Part I: Problem

Question 1.

1. The Galois connection is:

\[ \alpha(U(R)) \overset{\text{def}}{=} \{ (\rho, a(i)) \mid (\rho, a) \in R, i \in [0, \rho(N) - 1] \} \]

\[ \gamma(U(R^\sharp)) \overset{\text{def}}{=} \{ (\rho, a) \mid \forall i \in [0, \rho(N) - 1] : (\rho, a(i)) \in R^\sharp \} \]

This is indeed a Galois connection:

\[ \alpha(U(R)) \subseteq R^\sharp \iff \{ (\rho, a(i)) \mid (\rho, a) \in R, i \in [0, \rho(N) - 1] \} \subseteq R^\sharp \]

\[ \iff \forall (\rho, a) \in R : \forall i \in [0, \rho(N) - 1] : (\rho, a(i)) \in R^\sharp \]

\[ \iff (\rho, a) \in R \implies \forall i \in [0, \rho(N) - 1] : (\rho, a(i)) \in R^\sharp \]

\[ \iff R \subseteq \{ (\rho, a) \mid \forall i \in [0, \rho(N) - 1] : (\rho, a(i)) \in R^\sharp \} \]

\[ \iff R \subseteq \gamma(U(R^\sharp)) \]

Actually, we have a Galois embedding as \( \alpha(U) \) is onto: any \((\rho, x) \in E^U\) is the abstraction \( \alpha(U)\{ (\rho, a) \} \) of some \((\rho, a) \in E\) such that \( \forall i \in [0, \rho(N) - 1] : a(i) = x \).

2. Consider \( X \overset{\text{def}}{=} ([N \mapsto 2], [0 \mapsto 0, 1 \mapsto 1]) \in E \).

Then, \( \alpha(U \{ X \}) = \{ ([N \mapsto 2], 0), ([N \mapsto 2], 1) \} \).

Then, \( \gamma(U(\alpha(U \{ X \}))) = \{ ([N \mapsto 2], [0 \mapsto 0, 1 \mapsto 0]), ([N \mapsto 2], [0 \mapsto 0, 1 \mapsto 1]), ([N \mapsto 2], [0 \mapsto 1, 1 \mapsto 0]), ([N \mapsto 2], [0 \mapsto 1, 1 \mapsto 1]) \} \)

which is larger than \( \{ X \} \).

3. To find the exactness condition we compute:

\[ \gamma(U(\alpha(U(R)))) \]

\[ = \{ (\rho, a) \mid \forall i \in [0, \rho(N) - 1] : (\rho, a(i)) \in \alpha(U(R)) \} \]

\[ = \{ (\rho, a) \mid \forall i \in [0, \rho(N) - 1] : \exists (\rho, a') \in R : \exists j \in [0, \rho(N) - 1] : a(i) = a'(j) \} \]

\( \gamma(U(\alpha(U(R)))) = R \) if and only if \( \gamma(U(\alpha(U(R)))) \subseteq R \), i.e.:
\[ \gamma^U(\alpha^U(R)) \subseteq R \]

\[ \iff \{ (\rho, a) \mid \forall i \in [0, \rho(N) - 1] : \exists (\rho, a') \in R : \exists j \in [0, \rho(N) - 1] : a(i) = a'(j) \} \subseteq R \]

\[ \iff \forall i \in [0, \rho(N) - 1] : \exists j \in [0, \rho(N) - 1] : \exists (\rho, a') \in R : a(i) = a'(j) \implies (\rho, a) \in R \]

\[ \iff (\rho, a') \in R \implies \forall i, j \in [0, \rho(N) - 1] : \exists (\rho, a) \in R : a(i) = a'(j) \]

i.e., whenever the array \( A \) contains some value \( v \) at some index \( i \) while the scalar variables are defined by \( \rho \), then \( v \) can also appear at any other index \( j \) of \( A \) in another environment sharing the same \( \rho \). The abstraction is uniform: it collects the set of possible array element values for each \( \rho \) but does not distinguish between elements at different positions.

**Question 2.**

1. We derive abstract versions \( F^\sharp \) of each operator \( F \) using the Galois connection: \( F^\sharp \overset{\text{def}}{=} \alpha^U \circ F \circ \gamma^U \), so that our abstract operators are optimal by construction.

* Abstract assignment into a scalar:

\[
\left\{ (\rho[V \mapsto x], a) \mid (\rho, a) \in \gamma^U(R^\sharp) \right\} = \alpha^U(\{ (\rho[V \mapsto x], a) \mid (\rho, a) \in \gamma^U(R^\sharp) \})
\]

\[
= \alpha^U(\{ (\rho[V \mapsto x], a) \mid (\rho, x) \in \gamma^U(R^\sharp), \rho \cap [0, \rho(N) - 1] \neq \emptyset \})
\]

\[
= \{ (\rho[V \mapsto x], y) \mid (\rho, x), (\rho, y) \in R^\sharp, \rho \cap [0, \rho(N) - 1] \neq \emptyset \}
\]

The expression \( e \) is only evaluated to ensure that there is no out-of-bound access. We note that this abstraction is actually exact as:

\[
\{ (\rho[V \mapsto x], a) \mid (\rho, x) \in R^\sharp, (\rho, a) \in \gamma^U(R^\sharp) \}
\]

can be exactly represented in \( \mathcal{P}(\mathcal{E}^U) \) as it satisfies the formula from question 1.3.

* Abstract assignment into the array:

\[
\left\{ (\rho[A \leftarrow e], a) \mid (\rho, a) \in \gamma^U(R^\sharp) \right\} = \alpha^U(\{ (\rho[A \leftarrow e], a) \mid (\rho, a) \in \gamma^U(R^\sharp) \})
\]

\[
= \alpha^U(\{ (\rho, a[i \mapsto v]) \mid (\rho, a) \in \gamma^U(R^\sharp), i \in \mathcal{E}[e] \rho \cap [0, \rho(N) - 1], v \in \mathcal{E}[e'] \rho \})
\]

\[
= \{ (\rho, x), (\rho, v) \mid (\rho, x) \in R^\sharp, \mathcal{E}[e] \rho \cap [0, \rho(N) - 1] \neq \emptyset, v \in \mathcal{E}[e'] \rho \}
\]

As before, the expression \( e \) is only evaluated to ensure that there is no out-of-bound access. To prove that the operator is not exact, consider \( R^\sharp \overset{\text{def}}{=} \{ ([N \mapsto 2], 0) \} \), which represents:

\[
R = \gamma^U(R^\sharp) = \{ ([N \mapsto 2], [0 \mapsto 0, 1 \mapsto 0]) \}
\]

In the concrete, we have:

\[
C[A[0] \leftarrow 1] R = \{ ([N \mapsto 2], [0 \mapsto 1, 1 \mapsto 0]) \}
\]

