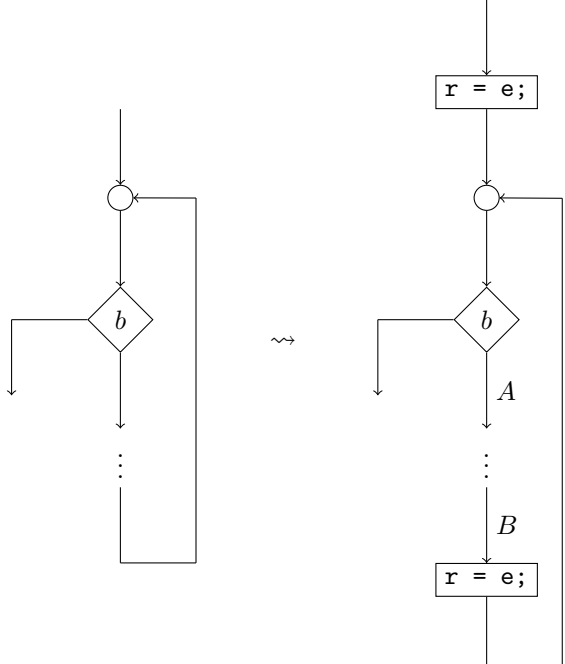


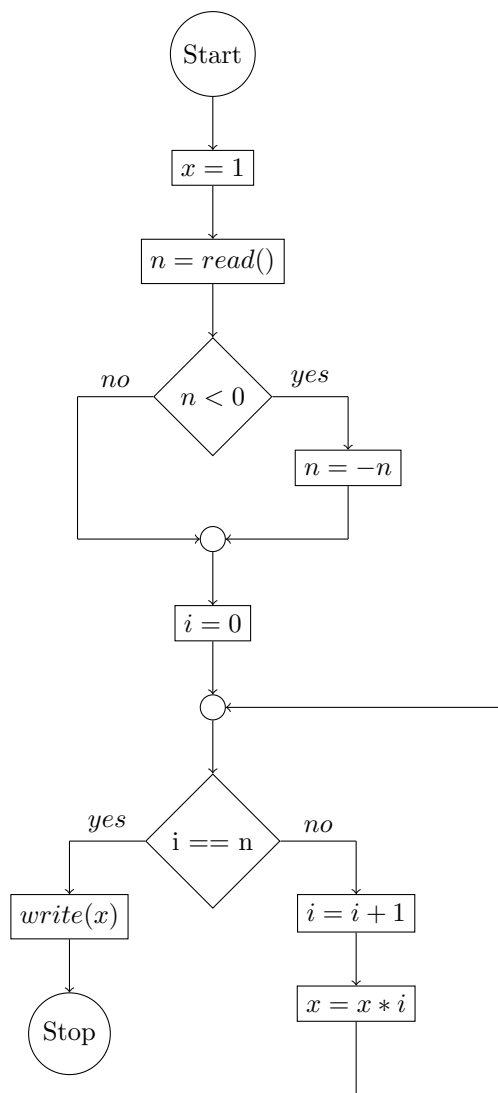
Program Modification	Proof Obligation
	<p>Annotate all program points with assertions s.t.</p> <ul style="list-style-type: none"> • the assertions are locally consistent • the first assertion is true • $A \implies r > 0$ (lower bound) • $B \implies r > e$ (strictly decreasing) <p><i>The lower bound can be any constant (not only 0). Alternatively, prove an upper bound and show that r is strictly increasing.</i></p>

4.1 Termination (2021, T4.3)

See <https://artemis.ase.in.tum.de/courses/147/exercises/5491>.

4.2 Termination without Modification (2017, T4.1)

Show that the following program terminates for all user inputs.



Hint: It is not necessary to introduce a new variable. Use what the program already provides.

Answer of exercise 4.2

In order to show that the program terminates, we have to show for a variable that inside the loop

- (1) the variable get strictly larger (or smaller) each iteration.
- (2) the variable does not exceed an upper (or lower) bound.

