# Compilers: P-language

*Sam Mylle, Federico Quin*
*University of Antwerp*
April 23, 2017

# 1 Progress

So far, we have allowed all mandatory C constructions in our grammar and we are able to create the AST and the symbol table. We have also implemented some extras, e.g. continue and break statement and for loops.
There is also a basic form of error checking. The symbol table is currently not yet fully operational though: when an identifier is encountered, it does not check whether it exists.

# 2 Grammar

Currently, the grammar we wrote supports all mandatory C constructions, and of course also the optional constructions we mentioned above. The grammar can be found in the ./res directory. We have provided some comments and we tried to make the names of nonterminals as clear as possible.

# 3 Sources

Our compiler currently consists of 5 big parts: The AST class, the ASTCreator class, the SymbolTable class, the SymbolTableBuilder class and the TypeChecker class. First, the AST related classes make the decorated AST. Then, the symbol table related classes make the symbol table based on this AST. Finally, the TypeChecker checks whether the types of the operands match. If anything fails, it will yield an error.

# 4 Testing

For testing our compiler, we chose for pytest. We have 26 test programs, where each program tests one functionality.

# 5 What we'll have to do

Our compiler will have to be more specific about its errors. We must also translate the AST and the symbol table to a real P program.