変数、入出力処理、演算子

ここまでにC言語プログラミングの様子を知ってもらうため、printf文、変数、scanf文、if文を使った簡単なプログラムを紹介した。今回は変数の詳細について習い、それに併せて使い方が増える入出力処理の方法を習う。また、演算子についての復習と供に新しい演算子を紹介する。

変数の宣言

プログラムでデータを取り扱う場合には対象となるデータを保存する必要がでてくる。このデータを保存する場所のことを「変数」と呼び、変数を説明する上ではよく「データを入れておく箱」として例えられ、必要に応じて中身（データ）は後から自由に変更することが可能である。

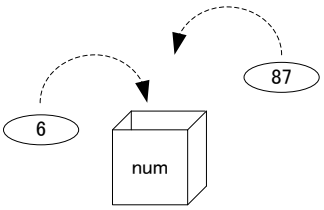


図１　変数

変数は以下のように宣言を行うことで利用する。

書式
変数型（データ型）　変数名;

使用できる変数名

変数名の付け方はプログラミング言語によって異なり、C言語では以下の規則がある。

（1）最初の文字は「英字」か「下線 (_)」を使う
（2）２文字目以降は「英字」、「数字」、「下線」を使う
（3）英字の大文字と小文字は区別される
（4）表１に示す予約語（予約済み識別子）は使えない
（5）変数名の長さに制限はないが、最初の３１文字で区別される。

表１　予約語

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

変数名の途中に英字、数字、下線以外があってはならず、空白も認められない。予約語そのものはだめだが、変数名の一部に使うことはできる（例えば、a_forなど）。

変数型の種類

C言語では変数を用いるときに、変数型を指定して宣言する。変数型には基本型として以下の４つ用意されている。

表２　基本型

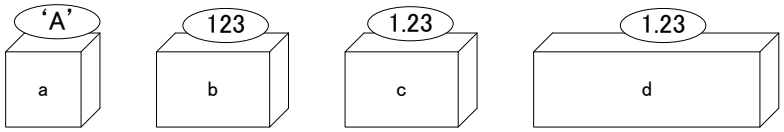
型名	説明	一般的な使用ビット数
char	文字型（小さな整数）	8ビット
int	整数型	32
float	浮動小数点型	32
double	倍精度浮動小数点型	64

「char」、「int」、「float」、「double」の変数型はそれぞれ「文字あるいは小さな整数」、「整数」、「実数」、「より精度の高い（数値のより大きい、より小さい）実数」を扱う時に用いる。

備考：
基本型の説明に示した「浮動小数点」とは数値を指数表現で表す方法である。例えば 0.001 は指数表現で書くと1.0×10^{-3}と表すことができる。これを C 言語では 1.0E-3 というように表す。この 1.0 を仮数部、-3 を指数部と呼ぶ。コンピュータではこの仮数部と指数部の値を変数の領域に分割して格納している。

それぞれの型はコンピュータのメモリ上にデータを格納する領域の大きさが異なる。それが「一般的な使用ビット数」で示した数であるが、これは図２に示すように「データを格納する箱のサイズ」が異なると考えておけばよい。

```
// 変数の宣言
char a;    // 文字型
int b;     // 整数型
float c;   // 浮動小数点型
double d;  // 倍精度浮動小数点型
```



それぞれ用意される箱の大きさが異なる

図２　変数のサイズ

変数型の応用

変数型の応用として、型の範囲を明確に定義するための「修飾子」として、short や long がある。これは int 型や double 型の頭につけて、「short　int」や「long　double」のように宣言する。また、変数型の基本型は通常では正と負の両方の値を扱うが、負の値を扱わない整数型（char 型や int 型）を宣言することもできる。その場合には unsigned と呼ばれる修飾子を用いて「unsigned char」のように宣言する。あるいは明確に負の値も扱うという宣言もすることができ、その場合は signed 修飾子を用いて、「signed char」のように宣言する。変数型の一覧を表３に示す。

表３　変数型の一覧

種類	型名	ビット数例	記憶できる値の範囲例	printf()や scanf()での 変換仕様例
整数型	int	32	-2147483648 ～ 2147483647	%d
	unsigned int	32	0～4294967295	%d
	short int	16	-32768～32767	%d
	unsigned short int	16	0～65535	%d
	long int	32	-2147483648 ～ 2147483647	%ld
	unsigned long int	32	0～4294967295	%ld
浮動小数点型	float	32	±3.4E-38～±3.4E+38	%f
	double	64	±1.7E-308～±1.7E+308	%lf
	long double	80	± 1.7E-4932 ～ ± 1.7E+4932	%lf

