λ計算とその楽しさ

@lmdexpr

関数好きですか？

え、好き？

やったぜ。

という訳で関数型

- Haskell
- Lisp
- Clojure
- LazyK
- Scala

関数って何だ

ここに値がある
じゃろ？

(^ω^)
⊃ n ⊂

これを

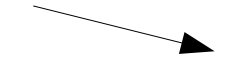
(^ω^)
⊃) n (⊂

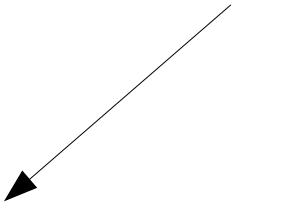
こうして

(^ω^)
≡⊃⊂≡

こうじゃ

(^ω^)
⊃ m ⊂





どういうことだってばよ。。。

関数は関数じゃ

で、λの意味は？

λ計算という計算があってだな

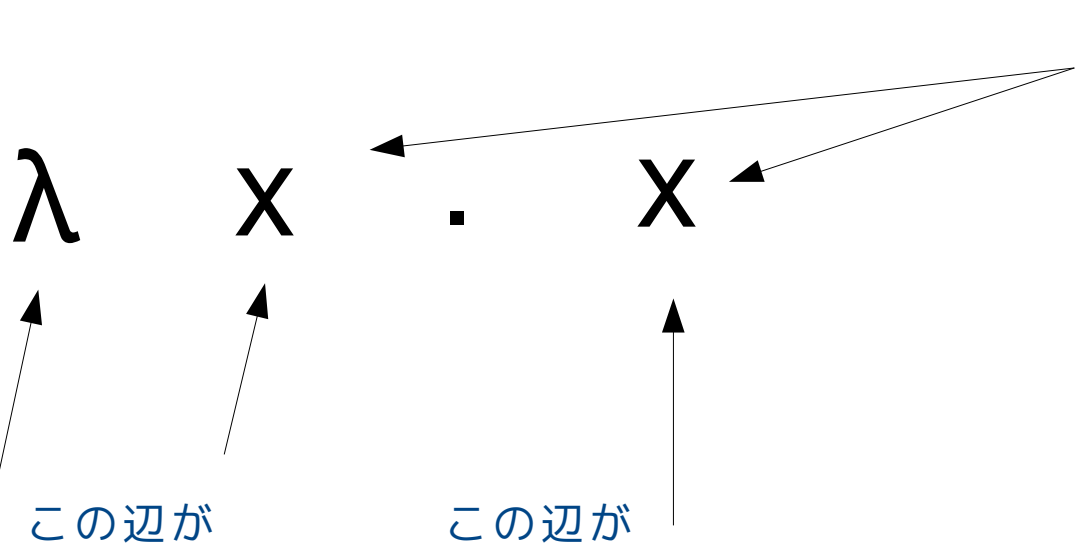
例えばこんなの↓

λx . x

λx . λy . x+y

λx . λy . λz . x+y+z

さっきの例を使ったてきとー解説



この辺は

可愛い！！！！！！！

変数(variable)

1,自由変数

→値が束縛されていない変数のこと

ex) λx.y における y

2,束縛変数

→値が束縛されている変数のこと

ex) λx.x における x

残りの細かい所

- 関数適用(application)

  関数を他の関数や値に適用すること

  例)

  $(\lambda x.x)2 \rightarrow 2$

  (上のように計算してしまうことを簡約と呼ぶ)

- これから出るキーワード

  SKIコンビネータ、チャーチ数、etc...

λ抽象は何が嬉しいのか

- 高階関数(higher-order function)
- 無名関数(anonymous function,
  nameless function)

これも全てλ抽象のおかげって訳よ！

無名関数

- 名前が無いのです。。。
- λ式を用いた表現が一般的
- 仕様は各言語へ

高階関数 ~上位の世界へ~

- 関数を引数に取ったり、返り値にしたり。。。
- 関数を呼び出す関数を呼び出す関数を呼び出す関数を.....
- あと、カリー化(食べられない)

カリー化の前に

引数を二つ取る関数は
高階関数の構文糖衣に過ぎぬ

何言ってんだ？？？？？？？？？？

こういうこと

λ x . λ y . x+y

↓

λ x y . x + y

食べられないカリー化

引数を一つ取る形にすること。

例)

$\lambda xyz.x+y+z \rightarrow \lambda x\ .\ \lambda y\ .\ \lambda z\ .\ x+y+z$

カリー化された関数おいしい！カリー化された関数おいしい！

カリー化について学んだぞ！

(˘⊖˘) 。o(待てよ？カリー化は何の役に立つんだ？)

|@lmdexpr| ┗(ᗢ｀　)┓三

(◠‿◠)☛ スマン、わからん

うわあああああああああああ

なんてことはありません

カリー化すると「部分適用」出来る

例えば+2する関数において

λx.x+2 と

(λx.λy.x+y)2 は

全く同じもの

Haskellでは

- 演算子(+)について

  Haskellの関数(演算子)はカリー化されてたりされてなかったり

  それを使えば、

  +2する関数において

  plusTwo = (+) 2

つまり、どういうことだってばよ

色々便利になるってことだよ。
言わせんな、恥ずかしい。

λ計算の同値関係、変換

α-変換

β-変換(β-簡約)

η-変換

α-変換

束縛変数の名前は需要でないってとこから来た奴(α-同値)

ようは変数名置き換えてたのちいってだけ

例)

λxyz.x z (y z)

α

→λzyx.z x (y x)

α

→λabc.a c (b c)

β-変換(β-簡約)

関数適用のことだよ。
言わせんな、恥ずか(ry

例)

$(\lambda x.x+2)1$

β

$\rightarrow 1+2$

ちなみに
これ以上β-簡約できない形
のことを
「正規形」と呼びます

η変換

関数の外延性を保証するもの
(訳分からないなんて言えない.....

