Часть 2: элементарный уровень B

Правила: решаются задачи из пункта 'C' Решение задачи состоит из двух частей — (1) преобразование исходного кода с использованием неподвижной точки и (2) проверки, то есть собственно вычисления скомпонованного программного кода для заданной конечной последовательности.

A: Приемы программирования конечных последовательностей. Основная конструкция, которой будем пользоваться — это список, который может быть пустым, либо непустым. В последнем случае у него есть "голова" и "хвост", которые в свою очередь также могут быть списками. Над списками могут выполняться следующие операции:

$$\begin{array}{lcl} null & : & \text{список} \to \text{булевский}, \\ car & : & \text{непустой список} \to (\text{список} + \text{атом}), \\ cdr & : & \text{непустой список} \to \text{список}, \\ list & : & (\text{атом} + \text{список}) \to (\text{список} \to \text{список}). \end{array}$$

Эти операции связаны друг с другом следующим образом:

$$\begin{array}{rcl} null\,() & = & true, \\ null\,(list\,x\,y) & = & false, \\ car\,(list\,x\,y) & = & x, \\ cdr\,(list\,x\,y) & = & y, \\ list\,(car\,z)(cdr\,z) & = & z. \end{array}$$

Кроме того, примем сокращение:

$$list\,x\,y = x : y,$$

и поэтому для $n \geq 2$ воспользуемся соглашением об обозначении:

$$x_1,\ x_2,\ x_3, \ldots, x_n = x_1 : (x_2 : (x_3 : (\ldots\ x_n) : \ldots () \ldots)).$$

B: Методический пример. В качестве примера приведем соответствующие выкладки для функций $length$ (вычисление длины списка) и map (функционал, "распределяющий" вдоль списка действие функции-аргумента). При вычислении этих функций проявляются основные особенности рекурсивных вычислений над списками.

$length$–1. Для функции $length$ исходное определение

$$\begin{array}{rcl} length & = & \lambda x.if\ null\ x \\ & & \ then\ 0 \\ & & \ else\ 1 + length(cdr\ x), \end{array}$$

перепишем в виде:

$$\begin{array}{rcl} length & = & (\lambda f.\lambda x.if\ null\ x \\ & & \ then\ 0 \\ & & \ else\ 1 + f(cdr\ x))length. \end{array}$$

$length$–2. Отсюда следует, что

$$\begin{array}{rcl} length & = & Y(\lambda f.\lambda x.if\ null\ x \\ & & \ then\ 0 \\ & & \ else\ 1 + f(cdr\ x)). \end{array}$$

Тем самым желаемая комбинаторная характеристика получена.

$length$–3. Произведем проверку определения для списка длины 2, то есть возьмем $x = (a_1, a_2)$:

$$\begin{array}{rcl} length(a_1, a_2) & = & Y(\lambda f \lambda x.if\ null\ x \\ & & \ then\ 0 \\ & & \ else\ 1 + f(cdr\ x))(a_1, a_2) \\ & = & (\lambda f \lambda x.if\ null\ x \\ & & \ then\ 0 \\ & & \ else\ 1 + f(cdr\ x))(Y(\ldots))(a_1, a_2) \\ & = & if\ null\ (a_1, a_2)\ then\ 0\ else\ 1 + (Y(\ldots))(cdr\ (a_1, a_2)) \\ & = & 1 + (\lambda f \lambda x.if\ null\ x\ then\ 0\ else\ 1 + f(cdr\ x))(Y(\ldots))(a_2) \\ & = & 1 + if\ null\ (a_2)\ then\ 0\ else\ 1 + (Y(\ldots))nil \\ & = & 1 + 1(\lambda f \lambda x.if\ null\ x\ then\ 0\ else\ 1 + f(cdr\ x))(Y(\ldots))nil \\ & = & 1 + 1 + 0 = 2. \end{array}$$

map−1. *Для функции map исходное определение*

$$\begin{array}{rcl} map & = & \lambda f.\lambda x. \quad if\ null\ x \\ & & \qquad then\ () \\ & & \qquad else\ ((f(car\ x)) : (mapf(cdr\ x)) \end{array}$$

перепишем в виде:

$$\begin{array}{rcl} map & = & (\lambda m.\lambda f.\lambda x. \quad if\ null\ x \\ & & \qquad then\ () \\ & & \qquad else\ (f(car\ x)) : (mf(cdr\ x)))\ \ map. \end{array}$$

map−2. *Отсюда следует, что*

$$\begin{array}{rcl} map & = & Y(\lambda m.\lambda f.\lambda x. \quad if\ null\ x \\ & & \qquad then\ () \\ & & \qquad else\ (f(car\ x)) : (mf(cdr\ x))). \end{array}$$

map−3. *Проверка для* $f = square,\ x = (2,3)$*:*

$$\begin{array}{rcl} map\ square\ (2,3) & = & (\lambda m\lambda f\lambda x.if\ null\ x \\ & & \quad then\ () \\ & & \quad else\ (f(car\ x)) : (mf(cdr\ x)))(Y(\ldots))square\ (2,3) \\ & = & (square\ 2) : ((Y(\ldots))square\ (3)) \\ & = & (square\ 2) : ((\lambda m.\lambda f.\lambda x.\ldots)(Y(\ldots))square\ (3)) \\ & = & (square\ 2) : ((square\ 3) : ((Y(\ldots))square\ ()) \\ & = & (square\ 2) : ((square\ 3) : ()) \\ & = & (4,9). \end{array}$$

В данном случае символ ‘:’ принят для обозначения инфиксной формы функции $list$*, поэтому принимаем в качестве соглашения об обозначениях, что* $x : (y : (z : ())) = x, y, z = (x, y, z)$.

C: Варианты заданий.

C-1: Пользуясь функцией поиска неподвижной точки Y, выразить определения приводимых ниже функций:

$$\begin{array}{rlcl} 1) & length & = & \lambda x.if\ null\ x \\ & & & then\ 0 \\ & & & else\ 1 + length\ (cdr\ x), \\ 2) & sum & = & \lambda x.if\ null\ x \\ & & & then\ 0 \\ & & & else\ (car\ x) + sum\ (cdr\ x), \\ 3) & product & = & \lambda x.if\ null\ x \\ & & & then\ 1 \\ & & & else\ (car\ x)\ \times\ product\ (cdr\ x), \\ 4) & append & = & \lambda x.\lambda y.if\ null\ x \\ & & & then\ y \\ & & & else\ list\ (car\ x)\ (append\ (cdr\ x)\ y), \\ 5) & concat & = & \lambda x.if\ null\ x \\ & & & then\ () \\ & & & else\ append\ (car\ x)\ (concat(cdr\ x)), \\ 6) & map & = & \lambda f\lambda x.if\ null\ x \\ & & & then\ () \\ & & & else\ list\ (f\ (car\ x))\ (map\ f\ (cdr\ x)). \end{array}$$

C-2: Для примеров “обращения” к каждой из функций выполнить проверку.

$$\begin{array}{lrcl} 1) & length(a_1, a_2, a_3, a_4) & = & 4; \\ 2) & sum(1,2,3,4) & = & 10; \\ 3) & product(1,2,3,4) & = & 24; \\ 4) & append(1,2)(3,4,5) & = & (1,2,3,4,5); \\ 5) & concat((1,2),(3,4),()) & = & (1,2,3,4); \\ 6) & map\ square\ (1,2,3,4) & = & (1,4,9,16). \end{array}$$