Language Processors

Lab 5 – Syntax analysis with bison

Max Cattafi (m.cattafi@imperial.ac.uk)

bison (an open and free version of yacc) is a tool for generating C programs ("parsers") that perform syntax analysis of text according to a grammar specification contained in a .y file.

In order to perform syntax analysis, parsers usually operate on the input already organized in *tokens*: this is the result of lexical analysis, as we saw previously, and often parsers are interfaced with lexers.

In our example we want to generate a parser which recognizes simple arithmetic expressions on integers (with addition, multiplication and parentheses) such as (3+10)*4 and outputs the result of the evaluation.

Calculator.l contains a flex file which can generate the source code for a scanner which recognizes the tokens for this language.

Calculator.y contains a commented bison file containing the grammar specification of the language.

Command bison -d Calculator.y generates the source code for the parser and a header including the token declarations (used also in the .l file which is given in input to flex).

You can generate the source files and compile the program in one line:

```
bison -d Calculator.y && flex Calculator.l && g++ Calculator.tab.c lex.yy.c -o calc
```

However, since the software components begin to have entangled dependencies, it is probably a good idea to organize the process of generating and compiling the source in a `Makefile`. You can find a commented example in file `Makefile` and you can test it with the command `make`.

Two additional files are provided for a different version covering also decimal numbers. Complete the missing parts following the instructions in the comments, build the example using a suitable `Makefile` and test it.