Eötvös Loránd University
Faculty of Informatics

PROGRAMMING

2019/20 1st semester

Lecture 8

ZS. PLUHÁR, H. TORMA

Content

- Construction by copy
- Selection + summation
- Selection+ maximum selection
- Maximum selection + multiple item selection
- Decision+ counting
- Decision+ decision
- Sequence calculations for matrix
- Decision for matrix

Construction by copy

Specification:

Input: $N \in \mathbb{N},\ X_{1..N} \in \mathbb{S}_1^N$

$f:\ \mathbb{S}_1 \to \mathbb{S}_2$

Output: $Y_{1..N} \in \mathbb{S}_2$

Precondition: –

Postcondition: $\forall i (1 \le i \le N) : Y_i = f(X_i)$

Construction by copy works for all patterns of algorithms.

Instead of the X_i elements of the $X \in \mathbb{S}$ sequence in the **input**, you only have to write $f(X_i)$, eg.

$$\sum_{i=1}^{N} X_i \to \sum_{i=1}^{N} f(X_i) \quad \text{or} \quad \max_{i=1}^{N} X_i \to \max_{i=1}^{N} f(X_i)$$

... in the **output**:

$$\underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{multiselect}}(X_i) \to \underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{multiselect}}(f(X_i))$$

Construction by copy

The copy PoA had, however, a version that gives way to new opportunities:

Postcondition: $\forall \mathtt{i}\,(1 \leq \mathtt{i} \leq \mathtt{N}):\ \mathtt{Y}_{\mathtt{p(i)}} = \mathtt{X}_{\mathtt{i}}$ where $\mathtt{p(i)}$ could be eg. $\mathtt{N-i+1}$, which means reversing the order of elements of the sequence.

Many PoAs make use of the order of elements, eg. it found the first among the possible solutions, or gave all the expected elements in the order of input.

With this construction, you could process the sequence backwards.

Copy + Search

Task: Find the **last** element that has a certain attribute.

Specification

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{S}^N$, $A:\mathbb{S}\to\mathbb{L}$

Output: $Exists \in \mathbb{L}$, $Index \in \mathbb{N}$, $Val \in \mathbb{S}$

Precondition: –

Postcondition: $Exists=(\exists i(1\le i\le N):A(X_i))$ and $Exists \to 1\le Ind\le N$ and $A(X_{ind})$ and $Val = X_{ind}$

Specification:

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{S}^N$, $A:\mathbb{S}\to\mathbb{L}$

Output: $Exists \in \mathbb{L}$, $Ind \in \mathbb{N}$

Precondition: –

Postcondition:

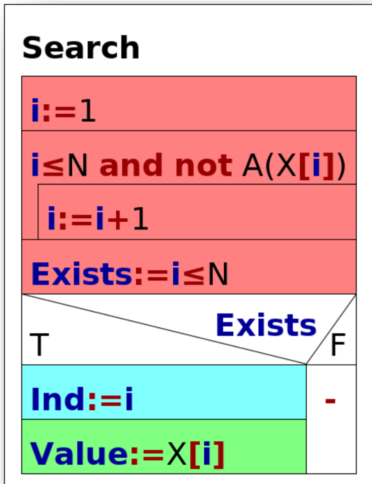
$Exists=\exists i(1\le i\le N): A(X_i)$ and

$Exists \to 1\le Ind\le N$ and $A(X_{ind})$ and

$\forall i(Ind \le i \le N):$ not $A(X_i)$

Copy + Search

Algorithm



Specification:

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{S}^N$, $A: \mathbb{S} \to \mathbb{L}$

Output: $Exists \in \mathbb{L}$, $Ind \in \mathbb{N}$

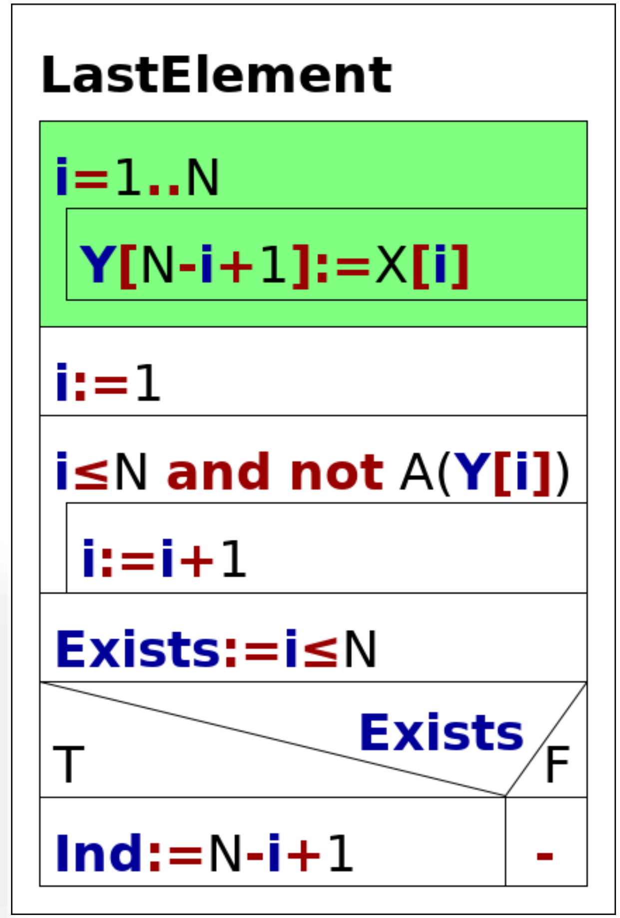
Precondition: –

Postcondition:

$Exists = \exists i (1 \leq i \leq N): A(X_i)$ and

$Exists \to 1 \leq Ind \leq N$ and $A(X_{ind})$ and

$\forall i (Ind \leq i \leq N):$ not $A(X_i)$

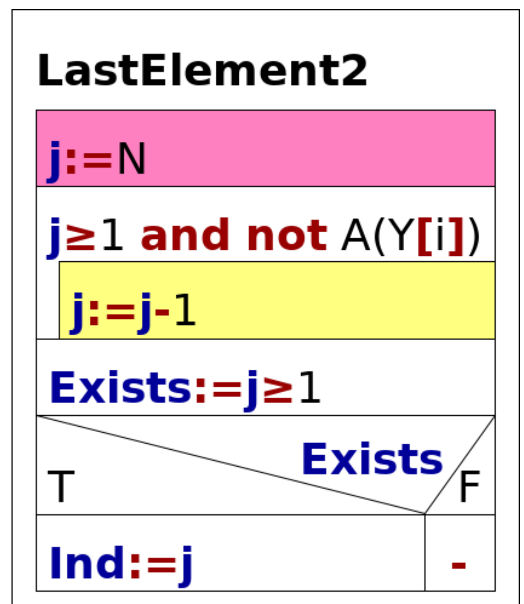


Copy + Search

Let's introduce the `j=N-i+1` notation.

Then in the case of `i=1` $\rightarrow$ `j=N`, and when we increase i, variable j will decrease;

instead of `i≤N`, we use `N-j+1≤N`, i.e. `1≤j`. Then the algorithm is the following:



Multiple item selection + summation

Task: Sum of elements with a certain attribute – conditional summation.

