Functional Programming and Verification

Prof. Dr. H. Seidl, N. Hartmann, R. Vogler

WS 2018/19

Exercise Sheet 4

Deadline: 18.11.2018

General Information

Detailed information about the lecture, tutorials and homework assignments can be found on the lecture website[1]. Solutions have to be submitted to Moodle[2]. Make sure your uploaded documents are readable. Blurred images will be rejected. Use Piazza[3] to ask questions and discuss with your fellow students.

Simplifications

Make sure you simplify terms whenever possible. Overly complex proofs with huge formulas due to lack of any meaningful simplification will not be awarded with full points.

OCaml Setup

We are going to start OCaml programming in next week's exercises. Please prepare your machines accordingly and bring them to the sessions. A detailed installation guide will be published soon. Please check the website and Moodle.

Assignment 4.1 (L) Termination

In the lecture, you have learned how to prove termination of a MiniJava program. Discuss these questions:

1. How can you decide whether a termination proof is required at all?
2. What is the basic idea of the termination proof?
3. How has the program to be modified?
4. What has to be proven?
5. How is the loop invariant influenced?

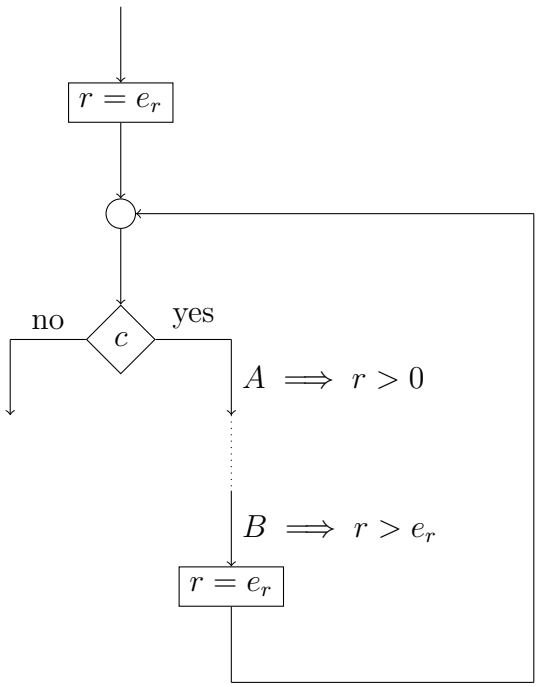
Suggested Solution 4.1

Consider this general structure of a loop:

[1] https://www.in.tum.de/i02/lehre/wintersemester-1819/vorlesungen/functional-programming-and-verification/

[2] https://www.moodle.tum.de/course/view.php?id=44932

[3] https://piazza.com/tum.de/fall2018/in0003/home



1. A loop-free program always terminates. If the program contains one or more loops, every single loop has to be considered. Classical `for`-style loops always terminate. Every other loop has to be checked.

2. A loop terminates if and only if the number of iterations is finite. Choose a variable of the program that is strictly increasing (or decreasing) in every single iteration. Then, if we can prove that there is an upper (or lower) bound the variable cannot reach, the loop must run only a finite number of iterations, because otherwise it would eventually reach the bound. The common strategy is to choose a variable that is decreasing by exactly 1 in every iteration and a lower bound of 0, although the proof might work with other values as well.

