WP-operator		
Assignment	$\mathsf{WP}[\![\texttt{x = e;}]\!](A)$	$:\equiv A[e/x]$
Read	$\mathsf{WP}[\![\texttt{x = read();}]\!](A)$	$:\equiv \forall x.A$
Write	$\mathsf{WP}[\![\texttt{write(x);}]\!](A)$	$:\equiv A$
Conditional branch	$\mathsf{WP}[\![\texttt{b}]\!](B_0, B_1)$	$:\equiv (\neg b \wedge B_0) \vee (b \wedge B_1)$

2.1 WP-operator

For each of the following WP-expressions,

(i) write the result after application of the WP-operator **without further simplification**.
(ii) simplify the result as much as possible.
(a) $\mathsf{WP}[\![\texttt{x = read();}]\!](x^2 \geq 0)$
(b) $\mathsf{WP}[\![\texttt{x = read();}]\!](x \geq 0)$
(c) $\mathsf{WP}[\![\texttt{x = read();}]\!](y \geq 0)$
(d) $\mathsf{WP}[\![\texttt{write(x);}]\!](x > 0 \vee x = 0)$
(e) $\mathsf{WP}[\![\texttt{x = y + 2;}]\!](z + 2 = x)$
(f) $\mathsf{WP}[\![\texttt{x = 3;}]\!](y - z \geq 0)$
(g) $\mathsf{WP}[\![\texttt{x = y + 1;}]\!](y - 1 > 0)$
(h) $\mathsf{WP}[\![\texttt{x = y*y + z;}]\!](x > z \wedge x < y)$
(i) $\mathsf{WP}[\![\texttt{x > y}]\!](\text{true}, \text{false})$
(j) $\mathsf{WP}[\![\texttt{x < 0}]\!](x > 0, x \leq 0)$

Answer of exercise 2.1

(a)

$$\begin{aligned}\mathsf{WP}[\![\texttt{x = read();}]\!](x^2 \geq 0) &\equiv \forall x. x^2 \geq 0 \\ &\equiv \text{true}\end{aligned}$$

(b)

$$\begin{aligned}\mathsf{WP}[\![\texttt{x = read();}]\!](x \geq 0) &\equiv \forall x. x \geq 0 \\ &\equiv \text{false}\end{aligned}$$

(c)

$$\begin{aligned}\mathsf{WP}[\![\texttt{x = read();}]\!](y \geq 0) &\equiv \forall x. y \geq 0 \\ &\equiv y \geq 0\end{aligned}$$

(d)

$$\begin{aligned}\mathsf{WP}[\![\texttt{write(x);}]\!](x > 0 \vee x = 0) &\equiv x > 0 \vee x = 0 \\ &\equiv x \geq 0\end{aligned}$$

(e)

$$\begin{aligned}\mathsf{WP}[\![\texttt{x = y + 2;}]\!](z + 2 = x) &\equiv z + 2 = y + 2 \\ &\equiv z = y\end{aligned}$$

(f)

$$\begin{aligned}\mathsf{WP}[\![\texttt{x = 3;}]\!](y - z \geq 0) &\equiv y - z \geq 0 \\ &\equiv y \geq z && \text{(also } y - z \geq 0 \text{ is possible)}\end{aligned}$$

(g)

$$\begin{aligned}\mathsf{WP}[\![\texttt{x = y + 1;}]\!](y - 1 > 0) &\equiv y - 1 > 0 \\ &\equiv y > 1\end{aligned}$$

(h)

$$\begin{aligned}\mathsf{WP}[\![\texttt{x = y*y + z;}]\!](x > z \land x < y) &\equiv y^2 + z > z \land y^2 + z < y \\ &\equiv y^2 \geq 0 \land y^2 - y + z < 0 \\ &\equiv y^2 - y + z < 0\end{aligned}$$

This could be further simplified by analyzing the parabola $y \mapsto y^2 - y + z$, but this is not necessary.

