

Языки программирования

silvia.lesnaia

28 октября 2025 г.

02.09.25

1 Введение

Standard VML

Kastrell

List

Ruby

2 Императивное программирование

Предполагает что программе отдают команды, и компьютер последовательно выполняет команды.

Пример: оператор присваивания

В высокоуровневом языке сложение двух чисел является операцией, когда же в низкоуровневым же это будет являться оператором.

Нечистые функции получают выражение с побочным эффектом

Чистое выражение(функция) получает выражение без побочного эффекта

Первая парадигма программирования - Функциональная парадигма программирования

Является одной из разновидностей декларативного программирования

Есть декларированное программирование и императивное программирования

К декларированному программированию относятся: Функциональное программирование, логическое программирование.

Любая функция преподлагает некоторые аргументы $f S_1 * S_2 * S_3 \dots * S_4 \rightarrow S$

Функцию называют **ЧИСТОЙ** если ее результат зависит от ее параметров, а не от внешней среды, кроме того при вычисление результата, функция не оставляет никаких побочных эффектов.

3 Standard ML

Пример:

```
fun f (a : int, b : int) : int =  
 2*a+b
```

fun <имя функции> (аргументы) : тип результата = выражение

λ - исчисление (это формализм) предполагает, что

Используется для тезиса Черча

Пример 1:

```
fun f1(a : int, b : int, c : int) : int = = a*a*b*c
```

Пример 2:

```
fun square(a: int) : int = a*a
```

```
fun f1 (a : int, b : int, c : int) = square(a)+square(b)+square(b)
```

```
let  
  декларация  
in  
  выражение  
end
```

```
let  
  val dx21 = x2-x1  
  val dy21 = y2-y1  
  val dx31 = x3-x1  
  val dy31 = y3-y1  
  val dy23 = y2-y3  
  val dx23 = x2-x3  
in  
  dx21*dy21*dx31*dy31*dx23*dy23
```

НИЧТО НЕ ДОЛЖНО ВЫЧИСЛЯТЬСЯ ДВАЖДЫ

НЕ ДОЛЖНО БЫТЬ НЕ ОБОСНОВАННЫХ ОБОЗНАЧЕНИЙ И ИСПОЛЬЗОВАТЬ ХОТЯ БЫ ДВА РАЗА

Затенение

```
val a = 5  a ← 5  
val a = 17 a ← 17
```

```
val a=5  
val b = let(b,28)  
        val a = 17  
        val b = 17  
in  
  a + b → 24
```

```
val a = if b > 25 then 45 - b  
else 45 + b
```

Является язык строго типизированным, это означает что у выражение должен быть определен тип, которое вернет выражение.

Конструкции цикла нет, будут использоваться рекурсия

16.09.25

Продолжение рекурсивных алгоритмов

Значение unit - () тип значение

Ввод:

```
val a = 5  
val b = 7  
val c = 9
```

Вывод:

```
val a = 5:unit  
val b = 7:unit  
val c = 9:unit  
val d = () unit
```

Если нажать f5, то это будет восприниматься как use $\Rightarrow ()$

fun f(l: a' list): 'a = hd(tl(tl l)) // функция возвращает третий элемент, если конечно он есть

Полиморфные функции - может принимать различные типы данных, наличие этих функций называется полиморфизмом. В Standard ML есть полиморфизм по типам данным.

Сравниваем два элемента списка

```
fun f''(l:           ):      =  
if null l then []  
else if null (tl l) then l  
else if hd l = (tl l)  
then f''(tl l)  
else if hd l :: f''(tl l )
```

КАК ДЕЛАТЬ НЕ НАДО

```
[1,2,3,4,5]⇒ [5,4,3,2,1]  
fun reverse (l:a list):a list =  
if null l then l
```

КАК НАДО

```
fun reverse (l:'a list) :'a list =  
let  
  fun revhelper( l:'a list, ace : 'a list)  
    :'a list  
    if null l then ace  
    else revhelper(tl l, hd l:: ace)  
  in  
  revhelper(l,[])  
end
```

23.09.25

4 Контейнеры. Объявления собственных типов данных

Контейнер

int	real
int list	[1,2,3]
real option	SOME 3.15
	NONE

date = int * int * int

1. Определение синонимов типов данных

person = name : string, age : int, num : int, salary : real, l : 'a list
v : int person

2. Определение контейнеров

datatype эта конструкция определяет новый тип данных

color имя типа

datatype color = RED | BLUE | WHITE
| ORANGE | BLACK
конструкторы

5 Механизм сопоставления с шаблоном

Сопоставление с образцом или размещение по шаблону.

d : date

val (day, month, year) = d

val (a1,a2,a3) = (b1,b2,b3)

Шаблон может содержать:

конструкторы

идентификаторы

джокер / wildcard или же нижнее подчеркивание

<шаблон> as

07.10.25

6 Динамическое окружение

Динамическое окружение это совокупность занчений связных имен в момент выполнения конкретной инструкции программного кода.

Для приера мы будем его представлять как список связанных значений. Каждый элемент пары, первый элемент имя, второй заненчение.

```
[(имя,значение),.....]  
[(a,25),(6,13),(c,28), (a,13)]  
a+b-c  
25+13-78
```

6.1 Формирование динамического окружения

1. val a =13
2. val b = 11
3. val c =8
4. val a = 25
5. val d = let val a = 2
 val b =8
 in a+b -c
 end
6. fun f(a,x) = a+x-c
7. fun g(a,x) = if a>0 then 2+f(a-2,x*2) else 1
8. val e = f(a,b)
9. val h = g(7,3)

Парметры которые используются при описании функции называются формальными параметрами.

Параметры которые используются при вызове функции называются фактическими параметрами. В строке 8 фактические параметры.

Замыкание функции это союзокупность определений функции вместе с динамическим окружением, в котором она была определена

Статичекое окружение при приведении к анализу программы, при выполнении оно не нужно.

14.10.25

7 Модель нормальных вычислений

тут пикча

8 Карированные функции

До этого мы функции определяли как:

тут пикча

Сигнатурой функции называется сочетания типов параметров с результатом функции, по сути типа значения функции.

тут новая пикча

Разница между сигнатурой g' и g нету разнице.

тут пикча

Карированиe предполагает, что если у нас есть функции от трех параметров кортежа, то вы получаете функцию которая выдаст первый параметр, второй, третий, а потом кортеж.

тут пикча

Обратный процесс

Декарированиe, или анкорированиe

21.10.25

```
fun zip x= x*25  
val a =f  
fun g1 x =fx
```

- Двумерный синтаксис
- Ленивые вычисления
- Меморизация - сохранение памяти

28.10.25

9 Бесконечные списки

СГНФ - вычислена с точностью до конструктора.

SML	Haskell
datatype	=
'a list	data
:: []	: []