第5章

制御文やデータの型，演算子などC言語の文法

絶対必要！ C言語の基礎の基礎

ここではC言語の基本について解説する．定数や変数，演算子などの基本から制御構造，関数に至るまで，C言語を使う上で重要になる概念について解説する．（編集部）

三好 健文

第2章でC言語でプログラムを書くメリットと手法について学びました．C言語を使うことで，プロセッサごとに異なる機械語の命令やデータ形式を，私達はあまり意識せずにプログラミングできます．

この章では，C言語で自由にプログラムを行うために絶対に必要となるC言語における数値の表現方法，データ型や演算子，制御構文などを中心に説明します．

1．C言語におけるデータの取り扱いをマスタしよう

● 数字と文字の表し方

C言語の定数，すなわちプログラム中にあらかじめ埋め込むことのできる値には，数字と文字があります（**表1**）．

数字は，10進数，8進数，16進数で書くことができます．10進数は，10でけたが繰り上がるおなじみの数です．8進数と16進数は，それぞれ8と16でけたが繰り上がる数です．このけたが上がり下がりする数を「基数」といいます．

表1に10進数，8進数，16進数で記述した「65」という数を示します．8進数では頭に「0」を付け，16進数は，頭に「0x」を付けて記述します．16進数では，16でけたが上がるため，各けたは，1～15までの数を表現する必要があります．そのため，0～9までは10進数と同じ数字を用い，10～15までを表現するにはアルファベットのA，B，C，D，E，Fを用います．Cプログラム中で数を表現する場合には大文字でも小文字でもかまいません（0xFも0xfも共に15を意味する）．

16進数の基数である16は，（プロセッサの動作の基本となる2進数の基数である）2の4乗であるため，プロセッサ内のビット・データの表現に適しています．そのため頻繁に利用されるので，特にA～Fの数の表現に慣れ親しむとよいでしょう．

文字は，'A'などのように「'」で囲んで記します．プロセッサは文字というデータを取り扱うわけではなく，ASCIIコードに従う8ビットの数として取り扱われます．単なる数なので，例えば，プログラム中に書いた'A'と，0x41（'A'のASCIIコードの値）は，全く同じです（**表1**のビット例を参照）．

また，C言語では文字列を記述することもできます．これは，"ARM"などのように「"」で囲んで表現します．このように記述した文字列は，連続するメモリ領域に'A'，'R'，'M'，'¥0'（NULL文字．ASCIIコードで0）として保存されます．

表1　定数表現の例（65の場合）

		記述例	ビット列（8ビットの場合）
数値	10進数	65	01000001
	8進数	0101	01000001
	16進数	0x41	01000001
文字		'A'	01000001

● 変数を利用する

プログラム実行中に計算によって決定した値をほかの計算のために一時的に保存しておきたいということがよくあります．C言語では，プログラムの実行中に決定される値を，名前を付けた「変数」として利用できます．

変数名として，

1文字目は，A～Z，a～zまでのアルファベット

KEYWORD ―― 定数，変数，型，配列，演算子，制御構造，関数，プリプロセス，最適化

あるいは「_」

2文字目以降は，A～Z，a～zまでのアルファベット，「_」あるいは数字

を用いることができます．

ただし，予約語として，C言語で定義されている名前は付けられません．また，先頭が「_」という名前の変数は，OSやそのほかのライブラリなどで使用されるケースも多いため，避ける方がよいでしょう．**図1**にC言語の規格であるC99で定義されている予約語の一覧を示します．

変数を用いることで，プログラム実行中に値を変数に代入したり，変数から値を読んだりできます．しかし，変数を使用するためには，あらかじめ使用することを「宣言」しておかなければなりません．

変数を宣言するには「型」と呼ぶ種類を決めておく必要があります．型には，C言語の言語規約で決められているプリミティブ型と，プログラマが作成できる構造体による型があります．型は，自由に名前を付けられます．型を伴って変数を宣言することで，その変数を利用するため，つまり値を書いたり読んだりするために必要な領域を，メモリ上の適切な位置に確保します．

例えば，int型の名前aの変数を宣言するためには，

```
int a;
```

と記述します．ここで，同じ型の変数を複数定義する場合には，コンマで変数名を複数並べて記述できます．例えば，int型の名前aと名前bの変数を定義する場合には，

```
int a, b;
```

のように記述できます．

● プリミティブ型

プリミティブ型はC言語の規約によって定められている型です．例を**表2**に示します．ここでは各型のサイズとビット幅，および表現可能な数の範囲を示しています．この値はC言語の処理系，すなわちコンパイラなどによって異なります．この例は，IARコンパイラの例にすぎないことに注意してください．

```
_Bool, char, short, int, long, signed, unsigned,
float, double, _Complex, _Imaginary, struct, union,
enum, volatile, const, restrict, auto, extern, static,
register, typedef, void, if, else, switch, case,
default, for, while, do, goto, continue, break, return,
inline, sizeof
```

図1　C言語の予約語（C99で規定されているもの）

表2　C言語の型とARMプロセッサにおけるサイズ

型　名	ビット幅	表現できる値
char	8	0～255
signed char	8	－128～127
unsigned char	8	0～255
short	16	－32768～32767
unsigned short	16	0～65535
int	32	-2^{31}～$-2^{31}-1$
unsigned int	32	0～$-2^{32}-1$
long	32	-2^{31}～$-2^{31}-1$
unsigned long	32	0～$-2^{32}-1$
long long	64	-2^{63}～$-2^{63}-1$
unsigned long long	64	0～$-2^{64}-1$
float	指数8 仮数23	±1.18E－38～ ±3.39E＋38
double	指数11 仮数52	±2.23E－308～ ±1.79E＋308

表2には，signedやunsingnedが付いている型があります．signedやunsingnedはビット列で表現された数を，正負を示す符号付きの数として見るか，そうではなく無条件に正の数と見なすかを指示するものです．これらを型修飾子といいます．IARコンパイラではchar型以外のプリミティブ型で型修飾子がない場合には正負を示す数として取り扱います．またchar型で型修飾子がない場合，正の数だけとして扱います．

図2に正負を持つsingned型のデータと正負を持たないunsigned型の，データの数としての表現方法を示します．最も大きなけたのビットを正負を示すために用いています．そのため，正負を持つ数として変数を利用する場合，表現できる数の絶対値の大きさが正の数のみとして扱う場合に比べて約半分になります．

変数は，プログラムを実行するために，その変数値が十

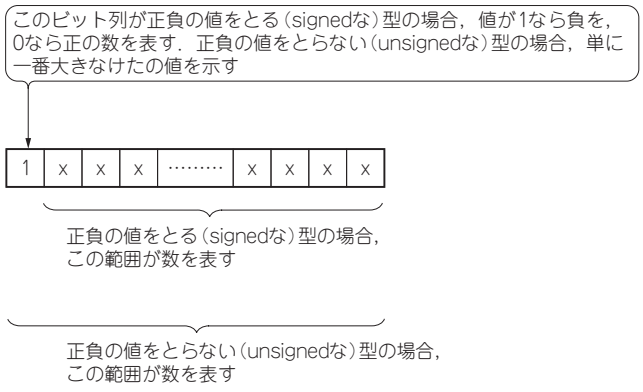

図2　正負のある場合と，ない場合の数の扱い