

ソフトゼミ A 第2回 変数・scanf

■ はじめに

今回は変数と `scanf` をテーマにした講座を行いたいと思います。変数って聞くと難しそうと思うかもしれませんが、まだ序盤ですし少しずつ慣れて理解していきましょう。

また `scanf` は前回やった `printf` と同じくよく使う関数なので、ゆっくりでいいので理解していきましょう。

■ 変数

変数とは、数値や文字列などを格納するための「箱」であると考えてください。

しかし、この箱（変数）には色々な種類の「型」があります。箱（変数）には整数の収納箱、少数の収納箱、文字の収納箱などの種類があります。そして、この箱（変数）を使うときは、箱を使う前にその箱がどんな「型」であるかを宣言しなければなりません。説明だけでは分かりにくいと思いますので、実際に下のプログラムを打ち込んでみましょう。

a02_1.c

```
#include <stdio.h>

int main( void ){

    /* int という型の変数（箱）に num という名前を付けて宣言 */
    int num;

    /* 変数 num に 1 を代入 */
    num = 1;

    /* 変数 num の中身を出力 */
    printf( "num に入れた値は%d です。¥n", num );

    return 0;
}
```

上のプログラムを正確に記入して、コンパイルに成功すると「num に入れた値は1です。」と表示されます。

さて、ここでは `int` 型の変数を宣言して使いましたが、`int` 型の変数とは何でしょうか？
下の表で変数の型について説明しましょう。

データ型名	扱えるもの	値の範囲
<code>int</code> 型	整数	-2,147,483,648～2,147,483,647
<code>char</code> 型	整数（文字として使うことが多い）	-128～127 または 0～255
<code>float</code> 型	実数	精度 6 ケタ
<code>double</code> 型	<code>float</code> 型より精度の高い実数	精度 10 ケタ

※ `char`型の説明のところで、整数が文字だとか書いていますが、ここでは本題からそれてしまうので説明は控えておきます。

この表で示した以外にも他にも色々な型ありますが、ゼミAの範囲では`int` 型、`double` 型をまずは覚えておけばいいと思います。ここで上のプログラムを軽く振り返ると、プログラムの中で使った`int`型の`num`という変数（箱）は、整数を入れるための変数(箱)という意味があったのです。

変数を宣言する時は、 型名と変数名を半角スペースで区切って行います。

```
int foo;
```

上の例では、この宣言によって、`int`型(整数)の「`foo`」という変数を使いますよ、と宣言したことになります。また、

```
int foo = 4;
```

のように、変数の最初の値を宣言時に決定することもできます。くどいかもしれませんが、これで、`int`型の変数「`foo`」を使うと宣言し、この時点で`foo`の値は「4」となりました。

宣言する変数名はある条件さえ守れば、自由に変数名を決めることができます。

条件 1. 「変数名に使える文字は半角英数とアンダースコア(下線, 「`_`」)のみ。」

条件 2. 「キーワード(C の文法上、意味のある言葉(`int`, `return` など))、関数名 (`printf` やこれから習う `scanf`) のように最初から役目のある名前を変数にすることはできません。」

条件 3. 「変数名の最初の文字は数字(0～9)であってはならない。」

(この他にも、アンダースコアのみの名前ではない、アンダースコアから始まる名前の変数は使わない方がいい、など細かい制約があります。)

`printf` や、今回の後半で扱う `scanf` では、「書式指定文字」と呼ばれる文字が使えます。ちょうど、前回の `printf` では整数を出力するのに「`%d`」という文字を使いました。これは、(主に)`int` 型の整数を表示する時に使います。この他、`float` 型の実数を入出力する時には「`%f`」を、`char`

型などの文字を入出力する時には「%c」という書式指定文字を使います。なお、long 型は scanf・printf とともに「%ld」(エル・ディー)を使います。また、double 型は scanf の場合には「%lf」(エル・エフ)を、printf の場合には「%f」を使います。scanf と printf の場合で微妙な違いがあるので注意してください。

■ 演算

さて、変数を学んだついでに変数を扱う演算についても学んでおきましょう。演算で使う記号は数学で使う記号と似たものが多いので、覚えやすいと思います。

記号	数学記号	意味
+	+	足す
-	-	引く
*	×	掛ける
/	÷	割る
%	mod	割り算の余り
=	(無し)	代入
==	=	等しい
!=	≠	等しくない

上の表について順版に説明していきます。まず、四則演算については数学で習った通りに A +B、A*B とすることで、A と B の値を足したり、掛けたりできます。例えば、A+B*C とすれば、数学と同じように、掛け算・割り算のほうが足し算・引き算より優先順位が高いため、B*C から計算します。その後「A」と「B*C」の結果を足します。

なお、int などの整数型同士で割り算をした時に割り切れない(商が整数にならない)場合は、小数点以下は無視されます(商が 0 以上の場合は切り捨て、0 未満の場合は切り上げ)。

