コンピュータ科学II

担当：武田敦志　<takeda@cs.tohoku-gakuin.ac.jp>

http://takeda.cs.tohoku-gakuin.ac.jp/

データ表現(1)

- コンピュータにおけるデータ表現

  コンピュータで扱う**全てのデータ**は

  ２進数の整数値で表現する

整数：100 ⟶ データ：1100100_2

小数：2.5
小数：0.625×2^3
} データ：$0.101_2 \times 11_2$

※本当は少し異なる表現をする

文字：A ⟶ データ：01000001_2

IPアドレス：192.168.0.254

⟶ データ：$11000000101010000000000011111111_2$

データ表現(2)

- データの長さ

  データの長さを『bit』で表す

  n桁の2進数整数値の長さは『n bit』となる

$7 = 111_2$ ⇒ 3bit

$100 = 1100100_2$ ⇒ 7bit

通常は決まった長さのデータを使う

（例えば 8bit）

$7 = 00000111_2$ ⇒ 8bit

$100 = 01100100_2$ ⇒ 8bit

2進数の計算(1)

- 計算問題 – 加算

(1) 197 ＋ 246

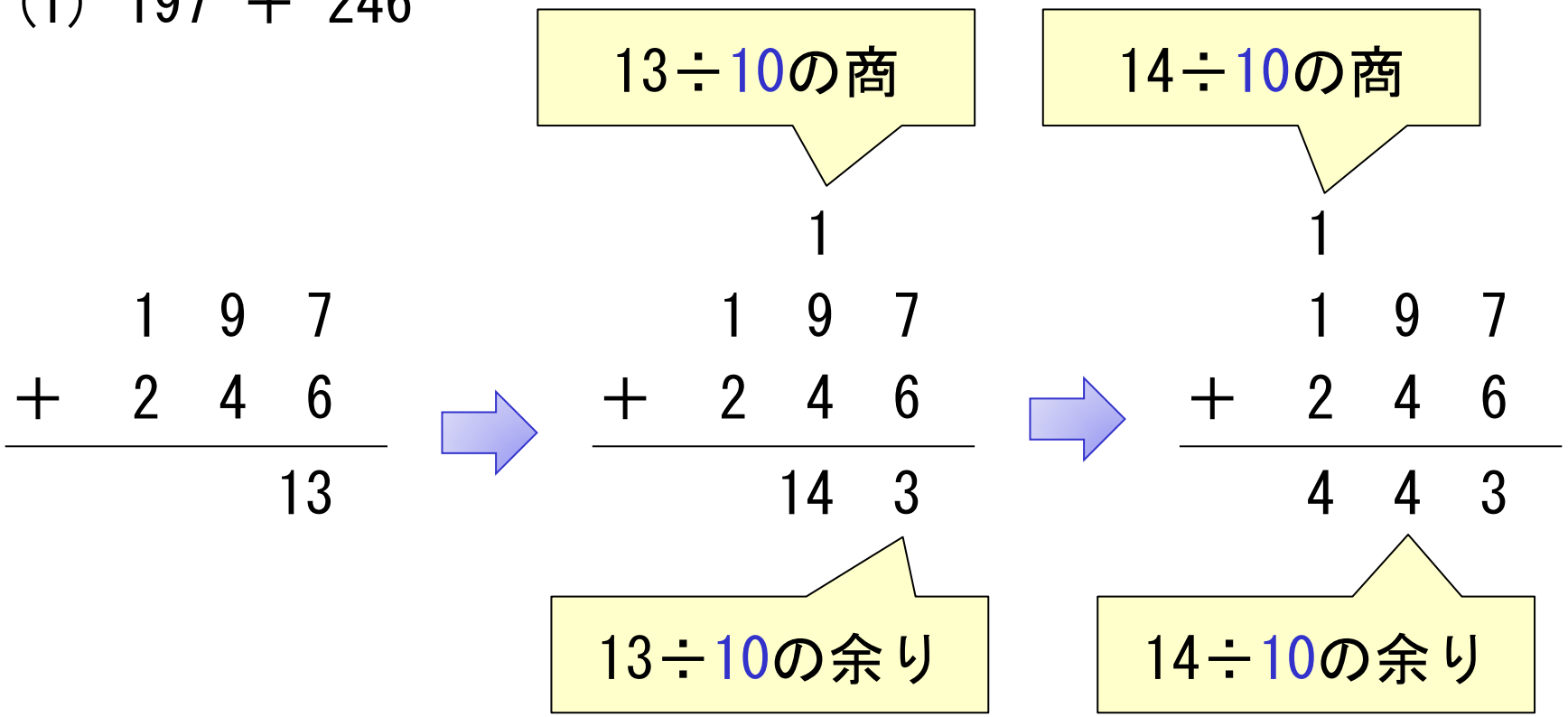
(2) 101101_2 ＋ 1101110_2

(3) 2時間42分 ＋ 1時間56分

２進数の計算(2)

