Eötvös Loránd University
Faculty of Informatics

PROGRAMMING

2019/20 1st semester

Lecture 3

Zs. Pluhár, H. Torma

Content

- Structures/records
- Patterns of Algorithms – the essence
  - Sequence calculations
  - Counting
  - Decision
  - Selection
  - Search
  - Maximum search

Structures/Records

Task: Let's determine which quarter-plain contains a given point P!

Towards the solution – **specification**:

Points on a plane ($\mathbb{R} \times \mathbb{R}$ $\times$:direct product) are defined with their X and Y coordinates.

In specification: $P \in Point,\ Point = X \times Y,\quad X, Y = \mathbb{R}$

In the specification, we could refer to them by their name:

$$P = (P.x,\ P.y)$$

$$Thus:\ P.x \in X,\ P.y \in Y$$

Structures/Records

Task: Let's determine which quarter-plain contains a given point P!

Specification:

Input: $P \in Point$, $Point = X \times Y,\ X, Y = \mathbb{R}$

definition of the new set

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

Precondition: –

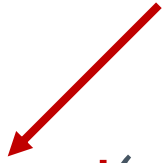
Postcondition:

$$P.x \geq 0 \text{ and } P.y \geq 0 \rightarrow QP = 1 \text{ and}$$
$$P.x < 0 \text{ and } P.y \geq 0 \rightarrow QP = 2 \text{ and}$$
$$P.x < 0 \text{ and } P.y < 0 \rightarrow QP = 3 \text{ and}$$
$$P.x \geq 0 \text{ and } P.y < 0 \rightarrow QP = 4$$

Structures/Records

Specification → algorithm$_{\text{data description:}}$

$P \in Point,\ Point = X \times Y,\ X, Y = \mathbb{R}$



Type TPoint=**Record**(x,y:**Real**)

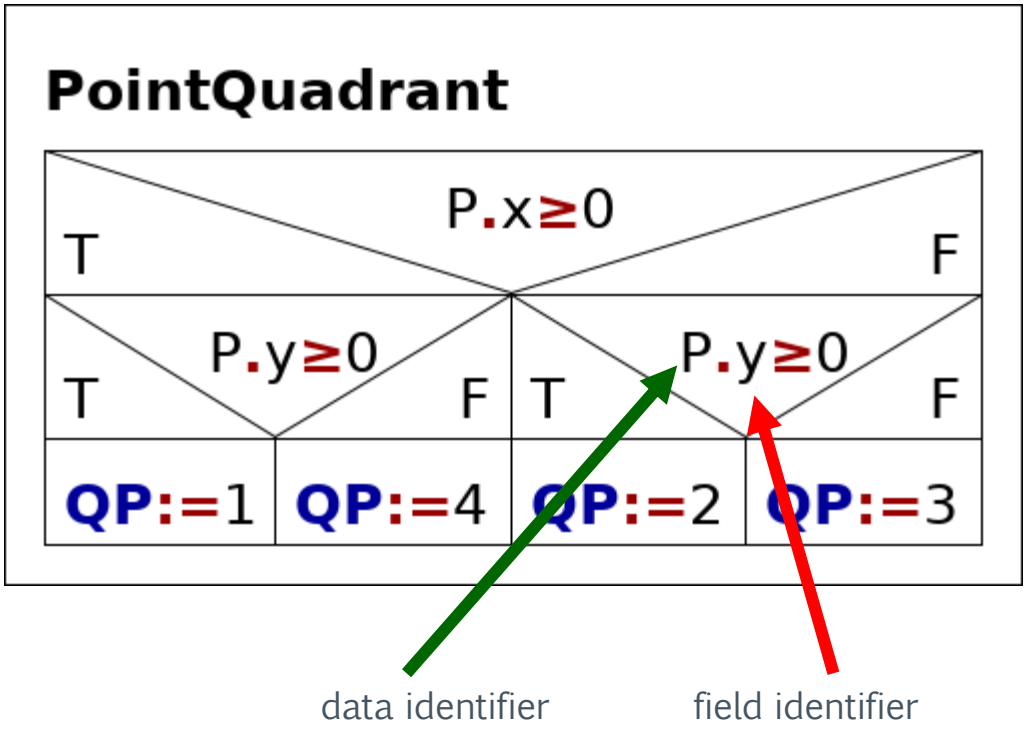
Variable P:Tpoint

(TPoint is a new compound **data type.**)

Records, similarly to arrays, are compound data types. However, the elements are not referred to by an index, but by their **name.**(P.**x**, P.**y**).

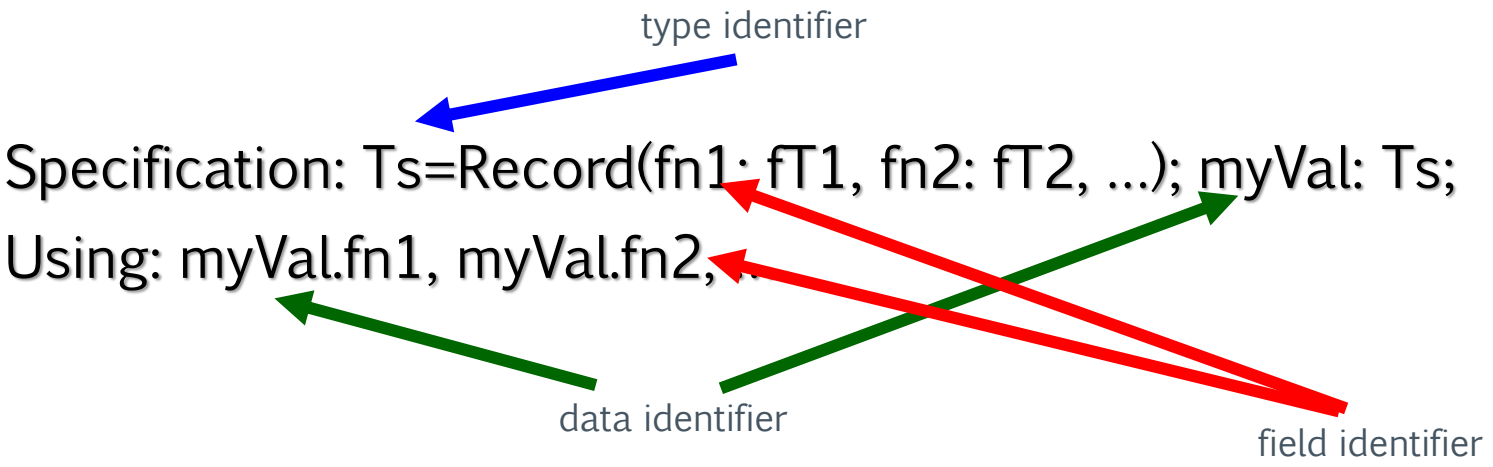
Structures/Records

Specification → algorithm$_{\text{activity:}}$



Structures/records

- define the struct by giving it a **name** and then adding the **member data types**
- *Structs* (also called *records*, or classes) group elements which don't need to all be the same type, and are **accessed by field** (member) name



Structures/records

Specification → algorithm → **code**:

Type definition: TPoint=**Record**(x,y:**Real**)

C++ type *definition*:

```
typedef struct {double x,y;} Tpoint;
                    //field type field identifiers type-identifier
```

Variable – type declaration P:TPoint

C++ type *declaration*:

```
TPoint P; //type-identifier data-identifier
```

Patterns of Algorithms (PoA) – the essence

Its **aim:**

- A proven template as a basis, on which we can build our solutions later. (Development will be quicker and safer)

Its **structure**:

1. abstract task specification [+program specification]
2. abstract algorithm

Previous comment: the input is at least a sequence

Patterns of Algorithms (PoA) – the essence

How we use them:

1. create a specification for the given task
2. guess the PoA from specification
3. map the parameters of the given task to the parameters of the corresponding abstract task
4. "generate" the task-specific algorithm based on the algorithm of the PoA, by changing the parameters as per point 3.
5. create a more efficient algorithm using program transformations

Patterns of Algorithms

What is PoA? It is the general solution of a typical programming task.

