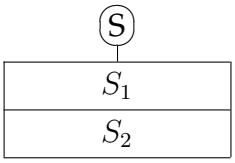
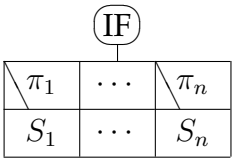
Algorithmic patterns

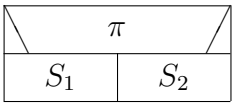
1 Structogram of program constructs

1.1 Sequence

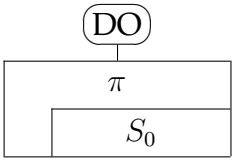


1.2 Branches





1.3 Loop



2 Algorithmic patterns

- Algorithmic patterns are problem-program pairs where the program solves the problem. They are frequently used as patterns to plan algorithms when the task to be solved is similar to the problem of the pattern.
- One of the common properties of algorithmic patterns is that they process a sequence of elementary values produced by an appropriate function. By expressing an algorithmic pattern in this way makes it more general instead of processing the elements of an array: each array can be interpreted as a function over integer interval.
- In the following, some algorithmic patterns will be given.

2.1 Summation

Problem

Let $\mathcal{H}$ be an arbitrary set where the operation of addition (+) is defined. Suppose that there exists a neutral element of the addition in H. Let the function $f\colon \mathbb{Z} \to \mathcal{H}$ be given. Let us calculate the sum of the values of f over the interval $[m..n]$.

Specification of the problem

$A = (m{:}\mathbb{Z}, n{:}\mathbb{Z}, s{:}\mathcal{H})$

$B = (m'{:}\mathbb{Z}, n'{:}\mathbb{Z})$

$Pre = (m = m' \wedge n = n')$

$Post = (Pre \wedge s = \sum_{i=m}^{n} f(i))$

Program

$i, s := m, 0$	
$i \leq n$	
	$s := s + f(i)$
	$i := i + 1$

2.2 Counting

Problem

Let β be a logical function defined over integers. Let us count the number of elements in the interval $[m..n]$ for which β holds.

Specification of the problem

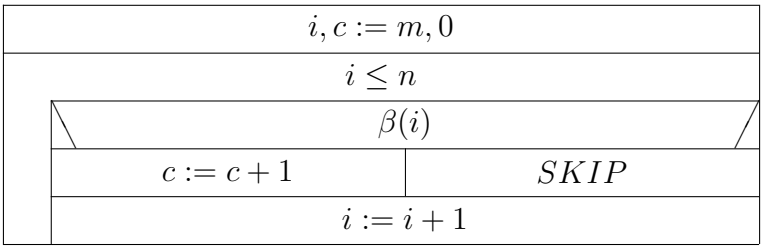
$A = (m{:}\mathbb{Z}, n{:}\mathbb{Z}, c{:}\mathbb{N})$

$B = (m'{:}\mathbb{Z}, n'{:}\mathbb{Z})$

$Pre = (m = m' \wedge n = n')$

$Post = (Pre \wedge c = \sum_{i=m}^{n} \chi(\beta(i)))$ where $\chi \colon \mathbb{L} \to \{0, 1\}$ and $\chi(true) = 1$ and $\chi(false) = 0$

Program



2.3 Maximum search

Problem

Consider a non-empty integer interval and a function $f \colon \mathbb{Z} \to \mathcal{H}$ where $\mathcal{H}$ is a totally ordered set. Let us seek the greatest value of the function f and an argument where the function takes its maximum value.

Specification of the problem

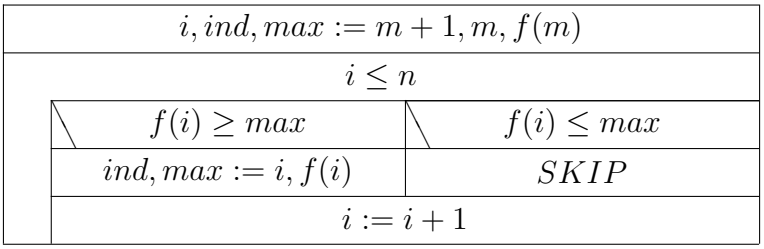
$A = (m{:}\mathbb{Z}, n{:}\mathbb{Z}, max{:}\mathcal{H}, ind{:}\mathbb{Z})$

$B = (m{:}\mathbb{Z}, n{:}\mathbb{Z})$

$Pre = (m = m' \wedge n = n' \wedge m \leq n)$

$Post = (Pre \wedge ind \in [m..n] \wedge max = f(ind) \wedge \forall j \in [m..n] : f(j) \leq max)$

Program



2.4 Conditional maximum search

Problem

Let $f \colon \mathbb{Z} \to \mathcal{H}$ and $\beta \colon \mathbb{Z} \to \mathbb{L}$ be functions defined over integers where $\mathcal{H}$ is a totally ordered set. Let us find the maximum value of the function f over the set $[m..n] \bigcap \lceil \beta \rceil$, and if exists, an argument in $[m..n] \bigcap \lceil \beta \rceil$ where the function takes its maximum value.

Specification of the problem

$A = (m{:}\mathbb{Z}, n{:}\mathbb{Z}, max{:}\mathcal{H}, ind{:}\mathbb{Z}, l{:}\mathbb{L})$

$B = (m'{:}\mathbb{Z}, n'{:}\mathbb{Z})$

$Pre = (m = m' \wedge n = n')$

$Post = (Pre \wedge l = (\exists k \in [m..n] : \beta(k)) \wedge (l \rightarrow (ind \in [m..n] \wedge max = f(ind) \wedge \beta(ind) \wedge \forall j \in [m..n] : \beta(j) \rightarrow f(j) \leq max)))$

Program

<table>
<tr><td colspan="4">$i, l := m, false$</td></tr>
<tr><td colspan="4">$i \leq n$</td></tr>
<tr><td>$\neg\beta(i)$</td><td>$\beta(i) \wedge l$</td><td colspan="2">$\beta(i) \wedge \neg l$</td></tr>
<tr><td rowspan="2">$SKIP$</td><td rowspan="2">$ind, max, l := i, f(i), true$</td><td>$f(i) \geq max$</td><td>$f(i) \leq max$</td></tr>
<tr><td>$ind, max := i, f(i)$</td><td>$SKIP$</td></tr>
<tr><td colspan="4">$i := i + 1$</td></tr>
</table>

2.5 Linear search

Problem

Let β be a logical function defined over integers. Let us find the first element (in case it exists) of interval $[m..n]$ for which β holds.

Specification of the problem

$A = (m{:}\mathbb{Z}, n{:}\mathbb{Z}, i{:}\mathbb{Z}, l{:}\mathbb{L})$

$B = (m'{:}\mathbb{Z}, n'{:}\mathbb{Z})$

$Pre = (m = m' \wedge n = n' \wedge m \leqslant n + 1)$

$Post = (Pre \wedge l = (\exists j \in [m..n]{:}\ \beta(j)) \wedge l \rightarrow (i \in [m..n] \wedge \beta(i) \wedge \forall j \in [m..i-1]{:}\ \neg\beta(j)))$

Program

<table>
<tr><td colspan="2">$i, l := m, false$</td></tr>
<tr><td colspan="2">$\neg l \wedge i \leq n$</td></tr>
<tr><td></td><td>$l := \beta(i)$</td></tr>
<tr><td></td><td>$i := i + 1$</td></tr>
</table>