- 10進数の加算

(1)　197　＋　246

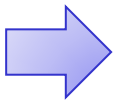


２進数の計算(3)

- 60進数の加算

(3)　2時間42分　＋　1時間56分

	2	時間	42	分
＋	1	時間	56	分
			98	分



98÷60の商

	1	時間		
	2	時間	42	分
＋	1	時間	56	分
	4	時間	38	分

98÷60の余り

2進数の計算(4)

- 2進数の加算

(2) $101101_2 + 1101110_2$

```
      1 0 1 1 0 1
+   1 1 0 1 1 1 0
-------------------
          2 1 1
```

2÷2の商

```
        1
      1 0 1 1 0 1
+   1 1 0 1 1 1 0
-------------------
        3 0 1 1
```

2÷2の余り

```
        1
      1 0 1 1 0 1
+   1 1 0 1 1 1 0
-------------------
        1 1 0 1 1
```

```
      1 0 1 1 0 1
+   1 1 0 1 1 1 0
-------------------
  1 0 0 1 1 0 1 1
```

２進数の計算(5)

- 加算・乗算・除算

  計算方法は10進数の計算方法と同じ

  基数が変わっても計算方法は変わらない

乗算：46 × 27

```
      4 6
×     2 7
-----------
    3 2 2
    9 2
-----------
  1 2 4 2
```

除算：106 ÷ 4

```
      2 6
   -------
4 ) 1 0 6
      8
   -------
      2 6
      2 4
   -------
        2
```

２進数の計算(6)

- 計算問題 – 乗算・除算

次の式を計算せよ

(1) $10110_2 \times 1011_2$

(2) $11010_2 \div 101_2$

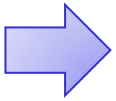
2の補数表現(1)