(i)

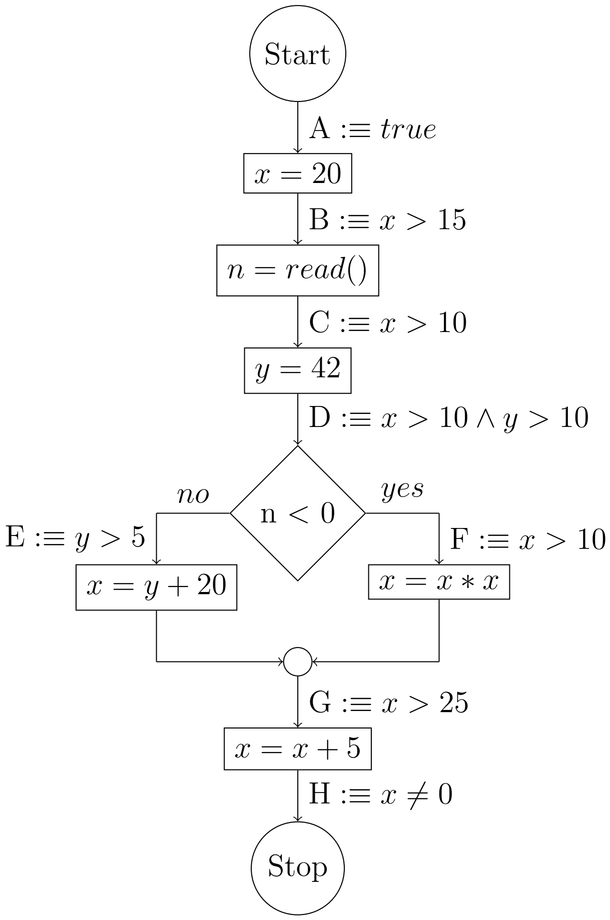
$$\begin{aligned}\mathsf{WP}[\![\texttt{x > y}]\!](\text{true}, \text{false}) &\equiv (\neg x > y \land \text{true}) \lor (x > y \land \text{false}) \\ &\equiv x \leq y \lor \text{false} \equiv x \leq y\end{aligned}$$

(j)

$$\begin{aligned}\mathsf{WP}[\![\texttt{x < 0}]\!](x > 0, x \leq 0) &\equiv (\neg x < 0 \land x > 0) \lor (x < 0 \land x \leq 0) \\ &\equiv x > 0 \lor x < 0 \equiv x \neq 0\end{aligned}$$

2.2 Local Consistency (*2021, T2.2*)

Link to Exercise: `https://artemis.ase.in.tum.de/courses/147/exercises/5346`.
In the following control flow graph assertions are annotated to all the edges.



Check whether the annotated assertions prove that the program computes an $x \neq 0$ and discuss why this is the case.

Answer of exercise 2.2

See `https://artemis.ase.in.tum.de/courses/147/exercises/5346`.

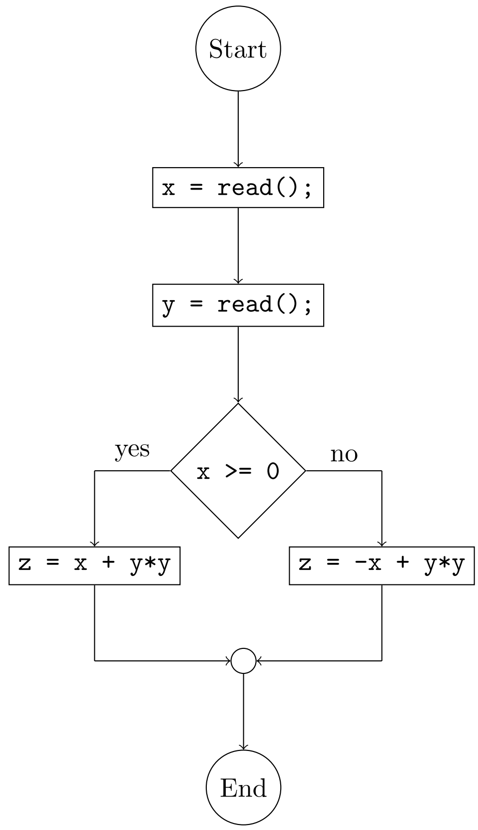
2.3 Writing and verifying a simple program

(a) Write a program in MiniJava that reads two integers x, y from the user input and returns $|x| + y^2$ in a third variable z. Use auxiliary variables if necessary.
(b) Draw the control flow graph of your program.
(c) Annotate the edge to the end point in the graph with an assertion fitting the specification of the program.
(d) Use the WP-operator to annotate all other program points.
(e) Do you still need to verify local consistency? Why or why not?

