Logotipo

Descripción generada automáticamenteImagen que contiene Logotipo

Descripción generada automáticamenteInstituto Politécnico Nacional

Escuela Superior de Computo

Materia:

Paradigmas de Programación

Grupo:

3CM3

Nombre:

Rodriguez Sanchez Daniel

Méndez Pichardo Cesar Daniel

Monroy Mora Arturo

Rubio Haro Diego

Luna Martínez Jesús Alberto Dorian

Titulo:

*“Manual Técnico”*



Texto

Descripción generada automáticamentePara empezar, tuvimos que usar estas librerías para poder exportar algunas funciones que tratan con caracteres y con esto comprobamos si un carácter es uno de los que están en el espacio en blanco incluyendo tabuladores y saltos de línea.

Después empezamos a definir variables, como Expr que es una biblioteca para manipular archivos de haskell dinámicamente. Y este a su vez nos proporciona mas de 100 funciones para leer, construir, evaluar, comparar, canonizar y emparejar expresiones.

Usamos la variable Var que es de tipo Char, Const que es de tipo Double, BinExp que creamos 2 variables (BinExp Exp) y (BinExp BinaryOp Exp) y estas van a recibir 2 datos de expresión, UnExp al igual tendremos 2 variables (UnExp UnaryOp y Exp), deriving Eq.

Creamos data BinaryOp, creamos esta función con variables, que nos van a ayudar a hacer la suma, resta, multiplicación, división, o potencia.

Texto

Descripción generada automáticamenteCreamos data UnaryOp, creamos esta función con variables, que nos ayudaran a simplificar operaciones unarias como sen, cos , tan, etc.

Texto

Descripción generada automáticamenteEn el instance Show se imprimirán las expresiones separadas ya sea por el paréntesis operadores aritméticos, derivadas de funciones, etc.

Captura de pantalla de computadora

Descripción generada automáticamente

Aquí empezaremos a hablar sobre la simplificación, por lo que hicimos reglas para poder simplificar expresiones usando la coincidencia de los patrones.

Creamos una función llamada *Simplificar* y los datos tienen que seguir una serie de patrones.

La función va a recibir una expresión *Expr* y va a devolver una expresión *Expr*., la función lo que va a hacer es comparar entre 2 variables o constantes y todo esto va a seguir la instrucción del parámetro que puede ser *Sumar, Restar Multiplicar, Dividir, Potencia y Negación, y nos tiene que regresar una función Expr.*

Captura de pantalla de computadora

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

Solo que aquí tenemos una regla, lo que pasa es que simplificar una variable pues esto será igual a la variable.

Captura de pantalla de computadora

Descripción generada automáticamente

Al final tenemos una función llamada “simplificarfull” y lo que hace es simplificar la función hasta que ya no se pueda simplificar mas.

Captura de pantalla de computadora

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamenteAhora llego el momento de hablar de las Derivadas, por lo que de nuevo hicimos ciertas reglas para poder escribir las derivadas de un método correcto.

En esta parte vamos a generar el lexer, esto para que una cadena sin sentido sea convertida en una lista de cosas como una literal de numero, literal de cadena, literal de identificador o de operador, y esto lo hacemos para que reconozca caracteres reservados y/o palabras clave y también descarta espacios en blanco.

Captura de pantalla de computadora

Descripción generada automáticamente

Es justo aquí donde se va a construir el analizador, y aquí es donde va a analizar un símbolo de operador básico o un prefijo.

Aquí nos vamos a encontrar con buildExpressionParser que se va a encargar de toda la complejidad involucrada al construir el analizador, en opTable, podemos definir una variedad de expresiones con mucha facilidad.

Captura de pantalla de computadora

Descripción generada automáticamente

En esta ultima parte usamos el fmap porque se define como una de las funciones cuyas implementaciones deben ser proporcionadas por los tipos de datos que deseen pertenecer a la clase expresión.