# Functional Programming and Verification

Prof. Dr. H. Seidl, N. Hartmann, R. Vogler WS 2018/19

**Exercise Sheet 3** Deadline: 11.11.2018

## 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.

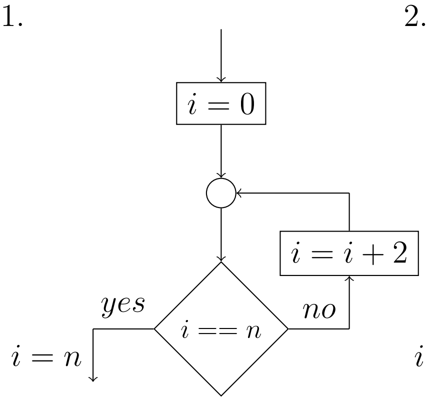
## Loop Invariants

In this exercise sheet, you will discuss and practice different strategies to find suitable loop invariants. For additional insight, tips and tricks on how to find a good invariant, we recommend watching the recording of last year's exercise on this particular topic[4].

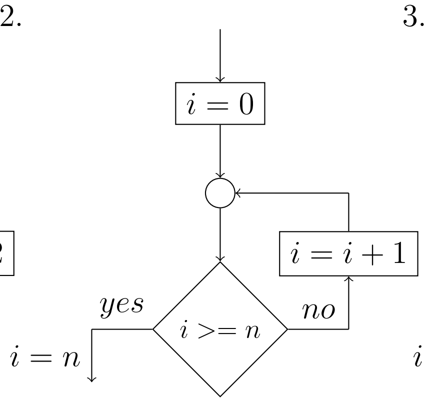
## Assignment 3.1 (L) Individual Loops

Inspect the following loops and discuss the preconditions that have to hold, such that the assertion $i = n$ is satisfied. In particular, discuss the results for positive and negative inputs, respectively.

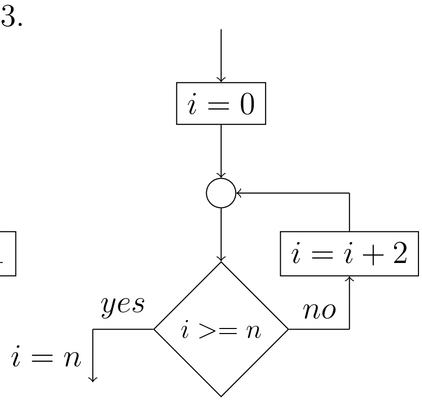
1.




2.




3.




[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`

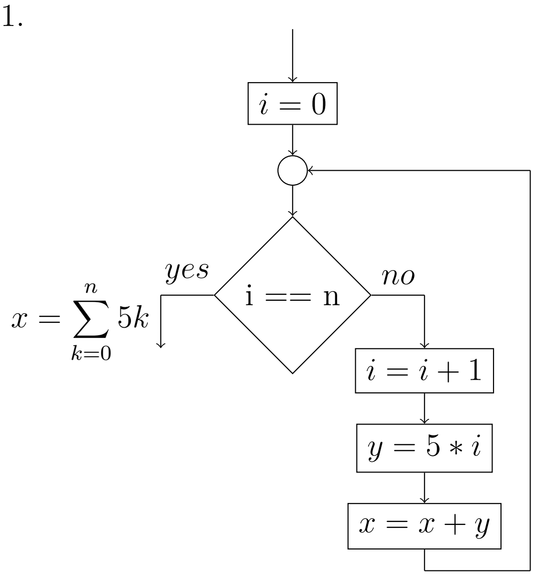
[3] `https://piazza.com/tum.de/fall2018/in0003/home`