However, this set cannot be exactly represented in the abstract; we get instead:

\[
C^U[A[0] \leftarrow 1] R^\sharp = \{ ([N \mapsto 2], 0), ([N \mapsto 2], 1) \}
\]

whose concretization is much larger (see question 1.2). Hence, the operator is not exact.
2. Abstract join: we have \( \cup^U = \cup \) as:

\[
R^\sharp \cup^U S^\sharp \\
\overset{\text{def}}{=} \alpha^U(\gamma^U(R^\sharp) \cup \gamma^U(S^\sharp)) \\
= \{ (\rho, a(i)) \mid (\rho, a) \in \gamma^U(R^\sharp) \cup \gamma^U(S^\sharp), i \in [0, \rho(N) - 1] \} \\
= \{ (\rho, a(i)) \mid (\rho, a) \in \gamma^U(R^\sharp), i \in [0, \rho(N) - 1] \} \cup \\
\{ (\rho, a(i)) \mid (\rho, a) \in \gamma^U(S^\sharp), i \in [0, \rho(N) - 1] \} \\
= \alpha^U(\gamma^U(R^\sharp)) \cup \alpha^U(\gamma^U(S^\sharp)) \\
= R^\sharp \cup S^\sharp
\]

The last line comes from the Galois embedding property: \( \alpha^U \circ \gamma^U = \text{id} \).

To show that \( \cup^U \) is not exact, consider \( R^\sharp \overset{\text{def}}{=} \{ ([N \mapsto 2], 0) \} \) and \( S^\sharp \overset{\text{def}}{=} \{ ([N \mapsto 2], 1) \} \).

Then:

\[
\gamma^U(R^\sharp) \cup \gamma^U(S^\sharp) = \{ ([N \mapsto 2], [0 \mapsto 0, 1 \mapsto 0]), ([N \mapsto 2], [0 \mapsto 1, 1 \mapsto 1]) \}
\]

But:

\[
\gamma^U(R^\sharp \cup^U S^\sharp) = \gamma^U(\{ ([N \mapsto 2], 0), ([N \mapsto 2], 1) \}) \\
= \{ ([N \mapsto 2], [0 \mapsto 0, 1 \mapsto 0]), ([N \mapsto 2], [0 \mapsto 0, 1 \mapsto 1]), 
\quad ([N \mapsto 2], [0 \mapsto 1, 1 \mapsto 0]), ([N \mapsto 2], [0 \mapsto 1, 1 \mapsto 1]) \}
\]

as in question 1.2.

Question 3.

1. At the end of the program, the array is completely initialized to 1; hence:

\[
\mathcal{F} \overset{\text{def}}{=} \{ ([I \mapsto n, N \mapsto n], a) \mid n \geq 2, \forall i \in [0, n - 1] : a(i) = 1 \}
\]

2. We have: \( \mathcal{I}^U \overset{\text{def}}{=} \alpha^U(\mathcal{I}) = \{ ([I \mapsto n, N \mapsto n], 0) \mid n \geq 2 \} \)

and \( \mathcal{F}^U \overset{\text{def}}{=} \alpha^U(\mathcal{F}) = \{ ([I \mapsto n, N \mapsto n], 1) \mid n \geq 2 \} \).

Note that \( \gamma^U(\mathcal{I}^U) = \mathcal{I} \) and \( \gamma^U(\mathcal{F}^U) = \mathcal{F} \), so that the abstractions of \( \mathcal{I} \) and \( \mathcal{F} \) are indeed exact in \( \mathcal{P}(\mathcal{E}^U) \).

3. The concrete loop invariant is: \( \{ ([I \mapsto i, N \mapsto n], a) \mid n \geq 2, i \in [0, n], \forall k \in [0, i - 1] : a(k) = 1, \forall k \in [i, n - 1] : a(k) = 0 \} \), which cannot be represented exactly in \( \mathcal{P}(\mathcal{E}^U) \).

The best over-approximation of the invariant in \( \mathcal{P}(\mathcal{E}^U) \) is \( \{ ([I \mapsto i, N \mapsto n], v) \mid n \geq 2, i \in [0, n], v \in \{0, 1 \} \} \).

The computation \( \mathcal{C}^U [P_1] \mathcal{I}^U \) would then give \( \{ ([I \mapsto n, N \mapsto n], v) \mid n \geq 2, v \in \{0, 1 \} \} \), which is coarser than \( \mathcal{F}^U \) and cannot prove that \( A \) is indeed initialized to 1.

Question 4.

* We use the interval Galois connection between \( \mathcal{P}(\mathcal{E}^U) \) and \( \mathcal{D}^I \):
These operators are sound by construction, but they are not optimal. Indeed, in both cases, the abstract assignment into the array:

$$\alpha^I(R) \overset{\text{def}}{=} \begin{cases} \lambda V \in \mathcal{V}.[\min \{ \rho(V) \mid (\rho, a) \in R \}, \max \{ \rho(V) \mid (\rho, a) \in R \}] & \text{if } R \neq \emptyset \\ \lambda A. [\min \{ a \mid (\rho, a) \in R \}, \max \{ a \mid (\rho, a) \in R \}] & \text{if } R = \emptyset \end{cases}$$

Abstract assignment into a scalar:

$$\gamma^I(R^d) \overset{\text{def}}{=} \begin{cases} \{ (\rho, a) \mid \forall V \in \mathcal{V} : \rho \in R^d(V), a \in R^d(a) \} & \text{if } R^d \neq \bot, \emptyset \text{ otherwise} \end{cases}$$

* Abstract assignment into a scalar:

$$C^{U,I}[V \leftarrow A[e]] R^d \overset{\text{def}}{=} \alpha^I(C^{U}[V \leftarrow A[e]] \gamma^I(R^d))$$

$$= \alpha^I(\{ (\rho[V \mapsto x], y) \mid (\rho, x), (\rho, y) \in \gamma^I(R^d), E[e] \rho \cap [0, \rho(N) - 1] \neq \emptyset \})$$

$$= \begin{cases} R^d[V \mapsto R^d(A)] & \text{if } (E[e] \gamma^I(R^d)) \cap [0, \max R^d(N) - 1] \neq \emptyset \\ \bot & \text{otherwise} \end{cases}$$

Abstract assignment into the array:

$$C^{U,I}[A[e] \leftarrow e'] R^d \overset{\text{def}}{=} \alpha^I(C^{U,I}[A[e] \leftarrow e'] \gamma^I(R^d))$$

$$= \alpha^I(\{ (\rho, x), (\rho, v) \mid (\rho, x) \in \gamma^I(R^d), E[e] \rho \cap [0, \rho(N) - 1] \neq \emptyset \})$$

$$= \begin{cases} R^d \cup^I R^d[A \mapsto \alpha^I(E[e'] \gamma(R^d))] & \text{if } (E[e] \gamma^I(R^d)) \cap [0, \max R^d(N) - 1] \neq \emptyset \\ \bot & \text{otherwise} \end{cases}$$

$$\subseteq \begin{cases} R^d \cup^I R^d[A \mapsto E[e'] R^d] & \text{if } (E[e] R^d) \cap [0, \max R^d(N) - 1] \neq \emptyset \\ \bot & \text{otherwise} \end{cases}$$

These operators are sound by construction, but they are not optimal. Indeed, in both cases, the last inclusion is not an equality because $E[I[e]]$ is, in general, not an optimal abstraction of $E[e]$ (e.g., $E[I[V - V]]$).

