universität innsbruck

Computability Theory | 26W | LVA 703317

Lecture 3 | November 9, 2026

## All Solutions

1 The inner loop

$w := 0;$
LOOP $z$ DO
  $v := w;$
  $w{+}{+}$
OD;
$z := v$

computes the predecessor function $\mathsf{p}(z)$ and is executed $y$ times in the outer loop. Hence upon termination of the outer loop $x$ (the initial value of $z$) has been decremented $y$ times and thus $z = x \dot{-} y$. Because of the final statement $z{+}{+}$, the function $(x \dot{-} y) + 1$ is computed.

2 Let $x_1, \ldots, x_n$ be the variables in the loop program $P$. We use induction on the construction of $P$.

- If $P$ is the assignment statement $x_j := 0$ then $f_j^P(a_1, \ldots, a_n) = \mathsf{z}(\pi_1^n(a_1, \ldots, a_n))$ and $f_i^P(a_1, \ldots, a_n) = \pi_i^n(a_1, \ldots, a_n)$ for $i \neq j$.
- If $P$ is the assignment statement $x_j := x_k$ then $f_j^P(a_1, \ldots, a_n) = \pi_k^n(a_1, \ldots, a_n)$ and $f_i^P(a_1, \ldots, a_n) = \pi_i^n(a_1, \ldots, a_n)$ for $i \neq j$.
- If $P$ is the increment statement $x_j{+}{+}$ then $f_j^P(a_1, \ldots, a_n) = \mathsf{s}(\pi_j^n(a_1, \ldots, a_n))$ and $f_i^P(a_1, \ldots, a_n) = \pi_i^n(a_1, \ldots, a_n)$ for $i \neq j$.
- Suppose $P = Q; R$. According to the induction hypothesis there are suitable primitive recursive functions

$$f_1^Q(a_1, \ldots, a_n), \ldots, f_n^Q(a_1, \ldots, a_n) \quad \text{and} \quad f_1^R(a_1, \ldots, a_n), \ldots, f_n^R(a_1, \ldots, a_n)$$

  and so the functions $f_i^P(a_1, \ldots, a_n) = f_i^R(f_1^Q(a_1, \ldots, a_n), \ldots, f_n^Q(a_1, \ldots, a_n))$ for $1 \leqslant i \leqslant n$ are primitive recursive.
- It remains to consider the program $P =$ LOOP $x_j$ DO $Q$ OD. For every $x \geqslant 0$ and $1 \leqslant i \leqslant n$ we define the function $f_i^{(x)}(a_1, \ldots, a_n)$ by induction on $x$:

$$f_i^{(x)}(a_1, \ldots, a_n) = \begin{cases} a_i & \text{if } x = 0 \\ f_i^Q(f_1^{(x-1)}(a_1, \ldots, a_n), \ldots, f_n^{(x-1)}(a_1, \ldots, a_n)) & \text{if } x > 0 \end{cases}$$

  These functions are easily shown to be primitive recursive. It follows that also the functions

$$f_i^P(a_1, \ldots, a_n) = f_i^{(a_j)}(a_1, \ldots, a_n)$$

  for $a_j = x_j$ and $1 \leqslant i \leqslant n$ are primitive recursive and it is easy to see that these are the functions that compute the behaviour of the program $P$.

3 (a) The following loop program $P(x, y; z)$ computes $z = x \dot{-} y$:

```
z := x;
LOOP y DO
    v := 0;
    w := 0;
    LOOP z DO
        v := w;
        w++
    OD;
    z := v
OD
```

Assuming that $Q(x; y)$ is a loop program that computes the predecessor function, we write

```
z := x;
LOOP y DO
    Q(z; v);
    z := v                (z ← z ∸ 1)
OD
```

for the above program. In general, when calling a loop program $Q(x_1, \ldots, x_n; y)$ within another program, we assume that the local variables in $Q$ are different from the variables in the surrounding program. This can always be achieved by renaming variables.