[4] `http://ttt.in.tum.de/recordings/Info2_2017_11_24-1/Info2_2017_11_24-1.mp4`

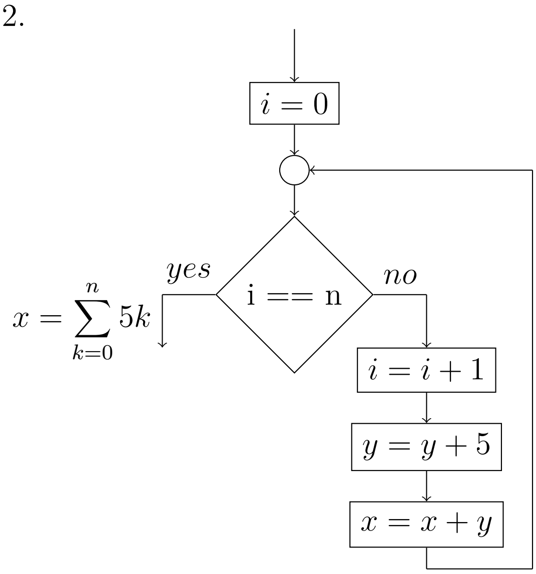
### Assignment 3.2 (L) Y?

Consider these control flow graph fragments (assume $x$ and $y$ to be 0 initially):

1.