In this particular program, i can serve as such a variable, since (1) i increases every loop iteration (2) it is bounded above by n . Since i is only modified exactly once in the loop body, the following approach is valid. Consider the program point immediately after entering the loop and the program point immediately before updating i . In this example, these coincide, namely at the *no*-edge of the condition. Call this point B . We have to annotate the program locally consistent, s.t. the first program point is annotated with true and

- (1) $B \implies i < i + 1$
- (2) $B \implies i < n$

Now, we choose an invariant. Informations regarding x are irrelevant for that, since we are not interested in what the program computes and x is irrelevant for termination. More important: between i and n , we have the relation $i \leq n$. So, choose the invariant $I \equiv i \leq n$. Now, we calculate:

$$\begin{aligned}
 \text{WP}[x = x * i](I) &\equiv \text{WP}[x = x * i](i \leq n) \\
 &\equiv i \leq n \equiv A \\
 \text{WP}[i = i + 1](A) &\equiv \text{WP}[i = i + 1](i \leq n) \\
 &\equiv i + 1 \leq n \\
 &\equiv i < n \equiv B
 \end{aligned}$$

$$\begin{aligned}
\text{WP}[i == n](B, \text{true}) &\equiv \text{WP}[i == n](i < n, \text{true}) \\
&\equiv (i \neq n \wedge i < n) \vee (i = n \wedge \text{true}) \\
&\equiv i \leq n \equiv I \\
\text{WP}[i = 0](I) &\equiv \text{WP}[i = 0](i \leq n) \\
&\equiv 0 \leq n \equiv: C \\
\text{WP}[n = -n](C) &\equiv \text{WP}[n = -n](0 \leq n) \\
&\equiv 0 \leq -n \\
&\equiv 0 \geq n \equiv: D \\
\text{WP}[n < 0](C, D) &\equiv \text{WP}[n < 0](0 \leq n, 0 \geq n) \\
&\equiv (n \geq 0 \implies 0 \leq n) \wedge (n < 0 \implies 0 \geq n) \\
&\equiv \text{true} \equiv: E \\
\text{WP}[n = \text{read()}](E) &\equiv \text{WP}[n = \text{read()}](\text{true}) \\
&\equiv \forall n. \text{true} \\
&\equiv \text{true} \equiv: F \\
\text{WP}[x = 1](F) &\equiv \text{WP}[x = 1](\text{true}) \\
&\equiv \text{true} \equiv: G
\end{aligned}$$

So, we have locally consistent annotations that hold for every program execution (since the first program point is annotated with true). Let us reconsider our initial question:

- (1) $B \implies i < i + 1$
- (2) $B \implies i < n$

Both implications hold trivially, and thus it is shown that the program always terminates.

4.3 Termination (2017, T4.2)

Show that the following program always terminates by using the method described in the lecture.

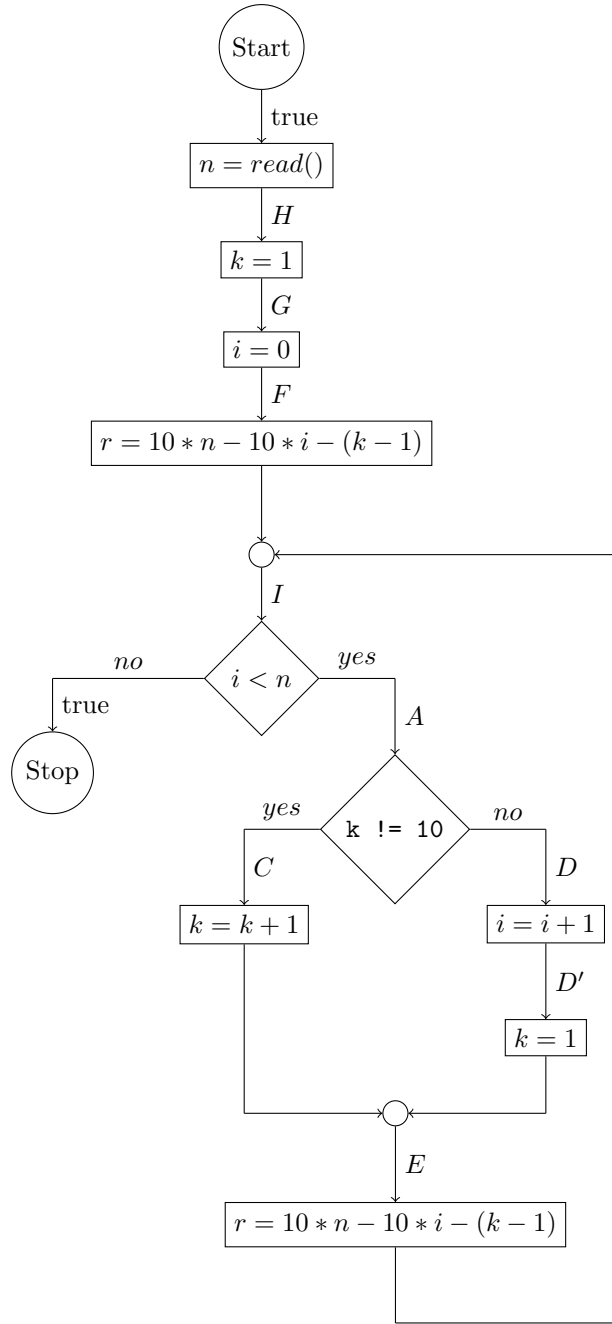
```

n = read ();
k = 1;
i = 0;
while (i < n){
    if (k != 10)
        k = k + 1;
    else {
        i = i + 1;
        k = 1;
    }
}

