Haskell. Домашнее задание №1

Горбунов Егор Алексеевич

28 сентября 2015 г.

1 Долг с пары

Задача №1.1 (функция or)

$$or\ x\ y = x\ true\ y$$

Если $x = true = \lambda ab.a$, то $or\ x\ y$ вернёт $true$ (т.е. x), иначе будет возвращён y, который и будет верным ответом, т.к. теперь всё от него и зависит (если $y = true$, то $or\ x\ y\ = true$ и с $false$ аналогично).

2 Примитивная рекурсия

Задача №2.1 (факториал)

Задача №2.2 (предыдущее число)

Задача №2.3 (меньше или равно)

Задача №2.4 (модуль разности)

3 Списки

$$nil = \lambda cx.x$$
$$cons\ a\ as = \lambda cx.c\ a\ (as\ c\ x)$$

Задача №3.1 ($isEmpty$)

$$isEmpty\ l = l\ (\lambda ht.false)\ true$$

Проверим для пустого списка nil: $isEmpty\ nil = nil\ (\lambda ht.false)\ true = \lambda cx.x\ (\lambda ht.false)\ true =_\beta true$. Ок. Любой непустой список $l = cons\ a_1(cons\ a_2(\ldots(cons\ nil)\ldots))$ представляется в виде:

$$\begin{aligned} l &= \lambda cx.c\ a_1\ (T\ c\ x) = && (1)\\ &= \lambda cx.c\ a_1\ (c\ a_2\ (T'\ c\ x))) = && (2)\\ &= \lambda cx.c\ a_1\ (c\ a_2\ (\ldots(c\ a_n\ (nil\ c\ x))\ldots)) = && (3)\\ &= \lambda cx.c\ a_1\ (c\ a_2(\ldots\ (c\ a_n\ x)\ldots) && (4)\end{aligned}$$

Ясно, что если вместо c подставить любую абстракцию с 2 связанными переменными, всегда возвращающую $false$, то $isEmpty\ l$ будет возвращать $false$. ■

Задача №3.2 ($head$)

$$head\ l = l\ true\ nil$$

Для nil: $head\ nil = nil\ true\ nil = (\lambda cx.x)\ true\ nil =_\beta nil$. Разумно.
Для непустого $l = cons\ a_1\ T$:

$$\begin{aligned} head\ l &= (\lambda cx.c\ a_1\ (c\ a_2(\ldots\ (c\ a_n\ x)\ldots))\ true\ nil =_\beta\\ &=_\beta true\ a_1\ (\ldots) =_\beta\\ &=_\beta a_1\end{aligned}$$

■

Задача №3.3 ($tail$)

Соберём хвост списка $l = [a_1, \ldots, a_{n-1}, a_n]$ начиная с конца. Пускай:

$$c\ e\ p\ \ = pair\ (snd\ p)\ (cons\ e\ (snd\ p)))$$

Тогда, например $c\ a_n\ (pair\ nil\ nil) = pair\ nil\ l'$, где l' — лист из элемента a_n, т.е. в первом элементе пары хранится хвост списка l' из одного элемента. Таким образом последовательные применения c при построении списка (см. формулу 4) приведут к тому, что после последнего применения c получится пара $pair\ tail\ l$. Таким образом ответ на задачу:

$$tail\ l = fst\ (l\ (\lambda ep.pair\ (snd\ p)\ (cons\ e\ (snd\ p)))\ (pair\ nil\ nil))$$

■

Задача №3.4 ($append$)

Аналогично операции $tail$ с одним изменением:

$$append\ l\ a = snd\ (l\ (\lambda ep.pair\ (snd\ p)\ (cons\ e\ (snd\ p)))\ (pair\ nil\ a))$$

Таким образом после первого применения c (см. формулу 4): $c\ a_n\ (pair\ nil\ a) = pair\ a\ (cons\ a_n\ a)$. Итого во втором элементе пары у нас будет записан исходный список l, с добавленным в конец элементом a. ■

4 Y-комбинатор

Задача №4.1 (про $F:\ \forall\ M(FM = F)$)

Равенство $F = FM$ β-эквивалентно следующему: $F = (\lambda fm.f)F$. Видим, что F — неподвижная точка терма $\lambda fm.f$. Y-комбинатор по определению таков, что: $\forall G:\ (YG = G(YG))$, но тогда $F = YG$, где $G = \lambda fm.f$, итого:

$$F = Y\ (\lambda fm.f)$$

■

Задача №4.2 (ещё один факториал)

Задача №4.3 (функция Аккермана)