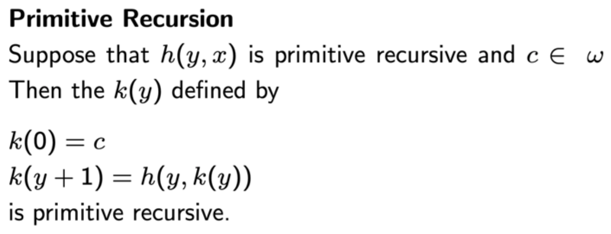
1) $pd$ is primitive recursion:



**Primitive Recursion**

Suppose that $h(y, x)$ is primitive recursive and $c \in \omega$

Then the $k(y)$ defined by

$k(0) = c$

$k(y+1) = h(y, k(y))$

is primitive recursive.

In our case, $k$ is $pd$.

$$pd(0) = c = 0$$

When $a > 0$,

$$pd(a) = a - 1$$

which is equivalent to

$$pd(a+1) = a$$

Say there's a projection function

$$h(y, x) = y$$

(which is of course primitive recursive), then

$$pd(a+1) = h\big(a, pd(a)\big) = a$$

QED.

2) This is a 2-parameter function. Let's call it $f$. We will do induction on $b$.

$$f(a,b) = \begin{cases} a - b, & a \geq b \\ 0, & otherwise \end{cases}$$

Base case:

$$f(a, 0) = a$$

Which is a projection function, which we know is primitive recursive.

Inductive case. Assuming $f(a,b)$ is primitive recursive, prove $f\big(a, s(b)\big)$ is also primitive recursive:

$$f(a, s(b)) = \begin{cases} f(a,b) - 1, & f(a-b) > 0 \\ 0, & otherwise \end{cases}$$

Which equals

$$f(a, s(b)) = pd(f(a,b))$$

And we just proved that $pd$ is primitive recursive. This is a composition schema. QED.

3) We define

$$\min(a,b) = a \dot{-} (a \dot{-} b)$$

Because if $a$ is the min, then $(a \dot{-} b) = 0$, and $a \dot{-} 0 = a$; similarly, if $b$ is the min, then $(a \dot{-} b) = a - b < a$, so it will also return $b$ correctly. $\min$ is thus primitive recursive by composition. QED.

4) $\min\,(a_1, \dots, a_n)$ can be done via an algorithm similar to mergesort:

$$\min(a_1, \dots, a_n) = \begin{cases} \min\,(\min(a_1, a_2)\,, \min(a_3, a_4)\,, \min(a_5, a_6)\,, \dots), & n > 1 \\ a_1, & n = 1 \end{cases}$$

As one can see, this is recursive. The base case is projection of one variable onto the variable itself (which is primitive recursive), and the inductive case is primitive recursive by composition. QED.

5) We define

$$\max(a, b) = a + (b \dot{-} a)$$

Because if $a$ is the max, then $(b \dot{-} a) = 0$, and it returns $a + 0 = a$; similarly, if $b$ is the max, then $(b \dot{-} a) = b - a$, and $a + b - a = b$, which is correct. $max$ is thus primitive recursive by composition of $+$ and $\dot{-}$.

6) Same as question 4, but just change $min$ to $max$.

7) We define

$$\bar{s}ignum(a) = 1 \dot{-} a$$

Because only when $a = 0$ does it return $1$. All other cases will make it return $0$. This function is thus primitive recursive by composition of $\dot{-}$ of two other primitive recursive functions: a constant function $h(a) = 1$ and a projection function $k(a) = a$.

8) We define

$$signum(a) = 1 \dot{-} \bar{s}ignum(a)$$

Because $1 \dot{-} 1 = 0$, which is the case when $a = 0$; and $1 \dot{-} 0 = 1$, as is the case when $a > 0$. $signum$ is thus primitive recursive by composition of $\dot{-}$ of two other primitive recursive functions: a constant function $h(a) = 1$ and $\bar{s}ignum(a)$, which we already proved to be primitive recursive. QED.

9) We define

$$|a - b| = \max(a \dot{-} b, b \dot{-} a)$$

Because at least one of $a \dot{-} b$ and $b \dot{-} a$ is $0$. The bigger value will be the absolute difference between $a$ and $b$. This function is thus primitive recursive because it is the composition schema by $max$ of two primitive recursive functions.

10) We define

$$\text{rem}(n, m) = \begin{cases} 0, & m = 0 \text{ or } n = 0 \text{ or } \text{rem}(n - 1, m) = m - 1 \\ \text{rem}(n - 1, m) + 1, & \text{otherwise} \end{cases}$$

Because when $n$ goes up by $1$, so does the remainder, unless the remainder is $m - 1$, in which case increasing $n$ by $1$ makes it divisible by $m$.

We now define $\mathrm{rem}$ again:

$$\mathrm{rem}(n,m) = (\mathrm{rem}(n-1,m)+1) \times \mathrm{signum}(m) \times \mathrm{signum}(n) \\ \times \overline{\mathrm{s}}\mathrm{ignnum}(\mathbb{I}(\mathrm{rem}(n-1,m) = m \dot{-} 1))$$

where $\mathbb{I}$ is the indicator function, which we now will prove is also primitive recursive:

claim:

$$\mathbb{I}(a,b) = \begin{cases} 1, & a = b \\ 0, & \text{otherwise} \end{cases}$$

Is primitive recursive.

