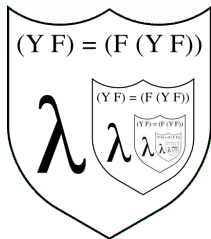


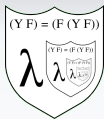
# Соответствие Карри–Ховарда. Лекция вторая

---



А. Н. Непейвода  
ИПС им. А.К. Айламазяна РАН,  
МГТУ им. Н.Э. Баумана

*MathClub,*  
*25 февраля 2022, Иннополис*



## Аксиомы комбинаторной логики

$$S(KK) = S(S(KS)(S(KK)(S(KS)K)))(KK)$$

$$S(KS)(S(KK)) = S(KK)(S(S(KS)(S(KK)I))(KI))$$

$$S(K(S(KS)))(S(KS)(S(KS)))$$

$$= S(S(KS)(S(KK)(S(KS)(S(K(S(KS)))S))))(KS)$$

$$I = S(S(KS)K)(KI)$$



## Упражнение

Известно, что композиция  $\lambda x y z. x (y z)$  — это `fmap` для функтора `Reader`. Выразить её в бесточечном виде по алгоритму и по логике.



## Упражнение

Известно, что композиция  $\lambda x y z. x (y z)$  — это `fmap` для функтора `Reader`. Выразить её в бесточечном виде по алгоритму и по логике.

```
newtype MyRead r a = Rd run :: r -> a
instance Applicative (MyRead r) where
  pure v = Rd (\_ -> v)
  Rd f <*> Rd g = Rd (\x -> f x (g x))
```



## Упражнение

Известно, что композиция  $\lambda x y z. x (y z)$  — это `fmap` для функтора `Reader`. Выразить её в бесточечном виде по алгоритму и по логике.

```
newtype MyRead r a = Rd run :: r -> a
instance Applicative (MyRead r) where
  pure v = Rd (\_ -> v)
  Rd f <*> Rd g = Rd (\x -> f x (g x))
```

```
fmap f (Rd x)
  = (Rd ((pure (<*>)) <*> pure) f x)
```



## Вычисления «от противного»

Проблема термов в типизированном  $\lambda$ -исчислении (и чистого функционального стиля): «что упало, то пропало». Если вычислено  $(M\ N)$ , то  $M$  и  $N$  по отдельности уже потеряны навсегда.



## Вычисления «от противного»

Проблема термов в типизированном  $\lambda$ -исчислении (и чистого функционального стиля): «что упало, то пропало». Если вычислено  $(M\ N)$ , то  $M$  и  $N$  по отдельности уже потеряны навсегда.

Логичный выход — возвраты, если стало ясно, что вычисления зашли в тупик. Аналог в доказательствах — работа с отрицанием.

$$\frac{\Gamma \vdash \tau, \Gamma \vdash \neg \tau}{\Gamma \vdash \sigma} \quad (\text{из лжи следует всё что угодно})$$

$$\frac{\Gamma \vdash \neg \neg \tau}{\Gamma \vdash \tau} \quad (\text{снятие двойного отрицания})$$



## Теорема Гливенко

Пусть  $\Phi$  — пропозициональная формула. Тогда  $\vdash \Phi$  в классической логике  $\Leftrightarrow \vdash \neg\neg\Phi$  в конструктивной интуиционистской логике.

Здесь под конструктивной интуиционистской логикой понимаем минимальную логику + «из лжи следует всё что угодно».





## Теорема Гливенко

Пусть  $\Phi$  — пропозициональная формула. Тогда  $\vdash \Phi$  в классической логике  $\Leftrightarrow \vdash \neg\neg\Phi$  в конструктивной интуиционистской логике.