文字型	char	8	英数字 1 文字　-128～127	%c
	unsined char	8	英数字１文字　0～255	%c

備考：
C 言語の変数型のサイズ（ビット数）は詳細には規定されておらず、変数型の使用ビット数はコンピュータの処理に依存する。そのため、ここで紹介したものは一般的に採用されているビット数である。ただし、float より double の方が精度は高いことは保証されている。

それでは基本型の変数の宣言方法と使用例を以下に示していく。

例１ 文字型	char c; c = 'a';
例２ 整数型	int i; i = 3;

例３
実数型
	float a;
	double b;
	
	a = 1.0;
	b = 3.0;

例４	float f, g;
	
	f = 3.0;
	g = f/4.0;

例５ 初期化	int i = 3;

※例１のように文字型 char はシングルコーテーション（'）で囲んだ文字を代入する。この使い方の例では代入できる文字は英数字１文字だけである。複数の文字列を扱う方法は後で習うことになる。
※例４のように同じ型を持つ変数は float f, g;のようにして、セミコロン（,）で区切ることで１つの文で宣言することができる。
※例５のように変数宣言と同時に値を格納することが出来る。その場合には値の「代入」と言わず、特に「初期化」という用語が用いられる。

標準出力と変数型
printf 文はデータや変数の値を表示することができる。表示するデータの種類（文字、整数、実数など）によって変換仕様（%〇で表されるもの）が異なる。主な変換仕様は以下の通り。

%c　　文字（char 型）
%d　　整数（int 型）
%f　　実数（float 型）
%lf　　より精度の高い実数（double 型）

注意：
printf 文の場合、double 型の変換仕様%lf は%f としても問題なく表示される。
ただし、scanf 文では double 型の変換仕様は%lf だけが利用できる。%f を用いると入力に失敗する。

この変換仕様を使うには以下のように書く。

書式	printf("%〇 … %〇", 変数名あるいは値（式）, …, 変数名あるいは値（式）);

例1	char t = 'a'; printf("%c", t);

例2	int i = 3;
	
	printf("%d", i);

例3	printf("%d %f", 3+4, 3.0/4.0);

例4	int i = 3;
	float f = 3.0;
	double d;
	d = f/4.0;
	
	printf("%d %f %lf", i, f, d);

printf 文は通常、実数を表示すると小数点以下6桁まで表示する。これを変換仕様によって表示桁数を変えることができる。例として3桁の整数ならば「%3d」、小数点以下2桁の実数(float の場合)ならば「%.2f」というように書く。実数の場合には「%7.3f」のように指定することもできる。その場合は以下の例のように、「13.200」の手前にはスペースが1つ表示され、全体で7桁（(.) も1桁と考える)、小数点以下で3桁の表示に調整される。

例	float f = 13.2;
	
	printf("%7.3f", f);

出力	13.200

printf 文と変数を使う上でのいくつかの注意点を述べる。

以下の例のように、変数や型と変換仕様が一致していなければ、正しく表示されない。私のパソコンの環境の場合では0が表示された。

例	float f = 3.0;
	
	printf("%d", i);

出力	0

以下の例のように、整数型変数に実数を代入するとその結果は小数点以下の値を切り捨てた整数が代入される。

例	int i = 3.5; printf("%d", i);
出力	3

以下の例のように、小数点以下の桁数を制限した場合には四捨五入された値が表示される。

例	float f = 3.567;
	
	printf("%.2f", f);

出力	3.57

標準入力と変数型

プログラム実行中にユーザがキーボードから値を打ち込み、その値を用いて計算したり、表示させたりし

たい場合がある。その場合に用いるのが scanf 文である。scanf 文の書式は次の通りである。

書式	scanf("%〇 … %〇", &変数名, …, &変数名);

例1	int i; scanf(“%d” , &i);
例2	int i; float f; double d; scanf("%d %f %lf", &i, &f, &d);

※"%〇 … %〇"の中には「¥n」を書かないで使用すること

scanf 文は変数に標準入力(C 言語ではキーボードが割り当てられている)から入力された値を格納する。そのため使用する変数を先に宣言しておかなければならない。"%〇 … %〇"の中には入力する変数のデータ型に合った変換仕様を指定する必要がある。変数に値を読み込むときには変数名の先頭に&を付けて書く。

scanf 文を使う上でよくやってしまう間違いを述べる。
以下の例のように、データ型と変換仕様が一致していなければ、正しく入力されない。

例1	float f; scanf("%d", &f);
例2	double d; scanf("%f", &d);