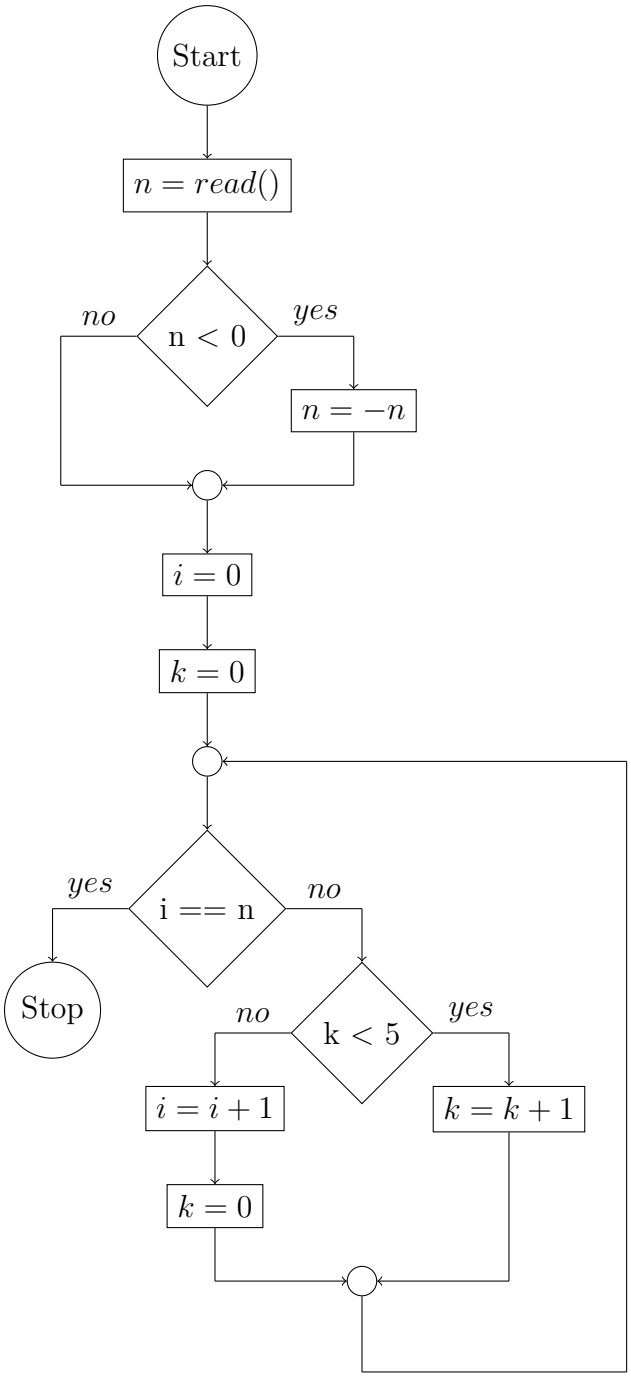
3. If the program does not provide a variable that satisfies these requirements, we introduce a new variable (typically named r) and compute it in such a way that it models this behavior before and after every iteration of the loop.

4. Now, we need to find locally consistent annotations in the program such that the assertion A at the beginning of the loop body enforces the lower bound ($A \implies r > 0$) and the assertion B at the end of the loop body (before the recomputation of r) guarantees that r has been decreased in every iteration ($r > e_r$, where e_r is the expression used to compute r).