- sequence → single value
- sequence → sequence
- sequence → sequences
- sequences → sequence

1. Sequence calculations

Tasks (examples):

- We know the monthly income and expenses of a person. Let's calculate by how much money his asset will change by the end of the year!
- We know the lap times of a car racer. Let's determine his average lap time!
- Let's calculate the factorial of N!
- We know which students take part in extracurricular activities in school listed by activity. Give all the students, who attend activities.
- We know N words. Give the sentence we get by joining them.

1. Sequence calculations

Grouping:

- Sum of numbers: “asset”, “lap times”
- Product of numbers: “factorial”
- Union of sets: “extracurriculars”
- Words joined: “words”

What is common?

We have N "something", and we have to calculate a single "something" from them.

eg. Σ – income/lap time; Π – factorial; $\cup$ – extracurriculars ; $\&$ – words

1. Sequence calculations

Specification:

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

Output: $SC \in \mathbb{S}$

Precondition: $N \geq 0$

Postcondition: $SC = F(X_{1..N})$

$F: \mathbb{S}^N \rightarrow \mathbb{S}$

Σ – sum of N elements;

Π – product of N elements;

$\cup$ - union of N sets

& – concatenation of N texts …

$\mathbb{S}$: an arbitrary set;
$\mathbb{S}^N = \{(s_1,\ldots,s_N) \mid s_i \in \mathbb{S}\}$

$(X_1,\ldots,X_N)$ sequence

1. Sequence calculations

Specification (summation):

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

$\mathbb{S}$: $\mathbb{Z}$ or $\mathbb{R}$

Output: $Sum \in \mathbb{S}$

Precondition: $N \geq 0$

Postcondition: $Sum = \sum_{i=1}^{N} X_i$

Definition:

$$\sum_{i=1}^{N} X_i := \begin{cases} 0 & , N = 0 \\ \left(\sum_{i=1}^{N-1} X_i \right) + X_N & , N > 0 \end{cases}$$

1. Sequence calculations

General problem:
F: operation with N parameters, where N is variable

Solution:
Decomposition to *2-parameter operations* (e.g. Σ to +) and to a *neutral-value* (in the case of + it is 0).

$$F(X_{1..N}) = f(F(X_{1..N-1}), X_N) \quad \text{if } N>0$$
$$F(-) = F_0 \quad \text{otherwise}$$

1. Sequence calculations

Specification (general)

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

Output: $SC \in \mathbb{S}$

Precondition: –

Postcondition: $SC = \mathbf{F}(\mathbf{X}_{1..N})$

Definition: $F : \mathbb{S}^* \rightarrow \mathbb{S}$

$$F(X_{1..N}) := \begin{cases} F_0 & , N = 0 \\ f(F(X_{1..N-1}), X_N) & , N > 0 \end{cases}$$

$f: \mathbb{S} \times \mathbb{S} \rightarrow \mathbb{S}$, $F_0 \in \mathbb{S}$

1. Sequence calculations

Specification (generalising further)

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

Output: $SC \in \mathbb{S}_2$

Precondition: –

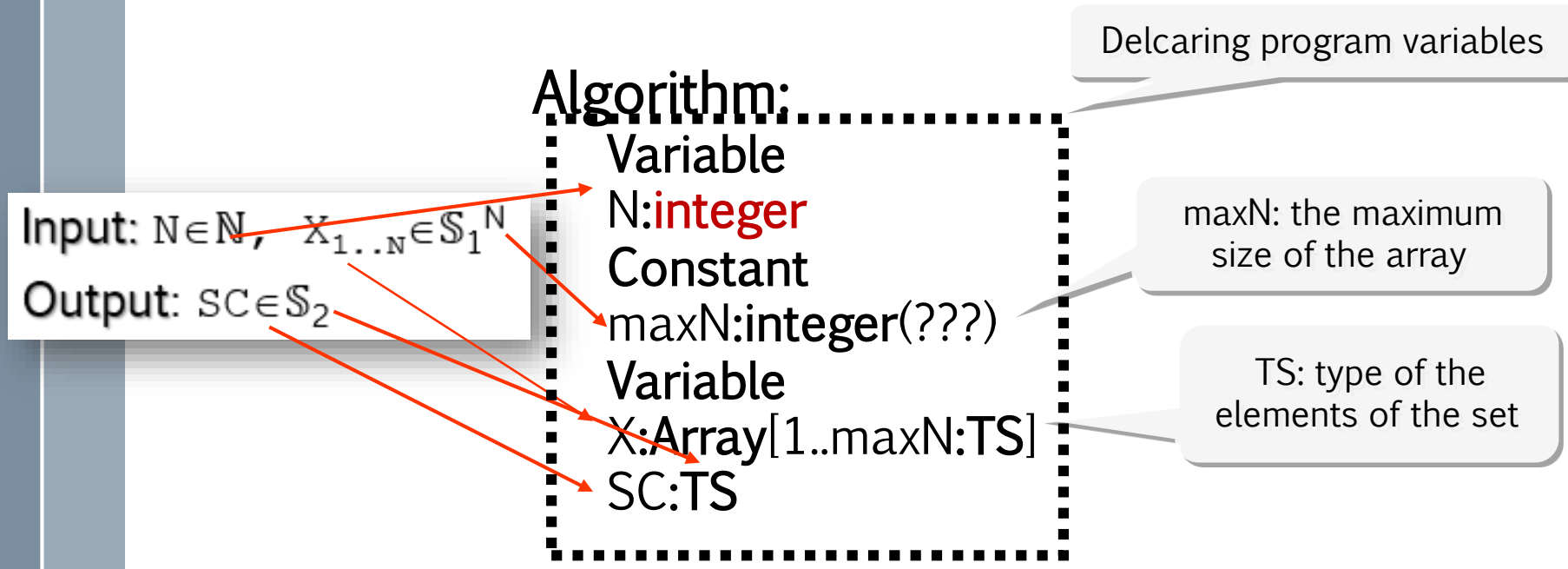
Postcondition: $SC = \mathbf{F}(\mathbf{X}_{1..N})$

Definition: $F : \mathbb{S}_1^* \rightarrow \mathbb{S}_2$

$$F(X_{1..N}) := \begin{cases} F_0 & , N = 0 \\ f(F(X_{1..N-1}), X_N) & , N > 0 \end{cases}$$

$$f: \mathbb{S}_2 \times \mathbb{S}_1 \rightarrow \mathbb{S}_2, \quad F_0 \in \mathbb{S}_2$$

