# Programming Languages: Imperative Program Construction
# Practicals 7: Loop Constuction III

Shin-Cheng Mu

Autumn Term, 2024

1. Solve:

$$\begin{array}{l}\textbf{con } A, B : \mathit{Int}\{A \geqslant 0 \ \wedge B \geqslant 0\};\\ \textbf{var } r : \mathit{Int};\\ S\\ \{r = A \times B\} \ \ ,\end{array}$$

   using only (/2) (integral division by two), (×2), *even*, *odd*, addition, and subtraction.

**Solution:** Use the invariant

$$r + a \times b = A \times B \ \wedge \ 0 \leqslant a \ \wedge \ 0 \leqslant b,$$

with initialisation $r, a, b := 0, A, B$. If $b$ is even:

$$\begin{array}{ll} & r + a \times b\\ = & r + a \times 2 \times (b/2)\\ = & r + (a \times 2) \times (b/2).\end{array}$$

If $b$ is odd:

$$\begin{array}{ll} & r + a \times b\\ = & r + a \times (1 + (b - 1))\\ = & (r + a) + a \times (b - 1).\end{array}$$

The program:

$$\begin{array}{l}\textbf{con } A, B : \mathit{Int}\{A \geqslant 0 \ \wedge B \geqslant 0\}\\ \textbf{var } r, a, b : \mathit{Int}\\ \\ r, a, b := 0, A, B\\ \{r + a \times b = A \times B \ \wedge \ 0 \leqslant a \ \wedge \ 0 \leqslant b, \ \ \mathit{bnd} : b\}\\ \textbf{do } b > 0 \wedge \mathit{even}\ b \rightarrow a, \ b \ := \ a \times 2, \ b/2\\ \ \ \mid \ b > 0 \wedge \mathit{odd}\ b \rightarrow r, \ b \ := \ r + a, \ b - 1\\ \textbf{od}\\ \{r = A \times B\} \ \ .\end{array}$$

2. The sum of all digits of a natural number can be computed by

$$
\begin{aligned}
&sd\ 0 = 0 \\
&sd\ x = x \mathbin{\%} 10 + sd\ (x \mathbin{/} 10) \quad , for|x > 0|,
\end{aligned}
$$

where $(/)$ is integral division and $a \mathbin{\%} b$ computes the remainder of $a \mathbin{/} b$. Solve

$$
\begin{aligned}
&con\ N : Int\ \{0 \leqslant N\} \\
&var\ r : Int \\
&? \\
&\{r = sd\ N\}
\end{aligned}
$$

**Solution:** Introduce an auxiliary variable $n$ and use the invariant

$$r + sd\ n = sd\ N \wedge 0 \leqslant n \ .$$

For $n > 0$ we have

$$
\begin{aligned}
&\quad r + sd\ n \\
&= r + x \mathbin{\%} 10 + sd\ (x \mathbin{/} 10) \ .
\end{aligned}
$$

The program is:

$$
\begin{aligned}
&con\ N : Int\ \{0 \leqslant N\} \\
&var\ r, n : Int \\
&r, n := 0, N \\
&\{r + sd\ n = sd\ N \wedge 0 \leqslant n, bnd : n\} \\
&\mathbf{do}\ n \neq 0 \rightarrow \quad r, n := r + n \mathbin{\%} 10, n \mathbin{/} 10 \quad \mathbf{od} \\
&\{r = sd\ N\}
\end{aligned}
$$

3. Given integral number $N$, derive a program that computes the number of factors 3 of $N$. For example, when $N = 945 = 3^3 \times 5 \times 7$, the program should store the value 3 in variable $r$. You are allowed to use integral division and (%) (the operator for remainders).

$$
\begin{aligned}
&con\ N : Int\ \{N\ldots\} \quad \text{-- what should the constraint on } N \text{ be to make the problem easier?} \\
&var\ r : Int \\
&? \\
&\{\ldots r = \ldots\text{how do you write the post condition?}\}
\end{aligned}
$$

**Solution:** Introduce an auxiliary variable $n$. Let the postcondition be

$$3^r \times n = N \wedge n \mathbin{\%} 3 \neq 0 \ .$$

Use the invariant

$$3^r \times n = N \wedge 0 < n \ ,$$

and let $n \mathbin{\%} 3 = 0$ be the guard of the loop.