```

Answer of exercise 4.3

We use the loop measure $r = 10n - 10i - (k - 1)$. First, draw the control flow diagram of the modified program:



We guess our loop invariant as $I \equiv r = 10n - 10i - (k - 1) \wedge k \leq 10$. Now, we need to annotate all program points s.t. (1) $A \implies r > 0$ and (2) $E \implies r > 10n - 10i - (k - 1)$. So, begin with

$$\begin{aligned} \text{WP}[r = 10*n - 10*i - (k - 1)](I) &\equiv k \leq 10 \\ &\iff k \leq 10 \wedge r > 10n - 10i - (k - 1) \equiv E \end{aligned}$$

Hence, (2) is satisfied. Now, continue our annotations:

$$\begin{aligned} \text{WP}[k = 1](E) &\equiv r > 10n - 10i \equiv D' \\ \text{WP}[i = i + 1](D') &\equiv r > 10n - 10i - 10 \equiv D \\ \text{WP}[k = k + 1](E) &\equiv r \geq 10n - 10i - (k - 1) \wedge k < 10 \equiv C \\ \text{WP}[k \neq 10](D, C) &\equiv (k = 10 \wedge D) \vee (k \neq 10 \wedge C) \\ &\equiv \dots \\ &\equiv k \leq 10 \wedge r \geq 10n - 10i - (k - 1) \\ &\iff k \leq 10 \wedge r = 10n - 10i - (k - 1) \wedge i < n \equiv A \end{aligned}$$

where we chose A s.t. (1) follows. Note that $A \equiv I \wedge i < n$. Now, the loop is completed, and we need to verify that the annotations at the loop condition are locally consistent:

