Kjetil Svalestuen / kjetisva
INF3110 Oblig 1

Design Document

Implementation

When working on the assignment, I focused on creating a modular structure that is readable and easy to expand on. I used a lot of datatypes to split the abstract syntax tree into logical, well defined parts. I edited the precode to support this with Program, Robot and Grid all being constructables.

The main interpret-function takes for input a program, which is divided into a grid and a robot, and runs an inner function that ultimately returns the resulting robot position. The inner function is recursive and loops through the Robot's statement list, assigning variables and moving the robot along the way before finally running into the stop command.

Each statement has its own pattern in the inner function, and some patterns runs a unique statement function (evalMove, evalAssignment, evalWhile). These again use the recursive evalExp-function, which has a pattern match for each type of expression. When finding an identifier expression, it calls the lookup function which searches for the given string in the var declaration list found in the AST.

My solution delays all expressions until the lookup function finally has to evaluate them. This makes for some lookups redundant, but may be the easiest implementation.

I do raise an OutOfBounds-exception when the robot falls off the grid, but do not handle this in any sofisticated way. The program will quit with an error, which I believe is a sufficient solution.

Running the Program

I've made AST representations of all the example programs in the assignment text. These should be easy to read, write and modify. You execute the program code by running the "interpret(program)"-function.