

# Примитивно-рекурсивные функции

# Цель

- Всякий алгоритм для любых исходных данных однозначно определяет некоторый результат, если, конечно, этот результат существует для них.
- Поэтому каждому алгоритму соответствует функция, которая вычисляет этот результат.
- **Цель: дать формальное определение произвольного алгоритма как функции, принадлежащей некоторому специальным образом построенному классу функций**

# Конструктивный метод

- Опишем некоторый класс функций с помощью рекурсивных определений.
- все множество исследуемых объектов строится из конечного числа исходных объектов — базиса — с помощью простых операций, эффективная выполнимость которых достаточно очевидна.
- Операции над функциями в дальнейшем будем называть операторами.

# Простейшие базисные функции:

1) нуль-функция

$$0(x_1, x_2, \dots, x_n) = 0;$$

2) функция следования

$$s'(x) = x + 1;$$

3) функция выбора (или тождества)

$$I_m^n(x_1, x_2, \dots, x_n) = x_m \quad (m \leq n).$$

# Оператор суперпозиции

из функций

$$f(x_1, \dots, x_m),$$

$$f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$$

строит новую функцию

$$g(x_1, \dots, x_n) = f(f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)).$$

Обозначим оператор суперпозиции через  $S(f, f_1, \dots, f_m)$

или, с явным указанием числа аргументов функций,  $S_m^n(f, f_1, \dots, f_m)$ .

# Пример

- Благодаря наличию функций выбора, стандартное представление суперпозиции с точным определением числа аргументов у всех функций  $f_1, f_2, \dots, f_m$  не уменьшает возможностей самого оператора суперпозиции, т.к. любую подстановку функции в функцию можно выразить через оператор  $S$  и функцию  $I$ .

Для функций

$$h(x, y) = x + y, \quad f(x) = 3x - 1, \quad g(x, y, z) = x/(y + z)$$

выражение

$$h(f(y), g(x, y, z)) = 3y - 1 + x/(y + z)$$

можно представить в виде стандартной суперпозиции  $h(f(I^3_2(x, y, z)), g(x, y, z))$ .

# Оператор примитивной рекурсии

**Определение:** Функция  $f(x_1, \dots, x_n, y)$  получается оператором примитивной рекурсии из функций

$g(x_1, \dots, x_n)$  и  $h(x_1, \dots, x_n, y, z)$ , если

$$\begin{cases} f(x_1, \dots, x_n, 0) = g(x_1, \dots, x_n), \\ f(x_1, \dots, x_n, y + 1) = h(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y)) \end{cases}$$

- Краткая запись:  $f(x_1, \dots, x_n, y) = R_n(g, h)$ .
- Схема примитивной рекурсии имеет вид:

$$f(0) = a,$$

$$f(x + 1) = h(x, f(x)),$$

где  $a$  — константа, принадлежащая множеству  $\mathbb{N}$ .

# Вычисление рекурсивной функции

Для того, чтобы в некоторой точке  $(x_1, \dots, x_n, y)$  вычислить значение функции, заданной оператором примитивной рекурсии, можно выполнить следующую конечную последовательность вычислений:

$$f(x_1, \dots, x_n, 0) = g(x_1, \dots, x_n),$$

$$f(x_1, \dots, x_n, 1) = h(x_1, \dots, x_n, 0, f(x_1, \dots, x_n, 0)),$$

...

$$f(x_1, \dots, x_n, y) = h(x_1, \dots, x_n, y - 1, f(x_1, \dots, x_n, y - 1)).$$

Существенной характеристикой оператора примитивной рекурсии является такое его важнейшее свойство, что независимо от числа аргументов функции  $f$  рекурсия ведется только по одному аргументу, остальные аргументы зафиксированы на момент применения рекурсии.

```
int f(int x1, ..., int xn, int y) {  
    if (y==0) return g(x1,..., xn);  
    return h(x1,..., xn,y-1,f(x1,...,xn,y-1));  
}
```

```
int f(int x1, ..., int xn, int y) {  
    int t = g(x1, ..., xn);  
    for ( int i = 0; i < y; i++ )  
        t = h(x1, ..., xn, i, t);  
    return t;  
}
```