1. Sequence calculations

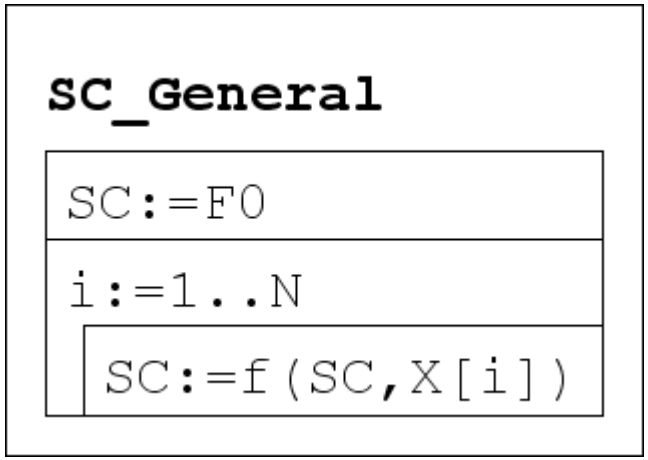


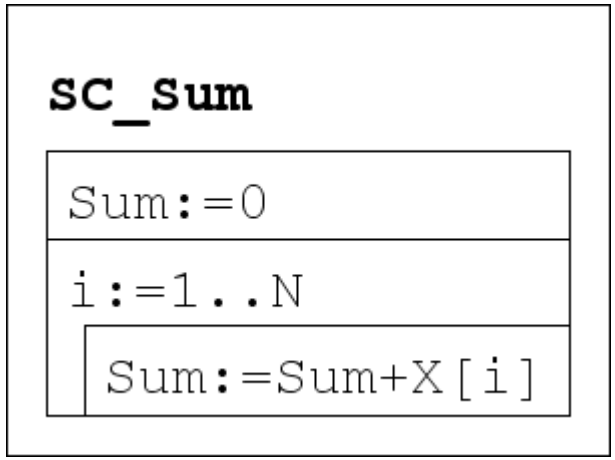
We are going to declare the array for the sequences in the PoAs in a **static** way

1. Sequence calculations

Algorithm

For example, in the case of Σ this looks like:





1. Sequence calculations - example

We know the monthly income and expenses of a person. Let's calculate by how much money his asset will change by the end of the year!

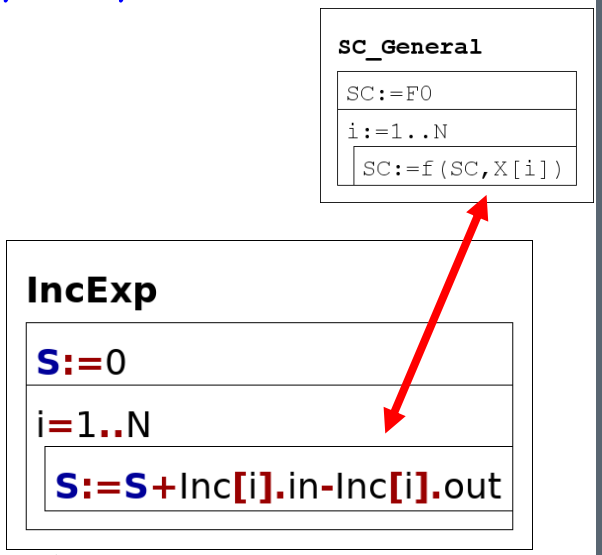
Specification

Input: $N \in \mathbb{N}$, $Inc_{1..N} \in (in \times out)^N$, $in, out = \mathbb{N}$

Output: $S \in \mathbb{Z}$

Precondition: –

Postcondition: $S = \sum_{i=1}^{N} Inc_i.\text{in} - Inc_i.\text{out}$



1. Sequence calculations

Lessons:

- The precondition of a given task might be more strict than that of the used pattern of algorithm.
- The postcondition of a given task might be weaker than that of the used pattern of algorithm (we will meet this).
- Instead of indexing from 1 to N, we could index from B to E.
- Instead of working with elements of an array, the function could ask for the value of the i^{th} element (from more arrays, from more array elements, or with a function independent of the array).

2. Counting

Tasks (examples):

- We know the monthly income and expenses of a person. Let's count the number of months where his assets grow!
- Let's calculate the number of divisors of an integer!
- Let's determine how many letter "a" can be found in the name of a person!
- Based on the annual daily statistics, let's count the number of days when frozen!
- We have birth date (month) data from N people. Let's count how many of them have birthday during the winter!


2. Counting

What is common?

We have N "something", and we have to count how many of them have a given attribute.


2. Counting

$\mathbb{S}$ is an arbitrary set

Specification

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

A is an arbitrary attribute (function)

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

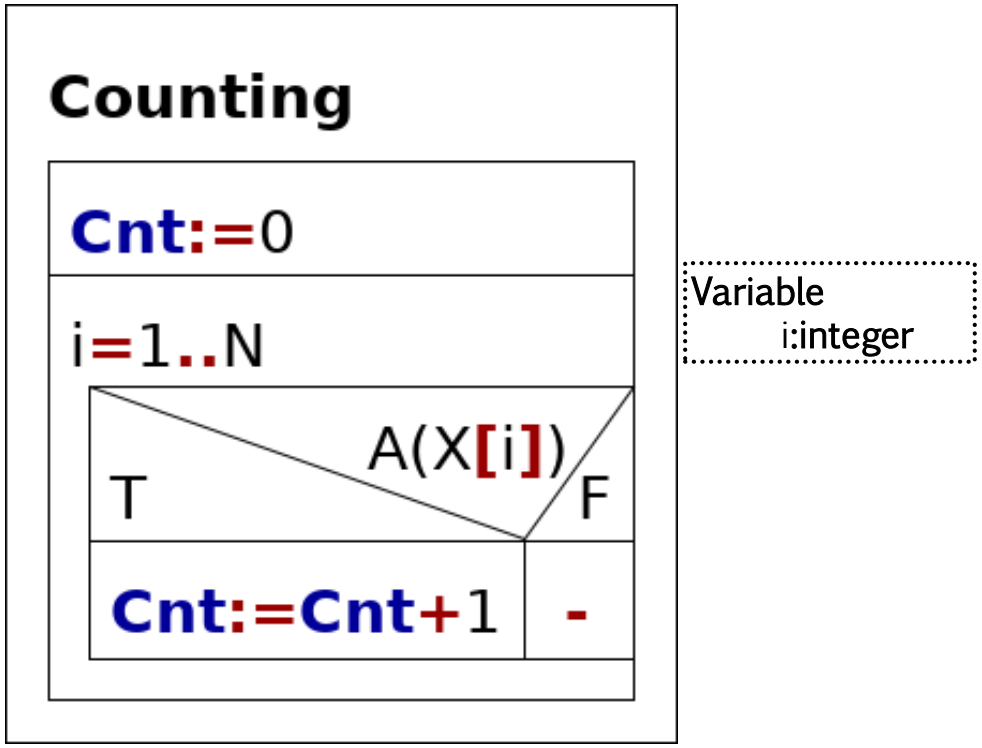
Precondition: –

Postcondition: $Cnt = \sum_{\substack{i=1 \\ A(X[i])}}^{N} 1$

Note: **A** is an attribute that can be given as a Boolean function. One can tell about each element of X (and $\mathbb{S}$) whether they have this attribute or not.

2. Counting

Algorithm



2. Counting

Specification (people having birthday in winter):