- コンピュータにおける除算

  加算回路を使って除算する

正の整数を除算

35 - 13 = 22



負の整数を加算

35 + (−13) = 22

負の整数をどのように表現するか？

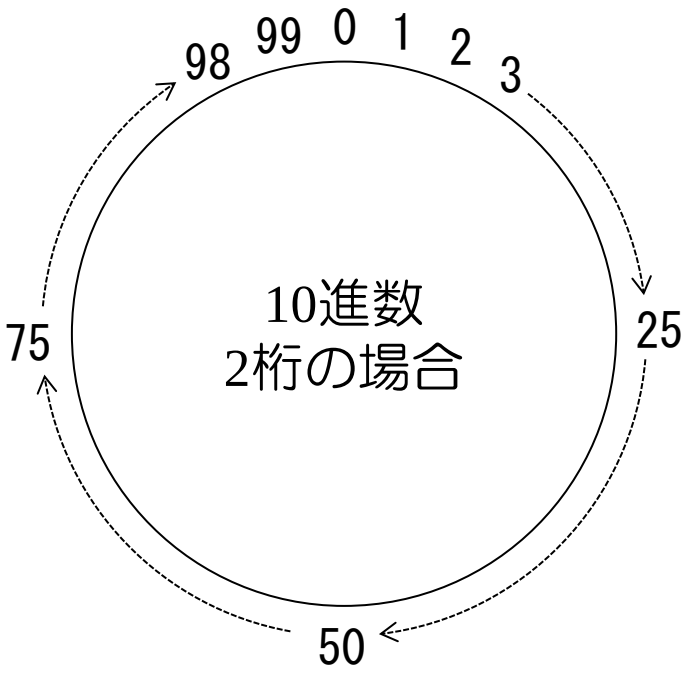
コンピュータにおける負数の表現方法＝２の補数表現

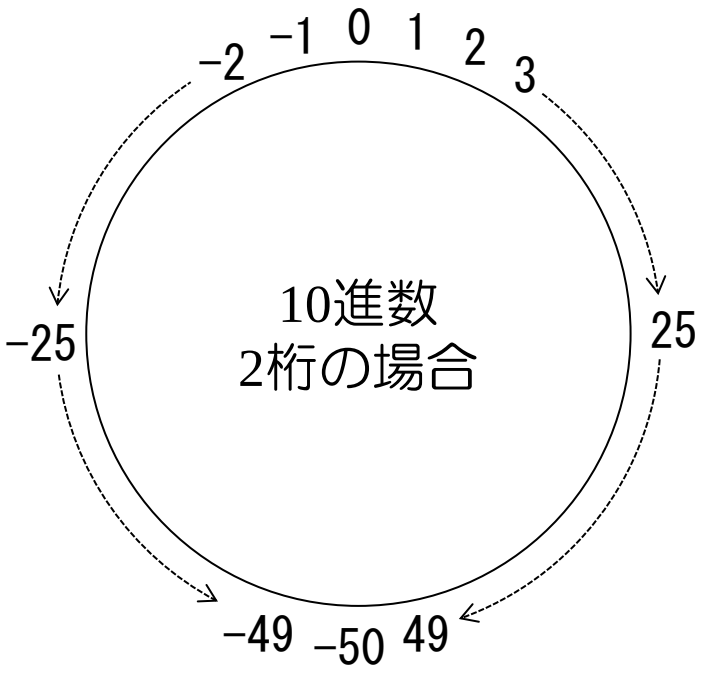
２の補数表現(2)

- ２の補数表現の考え方

(1) 桁数を設定
　円形の数直線ができる

(2) 負数の領域を設定
　数値の大きい領域に対し
　逆方向に負数を割り振る

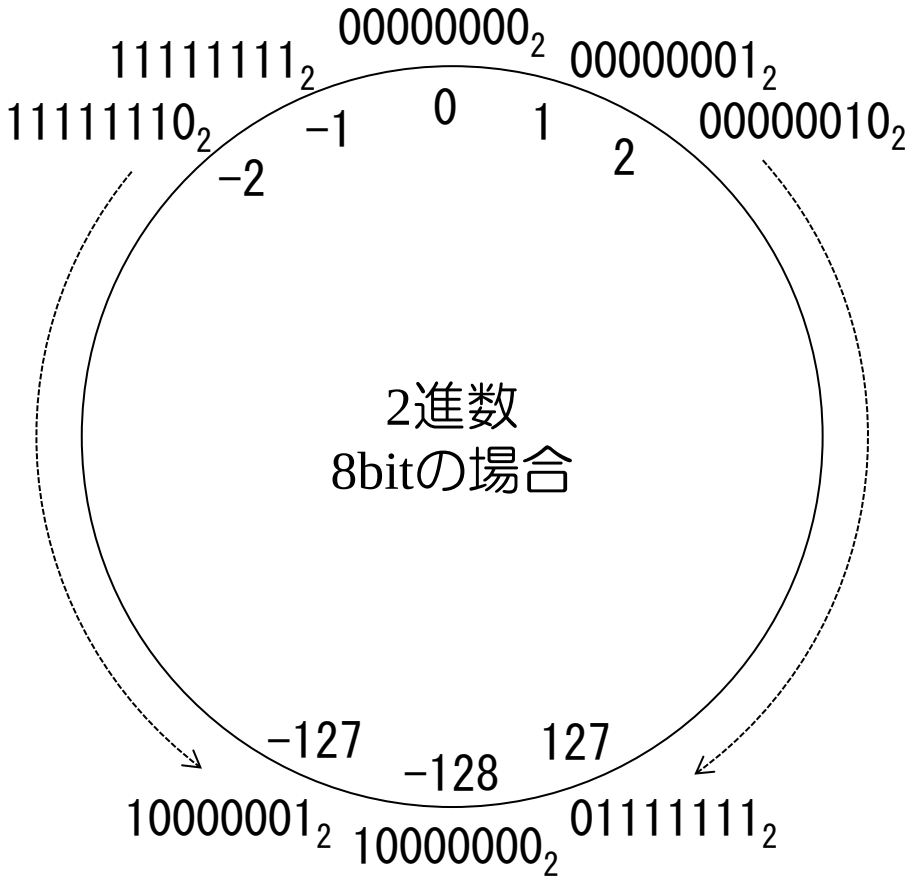




2の補数表現(3)