**Question 5.**

* $C[P_2] I = \{ (I \mapsto n, N \mapsto n), a \} \mid n \geq 2, \forall i \in [0, n - 1], a(i) = i + 1 \}$

* The best abstraction of $I$ in $P(U)$ is $I^U \overset{\text{def}}{=} \{ (I \mapsto 0, N \mapsto n), 0 \} \mid n \geq 2 \}$.

The computation is similar to that of question 3.3. We find, as loop invariant:

$$\{ (I \mapsto i, N \mapsto n), a \} \mid n \geq 2, i \in [0, n], a \in [0, i] \}$$

so that, at the end of the loop, we have:

$$C[P_2] I^U = \{ (I \mapsto n, N \mapsto n), a \} \mid n \geq 2, a \in [0, n] \}$$

Note that this abstraction not only forgets the relationship between the array index $i$ and the array contents $a(i) = i + 1$ at the index, but it also forgets that the array values are strictly positive ($0$ is allowed in the abstraction).

* The best abstraction of $I$ in $D$ is $I^{'} \overset{\text{def}}{=} \{ I \mapsto [0, 0], N \mapsto [2, +\infty], A \mapsto [0, 0] \}$.

Then, the interval loop invariant is:
\[ I \mapsto [0, +\infty], N \mapsto [2, +\infty], A \mapsto [0, +\infty] \]

and so:

\[ C^U[I \mid P_2] = [I \mapsto [2, +\infty], N \mapsto [2, +\infty], A \mapsto [0, +\infty]] \]

Compared to the uniform abstraction, the interval abstraction further loses the relationship between \( I, N, \) and \( A \). In particular, we cannot prove that the array contents is bounded by the array size: \( A \leq N \). Naturally, similarly to the uniform abstraction, the interval abstraction cannot prove that the array is initialized to strictly positive values: \( A \geq 1 \).

**Question 6.**

1. We have:

\[
C^U[\text{expand } A \mapsto B; V \leftarrow B; \text{remove } B] R
= C^U[\text{remove } B] (C^U[V \leftarrow B] (C^U[\text{expand } A \mapsto B] R))
= \{ \rho \mid \exists v \in Z : \rho \oplus [B \mapsto v] \in C^U[V \leftarrow B] (C^U[\text{expand } A \mapsto B] R) \}
= \{ \rho \mid \exists v \in Z : \exists \rho' \in C^U[\text{expand } A \mapsto B] R : \rho' \oplus [B \mapsto v] = \rho'[V \mapsto \rho'(B)] \}
= \{ \rho \mid \exists v, v' \in Z : \exists \rho'' \in R : \rho''[A \mapsto v] \in R
\quad \rho' = \rho'' \oplus [B \mapsto v'], \rho \oplus [B \mapsto v] = \rho'[V \mapsto \rho'(B)] \}
= \{ \rho \mid \exists v, v' \in Z : \exists \rho'' \in R : \rho''[A \mapsto v] \in R, \rho \oplus [B \mapsto v] = (\rho'' \oplus [B \mapsto v])[V \mapsto v'] \}
= \{ \rho \mid \exists v' \in Z : \exists \rho'' \in R : \rho''[A \mapsto v'] \in R, \rho = \rho''[V \mapsto v'] \}
= \{ \rho''[V \mapsto v'] \mid \rho'' \in R, \rho''[A \mapsto v'] \in R \}
\]

(the first four equalities are obtained by expanding the definition of the sequence, then \text{remove } B,
then \( V \leftarrow B \), then \text{expand } A \mapsto B; \) we then eliminate \( \rho' \), then \( v \), and finally \( \rho \).

On the other hand, when using \( E^U \simeq (\forall \cup \{A\}) \rightarrow Z \), the formula found in question 2 can be rewritten as:

\[ C^U[V \leftarrow A[e]] R = \{ \rho[V \mapsto v] \mid \rho \in R, \rho[A \mapsto v] \in R, E[\rho \cap [0, N - 1] \neq \emptyset] \} \]

2. The two formulas only differ for environments that necessarily cause an out-of-bound access
\((E[e] \rho \cap [0, N - 1] = \emptyset)\), in which case the first formula is less precise (returning some environments instead of \( \emptyset \)), so, it is sound but not optimal. When there is no out-of-bound access, the formulas are equal, i.e., the approximation is exact.

3. \* Let \( X \overset{\text{def}}{=} \{ \{A \mapsto a\} \mid a \in \{0, 1\} \} \).

Then, \( C^U[\text{add } B; B \leftarrow A] = \{ \{A \mapsto a, B \mapsto a\} \mid a \in \{0, 1\} \} \), which satisfies \( A = B \).

However, \( C^U[\text{expand } A \mapsto B] = \{ \{A \mapsto a, B \mapsto b\} \mid a, b \in \{0, 1\} \} \), which contains some environments that do not satisfy \( A = B \).

\* Let \( X \overset{\text{def}}{=} \{ \{V \mapsto 0, A \mapsto a\} \mid a \in \{0, 1\} \} \).

Then, \( C^U[V \leftarrow A[e]] = \{ \{V \mapsto v, A \mapsto a\} \mid a, v \in \{0, 1\} \} \).

However, \( C^U[V \leftarrow A] = \{ \{V \mapsto a, A \mapsto a\} \mid a \in \{0, 1\} \} \). This implies \( V = A \) after the assignment, i.e., in each environment, all the array elements are equal to the value of \( V \), which is obviously wrong.
Question 7.