Input: $N \in \mathbb{N}$

$Mon_{1..N} \in \mathbb{N}^N$

$Winter? : \mathbb{N} \to \mathbb{L}$

$Winter?(x) := (x<3 \text{ or } x=12)$

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

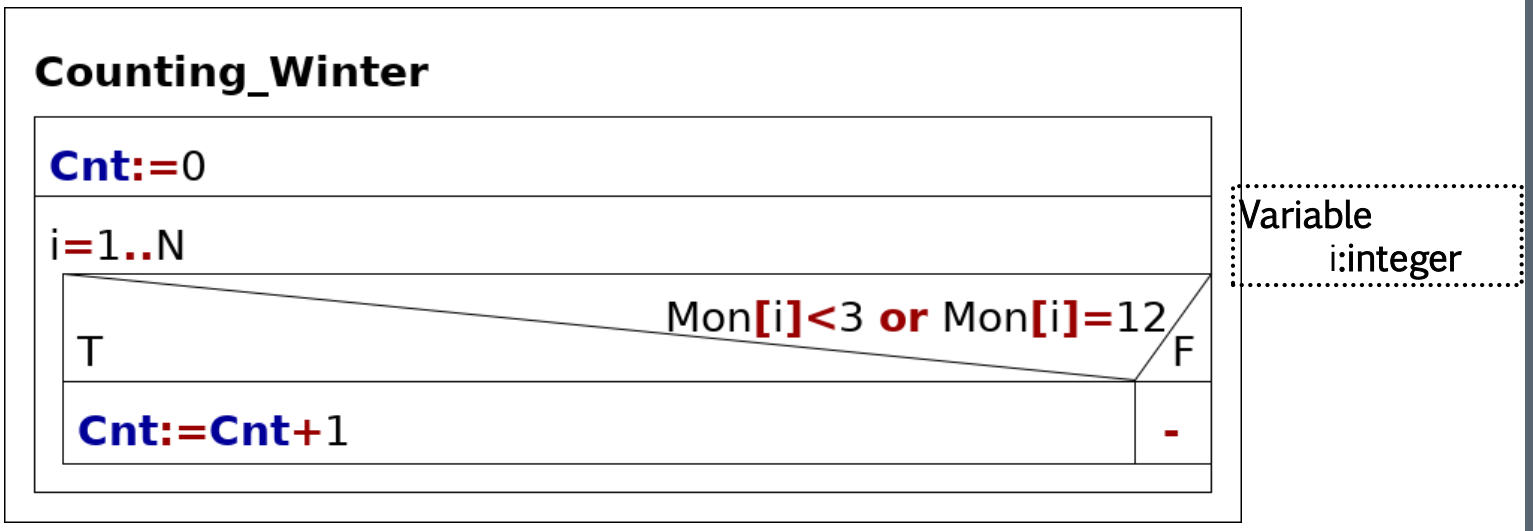
Precondition: $\forall i\,(1 \le i \le N) : Mon_i \in [1..12]$

Postcondition: $$Cnt = \sum_{i=1 \atop Winter?(Mon_i)}^{N} 1$$

Note: Postcondition can be more strict than that of the used pattern.

2. Counting

Algorithm



What would happen if the precondition was not met?

3. Maximum selection

Tasks (examples):

- We know the monthly income and expenses of a person. Let's determine the month where his assets grow the most!
- Let's pick the name of the person who is the last in the alphabetical order!
- Let's find the person, who likes the most types of food!
- Based on the annual daily statistics, let's define the warmest day during the year!
- We have birth dates from N people, let's find who has birthday in the year first!

3. Maximum selection

What is common?

We have to pick/find the greatest (or least) something from N elements.

Important:

- The somethings have an attribute based on which we can sort them (**sorting relation**).
- If we have at least 1 element, then we know that there exists a maximum (or minimum).

3. Maximum selection

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

Output: $Max \in \mathbb{N}$, $MaxVal \in \mathbb{S}$

Precondition: $N > 0$

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

$\forall i (1 \leq i \leq N): X_{Max} \geq X_i$ and $MaxVal = X_{max}$

another way: $(Max, MaxVal) = \underset{i=1}{\overset{N}{Max}} X_i$

Out aim is to express it with an an operator that is similarly concise to ∑

3. Maximum selection

Comments:

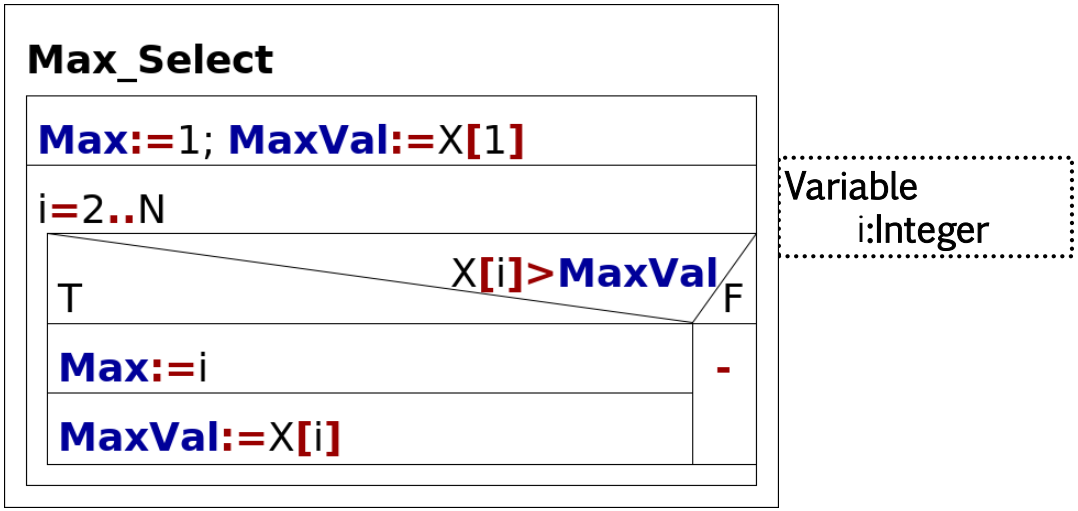
- We assume the following sort operator exists:

Someting×Something → Boolean;

- The sequence number of an element is a more general information, thus we provide that instead of the element itself.

3. Maximum selection

Algorithm:



Note: If there are more elements equal to the greatest then this algorithm will find the first one.

Questions:

› How can we find the last greatest one?

› How can we find the (first) least element?

3. Maximum selection (returning maximum **index**)

Specification (only modifications)

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

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

Precondition: $N > 0$

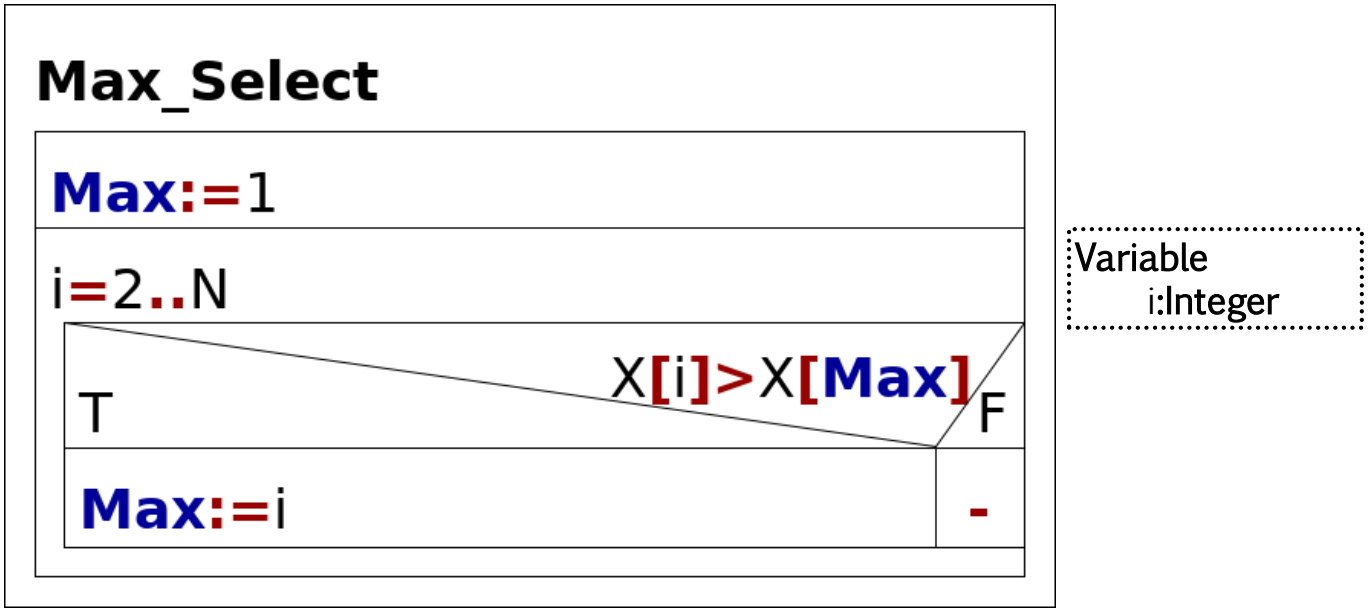
Postcondition: $1 \le Max \le N$ and

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

another way: $(Max) = MaxInd_{i=1}^{N} X_i$

3. Maximum selection (returning maximum **index**)

Algorithm



3. Maximum selection (returning maximum **value**)

Specification (only modifications)