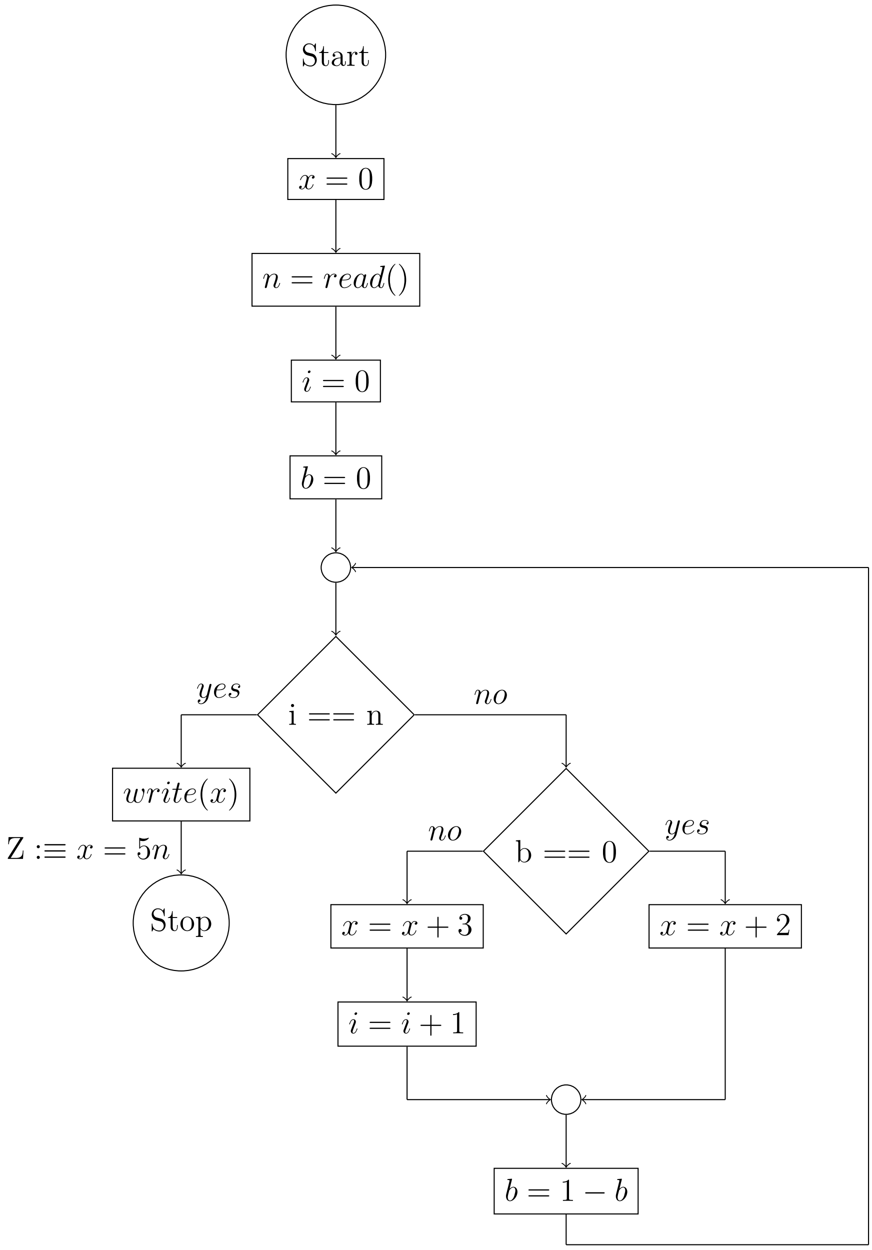

2.




Find suitable loop invariants and prove them locally consistent. Discuss, why these invariants have to be like that.

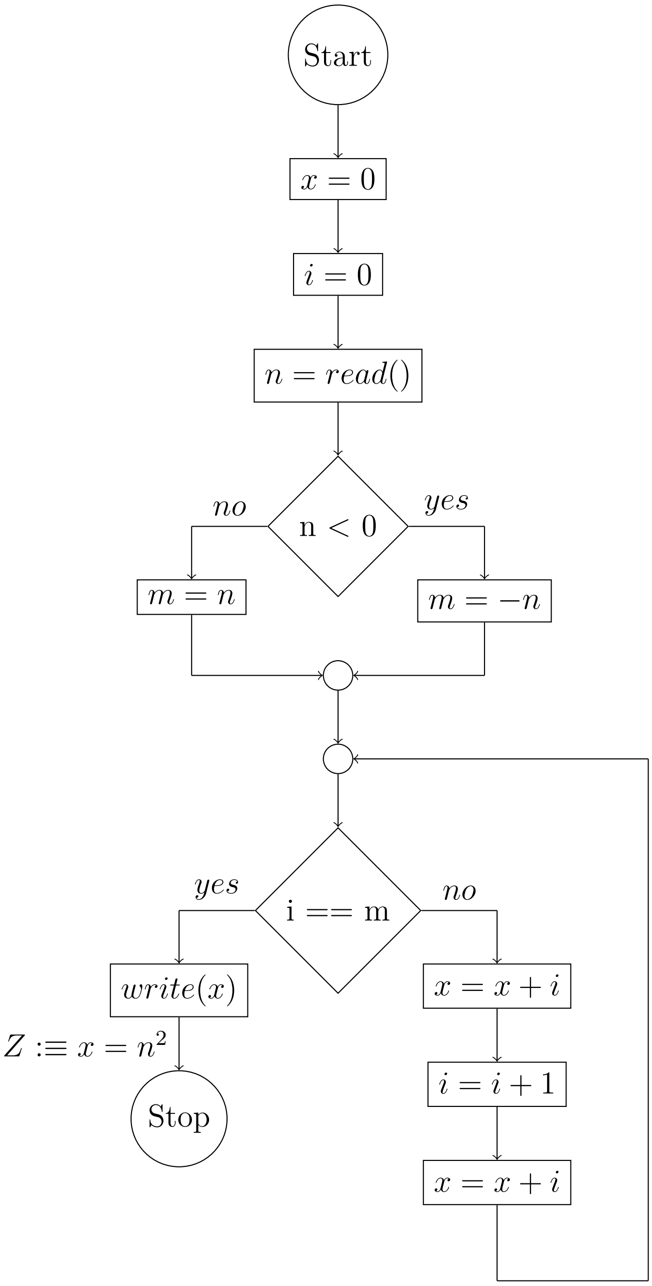
**Assignment 3.3 (L) Two b, or not two b**

Prove $Z$ using weakest preconditions.



