Project work - Phase 1

Principles of programming languages

Pinja Mikkonen 99219

1 Lexical analysis is a part of the compilation process that takes the source code and separates it into a list of tokens that get passed to the syntax analyser. Tokens are keywords, operators, symbols and identifiers that compilers needs to recognize in order to form coherent code sentences.

2 PLY tool is all about recognizing certain character combinations, or tokens. This is achieved by a list of reserved words and regular expressions that match certain character combinations. There are also functions that contain regular expressions that need to be recognized or ignored.

3

3.1 Keywords are all in a list of reserved words. Any time code finds a string literal it checks if that string matches any keywords.

3.2 `t_COMMENT` -function locates sets of three dots ('...') and marks anything between them as a comment that can be ignored. Matching is done in a lazy way to ensure that no actual code gets accidentally marked as a comment.

3.3 `t_ignore` -function causes the code to ignore whitespaces, as well as tabulators and \r -symbols.

3.4 Operators and delimiters have regular expression strings that match with them. They are part of the same tokens-list as reserved words and get recognized by the same function.

3.5 Decimals are defined by a regular expression as at least one int literal followed by a period and exactly one int literal.

3.6 When a string is located, the syntax analyzer checks if it matches any reserved words or predefined regular expressions.

3.7 A function name is matched by a regular expression that starts with uppercase letter and ends with one or more lowercase letters.

4

4.1 The regular expressions that match function names and variable names are otherwise identical, except that function names start with an uppercase letter and variable names start with a lowercase letter.

4.2 Keywords are all in a list for reserved words. If a string matches with any of them it's identified to be a keyword. Otherwise it's a variable name.

4.3 Both have a regular expression that matches to them, and keywords are checked in descending order of length. This way operator >= gets recognized as such and not >.

4.4 Any characters between two exclamation marks get marked as a comment. If there are no exclamation marks then a lexer proceeds to check if string literal matches any other regular expression.

4.5 Any string that is between two sets of three periods get marked as a comment and will be removed from code. This effect is done lazily to avoid any actual code getting marked as a comment.

4.6 Decimal literals have a decimal period included in their regular expression. Lexer also checks for decimal numbers before it checks for integer numbers – this way a number that matches a decimal number will get correctly identified.

5 Comments can span multiple lines. I also tried implementing code that would remove comments from inside string literals, but couldn't get it wo work. In theory `t_COMMENT` -function should get implemented before any other code, but it doesn't work for an unknown reason.

6 This assignment was quite fun in my opinion. It wasn't too hard, it was a good way to get used to the tools used in this assignment and it was a nice way to get used to Python again.