- 負の数の割り当て

  2進数8ビットの場合の数直線



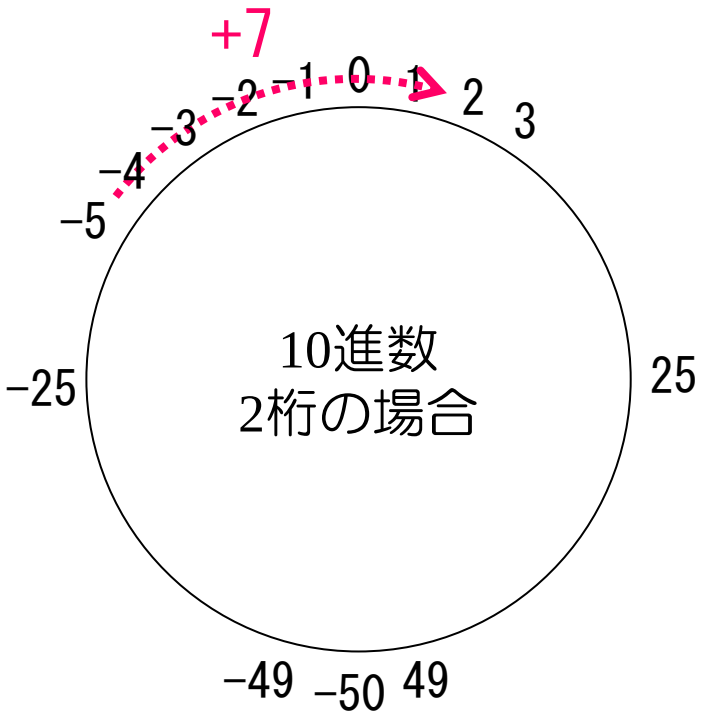
2の補数表現(4)

- ２の補数表現を使った除算
  加算回路を用いて計算する

7 − 5 = 2 ⇒ 7 + (−5) = 2

```
  0 0 0 0 0 1 1 1
+) 1 1 1 1 1 0 1 1
-----------------
1 0 0 0 0 0 0 1 0
```

最後の繰り上がりは
結果から消える



2の補数表現(5)

- 負数表現の計算方法

10進数2桁の場合

値	表現
−3	97
−2	98
−1	99
0	00
1	01
2	02

表現 = 100 − 値の絶対値

(例: 97 = 100 − 03)

2進数8bitの場合

値	表現
−3	11111101_2
−2	11111110_2
−1	11111111_2
0	00000000_2
1	00000001_2
2	00000010_2

表現 = 100000000_2 − 値の絶対値

(例: $11111101_2 = 100000000_2 - 00000011_2$)

2の補数表現(6)

- 負数表現の計算方法（繰り下がりなし）

10進数2桁の場合

負数表現 = 100 - 値の絶対値
= 99 - 値の絶対値 + 1

(例: 97 = 99 - 03 + 1)

2進数8bitの場合

負数表現 = 100000000_2 - 値の絶対値
= 11111111_2 - 値の絶対値 + 1

ビットの反転で計算できる

(例: 11111101_2 = 11111111_2 - 00000011_2 + 00000001)

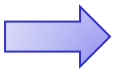
2の補数表現(7)

- 2の補数表現を使った減算

負数表現は「ビットの反転」と「加算」で計算できる

減算 ＝「正の整数」と「負の整数」の加算

2の補数表現を使った（2進数の）減算は
加算回路のみを用いて計算することができる

 加算回路があれば、四則演算のすべてが可能

2の補数表現(8)

- 演習問題

『−9』を2の補数表現(2進数8bit)で示せ

『17 + (−9)』 を2の補数表現(2進数8bit)を使って計算せよ

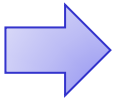
整数の計算(1)

