第３回Ｃ言語講座

2012/6/6

メインテーマ：配列＆文字列とその扱い

前回演習の残り

①

　キーボードから入力された値について以下の処理を行うプログラムを作れ（入力される値は正の整数とする）

　　b) １～入力された値までの整数の合計を表示せよ

〇配列

　配列は，簡単に言うと**変数の集合体**です．配列単独では普通の変数とあまり変わりませんが，前回やった for 文のループ処理と組み合わせるととても有用なものになります．

まず，配列の宣言の仕方ですが，
『 型 配列名[要素数] 』という様に
記述します．
右の例だと，arr[0] ， arr[1] ， arr[2] という名前の変数が作成されていると考えればおｋです．
宣言した時に値を代入したい場合は右の例のように中かっこを使います．

このプログラムで注目して欲しいのは，**for ループと配列の組み合わせ**です．
配列を変数として使うときに arr[1]というように書きますが，［このかっこの中］に for 文の ｉ などを入れるとループ回数に応じた配列の要素が使えますね．

言葉での説明は煩雑だと思うので，右の例の for 文を ｉ の値に気をつけながらどんな処理をしているか想像してみてください．

```
#include <stdio.h>
int main(void)
{
        int arr[3] = { 6 , 10 , 3 }; //配列の宣言＆初期化
        int i ;
        //普通の変数のように扱えます．
        printf(" arr[0] = %d ¥n" , arr[0] );
        printf(" arr[1] = %d ¥n" , arr[1] );
        printf(" arr[2] = %d ¥n" , arr[2] );

        arr[0] = 12;     //普通の変数のように代入できます．
        arr[1] = 5 * 3;  //計算も可能です．
        arr[2] = arr[0] * arr[2];
        printf("値を変更しました¥n");
        //forループと配列の組み合わせ．ｉの値をイメージ！
        for( i = 0 ; i < 3 ; i++ )
        {
                printf( "arr[%d] = %d¥n" , i , arr[i] );
        }
        return 0;
}
```

arr[3]と宣言して作ったので，３つの要素 arr[0],arr[1],arr[2]が作られていますが，**arr[3]が作られていない**ことに注意しましょう．[0]から作るので必ず１小さい数までしか作られません．

○char 型

新しい型の説明をします．char と書き，キャラ型と読みます．チャー型じゃないです o<´･ω･`>o
int 型が整数，double 型が小数を入れる型でしたが，**char 型は文字を入れるための型**です．しかし，変数に文字がそのまま格納されるわけではないので注意が必要であり，逆に利用できたりもします．

char 型の基本．
char 型の変数は１文字だけ格納することができます．初期化の際は　「'」　シングルクォーテーションで囲います．
また，printf で表示させるときは「%c」を使います．

右の例で char 型の変数に数字を足していますが，エラーを吐かずに正常に動作していると思います．**実は char 型の変数には文字ではなく，その文字に対応する数字が格納されています．**（ASCII コード）

a の５つ先は f ですよね．いろいろ数値を変えて遊んでみましょう．

```c
#include <stdio.h>
int main(void)
{
        char mozi = 'a'; // 'シングルクォーテーション'

        printf( "mozi1 = %c ¥n" , mozi );

        mozi = mozi + 5; // mozi += 5 と書いても同じです

        printf( "mozi1 = %c ¥n" , mozi );

        return 0;
}
```

○文字列

Ｃ言語で言う文字列は char 型の配列を指す。つまり，文字が入った配列の略で文字列って感じ．
右の例を実行するとわかるが，配列の各 char 型変数に１文字ずつ保存されている。
char 型の配列の最後の要素には，勝手に‘¥0’という文字が格納されている．これは文字列の終わりを表す文字（終端文字）で文字列には必ずこれがなくてはいけない．
（文字列の終わりがどこかわからなくなるため）
基本的には勝手に付加されるので気にしなくても問題はない。

```c
#include <stdio.h>
#include <string.h> //文字列操作系の関数がつかえるようになる
int main(void)
{
        char string[100] = "Hello World!"; //文字列≒char型の配列
        int i;
        printf("%s¥n", string); //&string[0]とstringは同じ意味
        printf("string[4] = %c¥n", string[4] );
        printf("string[8] = %c¥n", string[8] );

        printf("文字列を入力してください¥n");
        scanf("%s", string); //scanfを使った文字列入力 %s!!
        printf("入力された文字列は %s です¥n", string); //%s!!
        printf("文字列の長さは %d です¥n", strlen(string));
        return 0;
}
```

つまり，**半角文字 60 個を格納した文字列を作る場合，要素数は 61 個必要になります．**
何が起こるかわからないので，要素は多めに用意しておくのがいいでしょう．

〇便利な文字列操作関数

戻り値，引数などの用語は次回にやりますが，とりあえずサンプルプログラムで許してください;;

・strcpy(文字列１ , 文字列２);
文字列１に,文字列２の中身をコピーします．戻り値はありません.

・strlen(文字列);
文字列の長さを戻り値として返します.

```
#include <stdio.h>
#include <string.h>
int main(void)
{
        char string1[100] = "Hello World!";
        char string2[50] = "はろーわーるど";
        int length1 = 0 , length2 = 0;

        printf( "string1 = %s ¥n" , string1 );
        printf( "string2 = %s ¥n" , string2 );

        length1 = strlen( string1 );
        length2 = strlen( string2 );

        printf( "string1 の長さは%d ,string2の長さは%d¥n¥n" ,
length1 , length2);

        strcpy( string1 , string2 ); //コピー

        printf( "コピーしました¥nstring1 = %s ¥n" , string1 );
        return 0;
}
```

演習問題　（基本的に問題に書かれた動作をすれば正解とします。）

①10 個の数の入力を受け取った後，入力された順番に画面に出力するプログラムを作れ.
　数は，などで区切ってわかりやすく表示させること.（配列と for 文を使って楽をしよう）

②10 個の数の入力を受けとった後，小さい順番に並び替えて画面に出力するプログラムを作れ.

③まず入力を受け付け文字列を１つ作成し，次に１文字入力させる．その後，最初の文字列から，入力した１文字を抜いた文字列を出力させるプログラムを作れ.
（入力 abcdcba →　入力 b → 出力 acdca）　のような感じで.