

# Функциональное программирование

---

## ЛЕКЦИЯ 2. ТЕМА 2. БЕСТИПОВЫЕ АРИФМЕТИЧЕСКИЕ ВЫРАЖЕНИЯ

# ПЛАН ЛЕКЦИИ

---

1. Термы.
2. Синтаксис.
3. Индукция на термах.
4. Семантические стили.

# НЕОБХОДИМОСТЬ ИЗУЧЕНИЯ ТЕОРЕТИЧЕСКИХ ОСНОВ ФП

- Базовые характеристики языков программирования (во многом определяются системой типов в ФП)
- Четкие, ясные, точные механизмы описания синтаксиса и семантики программ

Пример: язык Haskell

Поскольку Haskell представляет собой чисто функциональный язык,

- все вычисления осуществляются с помощью исчисления выражений (синтаксических термов),
- производящих значения (абстрактные сущности, которые мы рассматриваем как ответы).

Каждое значение имеет связанный с ним тип (на интуитивном уровне мы можем рассматривать типы как множества значений).

Налицо построенная строгая система типов закономерности и свойства которой необходимо учитывать, уметь формально обращаться с ней

# ГРАММАТИКА ПСЕВДО-ЯЗЫКА

t ::=                           {- термы: -}  
true                       {- константа «истина» -}  
false                      {- константа «ложь» -}  
if t then t else t    {- условное выражение -}  
0                        {- константа «ноль» -}  
succ t                 {- следующее число -}  
pred t                 {- предыдущее число -}  
iszero t               {- проверка на ноль -}

# ОПРЕДЕЛЕНИЯ И ПОЯСНЕНИЯ

$t$  (с  $r$  до  $u$ ) – метаварiable (служит для описания),  
служит заместителем какого-либо термина

**Выражением** будут называться любые синтаксические  
объекты

**Термы** – выражения, которые представляют собой  
вычисления

**Программа** – терм, построенный на основе конструкций  
соответствующей приведенной грамматике

**Числа** – конструкции вида  $\text{succ}(\text{succ}(\text{succ}(0)))$   
**Результаты вычисления** 0 либо булевы константы, либо  
числа - это все термы

Такие термы будем называть **значениями**

# ДРУГИЕ ОПРЕДЕЛЕНИЯ СИНТАКСИСА

Определение 1 [термы через индукцию]

Множество термов – это наименьшее множество  $T$  такое , что

1.  $\{\text{true}; \text{false}; 0\} \subseteq T$  ;

2. если  $t_1 \in T$  , то  $\{\text{succ } t_1; \text{pred } t_1; \text{iszero } t_1\} \subseteq T$  ;

3. если  $t_1 \in T$  ,  $t_2 \in T$  ,  $t_3 \in T$  , то if  $t_1$  then  $t_2$  else  $t_3 \in T$  .

# ДРУГИЕ ОПРЕДЕЛЕНИЯ СИНТАКСИСА

Определение 2 [термы через правила вывода]: множество термов определяется следующим образом

$$\begin{array}{c} true \in \mathcal{T} \\ \hline succ\ t_1 \in \mathcal{T} \end{array} \quad \begin{array}{c} false \in \mathcal{T} \\ \hline pred\ t_1 \in \mathcal{T} \end{array} \quad \begin{array}{c} 0 \in \mathcal{T} \\ \hline iszero\ t_1 \in \mathcal{T} \end{array}$$
$$\frac{t_1 \in \mathcal{T} \quad t_2 \in \mathcal{T} \quad t_3 \in \mathcal{T}}{if\ t_1\ then\ t_2\ else\ t_3 \in \mathcal{T}}$$

# ДРУГИЕ ОПРЕДЕЛЕНИЯ СИНТАКСИСА

□ Определение 3 [термы через правила вывода]:  
множество термов определяется объединением  
множеств  $S = \bigcup_i S_i$  для каждого  $i \in \mathbb{N}$ , где

$$S_0 = \emptyset$$

$$S_{i+1} = \{\text{true}, \text{false}, 0\}$$

$$\cup \{\text{succ } t_1, \text{pred } t_1, \text{iszero } t_1 \mid t_1 \in S_i\}$$

$$\cup \{\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \mid t_1, t_2, t_3 \in S_i\}$$

$$T = S !!$$



## ДОМАШНЕЕ ЗАДАНИЕ:

---

1. Сколько элементов содержит  $S_3$ ?
2. Покажите  $\forall i, S_i \subseteq S_{i+1}$
3. Покажите, что  $S$  - наименьшее множество

# T=S !!

---

Проверим!!!!

