Exercise 1.10

Amusements with Ackermann's function

Open this door at your own risk and proceed cautiously! You could stretch your imagination to the point of brain damage!

This all starts very calmly. We are introduced to an innocent-looking little procedure:

```
(define (A x y)
  (cond ((= y 0) 0)
        ((= x 0) (* 2 y))
        ((= y 1) 2)
        (else (A (- x 1)
                 (A x (- y 1))))))
```

And we are suggested to test it out with a few little expressions:

```
(A 1 10) ; => 1024
(A 2 4)  ; => 65536
(A 3 3)  ; => 65536
```

Well, nothing fancy so far. `(A 1 10)` produces 1024, which looks suspiciously like 2^{10}. `(A 1 4)` is 16 and `(A 1 8)` is 256. That's great, we have a pattern:

$$(\texttt{A 1 y}) = 2^y$$

Let's try some more:

```
(A 2 1) ; => 2
(A 2 2) ; => 4
(A 2 3) ; => 16
(A 2 4) ; => 65536
(A 2 5) ; => ...
```

Fascinating! The last one produces a Very Big Number—several screenfuls of digits. Continuing with `(A 2 6)`... and waiting... Still waiting... Might as well take a break and have some tea. Bad idea! The procedure is obviously not tail-recursive and piles up the stack. Quickly, press Ctrl-C twice!

Going down another avenue, we try `(A 3 y)`:

```
(A 3 1) ; => 2
(A 3 2) ; => 4
(A 3 3) ; => 65536
(A 3 4) ; => ...
```

This sequence grows even faster! Let's unleash another monster:

```
(A 4 1) ; => 2
(A 4 2) ; => 4
(A 4 3) ; => ...
```

Highly explosive! What construct could possibly grow so fast? To figure out what Ackermann's function actually does, we start by putting it's definition into this form:

$$A(x,y) = \begin{cases} 0 & \text{if } y = 0, \\ 2y & \text{if } x = 0, \\ 2 & \text{if } y = 1, \\ A(x-1, A(x, y-1)) & \text{otherwise} \end{cases}$$

We trace the process inductively, building up from the simplest case:

x = 0: $A(0,y) = 2y$, the argument is doubled.

x = 1: $A(1,y) = A(0,A(1,y-1)) = 2 \cdot A(1,y-1) = 2 \cdot A(0,A(1,y-2)) = 2 \cdot 2 \cdot A(1,y-2) = 2^y$, so it recursively reduces to the previous case, multiplying by 2 on each step.

x = 2: $A(2,y) = A(1,A(2,y-1)) = 2^{A(2,y-1)} = 2^{A(1,A(2,y-2))} = 2^{2^{A(2,y-2)}} = \underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{y} = {}^{y}2$, it again calls the previous case, which results in successive top-down exponentiation. Repeated exponentiation is called *tetration*.

x = 3: $A(3,y) = A(2,A(3,y-1)) = {}^{A(3,y-1)}2 = {}^{A(2,A(3,y-2))}2 = {}^{{}^{A(3,y-2)}2}2 = \underbrace{{}^{2^{\cdot^{\cdot^{2}}}}2}_{y}$, this is successive tetration.

x = 4: ...and so on.

We have established a general pattern:

$$
\begin{array}{lllllll}
A(0,y) & = & 2y & = & \underbrace{2+2+\ldots+2}_{y} & = & 2\uparrow^{0} y \\
A(1,y) & = & 2^{y} & = & \underbrace{2\cdot 2\cdot\ldots\cdot 2}_{y} & = & 2\uparrow^{1} y \;=\; 2\uparrow y \\
A(2,y) & = & {}^{y}2 & = & \underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{y} & = & 2\uparrow^{2} y \;=\; 2\uparrow\uparrow y \\
A(3,y) & & & = & \underbrace{{}^{2^{\cdot^{\cdot^{2}}}}2}_{y} & = & 2\uparrow^{3} y \;=\; 2\uparrow\uparrow\uparrow y \\
A(4,y) & & & = & \underbrace{{}^{{}^{2^{\cdot^{\cdot^{2}}}}2}2}_{y} & = & 2\uparrow^{4} y \;=\; 2\uparrow\uparrow\uparrow\uparrow y \\
\vdots & & & & & & \vdots \\
A(x,y) & & & & & = & 2\uparrow^{x} y \;=\; 2\underbrace{\uparrow\uparrow\ldots\uparrow}_{x} y
\end{array}
$$

Stunning! Turns out the x in $A(x, y)$ chooses a particular operation from an infinite hierarchy of operations. It starts with multiplication, which is repeated addition. Next is exponentiation, which is repeated multiplication. After that we enter a cosmic realm of hyperoperations: tetration, pentation, hexation, ...

Let's see what happens with our earlier examples.

$$\begin{aligned}
\texttt{(A 2 3)} &= 2^{2^2} = 16 \\
\texttt{(A 2 4)} &= 2^{2^{2^2}} = 2^{16} = 65536 \\
\texttt{(A 2 5)} &= 2^{2^{2^{2^2}}} = 2^{65536} \approx 2 \cdot 10^{19728} \\
\texttt{(A 2 6)} &= 2^{2^{2^{2^{2^2}}}} = {}^{6}2
\end{aligned}$$

We get growing exponentiation towers. The last one is 2 raised to an exponent so big it will take 10 pages to print. The resulting number is vastly bigger than googolplex ($10^{10^{100}}$). The entire Universe isn't big enough to store this number! Not even after digital compression and using every quark to store a bit.

Still harder to comprehend are tetration towers:

$$\begin{aligned}
\texttt{(A 3 2)} &= {}^{2}2 = 2^2 = 4 \\
\texttt{(A 3 3)} &= {}^{2^2}2 = {}^{4}2 = 2^{2^2} = 2^{16} = 65536 \\
\texttt{(A 3 4)} &= {}^{2^{2^2}}2 = {}^{4}2 = {}^{65536}2 = \underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}_{65536}
\end{aligned}$$

How so compact notation unleashes so enormous processes and creates so gigantic objects is utterly mindblowing! That's the secret power of double recursion! All the other recursions we have seen so far are just primitive recursions. Be it mutual recursions or tree recursions with two or more branches, they are still primitive.