Specification:

- Input: $N \in \mathbb{N},\ X_{1..N} \in \mathbb{Z}^N,\ A:\mathbb{Z} \to \mathbb{L}$
- Output: $S \in \mathbb{Z}$
- Precondition: –
- Postcondition: $S = \sum_{\substack{i=1 \\ A(X_i)}}^{N} X_i$

Multiple item selection + summation

Specification$_a$:

Postcondition$_a$:

$$(Cnt, Y) = \underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{multiselect}}\ i \quad \wedge \quad S = \sum_{i=1}^{Cnt} X_{Y_i}$$

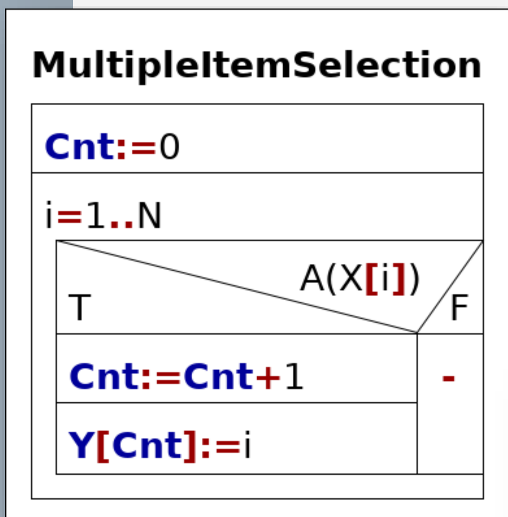
Specification$_b$:

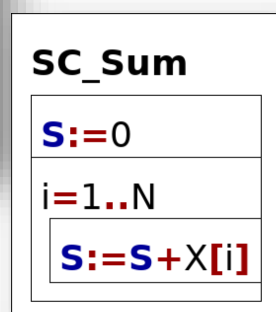
Postcondition$_b$:

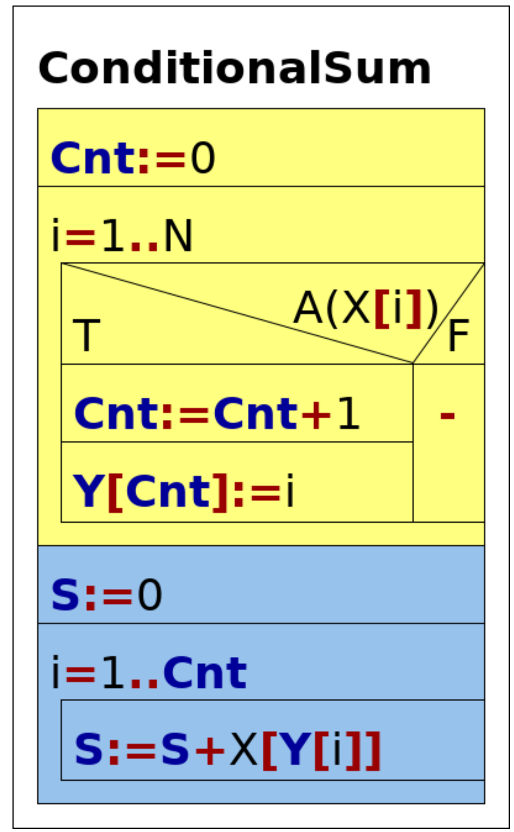
$$(Cnt, Y) = \underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{multiselect}}\ X_i \quad \wedge \quad S = \sum_{i=1}^{Cnt} Y_i$$

Multiple item selection + summation

1. solution idea$_a$: Select all the elements with the given attribute, then add them.

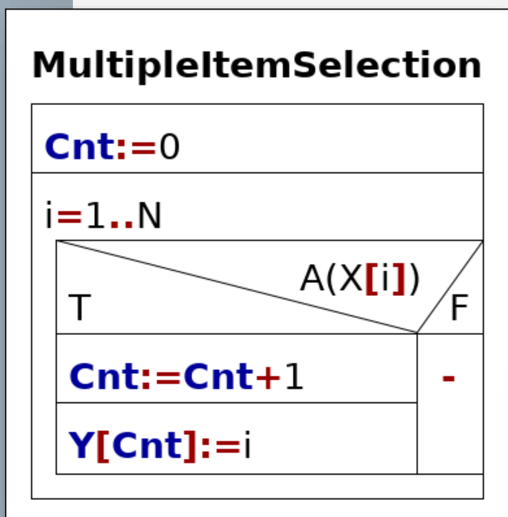


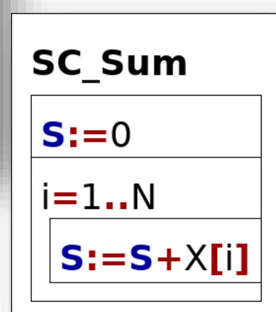


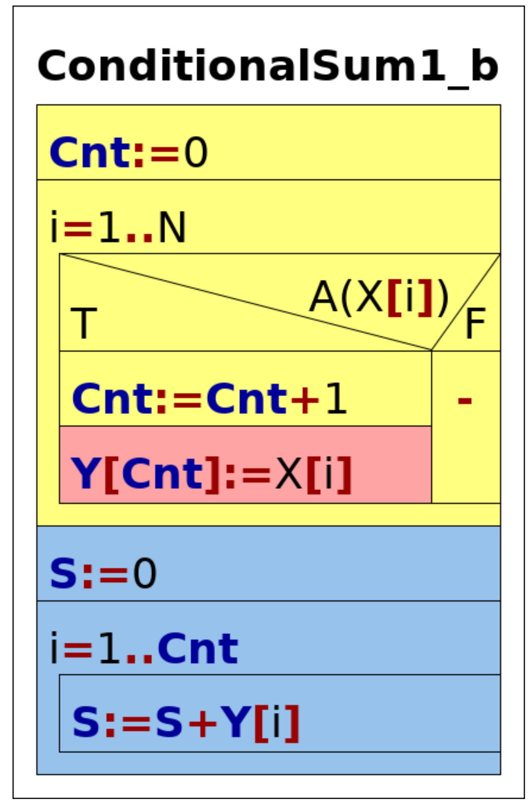


Multiple item selection + summation

1. solution idea$_b$: Select all the elements with the given attribute, then add them.

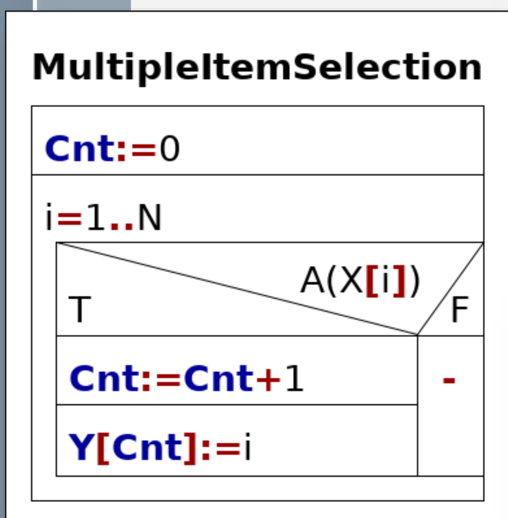


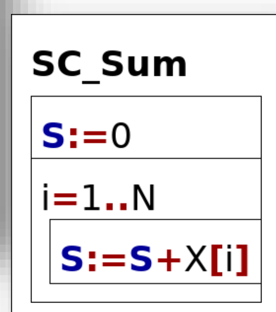


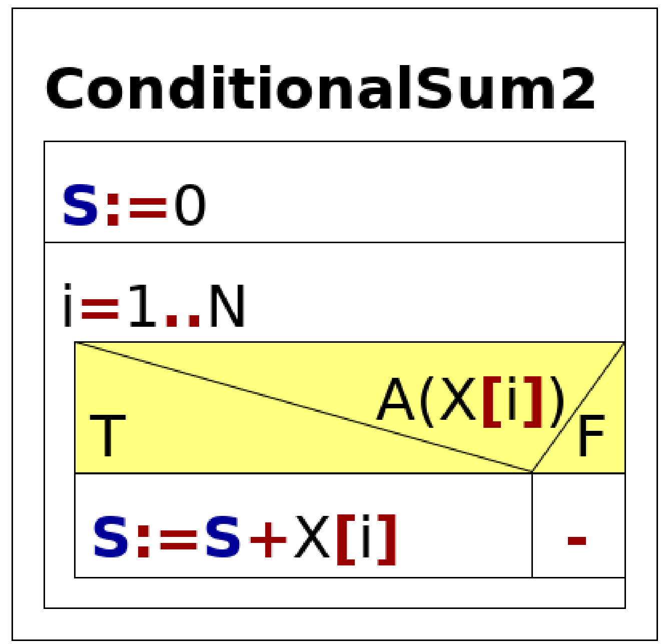


Multiple item selection + summation

2. **solution idea:** Instead of selecting the elements, add them immediately if they meet the condition → no need for storing values/indexes (array Y), no need for counting (variable cnt)







Multiple item selection + maximum selection

Task: Find the maximum of the elements that have a certain attribute – conditional maximum search.

