第４回 C 言語講座

2012/6/6

メインテーマ：関数

前回演習

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．

〇関数

　関数はそのまま，数学の関数と意味は同じです．数学の関数は例えば $y=5x+3$ といったものですが，この関数の場合 x の値が決まれば y の値も決まりますね．つまり**「何か値を受け取って，それを元に処理（計算）し，その結果を返している」**と言えます．

　ここから C 言語のお話になります．C 言語において関数は**「何か値を受け取って（受け取らない関数もある），処理し，その結果を返している（返さない関数もある）」**というものです．煩雑ですね．説明していきます．
実は今まで使ってきたprintf()もscanf()も関数です．printfは，printf("ここに表示された文字")を受け取り，画面に表示するという処理をしている関数です．scanfは，入力された値ををを受け取り，対応する変数に代入を行うという処理を行なっている関数です．ついでにいままで書いてきたint main(void)も関数ですが，ちょっと置いておきます．

〇関数の表記の仕方

　新しい用語の説明をします．先ほどの項で説明した，関数が受け取る値のことをを引数（ひきすう），処理の後の返す値のことを返り値または戻り値と呼びます．引数や返り値にも型の概念があります．(int, double など)

　まず最初に例として，円の半径の値を受け取り，円の面積を求め，その値を返す関数を自分で作って，更に使ってみましょう．

右のプログラムでは戻り値が double 型で関数名が circle、引数に input をとる関数を自作しています。
プログラムは基本的に上から実行されるため、main の後に関数を書く場合は右のようにプロトタイプ宣言を書いて、「こんな関数があるよ」と教えてあげる必要があります。
自作関数本体が main 関数より上にある場合はプロトタイプ宣言は必要ありません。
※main 関数自体はどこに書いてもそこから始まるため、場所は問いません。

```c
#include <stdio.h>
//円の面積を求める自作関数プロトタイプ宣言
double circle(int r);

int main(void)
{
	int input;
	double result = 0;
	printf("半径の値を入力してください¥n");
	scanf( "%d" , &input );
	result = circle( input );//自作関数の呼び出し＆戻り値の取得
	printf( "半径%dの円の面積は%lfです。¥n", input , result );
	return 0;
}

//自作関数の本体
double circle(int r)
{
	double menseki;
	menseki = r * r * 3.141592;
	return menseki;
}
```

関数を呼び出すときは，関数名(引数) と書き，戻り値がある関数の場合はその書いたものがそのまま戻り値となります．上の例で説明すると，`result = circle ( input );` この１行で circle 関数を呼び出しています．引数として変数 input を渡し，circle 関数で処理をした戻り値が result に代入されることになります．

circle 関数の本体の最初の一行目，`double circle(int r)` と書いてありますね．最初の double は circle 関数の戻り値の型を表しています．次にかっこの中味について，**circle 関数内で使う int 型の変数 r を作り，呼び出された時の引数 input を代入しています．**つまり，ここに書く変数は呼び出された時の引数と同じ型でないとバグの原因となりえるので注意しましょう．

circle 関数最後の行，`return menseki;` は，変数 menseki の値を戻り値として返すことを表しています．
circle 関数の戻り値の型を double としてあるので，変数 menseki も double 型で作っていますね．
戻り値が main 関数の変数 result に代入されて，プログラムは終了です．

このプログラムの処理を簡易的に記すと下のようになります．

main関数	circle関数（自作）

double circle(int r); //プロトタイプ宣言
int main(void)
{
　　int input;
　　double result = 0;
　　printf("半径の値を入力してください¥n");
　　scanf("%d" , &input);
 	
　　result = circle(input);
 	
　　printf("半径%dの円の面積は%lfです。¥n", input , result);
　　return 0;
}
1　2　4　5
	double circle(int r)
	{
	　　double menseki;
	　　menseki = r * r * 3.141592;
	　　return menseki;
	}
	3

1　まず main 関数の上に書いてあるプロトタイプ宣言がチェックされます．main 関数はプログラムのどこに書いてあっても最初に実行されます．

2　プロトタイプ宣言のあった circle 関数が main 関数に書いてあったので，circle 関数の処理を開始します．引数は input で，戻り値は result に代入されます．

3　**引数 input を，int 型の変数 r に代入**し，circle 関数内で計算に用います．circle 関数の戻り値は double 型なので，戻り値として利用するつもりの変数 menseki も予め double 型で作ります．

4　`return menseki;` で，変数 menseki の値が circle 関数の戻り値になりました．main 関数に処理が戻り，変数 result に menseki の値が代入されます．

5　main 関数の残りの処理を続行します．

○main 関数の戻り値

main 関数の引数と戻り値について説明します．

`int main(void)`

まず，引数の(void)は，引数が無いことを表しています．空欄でも警告メッセージがでるだけでコンパイルはで

きます．つぎに戻り値ですが，main の return から返される値はプログラム終了時に OS に渡されるエラーコードになり、０ならば正常終了，０以外ならば異常終了ということになっているようです．
（参考：main の戻り値の型は void でなく int ： http://www.6809.net/tenk/html/cgokai/int_main.htm ）

自作関数本体が上にある例。
この場合プロトタイプ宣言は必要ありません．
右の例のように引数は複数渡すこともできます．ただ，戻り値は１つしか返せません．

```
#include <stdio.h>
//n乗を返す
int ruizyou(int x , int y) //main関数のinputがxに，nがyに代入されます
{
        int kekka = x;
        for( int i = 2 ; i <= y ; i++ )
        {
                kekka *= x;
        }

        return kekka;
}
int main(void)
{
        int input , n ,result;
        printf("整数を入力してください¥n");
        scanf( "%d" , &input );
        printf("%dの累乗を計算します¥n何乗を計算しますか？¥n" , input);
        scanf( "%d" , &n );

        result = ruizyou( input , n );//自作関数の呼び出し&戻り値の取得
        printf( "%dの%d乗は%dです。¥n", input , n , result );
        return 0;
}
```

〇ヘッダのインクルード

自作関数の作り方，使い方はなんとなくわかってきたでしょうか．では，なぜ printf や scanf といった関数はなんにも書いていないのに使えるのでしょうか．

実は，書いていないのではなく書いているのです．どこに書いているのかというと `#include <stdio.h>` の部分です．printf や scanf の定義は stdio.h（拡張子 h はヘッダーファイルです）の中に書いてあります．#include<ヘッダー名>と書くとそのヘッダーの中身をまるまるそこに書いた扱いになるので，printf や scanf が使えたわけです．

〇変数の領域

今資料では，main 関数と自作関数では変数名を同じにしないようにしました．では，同じ変数名は使えないのかというとそんなことはなく，使えます．試しにサンプルプログラムを書き換えてみましょう．上手く書き換えれば必ず動きます．

では，違う関数内で同じ名前の変数名はどのような扱いになっているのでしょうか．答えは，**全く別の変数扱い**です．別の変数なので，自作関数内で `a = 5` のように代入したとしても，main 関数の変数 a の値が変更されることはないです．
（グローバル変数には当てはまりませんが，それは後ほど．）