5. The loop invariant typically contains $r = e_r$ as well as relations between the variables used in e_r.

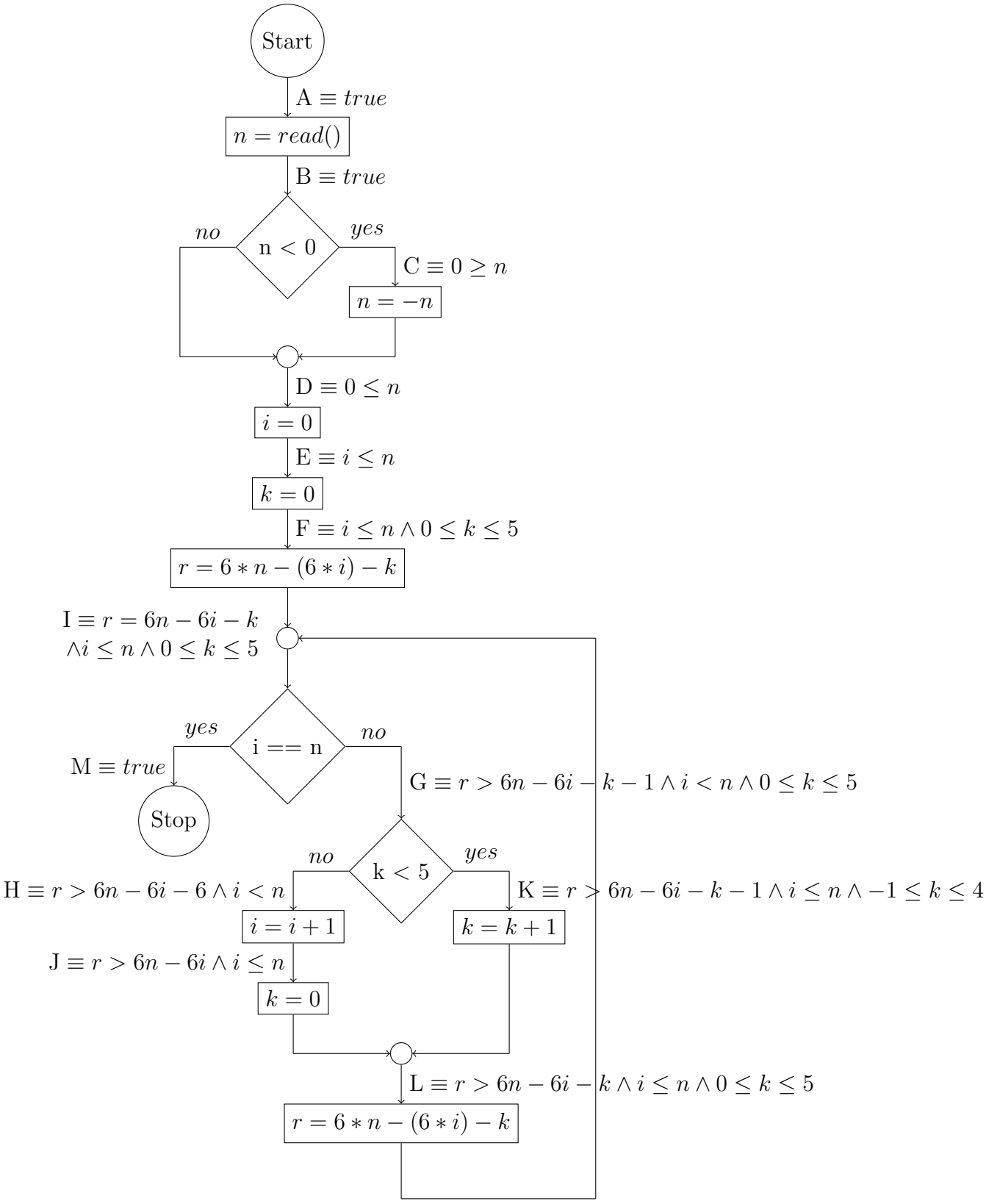
Assignment 4.2 (L) Counter Time

Prove that the following program does indeed terminate for all inputs.



Suggested Solution 4.2

We add an auxiliary variable $r = 6n - 6i - k$ to the control flow graph:



Now we have to find locally consistent annotations such that

- $G \implies r > 0$ and
- $L \implies r > 6n - 6i - k$

A suitable loop invariant is $I :\equiv r = 6n - 6i - k \wedge i \leq n \wedge 0 \leq k \leq 5$.