(b) The following loop program $P(x; y)$ computes $y = \lfloor \frac{x}{2} \rfloor$:

```
y := 0;
z := 0;
LOOP x DO
    y++;
    w := z;
    z := y;
    y := w
OD
```

(c) The following loop program $P(x; y)$ computes $(\mu\, y \leqslant x)\,(x = y^2)$:

```
u := x;
u++;
v := 0;
y := u;
LOOP u DO
    P×(v, v; w);
    P=(x, w; z);
    LOOP z DO
        y := v
    OD;
    v++
OD
```

Here $P_=(x, w; z)$ is a program that tests where its two inputs are equal. If that is the case $z$ is set to 1, otherwise $z = 0$. As a consequence, the inner loop is executed only when $x = v^2$, in which case $y$ is set to $v$.

4 We first prove $\mathsf{e}_{n+1}(x) \geqslant \mathsf{e}_n(x)$ by induction on $x \geqslant 0$. If $x = 0$ then $\mathsf{e}_{n+1}(x) = 2 = \mathsf{e}_n(x)$. For the induction step we have

$$\mathsf{e}_{n+1}(x+1) \;=\; \mathsf{e}_n(\mathsf{e}_{n+1}(x)) \;\geqslant\; \mathsf{e}_n(x+1)$$

using parts (1) and (2) of the lemma on slide 29. (Part (1) yields $\mathsf{e}_{n+1}(x) \geqslant x+1$ and part (2) implies the monotonicity of $\mathsf{e}_n$.)

Next we prove $\mathsf{e}_{n+1}(x+k) \geqslant \mathsf{e}_n^{(k)}(x)$ by induction on $k \geqslant 0$. If $k = 0$ then $\mathsf{e}_{n+1}(x+k) = e_{n+1}(x) > x = \mathsf{e}_n^{(k)}(x)$ by part (1) of the lemma on slide 29. For the induction step we have

$$\mathsf{e}_{n+1}(x+k+1) \;=\; \mathsf{e}_n(\mathsf{e}_{n+1}(x+k)) \;\geqslant\; \mathsf{e}_n(e_n^{(k)}(x)) \;=\; \mathsf{e}_n^{(k+1)}(x)$$

using the induction hypothesis and part (2) of the lemma on slide 29. As a matter of fact, the stronger statement $\mathsf{e}_{n+1}(x+k) \geqslant \mathsf{e}_n^{(k+1)}(x)$ holds.

5 First we show $\mathsf{e}_2(x) < 2^{2^{x+1}}$ for all $x \geqslant 0$ by induction on $x$. For $x = 0$ we obtain $2 < 4$. For the induction step we have

$$\mathsf{e}_2(x+1) \;=\; \mathsf{e}_2(x)^2 + 2 \;\leqslant\; (2^{2^{x+1}} - 1)^2 + 2 \;=\; 2^{2^{x+2}} - 2 \cdot 2^{2^{x+1}} + 3 \;<\; 2^{2^{x+2}}$$

An easy induction proof shows that $\mathsf{fib}(2 \cdot x) \geqslant 2^x$ for all $x$. Hence

$$\mathsf{e}_2(x) \;<\; 2^{2^{x+1}} \;\leqslant\; 2^{2^{2^x}} \;\leqslant\; \mathsf{fib}(2 \cdot \mathsf{fib}(2 \cdot \mathsf{fib}(2 \cdot x)))$$

Now, if $\mathsf{fib} \in \mathsf{E}_2$ then $\mathsf{fib}(2 \cdot \mathsf{fib}(2 \cdot \mathsf{fib}(2 \cdot x))) \in \mathsf{E}_2$ and thus $\mathsf{e}_2 \in \mathsf{E}_2$. Hence $\mathsf{fib} \notin \mathsf{E}_2$. We prove $\mathsf{fib} \in \mathsf{E}_3$. An easy induction proof gives $\mathsf{fib}(x) \leqslant 2^x$ for all $x$. We have $\mathsf{fib}(x) = \pi_1(\mathsf{f}(x))$ with $\mathsf{f}(x) = \pi(\mathsf{fib}(x), \mathsf{fib}(x+1))$ defined by primitive recursion. Hence

$$\mathsf{f}(x) \;\leqslant\; \pi(2^x, 2^{x+1}) \;=\; 2^{2^x}(2 \cdot 2^{x+1}) \mathbin{\dot{-}} 1 \in \mathsf{E}_3$$

Since $\pi_1$, $\pi_2$, $\pi$ and $+$ are elementary, it follows that $\mathsf{f}$ and $\mathsf{fib}$ are elementary.