We abstract $C^U[A[e \leftarrow e']$ as $C^U[\text{add } B; B \leftarrow e'; \text{fold } A \leftarrow B]$. This is sound as:

$$C^U[\text{add } B; B \leftarrow e'; \text{fold } A \leftarrow B] \cap$$

$$= C^U[\text{fold } A \leftarrow B] (C^U[ B \leftarrow e'] (C^U[\text{add } B] R))$$

$$= C^U[\text{fold } A \leftarrow B] (C^U[ B \leftarrow e'] \{ \rho \oplus [B \mapsto v] \mid \rho \in R, v \in Z \})$$

$$= C^U[\text{fold } A \leftarrow B] \{ \rho \oplus [B \mapsto v] \mid \rho \in R, v \in E[e'] \rho \}$$

$$= \{ \rho' \mid \exists v' \in Z : \exists \rho \in R : \exists v \in E[e'] \rho : \rho' \oplus [B \mapsto v'] = \rho \oplus [B \mapsto v] \} \cup$$

$$\{ \rho'[A \mapsto v'] \mid \exists \rho \in R : \exists v \in E[e'] \rho : \rho' \oplus [B \mapsto v'] = \rho \oplus [B \mapsto v] \}$$

$$= \{ \rho \mid \rho \in R, E[e'] \rho \neq \emptyset \} \cup \{ \rho[A \mapsto v] \mid \rho \in R, v \in E[e'] \rho \}$$

$$= \{ \rho, \rho[A \mapsto v] \mid \rho \in R, v \in E[e'] \rho \}$$

(as before the first lines are obtained by expanding the definition of the sequence, of \text{add } B, of B \leftarrow e', \text{ and then } \text{fold } A \leftarrow B).$

On the other hand, when using $E^U \simeq ((\forall \cup \{A\}) \rightarrow Z)$, the formula found in question 2 can be rewritten as:

$$C^U[A[e \leftarrow e'] R = \{ \rho, \rho[A \mapsto v] \mid \rho \in R, E[e] \rho \cap [0, N - 1] \neq \emptyset, v \in E[e'] \rho \}.$$

As before, the formulas are equivalent for environments that do not have an out-of-bound array access. Otherwise, $C^U[\text{add } B; B \leftarrow e'; \text{fold } A \leftarrow B]$ is less precise, and so, sound but not optimal.

Question 8.

We consider that each polyhedron $\gamma^P(C)$ is represented by a set of affine constraints $C$. We denote by $C^P[.]$ the regular polyhedra operators seen in the course.

* We set:

$$C^U,P[\text{add } W] C \overset{\text{def}}{=} C$$

i.e., we do not change the constraint presentation. That way, there is no constraint on $W$, which models the fact that $W$ can have an arbitrary value, independent from the values of other variables. The operator is exact.

* We set:

$$C^U,P[\text{remove } W] C \overset{\text{def}}{=} C^P[W \leftarrow [-\infty, +\infty]] C$$

i.e., use the “forget” (or “project”) operation. The resulting constraint system is guaranteed to not feature the variable $W$. On rationals, the operation is exact. Indeed, we have:

$$\gamma^P(C^P[W \leftarrow [-\infty, +\infty]] C) = \{ \rho[W \mapsto w] \mid \rho \in \gamma^P(P), w \in \mathbb{Q} \}$$

$$= \{ \rho \mid \exists w \in \mathbb{Q} : \rho[W \mapsto w] \in \gamma^P(P) \}.$$

However, on integers, as is the case here, the operation is not exact, as the projection of an integer polyhedron may not be an integer polyhedron. Consider for instance $C^U[\text{remove } Y] \gamma^P(\{X = 2Y\}) = \{ [X \mapsto x] \mid x \in 2\mathbb{Z} \}$, which is not a polyhedron, although $\gamma^P(\{X = 2Y\})$ is.

* We set

$$C^U,P[\text{expand } V \mapsto W] C \overset{\text{def}}{=} C \cup \{ c[W/V] \mid c \in C \}$$
i.e., to the constraint set $C$ we add a copy of each constraint $c \in C$ where the variable $W$ has been substituted for the variable $V$.

We now prove that this operation is exact on polyhedra. Recall that $C^U[\mathbf{expand} V \mapsto W] \overset{\text{def}}{=} \{ \rho \oplus \{ W \mapsto v \} \mid \rho \in R, \rho[V \mapsto v] \in R \}$. Given a map $\rho$ containing variable $V$ but not $W$, then $\rho' \overset{\text{def}}{=} \rho \oplus \{ W \mapsto v \}$ satisfies $C \cup \{ c[W/V] \mid c \in C \}$ if and only if $\rho'$ satisfies $C$ and $\rho'$ satisfies $\{ c[W/V] \mid c \in C \}$. As $W$ does not occur in $C$, $\rho'$ satisfies $C$ if and only if $\rho$ does. As $V$ does not occur in $\{ c[W/V] \mid c \in C \}$, $\rho'$ satisfies $\{ c[W/V] \mid c \in C \}$ if and only if $\rho \oplus \{ W \mapsto v \} \oplus V$ does (where $\oplus V$ indicates that we remove a variable from a map). By renaming back $W$ into $V$ in both $\rho \oplus \{ W \mapsto v \} \oplus V$ and $\{ c[W/V] \mid c \in C \}$, this is equivalent to having $\rho[V \mapsto v]$ satisfying $C$. To sum up, we have that $\rho \oplus \{ W \mapsto v \}$ satisfies $C \cup \{ c[W/V] \mid c \in C \}$ if and only if both $\rho$ and $\rho[V \mapsto v]$ satisfy $C$, which concludes the proof.

Consider the polyhedron $\gamma^P(C)$ defined by the constraint set $C \overset{\text{def}}{=} \{ V \in [0, 1], W \in [10, 11] \}$. Then, in the concrete, $C^U[\mathbf{fold} V \leftarrow W] \gamma^P(C) = \{ [V \mapsto v \mid v \in [0, 1] \cup [10, 11]] \}$, which is not convex. Hence, there cannot exist an exact abstraction of $\mathbf{fold} V \leftarrow W$ in the polyhedra domain. We propose the following abstraction:

$$C^{U,P}[\mathbf{fold} V \leftarrow W] C \overset{\text{def}}{=} C^{U,P}[\mathbf{remove} W] (C \cup \mathbf{P} \mathbf{C}[V \leftarrow W] C)$$

i.e., we join the polyhedron with a copy where $V$ is assigned to $W$, and then forget $W$. We justify the soundness as follows, using the soundness of the abstract $C^{U,P}[\mathbf{remove} W]$, of $\cup \mathbf{P}$, and $C^{U}[V \leftarrow W]$, as well as the complete $\cup$–morphism property of $C^{U}[\mathbf{remove} W]$ and $\cup$:

$$\gamma^P(C^{U,P}[\mathbf{remove} W] (C \cup \mathbf{P} \mathbf{C}[V \leftarrow W] C)) \supseteq C^{U}[\mathbf{remove} W] (\gamma^P(\mathbf{C}[V \leftarrow W] C))$$

$$= \{ \rho \mid \exists v \in Z : \rho \oplus \{ W \mapsto v \} \in \gamma^P(C) \} \cup \{ \rho \mid \exists v \in Z : \rho \oplus \{ W \mapsto v \} \in C[\mathbf{V} \leftarrow W] \gamma^P(C) \}$$

$$= \{ \rho \mid \exists v \in Z : \rho \oplus \{ W \mapsto v \} \in \gamma^P(C) \} \cup \{ \rho[V \mapsto v] \mid \rho \oplus \{ W \mapsto v \} \in \gamma^P(C) \}$$

$$= C^{U}[\mathbf{fold} V \leftarrow W] \gamma^P(C)$$

We start by giving the polyhedra semantics of $P_2$:

- The initial state is abstracted exactly in the polyhedra domain using the constraint set $I^P = \{ N \geq 2, I = 0, A = 0 \}$.