Specification

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{S}^N$, $A: \mathbb{S} \to \mathbb{L}$

Output: $Exists \in \mathbb{L}$, $Index \in \mathbb{N}$, $Val \in \mathbb{S}$

Precondition: –

Postcondition: $Exists = (\exists i(1 \le i \le N): A(X_i))$ and

$Exists \to 1 \le Ind \le N$ and $A(X_{ind})$ and $Val = X_{ind}$

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{S}^N$

Output: $Max \in \mathbb{N}$

Precondition: $N > 0$

Postcondition: $1 \le Max \le N$ and

$\forall i(1 \le i \le N): X_{Max} \ge X_i$

Specification:

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{S}^N$, $A: \mathbb{S} \to \mathbb{L}$

Output: $Exists \in \mathbb{L}$, $MaxI \in \mathbb{N}$

Precondition: –

Postcondition:

$Exists = \exists i(1 \le i \le N): A(X_i)$ and

$Exists \to 1 \le MaxI \le N$ and $A(X_{MaxI})$ and

$\forall i(1 \le i \le N): A(X_i) \to X_{MaxI} \ge X_i$

Multiple item selection + maximum selection

Specification$_2$:

Postcondition$_2$:

$$(\texttt{Exists},\texttt{MaxI}) = \underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{MaxInd}}\, X_i$$

Specification$_3$:

Postcondition$_3$:

$$(\texttt{Exists},\texttt{MaxI},\texttt{Val}) = \underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{Max}}\, X_i$$

Multiple item selection + maximum selection

Towards the solution:

Specification':

Postcondition:

$$(Cnt, Y) = \underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{mutiselect}}\ i \quad \text{and}$$

$$\text{Exists} = (\text{Cnt}>0) \text{ and}$$

$$\text{Exists} \rightarrow 1 \leq \text{MaxI} \leq \text{N and } A(X_{\text{MaxI}}) \text{ and}$$

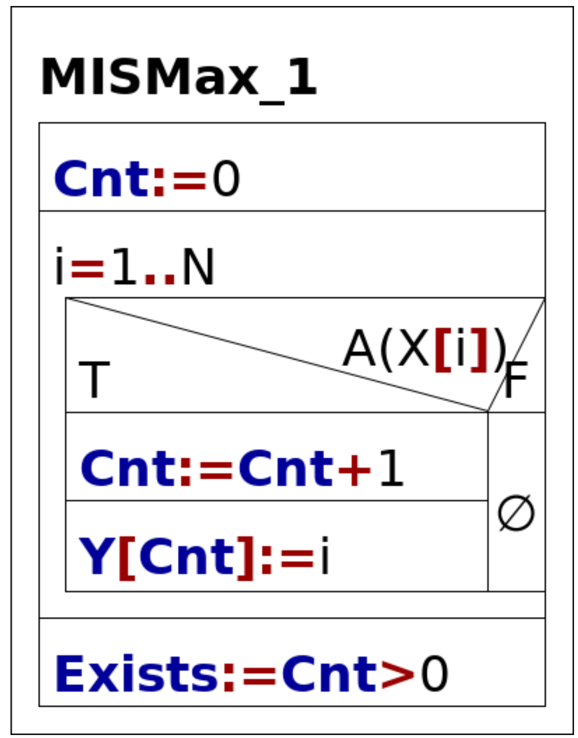
$$MaxI = \underset{i=1}{\overset{Cnt}{MaxInd}}\ X_{Y_i}$$

We could get the **idea** of algorithm: Select the elements with the given attribute, then, select the maximum, if it makes sense.

Multiple item selection + maximum selection

Algorithm:

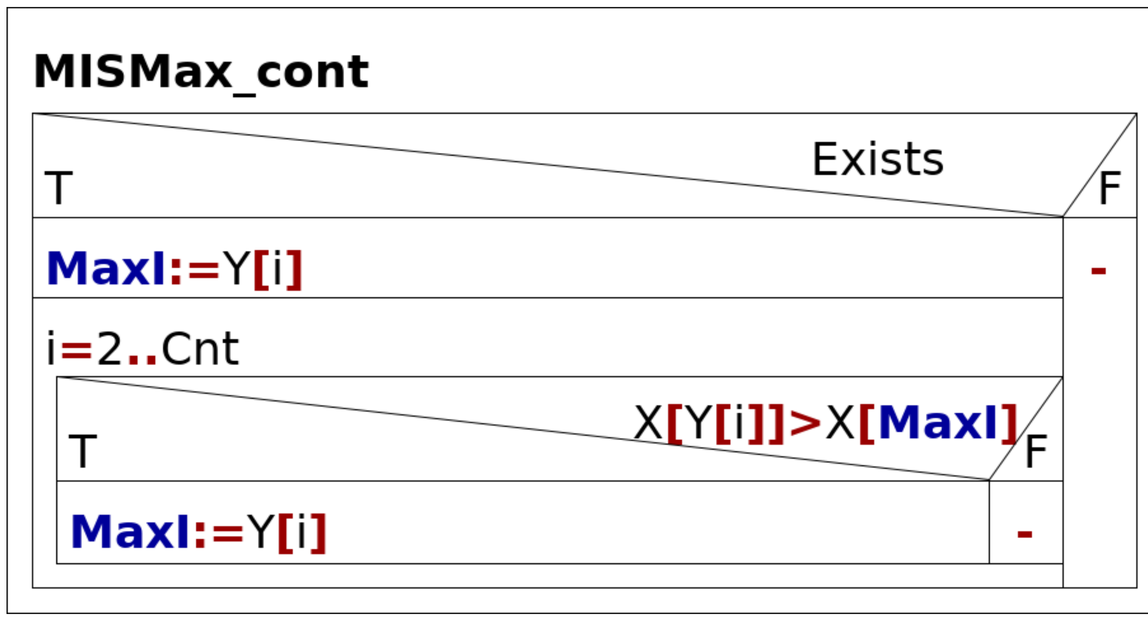
Select the elements with the given attribute…



Multiple item selection + maximum selection

Algorithm:

… then, select the maximum, if it makes sense.