Answer of exercise $\boxed{\textbf{2.3}}$

(a)
```
x = read();
y = read();
if (x >= 0) {
        z = x + y*y;
} else {
        z = -x + y*y;
}
```

(b)



(c) The annotation is $z = |x| + y^2$.
(d) The annotations (top to bottom) are true, true, true, $x \geq 0$ resp. $x \leq 0$, $z = |x| + y^2$:

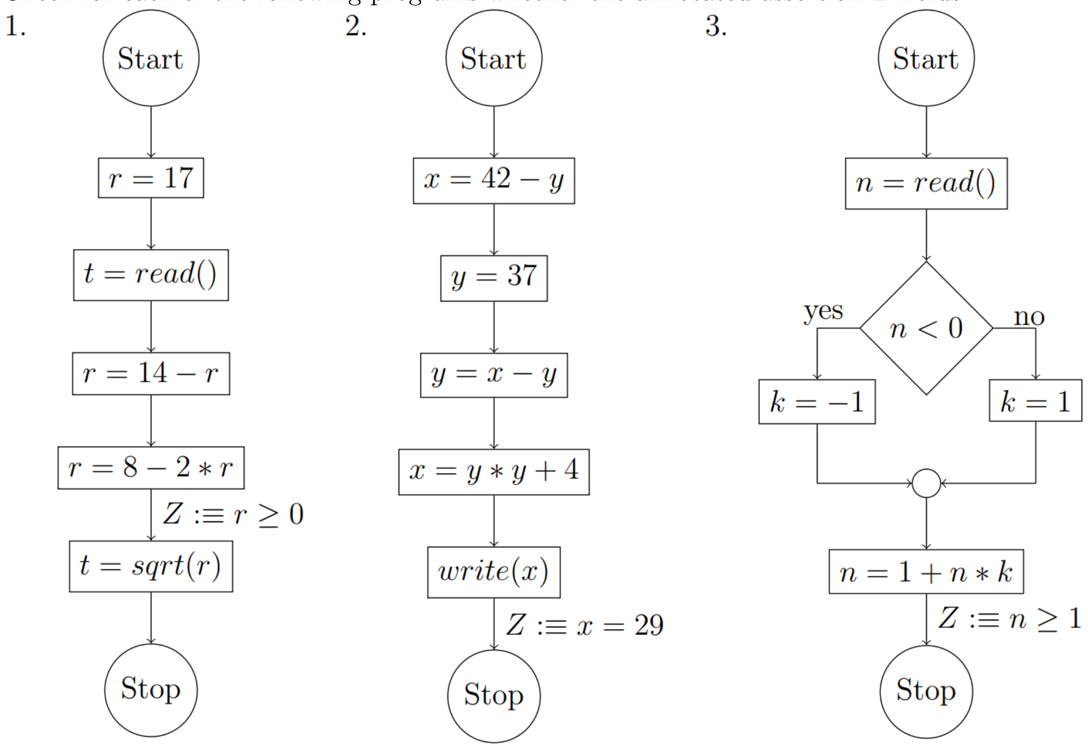
$$\begin{aligned}
\mathsf{WP}[\![\texttt{z = x + y*y}]\!](z = |x| + y^2) &\equiv x + y^2 = |x| + y^2 \\
&\equiv x = |x| \\
&\equiv x \geq 0 \\
\mathsf{WP}[\![\texttt{z = -x + y*y}]\!](z = |x| + y^2) &\equiv -x + y^2 = |x| + y^2 \\
&\equiv -x = |x| \\
&\equiv x \leq 0 \\
\mathsf{WP}[\![\texttt{x >= 0}]\!](x \leq 0, x \geq 0) &\equiv (x < 0 \wedge x \leq 0) \vee (x \geq 0 \wedge x \geq 0) \\
&\equiv x < 0 \vee x \geq 0 \\
&\equiv \text{true}
\end{aligned}$$

$$\begin{aligned}
\mathsf{WP}[\![\texttt{y = read();}]\!](\text{true}) &\equiv \forall y.\, \text{true} \\
&\equiv \text{true} \\
\mathsf{WP}[\![\texttt{x = read();}]\!](\text{true}) &\equiv \forall x.\, \text{true} \\
&\equiv \text{true}
\end{aligned}$$

(e) Local consistency does not need to be checked, since we annotated all program points with the weakest precondition, which always satisfies local consistency.