- In the first loop iteration, after an application of $A[I] \leftarrow I + 1$, we get $A^P = \{ N \geq 2, I = 0, 0 \leq A \leq 1 \}$ and, after incrementing $I$, we get $Y^P = \{ N \geq 2, I = 1, 0 \leq A \leq 1 \}$.

The control flow join at the loop head gives $I^P = I^P \cup P Y^P = \{ N \geq 2, 0 \leq I \leq 1, 0 \leq A \leq I \}$. Note that the join creates a relation $A \leq I$ between the array contents and $I$, stating that all array elements are smaller than $I$.

- After performing a second loop iteration from $I^P$, we get similarly $I^P = \{ N \geq 2, 0 \leq I \leq 2, 0 \leq A \leq I \}$.

Note that the assignment $A[I] \leftarrow I + 1$ maintains the relation between $A$ and $I$. 
We apply a widening and get $W^P_2 = I^P_1 \lor I^P_2 = \{ N \geq 2, 0 \leq I, 0 \leq A \leq I \}$.

To get the invariant when the program ends, we apply the exit loop condition.

An extra iteration shows that $W^P_2$ is stable; however, it is not very precise on the upper bound of $I$ due to the widening.

To recover some precision, we apply (as in the course) one iteration without widening. We get $W^P_3 = \{ N \geq 2, 0 \leq I \leq N, 0 \leq A \leq I \}$, i.e., we recover the relation between $I$ and $N$.

We are able to prove that all array elements are smaller than the array size $N$. However, we cannot prove that the array elements are greater than 1, nor that $\forall i : A[i] = i + 1$ (although both properties are true, and the first property is expressible in the polyhedra domain with the uniform abstraction).

The semantics of $P_1$ could be computed the same way. It is actually simpler as there is not relationship between $A$ and $I$. We thus find: $\{ N \geq 2, I = N, 0 \leq A \leq N \}$. Note that, apart from the relation $I = N$, the result is exactly the same with the polyhedra domain as with the interval domain. In particular, we cannot infer that $A = 1$, i.e., that the array is fully initialized to 1 when the program stops.

**Question 10.**

1. Neither the assignment $I \leftarrow I + 1$ nor the test $I < N$? updates the array, and so, they are identical to a semantics in $E^one$ where $L$ and $H$ have no special meaning. We set:

$$C^one[I \leftarrow I + 1] R = \{ \rho[I \mapsto \rho(I) + 1] \mid \rho \in R \}$$

$$C^one[I < N?] R = \{ \rho \in R \mid \rho(I) < \rho(N) \} .$$

These are obviously sound and exact abstractions.

The assignment $A[I] \leftarrow 1$ is more interesting as it is able to manipulate the predicate $one(L, H)$. More precisely, it can extend the range on which $one(L, H)$ holds whenever $I$ is adjacent to the range $[L, H]$ (i.e., $I = L - 1$ or $I = H + 1$). Formally:

$$C^one[A[I] \leftarrow 1] R \overset{\text{def}}{=} \{ f(\rho) \mid \rho \in R \}$$

where

$$f(\rho) \overset{\text{def}}{=} \begin{cases} 
\rho[H \mapsto \rho(H)] & \text{if } \rho(I) = \rho(H) + 1 \\
\rho[L \mapsto \rho(I)] & \text{else if } \rho(I) = \rho(L) - 1 \\
\rho & \text{otherwise .}
\end{cases}$$

The operator is obviously sound. However, it is not exact. Consider for instance $X^2 \overset{\text{def}}{=} \{ [N \mapsto 10, L \mapsto 0, H \mapsto 1] \}$ representing the set of 10-element arrays whose two first elements are ones. Then, $C[A[5] \leftarrow 1] \gamma(X^2)$ is the set of environments $Y \overset{\text{def}}{=} \{ ([N \mapsto 10], a) \mid a(0) = a(1) = a(5) = 1 \}$, which cannot be exactly represented in $P(E^one)$. We have instead: $C^one[A[5] \leftarrow 1] X^2 = X^2$.

This example also shows that there are no best abstraction (i.e., no $\alpha^one$) in $P(E^one)$: $Y$ can be over-approximated by both $X^2 = \{ [N \mapsto 10, L \mapsto 0, H \mapsto 1] \}$ and by $\{ [N \mapsto 10, L \mapsto 5, H \mapsto 5] \}$, neither of which is a better abstraction.

We now prove that the regular set union $\cup^one \overset{\text{def}}{=} \cup$ on $P(E^one)$ is a sound and exact abstraction of $\cup$ on $P(E)$:
Question 11.

1. When computing the semantics of \( P \), in the previous question, we have expressed

\[
\gamma_{\text{one}}(R \cup S) = \{ (\rho, a) \mid \rho \oplus [L \mapsto l, H \mapsto h] \in R \cup S, \forall i \in [l, h] : a(i) = 1 \}
\]

\[
= \{ (\rho, a) \mid \rho \oplus [L \mapsto l, H \mapsto h] \in R, \forall i \in [l, h] : a(i) = 1 \} \cup
\]

\[
\{ (\rho, a) \mid \rho \oplus [L \mapsto l, H \mapsto h] \in S, \forall i \in [l, h] : a(i) = 1 \}
\]

= \( \gamma_{\text{one}}(R) \cup \gamma_{\text{one}}(S) \)

In fact, we can check that \( \gamma_{\text{one}} \) is a \( \cup \)-morphism: \( \gamma_{\text{one}}(X) = \bigcup \{ \gamma_{\text{one}}(\{x\}) \mid x \in X \} \).

2. When computing the semantics of \( P_1 \) in \( \mathcal{P}(E^{\text{one}}) \), every application of \( C_{\text{one}}^1[I \leftarrow 1] \) triggers the first case of \( f \), i.e., \( H \) is incremented. At the beginning of the \( k \)-th iteration of the loop, we get the following set of abstract environments:

\[
\{ [N \mapsto n, I \mapsto i, L \mapsto 0, H \mapsto i - 1] \mid n \geq 2, i \leq \min(n, k) \}
\]

whose join over \( k \) gives the loop invariant:

\[
\{ [N \mapsto n, I \mapsto i, L \mapsto 0, H \mapsto i - 1] \mid n \geq 2, i \leq n \}.
\]

Hence, when the program stops, we have the property:

\[
\{ [N \mapsto n, I \mapsto n, L \mapsto 0, H \mapsto n - 1] \mid n \geq 2 \}
\]

which proves that the array is completely initialized to 1.