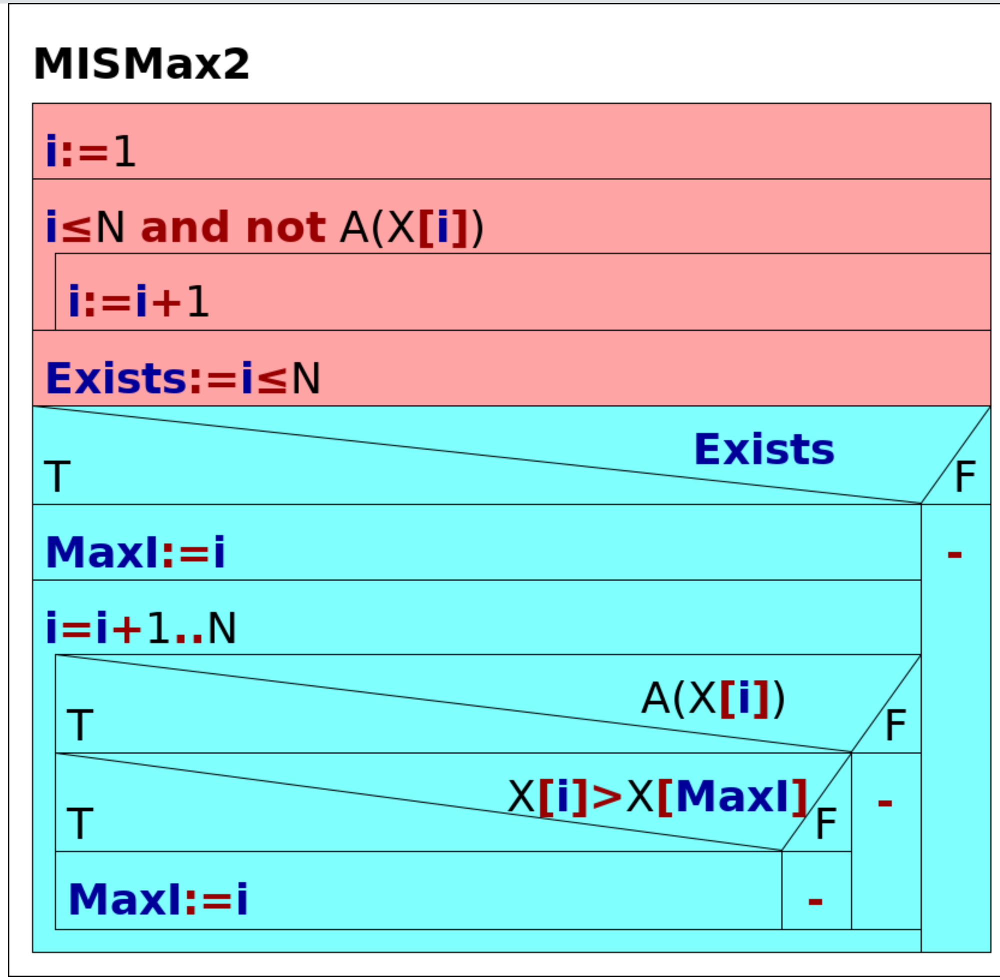


Multiple item selection + maximum selection

2. solution idea$_a$ (and algorithm):

Let’s start from the PoA-s that we noticed in the specification. Instead of multiple item selection,

- let’s find the first element with A attribute, then
- select the maximum of such elements.

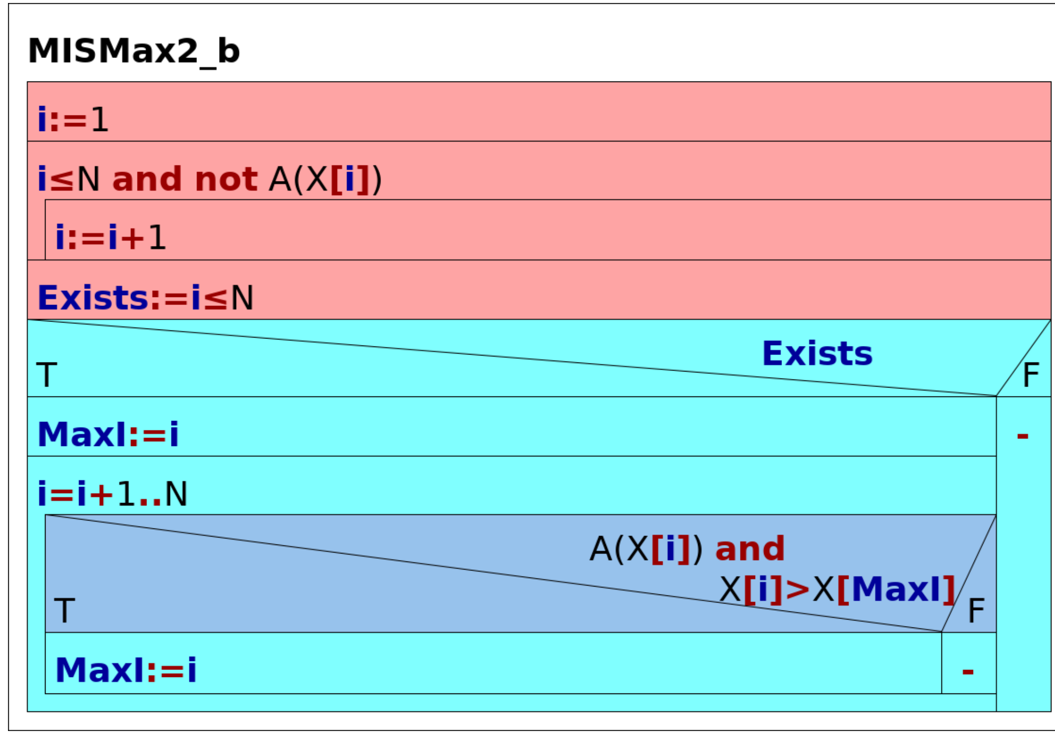


Multiple item selection + maximum selection

2. solution idea$_b$ (and algorithm):

Let’s start from the PoA-s that we noticed in the specification. Instead of multiple item selection,

- let’s find the first element with A attribute, then
- select the maximum of such elements.



Multiple item selection + maximum selection

3. solution idea (and algorithm):

Instead of selecting the elements first, let's find the maximum immediately. For this, we need a **fictive 0. element** that is **less than all the other elements**..

```
MISMax3
X[0]:=-∞
MaxI:=0
i=1..N
    A(X[i]) and X[i]>X[MaxI]
    T: MaxI:=i
    F: -
Exists:=MaxI>0
```

Maximum selection + multiple item selection

Task: Selecting **all** maximum elements.

Specification:

Using only the first Cnt elements

- **Input:** $N \in \mathbb{N},\ X_{1..N} \in \mathbb{S}^N$
- **Output:** $Cnt \in \mathbb{N},\ MaxI_{1..N} \in \mathbb{N}^N$
- **Precondition:** $N>0$
- **Postcondition:** $Cnt = \sum_{\substack{i=1 \\ X_i = X_{(MaxI_1)}}}^{N} 1$ and

$$\forall i(1 \leq i \leq Cnt): \forall j(1 \leq j \leq N): X_{MaxI_j} \geq X_j$$

and $MaxI \subseteq (1,2,\ldots,N)$

Maximum selection + multiple item selection

Task: Selecting **all** maximum elements.

Specification:

`MaxVal` is a local variable of Postcondition

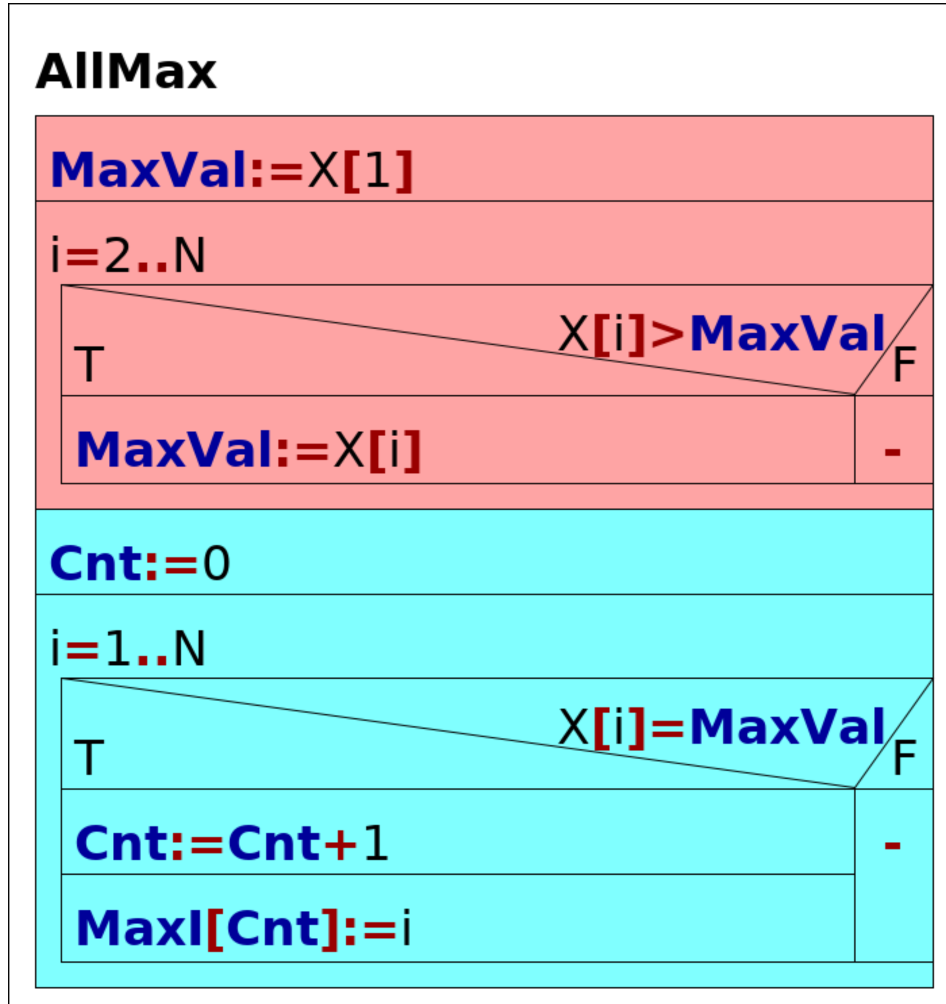
- **Input:** $\texttt{N}\in\mathbb{N},\ \texttt{X}_{1..N}\in\mathbb{S}^{N}$
- **Output:** $\texttt{Cnt}\in\mathbb{N},\ \texttt{MaxI}_{1..N}\in\mathbb{N}^{N}$
- **Precondition:** $\texttt{N>0}$
- **Postcondition:** $MaxVal = \underset{i=1}{\overset{N}{MaxV}}\, X_i$ and

