Лекция 8. Функциональные парсеры

Пережогин А.С.

11 октября 2012 г.

Что такое парсеры?

Парсер – программа, которая анализирует часть текста и выделяет синтаксическую структуру.
Парсеры используется во всех программах для предобработки введенной информации.
В функциональном языке Haskell парсеры рассматриваются как функции

```
type Parser = String -> Tree
```

Входные данные в виде строки преобразуются в дерево.
Парсер может обработать не всю строку и часть останется в виде строки

```
type Parser = String -> (Tree,String)
```

Парсер может содержать список нескольких разобранных деревьев и остатков от строк

```
type Parser = String -> [(Tree,String)]
```

В самом общем случае, парсер может давать не только дерево, но и более общий объект

```
type Parser a = String -> [(a,String)]
```

Базовые парсеры

Парсер *item* возвращает пустую строку или выделяет первый символ, если строка не пуста

```
item :: Parser Char
item = \ inp -> case inp of
                    [] -> []
                    (x:xs) -> [(x,xs)]
```

Парсер *failure* всегда выдает пустую строку

```
failure :: Parser a
failure = \ inp -> []
```

Парсер *return v* всегда выдает v независимо от входного значения

```
return :: a -> Parser a
return v = \ inp -> [(v,inp)]
```

Парсер *p+++q* возвращает парсер p в случае, если успешно завершилось или парсер q в противном случае

Базовые парсеры

```
(+++) :: Parser a -> Parser a -> Parser a
p +++ q = \inp -> case p inp of
                    [] -> parse q inp
                    [(v,out)] -> [(v,out)]
```

Функция parse преобразует парсер в строку

```
parse :: Parser a -> String -> [(a,String)]
parse p inp = p inp
```

Примеры работы простейших парсеров

```
> parse item ""
[]
> parse item "abc"
[('a',"bc")]

> parse failure "abc"
[]

> parse (return 1) "abc"
[(1,"abc")]

> parse (item +++ return 'd') "abc"
[('a',"bc")]

> parse (failure +++ return 'd') "abc"
[('d',"abc")]
```

Последовательность действий

Последовательность парсеров задается с помощью ключевого слова do

```
p :: Parser (Char,Char)
p  = do x <- item
        item
        y <- item
        return (x,y)
```

Контрольные задания

Создать примитивный парсер анализа строки.