\*  $\neg(\neg\neg A \Rightarrow A)$

\*  $A$

\*  $\neg\neg A$

$A$

$\neg\neg A \Rightarrow A$

$\perp$

$\neg A$

\*  $\neg\neg A$

$\perp$

$A$

$\neg\neg A \Rightarrow A$

$\perp$

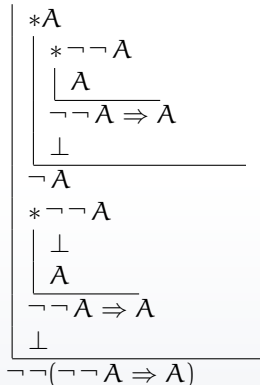
$\neg\neg(\neg\neg A \Rightarrow A)$



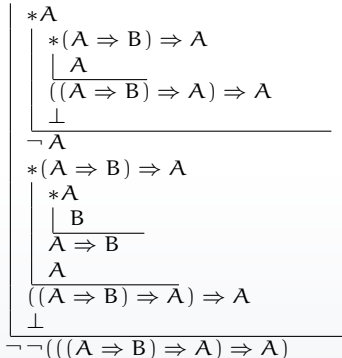
## Теорема Гливенко

Пусть  $\Phi$  — пропозициональная формула. Тогда  $\vdash \Phi$  в классической логике  $\Leftrightarrow \vdash \neg\neg\Phi$  в конструктивной интуиционистской логике.

\*  $\neg(\neg\neg A \Rightarrow A)$



\*  $\neg(((A \Rightarrow B) \Rightarrow A) \Rightarrow A)$





## Переход к относительному отрицанию

«Вычисления зашли в тупик»  $\Rightarrow$  приводят к нежелательному результату. Интерпретация  $\neg_R \Phi = (\Phi \Rightarrow R)$ .

Рассмотрим предыдущие два вывода с точки зрения извлечения термов из их конструкции.



## Переход к относительному отрицанию

«Вычисления зашли в тупик»  $\Rightarrow$  приводят к нежелательному результату. Интерпретация  $\neg_R \Phi = (\Phi \Rightarrow R)$ .

$$\begin{array}{l}
 *(\neg_R \neg_R A \Rightarrow A) \Rightarrow R \quad (\text{тип } x) \\
 \left| \begin{array}{l}
 *A \quad (\text{тип } y) \\
 \left| \begin{array}{l}
 *\neg_R \neg_R A \quad (\text{тип } z) \\
 \left| \begin{array}{l}
 A \quad (\text{терм } y) \\
 \hline
 \neg_R \neg_R A \Rightarrow A \quad (\text{терм } \lambda z.y) \\
 R \quad (\text{терм } x (\lambda z.y))
 \end{array}
 \end{array}
 \end{array}
 \right. \\
 A \Rightarrow R \quad (\text{терм } \lambda y.x (\lambda z.y)) \\
 *(A \Rightarrow R) \Rightarrow R \quad (\text{тип } w) \\
 \left| \begin{array}{l}
 R \quad (\text{терм } w (\lambda y.x (\lambda z.y))) \\
 A \quad (??!!!) \\
 \hline
 \neg_R \neg_R A \Rightarrow A \\
 R
 \end{array}
 \right. \\
 \hline
 ((\neg_R \neg_R A \Rightarrow A) \Rightarrow R) \Rightarrow R
 \end{array}$$



## Переход к относительному отрицанию

«Вычисления зашли в тупик»  $\Rightarrow$  приводят к нежелательному результату. Интерпретация  $\neg_R \Phi = (\Phi \Rightarrow R)$ .

$$\begin{array}{l}
 *(((A \Rightarrow B) \Rightarrow A) \Rightarrow A) \Rightarrow R \quad (\text{тип } x) \\
 \begin{array}{l}
 *A \quad (\text{тип } y) \\
 \begin{array}{l}
 *(A \Rightarrow B) \Rightarrow A \quad (\text{тип } z) \\
 \begin{array}{l}
 A \quad (\text{терм } y) \\
 \hline
 ((A \Rightarrow B) \Rightarrow A) \Rightarrow A \quad (\text{терм } \lambda z.y) \\
 R \quad (\text{терм } x (\lambda z.y)) \\
 \hline
 A \Rightarrow R \quad (\text{терм } \lambda y.x (\lambda z.y))
 \end{array}
 \end{array}
 \end{array}
 \end{array}
 \quad (\text{тип } u)$$

$$\begin{array}{l}
 *A \quad (\text{тип } w) \\
 \begin{array}{l}
 B \quad \text{???} \\
 \hline
 A \Rightarrow B \\
 A \\
 \hline
 ((A \Rightarrow B) \Rightarrow A) \Rightarrow A \\
 R
 \end{array}
 \end{array}$$

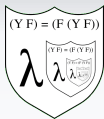
$$(((A \Rightarrow B) \Rightarrow A) \Rightarrow A) \Rightarrow R \Rightarrow R$$