Оператор ПР:  $f(x_1, \dots, x_n, 0) = g(x_1, \dots, x_n)$

$f(x_1, \dots, x_n, y+1) = h(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y))$

Наши функции:  $g(x) = x+2$

$h(x, y, z) = x+y+z$

Сначала определим, сколько параметров у НАШЕЙ функции  $f$ . Их на 1 больше, чем у функции  $g$  (см. оператор ПР). У нашей  $g$  – 1 параметр, значит у  $f$  будет 2 параметра!

$f(x, 0) = g(x) = x+2$

$f(x, y+1) = h(x, y, f(x, y)) = x+y+f(x, y)$

ЭТО ФОРМУЛЫ ДЛЯ РЕКУРСИВНОГО ВЫЧИСЛЕНИЯ  $f$ ! САМО ВЫЧИСЛЕНИЕ:

$f(x, 2) = x+1+f(x, 1) = x+1+x+0+f(x, 0) = 2x+1+x+2 = 3x+3$

РЕКУРСИВНАЯ

ФУНКЦИЯ:

```
int f(int x, int y)
{
    if (y==0) return x+2;
    return x+y-1+f(x, y-1);
}
```

НЕРЕКУРСИВНАЯ ФУНКЦИЯ:

```
int f1(int x, int y)
{ int q, i;
  q=x+2;
  for(i=0; i<y; i++)
    q=x+i+q; // h(x, i, q)
  return q;
}
```

# Определение ПРФ

**Функция называется примитивно – рекурсивной, если она может быть получена из простейших функций с помощью конечного числа операторов суперпозиции и примитивной рекурсии.**

Данное определение эквивалентно рекурсивному заданию множества функций.

- простейшие функции являются примитивно-рекурсивными по определению.
- Если некоторые функции являются примитивно–рекурсивными, то в результате применения к ним одного из операторов суперпозиции или примитивной рекурсии получаем новые примитивно–рекурсивные функции.
- Конечное число операторов суперпозиции или примитивной рекурсии, которые применяются при построении примитивно–рекурсивных функций, гарантируют завершение указанного рекурсивного процесса.

## Рекурсивный вариант определения

Функция называется примитивно–рекурсивной, если

- а) она является простейшей,
- б) она получена из примитивно–рекурсивных функций с помощью оператора суперпозиции;
- в) она получена из примитивно–рекурсивных функций с помощью оператора примитивной рекурсии.

Других примитивно–рекурсивных функций нет.

# Теорема 1

Примитивно рекурсивные функции всюду определены.

Доказательство.

В соответствии с рекурсивным вариантом определения примитивно–рекурсивной функции достаточно рассмотреть три шага доказательства.

Очевидно, что простейшие функции всюду определены.

Из определенных всюду функций с помощью оператора суперпозиции можно получить только всюду определенные функции.

Из определенных всюду функций в соответствии с алгоритмом вычисления функций, заданных оператором примитивной рекурсии, также получаем всюду определенные функции.

Ч.т.д.

# Способы доказательства ПРФ

- показать, что заданная функция является простейшей
- показать, что заданная функция построена из примитивно–рекурсивных функций с помощью оператора суперпозиции
- показать, что заданная функция построена из примитивно–рекурсивных функций с помощью оператора примитивной рекурсии.

# Примеры ПРФ.

## Доказательство ПРФ по определению

1)  $f(x)=k$  (функция-константа)

$f(x)=s'(s'(\dots s'(0(x))))$ , если применить функцию следования конечное число раз, равное константе  $k$ .