Для  $S$  должны выполняться условия

1.  $\{\text{true}; \text{false}; 0\} \sqsubseteq S$ ;

2. если  $t1 \in S$ , то  $\{\text{succ } t1; \text{pred } t1; \text{iszero } t1\} \sqsubseteq S$ ;

3. если  $t1 \in S, t2 \in S, t3 \in S$ , то if  $t1$  then  $t2$  else  $t3 \in S$ .

# $T=S !!$

---

Докажем!!!

1. Пусть  $S'$  удовлетворяет условиям наименьшего множества
2. При помощи полной индукции по  $i$  докажем что  $S_i \subseteq S'$ 
  - Для случая  $i=0$
  - Для случая  $i=j+1>0$ , пусть  $S_j \subseteq S'$

# ИНДУКЦИЯ НА ТЕРМАХ

---

Из определения 1 если  $t \in T$

1.  $t$  является константой;

2.  $t$  результатом  $\text{succ } t_1$ ;  $\text{pred } t_1$ ;  $\text{iszero } t_1$   
 $t_1 < t$ ;

3.  $t$  результатом  $\text{if } t_1 \text{ then } t_2 \text{ else } t_3$   $t_1, t_2, t_3$   
 $< t$

# ОПРЕДЕЛЕНИЯ

**Определение 1** Множество констант, встречающихся в терме  $t$  (записывается  $Consts(t)$ ), определяется так:

$$\begin{aligned} Consts(true) &= \{true\} \\ Consts(false) &= \{false\} \\ Consts(0) &= \{0\} \\ Consts(succ\ t_1) &= Consts(t_1) \\ Consts(pred\ t_1) &= Consts(t_1) \\ Consts(iszero\ t_1) &= Consts(t_1) \\ Consts(if\ t_1\ then\ t_2\ else\ t_3) &= Consts(t_1) \cup Consts(t_2) \cup Consts(t_3) \end{aligned}$$

# ОПРЕДЕЛЕНИЯ

**Определение 2**    Размер (*size*) терма  $t$  (записывается  $size(t)$ ) определяется так:

$$\begin{aligned} size(true) &= 1 \\ size(false) &= 1 \\ size(0) &= 1 \\ size(succ\ t_1) &= size(t_1) + 1 \\ size(pred\ t_1) &= size(t_1) + 1 \\ size(iszero\ t_1) &= size(t_1) + 1 \\ size(if\ t_1\ then\ t_2\ else\ t_3) &= size(t_1) + size(t_2) + size(t_3) + 1 \end{aligned}$$

# ОПРЕДЕЛЕНИЯ

## Определение 3

глубина (*depth*) терма *t* (записывается *depth(t)*)

определяется так:

$$\begin{aligned} \text{depth}(\text{true}) &= 1 \\ \text{depth}(\text{false}) &= 1 \\ \text{depth}(0) &= 1 \\ \text{depth}(\text{succ } t_1) &= \text{depth}(t_1) + 1 \\ \text{depth}(\text{pred } t_1) &= \text{depth}(t_1) + 1 \\ \text{depth}(\text{iszero } t_1) &= \text{depth}(t_1) + 1 \\ \text{depth}(\text{if } t_1 \text{ then } t_2 \text{ else } t_3) &= \max(\text{depth}(t_1), \text{depth}(t_2), \text{depth}(t_3)) + 1 \end{aligned}$$

# ПРИМЕР СООТНОШЕНИЯ МЕЖДУ ЧИСЛОМ КОНСТАНТ В ТЕРМЕ И ЕГО РАЗМЕРОМ

**Лемма 1** Число различных констант в терме  $t$  не больше его размера  
(т. е.,  $|Consts(t)| \leq size(t)$ ).



# ПРИНЦИПЫ ИНДУКЦИИ ПО ТЕРМАМ

## Индукция по глубине

Пусть  $P$  — предикат,  
определенный на множестве термов.

Если для каждого терма  $s$ ,

предполагая, что  $P(t)$  для всех  $t$ , таких, что  
 $depth(t) < depth(s)$ ,  
можно доказать, что  $P(s)$ ,

то  $P(s)$  выполняется для всех  $s$ .

# ПРИНЦИПЫ ИНДУКЦИИ ПО ТЕРМАМ

## Индукция по размеру

Пусть  $P$  — предикат,  
определенный на множестве термов.

Если для каждого терма  $s$ ,  
предполагая, что  $P(t)$  для всех  $t$ , таких, что  
 $size(t) < size(s)$ ,  
можно доказать, что  $P(s)$ ,  
то  $P(s)$  выполняется для всех  $s$ .