## Переход к относительному отрицанию

«Вычисления зашли в тупик»  $\Rightarrow$  приводят к нежелательному результату. Интерпретация  $\neg_R \Phi = (\Phi \Rightarrow R)$ .

$$\begin{array}{l}
 *(((A \Rightarrow B) \Rightarrow A) \Rightarrow A) \Rightarrow B \quad (\text{тип } x) \\
 \begin{array}{l}
 *A \quad (\text{тип } y) \\
 \begin{array}{l}
 *(A \Rightarrow B) \Rightarrow A \quad (\text{тип } z) \\
 \begin{array}{l}
 A \quad (\text{терм } y) \\
 \hline
 ((A \Rightarrow B) \Rightarrow A) \Rightarrow A \quad (\text{терм } \lambda z.y) \\
 B \quad (\text{терм } x (\lambda z.y))
 \end{array}
 \end{array}
 \end{array}
 \end{array}
 \end{array}
 \begin{array}{l}
 A \Rightarrow B \quad (\text{терм } \lambda y.x (\lambda z.y)) \\
 *(A \Rightarrow B) \Rightarrow A \quad (\text{тип } u) \\
 \begin{array}{l}
 *A \quad (\text{тип } w) \\
 \begin{array}{l}
 B \quad (\text{терм } (\lambda y.x (\lambda z.y)) w) \\
 \hline
 A \Rightarrow B \quad (\text{терм } \lambda y.x (\lambda z.y)) \\
 A \quad (\text{терм } u (\lambda y.x (\lambda z.y)))
 \end{array}
 \end{array}
 \end{array}
 \end{array}
 \begin{array}{l}
 ((A \Rightarrow B) \Rightarrow A) \Rightarrow A \quad (\text{терм } \lambda u.u (\lambda y.x (\lambda z.y))) \\
 B \quad (\text{терм } x (\lambda u.u (\lambda y.x (\lambda z.y)))) \\
 \hline
 (((A \Rightarrow B) \Rightarrow A) \Rightarrow A) \Rightarrow B \Rightarrow B \quad (\text{терм } (\lambda x.x (\lambda u.u (\lambda y.x (\lambda z.y))))
 \end{array}$$



## Навешивание двойного отрицания

Видно, что использование «возвратных» термов (переход от типа  $\Phi$  к типу  $\neg_R \neg_R \Phi$ ) расширяет возможности языка.

Возникает вопрос, в каких подформулах лучше это делать?

- Преобразование Колмогорова:

$$\tau(A) = \neg_R \neg_R A \quad \tau(\Phi \Rightarrow \Psi) = \neg_R \neg_R (\tau(\Phi) \Rightarrow \tau(\Psi))$$

- Вариант более слабого преобразования (в стиле Куроды) — это  $\neg_R \neg_R \tau'(\Phi)$ , где:

$$\tau'(A) = A \quad \tau'(\Phi \Rightarrow \Psi) = \tau'(\Phi) \Rightarrow \neg_R \neg_R \tau'(\Psi)$$

В примерах для краткости  $(\Phi \Rightarrow R) \Rightarrow R$  переобозначим как  $\Phi'$ .



## Переход по Колмогорову

$* (A' \Rightarrow ((A' \Rightarrow B')' \Rightarrow B')') \Rightarrow R$

$* (A \Rightarrow R) \Rightarrow R$

$* ((A' \Rightarrow B')' \Rightarrow B') \Rightarrow R$

$* ((A' \Rightarrow B') \Rightarrow R) \Rightarrow R$

$* B \Rightarrow R$

$* A' \Rightarrow B'$

$* A \Rightarrow R$

$R$

$(A \Rightarrow R) \Rightarrow R$

$(B \Rightarrow R) \Rightarrow R$

$R$

$(A' \Rightarrow B') \Rightarrow R$

$R$

$(B \Rightarrow R) \Rightarrow R$

$(A' \Rightarrow B')' \Rightarrow B'$

$R$

$((A' \Rightarrow B')' \Rightarrow B') \Rightarrow R$

$A' \Rightarrow ((A' \Rightarrow B')' \Rightarrow B')'$

$R$

$((A' \Rightarrow ((A' \Rightarrow B')' \Rightarrow B')') \Rightarrow R) \Rightarrow R$

