Übung 3: WHILE-Programme

Berechenbarkeit und Komplexitätstheorie

Aufgabe 1

Sei $f : \mathbb{N} \to \mathbb{N}$ eine WHILE-berechenbare, injektive, totale Funktion.

a) die Umkehrfunktion f^{-1} ist ebenfalls WHILE-berechenbar

$$f^{-1}(y) = \begin{cases} x, \text{ falls } f(x) = y \\ \\ \text{undefiniert, sonst} \end{cases}$$

wohldefiniert, da f injektiv!

Algorithmus, der f^{-1} berechnet:

$x_0 := 0$

$x_t = 1$

`while` $x_t \neq 0$ `do`

  $x_0 := x_0 + 1$

  `if` $f(x_0) =$ `y then`

    $x_t := 0$

Aufgabe 1

b) gilt das auch für LOOP-berechenbare Funktionen?

$\rightarrow$ Nein

Beispiel:

$f : \mathbb{N} \rightarrow \mathbb{N}$, $x \mapsto 2 \cdot x$ ist total

Aber:

$f^{-1} : \mathbb{N} \rightarrow \mathbb{N}$ ist nur für gerade Zahlen definiert

Aufgabe 2

a) Gilt $\langle 1,2,3\rangle \leq \langle 2,2,2\rangle$?

$$\langle x,y\rangle = y + \left(\sum_{i=1}^{x+y} i\right) = y + \frac{(x+y)(x+y+1)}{2}$$

$$\langle x_1,\ldots,x_{k-1},x_k\rangle = \langle x_1,\langle\ldots\langle x_{k-1},x_k\rangle\ldots\rangle$$

$$\begin{aligned}\langle 1,2,3\rangle &= \langle 1,\langle 2,3\rangle\rangle \\ &= \langle 1,18\rangle \\ &= 208\end{aligned}$$

$$\begin{aligned}\langle 2,2,2\rangle &= \langle 2,\langle 2,2\rangle\rangle \\ &= \langle 2,12\rangle \\ &= 117\end{aligned}$$

$$\Rightarrow \langle 1,2,3\rangle \not\leq \langle 2,2,2\rangle$$

Aufgabe 2

b) $a, b, c \in \mathbb{N}$, sodass $\langle a, b, c \rangle = 102$

$$\boxed{\begin{aligned} p_1(z) &= \frac{\lfloor \frac{\sqrt{8z+1}-1}{2} \rfloor^2 + 3\lfloor \frac{\sqrt{8z+1}-1}{2} \rfloor - 2z}{2} \\ p_2(z) &= \frac{2z - \lfloor \frac{\sqrt{8z+1}-1}{2} \rfloor^2 - \lfloor \frac{\sqrt{8z+1}-1}{2} \rfloor}{2} \end{aligned}}$$

$$p_1(102) = 2$$

$$p_2(102) = 11$$

$$\Rightarrow 102 = \langle 2, 11 \rangle$$

$$= \langle 2, 3, 1 \rangle$$

$$p_1(11) = 3$$

$$p_2(11) = 1$$

Aufgabe 3

a) Implementieren der Ackermann-Funktion (rekursiv)

$$A(0,y) = y+1$$
$$A(x+1,0) = A(x,1)$$
$$A(x+1,y+1) = A(x, A(x+1,y))$$

```
def AckRec(x,y):
    if x == 0:
        return y + 1
    elif y == 0:
        return AckRec(x-1, 1):
    else:
        return AckRec(x-1,AckRec(x, y-1)):
```

Aufgabe 3

b) Implementieren der Ackermann-Funktion (Stack)

```
def AckStack(x,y):
    stack = [x,y]
    while len(stack) > 1:
        y = stack.pop()
        x = stack.pop()
        if x == 0:
            stack.append(y + 1)
        elif y == 0:
            stack.append(x - 1)
            stack.append(1)
        else:
            stack.append(x - 1)
            stack.append(x)
            stack.append(y - 1)
    return stack.pop()
```

$$A(0, y) = y + 1$$
$$A(x + 1, 0) = A(x, 1)$$
$$A(x + 1, y + 1) = A(x, A(x + 1, y))$$

Aufgabe 3

c) Ackermann-Funktion als WHILE-Programm

```
def AckStack(x,y):
    stack = [x,y]
    while len(stack) > 1:
        y = stack.pop()
        x = stack.pop()
        if x == 0:
            stack.append(y + 1)
        elif y == 0:
            stack.append(x - 1)
            stack.append(1)
        else:
            stack.append(x - 1)
            stack.append(x)
            stack.append(y - 1)
   return stack.pop()
```

$s := \langle y, \langle x, 0\rangle\rangle$
$sz := 2$

while $sz > 1$ do
  $y := p_1(s)$ $s := p2(s)$
  $x := p_1(s)$ $s := p2(s)$
  $sz := sz - 2$
  if $x = 0$ then
    $s := \langle y+1, s\rangle$
    $sz := sz + 1$
  else if $y = 0$ then
    $s := \langle x-1, s\rangle$
    $s := \langle 1, s\rangle$
    $sz := sz + 2$
  else
    $s := \langle x-1, s\rangle$
    $s := \langle x, s\rangle$
    $s := \langle y-1, s\rangle$
    $sz := sz + 3$
  return $p_1(s)$

Aufgabe 3

c) Ackermann-Funktion als WHILE-Programm

```
def AckStack(x,y):
    stack = [x,y]
    while len(stack) > 1:
        y = stack.pop()
        x = stack.pop()
        if x == 0:
            stack.append(y + 1)
        elif y == 0:
            stack.append(x - 1)
            stack.append(1)
        else:
            stack.append(x - 1)
            stack.append(x)
            stack.append(y - 1)
    return stack.pop()
```

$\cancel{s := \langle y, \langle x, 0\rangle\rangle}$ $s := \langle x, 0\rangle$

$\cancel{sz := 2}$ $sz := 1$

~~while $sz > 1$ do~~ while $sz > 0$ do

$\cancel{y := p_1(s) \quad s := p2(s)}$

$x := p_1(s) \quad s := p2(s)$

$\cancel{sz := sz - 2}$ $sz := sz - 1$

if $x = 0$ then

$\cancel{s := \langle y + 1, s\rangle}$ $y := y + 1$

$\cancel{sz := sz + 1}$

else if $y = 0$ then

$s := \langle x - 1, s\rangle$

$\cancel{s := \langle 1, s\rangle}$ $y := 1$

$\cancel{sz := sz + 2}$ $sz := sz + 1$

else

$s := \langle x - 1, s\rangle$

$s := \langle x, s\rangle$

$\cancel{s := \langle y - 1, s\rangle}$ $y := y - 1$

$\cancel{sz := sz + 3}$ $sz := sz + 2$

~~return $p_1(s)$~~ return y