Question 11.

1. * In the previous question, we have expressed \( C_{\text{one}}^1[I \leftarrow I + 1] \), \( C_{\text{one}}^1[I < N?] \) and \( \cup_{\text{one}} \) using regular scalar concrete semantic operators over \( \mathbb{V} \cup \{L, H\} \), where \( L \) and \( H \) have no special meaning. By replacing the concrete scalar semantics with a polyhedral scalar semantics, we simply get:

\[
C_{\text{one}}^P[I \leftarrow I + 1] \quad \text{def} \quad C^P[I \leftarrow I + 1]
\]

\[
C_{\text{one}}^P[I < N?] \quad \text{def} \quad C^P[I < N?]
\]

\[
\cup_{\text{one}} \quad \text{def} \quad \cup^P
\]

* To abstract \( A[I] \leftarrow 1 \), we separate three possible cases, depending on the relative value of \( L \), \( I \), and \( H \). The predicate can be extended by increasing the upper bound \( H \) (when \( I = H + 1 \)), or extended by decreasing the lower bound \( L \) (when \( I = L - 1 \)), or left unchanged (when \( I \) is neither \( H + 1 \) nor \( L - 1 \)). Formally, this can be abstracted using regular polyhedra assignments, tests, and joins:

\[
C_{\text{one}}^P[A[I] \leftarrow 1] R \quad \text{def} \quad C^P[H \leftarrow H + 1](C^P[I = H + 1?] R) \cup^P
\]

\[
C^P[L \leftarrow L - 1](C^P[I = L - 1?] R) \cup^P
\]

\[
C^P[I \neq H + ?](C^P[I \neq L - ?] R)
\]

The soundness is a consequence of the soundness of each regular polyhedra operator we use. Note that equality and disequality tests can be decomposed into pairs of inequalities, for instance:

\[
C^P[I \neq H + 1] R \quad \text{def} \quad C^P[I > H + 1?] R \cup C^P[I < H + 1?] R.
\]
2. We note that all the abstract elements computed by the predicate semantic \( C^{one}[\cdot] \) for \( P_1 \) are actually exactly expressible in the polyhedra domain.

* We start with the constraint set \( I^P = \{ N \geq 2, I = 0, L = 0, H = -1 \} \) abstracting \( I \).
* The first application of \( A[I] \leftarrow 1 \) gives: \( C^{P,\text{one}}[A[I] \leftarrow 1] I^P = C^P[H \leftarrow H + 1] I^P = \{ N \geq 2, I = 0, L = 0, H = 0 \} \).
* After incrementing \( I \), we get \( \{ N \geq 2, I = 1, L = 0, H = 0 \} \).
* The join with \( I^P \) at the loop head gives: \( I^P = \{ N \geq 2, 0 \leq I \leq 1, L = 0, H = I - 1 \} \). Note that we discover the important relation \( H = I - 1 \).
* After a second iteration, we get: \( I^P = \{ N \geq 2, 0 \leq I \leq 2, L = 0, H = I - 1 \} \).
* The polyhedral widening gives: \( I^P \lor I^P = \{ N \geq 2, 0 \leq I \leq 2, L = 0, H = I - 1 \} \).
* A decreasing iteration recovers the constraint \( I \leq N \).

The polyhedral invariant is thus: \( \{ N \geq 2, 0 \leq I \leq N, L = 0, H = I - 1 \} \). The predicate \( \text{one}(0, N - 1) \) holds, which expresses the fact that \( A \) is completely filled with ones.

**Question 12.**

To analyze precisely \( P_3 \), it is necessary to express exactly the loop invariant, i.e., the fact that \( V \) is the maximum of a slice of the array (but not necessarily of the whole array). Naturally, we use a predicate \( V = \max A(L, H) \) that denotes that the value of \( V \) is the maximum of \( A \) between indices \( L \) and \( H \). However, this is not sufficient: we also need to keep the relationship between \( X \), \( V \), and \( A[I] \) in order to abstract precisely the assignment \( X \leftarrow A[I] \) and the test \( X > V? \).

There are several solutions to this problem. Here, we will use a simple solution, which consists in adding a new synthetic variable \( A_I \) that represents the array element at index \( I \) in the current environment. Hence, we set:

\[
E^{max} \overset{\text{def}}{=} (\forall \{ A, H, A_I \}) \rightarrow \mathbb{Z}
\]

with concretization \( \gamma^{max}: \mathcal{P}(E^{max}) \rightarrow \mathcal{P}(E) \) defined as:

\[
\gamma^{max}(R^E) \overset{\text{def}}{=} \{ (\rho, a) \mid \exists ! l, h, x \in \mathbb{Z} : \rho \oplus [L \mapsto l, H \mapsto h, A_I \mapsto x] \in R^E, \\
\rho(I) \in [0, \rho(N) - 1] \implies x = a(\rho(I)), \\
l \leq h \implies \rho(V) = \max \{ a(i) \mid i \in [l, h] \} \}
\]

Note that, when \( L > H \), we do not impose any constraint on \( V \), which is necessary to be able to represent the initial state where no array element equals one. Likewise, \( A[I] \) is not defined when \( I \not\in [0, N - 1] \), and the variable \( A_I \) does not enforce any constraint in that case.

* The initial state

\[
I \overset{\text{def}}{=} \{ ([N \mapsto n, V \mapsto v, I \mapsto i], a) \mid n \geq 2, v, i \in \mathbb{Z} \}
\]

is represented as the abstract set:

\[
I^{max} \overset{\text{def}}{=} \{ ([N \mapsto n, V \mapsto v, I \mapsto i, L \mapsto 0, H \mapsto -1, A_I \mapsto x] \mid n \geq 2, v, i, x \in \mathbb{Z} \}
\]

where the values of \( L, H, \) and \( A_I \) do not impose any constraint on the array contents.

* \( C^{max}[V \leftarrow A[0]] \overset{\text{def}}{=} C[L \leftarrow 0; H \leftarrow 0]. \)

We initialize the predicate as \( V = \max A(0, 0) \), indicating that \( V \) is the maximum of the \( A \) between 0 and 0 (i.e., \( A[0] \)).
\( C^\text{max} [ I \leftarrow e ] \stackrel{\text{def}}{=} C[I \leftarrow I + 1; A_I \leftarrow [-\infty, +\infty]] \).

In addition to updating the variable \( I \), we also forget the value of \( A_I \) to model the fact that any information on the prior value of \( A[I] \) is lost, as \( I \) may have changed its value.

\( C^\text{max} [ I \triangleright e? ] \stackrel{\text{def}}{=} C[I \triangleright e? ] \).

This test is unchanged, we do not update our predicate nor \( A_I \).

