[プログラミング入門Ｉ　2 回目　２００３・９・4]　入出力と分岐

前回は、変数を使って計算して、それを出力させるプログラムを作りました。今回は、数字を入力させてそれを計算して出力するプログラムを作ってみることにしましょう。入力できるようになるとかけるプログラムもずーっとおもしろいものになります。

入力の方法

値の入力は scanf 関数というものを使います。まずは、例を示します。

```
#include <stdio.h>
main()
{
    int x;
    printf(“please input ?“);
    scanf(“%d”,&x);
    printf(“input is %d¥n”,x);
    return 0;
 }
```

入力は scanf 関数というものを使います。まず、入力する値をいれておく変数 x を宣言しておきます。データ型は整数なので、int で宣言します。入力してくださいというメッセージ please input?を printf で表示します。ここで、scanf 関数が実行されるとプログラムは入力されるまで停止します。scanf 関数で、整数を入力する時には以下のような形で使います。

scanf(“%d”,&整数型の変数);

括弧のなかの”%d”は、整数型を入力するという意味です（%d の d は decimal の d)。次に、入力した値をいれる変数を書きますが、このまえに&を書くのを忘れないでください。（&の意味はポインターところで説明します）

あとはこの後、入力された変数を printf で出力します。printf の使い方は前回説明したとおりです。

if 文のつかい方：条件によって動きを変える

コンピュータの重要な機能は計算をするということですが、入力された値や計算した結果によって動きを変える（条件を判断する）という機能ももっとも基本的な機能です。これの組み合わせによって、コンピュータはいろいろな動きをすることができるわけです。これを行うのが、**if 文**です。

では、例として入力された数字が正の数なのか、負の数なのかを判断するプログラムを作りましょう。

```
#include <stdio.h>
main()
{
    int x;
    printf(“please input ?“);
    scanf(“%d”,&x);
    if(x > 0)
        printf(“input is positive¥n”);
   else
        printf(“input is negative or zero¥n”);
    return 0;
 }
```

最初に説明したとおり、プログラムは書かれた順番に実行されます。scanf までは、最初の例と同じです。if のところでは、変数の x の値が 0 よりも大きければ、printf(“input is positive¥n”);が実行され、そうでなければ、else の後の printf(“input is negative or zero¥n”);が実行されます。if 文は以下の形式で使います。

if(条件式)　条件が真の時に実行される文

または、

if(条件文)　条件が真の時に実行される文　else 条件が真でないときに実行される文

else がない場合には、条件が真にならないときには何も実行しないで、次にいきます。例で、if の後、else の後の文は１つの文ですから、; で終わっていることに注意してください。条件式のところには、条件を表す式を書きます。条件式には以下のものがあります。

式１ > 式２　　式１が式２よりも大きい場合に真
式２ < 式２　　式１が式２よりも小さい場合に真
式１ >= 式２　　式１が式２よりも大きい、または等しい場合に真
式２ <= 式２　　式１が式２よりも小さい、または等しい場合に真
式１ == 式２　　式１が式２と等しい場合に真
式２ != 式２　　式１が式２と等しくない場合に真
なお、式の部分には＋や－をつかった数式を書くことができます。

複文：複数の文を実行する

ある条件が成立した時に、実行したい文が１つの文であるとは限りません。複数の文を実行したい場合があります。その場合は、複数の文を{ …} で囲みます。これを複文（compound statement）といいます。以下のプログラムは、負の数だった場合にメッセージを出力して、正の数にして最後に絶対値をプリントアウトする例です。

```
#include <stdio.h>
main()
{
    int x;
    printf(“please input ?“);
    scanf(“%d”,&x);
    if(x < 0){
        printf(“input is negative¥n”);
        x = -x;
    }
    printf(“absolute number is %d¥n”,x);
    return 0;
 }
```

x = -x;は、変数 x の値を取り出し、それを－にして、また x に代入します。そのあとで、printf でメッセージを出力します。この２つの文は{ … } で囲まれていますから、条件が成立したときに順に実行されます。これがない場合には、x=…の文しか実行されません。

{} で囲まれた部分は、１つの文になりますので、文を書くところにはどこでも書くことができます。だだし、複文の場合は、}のあとに;をつけなくてもいいことに注意してください。

if 文では、if の実行文にまた if 文を使う（ネストする）ができます。たとえば、条件ごとにいろいろな処理をしたい場合には以下のようにします。

```
if(条件式１）　条件式１の時、実行する文
else if(条件２）　条件式２の時に実行する文
else それ以外に実行する文
```

論理演算子：複雑な条件式

条件が複数ある場合には、if 文を続けて書くことであらわすことができます。たとえば、x が１以上１０未満であることを判断したい場合には、以下のようにすればいいでしょう。

```
if(x > =1) if(x < 10) 条件が成立したときに実行する文
```

しかし、これではあまり複雑になってしまうので、二つ以上の条件がある場合には、以下のような条件式を使うことができます。

条件式１ && 条件式２　条件式１が真かつ条件式２が真、ならば真
条件式１ || 条件式２　条件式１が真または条件式２が真、ならば真
！条件式　　条件式が真ならば、偽

これらを**論理演算子**といいます。これをつかえば、

```
if(x > =1 && x < 10) 条件が成立したときに実行する文
```

と書くことができます。

いろいろな算術演算子と優先度

前回、式の説明をした時に演算子として＋、－、＊、／があると説明しました。数値を計算する演算子を算術演算子と呼びますが、ここでまとめておきましょう。（教科書８２ページ）

＋　加算　－　減算　＊　乗算　／　除算　％　剰余

このほかに、上で使った符号を帰る－の単項演算子があります。演算子には、計算の優先度があります。たとえば、

　１＋３＊２

は、３＊２のほうが最初に行われます。これを＊のほうが＋よりも優先度が高いといいます。これは、数学の記法と同じです。数学と同じように、優先度を変える場合には、かっこをつかいます。

　（１＋３）＊２

これでは、１＋３のほうが最初に計算されます。なお、論理演算子は算術演算子よりも優先度が低いので、算術演算子の方が最初に実行されます。たとえば、

　１＋３　＞　２

では、１＋３が最初に計算され、そのあとで、比較が行われます。

いろいろなデータ型：整数と実数（浮動小数点数）

これまで、例ではすべて int 型を使っていますが、コンピュータの中では整数と小数点を持つ実数とは別の仕組みで扱われますので、区別する必要があります。少数点を持つ実数はその仕組みから、浮動小数点数とも呼ばれます。整数型は int ですが、実数をあらわすデータ型としては表現できる範囲から、float と double があります（double のほうが大きい）。ここでは、実数の使い方について簡単に説明しておきます。

```
#include <stdio.h>
main()
{
    float x;
    printf(“please input ?“);
    scanf(“%f”,&x);
    printf(“input is %f¥n”,x);
    return 0;
 }
```

実数を扱う場合には、宣言に int の代わりに float を使います。あと違いは、scanf や printf に%d の代わりに%f を用いることです。ためしに、1.234 と入力してみてください。

算術演算子は、int と同じですが、%（剰余）はつかえません。Int 型の変数と float 型の変数を混ぜて使う場合はちょっとした注意が必要です。まぜて演算する場合には、整数は同じ値の実数に変換されて計算されます。また、int 型の変数に実数型を代入する場合には、少数点以下は切り捨てられて、整数にして代入されます。また、float 型の変数に整数を代入する場合には、整数の値と同じ実数に変換されて代入されます。たとえば、

```
main(){
    int n;
    float x;
    x = 1.234;
    n = x+1.2;
    printf(“n=%d¥n”,n);
}
```

では、n=2 と出力されます。