★前回の課題の解答

Q1. Which of the following decimal values is equivalent to a hexadecimal fraction 0.B1?

a) $2^0+2^{-2}+2^{-3}+2^{-7}$　　b) $2^0+2^{-3}+2^{-4}+2^{-8}$

c) $2^{-1}+2^{-3}+2^{-4}+2^{-7}$　　d) $2^{-1}+2^{-3}+2^{-4}+2^{-8}$

16進数$(0.B1)_{16}$は、2進数に変換すると、

$$(0.B1)_{16}=(0.10110001)_2$$

つまり、

$$(0.10110001)_2=1\times2^{-1}+1\times2^{-3}+1\times2^{-4}+1\times2^{-8}$$

したがって解答は、(d)です。

④-1　丸め誤差（rounding error）

・コンピュータは、正確な計算ができる機械です。しかし、その計算能力は有限の計算量、有限の計算時間に限られています。計算結果が決められたケタを超えた場合は、範囲外の計算結果は「切り捨て（round down）」または「四捨五入（round off）」します。

＜例＞10進数$(0.1)_{10}$を8ケタの2進数に変換する。

$(0.1)_{10}=(0.\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ \ldots)_2$

2進数のケタは８ケタまでとなっているので、$(0.00011001)_2$とする。

＜Q＞丸め誤差の例として正しいものをすべて選びなさい。

(1)　円周率πを４桁まで計算して$(3.142)_{10}$とした。
(2)　$1\div 3=0.333\cdots$の計算処理を途中で中断したら$(0.33)_{10}$であった。
(3)　$12.112-12.110=0.002$となり有効桁数は1ケタになる。
(4)　10進数$(300)_{10}$を8ケタの固定小数点数に変換したらオーバーフローした。

④-2 打ち切り誤差（truncation error）

・コンピュータの計算処理の結果が、循環小数（recurring decimal）であったり、無理数（irrational number）であったりすると、処理の途中で中断をすることになる。その場合、計算結果は正確ではなく誤差を含む。

＜例＞2の平方根（）の場合

$(\sqrt{20})_{10} = (1.4142135623\ldots)_{10}$

計算処理を3ケタで打ち切るのであれば、1.41となる。

④-3 情報落ちによる誤差（information omission error）

・浮動小数点数の足し算・引き算において、絶対値の非常に大きい値に対して非常に小さい値の影響が反映されないことによる誤差です。

＜例＞10進数の足し算

$0.123 \times 10^5 \quad + 0.123 \times 10^{-3} = 0.123 \times 10^5 + 0.00000000123 \times 10^5 = 0.12300000123 \times 10^5$