$\boxed{\textbf{2.4}}$ Weakest Preconditions (*2017, T2.1*)

Check for each of the following programs whether the annotated assertion Z holds.



Answer of exercise $\boxed{\textbf{2.4}}$

We just push the final assertions through using the WP-operator.

1.

$$\begin{aligned}
\mathsf{WP}[\![\texttt{r = 8 - 2*r}]\!](Z) &\equiv 8 - 2r \geq 0 \\
&\equiv 4 \geq r \\
\mathsf{WP}[\![\texttt{r = 14 - r}]\!](4 \geq r) &\equiv 4 \geq 14 - r \\
&\equiv r \geq 10 \\
\mathsf{WP}[\![\texttt{t = read()}]\!](r \geq 10) &\equiv \forall t. r \geq 10 \\
&\equiv r \geq 10 \\
\mathsf{WP}[\![\texttt{r = 17}]\!](r \geq 10) &\equiv 17 \geq 10 \\
&\equiv \text{true}
\end{aligned}$$

Hence, the assertion Z always holds.

2.

$$\begin{aligned}
\mathsf{WP}[\![\texttt{write(x)}]\!](Z) &\equiv Z \\
\mathsf{WP}[\![\texttt{x = y*y + 4}]\!](Z) &\equiv y^2 + 4 = 29 \\
&\equiv y \in \{-5, 5\} \\
\mathsf{WP}[\![\texttt{y = x - y}]\!](y \in \{-5, 5\}) &\equiv x - y \in \{-5, 5\} \\
\mathsf{WP}[\![\texttt{y = 37}]\!](x - y \in \{-5, 5\}) &\equiv x - 37 \in \{-5, 5\}
\end{aligned}$$

$$\begin{aligned} &\equiv x \in \{32, 42\} \\ \mathsf{WP}[\![\texttt{x = 42 - y}]\!](x \in \{32, 42\}) &\equiv 42 - y \in \{32, 42\} \\ &\equiv y \in \{0, 10\} \end{aligned}$$

Hence, the assertion Z does not always hold.

3.

$$\begin{aligned} \mathsf{WP}[\![\texttt{n = 1 + n*k}]\!](Z) &\equiv 1 + nk \geq 1 \\ &\equiv nk \geq 0 \\ \mathsf{WP}[\![\texttt{k = -1}]\!](nk \geq 0) &\equiv -n \geq 0 \\ &\equiv n \leq 0 \\ \mathsf{WP}[\![\texttt{k = 1}]\!](nk \geq 0) &\equiv n \geq 0 \\ \mathsf{WP}[\![\texttt{n < 0}]\!](n \geq 0, n \leq 0) &\equiv (n \geq 0 \land n \geq 0) \lor (n < 0 \land n \leq 0) \\ &\equiv n \geq 0 \lor n < 0 \\ &\equiv \text{true} \\ \mathsf{WP}[\![\texttt{n = read()}]\!](\text{true}) &\equiv \forall n.\, \text{true} \\ &\equiv \text{true} \end{aligned}$$

Hence, the assertion Z always holds.

2.5 Defining WP (*2021, H2.4*)

See `https://artemis.ase.in.tum.de/courses/147/exercises/5351`.

