TP2 Simple Translator

Lombardo Quentin
Hafsaoui Theo

1 Presentation

This *readme* is the result of the assignment that was ask in I53, the purpose of this document is to complement and help the comprehension of our work. To this end we will begin by enumerate what was our comprehension of the exercise :

1. the directory must be compose of :
   (a) a readme
   (b) a directory *A*
   (c) a directory *B*
   These two directory will be in turn compose 3 python scripte *Scanner.py Parser.py Codegen.py*
   (d) an a *compilo.py* in each directory to run each scripte
2. the readme must be in english afterall it's not a *lisez − moi*
3. the readme must be comprehensible for a "neuneu"
4. the *compilo.py* must take a file as parameter and result with an executable named *a.out*

For the commentary we believe that commentary are fix not answer to a probleme, that a code should be self-explanatory. obviously and sometimes commentary are necessary, but we try to economies theme.

2 Python scripte and their particularity

2.1 Scanner.py

When it come to the scanner not a lot can be said, this programme simply take a programme and return a list of tuple with two component.

For the implementation the choice has been made to stick with a for loop for readability and elegancy reason(that debatable of course), for the same reason dictionary was chosen. Some problem come with these choice but explain further in the code. The boolean part does have a few difference but the principle is the same.

2.2 Parser.py

The Parser is compose of 5 function, a sort of main, and four other function directly resulting from our grammar. The purpose is double, one virify through our grammar that our expression is correct, and two use this the grammar to obtain the r.p.n .

2.3 Codegen.py

This scripte is quite straightforward, it take a post fix expression (or r.p.n)and create a resulting code executable by python.
For this scripte two choice was possible, write A.out at this stage or wait until *compilo.py* the two was made the later was however choose for more modularity, indeed write after let more control on eventual problem and segmente bug. *Compilo.py* just link everything together and create a.out.

3 Test

Three ways are possible to test our code, the first one is through $test_arith.py$ that was given to ous, the second is manualy by using a file and a *compilo.py*. and the last one is by using *test.sh* wich is here to test with only argument in the cli.