UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

ESTRUCTURAS DISCRETAS 2020-1

Profesor: M. en C. I. C. Odín Miguel Escorza Soria
Ayudante de laboratorio: Salazar González Edwin Max

Práctica 5.

NÚMEROS NATURALES

1. OBJETIVOS

- Reforzar la construcción y propiedades de los números naturales

El principal objetivo de la práctica 5 es reforzar algunas propiedades de los números naturales.

Tiempo de entrega:	2 horas
Prerequisitos:	▪ Construcción y manejo de tipos en Haskell. ▪ Listas. ▪ Propiedades de los números naturales.

2. DESARROLLO

Esta práctica tomará al 0 como elemento perteneciente al conjunto de los números naturales (el cual se denotará como $\mathbb{N}$), dicho conjunto se define como [3]:

$$\mathbb{N} = \{0{,}1{,}2{,}3{,}...\}$$

Los números naturales es un conjunto infinito ordenado y numerable. Al ser un conjunto ordenado sus elementos pueden presentar las relaciones de igualdad, menor que o mayor que ($=$, $\leq y \geq$). Al ser un conjunto infinito siempre se puede encontrar el sucesor de un elemento, dicha propiedad de ser el sucesor de n se denota como $s(n)$ $[2, 4]$.

La definición de número natural es la siguiente $[1, 2]$:

- "El 0 es un número natural."
- "Si n es un número natural, entonces $S(n)$ es un número natural."
- "Estos y solo éstos son números naturales."

2.1. Axiomas de Peano

Los axiomas de peano son una estructura mínimia desarrollado por Giuseppi Peano[1] en 1889 en la cual definió la suma y producto como actualmente se conoce. Adicionalmente introdujo el concepto de función sucesor[12].

Los axiomas que se definen a continuación son de gran importancia pues éstos definen de manera exacta al conjunto de los números naturales $[1, 2, 4]$.

- 0 es un número natural.
- Para cada número natural n existe un número natural llamado sucesivo y se denota como $s(n)$.
- Para todo número natural n, $s(n)$ es diferente a 0.
- Si $s(n) = s(m)$ entonces se deduce $n = m$.
- Para cualquier predicado P se cumple: $P(0) \wedge \forall n(P(n) \rightarrow P(s(n))) \rightarrow \forall n P(s(n))$.

Este úlitmo axioma se le comoce como el axioma de inducción completa que dice: "El conjunto de números naturales, que contiene 0 y para cada uno de n elementos, el elemento sucesivo $s(n)$, contiene todos los números naturales" [4].

Para cualesquiera $m, n, r \in \mathbb{N}$ se tienen las siguientes propiedades para la adición y multiplicación [1–4]:

[1] (1853 - 1932) Matemático italiano que se destacó por contribuciones a la lógica matemática y teoría de números.

Neutro aditivo	$m + 0 = m$.
	$m + S(n) = s(m + n)$.
Conmutatividad	$m + n = n + m$.
Ley de cancelación por la izquierda	Si $m + n = m + r$ entonces $n = r$.
Elemento nulo	$n * 0 = 0$.
Neutro multiplicativo	$n * 1 = n$.
	$m * s(n) = m * n + m$.
Ley de cancelación por la izquierda	Si $m * n = m * r$ y $m \neq 0$ entonces $n = r$.

Cuadro 1: Propiedades para la adición y multiplicación de números naturales.

3. INSTRUCCIONES

Descarga el archivo Practica5.hs y resuelve los ejercicios definidos sobre éste.

4. EJERCICIOS

- Ejercicio 1: resolver la función *to_nat* que recibe un entero y regresa su tipo de dato Natural.

```
> to_nat 5
S(S(S(S(S(Zero)))))
> to_nat (-4)
error "A negative natural number? ._."
```

Código 1: Ejemplo de llamada función *to_nat*.

- Ejercicio 2: resolver la función *addition* que recibe dos naturales y regresa su suma.

```
> addition (S(S(Zero))) (S(S(Zero)))
S(S(S(S(Zero))))
```

Código 2: Ejemplo de llamada función *addition*.

- Ejercicio 3: resolver la función *mult* que recibe dos naturales y regresa su multiplicación.

```
> mult (S(S(S(Zero)))) (S(S(Zero)))
S(S(S(S(S(S(Zero))))))
```

Código 3: Ejemplo de llamada función *mult*.

- Ejercicio 4: resolver la función *lt* (less than) que recibe dos naturales 'n' y 'm' y regresa *True* si n<m, *False* en otro caso.

```
> lt (S(S(S(S(Zero))))) (S(S(Zero)))
False
> lt (S(S(Zero))) (S(S(S(S(Zero)))))
True
```

Código 4: Ejemplo de llamada función *lt*.

