

第 2 回 変数・scanf

今回の講座は変数と `scanf` です。双方ともに非常に重要ですのでしっかりと学んで行きましょう。最初は戸惑うかもしれませんが、そう難しいものではないので、徐々に慣れていきましょう。

○変数

変数とは、数値や文字などを格納するための「箱」という考え方をするとよいでしょう。この箱には様々な種類の「型」があり、その箱(変数)を使うには、箱を使うことを示す宣言が必要です。型を決めることで、箱に物をいれる際の入れ方や、箱そのものの大きさがプログラム上で明示され、コンピュータが処理できるようになるのです。

宣言された変数は、コンピュータのメモリを占有し、変数に数値や文字等のデータを入れる(代入する)際にはメモリ上に記憶されます。ゼミ A の後半で必要になってくる考え方ですので、一応覚えておいてください。

○変数の宣言

さきほど示したように、変数を使うためにはまず変数を宣言する必要があります。変数を宣言するには「変数の型」と「変数名」を記述します。変数の型によって箱に入るものが「整数」なのか「小数」なのか「文字」なのかといったことが決まります。これが先程述べた「物をいれる際の入れ方」にあたります。変数名は、ある一定の規則内であれば、自由につけることができます。ここで一度以下のプログラムを書いてみてください。

```
#include <stdio.h>
int main( void ){
    int num ; /* int 型の変数 num を宣言 */
    float f ; /* float 型の変数 f を宣言 */
    char c ; /* char 型の変数 c を宣言 */
    num = 5 ; /* 変数 num に 5 を代入*/
    printf( "num の値は%d です。¥n" , num ) ; /*変数 num の値を出力*/
    return ( 0 ) ;
}
```

上のプログラムを正しく記入し、コンパイルできれば、「num の値は 5 です。」と表示されます。これは `int` 型の `num` という箱(変数)を宣言してから、変数 `num` に 5 という数値を代入し、それを `printf` で出力して表示させているのです。その他に `float` 型の変数 `f` と `char` 型の変数 `c` を宣言していますが、プログラム内では使っていません。

型	書式指定文字	記憶するデータの種類	値の範囲
int	%d など	符号付き整数	-2,147,483,648～2,147,483,647
long	%d など	符号付き整数	-2,147,483,648～2,147,483,647
float	%f など	小数	6桁分の精度
double	%f など	小数	10桁分の精度
char	%s、%c など	文字(文字列ではない)	-128～127 または 0～255

他にも色々な型があります。また、値の範囲についてはコンパイラ・環境によって異なってくるので、あくまで目安です。他にも `long` は実際には型修飾子であるとか、`char` 型は文字なのに何故値の範囲が数字なのか、など、疑問に思う点が多々あるかと思いますが、現時点では気にしてもしょうがないので、気になる方は自己で調べてください。また、文字列については C 言語では厄介な仕組みとなっているのでこちらも現時点では説明しません。

最後に、変数名の規則について述べます。変数名は小文字・大文字のアルファベット、数字、_(アンダースコア、またはアンダーバー)によって構成されます。変数名の先頭に数字は使えず、_ だけの変数名は使えません。32 文字異常の変数名や、予約語と呼ばれる一部の文字列、関数名、すでに使われている名前は使えません。

○変数の初期値

変数は、宣言した時点では何が入っているかわかりません。何が入っているか分からない変数を使うのは危険な行為です。変数を他の計算に使う前に、その変数に具体的な内容である「初期値」を与えてやる必要があります。また、変数に初期値を与えることを「初期化」といいます。

変数の型 変数名 = 初期値 ;

とすることで、変数の宣言時に初期化することができます。関数内において、変数は最初に宣言する必要があります。気をつけてください。先程のプログラムの一部を改変して、宣言時に値を与える場合次のようになります。

```
int num = 5 ; /* int 型の変数 num を宣言、初期化 */
```

○代入式と計算式

変数には数値を代入することができます。

変数名 = データ ;

上のようにすることで、変数にデータを代入できます。最初のプログラムの「num = 5 ;」もこれにあたります。初期化も代入もそうですが、C 言語での = 記号に「等しい」という意味はありません。= の右にあるデータを左にある変数にコピーする、という意味になります。