$$(\texttt{Cnt},\texttt{MaxI}) = \underset{\substack{i=1 \\ X_i = MaxVal}}{\overset{N}{multiselect}}\ i$$

Maximum selection + multiple item selection

1. solution idea (and algorithm):

First, let's find the maximum. Then select all the elements that are equal to it.



Maximum selection + multiple item selection

2. solution idea (and algorithm):

Let's select the items equal to the current maximum. If we find a „bigger maximum", we overwrite the previously selected elements.

Cnt:=1; MaxI[1]:=1; MaxVal:=X[1]		
i=2..N		
	X[i]>MaxVal	X[i]=MaxVal
	Cnt:=1	Cnt:=Cnt+1
	MaxI[1]:=i	MaxI[Cnt]:=i
	MaxVal:=X[i]	—

Decision + counting

Task: Are there **at least K** elements with the given attribute in a sequence?

Specification:

Input: $N, K \in \mathbb{N}$, $X_{1..N} \in \mathbb{S}^N$, $A: \mathbb{S} \to \mathbb{L}$

Output: $Exists \in \mathbb{L}$

Precondition: $K>0$

Postcondition:

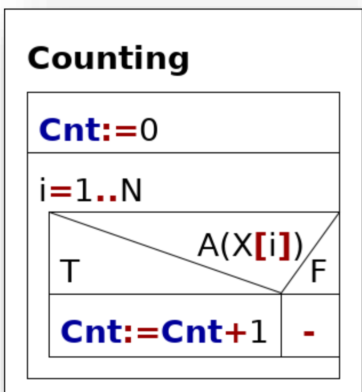
$$Cnt = \sum_{\substack{i=1 \\ A(X[i])}}^{N} 1 \quad \wedge \quad Exists = (Cnt \geq K)$$

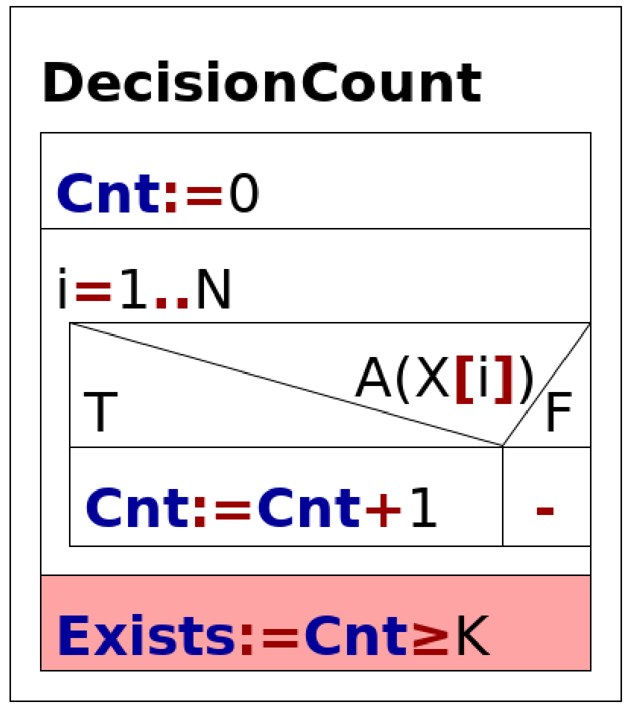
`Cnt` is a local variable of Postcondition

Decision + counting

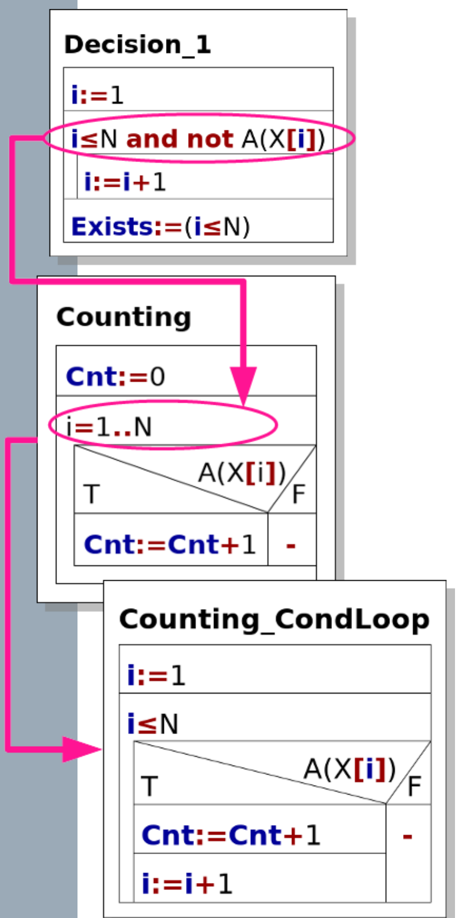
1. solution idea (and algorithm):

Count how many elements have the given attribute, then see whether it's more than K. (So, in fact, there is no decision PoA.)



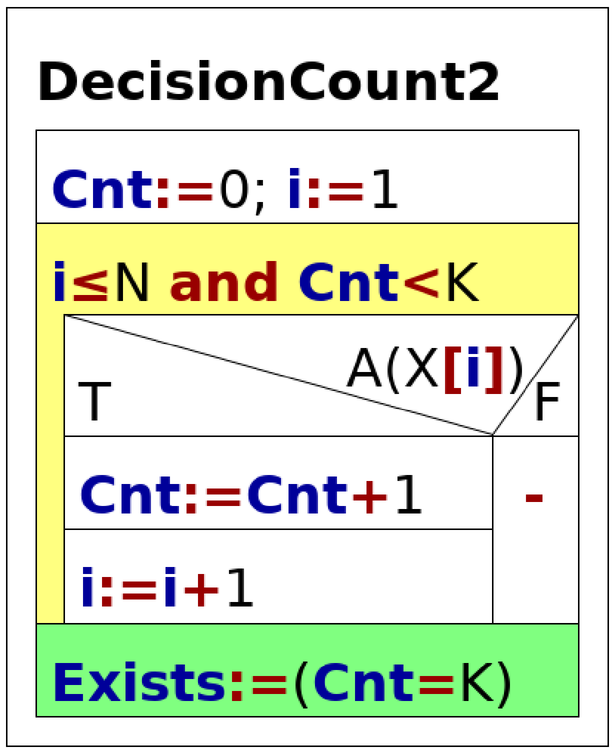


Decision + counting



1. solution idea (and algorithm):

If we have found K elements of the given attribute, then do not check any longer.



postcondition:

$$\exists i (1 \le i \le N):$$

$$Exists = \left(\sum_{\substack{j=1 \\ A(X_j)}}^{i} 1 = K \right)$$

Search + counting

Task: In a sequence, **which** is the **K.** element with a given attribute (if there are at least K elements with that attribute)?

