Formal Languages and Compilers
Proff. Breveglieri, Crespi Reghizzi, Morzenti
Written exam[1]: laboratory question
08/07/2015

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 **permutate** construct for arrays. An example is provided in the following.

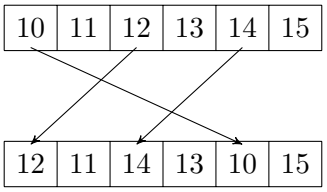
```
int arr[10];

// arr = [ 10, 11, 12, 13, 14, 15 ]
permutate( arr, q[ 4, 2, 0 ]p );
// arr = [ 12, 11, 14, 13, 10, 15 ]
```

The permutation is expressed as constant vector of indices, i.e. integer constants, that must be interpred as *source-destination* indices pairs. For example the chain q[4, 2, 0]p:

- element in *position 4* of the input array goes into *position 2* of the output array
- element in *position 2* of the input array goes into *position 0* of the output array
- element in *position 0* of the input array goes into *position 4* of the output array

A graphical representation of the semantic is the following:



The permutation must be **inplace**, meaning that the input array is modified to obtain the output array without the need of a temporary array.

Note that the vector of indices can be shorter or longer than the length of the array. Indeed each index must be smaller than the array size, otherwise the compiler must report an error.

[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:

```
int arr[10], x, y = 5;
;
```

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

5. (**Bonus**) Describe how to modify your solution to allow also an array as a vector of indices of the **permute** construct.

```
int arr[10], perm[3];

// arr = [ 10, 11, 12, 13, 14, 15 ]
// perm = [4, 2, 0]
permutate( arr, perm );
// arr = [ 12, 11, 14, 13, 10, 15 ]
```