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

Output: $MaxVal \mathbb{S}$

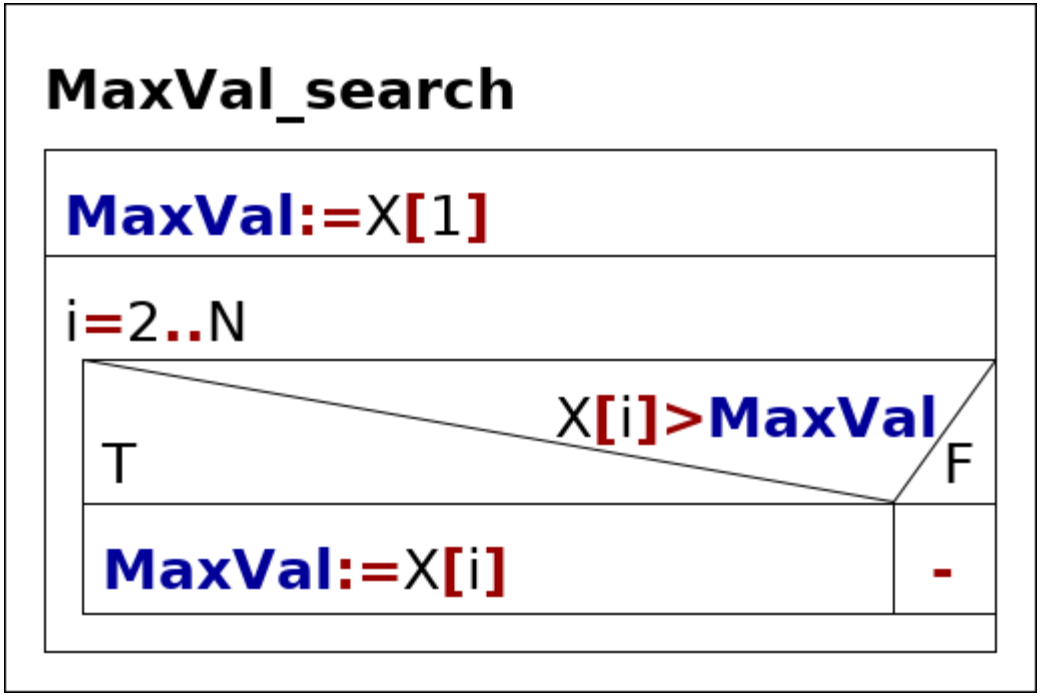
Precondition: $N > 0$

Postcondition: $MaxVal \in X$ and
$\exists i (1 \le i \le N) : MaxVal = X_i$ and
$\forall i (1 \le i \le N) : MaxVal \ge X_i$

another way: $MaxVal = \underset{i=1}{\overset{N}{MaxV}} X_i$

3. Maximum selection (returning maximum **value**)

Algorithm



Variable
i:Integer

3. Maximum selection – first birthday

Specification

Input: $N \in \mathbb{N}$,

$D_{1..N} \in (month \times day)^N$, $month, day = \mathbb{N}$

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

Precondition: $N > 0$ and

$\forall i (1 \le i \le N) : D_i.month \in [1..12]$ and

$D_i.day \in [1..31]$

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

$\forall i (1 \le i \le N)$: $D_{First}.month < D_i.month$ **or**

$(D_{First}.month = D_i.month$ **and** $D_{First}.day \le D_i.day)$

3. Maximum selection – first birthday

Specification (another way)

…

Postcondition: $First = \underset{i=1}{\overset{N}{MaxInd}}(D_i)$

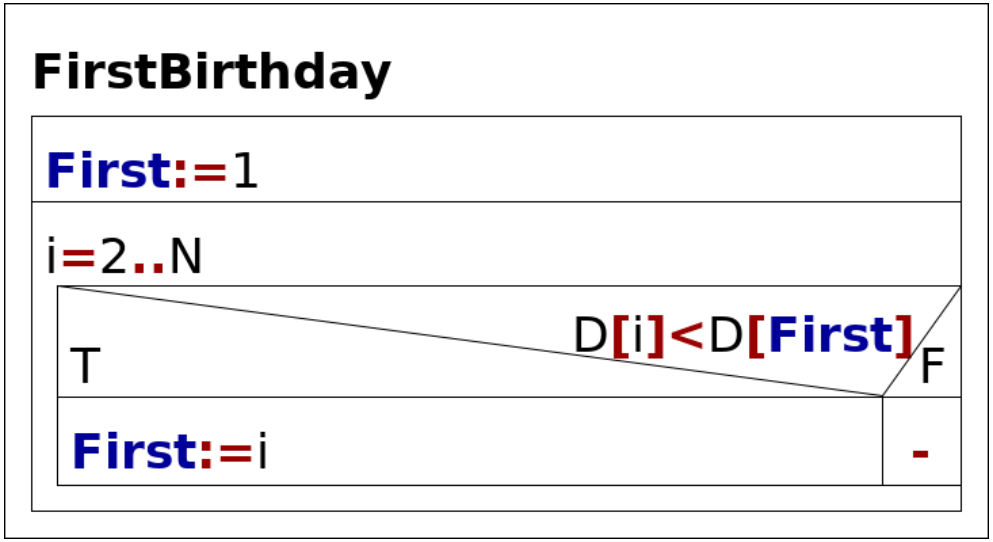
Definition:

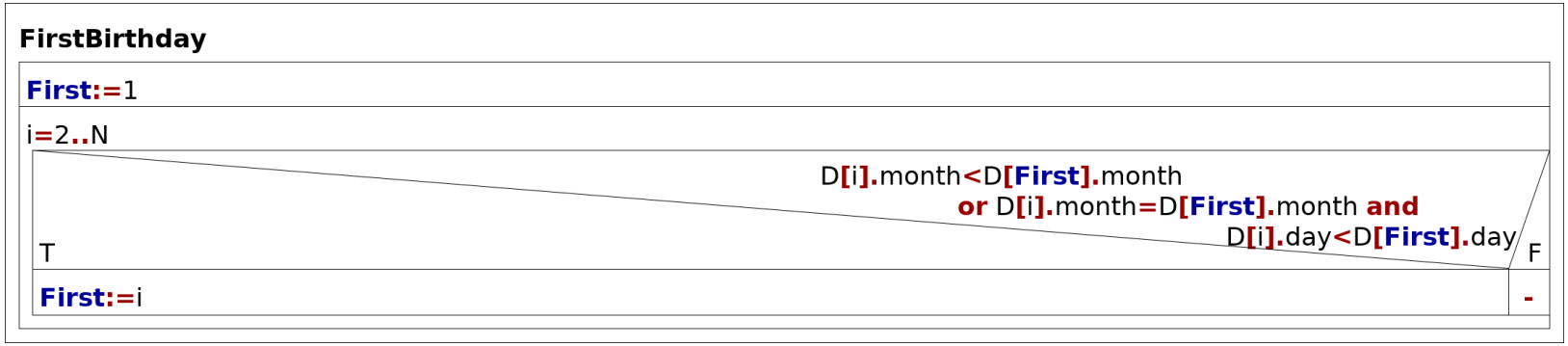
$D_i \leq D_j \leftrightarrow D_i.month < D_j.month$ or

$(D_i.month = D_j.month$ and $D_i.day \leq D_j.day)$

3. Maximum selection - first birthday

Algorithm





4. Search

Tasks (examples):

- We know the monthly incomes and expenses of a person. His assets grow by the end of the year. Let's give a month, when his assets didn't grow!
- Let's define a non-1 and non-N divisor of the integer N!
- Let's search for letter "a" in the name of a person!
- Let's search for a course in which the student failed!
- Let's find an element in a sequence which is greater than the previous element!


