関数プログラムの設計

胡 振江

東京大学 計数工学科

2008 年 12 月 1 日

関数プログラムの設計

ステップ１： 解きたい問題を理解する.

例：三つの数字の中間値を求めよ.

- $2, 3, 4$ の中間値は 3 である.
- $2, 2, 4$ の中間値は 2？それともなし？

Thinking before doing!

関数プログラムの設計

ステップ２：関数の型を決め，問題を解くための情報を集める.

型はプログラムのインターフェース (入出力）を表す.

$$\mathrm{middleNumber} :: \mathrm{Int} \rightarrow \mathrm{Int} \rightarrow \mathrm{Int} \rightarrow \mathrm{Int}$$

有益な情報：

- 型上の基本関数：e.g., Int 上の比較演算など
- 既存の関数ライブラリ: *max*, *min*, ...

関数プログラムの設計

ステップ３：複雑問題を簡単な問題に分割して解く.

どのような関数があればこの問題を解けるのかを考えながら，問題を解く.

$$
\begin{array}{l}
\text{middleNumber}\ x\ y\ z \\
\quad \begin{array}{|lcl}
\text{between}\ x\ y\ z & = & x \\
\text{between}\ y\ x\ z & = & y \\
\text{otherwise} & = & z
\end{array}
\end{array}
$$

$+$

m が n と p の中間値の時に True を返す関数
between m n p を設計する.

例題：平方根の計算

問題

任意正整数 x と小さい数 ϵ に対して

$$\text{sqrt}\ x \geq 0 \wedge |(\text{sqrt}\ x)^2 - x)| \leq \epsilon$$

を満たす sqrt 関数を作りたい。

例題：平方根の計算

関数の型を決め，問題を解くための情報を集める．

sqrt ::	Float $\longrightarrow$	Float $\longrightarrow$	Float
	入力 x	小さい数 ϵ	結果

有益な情報：

- Float 上の関数：e.g., square, abs
- 既存の計算法：Newton 法

例題：平方根の計算

Newton 法

- 要求される精度に到達するまで繰り返して解の近似値を求める方法.
- y_n を x の平方根の近似値とすると

$$y_{n+1} = (y_n + x/y_n)/2$$

  による y_{n+1} は y_n よりもよい近似値である.
- 例：2 の平方根の計算：

$$\begin{aligned}
y_0 &= 2 \\
y_1 &= (2+2/2)/2 && = 1.5 \\
y_2 &= (1.5+2/1.5)/2 && = 1.4167 \\
y_3 &= (1.4167+2/1.4167)/2 && = 1.4142157 \\
&\vdots
\end{aligned}$$

例題：平方根の計算

複雑問題を簡単な問題に分割して解く.

$$\text{sqrt}\ x\ \epsilon \quad = \quad \text{until satisfy improve}\ x$$

ここで，小問題の、until、satisfy と imrpove を次のように解く.

$$\begin{aligned}
\text{until} \quad &:: \quad (a \to \text{Bool}) \to (a \to a) \to a \to a \\
\text{until}\ p\ f\ x \quad &= \quad \textbf{if}\ p\ x\ \textbf{then}\ x\ \textbf{else}\ \text{until}\ p\ f\ (f\ x) \\
\text{satisfy}\ y \quad &= \quad \text{abs}(\text{square}\ y - x) \leq \epsilon \\
\text{improve}\ y \quad &= \quad (y + x/y)/2
\end{aligned}$$

最終のプログラム

$$
\begin{array}{lcl}
\text{sqrt} & :: & \text{Float} \rightarrow \text{Float} \rightarrow \text{Float} \\
\text{sqrt}\ x\ \epsilon & = & \text{until satisfy improve}\ x \\
 & & \textbf{where} \\
 & & \qquad \text{satisfy}\ y = \text{abs}(\text{square}\ y - x) \leq \epsilon \\
 & & \qquad \text{improve}\ y = (y + x/y)/2
\end{array}
$$

数をことばに

問題

0 以上 100 万以下の数を通常の英語で表現せよ。

例：

- 308,000 ⇒ three hundred and eight thousand
- 369,027⇒ three hundred and sixty-nine thousand and twenty-seven
- 369,401⇒ three hundred and sixty-nine thousand four hundred and one