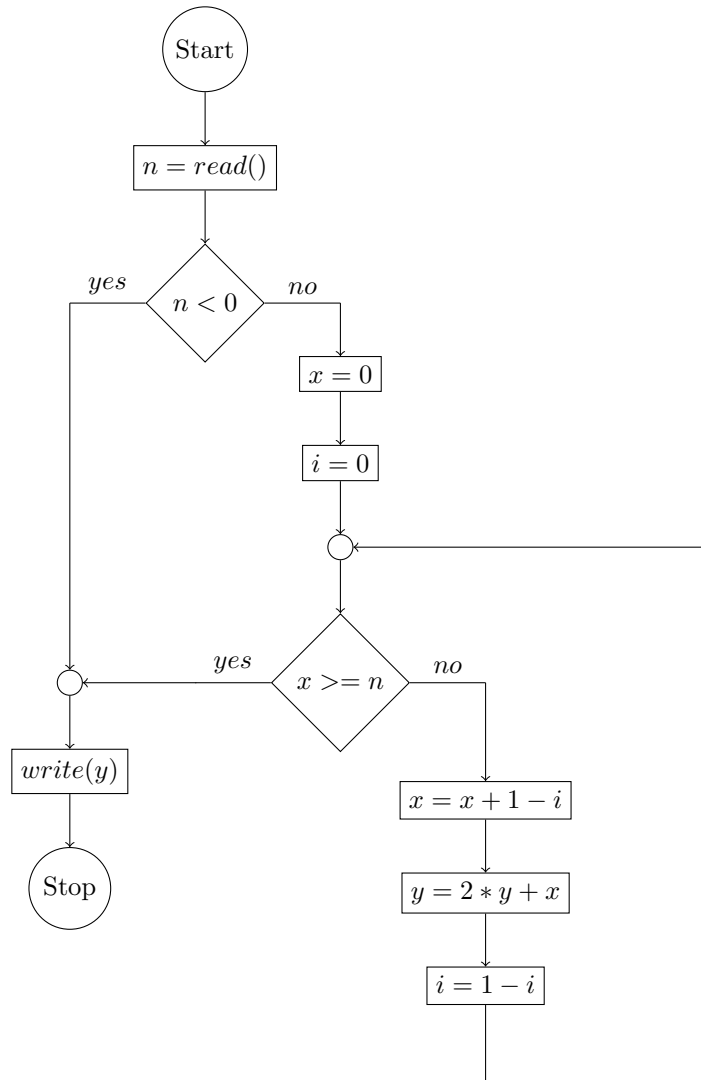
$$\begin{aligned}
 \text{WP}[\![i < n]\!](\text{true}, A) &\equiv i \geq n \vee (i < n \wedge A) \\
 &\equiv i \geq n \vee (i < n \wedge I) \\
 &\Leftarrow (i \geq n \wedge I) \vee (i < n \wedge I) \\
 &\equiv I
 \end{aligned}$$

Finally, we only need to push I through the first few statements:

$$\begin{aligned}
 \text{WP}[\![r = 10 * n - 10 * i - (k - 1)]\!](I) &\equiv k \leq 10 \equiv F \\
 \text{WP}[\![i = 0]\!](F) &\equiv k \leq 10 \equiv G \\
 \text{WP}[\![k = 1]\!](G) &\equiv 1 \leq 10 \equiv \text{true} \equiv H \\
 \text{WP}[\![n = \text{read}()]\!](H) &\equiv \forall n. \text{true} \equiv \text{true}
 \end{aligned}$$

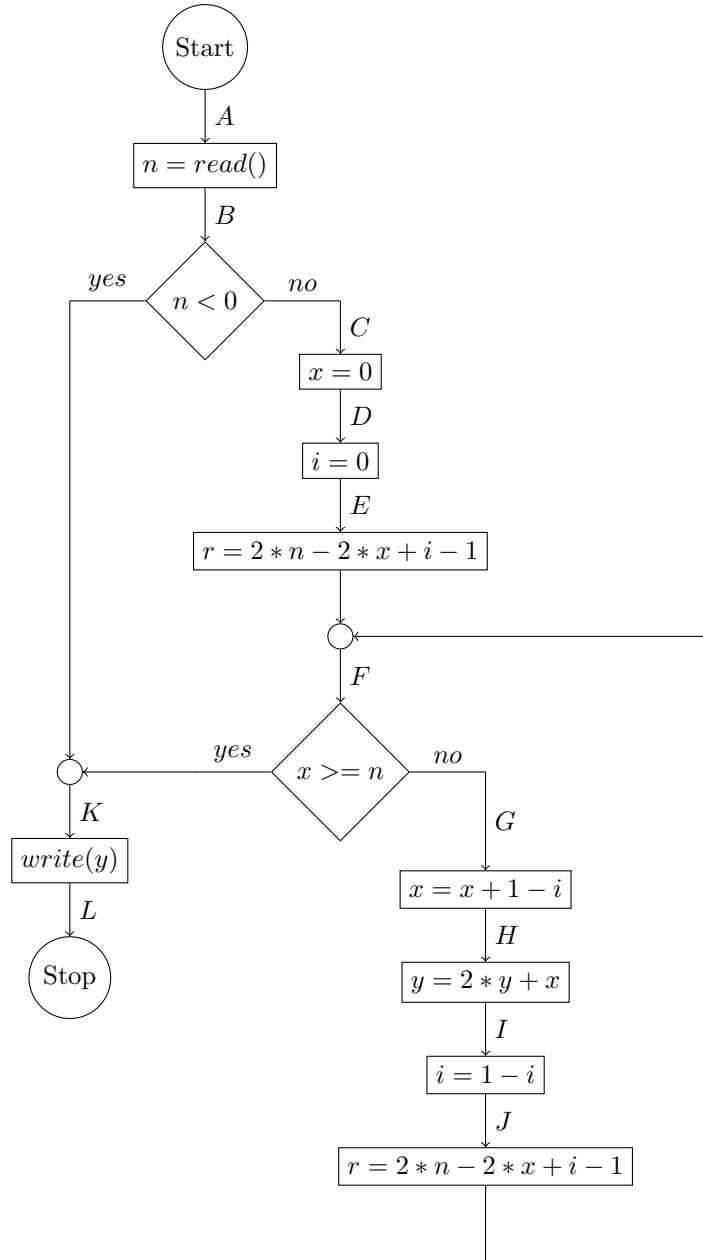
4.4 Termination II (2017, H4.6)

