■目次（**ほげ** が今回の範囲）

「コンピュータはなんでも 0 と 1 で表して計算している」ということは多くのひとが知っているだろう．でも実際のところ，内部では何が起こっているのだろう？ 第 1 部では，コンピュータと情報処理の世界への第 1 歩として，「整数を 0 と 1 で表して加算・減算する仕組み」を理解することを目標とする．この第 2 回の授業では，高校数学で学んだ 2 進法の話を復習し，0 か 1 を入れると 0 か 1 が出てくる「論理回路」というものについて学ぶ．コンピュータの中核は，この論理回路によって構成されている．

★ 1　p 進法による数の表現

★ 1.1　ビット列 — 0 と 1 のならび

コンピュータの内部では，情報を表現するのに 2 種類の値のみを用いる (☆ 1)．前回の授業では◯と●を使っていたが，人間の見慣れた数の表現と対応させた方が分かりやすいので，一般的には 0 と 1 を用いる．このとき，0 と 1 を並べたものを**ビット列**または**ビットパターン**という．

101010001

はビット列の例である．

0 または 1 を合わせて n 個並べてできたビット列があったとき，この n のことを**ビット長**といい，単位**ビット (bit)** (☆ 2) をつけて数える．上の例のビット列のビット長は 9 ビットである．

☆ 1) それでどうやって（10 進数の）数を表現したり計算したりするのかを学ぶのがこの授業の当面の目標．「ほな文字や音声，画像のようなんは？」ちう話はそのうち．

☆ 2) ビットという単位は，情報の量を測るために用いられる．いずれまた登場します．

★ 1.2　p 進法

ここでは，0 以上の整数に限定して話をすすめる．負の数についてはいずれまた．

★ 1.2.1　10 進法と 2 進法

■10 進法 (☆ 3) 人間の世界で一般的なやり方では，10 進法で位取り表示を用いる．これは，0 から 9 までの 10 種類の記号を一列に並べるものである．例えば，$\boxed{0}\boxed{2}\boxed{0}\boxed{3}\boxed{4}$ という表記 (☆ 4) では，左端が最上位の桁，右端が最下位の桁であり，

$$\boxed{0}\boxed{2}\boxed{0}\boxed{3}\boxed{4} = 0 \times 10^4 + 2 \times 10^3 + 0 \times 10^2 + 3 \times 10^1 + 4 \times 10^0$$

となる．つまり，$\boxed{0}\boxed{2}\boxed{0}\boxed{3}\boxed{4}$ と表される数は，$10^4 = 10000$ が 0 個，$10^3 = 1000$ が 2 つ，$10^2 = 100$ が 0 個，$10^1 = 10$ が 3 つ，$10^0 = 1$ が 4 つでできている．

Q1. 箱の数が n 個，すなわち n 桁の場合，10 進法で表せる最大の数はいくつか．また，全部で何通りの数が表せるか．

☆ 3) 英語では decimal numeral system または単に decimal という．10 進数は decimal number.

☆ 4) 箱は分かりやすくするために付け加えたものである．また，一般的には最上位の 0 は省略して書かないことが多いが，ここでは省略せずに書いている．

■2 進法 (☆ 5) 上記で 10 が出てきたところを 2 に置き換えると，2 進法での数の表記を考えることができる．2 進法では，0 と 1 の 2 種類の記号を並べる．例えば (☆ 6)

$$\boxed{0}\boxed{1}\boxed{0}\boxed{1}\boxed{1}_{(2)} = (0\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0)_{(10)}$$
$$= 11_{(10)}$$

であるから，この数は 10 進法の $2^4 = 16$ が 0 個，$2^3 = 8$ が 1 つ，$2^2 = 4$ が 0 個，$2^1 = 2$ が 1 つ，$2^0 = 1$ が 1 つでできており，10 進表記の 11 に等しい．

Q2. 箱の数が n 個，すなわち n ビット（n 桁）の場合，2 進法で表せる最大の数はいくつか．また，全部で何通りの数が表せるか．

☆ 5) 英語では binary numeral system または単に binary.

☆ 6) 「$101_{(p)}$」のような表記は，この 101 が p 進法で表された数であることを示している．文脈から明らかな場合は省略する．時給 $1000_{(2)}$ 円のバイトと $1000_{(8)}$ 円のバイトならどっちがいい？

★ 1.2.2　p 進法

上記の話を一般化すると，p を 2 以上の整数として，p 通りの記号を並べて数を表現する「p 進法」を考えることができる．p 進法は，

「p を底 (☆ 7) としてそのべき乗の数を基準に数を表す方法」

である (☆ 8)．5 桁の p 進表記の数は

$$\boxed{}\boxed{}\boxed{}\boxed{}\boxed{}_{(p)} = (\bigcirc\times p^4 + \bigcirc\times p^3 + \bigcirc\times p^2 + \bigcirc\times p^1 + \bigcirc\times p^0)_{(10)}$$

の右辺の 10 進表記の数に等しい．ここで，左辺の箱一つには p 通りの記号が入り，右辺の丸一つにはその記号が表す数を 10 進表記したものが入る．$2 \le p \le 10$ の場合は，箱の中に 10 進数と同じ数記号（$0, 1, \ldots, 9$）を使えばよいが，$p \ge 11$ の場合は記号が足りないので，工夫が必要となる（★ 1.2.4 参照）．

☆ 7) この「底」は対数の話で出てくるのと同じもの．

☆ 8) 60 秒で 1 分, 60 分で 1 時間とする時間の表現は，60 進法表記そのものではない（60 種類の記号を用いるわけではないので）が，その考え方に基づくものといえる．

★ 1.2.3　10 進表記と 2 進表記の間の変換

■10 進から 2 進へ　10 進法で表された数を 10 で割って余りを記録することを繰り返すと，10 進表記の各桁の数を取り出すことができる．では，2 で割って余りを記録していくとどうなるだろう?