$$
\begin{aligned}
& \mathsf{WP}[\![\texttt{r = 6n - 6i - k}]\!](I) \\
\equiv\ & \mathsf{WP}[\![\texttt{r = 6n - 6i - k}]\!](r = 6n - 6i - k \wedge i \leq n \wedge 0 \leq k \leq 5) \\
\equiv\ & i \leq n \wedge 0 \leq k \leq 5 \\
\Longleftarrow\ & r > 6n - 6i - k \wedge i \leq n \wedge 0 \leq k \leq 5 \quad \equiv: L
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{WP}[\![\texttt{k = 0}]\!](L) \\
\equiv\ & \mathsf{WP}[\![\texttt{k = 0}]\!](r > 6n - 6i - k \wedge i \leq n \wedge 0 \leq k \leq 5) \\
\equiv\ & r > 6n - 6i \wedge i \leq n \quad \equiv: J
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{WP}[\![\texttt{i = i + 1}]\!](J) \\
\equiv\ & \mathsf{WP}[\![\texttt{i = i + 1}]\!](r > 6n - 6i \wedge i \leq n) \\
\equiv\ & r > 6n - 6i - 6 \wedge i < n \quad \equiv: H
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{WP}[\![\texttt{k = k + 1}]\!](L) \\
\equiv\ & \mathsf{WP}[\![\texttt{k = k + 1}]\!](r > 6n - 6i - k \wedge i \leq n \wedge 0 \leq k \leq 5) \\
\equiv\ & r > 6n - 6i - k - 1 \wedge i \leq n \wedge -1 \leq k \leq 4 \quad \equiv: K
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{WP}[\![\texttt{k < 5}]\!](H, K) \\
\equiv\ & \mathsf{WP}[\![\texttt{k < 5}]\!](r > 6n - 6i - 6 \wedge i < n, r > 6n - 6i - k - 1 \wedge i \leq n \wedge -1 \leq k \leq 4) \\
\equiv\ & (k \geq 5 \wedge r > 6n - 6i - 6 \wedge i < n) \vee (r > 6n - 6i - k - 1 \wedge i \leq n \wedge -1 \leq k \leq 4) \\
\Longleftarrow\ & (k = 5 \wedge r > 6n - 6i - k - 1 \wedge i < n) \vee (r > 6n - 6i - k - 1 \wedge i < n \wedge 0 \leq k \leq 4) \\
\equiv\ & r > 6n - 6i - k - 1 \wedge i < n \wedge 0 \leq k \leq 5 \quad \equiv: G
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{WP}[\![\texttt{i == n}]\!](G, N) \\
\equiv\ & \mathsf{WP}[\![\texttt{i == n}]\!](r > 6n - 6i - k - 1 \wedge i < n \wedge 0 \leq k \leq 5, \mathit{true}) \\
\equiv\ & (i \neq n \wedge r > 6n - 6i - k - 1 \wedge i < n \wedge 0 \leq k \leq 5) \vee (i = n) \\
\Longleftarrow\ & i \leq n \wedge r = 6n - 6i - k \wedge 0 \leq k \leq 5 \quad \equiv I
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{WP}[\![\texttt{r = 6n - 6i - k}]\!](I) \\
\equiv\ & \mathsf{WP}[\![\texttt{r = 6n - 6i - k}]\!](i \leq n \wedge r = 6n - 6i - k \wedge 0 \leq k \leq 5) \\
\equiv\ & i \leq n \wedge 0 \leq k \leq 5 \quad \equiv: F
\end{aligned}
$$

$$
\begin{aligned}
& \mathsf{WP}[\![\texttt{k = 0}]\!](F) \\
\equiv\ & \mathsf{WP}[\![\texttt{k = 0}]\!](i \leq n \wedge 0 \leq k \leq 5) \\
\equiv\ & i \leq n \quad \equiv: E
\end{aligned}
\qquad\qquad
\begin{aligned}
& \mathsf{WP}[\![\texttt{i = 0}]\!](E) \\
\equiv\ & \mathsf{WP}[\![\texttt{i = 0}]\!](i \leq n) \\
\equiv\ & 0 \leq n \quad \equiv: D
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{WP}[\![\texttt{n = -n}]\!](D)\\
\equiv\ &\mathsf{WP}[\![\texttt{n = -n}]\!](0 \le n)\\
\equiv\ &0 \le -n\\
\equiv\ &0 \ge n \quad \equiv: C
\end{aligned}
\qquad
\begin{aligned}
&\mathsf{WP}[\![\texttt{n < 0}]\!](D, C)\\
\equiv\ &\mathsf{WP}[\![\texttt{n < 0}]\!](0 \le n, 0 \ge n)\\
\equiv\ &(n \ge 0 \implies 0 \le n) \wedge (n < 0 \implies 0 \ge n)\\
\equiv\ &\mathit{true} \quad \equiv: B
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{WP}[\![\texttt{n = read()}]\!](B)\\
\equiv\ &\mathsf{WP}[\![\texttt{n = read()}]\!](\mathit{true})\\
\equiv\ &\mathit{true} \quad \equiv: A
\end{aligned}
$$

