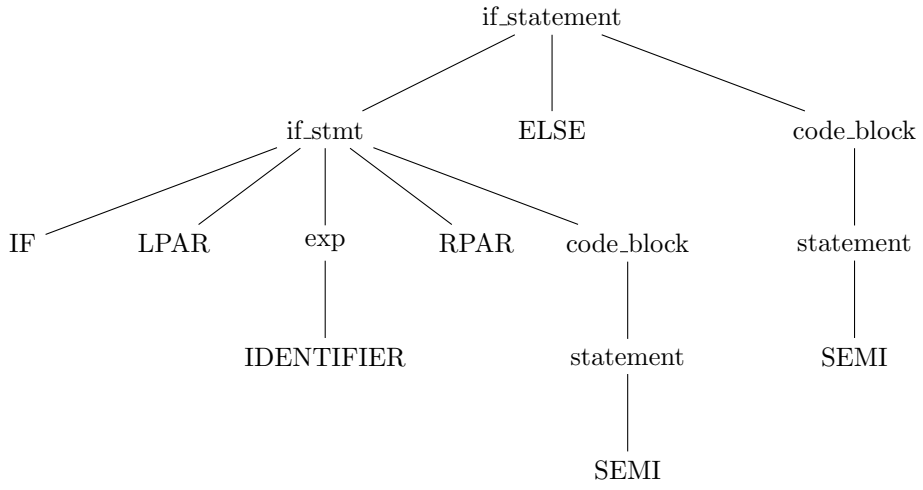
[1] Time 60'. Textbooks and notes can be used.
Pencil writing is allowed. Write your name on any additional sheet.

1. Define the tokens (and the related declarations in **Acse.lex** and **Acse.y**). (2 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:

```
if ( a ) ; else ;
```

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



5. (**Bonus**) Sketch which adaptations to the ACSE compiler should be made to allow the implementation of the component-wise product, employing the same syntax of the scalar one.