「利子を得る」関数の陰仕様と、ニュートン法による陽仕様

佐原伸
日本フィッツ株式会社
情報技術研究所
TEL：03-3623-4683
shin.sahara@jfits.co.jp

2005 年 2 月 28 日

概 要

「利子を得る」関数の陰仕様と、ニュートン法による陽仕様。「導関数」や「ニュートン法で方程式を解く」操作は、関数を引数として渡さず、「利子を得る」問題に特化している。

1 はじめに

ある年数で資金を何倍かにしたいとき、その利子は幾らであるかを求める関数の仕様と、そのニュートン法による解。関数型プログラミング技法を使えば、より汎用的になる。

1.0.1 責任

「利子を得る」関数の陰仕様と、ニュートン法による陽仕様。

1.0.2 注釈

「導関数」や「ニュートン法で方程式を解く」操作は、関数を引数として渡さず、「利子を得る」問題に特化している。

class $Newton$

values

1.0 $Rcsid = \text{"\$}Id\ :\ Newton.vpp, v\ 1.7\ 2005/07/11\ 00:58:41\ shin\ Exp\ \$\text{"};$

2.0 誤差 $= 10 \uparrow -10;$

3.0 変分 $= 10 \uparrow -5$

functions

4.0 利子を得る陰仕様-数学版 : $\mathbb{R} \times \mathbb{Z} \to \mathbb{R}$

.1 利子を得る陰仕様-数学版 $(倍数, 年) \triangleq$

.2 is not yet specified

.3 pre $倍数 > 1 \wedge 年 > 0$ post $倍数 > 1 \wedge 年 > 0 \wedge$

.4 $\exists!\ 利子 : \mathbb{R} \cdot$

.5 let $元利合計 = 元利合計倍数を得る(利子, 年)$ in

.6 $倍数 = 元利合計 \wedge RESULT = 利子;$

5.0 利子を得る陰仕様-計算機版 : $\mathbb{R} \times \mathbb{Z} \to \mathbb{R}$

.1 利子を得る陰仕様-計算機版 $(倍数, 年数) \triangleq$

.2 is not yet specified

.3 pre $倍数 > 1 \wedge 年数 > 0 \wedge 誤差 > 0$ post $倍数 > 1 \wedge 年数 > 0 \wedge 誤差 > 0 \wedge$

.4 $\exists!\ 利子 : \mathbb{R} \cdot$

.5 let $元利合計 = 元利合計倍数を得る(利子, 年数)$ in

.6 abs $(倍数 - 元利合計) < 誤差 \wedge RESULT = 利子;$

利子を得る陽仕様（ニュートン法を利用、関数プログラミング未使用）

public static

6.0 利子を得る : $\mathbb{R} \times \mathbb{Z} \to \mathbb{R}$

.1 利子を得る $(倍数, 年数) \triangleq$

.2 ニュートン法で方程式を解く $(倍数, 年数, 0);$

public static

7.0 元利合計倍数を得る : $\mathbb{R} \times \mathbb{Z} \to \mathbb{R}$

.1 元利合計倍数を得る $(利子, 年) \triangleq$

.2 $(1 + 利子) \uparrow 年$

.3 pre $利子 \geq 0 \wedge 年 > 0$;

public static

8.0 導関数 : $\mathbb{R} \times \mathbb{Z} \times \mathbb{R} \to \mathbb{R}$

.1 導関数 $(倍数, 年数, 利子) \triangleq$

.2 $(差を求める(倍数, 年数, 利子 + 変分) - 差を求める(倍数, 年数, 利子))/変分;$

public static

```
 9.0    差を求める : ℝ × ℤ × ℝ → ℝ
  .1    差を求める (倍数, 年数, 利子) ≜
  .2      倍数 – 元利合計倍数を得る (利子, 年数)
operations
public static
10.0    ニュートン法で方程式を解く : ℝ × ℤ × ℝ →o ℝ
  .1    ニュートン法で方程式を解く (倍数, 年数, 利子初期値) ≜
  .2      (   dcl 利子 : ℝ := 利子初期値;
  .3          while ¬ (abs (差を求める (倍数, 年数, 利子)) < 誤差)
  .4          do 利子 := 利子 – (差を求める (倍数, 年数, 利子)/導関数 (倍
数, 年数, 利子));
  .5          return 利子
  .6      )
end Newton
```

Test Suite : vdm.tc

Class : Newton

Name	**#Calls**	**Coverage**
Newton‘導関数	5	√
Newton‘利子を得る	1	√
Newton‘差を求める	21	√
Newton‘元利合計倍数を得る	21	√
Newton‘利子を得る陰仕様-数学版	0	0%
Newton‘ニュートン法で方程式を解く	1	√
Newton‘利子を得る陰仕様-計算機版	0	0%
Total Coverage		**51%**

```
  class NewtonT is subclass of TestDriver
functions
 11.0   tests : () → TestCase*
   .1   tests () ≜
   .2     [
   .3      new NewtonT01 ()]
  end NewtonT
  class NewtonT01 is subclass of TestCase
values
 12.0   誤差 = 10 ↑ −10
operations
protected
 13.0   test : () →o B
   .1   test () ≜
   .2     let r = new Newton () in
   .3     (    dcl w : R;
   .4          return (r.利子を得る (2, 10) − 0.071773 < 誤差)
   .5     );
protected
 14.0   準備する : () →o ()
   .1   準備する () ≜
   .2     テスト名 := "RealT01 : \t「利子を得る」を検査する";
protected
 15.0   後始末する : () →o ()
   .1   後始末する () ≜
   .2     return
  end NewtonT01
```

参考文献

[1] Cliff Jones： Systematic Software Development using VDM, Prentice Hall International(1990)

[2] IFAD： The IFAD VDM++ Language V6.8, IFAD(2001)

[3] 佐原伸：事務システムにおける形式仕様適用例, ソフトウェア技術者協会, ソフトウェア・シンポジウム (2001)

[4] 佐原伸：大規模事務処理システムにおける形式手法の適用経験, ソフトウェア技術者協会, ソフトウェア・シンポジウム (2003)

[5] 佐原伸：VDM++基本ライブラリの作成, ソフトウェア技術者協会, ソフトウェア・シンポジウム (2004)

[6] ジョン・フィッツジェラルド，ピーター・ゴーム・ラーセン著, 荒木啓二郎，張漢明，荻野隆彦，佐原伸，染谷誠 訳：ソフトウェア開発のモデル化技法, 岩波書店 (2003)