Извлечённый терм:  $\lambda k_0. k_0 (\lambda x k_1. k_1 (\lambda y k_2. y (\lambda k_3. k_3 (\lambda k_4. x k_4) k_2)))$





## Слабый переход по Куроде

$*(A \Rightarrow ((A \Rightarrow B') \Rightarrow B')') \Rightarrow R$

$*A$

$*((A \Rightarrow B') \Rightarrow B') \Rightarrow R$

$*A \Rightarrow B'$

$(B \Rightarrow R) \Rightarrow R$

$*B \Rightarrow R$

$R$

$(B \Rightarrow R) \Rightarrow R$

$((A \Rightarrow B') \Rightarrow B')$

$R$

$((A \Rightarrow B') \Rightarrow B') \Rightarrow R$

$A \Rightarrow ((A \Rightarrow B') \Rightarrow B')'$

$R$

$((A \Rightarrow ((A \Rightarrow B') \Rightarrow B')') \Rightarrow R) \Rightarrow R$

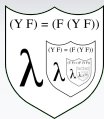
Извлечённый терм:  $\lambda k_0. k_0 (\lambda x k_1. k_1 (\lambda y k_2. y \ x \ k_2))$ .



## Система передачи продолжений

Система передачи продолжений в CBN-стиле:

- $\tau_N(\text{const}) = \lambda k. k \text{ const}$
- $\tau_N(x) = \lambda k. x \ k$
- $\tau_N(\lambda x. M) = \lambda k. k (\lambda x. \tau_N(M))$
- $\tau_N(M \ N) = \lambda k. \tau_N(M) (\lambda f. f \ \tau_N(N) \ k)$



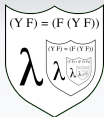
## Система передачи продолжений

Система передачи продолжений в CBN-стиле:

- $\tau_N(\text{const}) = \lambda k.k \text{ const}$
- $\tau_N(x) = \lambda k.x k$
- $\tau_N(\lambda x.M) = \lambda k.k (\lambda x.\tau_N(M))$
- $\tau_N(M N) = \lambda k.\tau_N(M) (\lambda f.f \tau_N(N) k)$

Система передачи продолжений в CBV-стиле:

- $\tau_V(\text{const}) = \lambda k.k \text{ const}$
- $\tau_V(x) = \lambda k.k x$
- $\tau_V(\lambda x.M) = \lambda k.k (\lambda x.\tau_V(M))$
- $\tau_V(M N) = \lambda k.\tau_V(M) (\lambda f.\tau_V(N) (\lambda a.f a k))$



## Применение CPS-преобразования

- $M \rightarrow^* \text{const}$  при CBV-стратегии  $\Leftrightarrow$   
 $\tau_V(M) \text{ id} \rightarrow^* \text{const}$  при какой угодно стратегии.
- $M \rightarrow^* \text{const}$  при CBN-стратегии  $\Leftrightarrow$   
 $\tau_N(M) \text{ id} \rightarrow^* \text{const}$  при какой угодно стратегии.



## Монада Cont

```
newtype MyCont r a = Mc  runCont :: (a -> r) -> r
instance Functor (MyCont r) where
  fmap f (Mc cps_a) = ...
instance Applicative (MyCont r) where
  pure v = ...
  Mc cps_f <*> Mc cps_a = ...
```



## Монада Cont

```
newtype MyCont r a = Mc  runCont :: (a -> r) -> r
instance Functor (MyCont r) where
  fmap f (Mc cps_a) = Mc (\cps -> cps_a (cps . f))
instance Applicative (MyCont r) where
  pure v = ...
  Mc cps_f <*> Mc cps_a = ...
```



## Монада Cont

```
newtype MyCont r a = Mc  runCont :: (a -> r) -> r
instance Functor (MyCont r) where
  fmap f (Mc cps_a) = Mc (\cps -> cps_a (cps . f))
instance Applicative (MyCont r) where
  pure v = Mc (\r -> r v)
  Mc cps_f <*> Mc cps_a = ...
```



## Монада Cont

```
newtype MyCont r a = Mc  runCont :: (a -> r) -> r
instance Functor (MyCont r) where
  fmap f (Mc cps_a) = Mc (\cps -> cps_a (cps . f))
instance Applicative (MyCont r) where
  pure v = Mc (\r -> r v)
  Mc cps_f <*> Mc cps_a =
    Mc (\cps_b -> cps_f (\f -> cps_a (cps_b . f)))
```





## Упражнение

Пользуясь соответствием Карри–Ховарда, построить явным образом терм, реализующий bind-оператор  $>>=$  для монады `Cont`.