# ПРИНЦИПЫ ИНДУКЦИИ ПО ТЕРМАМ

## Структурная индукция

Пусть  $P$  — предикат,  
определенный на множестве термов.

Если для каждого терма  $s$ ,  
предполагая, что  $P(t)$  для всех непосредствен-  
ных подтермов  $s$ , можно доказать, что  $P(s)$ ,  
то  $P(s)$  выполняется для всех  $s$ .

# О СИНТАКСИСЕ ПСЕВДО-ЯЗЫКА (ПОВТОР)

t ::=

- {- термы: -}
- true           {- константа «истина» -}
- false           {- константа «ложь» -}
- if t then t else t   {- условное выражение -}
- 0               {- константа «ноль» -}
- succ t           {- следующее число -}
- pred t           {- предыдущее число -}
- iszero t          {- проверка на ноль -}

# О СИНТАКСИСЕ ПСЕВДО-ЯЗЫКА (ПОВТОР)

Множество термов – это наименьшее множество  $T$  такое ,  
что

1.  $\{\text{true}; \text{false}; 0\} \subseteq T$  ;
2. если  $t_1 \in T$  , то  $\{\text{succ } t_1; \text{pred } t_1; \text{iszero } t_1\} \subseteq T$  ;
3. если  $t_1 \in T$  ,  $t_2 \in T$  ,  $t_3 \in T$  , то  $\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \in T$  .

Множество термов определяется объединением множеств  
 $S = \bigcup_i S_i$  для каждого  $i \in \mathbb{N}$ , где

$$S_0 = \emptyset$$

$$S_{i+1} = \{\text{true}, \text{false}, 0\}$$

$$\cup \{\text{succ } t_1, \text{pred } t_1, \text{iszero } t_1 \mid t_1 \in S_i\}$$

$$\cup \{\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \mid t_1, t_2, t_3 \in S_i\}$$

# СЕМАНТИЧЕСКИЕ СТИЛИ

---

Для определения смысла программы необходимо её математический эквивалент, а для его построения нам надо в точности знать свойства языка, на котором написана программа.

Проблема сводится к отысканию способа построения математических эквивалентов всех конструкций языка - к формализации его семантики.

# СЕМАНТИЧЕСКИЕ СТИЛИ

---

Семантика языка — это смысловое значение слов. В программировании — начальное смысловое значение операторов, основных конструкций языка и т. п.

1) `i=0; while(i<5){i++;}`

2) `i=0; do{i++;}while(i<4);`

# СЕМАНТИЧЕСКИЕ СТИЛИ

---

Различные подходы к формализации семантики:

- Операционная семантика
- Денотационная семантика
- Аксиоматическая семантика
- Интерпретационная семантика
- Трансляционная семантика
- Трансформационная семантика



# СЕМАНТИЧЕСКИЕ СТИЛИ

---

## Операционная семантика

Специфицирует поведение языка определяя простую абстрактную машину (использует термины языка, а не машинные команды)

Состояние машины – терм

Поведение определяется функцией перехода.

Смысл термина  $t$  – конечное состояние

# СЕМАНТИЧЕСКИЕ СТИЛИ

---

## **Трансляционная семантика**

Описание операционной семантики конструкций в терминах языков программирования высокого уровня.

С помощью этого способа можно изучать язык, схожий с уже известным программисту.

# СЕМАНТИЧЕСКИЕ СТИЛИ

---

## **Трансформационная семантика**

Описание операционной семантики конструкций языка в терминах этого же языка. Трансформационная семантика является основой метапрограммирования.

# СЕМАНТИЧЕСКИЕ СТИЛИ

---

## Денотационная семантика

Смысл термина - математические объекты  
(число, функция, их величины)

Построение семантики:

- 1) нахождение набора семантических доменов (СД)
- 2) Определение функции интерпретации – (соответствие СД и термов)

# СЕМАНТИЧЕСКИЕ СТИЛИ

## Аксиоматическая семантика

Семантика каждой синтаксической конструкции языка определяется как некий набор аксиом или правил вывода, который можно использовать для вывода результатов выполнения этой конструкции.

Чтобы понять смысл всей программы, эти аксиомы и правила вывода следует использовать так же, как при доказательстве обычных математических теорем.

### Смысл термина, то что можно о нем доказать.

Когда программа выполнена, получаем доказательство - что вычисленные результаты удовлетворяют необходимым ограничениям на их значения относительно входных значений. То есть, доказано, что выходные данные представляют значения соответствующей функции, вычисленной по значениям входных данных.

