Lehrstuhl für Informatik 2
Modellierung und Verifikation von Software

Static Program Analysis WS2012/13
Sheet 1 (Hand in until 27.10.2014)

apl. Prof. Dr. Thomas Noll

Christian Dehnert, Christina Jansen

Exercise 1 (WHILE programming language): (5 Points)

Consider the following algorithm given by an informal description. The algorithm takes two integer variables, say x and y, as input and performs the following steps. If either x or y are non-positive, then terminate directly. Otherwise we keep doing the following operations until y becomes zero: set some temporary variable t to y, set y to the remainder of the division of x by y and then set x to the value of t.

a) What is the content of the variable x after the algorithm has terminated (i.e. which problem does the algorithm solve for positive inputs x and y)?

b) Provide an implementation of the algorithm in the WHILE programming language as presented in the lecture. You may assume that WHILE includes the inequality operator ($\neq$).

c) Extend your program of b) to a labelled WHILE-program.

d) Specify the $init(c)$- and $final(c)$-mapping for your program c.

e) Specify the $flow(c)$-relation of your program c and give its corresponding flow graph. Does the program have an isolated entry and/or isolated exits?

Exercise 2 (Available Expressions Analysis): (5 Points)

Extend the WHILE programming language of the lecture by a *do-while*-construct.

a) Adapt the $init$- and $final$-mapping as well as the $flow$-relation to capture the newly introduced construct.

b) Additionally, adapt all concepts needed to perform an available expression analysis on programs using the *do-while*-construct.

c) Perform an available expression analysis on the following program:

```
y := x + 1;
x := x + y;
do
  y := x + 1;
while (y < x);
y := y * x;
```