計算の有効数字が3ケタであれば、0.123×10^5となる。

④-4 桁（ケタ）落ち誤差（cancellation of significant digits）

・値がほとんど同じ浮動小数点数同士の引き算において、計算結果の有効桁数が少なくなって今うこと。

＜例＞10進数$(15.441)_{10}$と$(15.438)_{10}$の引き算。

$$15.441 - 15.438 = 0.003$$

もともと有効桁数5ケタで有ったが、結果は1ケタとなる。

＜Q＞けた落ちの説明として，適切なものはどれか。
ア　値がほぼ等しい浮動小数点同士の減算において，有効けた数が大幅に減ってしまうことである。

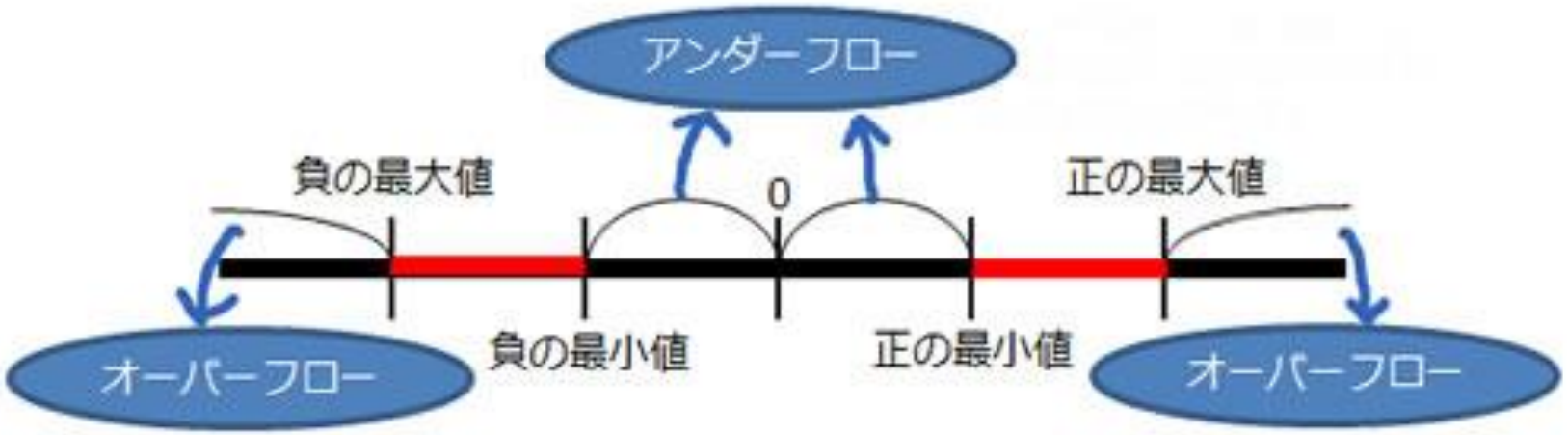
イ　演算結果が，扱える数値の最大値を超えることによって生じる誤差のことである。

ウ　数表現のけた数に限度があるとき，最小のけたより小さい部分について四捨五入，切上げ又は切捨てを行うことによって生じる誤差のことである。

エ　浮動小数点の加算において，一方の数値の下位のけたが結果に反映されないことである。

④-5 オーバーフロー、アンダーフロー（overflow, underflow）

・計算結果が表現可能な範囲を超えてしまうときに発生する誤差です。



＜Q＞次の選択肢のうち、10進数の計算13×17の結果を2進数8ビット（符号部無し）で表したものはどれですか？

A.11011101

B.『オーバーフロー』が起きてしまうため表せない

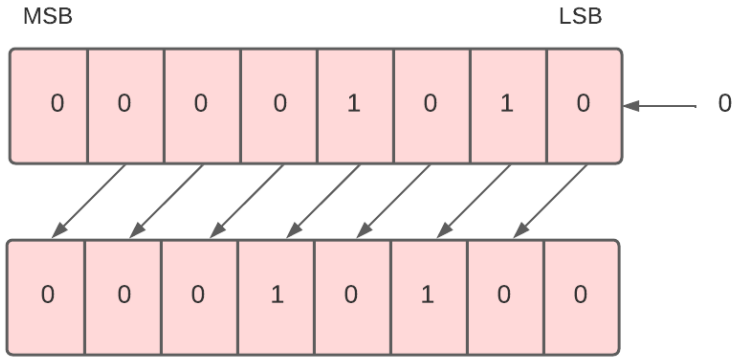
C.11010011

D.01110011

⑤-1　論理シフト（logical shift）

・シフト演算は、数値のビット列を左右にずらす塩酸のことです。論理シフトはビット列を単純に左右にずらします。その結果、MSBやLSBからはみ出した値は捨てて、空いたケタには0を入れます。

例えば、8桁の論理シフトをおこなう



＜例＞8桁の2進数$(00110110)_2$を左右に2ビットずらす論理シフトをおこないなさい。

左に2ビット論理シフト　$(11011000)_2$です。

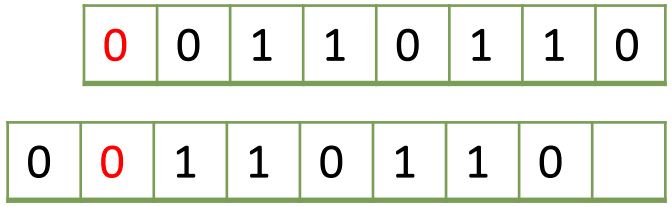
右の2ビット論理シフト　$(00001101)_2$です。

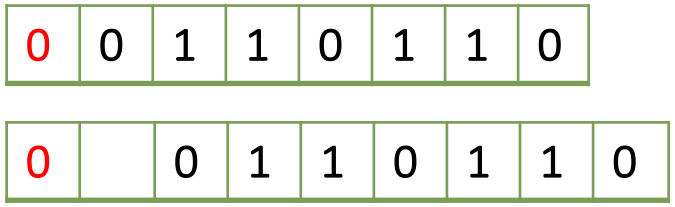
＜例＞ 13×8 の計算を8桁の2進数で論理シフトを使って求めなさい。

＜Q＞ 13×17 の計算を8桁の2進数で論理シフトを使って求めなさい。

⑤-2　算術シフト（arithmetic shift）

・符号つき固定小数点数では、MSBが符号ビットであり、論理シフトによって符号ビットが捨てられると元の符号が変わってしまう。算術シフトは、符号ビットであるMSBが変化しないようにシフト演算を行う方法です。





左算術シフトの場合：MSB（Most Significant Bit）はそのまま動かさずに、その他のビットを動かす。開いたところには0を入れます。

右算術シフトの場合：MSB（Most Significant Bit）はそのまま動かさずに、その他のビットを動かす。開いたところには符号ビットの値と同じものを入れます。

＜Q＞8ビットの2進数 11010000 を右に2ビット算術シフトしたものを,00010100 から減じた値はどれか。ここで,負の数は2の補数表現によるものとする。

