

ソフトゼミ A 第 3 回 if 文・switch 文

■ はじめに

前回までに習ったプログラムは、書かれた文をすべて実行するものでした。しかし、それでは同じ処理しか行うことができません。今回は、A という条件だったら X という処理を行い、B という条件だったら Y という処理を行い... というように処理を分岐し、様々な条件でプログラムの流れを変える方法を解説します。

■ if 文

プログラムを分岐させる最も基本的な制御文が if 文です。if は英語と同じで「もし～なら...する」という処理を行うことができます

a03_1.c

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

    printf( “整数を入力してください” );
    scanf( “%d”, &a);

    if( a > 10 ) {
        printf( “その整数は 10 より大きい数です。¥n” );
    }

    return 0;
}
```

太文字の部分が if 文です。この部分は、

```
if(条件式) {
    処理する内容;
}
```

という形になります。(「{ }」はつけなくていい場合があります。詳しくは最終ページで。)

最初のうちは、『もし「a > 10」なら処理内容を実行する』と覚えるのがいいと思います。細い話をする、次の通りです。

後述しますが、条件式は内容が条件に合えば 1 を、合わなければ 0 を返します。if 文は、(～) で囲われた中身の演算結果が 0 以外なら処理内容を実行し、0 の場合、処理内容は実行

されません。上の例では、`a` が 10 より大きければ、「`a > 10`」の演算結果は「1」となり、`printf` が実行されます。`a` が 10 以下であれば、「`a > 10`」の演算結果は「0」となり、`printf` 文は実行されません。`if` 文の(～)の演算結果が「0」であることを「偽(ぎ, false)」、「0」以外であることを「真(しん, true)」とすることがあります。

■ else 文

ところで上記のプログラムは 10 以下の数字を入力した場合何も表示されません。ではそれ以外の数字を入力しても文が表示されるようにしてみましょう。`else` は、『～でなければ』を意味します。

a03_2.c

```
#include <stdio.h>
int main( void ){
    int a;
    printf( "整数を入力してください" );
    scanf( "%d", &a);
    if(a > 10){
        printf("その整数は 10 より大きい数です。¥n");
    }else{
        printf("その整数は 10 以下です。¥n");
    }
    return 0;
}
```

これで「10 より大きい数」以外の値(すなわち、10 以下の数)を入力した場合もちゃんと文が表示されるようになりました。

■ 演算子

条件式に使われている『<』のような記号は、`a` と `b` の大小関係が成り立っていれば 1 を、成り立っていなければ 0 を返します。以下はその一覧です。

<code>a < b</code>	<code>a</code> は <code>b</code> より小さい
<code>a <= b</code>	<code>a</code> は <code>b</code> と同じか小さい
<code>a > b</code>	<code>a</code> は <code>b</code> より大きい
<code>a >= b</code>	<code>a</code> は <code>b</code> と同じか大きい
<code>a == b</code>	<code>a</code> は <code>b</code> と等しい
<code>a != b</code>	<code>a</code> は <code>b</code> と等しくない
<code>a && b</code>	<code>a</code> かつ <code>b</code>
<code>a b</code>	<code>a</code> または <code>b</code>

これらを用いることで様々な条件式を作ることができます。

■ else if 文

プログラムを様々な方向に分岐させるときは else if 文を使います。上記のプログラムを以下のように書き換えてみてください。(厳密には else if 文という文は存在せず、if 文と else 文の組み合わせで成り立っているものです →最終ページ参照)

a03_3.c

```
#include <stdio.h>
int main( void ){
    int a;
    printf( “整数を入力してください” );
    scanf( “%d”, &a);

    if(a > 10){
        printf( “その数は 10 より大きいです\n” );
    }else if(a == 10){
        printf( “その数は 10 です\n” );
    }else{
        printf( “その数は 10 より小さいです\n” );
    }

    return 0;
}
```

このようにすることによってプログラムを 3 通りに分岐させることができます。

■ switch 文

プログラムをいくつにも分岐する方法として if 文の他に、「switch 文」なるものがあります。まず下の if 文を見てください

a03_4.c