次に「=」(代入)について説明していきます。まず下の例を見てください。

```
int a = 1;
int b = 2;
a = b;
```

これをみると、式が成り立っていないと思うかもしれません。確かに数式としては成り立っていません。(「1=2」という等式は成り立たない)しかし、C 言語の「=」は数学のように等号を意味するのではなく、「右辺の値を左辺に代入する」という意味をあらわします。なので、このプログラムの最後では変数 a と変数 b の値はともに「2」となります。

では今学んだことを踏まえて以下のプログラム片を考えていきましょう。

```
int a = 5;

a = a + 1;
```

さて、実行後の a の値は何でしょうか？まず、1 行目では `int` 型の変数「 a 」を宣言すると同時に「 a 」に「10」を代入しています。2 行目は「 $a = a + 1$ 」なので「 a に $a + 1$ の値を代入する」という意味になります。つまり、「 a 」は $5 + 1$ で「6」になります。

なお、この例のようにある変数の値を四則演算だけして、その変数に代入し直す操作は、以下のとおり省略して書くことができます。(以下、 a と b は変数であるとする。 b については変数ではなく、数字でも可。)

省略前	省略後	意味
$a = a + b;$	$a += b;$	a の値を b 増やす
$a = a - b;$	$a -= b;$	a の値を b 減らす
$a = a * b;$	$a *= b;$	a の値を b 倍にする
$a = a / b;$	$a /= b;$	a の値を b で割る
$a = a \% b;$	$a \% = b;$	a の値を b で割った余りを a に代入する

例えば、さっきの「 $a = a + 1$ 」は「 $a += 1$ 」と省略できます。

さらに、「 $a += 1$ 」は「 $a++$ 」「 $++a$ 」と、「 $a -= 1$ 」は「 $a--$ 」「 $--a$ 」とさらに省略することができます。この「 $++$ 」を使って a の値を 1 増やすことを「インクリメント」、1 減らすことを「デクリメント」と言います。この表現は後で「for 文」というものを学ぶときに便利なので覚えておくといいでしょう。厳密には「 $a++$ 」と「 $++a$ 」は意味が微妙に違うのですが、ここではその説明は割愛させていただきます。

「 $==$ 」や「 $!=$ 」は次の第三回の「if 文」で使われるので、それぞれ数学の「 $=$ 」「 \neq 」の意味であることを、頭の片隅に入れておいてください。

■ scanf

次は scanf について説明します。scanf とはキーボードからの入力を書式に従って変数に読みこむ命令です。まず下のプログラムを見てください。

a02_2.c

```
#include <stdio.h>
int main( void ){
    /*変数 num を宣言*/
    int num;

    /*変数 num に 2 を代入*/
    num = 2;

    /*変数 num の値を出力*/
    printf("num の値は%d です。¥n", num);

    return 0;
}
```

このプログラムは今まで説明してきたように、変数 num を宣言し、変数 num に 2 を代入し、そして変数 num に代入した値を表示するプログラムです。ここで、5 行目の変数 num に違う値を代入しようと思った時、このプログラムをわざわざ違う値に書き換えなければならないので、その都度コンパイルしなければならないので、これでは不便です。

さて、ではこの不便さを解決するために上のプログラム scanf で書き換えてみましょう。

a02_3.c

```
#include <stdio.h>
int main( void ){
    /*変数 num を宣言*/
    int num;

    /*文章の表示*/
    printf("整数を入力してください・・・");
    (次ページへ続く)
```

```

/*キーボードで変数 num を入力*/
scanf("%d", &num);

/*変数 num の値を出力*/
printf("num の値は%d です。¥n", num);

return 0;
}

```

このプログラムを実行すると、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 の値は〇〇です。」と出力されるはずです。

scanf はキーボードで変数に値を入力する関数で以下のように使います。

```
scanf("書式指定文字", &変数名);
```

書式指定文字とは 2 ページ目～3 ページ目にあるように、int 型なら%d、double 型なら%lf といったものです。変数名はその値を入力したい変数名です。ここで scanf 関数内の変数名の横に「&」が付いていることに気付いたと思います。これは printf のときにはないので、始めのうちは「&」をつけ忘れがちです。注意しましょう。(後で出てくる配列・ポインタなどを扱う時に「&」をつけない場合もありますが、始めのうちは「scanf には「&」が必要」と覚えておけばいいです。)

■ 練習問題

1. キーボードから 2 つの数を入力すると、その 2 つの数の積を表示するようなプログラムになるように、以下の空所/* (1) */～/* (5) */を埋めよ。

```

#include <stdio.h>
int main( void ){
    int x, y;
    scanf( /* (1) */ , /* (2) */ );
    scanf( /* (1) */ , /* (3) */ );
    printf( /* (4) */ ¥n , /* (5) */ );
    return 0;
}

```

2. ある数字(正の実数)を入力すると、その長さを半径とする円の面積を表示するプログラムを書け。ただし、円周率は 3.1416 とする。
<<ヒント>> float 型もしくは double 型の変数を使うといいかもしれません。