Resolución del Parcial: Maria Fernanda Álvarez García y Carlos Iván Polanco Estrada.

Pregunta 1

Demuestre que para todo *natural de peano* "**n"** se cumple la siguiente propiedad:

$$Succ\ 0 + n = Succ\ n$$

Utilice la definición de *suma* estudiada en clase como definición del signo "+".

Respuesta:

Para esto, podemos concluir las siguientes propiedades base.

n + 0 = n
0 + m = m
n + Sa = S(n +a)
a + Sb = Sa + b

Por lo tanto

Succ 0 + n = Succ n
n = Succ Cero
Succ 0 + Succ Cero = Succ(Succ Cero)
Succ(Succ Cero + Cero)
Succ(Succ Cero) = Succ(Succ Cero)

ó

Succ 0 + n = Succ n
n = 0
Succ 0 + 0 = Succ 0
Succ 0 = Succ 0

Pregunta 2

Provea una definición inductiva para la propiedad "mayor que" (>) tal que:

$$a > b \begin{cases} Succ\ 0 & si\ a\ es\ mayor\ que\ b \\ 0 & de\ lo\ contrario \end{cases}$$

En otras palabras, la propiedad "mayor que" es equivalente a **Succ 0** si el primer valor es mayor que el segundo o **0** de lo contrario. Puede utilizar el operador ">" en su definición de la misma manera que se utiliza "+" en la definición de suma.

<u>Respuesta</u>

$a > b = \text{Succ Cero} > \text{Cero}$
$\text{Succ Cero} > \text{Cero} = a > b$

Las definiciones de suma son:

$n + 0 = n$
$0 + m = m$
$n + Sa = S(n + a)$
$a + Sb = Sa + b$

Esto será verdad siempre que se tome en cuenta lo siguiente:

$a > b = \text{Succ } a > b$
$a > \text{Cero} = \text{Succ } a > \text{Cero}$
$b > \text{Cero} = \text{Succ } b > \text{Cero}$
$b > a = \text{Cero}$

Y para demostrar:

$a > b = \text{Succ } a > \text{Cero}$
$a = \text{Succ Cero}$
Por lo tanto

$a > b = \text{Succ Cero} > \text{Cero}$

$a > b = \text{Succ } 0$
$a = \text{Succ } 0 , b = 0$
$\text{Succ } 0 > 0 = \text{Succ } 0$
$\text{Succ } (0 > 0) = \text{Succ } 0$
$\text{Succ } 0 = \text{Succ } 0$

$a > b = 0$
$a = 0, b = 0$
$0 > 0 = 0$
$0 = 0$

Pregunta 3

Provea una definición de las propiedades "esPar" e "esImpar" tal que:

$$esPar\ n \begin{cases} Succ\ 0 & si\ n\ es\ un\ numero\ par \\ 0 & de\ lo\ contrario \end{cases}$$

$$esImpar \begin{cases} Succ\ 0 & Si\ n\ es\ impar \\ 0 & de\ lo\ contrario \end{cases}$$

Se sugiere que, para implementar estas propiedades, utilice la propiedad inversa en la definición. En otras palabras, un numero "n" es par cuando cierto otro número es impar y viceversa.

<u>Respuesta.</u>

→ esPar Succ (n) = esImpar n
→ n = Succ (0)
→ esPar Succ (Succ (0)) = esImpar Succ (0)

→ es Impar Succ (n) = esPar n
→ n = 0
→ esImpar Succ (0) = esPar n

n será un número par, si el número al que está sumando es par y se le suma una unidad. n será un número impar, si un número de la suma es par y se le suma una unidad.

es Par n = Succ 0 + n : Esto, si y solo sí "n" es impar. Por ejemplo, esPar n = Succ Cero + 0
es Impar n = Succ 0 + n : Esto, sí y solo sí "n" es par. Por ejemplo, es Impar n = Succ Cero + Succ Cero

Ejemplo, para el caso de n es par en multiplicación:

Debemos tomar en cuenta que para todo número que sea par o impar, el multiplicarlo por 2, siempre nos dará un número par, por lo que:

Succ(Succ Cero)*n = n es par
Succ(Succ Cero)*n + Succ Cero = n es Impar

Si consideramos que cero es el primer par, y 1 el primero impar, entonces podemos establecer lo siguiente.

0 + Succ Cero = Succ Cero --Para impares

Entonces, siempre que n sea par, y al sumar una unidad, el resultado es impar, de sumar valor 0, el resultado es par. Y tendríamos.

1. n + Succ Cero = Succ n -- es par
2. n + Cero = n --es Impar

Pregunta 4

```
{-# LANGUAGE NoImplicitPrelude #-}

module Main where
import Prelude (IO, show, undefined, Show)


data Natural = Cero | Succ Natural deriving Show

Cero + m = m
n + Cero = n
n + (Succ a) = Succ (n + a)

--Reglas de multiplicación
n * Cero = Cero
n * Succ Cero = n
n * Succ a = n + (n * a)

sonIguales Cero Cero = Succ Cero
sonIguales Cero n = Cero
sonIguales n Cero = Cero
sonIguales (Succ a) (Succ b) = sonIguales a b

--Definir Fibonacci para factorial
fib Cero = Cero
fib (Succ Cero) = Succ Cero
fib (Succ (Succ a)) = fib (Succ a) + fib a

--Definición de factorial
factorial Cero = Succ Cero
factorial (Succ Cero) = Succ Cero
factorial (Succ(Succ a)) = Succ(Succ a) * factorial (Succ a)

--Fórmula del Predecesor,punto 4 del examen
predecesor Cero = Cero
predecesor (Succ Cero) = Cero
predecesor (Succ a) = a



main :: IO ()
main = undefined
```