Lenguajes de Programación

Práctica 1

Semestre 2023-1
Facultad de Ciencias, UNAM

Profesor: Manuel Soto Romero
Ayud. Lab.: Silvia Díaz Gómez

Fecha de inicio: 19 de Agosto de 2022
Fecha de entrega: 02 de Septiembre de 2022

Descripción

La práctica consiste en completar el cuerpo de las funciones faltantes del archivo **`Practica1.hs`** y las firmas de las funciones deben ser idénticas a las que se muestran en cada ejercicio. Cada función debe estar debidamente comentada con la especificación de ésta. No está permitido usar funciones predefinidas del lenguaje que resuelvan directamente los ejercicios.

Ejercicios

1. (0.5 pts.) Completar el cuerpo de la función **`areaLateral`** que dados el largo, ancho y altura de un paralelepípedo rectángulo respectivamente, calcula el área lateral del mismo. Fórmula:

$$Al = 2\,(a + b)\,c$$

```
areaLateral :: Float -> Float -> Float -> Float
```

2. (0.5 pts.) Completar el cuerpo de la función **`areaTotal`** que dada la generatriz y el diámetro de la base de un cono circular recto, calcula el área total del mismo. Usar asignaciones locales **`let`** o **`where`** para evitar cálculos repetitivos. Fórmula:

$$At = \pi r g + \pi r^2$$

```
areaTotal :: Float -> Float -> Float
```

3. (0.5 pts.) Completar el cuerpo de la función **`distancia`** que recibe dos puntos y regresa la distancia entre ellos.

```
distancia :: (Float, Float) -> (Float, Float) -> Float
```

```
> distancia (1.3,4.0) (3.0,4.5)
  1.7720045
> distancia (1.3,4.0) (1.3,4.0)
  0.0
```

4. (0.5 pts.) Completar el cuerpo de la función **impl** que recibe dos booleanos y regrese la implicación lógica de estos.

```
impl :: Bool -> Bool -> Bool
```

```
> impl True False
  False
> impl True True
  True
```

5. (0.5 pts.) Completar el cuerpo de la función **calculadora** que recibe parámetros, el primero indica la operación que se va a realizar y el segundo con los operandos. Las posibles operaciones son:
"first" = devuelve el primer elemento de la tupla
"last" = devuelve el segundo elemento de la tupla
"sum" = suma
"dif" = resta
"mul" = realiza la multiplicación
"div" = división
"pow" = potencia(el primer elemento de la tupla elevando al segundo)

```
calculadora :: String -> (Int, Int) -> Int
```

```
> calculadora "first" (134, 389)
  134
> calculadora "pow" (2,4)
  16
```

6. (1 pto.) Al gato Loki le encanta pasar el día jugando en la calle, pero sólo lo hace cuando la temperatura está entre 15 y 25 grados, menos en verano que tolera un poco más el calor y sale a jugar cuando la temperatura ésta entre 20 y 30. Definir una función **loki** que recibe la temperatura y un booleano indicando si es o no verano, y le diga a Loki si puede salir a jugar.

```
loki :: Int -> Bool -> String
```

```
> loki  35 False
  "No sale a jugar"
> loki  20 False
  "Sale a jugar"
```

7. (1 pto.) Un niño quiere subir saltando una escalera. Con cada salto que da, puede subir 1, 2 o 3 escalones. Por ejemplo, si la escalera tiene tres escalones, la puede subir de cuatro formas distintas: 1, 1, 1; 1, 2; 2, 1 o 3. Completar el cuerpo de la función recursiva **numeroFormas** que dado el número de escalones de una escalera, indica el número de formas que puede subir saltando el niño. Por ejemplo:

```
numeroFormas :: Int -> Int
```

```
> numeroFormas 3
  4
> numeroFormas 4
  7
> numeroFormas 10
  274
```

8. Completar el cuerpo de las siguientes funciones sobre listas.

   (a) (0.5 pts.) Completar el cuerpo de la función divisoresPropios que recibe un entero y regresa una lista con los divisores propios de éste. Los divisores propios de un número n son todos los enteros distintos de n que son divisores de n.

```
divisoresPropios :: Int -> [Int]
```

```
> divisoresPropios 45
  [1,3,5,9,15]
```

   (b) (1 pto.) Una forma de encontrar los números primos en un determinado rango es mediante la conocida *Criba de Eratóstenes*. El algoritmo consiste en ir tomando los números del rango y eliminar todos los números que sean múltiplos de éste. El algoritmo terminará cuando no se pueda eliminar ningún número más.