Show that the following program terminates for all inputs.



Answer of exercise 4.4

We use $r = 2n - 2x + i - 1$ as loop measure. First, modify the control flow graph:



Now, we need locally consistent annotations, s.t.:

$$(1) \ G \implies r > 0$$

$$(2) \ J \implies r > 2n - 2x + i - 1$$

As invariant, we choose $F := r = 2n - 2x + i - 1 \wedge i \in \{0, 1\}$. Moreover, set $L := K := \text{true}$, since we don't want to state anything about the program result. We calculate:

$$\begin{aligned} \text{WP}[r = 2*n - 2*x + i - 1](F) &\equiv \text{WP}[r = 2*n - 2*x + i - 1](r = 2n - 2x + i - 1 \wedge i \in \{0, 1\}) \\ &\equiv i \in \{0, 1\} \end{aligned}$$

$$\iff r > 2n - 2x + i - 1 \wedge i \in \{0, 1\} \equiv J$$

$$\text{WP}[i = 1 - i](J) \equiv \text{WP}[i = 1 - i](r > 2n - 2x + i - 1 \wedge i \in \{0, 1\})$$

$$\equiv r > 2n - 2x + 1 - i - 1 \wedge i - 1 \in \{0, 1\}$$

$$\equiv r > 2n - 2x - i \wedge i \in \{0, 1\} \equiv I$$

$$\text{WP}[y = 2*y + x](I) \equiv \text{WP}[y = 2*y + x](r > 2n - 2x - i \wedge i \in \{0, 1\})$$

$$\equiv r > 2n - 2x - i \wedge i \in \{0, 1\} \equiv H$$

$$\text{WP}[x = x + 1 - i](H) \equiv \text{WP}[x = x + 1 - i](r > 2n - 2x - i \wedge i \in \{0, 1\})$$

