Design Document

Mandatory Assignment 1

IN3040

Alexander Presthus

I have chosen to write the interpreter in Python.

The interpreter is implemented in a single Python module named Robol.py, and utilizes Python classes for polymorphism.

The interpreter (Robol.py):

Non-terminals are classified as separate Python classes, with naming similar to the ROBOL Grammar notation specified in the assignment documentation. Similar non-terminals that share attributes and purpose (e.g. different types of expressions (number, identifier, Boolean etc.) all evaluate to some value), are divided into subclasses of a common super class (e.g. Exp is the parent class of NumberExp and BooleanExp).

Terminals are identified as symbols; Strings as a string in quotes; Symbols as a symbol in quotes; Numbers as an integer; Boolean values as a Boolean True or False.

The test code (testCode.py)

The test code defines a class TestCode which has methods for running some test programs. Each method has a set of commands. The commands simulate "parsed" commands, and calls relevant parts of the interpreter to instantiate the AST, then calls the interpret() method of the instance of the Program Class to start interpreting the AST.

The AST:

Program is the top-level of the AST. Program is instantiated with an instance of Robot and Grid. Robot is instantiated with an instance of Start, a list of instances of Statements and a list of instances of Bindings.

The initial call on Program.interpret() starts the interpretation of the AST-instance, initializing a recursive descent:

Program.interpret() calls Robot.interpret(grid), which attempts to call the interpret-methods on Start, the bindings in the bindings-list, and the statements in the statement-list. Furthermore, bindings and statements are interpreted until non-terminal expressions are interpreted and evaluated, and actions are executed or errors are raised.

Entry file / main (Oblig1.py):

Checks that the given command is correct and executes relevant test code, or displays usage guidelines if command is invalid.

How to run

Run using:

python Oblig1.py <program>

<program>: 1 | 2 | 3 | 4 | all

''Which test program to run. 1 – 4 runs selected program. all runs all test programs.