- コンピュータにおける除算

  加算回路を使って除算する

正の整数を除算

35 - 13 = 22



負の整数を加算

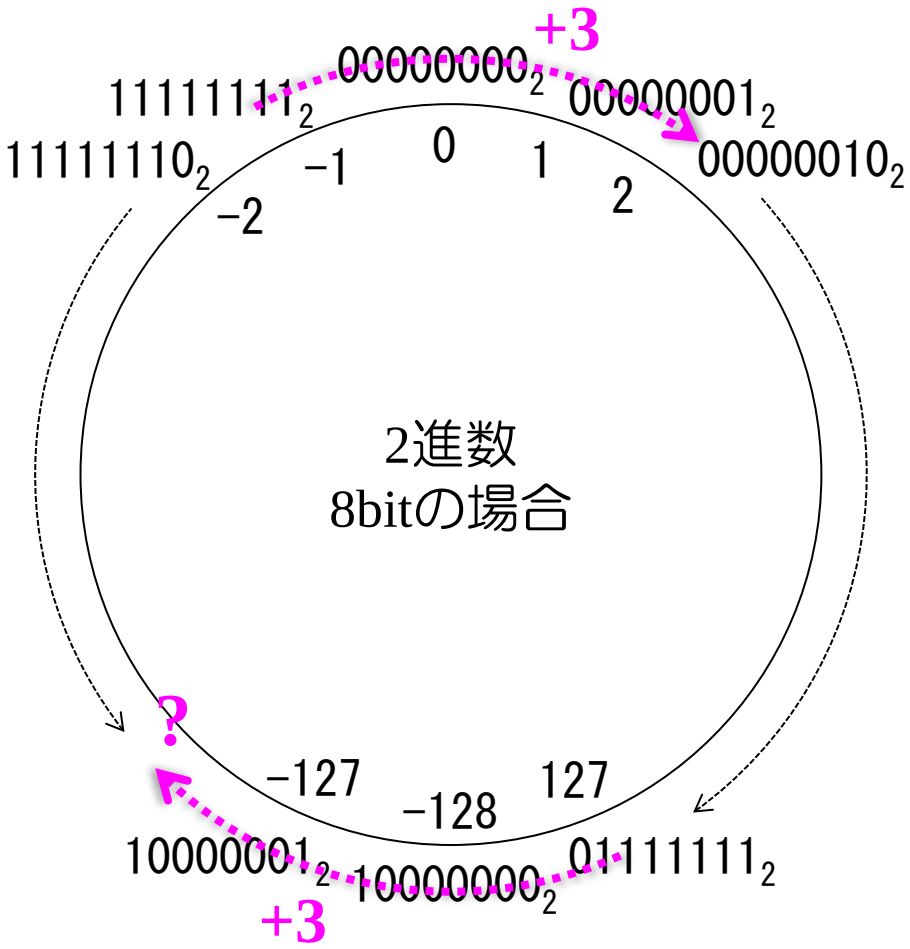
35 + (−13) = 22

負の整数をどのように表現するか？

コンピュータにおける負数の表現方法＝２の補数表現

整数の計算(2)

- ２の補数表現



円形の数直線を使って計算

整数の桁数は決まっている

不連続な部分がある

整数の計算(3)

- 桁あふれ

  表現できる値の範囲を超えて計算すると
  間違った計算結果となる

  2進数8bit（２の補数表現）の場合

$$127 + 1 = 01111111_2 + 00000001_2 = 10000000_2 = -128$$

$$127 + 2 = 01111111_2 + 00000010_2 = 10000001_2 = -127$$

$$127 + 3 = 01111111_2 + 00000011_2 = 10000010_2 = -126$$

整数の計算(4)

- 整数計算を行う場合の注意

  整数で表現できる値の最小値・最大値を意識する

  最小値～最大値の中での計算であれば正しい結果となる

8ビット整数：
  最小値 = -128
  最大値 = 127

16ビット整数：
  最小値 = -32768
  最大値 = 32767

32ビット整数：
  最小値 = - 2147483648
  最大値 = 2147483647

整数の計算(5)

- 桁あふれを発生させるプログラム

プログラム（int型は32bit）

```
#include <stdio.h>

int main()
{
  int a = 1;
  int i;

  for(i = 0; i < 13; i++){
    printf("10の%2d乗 = %d¥n", i, a);
    a = a * 10;
  }
}
```

実行結果

```
10の 0乗 = 1
10の 1乗 = 10
10の 2乗 = 100
10の 3乗 = 1000
10の 4乗 = 10000
10の 5乗 = 100000
10の 6乗 = 1000000
10の 7乗 = 10000000
10の 8乗 = 100000000
10の 9乗 = 1000000000
10の10乗 = 1410065408
10の11乗 = 1215752192
10の12乗 = -727379968
```

練習問題

- 2進数8bit（２の補数表現）の環境で、以下の計算せよ

(1) 120 + 80

(2) 100 × 4

コンピュータにおける小数の表現(1)

- 仮数と指数

  仮数と指数を使って数値を表現することができる

  例：光の速さ（秒速約30万km）

$$300,000,000 \text{ m/s}$$

$$3.0 \times 10^{8} \text{ m/s}$$

仮数　　指数

コンピュータにおける小数の表現(2)

- 浮動小数点数（IEEE754）

  コンピュータでは、小数を仮数＋指数で表現する

  例：35.375

$$100011.011_2 \Rightarrow 1.00011011_2 \times 2^5$$

仮数の小数点以下　指数

単精度 浮動小数点数（float）

0 10000100 00011011000000000000000

指数+127　仮数の小数点以下

コンピュータにおける小数の表現(3)

