МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное образовательное
учреждение высшего образования
ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

Институт математики, механики и компьютерных наук
имени И. И. Воровича

Направление подготовки Фундаментальная информатика
и информационные технологии

МОДИФИКАЦИЯ МЕХАНИЗМА ОБРАБОТКИ ОТРИЦАТЕЛЬНЫХ ЛИТЕРАЛОВ В КОМПИЛЯТОРЕ GHC

Курсовая работа

Студента 3 курса
Д. Ш. Мирзоева

Научный руководитель:
Старший преподаватель кафедры информатики и вычислительного
эксперимента Мехмата ЮФУ В. Н. Брагилевский

Оглавление

Введение

Во многих языках программирования присутствует понятие литерала. Литерал — это способ записи данных, неизменяемых в ходе работы программы.[1]

В данной курсовой работе нами рассмотрен язык Хаскель. В его стандарте [2] отсутствуют отрицательные литералы в качестве отдельной лексемы. Для записи отрицательных чисел используется унарная операция минус. Такой подход порождает некоторые особенности, что побудило сообщество разработчиков создать расширение для языка, добавляющее настоящие отрицательные литералы.

Как выяснилось, способ, которым было реализовано данное расширение, не позволял корректно обрабатывать значение “отрицательный ноль”. `-0` компилятор неверно распознавал как обыкновенный, положительный ноль. Такое поведение было воспринято как ошибка. Нами были рассмотрены несколько путей решения данной проблемы. Один из этих способов оказался приемлем для разботчиков компилятора. Изменения подготовленные нами были приняты в основной репозиторий.

1 Литералы в Haskell

1.1 Виды литералов

В Хаскель можно выделить три вида литералов: целочисленные, вещественные, символьные и строковые.

data Bool = True | False

Рис. 1.1: Определение значений True и False

В отличие от многих других языков, логические значения `True` и `False` являются идентификаторами. Они входят в состав стандартного модуля `Prelude` и определены как показано в листинге 1.1.

$$
\begin{array}{lcl}
\mathit{decimal} & \rightarrow & \mathit{digit}\{\mathit{digit}\} \\
\mathit{octal} & \rightarrow & \mathit{octit}\{\mathit{octit}\} \\
\mathit{hexadecimal} & \rightarrow & \mathit{hexit}\{\mathit{hexit}\}
\end{array}
$$

$$
\begin{array}{lcl}
\mathit{integer} & \rightarrow & \mathit{decimal} \\
 & | & \texttt{0o}\ \mathit{octal} \mid \texttt{0O}\ \mathit{octal} \\
 & | & \texttt{0x}\ \mathit{hexadecimal} \mid \texttt{0X}\ \mathit{hexadecimal}
\end{array}
$$

Рис. 1.2: Лексическая структура целочисленных литералов

Целочисленные литералы представляют значения натуральных чисел, включая ноль. Существует синтаксис для записи в различных системах счисления: десятичной, восьмеричной, шестнадцатеричной. Примеры: `30`, `0o36`, `0O36`, `0x1E`, `0X1E`.

$$
\begin{array}{lcl}
\textit{float} & \rightarrow & \textit{decimal}\ \texttt{.}\ \textit{decimal}\ [\textit{exponent}] \\
& | & \textit{decimal}\ \textit{exponent} \\
\\
\textit{exponent} & \rightarrow & (\texttt{e} \mid \texttt{E})\ [\texttt{+} \mid \texttt{-}]\ \textit{decimal}
\end{array}
$$

Рис. 1.3: Лексическая структура вещественных литералов

Вещественные литералы представляют рациональные числа. Можно использовать как обычные десятичные дроби, так и научную форму записи с указанием мантиссы и порядка. Примеры: `123.456`, `0.123456e3`.

$$
\begin{array}{lcl}
\textit{char} & \rightarrow & \texttt{'}\ (\textit{graphic}_{\langle \texttt{'} \mid \texttt{\textbackslash} \rangle} \mid \textit{space} \mid \textit{escape}_{\langle \texttt{\textbackslash\ \&} \rangle})\ \texttt{'} \\
\textit{string} & \rightarrow & \texttt{"}\ \{\textit{graphic}_{\langle \texttt{"} \mid \texttt{\textbackslash} \rangle} \mid \textit{space} \mid \textit{escape} \mid \textit{gap}\}\ \texttt{"} \\
\textit{escape} & \rightarrow & \texttt{\textbackslash}\ (\ \textit{charesc} \mid \textit{ascii} \mid \textit{decimal} \mid \texttt{o}\ \textit{octal} \mid \texttt{x}\ \textit{hexadecimal}\ ) \\
\textit{charesc} & \rightarrow & \texttt{a} \mid \texttt{b} \mid \texttt{f} \mid \texttt{n} \mid \texttt{r} \mid \texttt{t} \mid \texttt{v} \mid \texttt{\textbackslash} \mid \texttt{"} \mid \texttt{'} \mid \texttt{\&} \\
\textit{ascii} & \rightarrow & \texttt{\^{}}\ \textit{cntrl} \mid \texttt{NUL} \mid \texttt{SOH} \mid \texttt{STX} \mid \texttt{ETX} \mid \texttt{EOT} \mid \texttt{ENQ} \mid \texttt{ACK} \\
& | & \texttt{BEL} \mid \texttt{BS} \mid \texttt{HT} \mid \texttt{LF} \mid \texttt{VT} \mid \texttt{FF} \mid \texttt{CR} \mid \texttt{SO} \mid \texttt{SI} \mid \texttt{DLE} \\
& | & \texttt{DC1} \mid \texttt{DC2} \mid \texttt{DC3} \mid \texttt{DC4} \mid \texttt{NAK} \mid \texttt{SYN} \mid \texttt{ETB} \mid \texttt{CAN} \\
& | & \texttt{EM} \mid \texttt{SUB} \mid \texttt{ESC} \mid \texttt{FS} \mid \texttt{GS} \mid \texttt{RS} \mid \texttt{US} \mid \texttt{SP} \mid \texttt{DEL} \\
\textit{cntrl} & \rightarrow & \textit{ascLarge} \mid \texttt{@} \mid \texttt{[} \mid \texttt{\textbackslash} \mid \texttt{]} \mid \texttt{\^{}} \mid \texttt{\_} \\
\textit{gap} & \rightarrow & \texttt{\textbackslash}\ \textit{whitechar}\ \{\textit{whitechar}\}\ \texttt{\textbackslash}
\end{array}
$$

Символьные литералы записываются в одинарных кавычках, а строковые в двойных. Примеры: `'a'`, `"Hello, World!"`.

2 Расширение NegativeLiterals

3 Проблема обработки отрицательного нуля в расширении NegativeLiterals

Литература

[1] Harry Henderson, *Encyclopedia of computer science and technology*, 2003

[2] Simon Marlow(editor), *Haskell 2010 Language Report*