```
#include <stdio.h>

int main(void) {
    int a;

    printf( “整数を入力して下さい” );
    scanf( “%d”, &a );

    if(a==0){
        printf( “0 が入力されました\n” );
    }
    (次ページへ)
```

```

    }else if(a==1){
        printf("1 が入力されました\n");
    }else if(a==2){
        printf("2 が入力されました\n");
    }else if(a==3){
        printf("3 が入力されました\n");
    }else if(a==4){
        printf("4 が入力されました\n");
    }else{
        printf("0～4 以外が入力されました\n");
    }
    return 0;
}

```

条件式が似ていて打つのが面倒だと思いませんか？では、これとほぼ同じ意味のプログラムを switch 文で書いてみましょう。

a03_5.c

```

switch( a ) {
case 0:
    printf( "0 が入力されました\n" );
    break;
case 1:
    printf( "1 が入力されました\n" );
    break;
case 2:
    printf( "2 が入力されました\n" );
case 3:
    printf( "3 が入力されました\n" );
    break;
case 4:
    printf( "4 が入力されました\n" );
    break;
default:
    printf( "0～4 以外が入力されました\n" );
    break;
}

```

default はどの case にも当てはまらない場合に向かう目印です。

break には、switch 文を終了させる働きがあります。(次回、for 文・while 文という文のループから抜ける働きも学びます。) 実は、前ページのプログラムは意図しない動作をする場合があります。case 2 の場合 break がないので、2 を入力した場合 case 2 と case 3 が実行されてしまいます。

このようなことを防ぐために、break は必ずつけましょう。(わざと break を抜かして、複数の case を実行させるテクニックもあります。)

■ 練習問題

1. a03_2.c を改造して、入力された数が 3 の倍数であれば、「3 の倍数です」と、そうでなければ「3 の倍数ではありません」と出力するプログラムを作れ。
2. 身長が 120[cm]未満の人は乗れない、130[cm]未満の人は保護者の同伴が必要なジェットコースターがある。身長[cm]を入力して、その人がジェットコースターに乗るための条件を満たしているかどうかを判定し、結果を画面に表示するプログラムを書け。保護者の同伴が必要な場合は、その旨を示せ。
3. 整数を 3 つ入力させ、1 番大きい数を表示させるプログラムを作れ。
4. 西暦を入力し、その年の干支を教えてくれるプログラムを switch 文で作れ。

参考 1 :

2012 年はたつ年です。

参考 2 :

(十二支の順) ねずみ / うし / とら / うさぎ / たつ / へび / うま / ひつじ / さる / とり / いぬ / いのしし

■ コラム: { ~ } は必要か?

※この欄の事は、ちょっとマニアックな内容なので、覚える必要はありません。

if 文(if(~))の本当の意味は、「(~)内が真であれば、直後の 1 命令を実行する」です。また、波かっこ{ ~ }は、複数の命令を 1 命令にまとめる効果があります。ですから、if 文の処理内容が 1 文で済む場合は、波かっこ{ ~ }をつけずに書いても構いません。例えば、a03_1.c の場合は

```
if( a > 10 )
    printf( “その整数は 10 より大きい数です。 %n” );
```

という具合に書いても構いません。しかし、波かっこをつけずに書いた場合、突然 if 文の中の命令が 2 つに変更する時にミスを犯しやすくなります。ですので、ソフトゼミ A では、1 命令であっても波かっこをつけるように教えています。なお、1 命令の場合は、

```
if(a > 10){ printf(“その整数は 10 より大きい数です。¥n”); }
```

のように、波かっこをつけたまま 1 行で書くことがあります。(ゼミ B でもこの表記で書いてある箇所があります。)

なお、else if 文は、本当は else if というキーワードがあるのではなく、else ブロックに if 文が内包されているのです。a03_3.c の else if 文を略さず書くと、

```
if(a > 10){
    printf(“その数は 10 より大きいです¥n”);
}else{
    if(a == 10){
        printf(“その数は 10 です¥n”);
    }else{
        printf(“その数は 10 より小さいです¥n”);
    }
}
```

となります。こう書くくらいなら、else if～と書いた方が見やすいですね。意味は同じです。というわけで、実際には else if 文という文があるわけではなく、if 文と else 文の組み合わせで成り立っているのですが、わかりやすいので便宜上、else if 文という表現をさせていただきました。

蛇足ですが、「{ ～ }」は、わかりやすくするために「波かっこ」という呼び方をしていますが、正式には「curly brackets」と言うようです。