2.6 Defining WP II (*2017, H3.8*)

Let MiniJava++ be a MiniJava-extension with the following new statements:

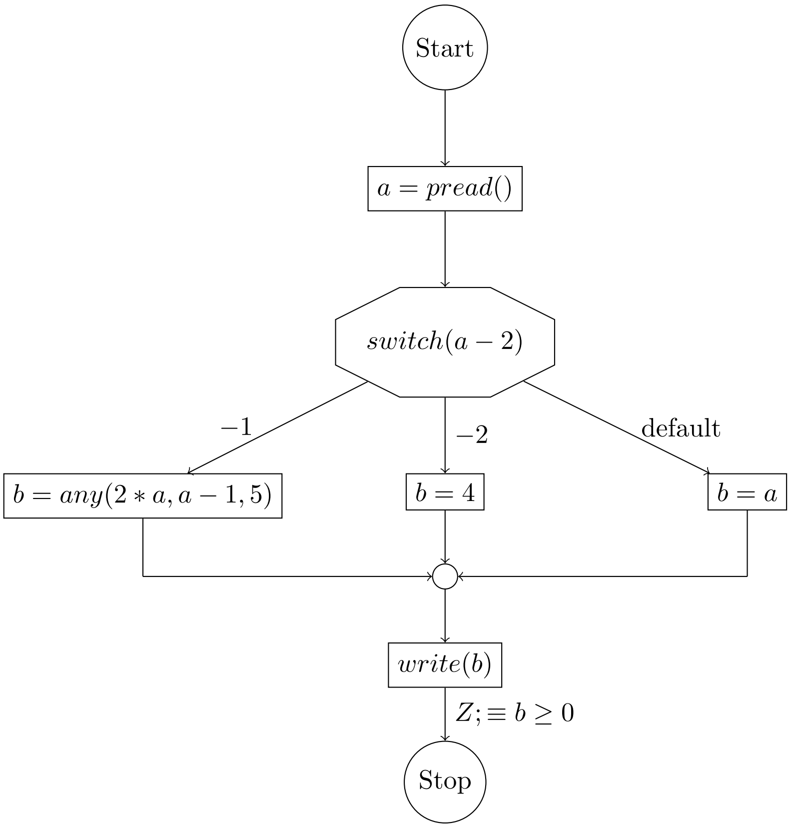
- x = pread();
- ```
  switch(e) {
          case v0: // ...
          case v1: // ...
          case v2: // ...
          // ...
          default: // ...
  }
  ```
- x = any(e_1, e_2, /* ... */, e_n);

Here, x is an arbitrary variable, $e, e_1, e_2, \ldots, e_n$ arbitrary expressions and $v_0, v_1, v_2, \ldots$ arbitrary values with $i \neq j \implies v_i \neq v_j$.
The statement `pread` reads a non-negative number v ($v \geq 0$). The `switch`-construct behaves as it does in Java, except that no explicit break is needed, but the execution of a `case`-branch ends automatically when the next `case`- or `default`-label or the next closing bracket is reached. Inside any `switch`-block, all labels have different values and there is at most one `default`-label. The `any`-statement chooses an expression from a given list of expressions non-deterministically and assigns its value to the variable on the left hand side.

(a) Define the WP-operator $\mathsf{WP}[\![\ ]\!](\ldots)$ for the three statements.[1]

(b) In the following MiniJava++ program, show that assertion Z holds.