4. Search

What is common?

We have N "something", and we have to search for an element which has a given attribute, and we do not know whether such an element exists among N.

4. Search

Specification

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

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

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 $Val = X_{ind}$

Another way: $(\text{Exists}, \text{Ind}, \text{Val}) = \underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{Search}}\, i$

Note: this specification comes from 2 parts: „Is there any?" (decision) and from „giving the value or the index" (selection)

4. Search

Algorithm$_1$:

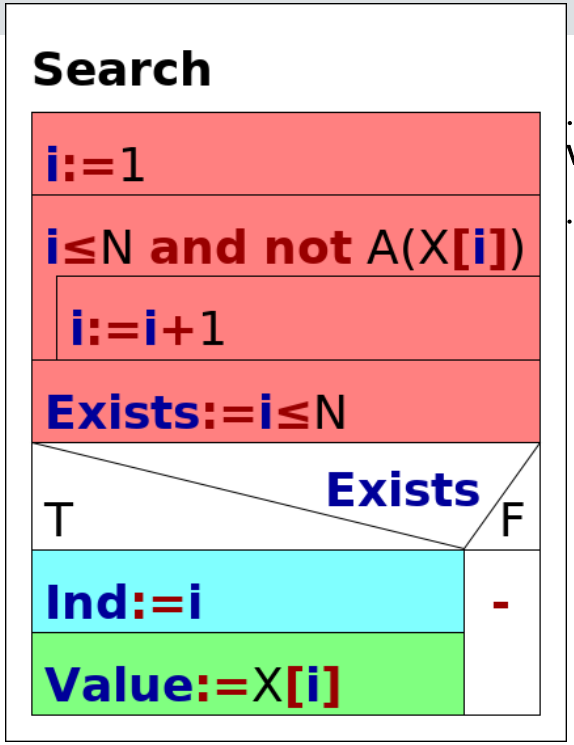
Specification

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

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

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 $Val = X_{ind}$



Note:

We know that the solution provides the first element that has the given attribute.

4. Search

Algorithm$_2$:

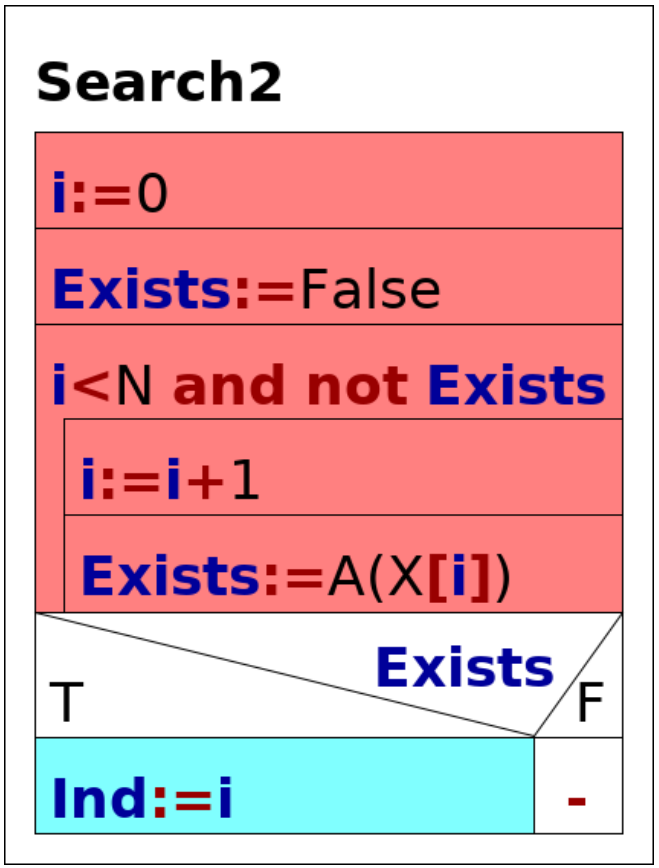
Specification

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

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

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 $Val = X_{ind}$



Let's search for a course in which the student failed!

4. Search

Specification

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

Output: $Fals \in \mathbb{L}$, $CI \in \mathbb{N}$

Precondition: $\forall i (1 \le i \le N) : Mark_{1..N} \in [1..5]$

Postcondition: $Fals = (\exists i (1 \le i \le N) : Mark_i = 1$ and

$Fals \to 1 \le CI \le N$ and $Mark_{CI} = 1$

or: $(\text{Fails}, \text{CI}) = \underset{\substack{i=1 \\ Mark_i = 1}}{\overset{N}{search}} i$

The A Attributes function

4. Search

Algorithm

Specification

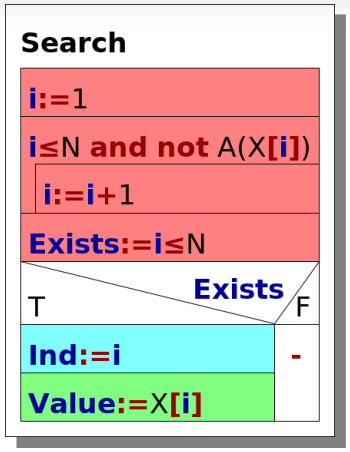
Input: $N \in \mathbb{N}$, $Mark_{1..N} \in \mathbb{N}^N$, $A: \mathbb{S} \to \mathbb{L}$

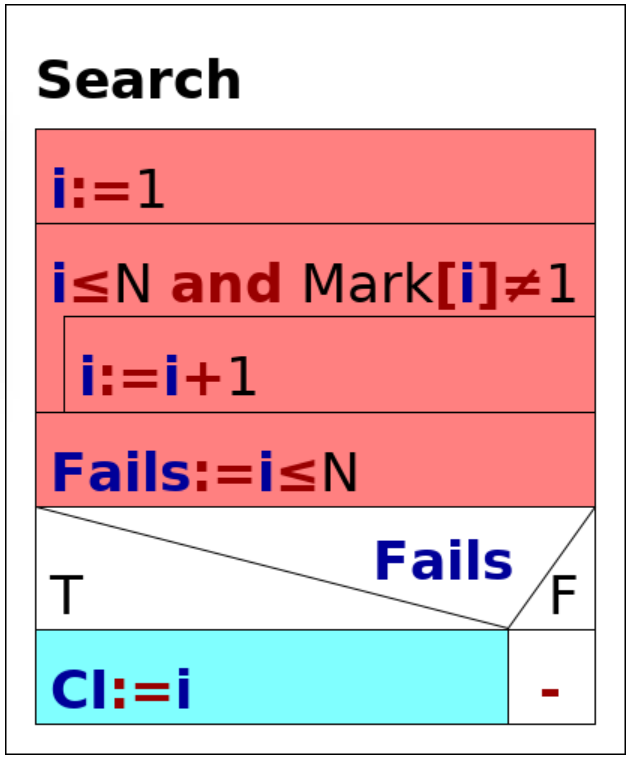
Output: $Fals \in \mathbb{L}$, $CI \in \mathbb{N}$

Precondition: $\forall i\,(1 \le i \le N)$: $Mark_{1..N} \in [1..5]$

Postcondition: $Fals = (\exists i\,(1 \le i \le N) : Mark_i = 1$ and

$Fals \to 1 \le CI \le N$ and $Mark_{CI} = 1$





variable
i:integer

5. Decision

Tasks (examples):

- Let's **decide** if an integer is prime or not!
- Let's **tell** if a given word is the name of a month!
- Based on the final marks of a student, **let's determine** if he/she fails the semester!
- **Let's find** if a given word contains a vowel!
- **Let's decide** if a sequence is monotonically increasing!
- Based on the final marks, **let's determine** if the student is excellent (all marks are the best).