**Assignment 3.4 (L) Squared**

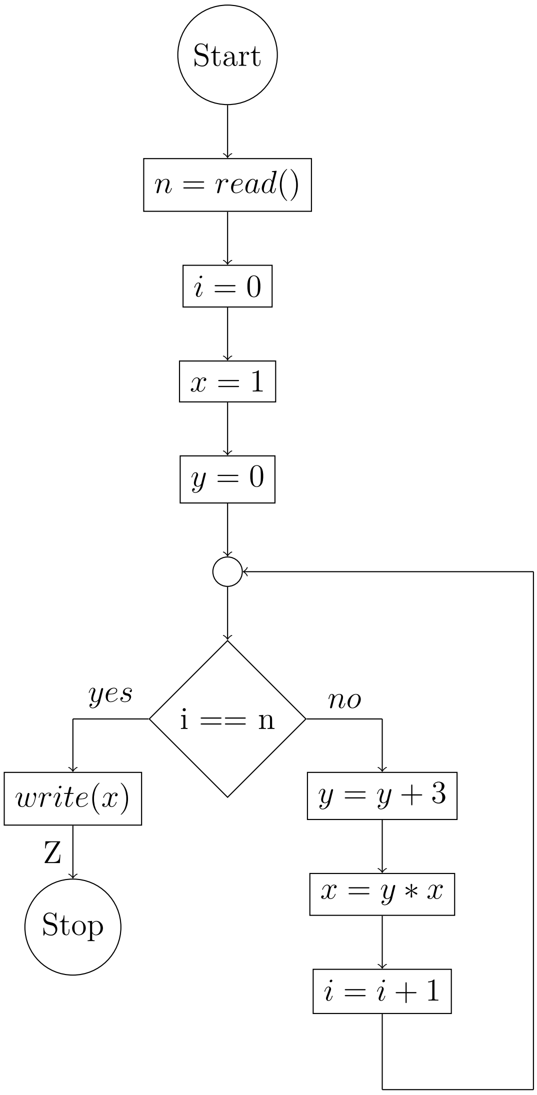
Given is the following control flow graph:




Prove that $Z$ holds.

**Assignment 3.5 (H) Ready, Z, go!** [6 Points]

Find a formula $Z$ to express the exact value $x$ the program computes. Then prove this $Z$ using weakest preconditions.




Das Programm liest den Input n ein. Dieser entspricht der Anzahl der Schleifendurchläufe. Innerhalb der Schleife wird x dann immer wieder mit y multipliziert. y wird dabei bei jedem Durchlauf um 3 inkrementiert. Das bedeutet, dass $x = \prod limits_{i=1}^{n} 3i$ berechnet wird. Dieses Produkt wählen wir als Schleifeninvariante I. Wenn wir uns dieses Produkt genauer anschauen, sehen wir, dass es $3^n$ ( $\prod\limits_{i=1}^{n} 3$ ) sowie $n!$ ( $\prod\limits_{i=1}^{n} i$ ) berechnet,das Produkt also $n! \cdot 3^n$ entspricht. $\rightarrow Z :\equiv x = n! \cdot 3^n$

$$
\begin{aligned}
\mathsf{WP}[\![\texttt{write(x)}]\!](Z) &\equiv \mathsf{WP}[\![\texttt{write(x)}]\!](n! \cdot 3^n) \\
&:\equiv A
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{WP}[\![\texttt{i = i + 1}]\!](I) &\equiv \mathsf{WP}[\![\texttt{i = i + 1}]\!](x = \prod_{i=1}^{n} y \wedge y = 3i) \\
&\equiv x = \prod_{(i+1)=1}^{n} y \wedge y = 3(i+1) \\
&\equiv x = \prod_{i=0}^{n} y \wedge y = 3i + 3 \\
&:\equiv B
\end{aligned}
$$