解決法

簡単な問題から複雑問題へ

- $n < 100$ の数字を対象に
- $n < 1000$ の数字を対象に
- $n < 1000,000$ の数字を対象に

リストによる辞書の表現

$$
\begin{aligned}
units &= [\text{"}one\text{"}, \text{"}two\text{"}, \text{"}three\text{"}, \text{"}four\text{"}, \text{"}five\text{"},\\
&\quad \text{"}six\text{"}, \text{"}seven\text{"}, \text{"}eight\text{"}, \text{"}nine\text{"}]\\
teens &= [\text{"}ten\text{"}, \text{"}eleven\text{"}, \text{"}twelve\text{"}, \text{"}thirteen\text{"}, \text{"}fourteen\text{"}, \text{"}fifteen\text{"},\\
&\quad \text{"}sixteen\text{"}, \text{"}seventeen\text{"}, \text{"}eighteen\text{"}, \text{"}nineteen\text{"}]\\
tens &= [\text{"}twenty\text{"}, \text{"}thirty\text{"}, \text{"}forty\text{"}, \text{"}fifty\text{"},\\
&\quad s\text{"}sixty\text{"}, \text{"}seventy\text{"}, \text{"}eighty\text{"}, \text{"}ninety\text{"}]
\end{aligned}
$$

$0 < n < 100$ の場合

$$
\begin{array}{lcl}
convert2 & :: & Int \rightarrow String \\
convert2\ n & = & combine2\ (digits2\ n) \\
\\
digits2 & :: & Int \rightarrow (Int, Int) \\
digits2\ n & = & (n\ `div`\ 10, n\ `mod`\ 10) \\
\\
combine2 & :: & (Int, Int) \rightarrow String \\
combine2\ (0, u+1) & = & units\ !!\ u \\
combine2\ (1, u) & = & teens\ !!\ u \\
combine2\ (t+2, 0) & = & tens\ !!\ t \\
combine2\ (t+2, u+1) & = & tens\ !!\ t +\!\!+ "-" +\!\!+ units\ !!\ u
\end{array}
$$

$0 < n < 1000$ の場合

$$
\begin{array}{lcl}
convert3 & :: & Int \rightarrow String \\
convert3\ n & = & combine3\ (digits3\ n) \\
\\
digits3 & :: & Int \rightarrow (Int, Int) \\
digits3\ n & = & (n\ `div`\ 100, n\ `mod`\ 100) \\
\\
combine3\ (0, t+1) & = & convert2\ (t+1) \\
combine3\ (h+1, 0) & = & units\ !!\ h +\!\!+ "hundred" \\
combine3\ (h+1, t+1) & = & units\ !!\ h +\!\!+ "hundred\ and\ " +\!\!+ \\
 & & \quad convert2\ (t+1)
\end{array}
$$

$0 < n < 1000,000$ の場合

$$
\begin{array}{lcl}
convert6 & :: & Int \rightarrow String \\
convert6\ n & = & combine6\ (digits6\ n) \\
 & & \\
digits6 & :: & Int \rightarrow (Int, Int) \\
digits6\ n & = & (n\ \text{`}div\text{`}\ 1000, n\ \text{`}mod\text{`}\ 1000) \\
 & & \\
combine6 & :: & (Int, Int) \rightarrow String \\
combine6\ (0, h+1) & = & convert3\ (h+1) \\
combine6\ (m+1, 0) & = & convert3\ (m+1) +\!\!+ \text{"}\ thousand\text{"} \\
combine6\ (m+1, h+1) & = & convert3\ (m+1) +\!\!+ \text{"}\ thousand\text{"}\ +\!\!+ \\
 & & \quad link\ (h+1) +\!\!+ convert3\ (h+1)
\end{array}
$$

where

$$
\begin{array}{llcl}
link\ h & \mid h < 100 & = & \text{"}\ and\ \text{"} \\
 & \mid otherwise & = & \text{""}
\end{array}
$$

実行例

```
Convert> convert6 308000
"three hundred and eight thousand"
(985 reductions, 1350 cells)

Convert> convert6 369027
"three hundred and sixty-nine thousand and twenty-seven"
(1837 reductions, 2547 cells)

Convert> convert6 369401
"three hundred and sixty-nine thousand four hundred and one
(1851 reductions, 2548 cells)
```

Reference

- Simon Thompson, **Haskell: The Craft of Functional Programming** (Second Edition), Addison-Wesley, ISBN 0-201-34275-8, 1999.
  Chapter 4: Designing and Writing Programs.
- 教科書：2.1.5, 4.1, 5.1