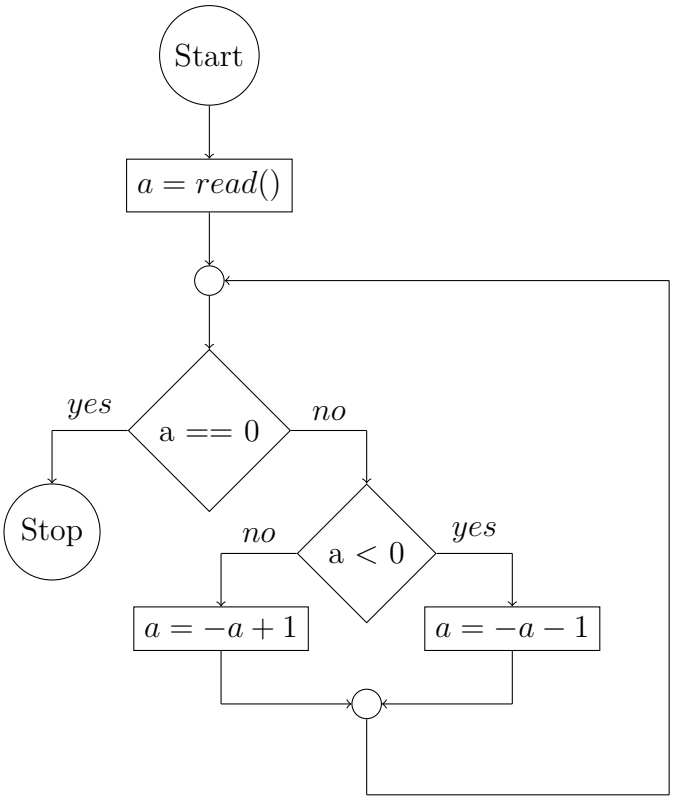
We finally check that for our locally consistent annotations

- $G \equiv r > 6n - 6i - k - 1 \wedge i < n \wedge 0 \le k \le 5 \implies r > 0$ and
- $L \equiv r > 6n - 6i - k \wedge i \le n \wedge 0 \le k \le 5 \implies r > 6n - 6i - k$