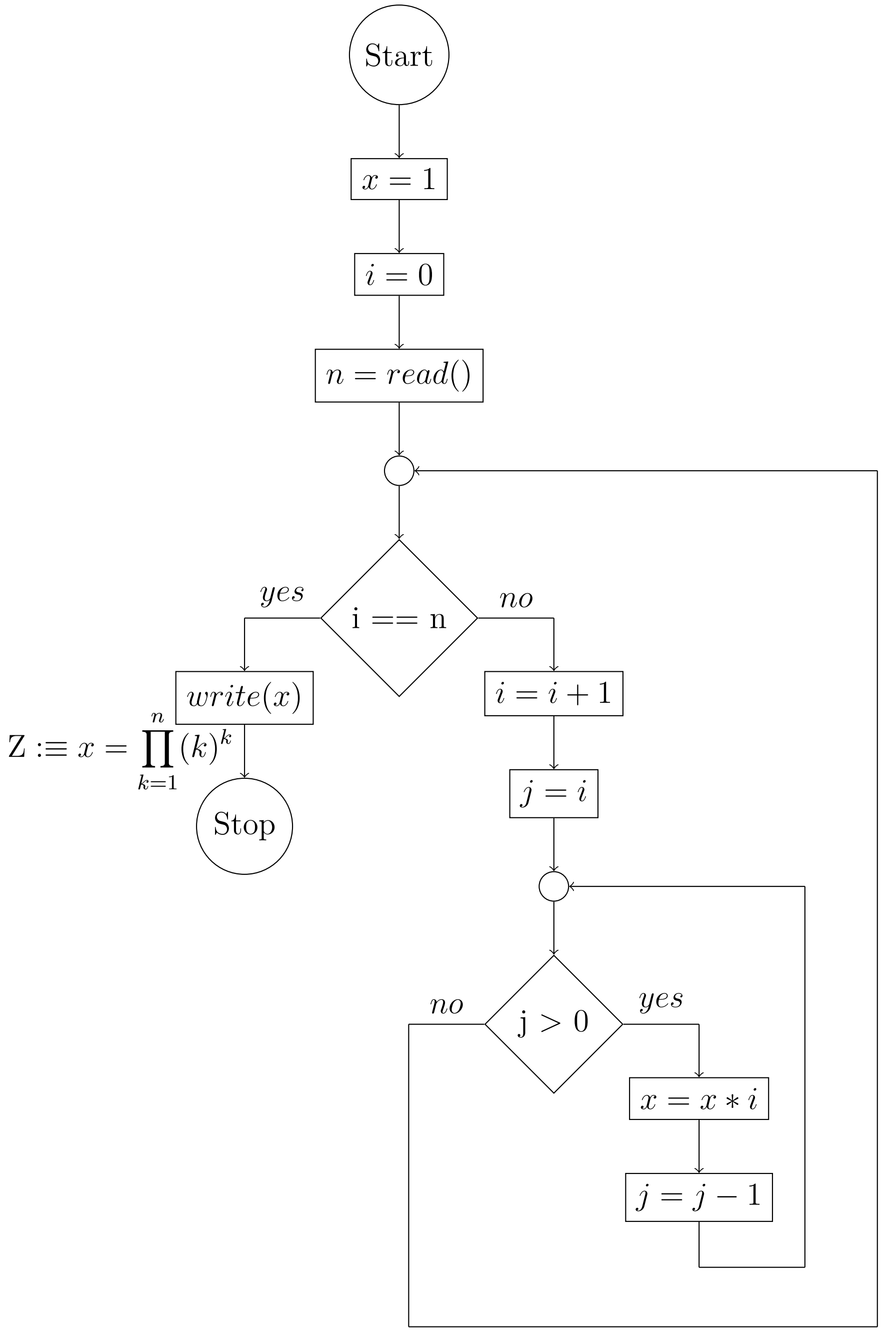
$$
\begin{aligned}
\mathsf{WP}[\![\texttt{x = y * x}]\!](B) &\equiv \mathsf{WP}[\![\texttt{x = y * x}]\!](x = \prod_{i=0}^{n} y \wedge y = 3i + 3) \\
&\equiv y * x = \prod_{i=0}^{n} y \wedge y = 3i + 3 \\
&\equiv (3i+3) * x = \prod_{i=0}^{n} (3i+3) \wedge y = 3i + 3 \\
&\equiv (3i+3) * x = 3 * \prod_{i=1}^{n} (3i+3) \wedge y = 3i + 3 \\
&\equiv (3i+3) * x = 3 * \prod_{i=1}^{n} (3i+3) \wedge y = 3i + 3 \\
&\equiv y * x = 3^{(n+1)} * y^(n+1) \wedge y = 3i + 3 \\
&\equiv x = 3^{(n+1)} * y^n \wedge y = 3i + 3 \\
&\equiv x = 3 * (3y)^n \wedge y = 3i + 3 \\
&:\equiv C
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{WP}[\![\texttt{y = y + 3}]\!](C) &\equiv \mathsf{WP}[\![\texttt{y = y + 3}]\!](x = 3 * (3y)^n \wedge y = 3i + 3) \\
&\equiv x = 3^n * (y+3)^(n-1) \wedge y + 3 = 3i + 3 \\
&\equiv x = 3 * 3^(n-1) * (y+3)^(n-1) \wedge y = 3i \\
&\equiv x = 3 * 3(y+3)^(n-1) \wedge y = 3i \\
&\equiv x = 3 * 3(y+3)^(n-1) \wedge y = 3i \\
&:\equiv D
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{WP}[\![\texttt{i == n}]\!](D, A) &\equiv \mathsf{WP}[\![\texttt{i == n}]\!](x = \prod_{i=0}^{n} 3 \wedge y = 3i, x = n! * 3^n) \\
&\equiv (i \neq n \wedge x = \prod_{i=0}^{n} 3 \wedge y = 3i) \vee (i = n \wedge x = n! * 3^n) \\
&\equiv (i \neq n \wedge x = 3^n \wedge y = 3i) \vee (i = n \wedge x = i! * 3^i)
\end{aligned}
$$