2)  $f(x)=x$

Первый способ доказательства:  $f(x)=I^1_1(x)$

Второй способ доказательства:

$f(0)=0=const$  – что  $const$  – ПРФ – доказано

$f(x+1)=x+1=s'(x)=s'(f(x))=I^2_2(x, s'(f(x)))$  – получено с помощью конечного числа ПРФ, операторов ПР и суперпозиции

3)  $f(x,y)=x+y$

$f(x,0)=x$  – ПРФ

$f(x,y+1)=x+y+1=f(x,y)+1=s'(f(x,y))=I^3_3(x, y, s'(f(x,y)))$

# Расширение набора ПРФ

**Определение:** Функция  $f(x_1, \dots, x_n, y)$  получается оператором примитивной рекурсии из функций

$g(x_1, \dots, x_n)$  и  $h(x_1, \dots, x_n, y, z)$ , если

$$\begin{cases} f(x_1, \dots, x_n, 0) = g(x_1, \dots, x_n), \\ f(x_1, \dots, x_n, y + 1) = h(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y)) \end{cases}$$

- Краткая запись:  $f(x_1, \dots, x_n, y) = R_n(g, h)$ .
- Схема примитивной рекурсии имеет вид:

$$f(0) = a,$$

$$f(x + 1) = h(x, f(x)),$$

**Определение:** Функция  $f(x_1, \dots, x_n, y)$  получается оператором примитивной рекурсии из функций

$g(x_1, \dots, x_n)$  и  $h(x_1, \dots, x_n, y, z)$ , если

$$\begin{cases} f(x_1, \dots, x_n, 0) = g(x_1, \dots, x_n), \\ f(x_1, \dots, x_n, y + 1) = h(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y)) \end{cases}$$

- Краткая запись:  $f(x_1, \dots, x_n, y) = R_n(g, h)$ .
- Схема примитивной рекурсии имеет вид:

$$f(0) = a,$$

$$f(x + 1) = h(x, f(x)),$$

где  $a$  — константа, принадлежащая множеству  $N$ .<sup>14</sup>

# Предикаты и примитивно-рекурсивные операторы

**Определение:** Функция  $f(x_1, \dots, x_n, y)$  получается оператором примитивной рекурсии из функций  $g(x_1, \dots, x_n)$  и  $h(x_1, \dots, x_n, y, z)$ , если

$$\begin{cases} f(x_1, \dots, x_n, 0) = g(x_1, \dots, x_n), \\ f(x_1, \dots, x_n, y + 1) = h(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y)) \end{cases}$$

- Краткая запись:  $f(x_1, \dots, x_n, y) = R_n(g, h)$ .
- Схема примитивной рекурсии имеет вид:

$$f(0) = a,$$

$$f(x + 1) = h(x, f(x)),$$

где  $a$  — константа, принадлежащая множеству  $\mathbb{N}$ .

# Операторы суммирования и умножения

$f(x_1, \dots, x_n, y)$  – функция от  $(n+1)$ -й переменной. Операции суммирования и умножения по переменной  $y$  с пределом  $z$  – это операторы, которые из функции  $f(x_1, \dots, x_n, y)$  порождают новые функции:

$$q(x_1, \dots, x_n, z) = \sum_{y < z} f(x_1, \dots, x_n, y),$$

$$h(x_1, \dots, x_n, z) = \prod_{y < z} f(x_1, \dots, x_n, y).$$

Они примитивно-рекурсивны (если  $f$  – примитивно-рекурсивна) в силу следующих соотношений:

$$g(x_1, \dots, x_n, 0) = 0 \text{ (ПРФ по определению);}$$

$$g(x_1, \dots, x_n, z+1) = g(x_1, \dots, x_n, z) + f(x_1, \dots, x_n, z);$$

$$h(x_1, \dots, x_n, 0) = 1 \text{ (по определению);}$$

$$h(x_1, \dots, x_n, z+1) = h(x_1, \dots, x_n, z) \cdot f(x_1, \dots, x_n, z).$$

Пример: количество делителей числа  $x$ :  $\tau(x) \stackrel{\text{sg}}{=} \left\{ \frac{x}{i} \right\}$