例)

$\lambda x.M$

η

$\rightarrow M$

※外延性とは
二つの関数があらゆる引数に対して同値を返すなら互いに同値であるということ

次はSKIコンビネータとの戦いへ

$S = \lambda x y z . x z (y z)$

$K = \lambda x y . x$

$I = \lambda x . x$

ところでSKIコンビネータ is 何

それさえあれば全ての関数を表現できるというコンビネータ

それを利用した言語がLazyK

ちなみにIコンビネータはSとK
で表現できるので実質SKコンビネータでもいい

コンビネータ(Identity Combinator)

- **λ x . x**
- 恒等関数
- 常に引数と同値を返す

Kコンビネータ(Constant Combinator)

- **λ x y . x**

- 定数関数

  → ドイツ語よりKonstant

Sコンビネータ(Substitution Combinator)

- **λfgx.fx(gx)**

- xをgに適用した結果を
  xをfに適用した結果に対して
  適用する

成し遂げたぜ。

このSKIコンビネータを覚えて
次のステップへ

最初に真偽値があった

True　= λ x y . x

False　= λ x y . y

if B then T else F = BTF

λ式で真偽値を定義できた！！！

神は真偽値を見て、良しとされた

使ってみよう。

If True then 42 else 54

= True 42 54

= (λ x y . x) 42 54

β

→42

そして、くりかえす

- 条件分岐はうまくいくっぽい
- じゃあ、反復処理は？
- 作るか

コンビネータが増えるよ。
やったね、たえちゃん！

- Yコンビネータ(不動点コンビネータ)
- 深く考えるな、感じろ

- $Y = \lambda f . (\lambda x . f (x x)) (\lambda x . f (x x))$

- 定義を追うのはおすすめしない(遠い目

- ちなみにSKIで示すと

  $Y = S (K (S I I)) (S (S (K S) K) (K (S I I)))$

感じちゃう///

$$Y\,M = \lambda f.(\lambda x.f(x\,x))(\lambda x.f(x\,x))\,M$$

$$\xrightarrow{\beta} (\lambda x.M(x\,x))(\lambda x.M(x\,x))$$

$$\xrightarrow{\beta} M(\lambda x.M(x\,x))(\lambda x.M(x\,x))$$

$$M(YM) = M(\lambda x.M(x\,x))(\lambda x.M(x\,x))$$

ﾍﾟﾛｯ、これは！同値！！

$$Y\,M = M\,(Y\,M)$$

これこそ不動点演算子(コンビネータ)であるということ

つまり、f (g (x)) = g (x) が成り立つ

神は六日で命令型を作られた

- 条件分岐ができる。
- 反復処理ができる。
- 逐次実行もできる。
- これって命令型じゃん！！！

→プログラミングができるということ

んでんでんでwwwww

- なんかYコンビネータをこのまま使おうとするとバッファオーバーランしちゃうらしい
- どうしよう
- 遅延評価だ！！！！！

Zコンビネータです！！！

$Z = \lambda f.(\lambda x.\lambda m.f(xx)(m))(\lambda x.\lambda m.f(xx)(m))$

(今回は紹介だけ

自然数(チャーチ数)

$0 = \lambda\, s\, z\, .\, z$

$1 = \lambda\, s\, z\, .\, s\, z$

$2 = \lambda\, s\, z\, .\, s\, s\, z$

$3 = \lambda\, s\, z\, .\, s\, s\, s\, z$

$s = succ$
$= \lambda\, x\, .\, x + 1$

$z = 0$

いつから自然数だけと錯覚していた？

- 真偽値 ← 実装済み
- List
- Tuple
- 演算子
- その他あらゆる関数
- 制御文
- プログラムで出来るあらゆること

→ これぞまさにチューリング完全！

lmd先生の次回作にご期待ください！

- 今回は時間的にも体力的にもここまで
- 次回までにはλ計算処理系実装したいなあ
  (JOIあるから無理
- 多分次回はC++の話
- ではでは

まとめ

λの力は凄いじゃろ？
偉大じゃろ？
可愛いじゃろ？

勉強、参考に使ったサイト

- http://www.slideshare.net/_yingtai/lambda-guide
  素晴らしいプレゼン資料
- http://d.hatena.ne.jp/naokirin/20120309/1331286179
  変換とかについて参考にしました
- http://www.tatapa.org/~takuo/kotori_ski/
  ことりちゃん is 可愛い
- http://d.hatena.ne.jp/tarao/20100208/1265605429
  一番お世話になったかも
- http://d.hatena.ne.jp/nowokay/20090409#1239268405
  チャーチ数とか

質疑応答