5. Decision

What is common?

Let's determine if there is an item with given attribute among N "something„

A „narrowed” (output) version of search

4. Decision

Specification

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

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

Precondition: –

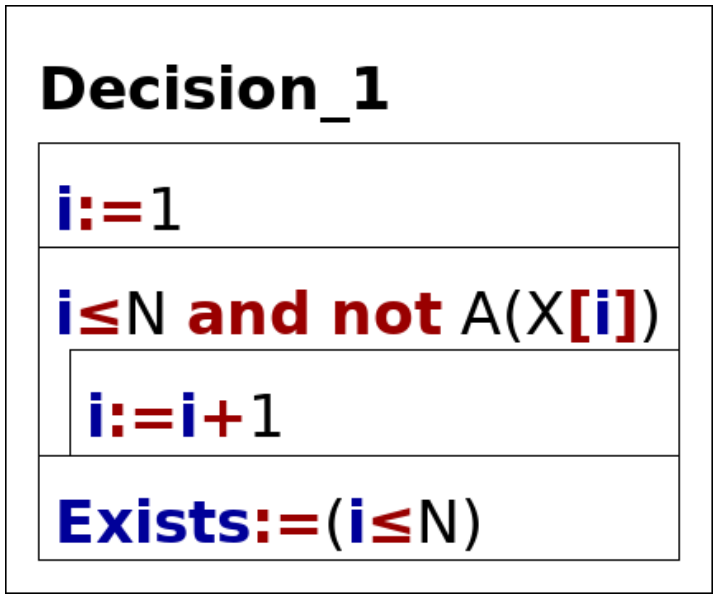
Postcondition: $Exists = (\exists i\ (1 \le i \le N) : A(X_i))$

another way: $$Exists = \exists_{i=1}^{N} A(X_i)$$

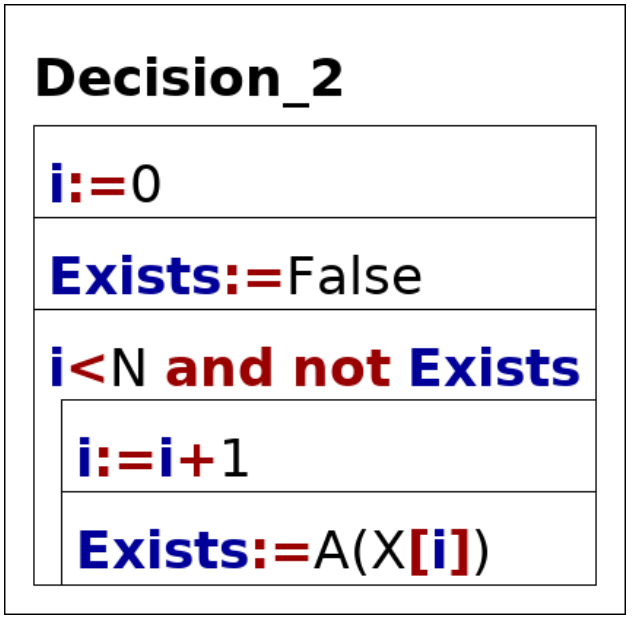
Comment: Attribute A can be defined as a boolean function. All elements can be tested against attribute A.

4. Decision

Algorithm1



Algorithm2



4. Decision

Task Variant:

… all the items are of attribute A …

Specification (only the difference):

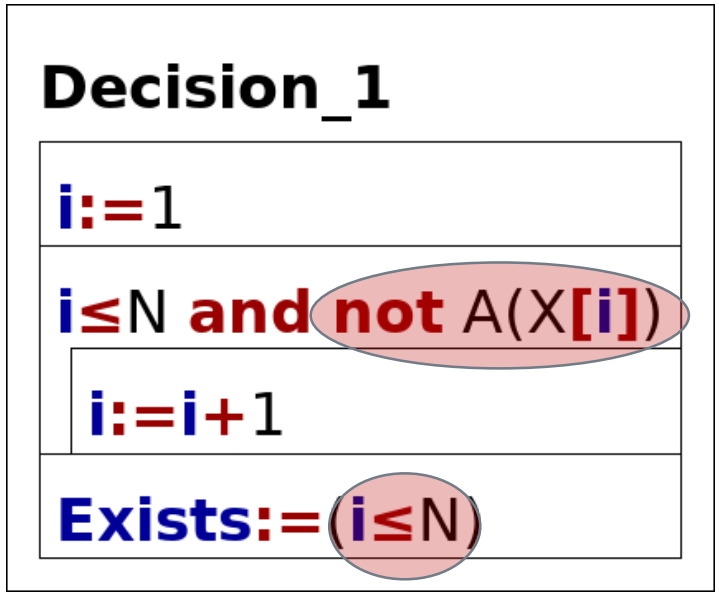
Output: $\mathtt{All} \in \mathbb{L}$

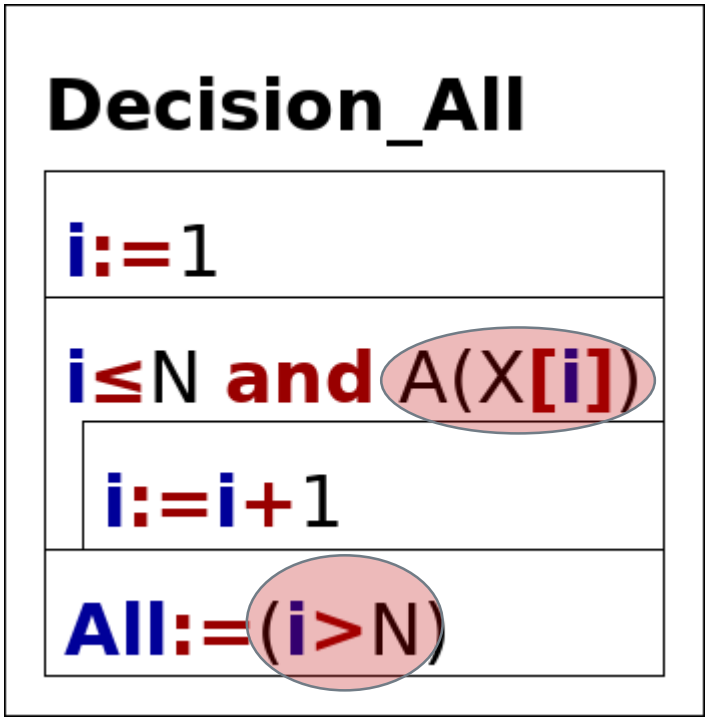
Post condition: $\mathtt{All} = (\forall \mathtt{i}\,(1 \le \mathtt{i} \le \mathtt{N}) : \mathtt{A}(X_i))$

another way: $$\mathrm{All} = \bigwedge_{i=1}^{N} A(X_i)$$

4. Decision

Algorithm:





4. Decision

Based on the final marks of a student, let's determine if he/she fails the semester!