[1]Note: if you already solved exercise **2.5**, you should have seen the `any`-construct already.
  ```



Answer of exercise $\boxed{2.6}$

(a)
- $\mathsf{WP}[\![\texttt{x = pread();}]\!](A) :\equiv \forall x \geq 0.A$
- $\mathsf{WP}[\![\texttt{s}]\!](A_0, \ldots, A_n, A_d) :\equiv (e = v_0 \implies A_0) \wedge \cdots \wedge (e = v_n \implies A_n) \wedge (e \neq v_0 \wedge \cdots \wedge e \neq v_n \implies A_d)$ (where s denotes a `switch`-statement as given)
- $\mathsf{WP}[\![\texttt{x = any(}e_0, ..., e_n\texttt{);}]\!](A) :\equiv A[e_0/x] \wedge \cdots \wedge A[e_n/x]$

(b) We use these definitions in order to verify the program. Since no loops occur, we can simply push Z through by using the WP-operator and check that the initial assertion is true.

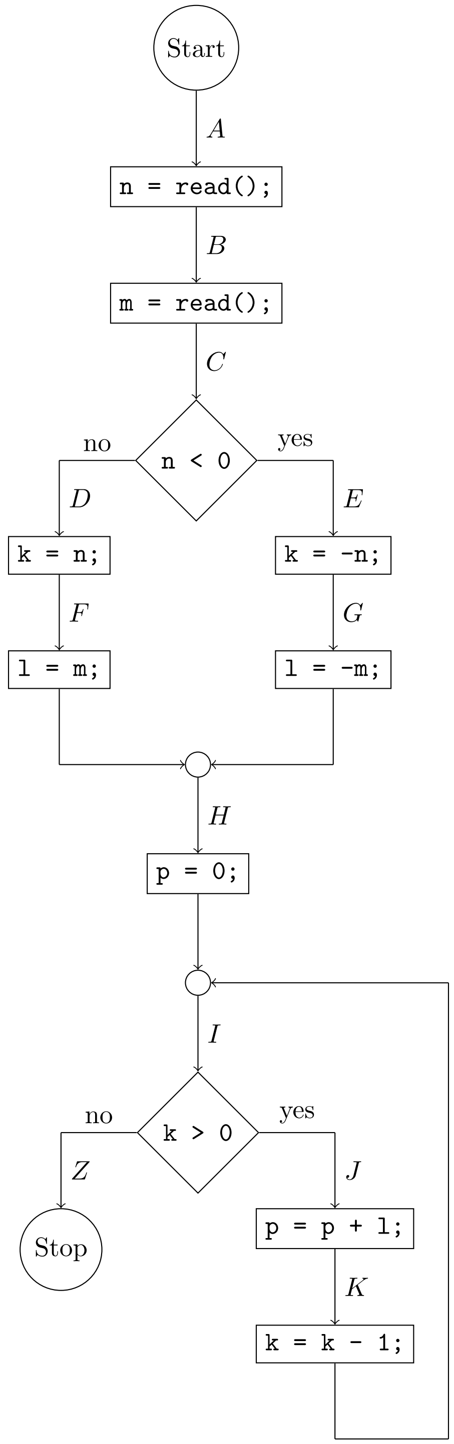
$$
\begin{aligned}
\mathsf{WP}[\![\texttt{write(b)}]\!](Z) &\equiv \mathsf{WP}[\![\texttt{write(b)}]\!](b \geq 0) \\
&\equiv b \geq 0 \equiv: A \\
\mathsf{WP}[\![\texttt{b = any(2*a, a-1, 5)}]\!](A) &\equiv \mathsf{WP}[\![\texttt{b = any(2*a, a-1, 5)}]\!](b \geq 0) \\
&\equiv 2a \geq 0 \wedge a - 1 \geq 0 \wedge 5 \geq 0 \\
&\equiv a > 0 \equiv: B \\
\mathsf{WP}[\![\texttt{b = 4}]\!](A) &\equiv \mathsf{WP}[\![\texttt{b = 4}]\!](b \geq 0) \\
&\equiv 4 \geq 0 \\
&\equiv \text{true} \equiv: C \\
\mathsf{WP}[\![\texttt{b = a}]\!](A) &\equiv \mathsf{WP}[\![\texttt{b = a}]\!](b \geq 0) \\
&\equiv a \geq 0 \equiv: D \\
\mathsf{WP}[\![\texttt{switch(a-2)}]\!](B, C, D) &\equiv \mathsf{WP}[\![\texttt{switch(a-2)}]\!](a > 0, \text{true}, a \geq 0) \\
&\equiv (a - 2 = -1 \implies a > 0) \wedge (a - 2 = -2 \implies \text{true}) \\
&\qquad \wedge (a - 2 \neq -1 \wedge a - 2 \neq -2 \implies a \geq 0) \\
&\equiv (a = 1 \implies a > 0) \wedge (a \neq 1 \wedge a \neq 0 \implies a \geq 0) \\
&\equiv a \neq 1 \wedge a \neq 0 \implies a \geq 0 \equiv: E \\
\mathsf{WP}[\![\texttt{a = pread()}]\!](E) &\equiv \mathsf{WP}[\![\texttt{a = pread()}]\!](a \neq 1 \wedge a \neq 0 \implies a \geq 0) \\
&\equiv \forall a \geq 0.(a \neq 1 \wedge a \neq 0 \implies a \geq 0) \\
&\equiv \text{true} \equiv: F
\end{aligned}
$$

2.7 Verifying a more complex program

In the following control flow graph, three assertions are given:

$$\begin{aligned} A &:\equiv \text{true} \\ I &:\equiv (n < 0 \implies (l = -m \land 0 \leq k \leq -n \land p = l \cdot (-n - k))) \\ &\quad \land (n \geq 0 \implies (l = m \land 0 \leq k \leq n \land p = l \cdot (n - k))) \\ Z &:\equiv p = n \cdot m \end{aligned}$$

Use the WP-operator to annotate all remaining program points with assertions. Then, prove local consistency.
