Министерство образования и науки РФ
Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа программной инженерии

КУРСОВАЯ РАБОТА

по дисциплине «Языки и средства функционального программирования»

Реализация калькулятора на языке Haskell

Выполнил

студент группы 3530904/80005 Василевский А. Н.

Руководитель: Лукашин А. А.

Санкт-Петербург

2019

Оглавление:

1. Введение

Функциональное программирование — парадигма программирования, в которой процесс вычисления трактуется как вычисление значений функций в математическом понимании последних (в отличие от функций как подпрограмм в процедурном программировании). Функциональное программирование предполагает обходиться вычислением результатов функций от исходных данных и результатов других функций, и не предполагает явного хранения состояния программы. Соответственно, не предполагает оно и изменяемость этого состояния.

Функциональное программирование характеризует следующие особенности:

- краткость и простота;
- строгая типизация;
- модульность;
- функции — это значения;
- чистота (отсутствие побочных эффектов);
- отложенные (ленивые) вычисления.

2. Задание

Реализовать калькулятор простейших арифметических выражений на функциональном языке программирования Haskell.

3. Описание работы программы

3.1 Описание библиотеки

В процессе разработки калькулятора была использована библиотека *Parsec* и несколько вспомогательных модулей:

1. *Text.Parsec.Token* – служит для разбора лексических элементов (токенов);

2. *Text.Parsec.String* – делает String экземпляром Stream с типом токена Char.

3. *Text.Parsec.Language* – распознает некоторые определения языка

4. *Text.Parsec.Expr* – служит для разбора "выражений"

3.2 Описание некоторых функций:

1. Функция *lexis*

С помощью выражения *makeTokenParser* создает запись, которая содержит лексические синтаксические анализаторы, полученные с использованием определений языка

2. Функция *getNumber*

Анализатор *naturalOrFloat*, исходя из названия разбирает, является число натуральным или с плавающей запятой и возвращает это число в переменную *value*. С помощью оператора выбора *case* возвращается значение числа в зависимости от его типа

3. Функция *input* служит в процессе работы программы для удаления пробелов между арифметическими значениями.

4. Функция *analysisExpression* служит для разбиения выражения на унарные операторы числа (+ или -) и бинарные операторы арифметики (+ - сложение, — - вычитание, * - умножение, / - деление, % - деление с остатком). Использованная функция *flip* меняет порядок аргументов в передаваемой функции.

Исходный код программы -

4. Скриншоты работы программы

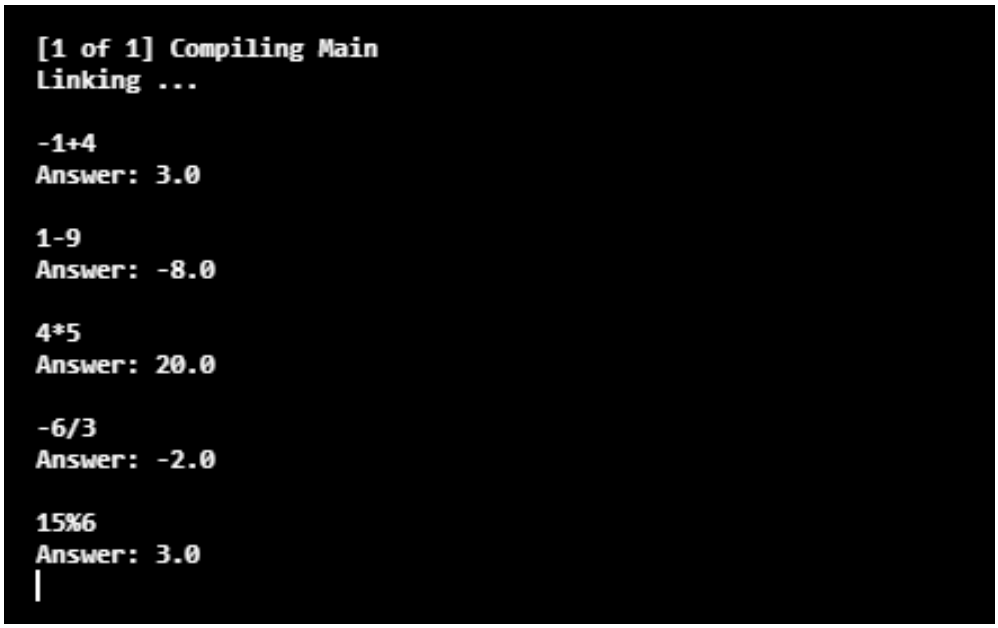

Рис. 1 Результат работы программы

Рис. 2 Результат работы программы для выражения

Рис. 3 Проверка работы программы с пробелами в выражениях

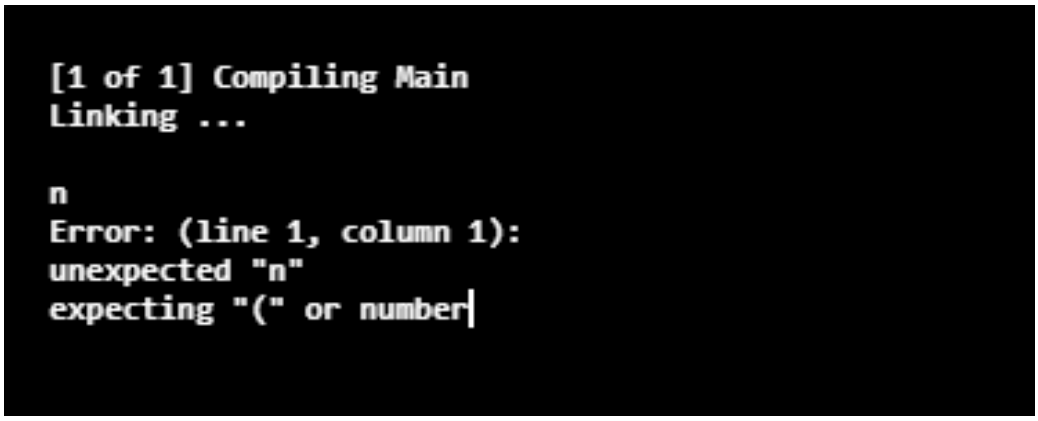

Рис. 4 Ввод некорректных данных

5. Заключение

В данном курсовом проекте был создан проект калькулятора с помощью функционального языка Haskell на основе библиотеки *Parsec*. Калькулятор поддерживает арифметические операции: сложение, вычитание, умножение, деление, включая деление с остатком, а также простейшие скобочные выражения. Все данные поступают через стандартный ввод, результат выводится в стандартный поток вывода. При вводе символов, не предусмотренных программой, выводится сообщение об ошибке.