does indeed hold. □

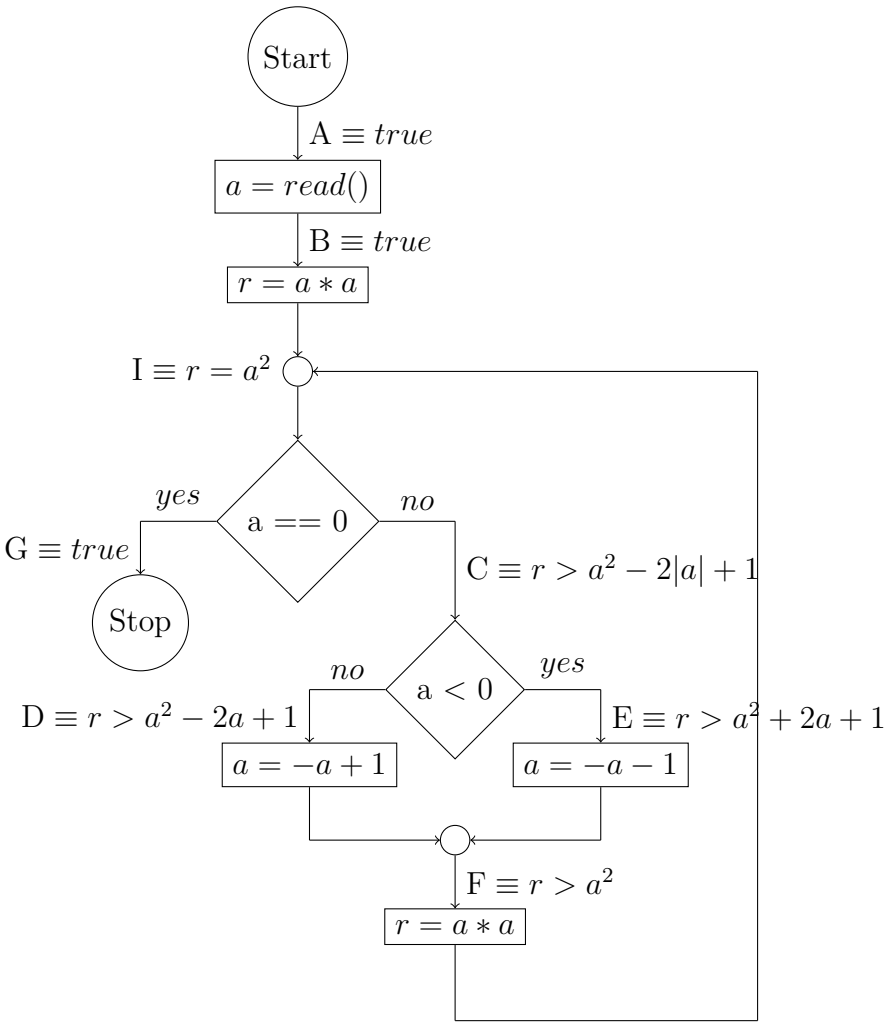
Assignment 4.3 (L) A Wavy Approach

Prove termination of the following program:



Suggested Solution 4.3

We insert the auxiliary variable $r = a * a$ into the control flow graph:



Then, we start with a loop invariant of $r = a^2$ and try to find locally consistent annotations such that

(1) $C \implies r > 0$ and

(2) $F \implies r > a^2$

are satisfied:

$$
\begin{aligned}
&\mathsf{WP}[\![\texttt{r = a * a}]\!](I) \\
\equiv\ &\mathsf{WP}[\![\texttt{r = a * a}]\!](r = a^2) \\
\equiv\ &\mathit{true} \\
\Longleftarrow\ &r > a^2 \quad \equiv: F
\end{aligned}
\qquad
\begin{aligned}
&\mathsf{WP}[\![\texttt{a = -a + 1}]\!](F) \\
\equiv\ &\mathsf{WP}[\![\texttt{a = -a + 1}]\!](r > a^2) \\
\equiv\ &r > a^2 - 2a + 1 \quad \equiv: D
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{WP}[\![\texttt{a = -a - 1}]\!](F) \\
\equiv\ &\mathsf{WP}[\![\texttt{a = -a - 1}]\!](r > a^2) \\
\equiv\ &r > a^2 + 2a + 1 \quad \equiv: E
\end{aligned}
\qquad
\begin{aligned}
&\mathsf{WP}[\![\texttt{a < 0}]\!](D, E) \\
\equiv\ &\mathsf{WP}[\![\texttt{a < 0}]\!](r > a^2 - 2a + 1, r > a^2 + 2a + 1) \\
\equiv\ &(a \geq 0 \wedge r > a^2 - 2a + 1) \vee (a < 0 \wedge r > a^2 + 2a + 1) \\
\equiv\ &r > a^2 - 2|a| + 1 \quad \equiv: C
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{WP}[\![\texttt{a == 0}]\!](C, G) \\
\equiv\ &\mathsf{WP}[\![\texttt{a == 0}]\!](r > a^2 - 2|a| + 1, true) \\
\equiv\ &(a \neq 0 \wedge r > a^2 - 2|a| + 1) \vee a = 0 \\
\Longleftarrow\ &r = a^2 \quad \equiv I
\end{aligned}
$$