- Ejercicio 6: resolver la función *min_nat* que recibe una lista de Naturales y regresa el elemento más pequeño.

```
> min_nat [S(S(S(S(Zero)))), S(S(S(S(S(Zero))))), S(S(Zero)), ↩
   S(S(S(S(S(S(S(Zero)))))))]
S(S(Zero))
```

Código 5: Ejemplo de llamada función *min_nat*.

- Ejercicio 7: resolver la función *to_int* que recibe un tipo de dato Natural y regresa su número entero.

```
> to_int (S(S(S(S(S(S(S(Zero))))))))
7
```

Código 6: Ejemplo de llamada función *to_int*.

- Ejercicio 8: definir una instancia de la clase *Show* para el tipo de dato Natural de tal manera que las instancias creadas sean cadenas de números (cadenas de números porque deben recordar que *show* va a mostrar únicamente caracteres y cadenas, ¡Números no!)

  Nota: no olviden *Show* que viene declarado al final del tipo de dato Natural, solo debe quedar *derivingEq*

```
> (S(S(S(S(S(S(S(Zero))))))))
7
> addition (S(S(Zero))) (S(S(Zero)))
4
> mult (S(S(S(Zero)))) (S(S(Zero)))
6
> min_nat [S(S(S(S(Zero)))), S(S(S(S(S(Zero))))), S(S(Zero)), ↩
   S(S(S(S(S(S(S(Zero)))))))]
2
```

Código 7: Ejemplos de tipo de dato Naturales ya con instancias de la clase *Show*

- Ejercicio 9: definir una función cuyo nombre sea fac que recibe un Natural 'n' y regresa el factorial de 'n'

```
> fac (S(S(S(Zero))))
S(S(S(S(S(S(Zero))))))
```

Código 8: Ejemplo de llamada función fac.

- Ejercicio 10: resolver la función *substraction* que recibe dos tipo de datos Natural 'n' y 'm' y regresa la substracción de n a m (n-m).

```
> substraction (S(S(S(S(S(S(S(Zero)))))))) (S(S(Zero)))
S(S(S(S(S(Zero)))))
> substraction (S(S(Zero))) (S(S(S(S(S(S(S(Zero))))))))
error "Hey!␣Watch␣out!"
```

Código 9: Ejemplo de llamada función *substraction*.

- Ejercicio 11: resolver la función *isordered* que recibe una lista de tipos Natural y regresa $True$ si la lista esta ordenada de manor a mayor, $False$ en otro caso.

```
> isordered [S(S(S(S(Zero)))), S(S(S(S(S(Zero))))), S(S(Zero))↩
   , S(S(S(S(S(S(S(Zero)))))))]
False
> isordered [S(S(Zero)), S(S(S(S(Zero)))), S(S(S(S(S(Zero)))))↩
   ,  S(S(S(S(S(S(S(Zero)))))))]
True
```

Código 10: Ejemplo de llamada función *isordered*.

- Ejercicio 12: resolver la función *division* que recibe dos tipos de datos Natural 'n' y 'm' y regresa la división de n entre m.

```
> division (S(S(S(S(S(S(S(Zero))))))) (S(S(Zero)))
S(S(S(Zero)))
```

Código 11: Ejemplo de llamada función *division*.

- Ejercicio Extra: no aplica para esta practica.

5. ESPECIFICACIONES

✓ Respetar las firmas de las funciones. Cambiarlas podría ser motivo de anulación del ejercicio.

× Cualquier plagio de prácticas será evaluado con 0, sin hacer indagaciones. **Creen** su propio código.

× Cualquier práctica entregada posterior a la fecha límite NO será tomada en cuenta.

Se deberá contar con un difectorio cuyo nombre sea Practica5. Dentro del directorio se deben tener:

- README.txt, donde se incluya nombre y número de cuenta de los integrantes del equipo junto con comentarios que crean pertinentes sobre la práctica.
- Practica5.hs, script requerido para esta práctica.

Compirmir el directorio con el formato **ApellidoNombreP5**. Comprimir con extensión .tar.gz o .zip

Solamente un integrante del equipo deberá enviar la práctica.

Enviar la práctica al correo **ciclomax9@ciencias.unam.mx** con el asunto `[LC-Apellido-Nombre-P5]`.

"A computer once beat me at chess, but it was no match for me at kick boxing"
- Emo Philips

Suerte ☺

Referencias

[1] Miranda, Favio y Viso, Elisa, *Matemáticas Discretas*. México DF, México: Las prensas de ciencias, 2010.

[2] Karl G., Vinfired y Tremblay, Jean-Paul, *Matemática Discreta y lógica: Una perspectiva desde la Ciencia de la Comptuación*. España: Prentice Hall, 1997.

[3] Gómez Laveaga, Carmen, *Álgebra superior. Curso completo*. UNAM, México: UNAM, 2014.

[4] Tsipkin, A.G. *Manual de matemáticas para la enseñanza media*. Rusia, 1979.