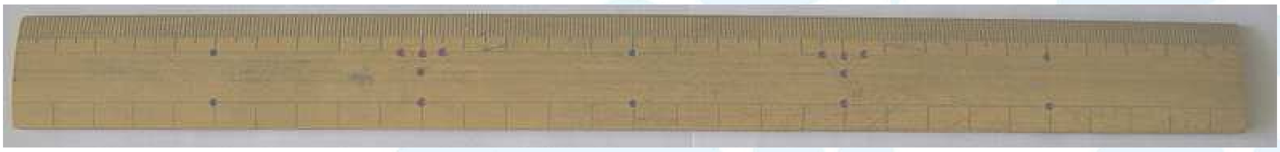
浮動小数点とは

その前に、「有効数字」の考え方を

「有効数字」

ものさしで長さを測る時

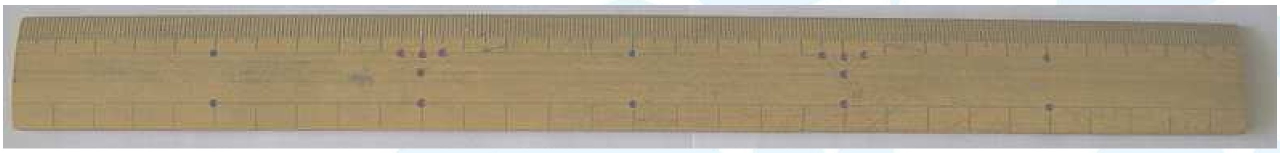


どれだけ細かく読めますか？

東邦大学

「有効数字」

ものさしで長さを測る時



どれだけ細かく読めますか？

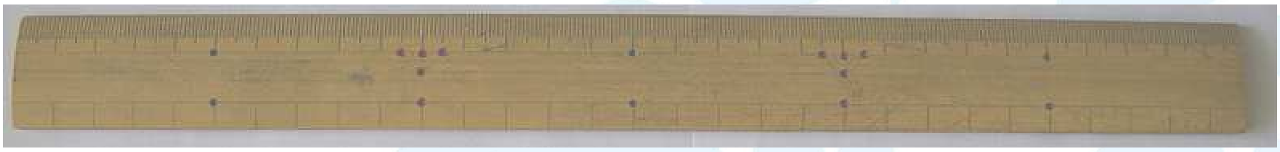
12.3 mm ?

12.34 mm ?

東邦大学

「有効数字」

ものさしで長さを測る時



どれだけ細かく読めますか？

12.3 mm　← まあ読めるか

12.34 mm ← 無理！

「有効数字」

測定器(方法)には限界があって
それ以上細かくは読み取れない

12.3●●● mm

読み取れない　書いても意味がない

「有効数字」

測定器(方法)には限界があって
それ以上細かくは読み取れない

12.3●●● mm

つまり、その桁以下の数字は
意味がないことになる

東邦大学

「有効数字」

測定器(方法)には限界があって
それ以上細かくは読み取れない

$\boxed{12.3}$●●● mm

つまり、その桁以下の数字は
意味がないことになる

意味のある部分を有効数字という

東邦大学

有効数字の表現

12.3 mm　有効数字がどれかわかる

でも

有効数字の表現

12.3 mm　有効数字がどれかわかる

でも

12300 μm　123まで確かなのか
12300まで確かなのか
わからない

有効数字の表現

12.3 mm　有効数字がどれかわかる

でも

12300 μm　123まで確かなのか
12300まで確かなのか
わからない

困る ⇒ そこで書き方工夫

有効数字の表現

12.3 mm　有効数字がどれかわかる

でも

12300 μm　123まで確かなのか
12300まで確かなのか
わからない

困る ⇒ そこで書き方工夫

1.23×10^4 μm　と書くことにする

有効数字の表現

有効数字分の桁数のみ書く
（この例では３桁）

$\boxed{1.23} \times 10^4$ μm

東邦大学

有効数字の表現

有効数字分の桁数のみ書く

$\boxed{1}.23 \times 10^4$ μm

小数点以上を
１桁だけにする

東邦大学

有効数字の表現

有効数字分の桁数のみ書く

$1.23 \times \boxed{10^4}$ μm

小数点以上を
１桁だけにする

桁を10○で合わせる
この例では
$12300=1.23\times10^4$

東邦大学

浮動小数点表示は
この形をまねる

有効数字の表現

有効数字分の桁数のみ書く

1.23×10^4 μm

小数点以上を
１桁だけにする

桁を10○で合わせる
この例では
$12300=1.23\times10^4$

東邦大学

10進での浮動小数だと

1.23×10^4

有効数字分の桁数のみ書く

小数点以上を１桁だけにする

桁を10○で合わせる

東邦大学

2進での浮動小数だと

$\boxed{1.101} \times \boxed{2}^4$

原理は同じで
2進数になる

有効数字分の桁数のみ書く

小数点以上を１桁だけにする

桁を2○で合わせる

東邦大学

２進浮動小数の原理が
分かりましたか？

○	×

次へ

浮動小数の
コンピュータ上での表現は？

２つの要素がある

$$\boxed{1.101} \times 2^{\boxed{4}}$$

２つの要素がある

$$\boxed{1.101} \times 2^{\boxed{4}}$$

仮数部
フラクション
(マンティッサ)

指数部
エクスポーネント

2つの要素がある

$\boxed{1.101} \times 2^{\boxed{4}}$

仮数部
フラクション
(マンティッサ)