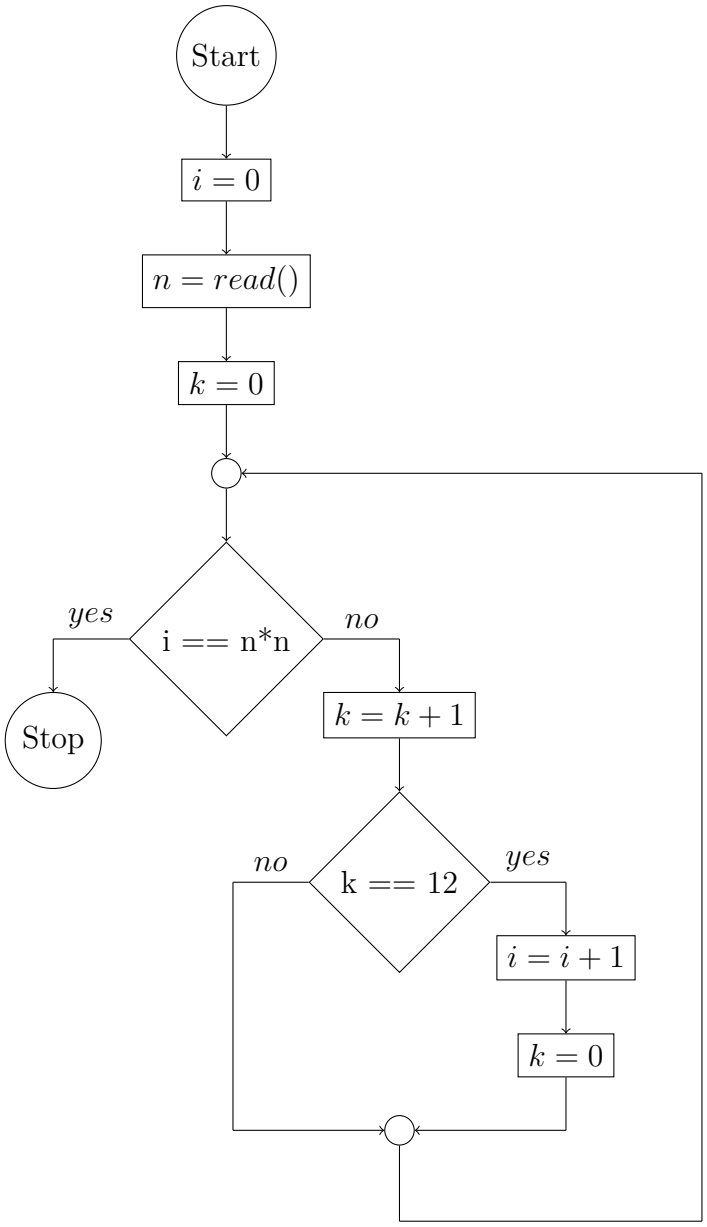
$$
\begin{aligned}
&\mathsf{WP}[\![\texttt{r = a * a}]\!](I) \\
\equiv\ &\mathsf{WP}[\![\texttt{r = a * a}]\!](r = a^2) \\
\equiv\ &true \quad \equiv: B
\end{aligned}
$$

$$
\begin{aligned}
&\mathsf{WP}[\![\texttt{a = read()}]\!](B) \\
\equiv\ &\mathsf{WP}[\![\texttt{a = read()}]\!](true) \\
\equiv\ &\texttt{true} \quad \equiv: A
\end{aligned}
$$