- 浮動小数点数の最大値と最小値（単精度浮動小数点数の場合）

  指数の表現範囲： $-126 \sim 127$

  仮数の表現範囲： $1 \sim (2 - 2^{-23})$

最小値：$1.0 \times 2^{-126} \fallingdotseq 1.175494 \times 10^{-38}$

0 00000001 00000000000000000000000

指数　　仮数の小数点以下

最大値：$(2 - 2^{-23}) \times 2^{127} \fallingdotseq 3.402823 \times 10^{38}$

0 11111110 11111111111111111111111

指数　　仮数の小数点以下

コンピュータにおける小数の表現(4)

- 2進数の小数（小数点以下の計算）

2進数から10進数への変換

$0.1_2 = 1/2$

$0.01_2 = 1/4$

$0.001_2 = 1/8$

⇒ $0.011_2 = 3/8 = 0.375$

10進数から2進数への変換

0.375

$= 375/1000$

$= 101110111_2/11111010000_2$

$= 0.011_2$

（工夫をすれば簡単に計算可能）

$$
\begin{array}{r}
0.011_2 \\
\hline
11111010000_2 \overline{)\ 101110111.000_2} \\
\quad 11111010\ \ 00_2 \\
\hline
\quad 01111101\ \ 000_2 \\
\quad\ \ 1111101\ \ 000_2 \\
\hline
\quad\ \ 0_2
\end{array}
$$

練習問題

- 下記の小数を2進数で表現せよ

(1) 0.625

(2) 9.4375

少数計算に発生する誤差(1)

- 2進数の無限小数

10進数と2進数では、無限小数となる値に違いがある

（例）

$0.1 = 1/10$

$= 0.00011001100..._2$

$$
\begin{array}{r|l}
 & 0.0001100110011...._2 \\
\hline
1010_2 & 1.0000000000000_2 \\
 & \quad 1010_2 \\
\hline
 & \quad\ 1100_2 \\
 & \quad\ 1010_2 \\
\hline
 & \quad\ 10000_2 \\
 & \quad\ \ 1010_2 \\
\hline
 & \quad\ \ 1100_2 \\
 & \quad\ \ 1010_2 \\
\hline
 & \quad\ \ \ 10...._2
\end{array}
$$

少数計算に発生する誤差(2)

- 丸め誤差

10進数では有限でも、2進数だと無限小数になるものがある

⟺ コンピュータ表現できる小数の長さは有限

（例）2進数（小数点以下8bit）で表現する場合

$0.1 = 0.0001100110011\ldots_2 \Rightarrow 0.00011010_2 = 0.1015625$

入力　　内部表現　　出力

$0.2 = 0.0011001100011\ldots_2 \Rightarrow 0.00110011_2 = 0.19921875$

入力　　内部表現　　出力

少数計算に発生する誤差(3)

- 丸め誤差を確認するプログラム

プログラム
（float型の小数点以下は23bit）
（double型の小数点以下は52bit）

```
#include <stdio.h>

int main()
{
  float a = 1.1;
  double b = 1.1;

  printf("a = %.20f¥n", a);
  printf("b = %.20f¥n", b);
}
```

実行結果

```
a = 1.10000002384185791016
b = 1.10000000000000008882
```

少数計算に発生する誤差(4)

- 丸め誤差が発生する計算

  コンピュータ内の小数の桁数は有限

  ⇒ 近似値で計算している場合がある

（例）2進数（小数点以下8bit）で表現

『(0.2 − 0.1) × 10』を計算する

$0.2 \Rightarrow 0.00110011_2$

$0.1 \Rightarrow 0.00011010_2$

$0.2 - 0.1 = 0.00011001_2 = 0.09765625$

$(0.2 - 0.1) \times 10 = 0.9765625$

少数計算に発生する誤差(5)

- 計算誤差が発生するプログラム

プログラム

```
#include <stdio.h>

int main()
{
  float a = 1.0 + 2.0 / 1000.0;
  float b = 1.0 + 1.0 / 1000.0;
  float c = (a - b) * 1000.0;

  printf("2.0 - 1.0 = %.20f¥n", c);
}
```

実行結果

```
(0.002 - 0.001) * 1000 = 0.99992752075195312500
```

練習問題

- 次の数値・計算を2進数(小数点以下8bit)で表現せよ

(1) 0.7

(2) 0.7 − 0.5

(3) (0.7 / 4 − 0.5 / 4) × 4