Specification

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

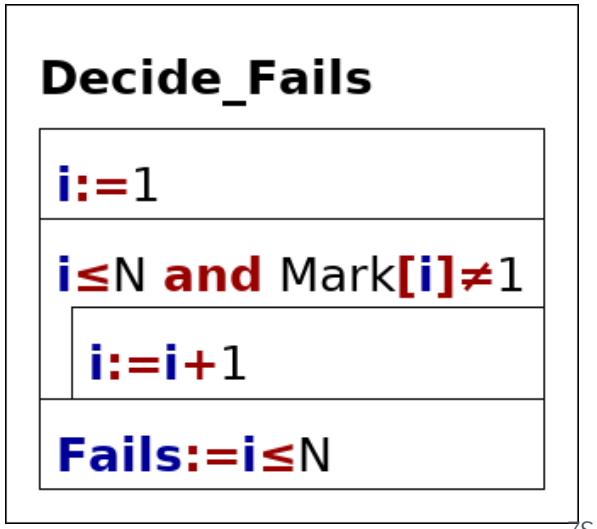
Output: $Fals \in \mathbb{L}$

Precondition: $\forall i (1 \leq i \leq N) : Mark_i \in [1..5]$

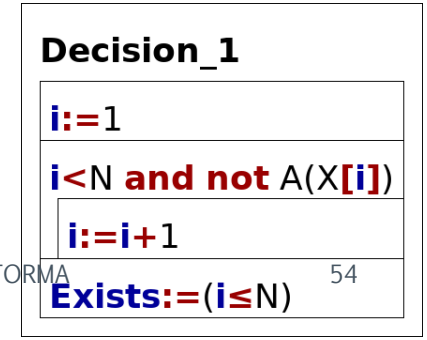
Postcondition: $Fals = (\exists i (1 \leq i \leq N) : Mark_i = 1)$

The A Attributes function

Algorithm:



variable
i:integer



5. Selection

Tasks:

- We know the monthly income and expenses of a person. His assets grow by the end of the year. Let's give a month, when his asset grows!
- Let's define the least divisor of a non-1 integer!
- Let's find a vowel in an English word!
- Let's define the serial number of a month given by its name!


5. Selection

What is common?

We have N "something", and we have to select **an** element which has a given attribute, so that we know that at least one such element exists among N.

This is the version of search when we do not have to prepare for the case when the element cannot be found.


5. Selection

Specification

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

Output: $Ind \in \mathbb{N}$, $Val \in \mathbb{S}$

Precondition: $N>0$ and $\exists i (1 \le i \le N) : A(X_i)$

Postcondition: $(1 \le Ind \le N)$ and $A(X_{Ind})$

and $Val = X_{Ind}$

Another way: $(\text{Ind}, \text{Val}) = \underset{\substack{i=1 \\ A(X_i)}}{\overset{N}{Select}}\, i$

5. Selection

Algorithm

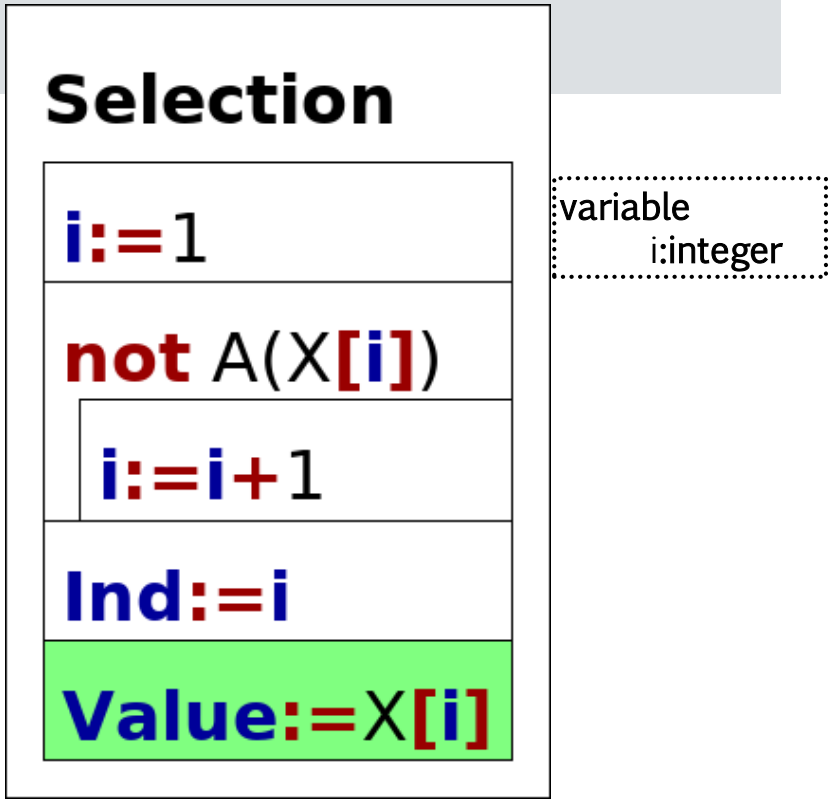
Specification

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

Output: $Ind \in \mathbb{N}$, $Val \in \mathbb{S}$

Precondition: $N>0$ and $\exists i\,(1 \le i \le N) : A(X_i)$

Postcondition: $(1 \le Ind \le N)$ and $A(X_{Ind})$ and $Val = X_{Ind}$



Comment:

We know more: this solution gives the very first element that is of attribute A – so the program does more than expected. How could we find the last one?

5. Selection

Let's find a vowel in an English word!

Specification:

Input: $Word \in \mathbb{T}$

$\mathbb{T}$ = set of texts (string)

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

Precondition: $length(Word) > 0$ and $\exists (1 \leq i \leq length(Word)) : isVowel(Word_i)$

Postcondition: $1 \leq VW \leq length(Word)$ and $isVowel(Word_{VW})$

The A Attributes function

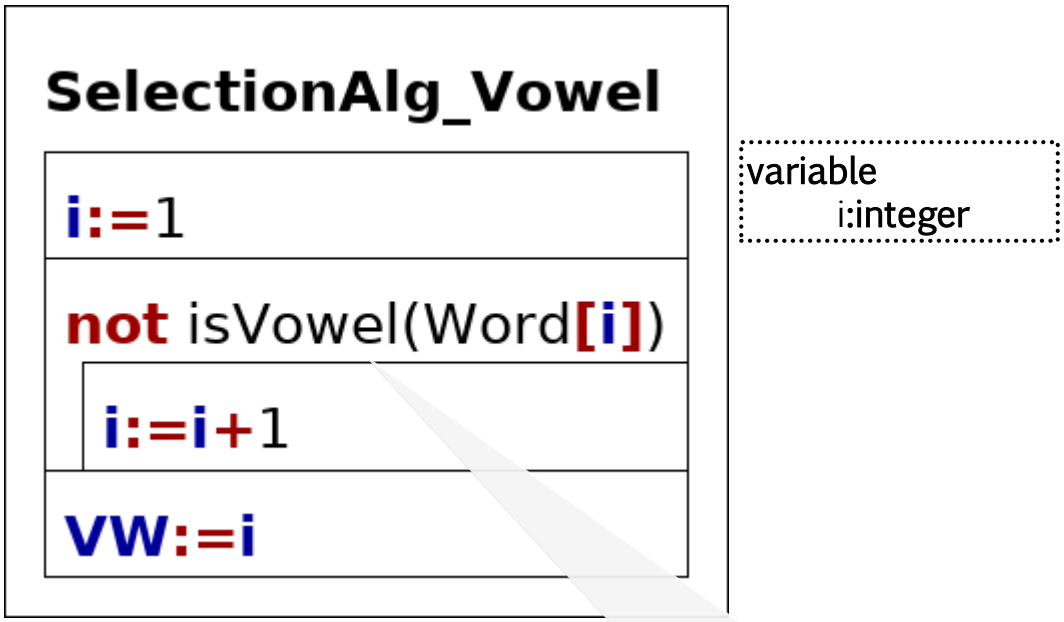
Definition: $isVowel: \mathbb{C}h \rightarrow \mathbb{L}(Character \rightarrow Boolean)$

$isVowel(ch) := capital(ch) \in \{'A', \ldots, 'U'\}$

Let's find a vowel in an English word!

5. Selection

Algorithm:



Write a function – which PoA?