指数部
エクスポーネント

$\boxed{1.1010000} \times 2^{\boxed{4}}$

4	1.10100000

東邦大学

もう一工夫されている

$\boxed{1.101} \times 2^{\boxed{4}}$

0	10000011	10100000 … 000
	8ビット	23ビット

東邦大学

もう一工夫されている

$\boxed{1.101} \times 2^{\boxed{4}}$ → 負も表す必要

2^{-4}とか

0	10000011	10100000 … 000

$\underbrace{10000011}_{8ビット}$ $\underbrace{10100000 \cdots 000}_{23ビット}$

東邦大学

もう一工夫されている

$\boxed{1.101} \times 2^{\boxed{4}}$ → 負も表す必要

⇩

下駄をはかす

(単精度) 8ビットなら+127　-128～127 ⇒ 0～255

(倍精度)11ビットなら+1023　-1024～1023 ⇒ 0～2047

0	10000011	10100000 … 000

$\underbrace{10000011}_{8ビット}$ $\underbrace{10100000 \cdots 000}_{23ビット}$

東邦大学

もう一工夫されている

$1.101 \times 2^{\boxed{4}}$ → 負も表す必要

⇩

下駄をはかす

(単精度) 8ビットなら+127　-128～127 ⇒ 0～255
(倍精度)11ビットなら+1023　-1024～1023 ⇒ 0～2047

⇩

(単精度) 4+127 = 131

131

0	10000011	10100000 … 000

8ビット　23ビット

もう一工夫されている

$\boxed{1.101} \times 2^{4}$ → 負も表す必要

⇩

下駄をはかす

(単精度) -128～127 ⇒ 0～255
(倍精度) -1024～1023 ⇒ 0～2047

⇩

(単精度) 4+127 = 131

131

0	10000011	10100000 … 000

8ビット　23ビット

もう一工夫されている

ここが必ず１になる
ように2^nを調整する
（正規化と呼ぶ）

$\boxed{1.101} \times 2^{\boxed{4}}$

$\boxed{1.}101$

負も表す必要

下駄をはかす

(単精度) -128～127 ⇒ 0～255
(倍精度) -1024～1023 ⇒ 0～2047

(単精度) 4+127 ＝ 131

131

0	10000011	10100000 ⋯ 000
	8ビット	23ビット

東邦大学

もう一工夫されている

ここが必ず１になる
ように2^nを調整する
（正規化と呼ぶ）

$\boxed{1.101} \times 2^{\boxed{4}}$

$\boxed{1.}101$

必ず１なので
1.は書かなくても
残りだけ書けばよい

負も表す必要

下駄をはかす

(単精度) -128～127 ⇒ 0～255
(倍精度) -1024～1023 ⇒ 0～2047

(単精度) 4+127 ＝ 131

131

0	10000011	10100000 ⋯ 000
	8ビット	23ビット

東邦大学

もう一工夫されている

ここが必ず１になる
ように2^nを調整する
（正規化と呼ぶ）

1.101×2^{4}

$\boxed{1.}101$

必ず１なので
1.は書かなくても
残りだけ書けばよい

負も表す必要

下駄をはかす

(単精度) -128～127 ⇒ 0～255
(倍精度) -1024～1023 ⇒ 0～2047

(単精度) 4+127 ＝ 131

131

符号		
0	10000011	10100000 … 000
	8ビット	23ビット

東邦大学

もう一工夫されている

ここが必ず１になる
ように2^nを調整する
（正規化と呼ぶ）

1.101×2^{4}

$\boxed{1.}101$

必ず１なので
1.は書かなくても
残りだけ書けばよい

負も表す必要

下駄をはかす

(単精度) 8ビットなら -128～127 ⇒ 0～255
(倍精度)11ビットなら -1024～1023 ⇒ 0～2047

(単精度) 4+127 ＝ 131

131

単精度
完成

符号		
0	10000011	10100000 … 000
	8ビット	23ビット

東邦大学

単精度・倍精度

単精度・倍精度

単精度浮動小数　～　float型

符号	指数部	仮数部
0	10000011	10100000 ··· 000
1ビット	8ビット	23ビット

32ビット

単精度・倍精度

単精度浮動小数　～　float型

符号	指数部	仮数部
0	10000011	10100000 … 000
1ビット	8ビット	23ビット

32ビット

倍精度浮動小数　～　double型

符号	指数部	仮数部
0	10000011	10100000 … 000
1ビット	11ビット	52ビット

64ビット

では問題です

下記の単精度浮動小数はいくつか？

符号	指数部	仮数部
0	10000000	01000000 … 00
1ビット	8ビット	23ビット

では問題です

下記の単精度浮動小数はいくつか？

符号	指数部	仮数部
0	10000000	01000000 … 00
1ビット	8ビット	23ビット

隠れている１．を付け加えて

$1.01_2 = 1.25_{10}$

では問題です

下記の単精度浮動小数はいくつか？

符号	指数部	仮数部
0	10000000	01000000 … 00
1ビット	8ビット	23ビット

隠れている１．を付け加えて

1.01 = 1.25

128からゲタの127を引いて

128－127 ＝ 1 つまり 2^1

では問題です

下記の単精度浮動小数はいくつか？

符号	指数部	仮数部
0	10000000	01000000 ··· 00
1ビット	8ビット	23ビット

隠れている１．を付け加えて
$1.01_2 = 1.25_{10}$

128からゲタの127を引いて
$128-127 = 1$ つまり 2^1

$1.25_{10} \times 2^1 = 2.5_{10}$　（符号は＋）

浮動小数点の仕組みが分かりましたか？

○

×

次へ