**Assignment 3.6 (H) Loloopop** [8 Points]

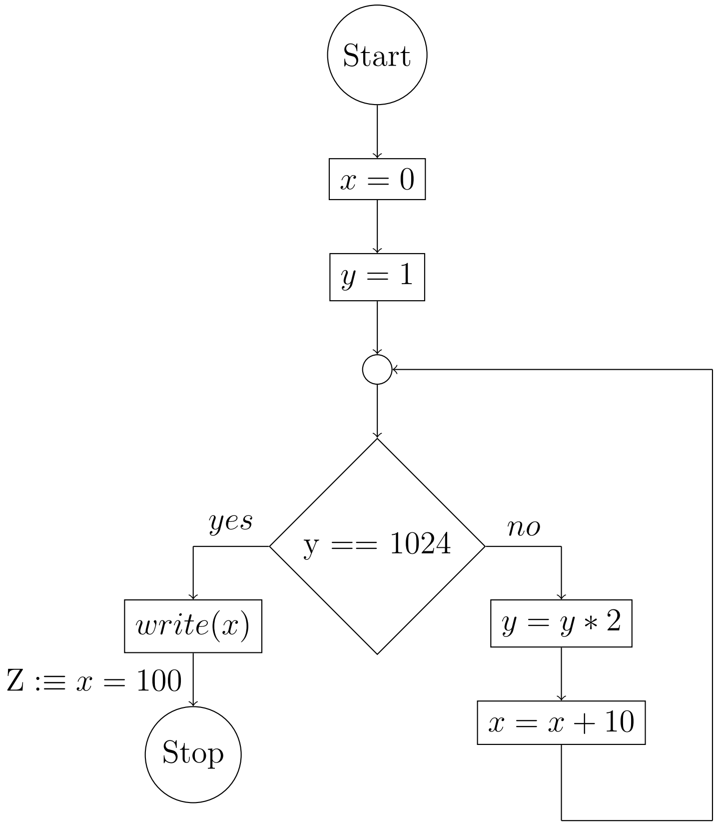
Prove $Z$ using weakest preconditions:




*Hinweis: If you have to find invariants for nested loops, it is usually easiest to work from outermost loop to innermost loop.*

**Assignment 3.7 (H) Somethng s wrong wth ths program...** [3 Points]

Prove $Z$ using weakest preconditions.




**Assignment 3.8 (H) A Neverending Story** [3 Points]

Prove that the following program cannot terminate using weakest preconditions.