※double 型の変換仕様は「%f」ではなく「%lf」である。

それでは、変数型の基本型である文字、整数、実数の変数を用いて、キーボードから入力された値を出力するプログラム例を示す。

プログラム例1

```
/* 文字と整数をそれぞれ1つ、実数を2つ入力し、表示するプログラム*/
#include <stdio.h>

int main(void)
{
        char a;
        int b;
        float c;
        double d;

        printf("文字を1つ入力してください：");
        scanf("%c", &a);

        printf("整数を入力してください：");
        scanf("%d", &b);

        printf("実数1を入力してください：");
        scanf("%f", &c);
        printf("実数2を入力してください：");
```

```
        scanf("%lf", &d);

        printf("¥n入力された文字は %c です¥n", a);
        printf("入力された整数は %d です¥n", b);
        printf("入力された実数1は %f です¥n", c);
        printf("入力された実数2は %lf です¥n", d);

        return 0;
}
```

実行例

```
文字を1つ入力してください : a
整数を入力してください : 5
実数1を入力してください : 1.5
実数2を入力してください : 3.14

入力された文字は a です
入力された整数は 5 です
入力された実数1は 1.500000 です
入力された実数2は 3.140000 です
```

演算子

ここまでに数値や変数を用いた計算方法も紹介した。これは以下のような形で計算し、変数に値を代入することが出来る。

```
a = b + 10;
```

「a」、「b」、「10」のように演算の対象となるものを「オペランド」という。また、「=」や「+」は「演算子」と呼ばれ、「=」は特に「代入演算子」、「+」は「算術演算子」というものに分類される。以下に算術演算子と代入演算子を示す。ただし、代入演算子はここに示したもの以外にもある。

表４　演算子

算術演算子	
+	加算
-	減算
*	乗算
/	除算
%	余り（ただし整数での演算に限る）

代入演算子	
=	代入
+=	加算したものを代入する
-=	減算したものを代入する
*=	乗算したものを代入する
/=	除算したものを代入する
%=	剰余を求め、代入する

※　例えば、「a += b;」は「a = a + b;」と同じ処理となる。

演算子には演算の優先順位がある。括弧を含め、ここに示した演算子の優先順位は以下である。

表５　演算子の優先順位

1	( )
2	*, /, %

3	+, −
4	=, +=, −=, *=, /=, %=

演習　プログラム例１を作成する。**標準出力と変数型**で紹介した表示桁数の変更を行ってみること。

課題１

kadai1-1（30 点）

電圧と電流を入力し、オームの法則を用いて抵抗を計算し、表示するプログラムを作成しなさい。実数の表示については小数点以下１桁で表示すること。今回のプログラムの変数は double 型を用いること。

実行結果例（斜体はキーボードから数値を入力する部分）
抵抗 R(Ω)を求めます 電圧 V(V)を入力してください *12.5* 電流 I(A)を入力してください *0.1* 抵抗 R は ○○(Ω) です

kadai1-2（30 点）

半径 r (cm)の値をキーボードから入力し、半径 r(cm)の円周（cm）と円の面積(平方 cm)を計算し、結果を表示するプログラムを作成しなさい。ただし、以下の条件を満たすようにすること。

・使用する変数は、次の４つ
　　半径
　　円周率
　　円周
　　円の面積

・半径の値はキーボードから入力する。
・円周率 pai はプログラムの中で代入演算子(=)を使う。値は 3.141592 とする。
・実行結果を以下のように表示する。円周と円の面積は 2 度表示しているが、2 度目の表示では全体の桁数を 10 桁、小数点以下の桁数を 4 桁とすること。

半径を入力してください：99 半径 99cm の円の円周：622.035216cm 半径 99cm の円の面積：30790.743192 平方 cm 半径 99cm の円の円周：　622.0352cm 半径 99cm の円の面積：30790.7432 平方 cm

kadai1-3（40 点）

ここまでに printf、scanf、変数、四則演算、if 文について習った。これらを利用し、自分で使用例を考えてオリジナルの処理を行うプログラムを作成しなさい。時間に余裕がある人は複数作成して提出してもよい。ただし、少なくとも一つは作成して提出すること。

このプログラムについては、どのような動作を行うものかを簡単でよいので説明を付けること。

kadai1-3 は習った内容を理解し、作成していれば 25 点。
以下は 40 点の例です。
（1）プログラムにある程度の行数がある。
（2）とても良く応用できている。
（3）複数のプログラムを作成している。