$$\equiv r > 2n - 2(x + 1 - i) - i \wedge i \in \{0, 1\}$$

$$\begin{aligned}
&\equiv r > 2n - 2x - 2 + i \wedge i \in \{0, 1\} \\
&\Leftarrow r > 2n - 2x - 2 + i \wedge i \in \{0, 1\} \wedge x < n \equiv: G \\
\text{WP}[\mathbf{x} \geq \mathbf{n}](G, K) &\equiv \text{WP}[\mathbf{x} \geq \mathbf{n}](r > 2n - 2x - 2 + i \wedge i \in \{0, 1\} \wedge x < n, \text{true}) \\
&\equiv (x < n \implies r > 2n - 2x - 2 + i \wedge i \in \{0, 1\} \wedge x < n) \wedge (x \geq n \implies \text{true}) \\
&\equiv x < n \implies r > 2n - 2x - 2 + i \wedge i \in \{0, 1\} \\
&\Leftarrow r = 2n - 2x + i - 1 \wedge i \in \{0, 1\} \equiv F \\
\text{WP}[\mathbf{r} = 2*\mathbf{n} - 2*\mathbf{x} + \mathbf{i} - 1](F) &\equiv \text{WP}[\mathbf{r} = 2*\mathbf{n} - 2*\mathbf{x} + \mathbf{i} - 1](r = 2n - 2x + i - 1 \wedge i \in \{0, 1\}) \\
&\equiv 2n - 2x + i - 1 = 2n - 2xi - 1 \wedge i \in \{0, 1\} \\
&\equiv i \in \{0, 1\} \equiv: E \\
\text{WP}[\mathbf{i} = 0](E) &\equiv \text{WP}[\mathbf{i} = 0](i \in \{0, 1\}) \\
&\equiv 0 \in \{0, 1\} \\
&\equiv \text{true} \equiv: D \\
\text{WP}[\mathbf{x} = 0](D) &\equiv \text{WP}[\mathbf{x} = 0](\text{true}) \\
&\equiv \text{true} \equiv: C \\
\text{WP}[\mathbf{n} < 0](C, K) &\equiv \text{WP}[\mathbf{n} < 0](\text{true}, \text{true}) \\
&\equiv (n \geq 0 \implies \text{true}) \wedge (n < 0 \implies \text{true}) \\
&\equiv \text{true} \equiv: B \\
\text{WP}[\mathbf{n} = \text{read}()](B) &\equiv \text{WP}[\mathbf{n} = \text{read}()](\text{true}) \\
&\equiv \forall n. \text{true} \\
&\equiv \text{true} \equiv: A
\end{aligned}$$

So, we have locally consistent annotations and annotated the start node with true. It remains to verify our conditions (1) and (2):

$$\begin{aligned}
G &\equiv r > 2n - 2x - 2 + i \wedge i \in \{0, 1\} \wedge x < n \\
&\equiv r > 2n - 2x - 2 + i \geq 2 - 2 + i = i \geq 0 \wedge i \in \{0, 1\} \wedge x < n \\
&\implies r > 0 \\
J &\equiv r > 2n - 2x + i - 1 \wedge i \in \{0, 1\} \\
&\implies r > 2n - 2x + i - 1
\end{aligned}$$

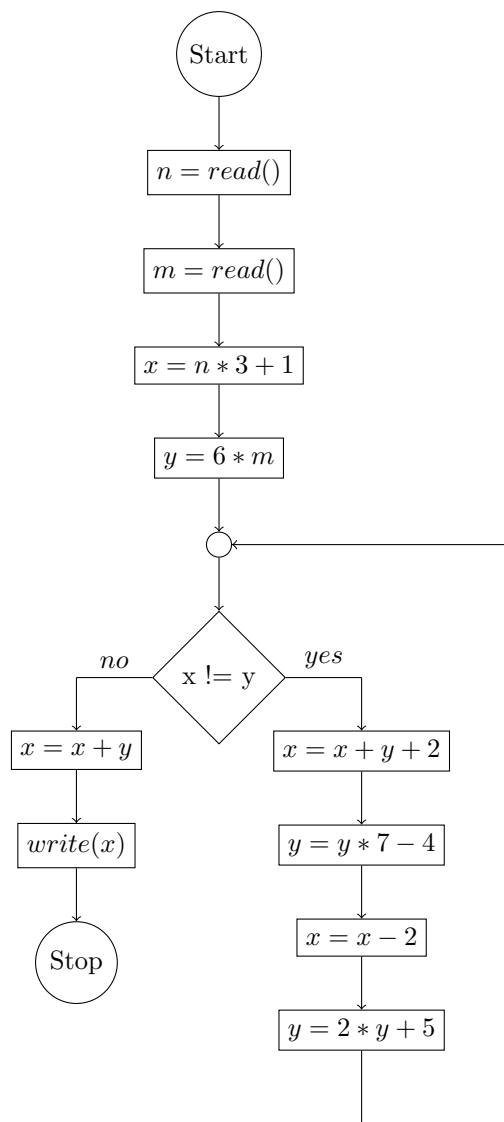
After this final verification, we have proven that the program terminates.

4.5 Non-Termination (2021, H3.4)

See <https://artemis.ase.in.tum.de/courses/147/exercises/5424>.