Specification:

Input: $N,K\in\mathbb{N},\ X_{1..N}\in\mathbb{S}^N,\ A:\mathbb{S}\to\mathbb{L}$

Output: $Exists\in\mathbb{L},\ KI\in\mathbb{N}$

Precondition: $K>0$

Postcondition: $Exists=\exists i(1\le i\le N):\sum_{\substack{j=1\\A(X_j)}}^{i}1=K$ and

$$Exists\to 1\le KI\le N \text{ and } \sum_{\substack{j=1\\A(X_j)}}^{KI}1=K\ \wedge\ A(X_{KI})$$

Search + counting

1. **solution idea**:

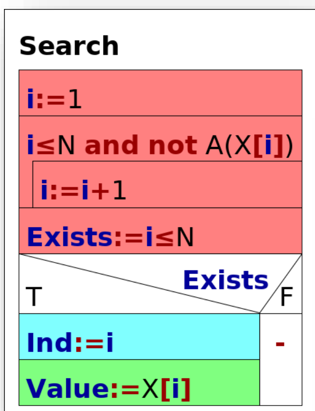
- First, let's **count** how many elements have the given attribute, then observe **whether it's at least K**. But this is not enough, we have to go back, and find the K. element.

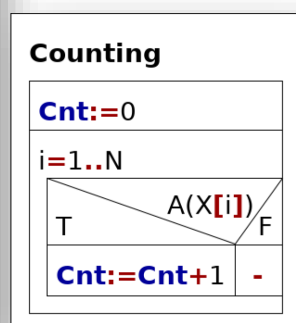
- A solution that seems to be working: instead of counting, **multiple item selection** is needed. Then we **do not need search**. But this is memory consuming , and a long process.

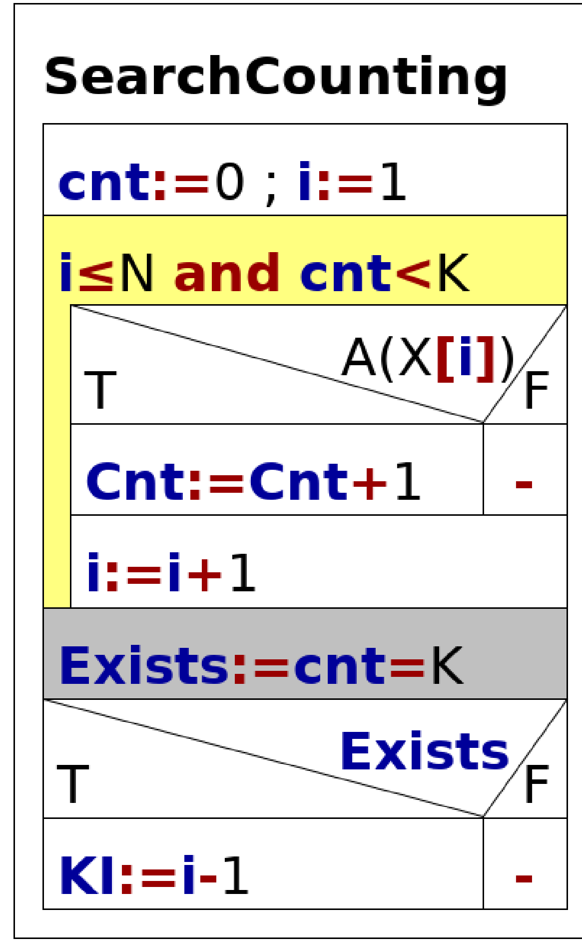
Search + counting

2. solution idea:

If we have found K elements of the given attribute, do not check any longer: search until the K. element. Then note the index of the K. element.







Search + copy

Task: Selection of all the elements of a sequence that are before an element having attribute A. (If no element with the attribute, then all the elements.)

Specification:

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{S}^N$, $A: \mathbb{S} \to \mathbb{L}$

Using only the first Cnt elements

Output: $Cnt\mathbb{N}$, $Y_{1..N} \in \mathbb{S}^N$

Precondition: –

Postcondition: $Exists = \exists i(1 \le i \le N): A(X_i)$ and

$(Exists$ and $1 \le Cnt < N$ and $A(X_{Cnt+1})$ or $Cnt = 1)$

and $\forall i(1 \le i \le Cnt)$: not $A(X_i)$ and $Y_i = X_i$

Search + copy

1. solution idea:

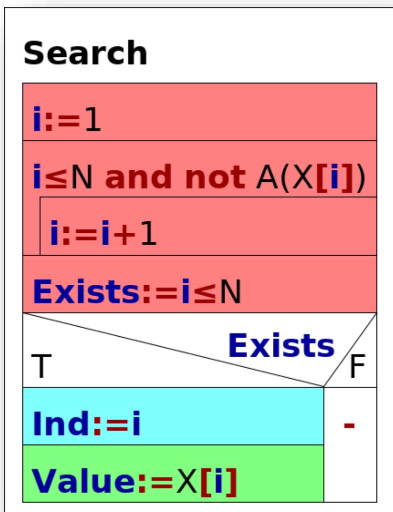
Let's find the first element with the given attribute, then copy the elements before it.

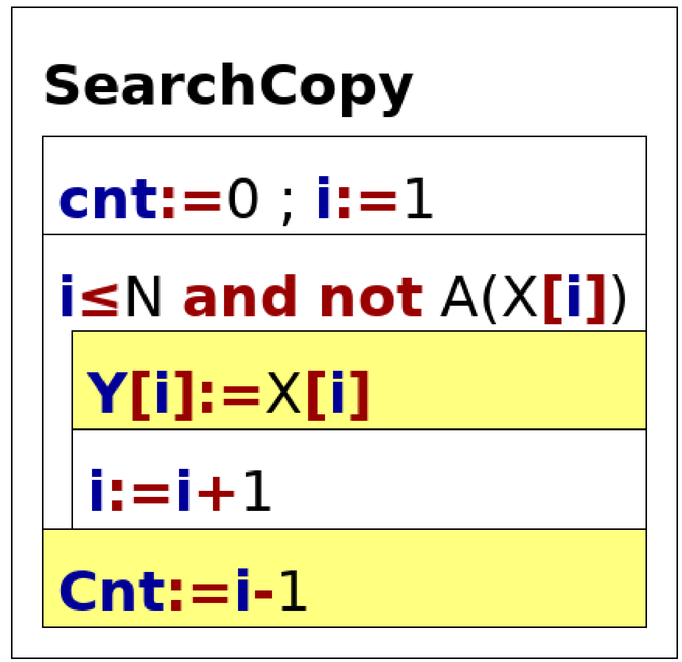
… long process!

Search + copy

2. solution idea:

Copy the elements **while searching** for the first element with the given attribute:





Decision + decision

Task: Do two sequences have a common element?

Specification:

- **Input**: $N, M \in \mathbb{N},\ X_{1..N} \in \mathbb{S}^N,\ Y_{1..M} \in \mathbb{S}^M$
- **Output**: $Exists \in \mathbb{L}$
- **Precondition**: –
- **Postcondition**:

$$Exists = \exists i (1 \le i \le N): \exists j (1 \le j \le M): Y_i = X_j$$

Postcondition': $Exists = \overset{N}{\underset{i=1}{\exists}} \left(\overset{M}{\underset{j=1}{\exists}} X_i = Y_j \right)$

Decision + decision

1. solution idea:

- Let's collect all the common elements (intersection), and if the count of elements is at least 1, then there is a common element

Specification: The „rewriting" of postcondition:

- The subresult of intersection: $\texttt{Cnt}\in\mathbb{N}$
- The modifed postcondition: the postcondition of intersection and `Exists=Cnt>0`

Comment:

intersection = multiple item selection + decision

Decision + decision

2. solution idea:

If there is at least one common element, do not check any longer.