次に計算式について述べていきます。計算に使える記号(演算子)には、次のものがあります。

演算子	意味
+	左辺 + 右辺
-	左辺 - 右辺
*	左辺 × 右辺
/	左辺 ÷ 右辺
%	左辺 ÷ 右辺の余り(剰余)
==	同値
!=	非同値

一応==も演算子の一つです。また、==、!=については第3章で触れていきます。記憶の片隅に留めて奥程度でいいので覚えておいてください。これらの演算子を利用して以下のように様々な計算が可能です。

```
int num1 = 10 ;
int num2 = 20 ;
int ans ;
ans = 15 + 25 ; /*ans には 40 が代入される*/
ans = num1 + num2 ; /*ans には 30 が代入される*/
ans = ans + num1 * num2 ; /*ans には 230 が代入される*/
ans = num1 = num2 ; /*ans、num1 に 20 が代入される*/
```

上記のように、定数や変数の間に演算子を挟むことで、計算式をどんどん連結していくことができます。また、このとき数学同様に計算式中で演算子の優先順位が生まれます。掛け算や割り算が優先され、剰余も割り算と同じ優先順位ですので、6 行目では+演算子よりも*演算子が優先され、結果として 230 が代入されます。計算中に ans が入っていますが、右辺の計算が全て終了次第左辺に代入されます。このあたりは演算子の難しい深淵な部分ですので、ここでは触れません。興味がある人は自分で調べてください。演算子の優先順位はやはり数学同様に()を使うことで変更できます。ただし、中括弧や大括弧は使えません。その代わりに、()の中にさらに()を持つことができます。最後の行では非常に難解なことをしています。先程述べたように、=演算子については右辺の計算が全て終了次第、左辺に代入されますので、num2 が num1 に、num1 が ans に代入されます。一応このような書き方もできる、という例ですが、できるだけ避けた方が良いでしょう。

○scanf 関数

scanf 関数はキーボードからの入力を受け取る関数です。数値や文字列の入力を受け取ることができます。前回学んだ画面に文字を出力する printf 関数と対になる関数です。まず、以下のプログラムを見てください。

```
#include <stdio.h>
int main( void ){
    int num ; /*変数 num を宣言*/
    num = 5 ; /*変数 num に 5 を代入*/
    printf( "num の値は%d です。¥n" , num ) ; /*変数 num を出力*/
    return ( 0 ) ;
}
```

先程説明したプログラムです。num に値を入れて表示しています。この num = 5 ; の部分を scanf 関数を用いてキーボードからの入力に書き換えたプログラムが以下のプログラムです。

```
#include <stdio.h>
int main( void ){
    int num ; /*変数 num を宣言*/
    scanf( "%d" , &num ) ;
    printf( "num の値は%d です。¥n" , num ) ; /*変数 num を出力*/
    return ( 0 ) ;
}
```

これを実行すると数字の入力を求められます。そこに適当な数字を入力すると、「num の値は XX です。」と、キーボードから入力した数値が出力されるはずです。Scanf 関数の記述方法は以下の通りです。

```
scanf( "書式指定文字" , &変数名 ) ;
```

書式指定文字とは int 型なら %d、float 型で %f のような、先程変数のところで述べた、データの形式のような物です。%d なら整数、%f なら小数をさします。変数名は代入先の変数の変数名です。

printf 関数を学んだ際にはなかった「&」が変数名の前についています。これは必須です。困ったことになくてもエラーは出ません。しかし実行すると正しく動作しません。この&の意味はまだ理解する必要はありません。ソフトゼミ A の後半の第 7 回「ポインタ」で学びますので、今はこういうものだ、と思っておいてください。

○練習問題

以下のプログラムのソースコードの穴を埋めて、2 つの変数を入力させて、それを足し算し、表示するプログラムを完成させてください。

```
#include <stdio.h>
int main( void ){
    int a ;
    int b ;
    int ans ;

    printf( "2 つの値 a,b を入力してください。¥n" ) ;
    ■■■■ここに適切な 1 行を入れてください■■■■
    ■■■■ここに適切な 1 行を入れてください■■■■

    ans = a + b ;
    printf( "Ans=%d" , ans ) ;
    return ( 0 ) ;
}
```