\( C^\text{max} [ X \leftarrow A[I] ] \stackrel{\text{def}}{=} C[X \leftarrow A_I] \).

We update the relation between \( X \) and the synthetic variable \( A_I \) to remember the relation between \( X \) and \( A[I] \).

\( C^\text{max} [ X > V? ] \stackrel{\text{def}}{=} C[X > V?] \).

This test is also unchanged. Note that, as this test is executed after the assignment \( X \leftarrow A[I] \) in our program, we get \( \rho(X) = \rho(A)_I \) and the semantics will naturally track the relation between \( A[I] \) and \( V \): we get that \( \rho(A_I) > \rho(V) \) holds in all environments after the test.

\( C^\text{max} [ X \leq V? ] \ R \stackrel{\text{def}}{=} \{ f(\rho) \mid \rho \in R \} \) where:

\[
f(\rho) \stackrel{\text{def}}{=} \begin{cases} 
\rho[H \mapsto \rho(I)] & \text{if } \rho(I) = \rho(H) + 1, \ \rho(A_I) = \rho(X) \\
\rho & \text{otherwise}
\end{cases}
\]

This tests uses the knowledge that \( A[I] = X \leq V \) to enlarge the interval \([L, H]\) over which \( V \) equals the maximum of \( A \).

\( C^\text{max} [ V \leftarrow X ] \ R \stackrel{\text{def}}{=} \{ g(\rho) \mid \rho \in R \} \) where:

\[
g(\rho) \stackrel{\text{def}}{=} \begin{cases} 
\rho[V \mapsto \rho(X), H \mapsto \rho(I)] & \text{if } \rho(I) = \rho(H) + 1, \ \rho(A_I) > \rho(V) \\
\rho[V \mapsto \rho(X)] & \text{otherwise}
\end{cases}
\]

Similarly, this assignment uses the knowledge that \( A[I] > V \) to enlarge the interval \([L, H]\) over which \( V \) equals the maximum of \( A \).

\( \star \) As \( \gamma^\text{max} \) is a \( \cup \)-morphism, similarly to \( \mathcal{P}(E^{\text{one}}) \), we have that \( \cup^{\text{max}} \stackrel{\text{def}}{=} \cup \) is the best abstraction of the join.

We could further abstract \( \mathcal{P}(E^{\text{max}}) \) using the polyhedra abstract domain over \((V \cup \{L, H, A_I\}) \rightarrow \mathbb{Z})

Historical notes:

The uniform abstraction has been used for a long time in combination with non-relational abstract interpretations (such as the interval analysis). It has been also used in data-flow analysis, which is inherently non-relational (the abstraction is also called “field-insensitive”). The first use of an uniform abstraction on a relational abstract domain is in the following article, that introduces the expand and remove operators: “D. Gopan, F. DiMaio, N. Dor, T. Reps, M. Sagiv. Numeric domains with summarized dimensions. In Proc. of 10th International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS), LNCS 2988, p. 512–529. Springer, 2004.”


\[ \star \star \star \]
Part II: Exercise

1. ★ Assume that $M$ is a lower Moore family. Given $x \in X$, we use the notation $M_x \overset{\text{def}}{=} \{ y \in M \mid x \sqsubseteq y \}$. We know by hypothesis that $M_x \neq \emptyset$ and that $M_x$ has a least element $\sqcap M_x$ in $M$.

We have by definition $M_\top = \{ y \in M \mid \top \sqsubseteq y \}$. The only element greater than $\top$ is $\top$ itself, so that $M_\top \subseteq \{ \top \}$. As $M_\top \neq \emptyset$, we must have $M_\top = \{ \top \}$. As $M_\top \subseteq M$, we have $\top \in M$.

Consider now $S \subseteq M$ and $M_{\cap S} = \{ y \in M \mid \sqcap S \subseteq y \}$. By Moore family property, $\sqcap M_{\cap S} \subseteq M_{\cap S}$, so that $\sqcap S \subseteq \sqcap M_{\cap S}$. Moreover, as $\forall s \in S : \sqcap S \subseteq s$, we have $S \subseteq M_{\cap S}$, so that $\sqcap M_{\cap S} \subseteq \sqcap S$. Hence, $\sqcap S = \sqcap M_{\cap S} \in M$.

★ For the other direction, assume that $\top \in M$ and $M$ is closed by $\sqcap$. Take $x \in X$ and consider $M_x \overset{\text{def}}{=} \{ y \in M \mid x \sqsubseteq y \}$. As $\top \in M$, $\top \in M_x$ so that $M_x$ is not empty. As $X$ is a complete lattice, $M_x$ has a least element $\sqcap M_x$ in $X$. As $M_x \subseteq M$ and $M$ is closed by $\sqcap$, we have that $\sqcap M_x \in M$.

2. ★ Assume that $M$ is a lower Moore family. We construct the following operator $\rho(x) \overset{\text{def}}{=} \sqcap \{ y \in M \mid x \sqsubseteq y \}$. We now prove that it is an upper closure operator.

Monotony: Assume $x \subseteq x'$, then $\forall y \in M : x' \subseteq y \implies x \subseteq y$. Hence $\{ y \in M \mid x \subseteq y \} \subseteq \{ y \in M \mid x' \subseteq y \}$. This implies $\sqcap \{ y \in M \mid x \subseteq y \} \subseteq \sqcap \{ y \in M \mid x' \subseteq y \}$, i.e., $\rho(x) \subseteq \rho(x')$.

Extensivity: Assume $x \in X$, then $\{ y \in M \mid x \subseteq y \}$ contains only elements greater than $x$, hence $x \sqsubseteq \sqcap \{ y \in M \mid x \subseteq y \} = \rho(x)$.

Idempotence: By Moore family property, we known that $\forall x \in X : \rho(x) \in M$. Take now $x' \in M$. Then, $x' \subseteq \{ y \in M \mid x' \subseteq y \}$. Thus, $x' = \sqcap \{ y \in M \mid x' \subseteq y \} = \rho(x')$. This is true in particular if $x' = \rho(x)$ for some $x \in X$. We thus deduce that $\forall x \in X : \rho(\rho(x)) = \rho(x)$.

Finally, note that when proving the idempotence, we proved that $\forall x \in X : \rho(x) \in M$, which means that $\{ \rho(x) \mid x \in X \} \subseteq M$, and we proved that $\forall x \in M : \rho(x) = x$, which means that $M \subseteq \{ \rho(x) \mid x \in X \}$. Hence, $M = \{ \rho(x) \mid x \in X \}$.

★ To prove the converse, assume that $\rho$ is an upper closure operator and define $M \overset{\text{def}}{=} \{ \rho(x) \mid x \in X \}$. We prove that $M$ is an upper closure operator by proving that it contains $\top$ and is closed by intersection (see question 1).

By extensivity of $\rho$, we have $\top \subseteq \rho(\top)$, which means that $\top = \rho(\top)$, and so, $\top \in M$.