When $n \mathbin{\%} 3 = 0$ we have

$$
\begin{aligned}
& 3^r \times n \\
= \quad & \{ \text{ division and remainder: } 3 \times n \,/\, 3 + n \,\%\, 3 \} \\
& 3^r \times (3 \times n \,/\, 3 + n \,\%\, 3) \\
= \quad & \{ n \,\%\, 3 = 0 \} \\
& 3^r \times 3 \times n \,/\, 3 \\
= \; & 3^{r+1} \times n \,/\, 3 \; .
\end{aligned}
$$

The program is shown below. Note that we need $0 < n$ to prove that the bound decreases. Therefore we want $N > 0$.

$$
\begin{aligned}
& \mathit{con}\ N : \mathit{Int}\ \{N > 0\} \\
& \mathit{var}\ r, n : \mathit{Int} \\
& r, n := 0, N \\
& \{3^r \times n = N \wedge 0 < n, \mathit{bnd} : n\} \\
& \mathbf{do}\ n \,\%\, 3 = 0 \rightarrow \quad r, n := r + 1, n \,/\, 3 \quad \mathbf{od} \\
& \{3^r \times n = N \wedge n \,\%\, 3 \neq 0\}
\end{aligned}
$$

4. Solve:

$$
\begin{aligned}
& \mathbf{con}\ N, X : \mathit{Int}\ \{0 \leqslant N\} \\
& \mathbf{con}\ f : \mathbf{array}\ [0..N)\ \mathbf{of}\ \mathit{Int} \\
& \mathbf{var}\ r : \mathit{Int} \\
& \ ? \\
& \{r = \langle \Sigma i : 0 \leqslant i < N : f[i] \times X^i \rangle\}
\end{aligned}
$$

We have seen this problem before but let us do it slightly differently this time. (This problem is not that much about associativity, but a practice constructing and using recursive function definition.)

(a) Define $g\ n = \langle \Sigma i : n \leqslant i < N : f[i] \times X^{i-n} \rangle$ for $0 \leqslant n \leqslant N$, derive a recursive definition of $g$.

**Solution:** For an easy base case, $g\ N = \langle \Sigma i : N \leqslant i < N : f[i] \times X^{i-n} \rangle = 0$. For $0 \leqslant n < N$ we calculate:

$$
\begin{aligned}
& g\ n \\
= \; & \langle \Sigma i : n \leqslant i < N : f[i] \times X^{i-n} \rangle \\
= \quad & \{ \text{ since } 0 \leqslant n < N, \text{ split off } i = n \} \\
& f[n] \times X^{n-n} + \langle \Sigma i : n + 1 \leqslant i < N : f[i] \times X^{i-n} \rangle \\
= \quad & \{ \text{ with } n + 1 \leqslant i < N, \text{ arithmetics } \} \\
& f[n] + \langle \Sigma i : n + 1 \leqslant i < N : f[i] \times X^{i-(n+1)} \times X \rangle \\
= \quad & \{ \text{ distributivity } \} \\
& f[n] + X \times \langle \Sigma i : n + 1 \leqslant i < N : f[i] \times X^{i-(n+1)} \rangle \\
= \; & f[n] + X \times g\ (n + 1) \; .
\end{aligned}
$$

Therefore we conclude:

$$
\begin{aligned}
& g\ N = 0 \\
& g\ n \ = f[n] + X \times g\ (n + 1), \ \text{ if } 0 \leqslant n < N.
\end{aligned}
$$

(b) Use $r = g\ n$ as the main invariant, construct a program that solves the problem.

**Solution:** Introduce a new variable $n$ and use $r = g\ n \wedge 0 \leqslant n \leqslant N$ as the main invariant, and use $n \neq 0$ as the loop guard since $r = g\ 0$ is the postcondition we want, which can be satisfied by initialising $r, n := 0, N$.

We decrease the bound by $n := n - 1$. To find out how to update $r$ we calculate:

$$
\begin{array}{ll}
 & (g\ n)[n \backslash n-1] \\
= & f[n-1] + X \times g\ n \\
= & \quad \{\ r = g\ n \wedge 0 \leqslant n \leqslant N \wedge n \neq 0\ \} \\
 & f[n-1] + X \times r\ \ .
\end{array}
$$