Тип  $>>= :: \text{MyCont } a \rightarrow (a \rightarrow \text{MyCont } b) \rightarrow \text{MyCont } b$ , и это всё, что нужно знать для решения задачи.



## Прерывания вычислений

- Жёсткое прерывание — полный выход из контекста  $(\neg_R \neg_R A \Rightarrow A)$ ;
- Мягкое прерывание — возвращает текущий контекст вычислений  $((\neg_R A \Rightarrow A) \Rightarrow A)$  — закон Пирса, а также тип оператора call-with-current-continuation.



## Вывод = конструкция

\*(((A  $\Rightarrow$  B)  $\Rightarrow$  B)  $\Rightarrow$  A)  $\Rightarrow$  B)

\*B  $\Rightarrow$  A

\* (A  $\Rightarrow$  B)  $\Rightarrow$  B

\*A  $\Rightarrow$  B

B

A

(A  $\Rightarrow$  B)  $\Rightarrow$  A

A **Закон Пирса!**

((A  $\Rightarrow$  B)  $\Rightarrow$  B)  $\Rightarrow$  A

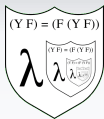
B

(B  $\Rightarrow$  A)  $\Rightarrow$  B

B

**Закон Пирса!**

((((A  $\Rightarrow$  B)  $\Rightarrow$  B)  $\Rightarrow$  A)  $\Rightarrow$  B)  $\Rightarrow$  B



## Вывод = конструкция

$$\begin{array}{l}
 *(((A \Rightarrow B) \Rightarrow B) \Rightarrow A) \Rightarrow B \quad (\text{тип терма } x) \\
 \begin{array}{l}
 *B \Rightarrow A \quad (\text{тип терма } y) \\
 \begin{array}{l}
 *(A \Rightarrow B) \Rightarrow B \quad (\text{тип терма } z) \\
 \begin{array}{l}
 *A \Rightarrow B \quad (\text{тип терма } w) \\
 \begin{array}{l}
 B \quad (\text{терм } z \ w) \\
 A \quad (\text{терм } y \ (z \ w))
 \end{array} \\
 \hline
 (A \Rightarrow B) \Rightarrow A \quad (\text{терм } \lambda w.y \ (z \ w)) \\
 A \quad (T_1 = \text{callcc } \lambda w.y \ (z \ w)) \\
 \hline
 ((A \Rightarrow B) \Rightarrow B) \Rightarrow A \quad (\text{терм } \lambda z.T_1) \\
 B \quad (\text{терм } x \ (\lambda z.T_1)) \\
 \hline
 (B \Rightarrow A) \Rightarrow B \quad (\text{терм } \lambda y.x \ (\lambda z.T_1)) \\
 B \quad (T_2 = \text{callcc } \lambda y.x \ (\lambda z.T_1)) \\
 \hline
 (((A \Rightarrow B) \Rightarrow B) \Rightarrow A) \Rightarrow B \Rightarrow B \quad \text{терм } \lambda x.T_2
 \end{array}
 \end{array}
 \end{array}$$



## Определение монады

```
class Applicative M => Monad M where  
  return :: a -> M a  
  (>>=) :: M a -> (a -> M b) -> M b
```

Дополнительные монадические операторы:

```
join :: M M a -> M a  
fmap :: (a -> b) -> M a -> M b
```



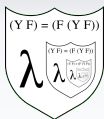
## Соответствие Карри–Ховарда

Propositional Lax Logic — монадическая логика с единственным модальным оператором  $\bigcirc$ .

Правила вывода (+ стандартные правила естественного вывода в интуиционистской логике):

- $\frac{A}{\bigcirc A}$  (Return)
- $\frac{A \Rightarrow \bigcirc B}{\bigcirc A \Rightarrow \bigcirc B}$  (Bind)

Сохраняет возможность извлечь монадический оператор из вывода типов! Обратим внимание на инвертированный порядок аргументов.



## Пример разбора задачи на монадическое соответствие

Построим оператор `join` из операторов `>>=` и `return`.