Q3. 10 進法で表された数 337 を 10 で割って余りを求める計算を繰り返し，得られる余りを順に答えなさい．

Q4. 10 進法で表された数 337 を 2 で割って余りを求める計算を繰り返し，得られる余りを順に答えなさい．

■2 進から 10 進へ (☆ 9)

Q5. 2 進法で表された数 101010001 に対して，最上位の桁から最下位の桁に向かって順に，$x = 0$ からはじめて

「現在の x の値を 2 倍して，現在の桁の数を加える」

という計算を繰り返すと，最終的な x はいくつになるか．

Q6. 2 進法で表された数 101010001 に対して，最下位桁の値の 1 倍，その上位桁の 2 倍，その上位桁の 4 倍，その上位桁の 8 倍，⋯，の値を計算し，それらの和を求めなさい．

☆ 9) このパラグラフでもそうですが，2 進表記の数を扱うことの多いコンピュータの世界では，$2^4 = 16, 2^8 = 256, 2^{10} = 1024, 2^{16} = 65536$ などの数が頻繁に登場します．いちいち計算せずに済むように覚えておこう．

★ 1.2.4　16 進法

2 進法による数の表記は，人にあまりやさしくない．そこで，コンピュータに関わる世界では 16 進法を使うこともある (☆ 10)．ただし，0 から 9 までの記号では足りないので，10 進表記の 10 から 15 までを A から F までの英字で表すことが多い．例：

$$\boxed{\mathrm{E}}\boxed{2}\boxed{9} = 14 \times 16^2 + 2 \times 16^1 + 9 \times 16^0 = 3625$$

Q7. ★ 1.2.3 で説明されたやり方を応用して 10 進表記の 3625 の 16 進表記を求め，上記に一致することを確かめなさい．

Q8. 10 進数の数を適当に考え，それを 16 進法で表しなさい．さらに，それを 10 進法に変換しなさい．正しい計算ができれば，当然元に戻るはずである．

☆ 10) 英語では hexadecimal. $2^4 = 16$ だから，2 進表記の 4 ビット毎に区切ったものをちょうど 1 桁で表すことになる．例えば 16 進数 E29 は，$\mathrm{E} = 1110, 2 = 0010, 9 = 1001$ だから，

111000101001

となる．2 進数より 16 進数の方がちょっとは人にやさしい⋯かな？

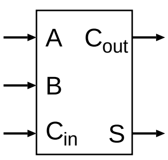
★ 2　論理回路と加算器

★ 2.1　論理回路と真理値表

右の図と表は，前回登場した，2 つの 2 進数の加算を行う仕掛けに用いる部品と，その入力–出力の対応を表にまとめたものである．このように入出力が 2 種類の値で表される仕掛けのことを**論理回路** (☆ 11) といい，その入出力をまとめた下のような表を**真理値表** (☆ 12) という．

☆ 11) 何が「論理」なのかはいずれまた．

☆ 12) 何が「真理」の値なのかはいずれまた．



A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

前回の授業で説明したように，上図の論理回路を複数組み合わせると，2 進数で表された 2 つの数の和を求める論理回路を作ることができる．このことから想像できるように，コンピュータ（の中核をなしている部分）の正体は，非常に複雑な論理回路である．

上図の論理回路は 2 つの数の和を求める論理回路の部品となるが，この論理回路自身も，もっと単純な論理回路を部品として組み上げることができる．そのことを実際に示すのは次回にまわして，今回はもう少し簡単な例で感じをつかんでみよう．

NOT ゲートの真理値表

A	X
0	1
1	0

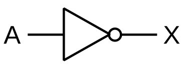
AND ゲートの真理値表

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

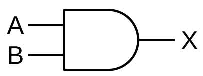
OR ゲートの真理値表

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

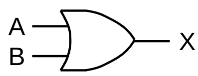
NOT ゲートの図記号



AND ゲートの図記号



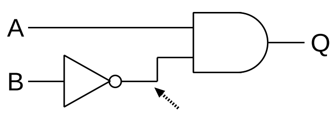
OR ゲートの図記号



上記の 3 つの表と 3 つの図は，論理回路を構成する部品となる**論理ゲート**のうち，最も単純な **NOT ゲート**，**AND ゲート**，**OR ゲート**の 3 つの真理値表と論理回路の図を描くための図記号を示している (☆ 13)．ここでは，A や B が入力，X が出力を表している．例えば，A に 1 を，B に 0 を入力された AND ゲートの出力 X は 0 となる．

☆ 13) NOT, AND, OR などの意味についてはいずれまた．

Q9. 下図は，A, B にそれぞれ 0 または 1 の値が入力されると，0 または 1 の値 Q を出力する論理回路の図である（矢印は説明のために描いたもので，回路図の一部ではない）．下の表を埋めることで，この回路の真理値表を求めなさい．ただし，表の P は，図の矢印の部分の値（NOT ゲートの出力）を表すものとする．



A	B	P	Q
0	0		
0	1		
1	0		
1	1		