The resulting program (supplementary proofs omitted for now):

$$
\begin{array}{l}
\textbf{con}\ N, X : \mathit{Int}\ \{0 \leqslant N\} \\
\textbf{con}\ f : \textbf{array}\ [0..N)\ \textbf{of}\ \mathit{Int} \\
\textbf{var}\ r, n : \mathit{Int} \\
r, n := 0, n \\
\{r = g\ n \wedge 0 \leqslant n \leqslant N, \mathit{bnd} : n\} \\
\textbf{do}\ n \neq 0 \rightarrow \\
\quad r := f[n-1] + X \times r \\
\quad n := n - 1 \\
\textbf{od} \\
\{r = \langle \Sigma i : 0 \leqslant i < N : f[i] \times X^i \rangle\}
\end{array}
$$

5. The function *fusc* is defined on natural numbers by:

$$
\begin{array}{lcl}
\mathit{fusc}\ 0 & = & 0 \\
\mathit{fusc}\ 1 & = & 1 \\
\mathit{fusc}\ (2 \times n) & = & \mathit{fusc}\ n \\
\mathit{fusc}\ (2 \times n + 1) & = & \mathit{fusc}\ n + \mathit{fusc}\ (n+1).
\end{array}
$$

Derive a program computing $\mathit{fusc}\ N$ for $N \geqslant 0$. Hint: try $\mathit{fusc}\ 78$.

**Solution:** Use the invariant

$$a \times \mathit{fusc}\ n + b \times \mathit{fusc}\ (n+1) = \mathit{fusc}\ N\ \wedge\ 0 \leqslant n \leqslant N,$$

which can be established by $a, b, n := 1, 0, N$. When $n$ is even (let $n = 2 \times m$):

$$
\begin{array}{ll}
 & a \times \mathit{fusc}\ n + b \times \mathit{fusc}\ (n+1) \\
= & a \times \mathit{fusc}\ (2 \times m) + b \times \mathit{fusc}\ (2 \times m + 1) \\
= & a \times \mathit{fusc}\ m + b \times \mathit{fusc}\ m + b \times \mathit{fusc}\ (m+1) \\
= & (a+b) \times \mathit{fusc}\ m + b \times \mathit{fusc}\ (m+1) \\
= & (a+b) \times \mathit{fusc}\ (n\ \textbf{div}\ 2) + b \times \mathit{fusc}\ (n\ \textbf{div}\ 2 + 1).
\end{array}
$$

When $n$ is odd (let $n = 2 \times m + 1$):

$$
\begin{array}{ll}
 & a \times \mathit{fusc}\ n + b \times \mathit{fusc}\ (n+1) \\
= & a \times \mathit{fusc}\ (2 \times m + 1) + b \times \mathit{fusc}\ (2 \times m + 2) \\
= & a \times \mathit{fusc}\ m + a \times \mathit{fusc}\ (m+1) + b \times \mathit{fusc}\ (m+1)
\end{array}
$$

$$
\begin{aligned}
&= a \times \mathit{fusc}\ m + (a+b) \times \mathit{fusc}\ (m+1) \\
&= a \times \mathit{fusc}\ (n\ \mathbf{div}\ 2) + (a+b) \times \mathit{fusc}\ (n\ \mathbf{div}\ 2 + 1).
\end{aligned}
$$

When $n = 0$, we have $b = \mathit{fusc}\ N$.

The program:

$$
\begin{array}{l}
\mathbf{con}\ N : \mathit{Int}\ \{N \geqslant 0\} \\
\mathbf{var}\ a, b, n : \mathit{Int} \\
\\
a, b, n := 1, 0, N \\
\{a \times \mathit{fusc}\ n + b \times \mathit{fusc}\ (n+1) = \mathit{fusc}\ N\ \wedge\ 0 \leqslant n \leqslant N,\ \ \mathit{bnd} : n\} \\
\mathbf{do}\ n > 0 \wedge \mathit{even}\ n \rightarrow a,\ n\ :=\ a+b,\ n\ \mathbf{div}\ 2 \\
\quad \mid\ n > 0 \wedge \mathit{odd}\ n \rightarrow b,\ n\ :=\ a+b,\ n\ \mathbf{div}\ 2 \\
\mathbf{od} \\
\{b = \mathit{fusc}\ N\}\ \ .
\end{array}
$$

6. Solve:

$$
\begin{array}{l}
\mathbf{con}\ N : \mathit{Int}\ \{0 \leqslant N\} \\
\mathbf{con}\ f : \mathbf{array}\ [0..N)\ \mathbf{of}\ \mathit{Int} \\
\mathbf{var}\ r : \mathit{Bool} \\
\ ? \\
\{r = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle\}
\end{array}
$$

(a) Define, for $0 \leqslant n \leqslant N$, $g\ n = \langle \exists i : n \leqslant i < N : f[i] = 0 \rangle$. Come up with a recursive definition of $g$.