Decision_1

```
i:=1
i≤N and not A(X[i])
    i:=i+1
Exists:=(i≤N)
```

Decision_2

```
i:=0
Exists:=False
i<N and not Exists
    i:=i+1
    Exists:=A(X[i])
```

Decision_Decision

```
i:=0
Exists:=False
i≤N and not Exists
    i:=i+1
    j:=1
    j≤M and X[i]≠Y[j]
        j:=j+1
    Exists:=j≤M
```

Summation for matrix

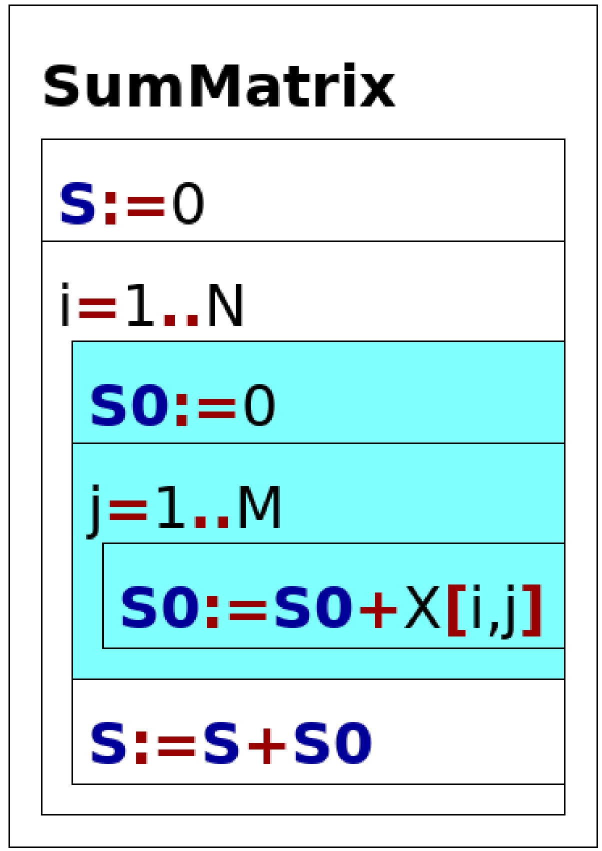
Task: The sum of elements of a matrix.

Specification:

- **Input:** $N, M \in \mathbb{N}, \ X_{1..N,1..M} \in \mathbb{Z}^{N \times M}$
- **Output:** $S \in \mathbb{Z}$
- **Precondition:** –
- **Postcondition:** $S = \sum_{i=1}^{N} \left(\sum_{j=1}^{M} X_{i,j} \right)$

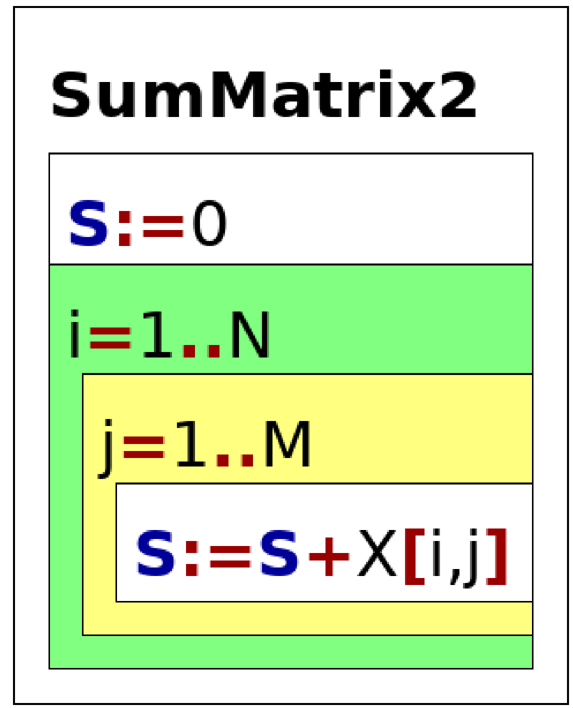
Summation for matrix

Algorithm: Two embedded **summation** PoA-s.



Summation for matrix

Algorithm: Even smaller difference from basic pattern: just two embedded loops



Comment: copy, count and maximum selection can be done similarly for matrices

Decision for matrix

Task: Is there an element with a given attribute in the matrix?

Specification:

Input: $N, M \in \mathbb{N}$, $X_{1..N,1..M} \in \mathbb{S}^{N \times M}$, $A: \mathbb{S} \to \mathbb{L}$

Output: $Exists \in \mathbb{L}$

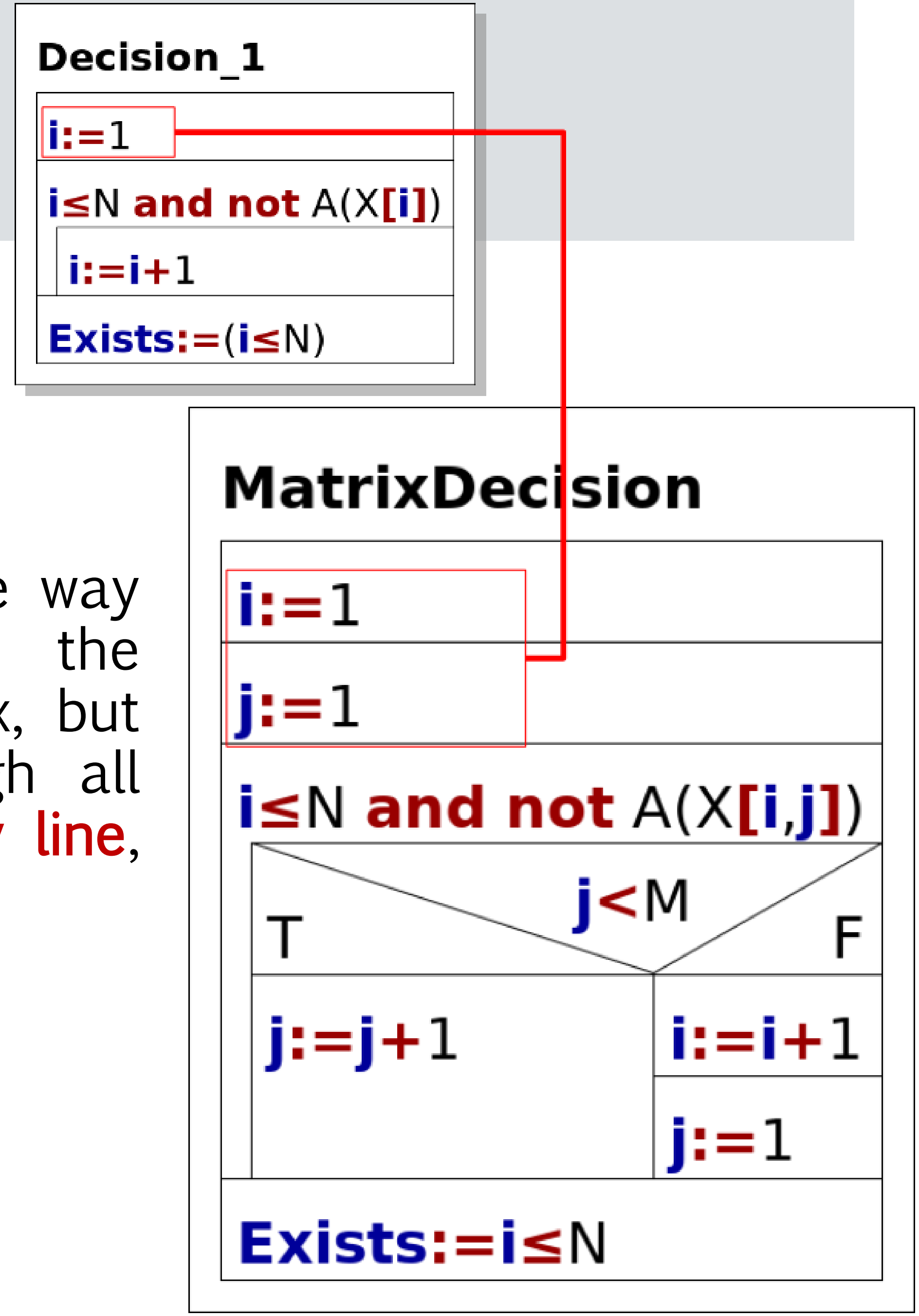
Precondition: –

Postcondition:

$$Exists = \exists i (1 \le i \le N): \exists j (1 \le j \le M): A(X_{i,j})$$

Postcondition': $Exists = \underset{i=1}{\overset{N}{\exists}} \left(\underset{j=1}{\overset{M}{\exists}} A(X_{i,j}) \right)$