   Por ejemplo, para encontrar los números primos del 2 al 20, se tiene la siguiente lista:

```
'(2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20)
```

El primer paso consiste en eliminar todos aquellos números que sean múltiplos de dos (excepto el primero), con lo cual quedaría la siguiente lista:

```
'(2 3 5 7 9 11 13 15 17 19)
```

Ahora, se pasa al siguiente número en la lista, en este caso el tres, y se repite el procedimiento:

```
'(2 3 5 7 11 13 17 19)
```

Para el siguiente paso, se deben eliminar los múltiplos de cinco, sin embargo no queda ningún múltiplo de este número en la lista con lo cual, termina el algoritmo y se concluye que los números primos del 2 al 20 son: **2**, **3**, **5**, **7**, **11**, **13**, **17** y **19**.

Completar el cuerpo de la función recursiva **cribaEratostenes** que encuenta los números primos en un rango de 2 a n usando la *Criba de Eratóstenes*. Por ejemplo:

```
cribaEratostenes :: Int -> [Int]
```

```
> cribaEratostenes 20
  [2 3 5 7 11 13 17 19]
```

9. Completar los siguientes ejercicios haciendo uso de las funciones de orden superior **map**, **filter**, **foldr** y/o **foldl**. Para este ejercicio se prohíbe definir funciones auxiliares, en caso de requerirlas, usar asignaciones locales **let** o **where**.

   (a) (1 pto.) Completar el cuerpo de la función **perfectos** que recibe una lista de números y regresa una nueva lista que contiene únicamente aquellos que son perfectos. Un *número perfecto* es un número natural que es igual a la suma de sus divisores propios positivos. Por ejemplo, 6 es un número perfecto porque sus divisores propios son 1, 2 y 3; y $6 = 1 + 2 + 3$. Por ejemplo:

```
perfectos :: [Int] -> [Int]
```

```
> perfectos [1,3,6,7]
  [6]
```

   (b) (1 pto.) Completar, usando **foldr**, el cuerpo de la función recursiva **(aproxima n)** tal que:

$$aproxima\ n = \sum_{i=0}^{i=n} \frac{1}{i!}$$

Por ejemplo:

```
aproxima :: Float -> Float
```

```
> aproxima 5
  2.7166667
```

10. (1 pto.) Definir un tipo de dato **Figura** que sea utilizado para trabajar con figuras geométricas.

    El tipo de dato debe contener:

    - Un constructor **(Triangulo a b c)** donde **a**, **b** y **c** son números reales y representan los lados del triángulo.
    - Un constructor **(Cuadrado a)** donde **a** es un número real y representa el lado del cuadrado.
    - Un constructor **(Rectangulo a b)** donde **a** y **b** son números reales y representan la altura y base del rectángulo.
    - Un constructor **(Rombo a D d)** donde **a**, **D** y **d** son números reales y representan el lado, diagonal mayor y diagonal menor del rombo respectivamente.
    - Un constructor **(Paralelogramo a b h)** donde **a**, **b** y **h** son números reales y representan los lados y altura del paralelogramo respectivamente.
    - Un constructor **(Circulo D)** donde **D** es un número real y representa el diámetro del círculo.
    - Un constructor **(Elipse a b)** donde **a** y **b** son números reales y representan el semieje mayor y el semieje menor de la elipse respectivamente.

    Una vez definido el tipo de dato, se deben de definir las siguientes funciones:

11. (1 pto.) Una función **area** que dada una figura calcule el área de ésta.

```
area :: Figura -> Float
```

```
> area (Cuadrado 3)
  9.0
```

Requerimientos

- La entrega se realiza mediante correo electrónico a la dirección del ayudante de laboratorio **silvinha@ciencias.unam.mx**
- Se debe entregar un directorio **`numeroCuenta_P01`**, dónde numeroCuenta es el número de cuenta de uno de los integrantes del equipo. Dentro del directorio se debe incluir:
  - Un archivo readme.txt con el nombre y número de cuenta de los integrantes, comentarios, opiniones, críticas o ideas sobre la práctica.
  - Los archivos requeridos en la práctica. El único archivo requerido es Practica1.hs.
  - El directorio se deberá comprimir en un archivo con **`numeroCuenta_P01`**.
- El asunto del correo debe ser [LDP-20231-P01].
- Se recibirá la práctica hasta las 23:59:59 horas del día fijado como fecha de entrega, cualquier práctica recibida después no será tomada en cuenta.
- El orden en el que aparezcan las funciones en el archivo solicitado, debe ser el orden especificado en este PDF, de lo contrario podrán penalizarse algunos puntos. Puedes utilizar funciones auxiliares, se recomienda definirla después de la función que la va a utilizar. No dudes en aclarar tus dudas en la sesión de laboratorio y vía correo electrónico.

¡Que tengas éxito en tu primera práctica!.