4.6 Non-Termination II (2017, H4.8)

Show that the following program never terminates.



Hint: Instead of considering all actual values of the variables, it can be helpful to consider other properties as well. What can you conclude from the initializations of x and y , even if the values of m and n are unknown?

Answer of exercise 4.6

In order to prove non-termination, we cannot use the approach for termination. Instead, we claim that $Z \equiv \text{false}$ holds at the stop node and try to prove that this holds for all program executions. For this, it must be shown that there exist locally consistent annotations of assertions (with false at the stop node) with true at the start node.

We look at our program and note two things:

- There is no loop counter variable.
- We don't know anything about the exact values of x and y , since they directly depend on the unknown inputs n and m .

Because of these reasons, it is impossible to find an invariant that makes a statement about the exact values of x and y . Instead, we consider their divisibility by 3. After initialization, it holds that $x \equiv_3 1 \wedge y \equiv_3 0$ (where $a \equiv_n b$ means $a \equiv b \pmod n$, i.e. $a - b \in n\mathbb{Z} = \{nk \mid k \in \mathbb{Z}\}$). So, we try to prove the statement by using the relatively weak invariant $I \equiv x \equiv_3 1 \wedge y \equiv_3 0$. We start our calculations¹:

$$\text{WP}[\text{write}(x)](Z) \equiv \text{false} \equiv: A$$

$$\text{WP}[x = x + y](A) \equiv \text{false} \equiv: B$$

¹Note: the calculations frequently use rules of arithmetic in \mathbb{Z}_n . Most importantly: in \mathbb{Z}_n , we have $x + a = b \iff x = b - a$, where $-a$ is the additive inverse of a in \mathbb{Z}_n , and $ax = b \iff x = a^{-1}b$ whenever $a \in \mathbb{Z}_n^\times$ is an invertible element and a^{-1} is its (multiplicative) inverse. For example, $2x \equiv_3 1 \iff x \equiv_3 2^{-1} \cdot 1 = 2 \cdot 1 = 2$.

$$\begin{aligned}
\text{WP}[y = 2*y + 5](I) &\equiv x \equiv_3 1 \wedge 2y + 5 \equiv_3 0 \\
&\equiv x \equiv_3 1 \wedge y \equiv_3 2 \equiv: C \\
\text{WP}[x = x - 2](C) &\equiv x - 2 \equiv_3 1 \wedge y \equiv_3 2 \\
&\equiv x \equiv_3 2 \wedge y \equiv_3 2 \equiv: D \\
\text{WP}[y = y*7 - 4](D) &\equiv x \equiv_3 0 \wedge 7y - 4 \equiv_3 2 \\
&\equiv x \equiv_3 0 \wedge y \equiv_3 0 \equiv: E \\
\text{WP}[x = x + y + 2](E) &\equiv x + y + 2 \equiv_3 0 \wedge y \equiv_3 0 \\
&\equiv x \equiv_3 1 \wedge y \equiv_3 0 \equiv: F \\
\text{WP}[x \neq y](B, F) &\equiv (x = y \wedge \text{false}) \vee (x \neq y \wedge x \equiv_3 1 \wedge y \equiv_3 0) \\
&\equiv x \neq y \wedge x \equiv_3 1 \wedge y \equiv_3 0 \\
&\equiv x \equiv_3 1 \wedge y \equiv_3 0 & (\text{since } x \not\equiv_3 y \implies x \neq y) \\
&\equiv I \\
\text{WP}[y = 6 * m](I) &\equiv x \equiv_3 1 \wedge 6m \equiv_3 0 \\
&\equiv x \equiv_3 1 \equiv: G & (\text{since } 6m \equiv_3 0 \text{ holds}) \\
\text{WP}[x = n*3 + 1](G) &\equiv 3n + 1 \equiv_3 1 \\
&\equiv \text{true} \equiv: H \\
\text{WP}[m = \text{read()}](H) &\equiv \forall m. \text{true} \\
&\equiv \text{true} \equiv: J \\
\text{WP}[n = \text{read()}](J) &\equiv \forall n. \text{true} \\
&\equiv \text{true} \equiv: K
\end{aligned}$$