Proof:

We can define it as

$$\mathbb{I}(a,b) = \bar{s}ignum(|a-b|)$$

Because if $a = b$, then $|a-b| = 0$, and $\bar{s}ignum$ of that is $1$; similarly, when they aren't equal, $|a-b| > 0$, which $\bar{s}ignum$ will return $0$.

Thus, $\mathrm{rem}$ is primitive recursive because it is a composition of primitive recursive functions $\times$, $\mathbb{I}$, $\dot{-}$, $+$, and $signum$. QED.

11) We define

$$[a+1|b] = \begin{cases} [a|b]+1, & rem(a+1,b) = 0 \\ [a|b], & \text{otherwise} \end{cases}$$

Which can be redefined as

$$[a+1|b] = [a|b] + \bar{s}ignum(rem(a+1,b))$$

Which is primitive recursive via composition of addition, $\bar{s}ignum$, $rem$, and $[x|y]$ via induction assumption.

12) I think $\phi(x,z)$ is missing the base case $\phi(x,0)$, because $y < 0$ does not exist in the natural numbers. I'm gonna assign $\phi(x,0) = 0$. Therefore,

$$\phi(x,z) = \begin{cases} 0, & z = 0 \\ \sum_{y<z} \psi(x,y), & \text{otherwise} \end{cases}$$

We just said $\phi$ satisfies the base case. In the inductive case:

$$\phi(x,z+1) = \sum_{y<z+1} \psi(x,y) = \left(\sum_{y<z} \psi(x,y)\right) + \psi(x,y+1) = \phi(x,z) + \psi(x,y+1)$$

Thus, $\phi$ is primitive recursive via $+$ and $\psi$ and $s(\cdot)$.

13) Once again, I think $\phi(x,z)$ is missing the base case $\phi(x,0)$, so we're gonna make it $1$. The function can now be defined as

$$\phi(x,z) = \begin{cases} 1, & z = 0 \\ \prod_{y<z} \psi(x,y), & \text{otherwise} \end{cases}$$

We just did the base case. For the inductive case,

$$\phi(x,z+1) = \prod_{y<z+1} \psi(x,y) = \left(\prod_{y<z} \psi(x,y)\right) \times \psi(x,y+1) = \phi(x,z) \times \psi(x,y+1)$$

Thus, $\phi$ is primitive recursive via $\times$ and $\psi$ and $s(\cdot)$.

14) That means we can have $(Q = 1, R = 1)$, $(Q = 0, R = 1)$, $(Q = 1, R = 0)$, but not $(Q = 0, R = 0)$. We could thus define

$$(Q \vee R)(x_1, \dots x_n) = signum(Q(x_1, \dots x_n) + R(x_1, \dots x_n))$$

Because $signum$ returns $0$ when the input is $0$ and outputs $1$ for everything greater than that.

$$\begin{gathered} 0 + 0 = 0 \\ 0 + 1 = 1 \\ 1 + 0 = 1 \\ 1 + 1 = 2 \end{gathered}$$

15) That means we need both $Q$ and $R$ to be $1$, so we could define

$$(Q \wedge R)(x_1, \dots x_n) = signum(Q(x_1, \dots x_n) \times R(x_1, \dots x_n))$$

Because

$$\begin{gathered} 0 \times 0 = 0 \\ 0 \times 1 = 0 \\ 1 \times 0 = 0 \\ 1 \times 1 = 1 \end{gathered}$$

16) When $Q$ returns $0$, we need to return $1$; when $Q$ returns $1$, we need to return $0$. We define

$$(\neg Q)(x_1, \dots x_n) = \bar{s}ignum(Q(x_1, \dots x_n))$$

17) We define

$$\phi(x,z) = (\exists y)\big(y < z \wedge R(x,y)\big) = \bigvee_{y<z} R(x,y)$$

This is primitive recursive because it is composition by $\vee$, which is primitive recursive.

18) We define

$$\psi(x,z) = (\forall y)\big(y < z \rightarrow R(x,y)\big) = \bigwedge_{y<z} R(x,y)$$

Because $y < z \rightarrow R(x, y) = \neg(y < z) \vee R(x, y) = (y \geq z) \vee R(x, y)$. Which makes the function $\forall y < z, R(x, y)$. And we proved $\wedge$ is primitive recursive.

19) Proof of Skolem's theorem

We will use strong induction on the length of the formula.

Base case: an atom is recursively defined to be a constant (constant function, which is primitive recursive) or a variable from the input variables (projection function, which is also primitive recursive), or an $n$-ary function $f$ whose arguments are atoms who is a composition of primitive recursive functions (primitive recursive functions are closed under composition).

Inductive case: assuming all sub-functions of $\phi$ at length $< i$ are primitive recursive, then $\phi$ is recursive. This is because $\phi$ is these smaller sub-functions joined by $\wedge$,$\vee$, $\neg$, $\exists(.)(.)$, or $\forall(.)(.)$, which we all proved in exercises 14 to 18 are primitive recursive.

QED.