Hint: When proving local consistency, there is not much to do. However, at two points there is actually something to show. Why is that?



Answer of exercise $\boxed{2.7}$

First, use the WP-operator to annotate all remaining assertions:

$$
\begin{aligned}
\mathsf{WP}[\![\texttt{k = k - 1;}]\!](I) &\equiv (n < 0 \implies (l = -m \land 0 \le k-1 \le -n \land p = l \cdot (-n-(k-1)))) \\
&\quad \land (n \ge 0 \implies (l = m \land 0 \le k-1 \le n \land p = l \cdot (n-(k-1)))) \\
&\equiv (n < 0 \implies (l = -m \land 1 \le k \le -(n-1) \land p = l \cdot (-n-k+1))) \\
&\quad \land (n \ge 0 \implies (l = m \land 1 \le k \le n+1 \land p = l \cdot (n-k+1)) \\
&\equiv: K \\
\mathsf{WP}[\![\texttt{p = p + l;}]\!](K) &\equiv (n < 0 \implies (l = -m \land 1 \le k \le -(n-1) \land p + l = l \cdot (-n-k+1))) \\
&\quad \land (n \ge 0 \implies (l = m \land 1 \le k \le n+1 \land p + l = l \cdot (n-k+1))) \\
&\equiv (n < 0 \implies (l = -m \land 1 \le k \le -(n-1) \land p = l \cdot (-n-k))) \\
&\quad \land (n \ge 0 \implies (l = m \land 1 \le k \le n+1 \land p = l \cdot (n-k))) \\
&\equiv: J \\
\mathsf{WP}[\![\texttt{p = 0;}]\!](I) &\equiv (n < 0 \implies (l = -m \land 0 \le k \le -n \land 0 = l \cdot (-n-k))) \\
&\quad \land (n \ge 0 \implies (l = m \land 0 \le k \le n \land 0 = l \cdot (n-k))) \\
&\equiv: H \\
\mathsf{WP}[\![\texttt{l = m;}]\!](H) &\equiv (n < 0 \implies (m = -m \land 0 \le k \le -n \land 0 = m \cdot (-n-k))) \\
&\quad \land (n \ge 0 \implies (m = m \land 0 \le k \le n \land 0 = m \cdot (n-k))) \\
&\equiv (n < 0 \implies (m = 0 \land 0 \le k \le -n)) \\
&\quad \land (n \ge 0 \implies (0 \le k \le n \land 0 = m \cdot (n-k))) \\
&\equiv: F \\
\mathsf{WP}[\![\texttt{k = n;}]\!](F) &\equiv (n < 0 \implies (m = 0 \land 0 \le n \le -n)) \\
&\quad \land (n \ge 0 \implies (0 \le n \le n \land 0 = m \cdot (n-n))) \\
&\equiv (n < 0 \implies (m = 0 \land n = 0)) \\
&\quad \land (n \ge 0 \implies n \ge 0) \\
&\equiv n \ge 0 \land n \ge 0 \\
&\equiv n \ge 0 \equiv: D \\
\mathsf{WP}[\![\texttt{l = -m;}]\!](H) &\equiv (n < 0 \implies (-m = -m \land 0 \le k \le -n \land 0 = -m \cdot (-n-k))) \\
&\quad \land (n \ge 0 \implies (-m = m \land 0 \le k \le n \land 0 = -m \cdot (n-k))) \\
&\equiv (n < 0 \implies (0 \le k \le -n \land 0 = -m \cdot (-n-k))) \\
&\quad \land (n \ge 0 \implies (m = 0 \land 0 \le k \le n)) \\
&\equiv: G \\
\mathsf{WP}[\![\texttt{k = -n;}]\!](G) &\equiv (n < 0 \implies (0 \le -n \le -n \land 0 = -m \cdot (-n+n))) \\
&\quad \land (n \ge 0 \implies (m = 0 \land 0 \le -n \le n)) \\
&\equiv (n < 0 \implies n \le 0) \\
&\quad \land (n \ge 0 \implies (m = 0 \land n = 0)) \\
&\equiv \text{true} \land n < 0 \\