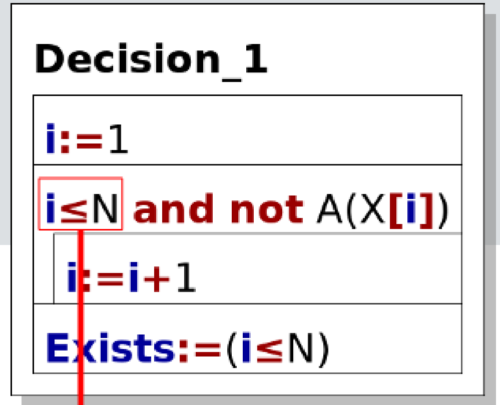
Decision for matrix



Algorithm:

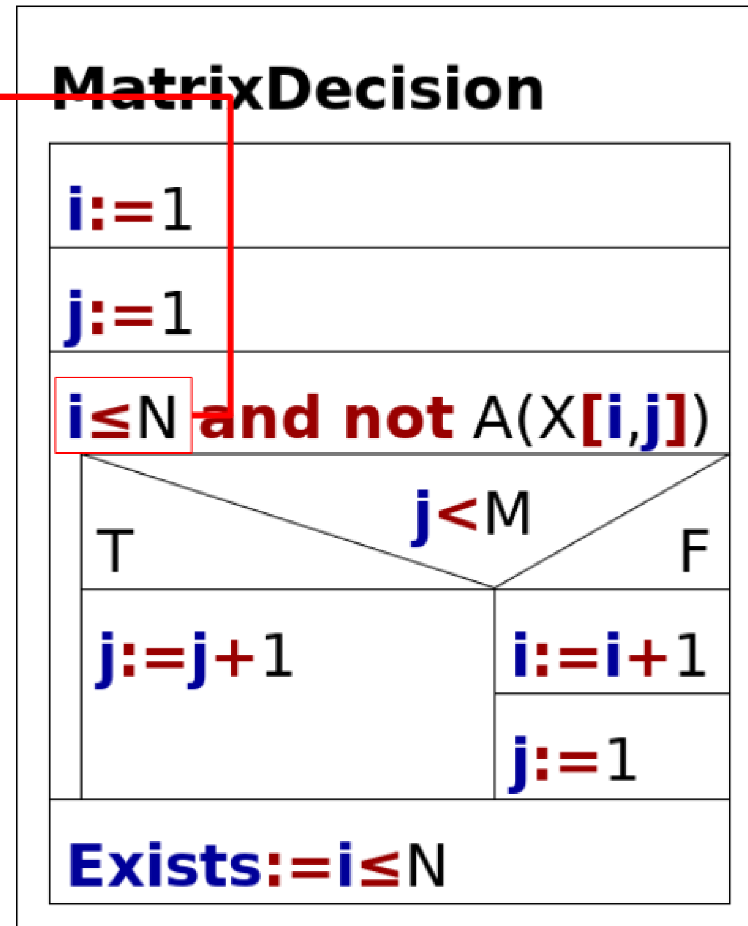
You have to define the way of going through the **elements** of the matrix, but not necessarily through all the elements – **line by line**, from left to right

Decision for matrix

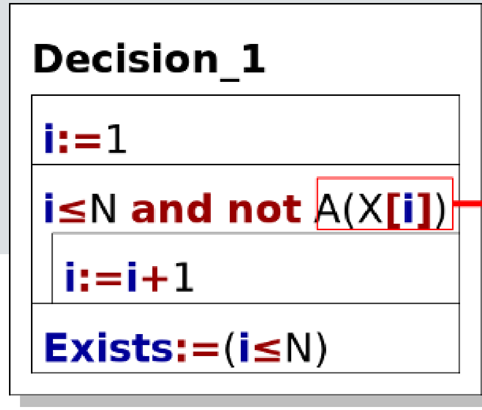


Algorithm:

You have to define the way of going through the elements of the matrix, but not necessarily through all the elements – line by line, from left to right

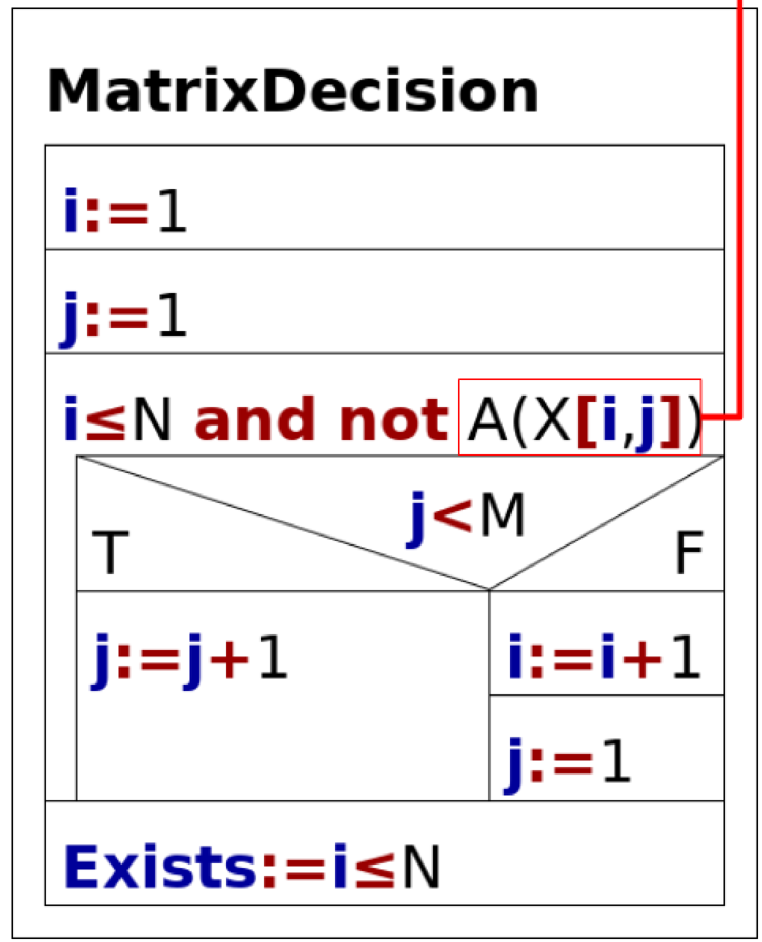


Decision for matrix

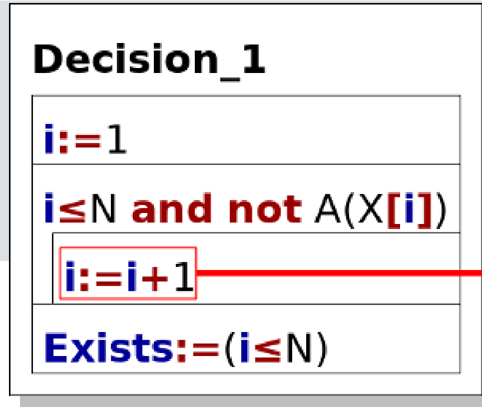


Algorithm:

You have to define the way of going through the elements of the matrix, but not necessarily through all the elements – line by line, from left to right



Decision for matrix



Algorithm:

You have to define the way of going through the **elements** of the matrix, but not necessarily through all the elements – **line by line**, from left to right

Comment: selection and search can be done similarly