# СЕМАНТИЧЕСКИЕ СТИЛИ

---

## **Интерпретационная семантика**

описание операционной семантики конструкций в терминах языков программирования низкого уровня (язык ассемблера, машинный код).

Позволяет выявлять медленно выполняемые участки программы, и зачастую используется в целях оптимизации кода программ.

# ВЫЧИСЛЕНИЯ

В (бестиповое)

Синтаксис

$t ::=$  *термы:*  
`true` константа «истина»  
`false` константа «ложь»  
`if t then t else t` условное  
выражение

$v ::=$  *значения:*  
`true` константа «истина»  
`false` константа «ложь»

Вычисление

$t \rightarrow t'$

`if true then t2 else t3 → t2`

(E-IFTRUE)

`if false then t2 else t3 → t3`

(E-IFFALSE)

$$\frac{t_1 \rightarrow t'_1}{\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \rightarrow \text{if } t'_1 \text{ then } t_2 \text{ else } t_3}$$

(E-IF)

# ВЫЧИСЛЕНИЯ (СЛУЧАЙ ТОЛЬКО БУЛЕВЫХ ВЫРАЖЕНИЙ)

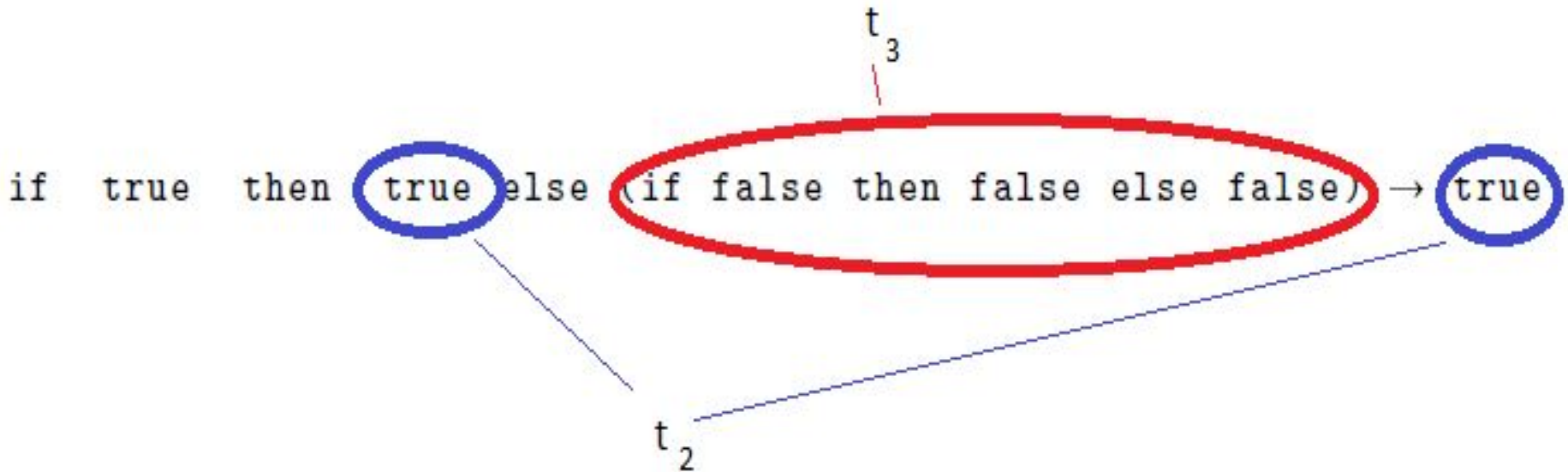
---

- 1)  $t \rightarrow t'$  понимается как  $t$  за 1 шаг вычисляется как  $t'$
- 2) Константы ни во что не вычисляются
- 3)  $E\text{-IfTrue}$  и  $E\text{-IfFalse}$  – рабочие правила
- 4)  $E\text{-If}$  – правило соответствия
- 5) Определение: Экземпляр правила вывода получается при замене каждой метавариабельной в заключении правила и во всех его предпосылках



# ВЫЧИСЛЕНИЯ

## Пример



# ВЫЧИСЛЕНИЯ (случай только булевых выражений)

---

- 1) *Определение: Правило выполняется на отношении, если для каждого экземпляра правила его заключение является элементом отношения, либо одна из его предпосылок не является таковой*

# ВЫЧИСЛЕНИЯ (случай только булевых выражений)

---

2. *Определение: Одношаговое отношение вычисления  $\rightarrow$  есть наименьшее бинарное отношение на термах, на котором выполняются все три правила*
- Если пара  $(t \rightarrow t')$  является элементом отношения вычисления, то говорят, что утверждение о вычислении  $t \rightarrow t'$  выводимо*

