2012 年 5 月 15 日（火）

第 4 回 C 言語講習会 資料

■ switch 文

例えば、以下のような表があります。

月曜日の時間割		
1	電子回路基礎	独立館
2	応用数学	独立館
3	英語リーディング 2	第 4 校舎
4	コンピュータ実習	第 7 校舎
5	通信システム	独立館

時限を入力したら、科目名が表示されるプログラムを作ってみましょう。これは前回勉強した else-if 文で作れます。

```
#include <stdio.h>

int main()
{
        int period;
        printf("時限 ＝ ");
        scanf("%d", &period);
        if (period == 1)
                printf("電子回路基礎¥n");
        else if (period == 2)
                printf("応用数学¥n");
        else if (period == 3)
                printf("英語リーディング 2¥n");
        else if (period == 4)
                printf("コンピュータ実習¥n");
        else if (period == 5)
                printf("通信システム¥n");
        else
                printf("時間割外です¥n");
        return 0;
}
```

さて、これでプログラムは作られたのですが、else-if ばかりで見づらくなってしまいました。そこで **switch 文～case** を使ってみましょう！
switch 文～case の使い方は次の通りです。

```
--------------------------------
    switch (条件式) {
    case 数値:
            実行文;
             break;
    case 数値:
             実行文;
            break;
    }
--------------------------------
```

switch 文は、指定された条件式の値と同じ値の case へジャンプします。ジャンプ先では case 以下の文を実行し、break 文を見つけたら、switch 文の後で囲っている {} の中から抜け出します。
では、先ほどのプログラムを switch 文～case で書き直してみましょう！

```c
#include <stdio.h>

int main()
{
        int period;
        printf("時限 ＝ ");
        scanf("%d", &period);
        switch (period) {
        case 1:
                printf("電子回路基礎¥n");
                break;
        case 2:
                printf("応用数学¥n");
                break;
        case 3:
                printf("英語リーディング 2¥n");
                break;
        case 4:
                printf("コンピュータ実習¥n");
                break;
        case 5:
                printf("通信システム¥n");
                break;
       }
        return 0;
}
```

番号との対応が分かりやすくなりました。
ところで、時間割外の数字を入力した時はどうなるでしょうか。何も表示してくれませんね。
このように、他の case の値に当てはまらない場合に処理を実行させるには、**default** を使用することができます。default は、case 文の代わりとして使うことができます。default には、他の case に当てはまる数値が無かった場合にジャンプします。
上のプログラムに default を追加してみましょう！

```c
#include <stdio.h>

int main()
{
        int period;
        printf("時限 ＝ ");
        scanf("%d", &period);
        switch (period) {
        case 1:
                printf("電子回路基礎¥n");
                break;
        case 2:
                printf("応用数学¥n");
                break;
        case 3:
                printf("英語リーディング 2¥n");
                break;
        case 4:
                printf("コンピュータ実習¥n");
                break;
        case 5:
                printf("通信システム¥n");
                break;
        default:
                printf("時間割外です¥n");
                break;
       }
        return 0;
}
```

さて、最後に、時限を入力したら教室がどこの校舎にあるかを表示してくれるプログラムを作りたいと思います。実は、case は、複数を連続させて使うことが可能なのです。次のプログラムを見てください。

```c
#include <stdio.h>

int main()
{
        int period;
        printf("時限 ＝ ");
        scanf("%d", &period);
        switch (period) {
        case 1:
        case 2:
        case 5:
                printf("独立館¥n");
                break;
        case 3:
                printf("第４校舎¥n");
                break;
        case 4:
                printf("第７校舎¥n");
                break;
        default:
                printf("時間割外です¥n");
                break;
       }
        return 0;
}
```

このプログラムでは 1、2、5 が入力された場合には、いずれも「独立館」を表示します。

■ for 文

例えば、「A」という文字を10回表示させたい時、1つ1つprintfを使っていては面倒ですね。回数の決まっている繰り返しを処理したい時、**for文**を使います。
for文の使い方は次の通りです。

```
------------------------------------------
    int i;
    for (i = 1; i <= 繰り返し回数; i++) {
          繰り返す文 ;
    }
------------------------------------------
```

このiというのは、繰り返しの回数を数えるために使います。
Aを10回表示するプログラムは以下のようになります。

```c
#include <stdio.h>

int main()
{
        int i;
        for (i = 1; i <= 10; i++) {
                printf("A¥n");
        }
        return 0;
}
```

さて、for 文で繰り返し(ループ)が実現出来ることを説明しましたが、ここでは、その for 文の動作の仕組みを、詳しく説明します。より具体的な for 文の使い方は、次のようになります。

```
------------------------------------------
    for (初期化;条件式;更新) {
          繰り返す文 ;
    }
------------------------------------------
```

初期化とは、カウント変数の初期化を行うための文です。ここに書かれた式は、最初に 1 回だけ実行されます。
条件式とは、ループの終了条件を設定するための文です。ここに書かれた式の値が真の間は、繰り返す文を実行し続けます。
更新とは、カウント変数の更新を行うための文です。ここに書かれた式は、繰り返しを行う文を実行した後に実行されます。
これを元にして、上のプログラムの動作を調べてみます。

ここでは、初期化の式が i=1 となっています。この式は最初に 1 回だけ実行されるので、ループの開始時点で i は 1 になります。
次に、条件式の比較を行います。この段階では i の値は 1 のままなので i<=10 の結果は真となり、その結果、まだループは実行を続けることになります。
次に、繰り返す文を実行します。ここでは printf 文が実行されます。
次に、更新の式が実行されます。それまで i の値は 1 でしたが、この更新の式は i++となっているので i の値が 1 増加して 2 になります。
この、条件式→繰り返す文→更新、という実行を何度も繰り返して、i が 11 になった時 i<=10 の条件が偽となりループから抜け出すことになります。
この様に、カウント変数の値を変化させながら、条件が偽になるまで繰り返すことで、決まった回数のループ処理を実現させているのです。

■ 参考文献

山崎信行：『プログラミング言語 C -入門から中級へ-』, (コロナ社, 2007)
苦しんで覚える C 言語 http://9cguide.appspot.com/