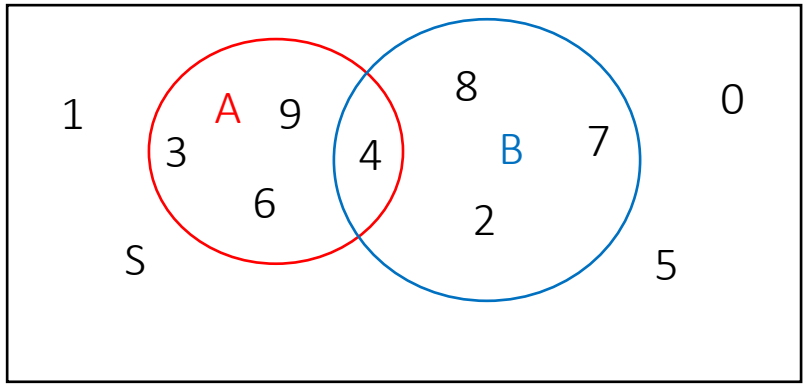
ア　00001000
イ　00011111
ウ　00100000
エ　11100000

⑥-1 集合（set）

・集合とは、ある条件を満たし、他と区別できる、有限または無限のものの集まりのことです。

集合$S=\{0,1,2,3,4,5,6,7,8,9\}$、$A=\{3,4,6,9\}$、$B=\{2,4,7,8\}$とする。

①集合を表すベン図（Venn diagram）



・AとBはSの部分集合　　$A \subseteq S$　$B \subseteq S$

・集合の和（和集合）2つの集合のすべての要素を含む集合　$A \cup B$
・集合の積（積集合）2つの集合に共通する要素の集合　　$A \cap B$
・集合の否定（補集合）集合Aに含まれない要素をもつ集合　$\bar{A}$
・集合の差（差集合）一方の集合から他方の集合を引いた集合　$A-B=A\cap\bar{B}$
・対集合　差集合$(\boldsymbol{A-B})$と$(\boldsymbol{B-A})$の和集合　$\boldsymbol{A\triangle B=(A-B)\cup(B-A)=(A\cap\bar{B})\cup(\bar{A}\cap B)}$
・直積集合　2つの集合から1つずつ取り出した要素の組による集合

$\boldsymbol{A\times B=\{(3,2),(3,4),(3,7),(3,8),(4,2),(4,4),(4,7),(4,8),(6,2),(6,4),(6,7),(6,8),(9,2),(9,4),(9,7),(9,8)\}}$

・べき集合　その集合のすべての部分集合をふくむ集合

$2^A=\{\phi,(3),(4),(6),(9),(3,4),(3,6),(3,9),(4,6),(4,9),(6,9),(3,4,6),(4,6,9),(6,9,3),(9,3,4),(3,4,6,9)\}$

⑥-2　論理演算

・論理和（OR）AとBのいずれかが1であれば結果は1である　$A+B$　$A \vee B$
・論理積（AND）AとBがともに1のときだけ結果は１である　$A \cdot B$　$A \wedge B$
・否定（NOT）論理値の否定　$\bar{A}$
・否定論理和（NOR　NOT OR）論理和の否定　$\overline{A+B}, \overline{A \vee B}$
・否定論理積（NAND　NOT AND）論理積の否定　$\overline{A \cdot B}$　$\overline{A \wedge B}$
・排他的論理和（XOR　EOR　EXOR）AとBが異なるときだけ結果が1になる　$\bar{A}B + A\bar{B}$　$A \oplus B$　$\bar{A} \wedge B \vee A \wedge \bar{B}$

＜Q＞XとYの否定論理積 X NAND Yは，NOT(X AND Y)として定義される。X OR YをNANDだけを使って表した論理式はどれか。

ア. ((X NAND Y) NAND X) NAND Y

イ. (X NAND X) NAND (Y NAND Y)

ウ. (X NAND Y) NAND (X NAND Y)

エ. X NAND (Y NAND (X NAND Y))

⑥-3 ビット演算（bitwise operation）

・ビット単位での論理演算のことです。

・マスク演算　データをビット単位で保持、消去する演算のことです。たとえば8ビットのデータの上位4ビットを0、下位4ビットを1にするには、

$$(11010011)_2 \cdot (00001111)_2 = (00000011)_2$$

ここで$(\mathbf{00001111})_2$はビットマスクと呼ぶ。

・ビットの反転とクリア　ビット反転は各ビットごとに1との排他的論理和を求めます。クリアには、そのビットと同じ値で排他的論理和を求めます。

$$(00011100)_2 \oplus (11111111)_2 = (11100011)_2 \quad \text{ビット反転}$$

$$(01001110)_2 \oplus (01001110)_2 = (00000000)_2 \quad \text{クリア}$$

＜Q＞8ビットで表される符号なし2進数xが16の倍数であるかどうかを調べる方法として，適切なものはどれか。

x と2進数 00001111 のビットごとの論理積をとった結果が0である。
x と2進数 00001111 のビットごとの論理和をとった結果が0である。
x と2進数 11110000 のビットごとの論理積をとった結果が0である。
x と2進数 11110000 のビットごとの論理和をとった結果が0である。