# ВЫЧИСЛЕНИЯ

## Пример

```
s  $\stackrel{\text{def}}{=}$  if true then false else false  
t  $\stackrel{\text{def}}{=}$  if s then true else true  
u  $\stackrel{\text{def}}{=}$  if false then true else true
```

```
if t then false else false  $\rightarrow$  if u then false else false
```

$$\frac{\frac{\frac{}{s \rightarrow \text{false}}{\text{E-IFTRUE}}}{t \rightarrow u}{\text{E-IF}}}{\text{if t then false else false} \rightarrow \text{if u then false else false}}{\text{E-IF}}$$

# ВЫЧИСЛЕНИЯ (СВОЙСТВА ОТНОШЕНИЯ ВЫЧИСЛЕНИЯ)

---

Теорема [теорема о детерминированности  
одношагового вычисления]

*Если  $t \rightarrow t'$  и  $t \rightarrow t''$ , то  $t' = t''$ .*

# ВЫЧИСЛЕНИЯ (СВОЙСТВА КОНЕЧНОГО СОСТОЯНИЯ)

---

Результат вычисления – конечное состояние

**Определение 3** Терм  $t$  находится в нормальной форме если к нему не применимо никакое правило вычисления ( ~~$\exists t', m.c. t \rightarrow t'$~~ )

**Теорема 2.** Всякое значение находится в нормальной форме.

**Теорема 3.** Всякая нормальная форма есть значение

# ВЫЧИСЛЕНИЯ

---

Определение. Отношение многошагового вычисления  $\rightarrow^*$  это наименьшее отношение, т.ч.

(1) Если  $t \rightarrow t'$ , то  $t \rightarrow^* t'$

(2) Для всех  $t$  выполняется  $t \rightarrow^* t$

(3) Если  $t \rightarrow^* t' \text{ и } t' \rightarrow^* t''$ , то  $t \rightarrow^* t''$

# ВЫЧИСЛЕНИЯ

---

Теорема [теорема о единственности нормальных форм]

Если  $t \rightarrow^* u$  и  $t \rightarrow^* u'$  нормальные формы, то  $u = u'$



# ВЫЧИСЛЕНИЯ

---

Каждый терм можно вычислить и получить значение

Теорема [завершение вычислений]

Для каждого терма  $t$  существует нормальная форма  $t'$  такая что  $t \rightarrow^* t'$

# ВЫЧИСЛЕНИЯ

---

Домашнее задание: упражнение 3.5.13, стр.  
59

# ВЫЧИСЛЕНИЯ (АРИФМЕТИЧЕСКИЕ ВЫРАЖЕНИЯ)

## Новые синтаксические формы

<b>t</b>	::= ...	термы:
0		константа ноль
succ t		следующее число
pred t		предыдущее число
iszero t		проверка на ноль
<b>v</b>	::= ...	значения:
nv		числовое значение
<b>nv</b>	::= ...	числовые значения:
0		нулевое значение
succ nv		значение-последователь

## Новые правила вычисления

$$t \rightarrow t'$$

$$\frac{t_1 \rightarrow t'_1}{\text{succ } t_1 \rightarrow \text{succ } t'_1} \quad (\text{E-SUCC})$$

$$\text{pred } 0 \rightarrow 0 \quad (\text{E-PREDZERO})$$

$$\text{pred } (\text{succ } nv_1) \rightarrow nv_1 \quad (\text{E-PREDSUCC})$$

$$\frac{t_1 \rightarrow t'_1}{\text{pred } t_1 \rightarrow \text{pred } t'_1} \quad (\text{E-PRED})$$

$$\text{iszero } 0 \rightarrow \text{true} \quad (\text{E-ISZEROZERO})$$

$$\text{iszero } (\text{succ } nv_1) \rightarrow \text{false} \quad (\text{E-ISZEROSUCC})$$

$$\frac{t_1 \rightarrow t'_1}{\text{iszero } t_1 \rightarrow \text{iszero } t'_1} \quad (\text{E-ISZERO})$$

# ВЫЧИСЛЕНИЯ (АРИФМЕТИЧЕСКИЕ ВЫРАЖЕНИЯ)

---

- Определение. Терм если он находится в нормальной форме, но не является значением называется тупиковым термом

Пример: *succ false*

# ВЫЧИСЛЕНИЯ

---

Домашнее задание: упражнение  
3.5.16-3.5.18, стр. 61,62

Кто **УСПЕШНО** выступит в понедельник – 5  
дополнительных баллов на экзамене