\*  $\bigcirc \bigcirc A$  (тип терма  $x$ )

*(Здесь нужно придумать, как объединить контейнеры, обрабатывающие  $x$ )*

$\bigcirc A$

$\bigcirc \bigcirc A \Rightarrow \bigcirc A$  (тип терма  $\lambda x. \dots$ )

Единственный оператор, умеющий заглядывать внутрь контейнера — это `bind`. Он требует два аргумента: контейнерный типа  $M\ a$  и монадическую функцию типа  $a \rightarrow M\ b$ . Поскольку нужно заглянуть внутрь только одной из двух контейнерных оболочек, в роли типа  $a$  выступает также монада  $M\ a'$ . Значит, нам нужна стрелка типа  $M\ a' \rightarrow M\ a'$ . Это функция `id`, вывод которой мы уже умеем строить.



## Пример разбора задачи на монадическое соответствие

\*  $\bigcirc \bigcirc A$  (тип терма  $x$ )

\*  $\bigcirc A$  (тип терма  $y$ )

|  $\bigcirc A$  (просто возвращаем сам  $y$ )

$\bigcirc A \Rightarrow \bigcirc A$  (тип терма  $\lambda y. y$ )

*(Аргументы bind-оператора построены,  
осталось применить их в правильном порядке)*

$\bigcirc A$

$\bigcirc \bigcirc A \Rightarrow \bigcirc A$  (тип терма  $\lambda x. \dots$ )





## Пример разбора задачи на монадическое соответствие

\*  $\circ \circ A$  (тип терма  $x$ )

\*  $\circ A$  (тип терма  $y$ )

|  $\circ A$  (просто возвращаем сам  $y$ )

$\circ A \Rightarrow \circ A$  (тип терма  $\lambda y. y$ )

*(Аргументы bind-оператора построены, осталось применить их в правильном порядке)*

$\circ A$  (тип терма  $x \gg = (\lambda y. y)$ )

$\circ \circ A \Rightarrow \circ A$  (тип терма  $\lambda x. (x \gg = (\lambda y. y))$ )



## Пример разбора задачи на монадическое соответствие

\*  $\bigcirc \bigcirc A$  (тип терма  $x$ )  
| \*  $\bigcirc A$  (тип терма  $y$ )  
| |  $\bigcirc A$  (просто возвращаем сам  $y$ )  
|  $\bigcirc A \Rightarrow \bigcirc A$  (тип терма  $\lambda y. y$ )  
| (*Аргументы bind-оператора построены, осталось применить их в правильном порядке*)  
|  $\bigcirc A$  (тип терма  $x \gg= (\lambda y. y)$ )  
|  $\bigcirc \bigcirc A \Rightarrow \bigcirc A$  (тип терма  $\lambda x. (x \gg= (\lambda y. y))$ )

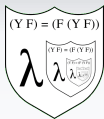
С помощью построения вывода в модальной логике мы просто решили уравнение в ФВП, получив ответ:

`join x = x >>= id.`



## Задачи

- Построить вывод  $(A \Rightarrow B) \Rightarrow \bigcirc A \Rightarrow \bigcirc B$ .
- Построить вывод  $\bigcirc(A \Rightarrow B) \Rightarrow \bigcirc A \Rightarrow \bigcirc B$ .



## Задачи

- Построить вывод  $(A \Rightarrow B) \Rightarrow \bigcirc A \Rightarrow \bigcirc B$ .
- Построить вывод  $\bigcirc(A \Rightarrow B) \Rightarrow \bigcirc A \Rightarrow \bigcirc B$ .

`liftM` — аналог `fmap` для монад, стандартно определяется через `bind` и `return`. Как?

А как определить через них `<*>`?



## Задачи

- Построить вывод  $(A \Rightarrow B) \Rightarrow \bigcirc A \Rightarrow \bigcirc B$ .
- Построить вывод  $\bigcirc(A \Rightarrow B) \Rightarrow \bigcirc A \Rightarrow \bigcirc B$ .

`liftM` — аналог `fmap` для монад, стандартно определяется через `bind` и `return`. Как?

```
liftM f m = m >>= \i -> return (f i)
```

А как определить через них `<*>`?

```
f <*> a  
  = f >>= (\xf -> a >>= (\xa -> return (xf xa)))
```

Всё!

Спасибо за внимание!