□

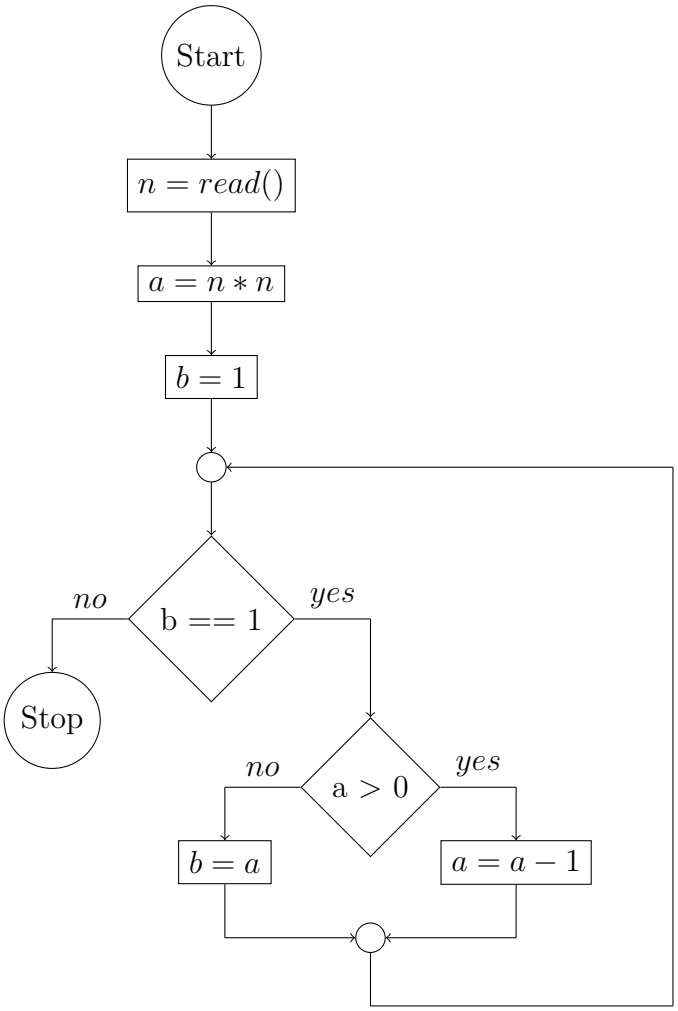
Assignment 4.4 (H) Counting Twelves [6 Points]

Prove termination of the following program:



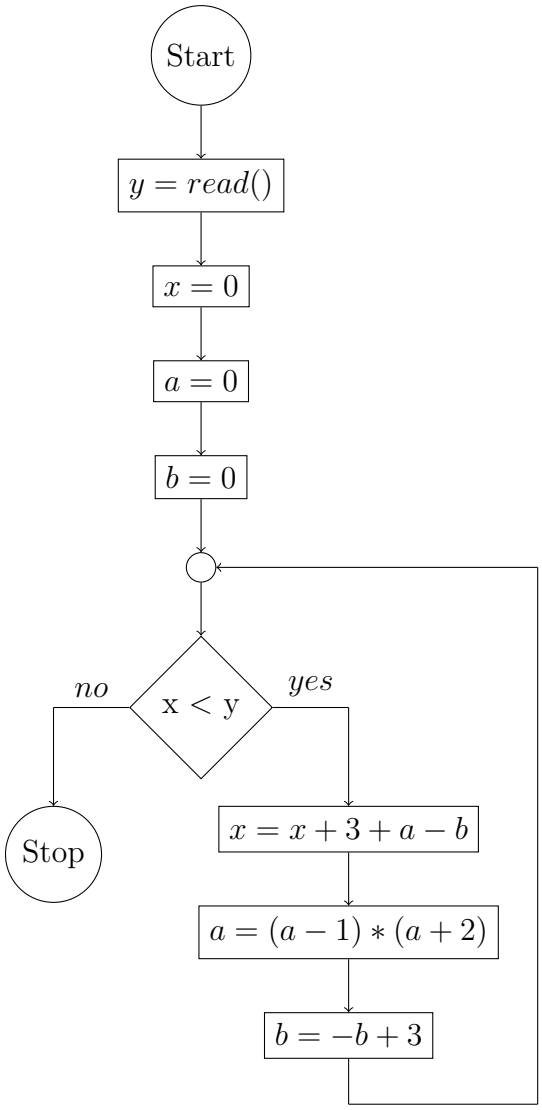
Assignment 4.5 (H) aaaaaaaaaab [4 Points]

Prove termination of the following program:



Assignment 4.6 (H) Term-ination [10 Points]

Prove that the following program terminates for all inputs:



Hint: Finding a formula for r is non-trivial in this program. Make sure you understand what the program is doing and consider writing down the values of variables for a couple of iterations.
Hint: Keep in mind, that r need not necessarily represent the exact number of remaining loop iterations, but a finite upper limit.