Consider $S \subseteq M$. As $\forall s \in S : \sqcap S \subseteq s$, by monotony, $\forall s \in S : \rho(\sqcap S) \subseteq \rho(s)$ and, by idempotence, $\forall s \in S : \rho(s) = s$ so that $\forall s \in S : \rho(\sqcap S) \subseteq s$, i.e., $\rho(\sqcap S) \subseteq \sqcap S$. By extensivity, however, $\sqcap S \subseteq \rho(\sqcap S)$. We deduce that $\rho(\sqcap S) = \sqcap S$, i.e., $\sqcap S$ is in the image of $\rho$: $\sqcap S \in M$. Hence, $M$ is closed by intersection.

3. ★ $X^\sharp$ is not closed by intersection because $\{ x \mid x \geq 0 \}, \{ x \mid x \leq 0 \} \in X^\sharp$, but $\{ x \mid x \geq 0 \} \cap \{ x \mid x \leq 0 \} = \{ 0 \} \not\in X^\sharp$. Hence it is not a Moore family.

★ By question 2, because $X^\sharp$ is not a Moore family of $\mathcal{P}(\mathbb{Z})$, it is not the image of $\mathcal{P}(\mathbb{Z})$ by any upper closure operator. We saw in the course that the existence of a Galois connection between a set $\mathcal{P}(\mathbb{Z})$ and one of its subset $X^\sharp$ is equivalent to the existence of an upper closure operator whose image of $\mathcal{P}(\mathbb{Z})$ is $X^\sharp$. Hence, we know that there cannot exist any best abstraction function $\alpha \in \mathcal{P}(\mathbb{Z}) \to X^\sharp$.

In particular, the set $\{ 0 \}$ has no best abstraction in $X^\sharp$. Both properties $\{ x \mid x \geq 0 \}$ and $\{ x \mid x \leq 0 \}$ are equally good.

4. ★ A natural way to make $X^\sharp$ a Moore family is to complete it by adding all the missing in-
tersections. In our case, we simply need to add \( \{0\} \). We then retrieve the domain of simple signs.

* Alternatively, we can remove either \( \{x \mid x \geq 0\} \) or \( \{x \mid \leq 0\} \). We obtain a linear three-element domain: either \( \emptyset \subseteq \{x \mid x \geq 0\} \subseteq \mathbb{Z} \) or \( \emptyset \subseteq \{x \mid x \leq 0\} \subseteq \mathbb{Z} \).

We can even remove both and obtain the two-element lattice \( \{\emptyset, \mathbb{Z}\} \), i.e., \( \{\bot, \top\} \).

**Historical notes:**

The fact that Moore families are equivalent to upper closure operators and Galois connections is mentioned, in the context of abstract interpretation, as early as in: “P. Cousot. Méthodes itératives de construction et d’approximation de points fixes d’opérateurs monotones sur un treillis, analyse sémantique des programmes. Thèse ès Sciences Mathématiques, Université Joseph Fourier, Grenoble, France, 21 March 1978.”

---

**Part III: Exercise**

1. * We have \([0, 1] \sqsubseteq [0, 2] \).
   However, \([0, 1] \vee [0, 2] = [0, +\infty] \) while \([0, 2] \vee [0, 2] = [0, 2] \) and \([0, +\infty] \not\subseteq [0, 2] \).
   Hence, \( \vee \) is not monotonic in its first argument.

   * For the second argument, consider \([c, d] \sqsubseteq [c', d'] \), i.e., \( c \geq c' \) and \( d \leq d' \).
   Consider the upper bound \( u \) of \([a, b] \vee [c, d] \) and the upper bound \( u' \) of \([a, b] \vee [c', d'] \).
   If \( b < d \), then \( u = +\infty \), but we also have \( b < d' \), so that \( u' = +\infty = u \).
   If \( b \geq d \), then \( u = b \). As \( u' \in \{+\infty, b\} \), we have \( u' \geq b = u \).
   In all cases \( u \leq u' \). A similar reasoning on the lower bounds \( l \) and \( l' \) gives \( l \geq l' \).
   Hence, \([a, b] \vee [c, d] \subseteq [a, b] \vee [c', d'] \), i.e., \( \vee \) is monotonic in its second argument.

2. In the concrete, the loop invariant states that \( 0 \leq X \leq 10 \).

   * The first iteration with widening gives \([0, 0] \vee [0, 1] = [0, +\infty] \), which is then stable. Hence, the interval domain with the classic widening is only able to prove that \( X \geq 0 \).
   Note that, on this program, using a narrowing would not gain us any precision (however, using a widening with threshold would).

   * When starting the iteration from \([0, 10] \) instead of \([0, 1] \), we get \([0, 10] \vee [0, 10] = [0, 10] \), which is stable. Hence, we find the precise result \([0, 10] \).

3. We now assume that \( \vee \) is a stable widening that is monotonic in its first argument.

   Consider a strictly increasing chain \( y_0 \subset y_1 \subset \cdots \), and construct the derived iteration with widening: \( x_0 \bydef y_0 \) and \( \forall i \in \mathbb{N} : x_{i+1} \bydef x_i \vee y_{i+1} \). We prove by recurrence on \( i \) that, \( \forall i : x_i = y_i \).

   The base case \( i = 0 \) holds by hypothesis.

   Assume now that \( x_i = y_i \). Then, \( x_{i+1} = x_i \vee y_{i+1} = y_i \vee y_{i+1} \).

   As by hypothesis \( y_i \subseteq y_{i+1} \), we have, by monotony, that \( y_i \vee y_{i+1} \subseteq y_{i+1} \vee y_i \).

   By stability, \( y_{i+1} \vee y_{i+1} = y_{i+1} \), which gives: \( y_i \vee y_{i+1} \subseteq y_{i+1} \).

   Moreover, by soundness, \( y_{i+1} \subseteq y_i \vee y_{i+1} \).

   We deduce that \( y_i \vee y_{i+1} = y_{i+1} \), i.e., \( x_{i+1} = y_{i+1} \).

   If the sequence \( y_0, y_1, \ldots \) is infinite and strictly increasing, then so is the sequence \( x_i \). This
violates the convergence property of the widening. We deduce that $D$ cannot have strictly increasing infinite chains.

Note that one way to obtain a monotonic widening is to relax the stability condition. For instance, the widening $\forall a, b : a \triangleright b \overset{\text{def}}{=} \top$ is indeed monotonic, sound and always terminating. It is not stable as $x \triangleright x = \top$ for $x \neq \top$. Moreover, it is not a very interesting widening.

**Historical notes:**


The mention that interesting widenings cannot be monotonic as well as the proof from the last question can be found in: “P. Cousot. Abstract Interpretation Scene-Setting Talk. In Dagstuhl Seminar 14352, Aug. 2014.”