&\equiv n < 0 \equiv: E \\
\mathsf{WP}[\![\texttt{n < 0}]\!](D, E) &\equiv (n \ge 0 \land n \ge 0) \lor (n < 0 \land n < 0) \\
&\equiv \text{true} \equiv: C \\
\mathsf{WP}[\![\texttt{m = read();}]\!](C) &\equiv \forall m.\, \text{true} \\
&\equiv \text{true}
\end{aligned}
$$

Now, all assertions are defined. All annotations that were derived using the WP-operator are automatically locally consistent. However, in this exercise, assertions A and I were *not* derived using the WP-operator. Hence, we still need to check

$$\mathsf{WP}[\![\texttt{n = read();}]\!](B) \impliedby A$$

as well as

$$\mathsf{WP}[\![\texttt{k > 0}]\!](Z, J) \impliedby I.$$

Let's do the calculations: the first statement is trivial, as

$$\mathsf{WP}[\![\texttt{n = read();}]\!](B) \equiv \forall n.\,\text{true} \equiv \text{true} \equiv A.$$

Now, consider the second statement. First, recall the law

$$(A \implies B) \wedge (\neg A \implies C) \equiv (A \wedge B) \vee (\neg A \wedge C).$$

We will use this to both rewrite I and J as

$$\begin{aligned} I \equiv\ & (n < 0 \wedge l = -m \wedge 0 \le k \le -n \wedge p = l \cdot (-n-k)) \\ & \vee (n \ge 0 \wedge l = m \wedge 0 \le k \le n \wedge p = l \cdot (n-k)) \\ J \equiv\ & (n < 0 \wedge l = -m \wedge 1 \le k \le -n+1 \wedge p = l \cdot (-n-k)) \\ & \vee (n \ge 0 \wedge l = m \wedge 1 \le k \le n+1 \wedge p = l \cdot (n-k)) \end{aligned}$$

Now, we can rewrite our final goal $\mathsf{WP}[\![\texttt{k > 0}]\!](Z, J) \impliedby I$ as

$$\begin{pmatrix} & k \le 0 & \wedge & p = nm \\ \vee & k > 0 & \wedge & J \end{pmatrix} \impliedby \begin{pmatrix} & k \le 0 & \wedge & I \\ \vee & k > 0 & \wedge & I \end{pmatrix}.$$

This can be proven by showing both $k \le 0 \wedge I \implies k \le 0 \wedge p = nm$ and $k > 0 \wedge I \implies k > 0 \wedge J$. In order to show this, however, it suffices to show

$$k \le 0 \wedge I \implies p = nm$$

and

$$k > 0 \wedge I \implies J.$$

So, first assume that $k \le 0 \wedge I$ holds. Then, since I implies $k \ge 0$, we have $k = 0$. This in turn implies (using I again) that $(n < 0 \wedge l = -m \wedge p = l \cdot (-n)) \vee (n \ge 0 \wedge l = m \wedge p = l \cdot n)$, which implies $(n < 0 \wedge p = (-m)(-n)) \vee (n \ge 0 \wedge p = nm)$, or equivalently, $p = nm$.
Now, consider the second equation and assume $k > 0 \wedge I$ holds. This implies $I \wedge k \ge 1$, which is equivalent to

$$\begin{aligned} & (n < 0 \wedge l = -m \wedge 1 \le k \le -n \wedge p = l \cdot (-n-k)) \\ & \vee (n \ge 0 \wedge l = m \wedge 1 \le k \le n \wedge p = l \cdot (n-k)) \end{aligned}$$

Since $-n < -n+1$ and $n < n+1$, this implies J and we are done.