数値計算法☆演習 L01
2 進数とオーバーフロー/アンダーフロー

樋口さぶろお

龍谷大学理工学部数理情報学科

2010-04-09

数値計算とは? こんなことに答えます

- $10.5x^{100} + 3.2x^{17} + 1.3 = 0$ を解け.
- $\cos x = x$ を解け.
- $\frac{\mathrm{d}^2x}{\mathrm{d}t^2}(t) + 1.3x(t) + e^{-t}\cos(x(t)) = 0, x(0) = 1.3, x'(0) = 1.9$ を解け.

1. 小数の答えでいい. `double` . → 誤差?
2. 電卓使えばいいじゃん. → 電卓はだれがつくったの?
3. Google で検索すればいいじゃん. → だれもやったことのない問題だったら?

数学を現実世界や (現実世界をまねた) ヴァーチャルな世界 (ゲーム?) に適用するには, ほとんどの場合, 数値計算が必要.
きみは `double`(=倍精度浮動小数点小数) の涙を見る!

2 進 →10 進変換

栗原 1.1

$$41_{10} = \begin{array}{|c|c|}\hline 4 & 1 \\ \hline 10^1 & 10^0 \\ \hline \end{array}$$

$$= 4 \times 10^1 + 1 \times 10^0$$

$$101001_2 = \begin{array}{|c|c|c|c|c|c|}\hline 1 & 0 & 1 & 0 & 0 & 1 \\ \hline 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ \hline \end{array}$$

$$= 1 \times 2^5 + 0 \times 2^4 + 1^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 41_{10}$$

10 進 → 2 進変換

$$101001_2 = 1 \times 2^5 + 0 \times 2^4 + 1^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 41_{10}$$

$$\begin{array}{rcl} 2)\underline{41} & & \\ 2)\underline{20} & \cdots & 1 \\ 2)\underline{10} & \cdots & 0 \\ 2)\underline{\ 5} & \cdots & 0 \\ 2)\underline{\ 2} & \cdots & 1 \\ 1 & \cdots & 1 \end{array} \rightarrow 11001_2$$

今やる問題

$126_{10}=$ $\qquad$ $11011_2=$ $\qquad$

2 進数の演算

$$101001_2 1 \times 2^5 + 0 \times 2^4 + 1^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 41_{10}$$

- **加算** $0 + 0 = 0, 1 + 0 = 0 + 1 = 1,$ $1 + 1 = 0$ **繰り上がり**

黒板でやるね〜

- 2_{10} **倍** $=$

黒板でやるね〜

- $1/2_{10}$ **倍** $=$

計算機で扱う数は有限長

ぜんぶ整数を表します.

bit=**桁数**

`char` **なら** 8bit.  2^8 **個のものが表現できる**.

多くのシステム上での C **言語の場合**.

型	**長さ**	**符号有り**
`char`	8bit	$-2^7 \leq x \leq +2^7 - 1$
`short`	16bit	
`int`	32bit	
`long`	**システムによる**	
`long long`	64bit	

```
int x=0;
char c;
```

`float`, `double` **は**?

全然違う仕組み. **待て次週**.

学籍番号/納税者番号番号

- 理工学部の学籍番号は t000000 から t999999. これを (t のことは忘れて, 変数に保存するには, 何 bit 必要でしょう?)
- 日本の人口は 1 億 2 千万人. これに納税者番号という通し番号をつけて管理するシステムは, その番号にどんな型を使えばいいでしょう?

有限桁での負の数の表現

8bit **の場合**, 2^8 **個の数が表せる**.

$\pm$	1	1	0	1	0	0	1
$\pm$	2^6	2^5	2^4	2^3	2^2	2^1	2^0

が自然.

もっといい方法:

$0111 = 7 = 2^3 - 1$

$0110 = 6$

$0101 = 5$

$0100 = 4$

$0011 = 3$

$0010 = 2$

$0001 = 1$

$0000 = 0$

$1111 = -1 \quad (0001_2$の補数$)$

$1110 = -2 \quad (0100_2$の補数$)$

$1101 = -3$

$1100 = -4$

$1011 = -5$

$1010 = -6$

$1001 = -7 \quad (0110_2$の補数$)$

$1000 = -8 = -2^3$

補数

2 進表現の全 bit を反転して, 1 を加えると, (-1) 倍した数の 2 進表現 (補数) が得られる.

いいこと

- $n$bit 分だけ見ると和の法則がそのまま成り立っている
- $n$bit 分だけ見ると $\times 2$ の法則がそのまま成り立っている
- 単純な計算機にぴったり!

有限長で表現できる符号付き整数

符号を補数を使って表現した場合, 符号に 1bit とられるので, $n$bit で表現できる整数は

$$-2^{n-1} \leq x \leq 2^{n-1} - 1.$$

$+1$ を繰り返すと?

電卓いじめ

$$\boxed{9}\boxed{9}\boxed{9}\boxed{9}\boxed{9}\boxed{9}\boxed{9}\boxed{9} + 1 = \boxed{}$$

C 言語は加算の計算方法を単純に適用しエラーも返さない

$n = 4$ bit の 2 進数の場合

$$2^3 - 1 = \boxed{0}\boxed{1}\boxed{1}\boxed{1} + 1 = \boxed{1}\boxed{0}\boxed{0}\boxed{0} = -8$$

つまり, 1 ずつ加えていくと,

$0, 1, 2, \ldots, 2^{n-1} - 1, -2^{n-1}, -2^{n-1} + 1, \ldots, 0$ と変化する.

そしてそのまま, 何事もなかったように計算を続ける…

$\times 2$ を繰り返すと?

電卓いじめその 2

$$\boxed{1}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0} \times 10 = \boxed{}$$

C 言語は $\times 2$ の計算方法を単純に適用しエラーも返さない

$$\boxed{0}\boxed{1}\boxed{0}\boxed{0} \times 2 = \boxed{1}\boxed{0}\boxed{0}\boxed{0} = -8.$$

つまり, $\times 2$ していくと $1, 2, 4, \ldots, 2^{n-1}, -2^{n-1}, 0, 0, \ldots$ と変化する.

オーバーフロー

固定小数点数 fixed point number

決まった長さの bit **列の一部を整数部分, 一部を小数部分と思う.**
1.01_2: **小数点** $\bullet$ **の左側が** 2^0 **を表す.**
2 進 10 進変換

0	$1_\bullet$	0	1
2^1	2^0	2^{-1}	2^{-2}

$$= 0\times 2^1 + 1\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2} = 1 + 0.25 = 1.25$$

有限長で $\times 1/2_{10}$ **を繰り返すと,**
$01.01_2 \to 00.10_2 \to 00.01_2 \to 00.00_2$

10 進 2 進変換

例 3.6875_{10}.

整数部分は前と同様 $3_{10} = 11_2$.

小数部分

		小数部分	整数部分	
$2\times$	0.6875	$= 0.375$	$+1$	
$2\times$	0.375	$= 0.75$	$+0$	$\rightarrow 1011_2$
$2\times$	0.75	$= 0.5$	$+1$	
$2\times$	0.5	$= 0$	$+1$	

答: 11.1011_2.

例題

- $1111.101_2 = ?$
- $3.14_{10} = ?$

黒板でやるね〜

有限桁で終わるとは限らない.

来週朝の quiz 計画

- この 10 進小数を 2 進小数に直せ
- この 2 進小数を 10 進小数に直せ