**Solution:** We have $g\ N = \mathit{False}$. For $0 \leqslant n < N$,

$$
\begin{array}{ll}
& g\ n \\
= & \langle \exists i : n \leqslant i < N : f[i] = 0 \rangle \\
= & \quad \{\ 0 \leqslant n < N, \text{ split off } i = n\ \} \\
& f[n] = 0 \vee \langle \exists i : n+1 \leqslant i < N : f[i] = 0 \rangle \\
= & f[n] = 0 \vee g\ (n+1)\ \ .
\end{array}
$$

Therefore

$$
\begin{array}{l}
g\ N \equiv \mathit{False} \\
g\ n\ \equiv f[n] = 0 \vee g\ (n+1), \quad \text{if } 0 \leqslant n < N.
\end{array}
$$

(b) Try come up with a program that, as soon as a zero is found in the array, terminates without having to scan the entire list. What invariant would you choose?

**Solution:** Define

$$
P \equiv (r \vee g\ n) = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle\ \ .
$$

and use $P \wedge 0 \leqslant n \leqslant N$ as the invariant. It can be established by the initialisation $r, n := \mathit{False}, 0$.

To allow early termination we use $\neg\ r \wedge n \neq N$ as the loop guard, since

$$
\begin{array}{ll}
& \neg(\neg r \wedge n \neq N) \wedge P \wedge 0 \leqslant n \leqslant N \\
\equiv & \{ \text{de Morgan} \} \\
& (r \vee n = N) \wedge P \wedge 0 \leqslant n \leqslant N \\
\equiv & \{ \text{distributivity} \} \\
& (r \wedge P \wedge 0 \leqslant n \leqslant N) \vee (n = N \wedge P) \ .
\end{array}
$$

Consider the branch $n = N \wedge P$:

$$
\begin{array}{ll}
& n = N \wedge ((r \vee g\ n) = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle) \\
\equiv & (r \vee g\ N) = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle \\
\equiv & \{ g\ N = \mathit{False} \} \\
& r = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle \ .
\end{array}
$$

Consider the branch $r \wedge P \wedge 0 \leqslant n \leqslant N$:

$$
\begin{array}{ll}
& r \wedge P \wedge 0 \leqslant n \leqslant N \\
\Rightarrow & r \wedge ((r \vee g\ n) = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle) \\
\equiv & \{ \text{replace } r \text{ by } \mathit{True}\text{, and } \mathit{True} \vee g\ n = \mathit{True} \} \\
& r \wedge \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle \\
\Rightarrow & r = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle \ .
\end{array}
$$

Therefore $\neg(\neg r \wedge n \neq N) \wedge P \wedge 0 \leqslant n \leqslant N$ implies $r = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle$.
Let the bound be $N - n$ and let the last statement of the loop be $n := n + 1$. Consider:

$$
\begin{array}{ll}
& (r \vee g\ n)[n \backslash n + 1] \\
= & r \vee g\ (n + 1)
\end{array}
$$

If we apply a substitution $[r \backslash r \vee f[n] = 0]$ we get

$$
\begin{array}{ll}
& ((r \vee g\ n)[n \backslash n + 1])[r \backslash r \vee f[n] = 0] \\
= & (r \vee g\ (n + 1))[r \backslash r \vee f[n] = 0] \\
= & (r \vee f[n] = 0) \vee g\ (n + 1) \\
= & \{ \text{disjunction associative} \} \\
& r \vee (f[n] = 0 \vee g\ (n + 1)) \\
= & \{ \text{calculation above} \} \\
& r \vee g\ n \ .
\end{array}
$$

Therefore we get $(P[n \backslash n + 1])[r \backslash r \vee f[n] = 0] = P$.
In conclusion, the program can be

**con** $N : \mathit{Int}\ \{0 \leqslant N\}$
**con** $f$ : **array** $[0..N)$ **of** $\mathit{Int}$
**var** $r : \mathit{Bool}$
**var** $n : \mathit{Int}$

$r, n := \mathit{False}, 0$
$\{P \wedge 0 \leqslant n \leqslant N, \mathit{bnd} : N - n\}$
**do** $\neg r \wedge n \neq N \rightarrow$
  $r := r \vee f[n] = 0$
  $n := n + 1$
**od**
$\{r = \langle \exists i : 0 \leqslant i < N : f[i] = 0 \rangle\}$

Supplementary proofs omitted for now.