# 第３回 C 言語講座

１．関数とは

数学において関数とは、f(x) = 「$x^2 + x + 3$」のようになっていて、例えば x に 2 を代入したら 9 になりますよね。プログラミングにおいても似たようなものです。ある値を与えてやると、なんらかの結果を返してくる。そんな命令群を C 言語では関数というわけです。

（何の値も与えず、かつ、答えが返ってこない関数もありますが）

因みに、printf()や scanf()、それだけではなく main()っていうのも関数なわけです。

printf()は、()内に文字列や変数などを入れ込んでこれを利用して表示する命令郡となっています。

２．関数の形式

使うときの話ではないので注意

関数には決まった書式があります。いや、ないと困るっしょ。

関数の書式は、

「戻り値の型 関数名 ( 引数 )」

のようになっています。

例えば、int main( void )は、戻り値の型が int 型、関数名 main、引数なし、というようになります。

……戻り値って何さ。

戻り値とは、関数の処理が終わった時に関数内の return によって返される値のことです。

関数名は、関数呼び出しの時に使う名前のことです。main とか、printf とか scanf とか。

引数とは、関数に渡す値（情報）です。例えば、「printf( “Hello World!” );」の「”Hello World!”」の部分のことです。

ここで引数について補足。引数に関しては、合ってもなくても良かったりするわけですね。

引数が必要ないときは、()内に void を書いてください。

main 関数は、引数とかなくても動けるわけです。だから void を書いてるってわけ。

ただ、実は引数を取るものがあったりします。

けど、ゲームを作る上ではほとんど使いません。ので、興味があったら自分で。

３．関数のインクルード

C 言語の規格によってあらかじめ用意されている関数を標準ライブラリ関数といいます。

これらの関数を使うには対応する#include から始める行を書く必要があります。

既に暗黙の了解という感じでつかってますけど、printf()関数や、scanf()関数を使うには、

#include<stdio.h>と書く必要があるわけです。

標準ライブラリ関数は色々あるので、気になったら自分で調べてみましょう。

一応、いくつかの例だけを。

・#include<stdio.h>

主にファイル関係を扱えます。 printf()や、scanf()などのコンソール画面への入出力や、

外部ファイルからの読み込み、書き込みなども出来ます。

ファイル関係に関しては、また後日。

・#include<math.h>
数学関係の関数群で、sin()、cos()、tan()などの三角関数や累乗の pow()、平方根を求める sqrt()、対数 log()などがあります。

・#include<string.h>
文字列を扱う関数群。コイツもまた後日に。

4．自作関数
実は、関数は自分で作れるわけです。
取り敢えずサンプル。

math.h 内の定義を利用する為に必要だそうです

プロトタイプ宣言

関数の戻り値が関数に置き換わると思うといい

関数本体

```c
#define _USE_MATH_DEFINES

#include<stdio.h>
#include<math.h>

/*--- プロトタイプ宣言 ---*/
double circle_area( double r );

/*--- メイン ---*/
int main( void )
{
    double radius;
    double area;

    printf( "半径の長さを入力してください¥n" );
    scanf( "%lf" , &radius );

    area = circle_area( radius );

    printf( "半径 %.2lf の円の面積は %.2lf です¥n" , radius , area );

    return 0;
}

/*--- 円の面積を求める関数 ---*/
double circle_area( double r )
{
    double ans;

    ans = pow( r , 2 ) * M_PI;

    return ans;
}
```

コイツは半径の値を引数に、面積を返す関数として作成しました。

ここで注意。
プログラムは上から解釈されるので、main()関数にいきなり自作関数を書いても、理解出来ないわけです。そんなの知らんって言う感じになります。なので、プロトタイプ宣言と呼ばれる方法を使います。まず、main 関数を呼び出す前に、こういう関数があるんだよ、っていうことを示すために使います。
コンパイラは、この宣言によって、main()関数内に出現した関数を main()より後のところから読む事が出来ます。

因みに、コンパイラは上から順に解釈するので、main()関数より上に関数本体を書けばプロトタイプ宣言はいりません。ただ、これから先にファイル分割などをしたり、ヘッダーファイルを作成したりするときには、このプロトタイプ宣言を使うので、知っておいてくださいね。

main()関数より上に記述した場合のサンプルは以下。

```c
#include<stdio.h>

/*--- 総和を求める関数 ---*/
int all_sum( int end )
{
    int sum = 0;

    for( int i = 1 ; i < end + 1 ; i++ )
    {
        sum += i;
    }

    return sum;
}

int main( void )
{
    int end_number;
    int ans;

    printf( "1から入力された値までの和を出力します¥n" );
    printf( "末項の値を入力してください¥n" );
    scanf( "%d" , &end_number );

    ans = all_sum( end_number );

    printf( "総和は %d です¥n" , ans );

    return 0;
}
```

5．ヘッダーファイル

ヘッダーファイルは「～.h」のファイルのこと。

ヘッダーファイルはソースコード形式になっていて、コンパイラが別のソースファイルの一部として自動的に展開し、使用するファイルです。つまり、#include<～.h>でインクルードすると、自動的にその部分にヘッダーファイルの内容を書き込んでくれるわけです。

なので、コイツの中にはプロトタイプ宣言やグローバル変数の宣言などを入れることが出来るわけです。

後述するのですが、構造体などもこの中に書き込むことが出来ます。

グローバル変数はあまりオススメしたくないのですが……。

＊ファイル分割について

取り敢えず補足的説明ということで。

実は、.cpp ファイルはいくつか別々に作って、あとで組み合わせることが出来るわけです。

そんなことする意味あるの？だって？

まぁ、今の所必要ないよね。だって main()関数の中にちょっとだけ書いてるだけだから。

でも、これから先ゲームを作る時色んな関数が必要になるわけ。

色々関数を作っていると、まぁうん万とか行数を書くわけですよ。

シューティングゲームなら、画像描画用に関数作って、当たり判定の関数作って、弾幕用の関数作って、って色々してると、行数とかヤバくなって、どこに何書いたかわかんねー！！！！ってなるわけ。

だから、意味のある単位にわけて、ファイルを作成するといいよ。

もちろん、全部１つのファイルに書くっていう猛者もいるから、一概には言えないけど。

でも、やっぱり、分割した方がどのファイルがどのものなのかわかりやすいから、分けられるようにしておくといいよ。

サンプルはこんな感じ。
ヘッダーファイル

original.h × quadrilateral.cpp main.cpp

(グローバル スコープ)

```
1  #ifndef OR_H
2  #define OR_H
3
4  #include<stdio.h>
5
6  /*--- プロトタイプ宣言 ---*/
7  double quadrilateral( double height , double width );
8
9  #endif
```

自作関数用の cpp ファイル

original.h quadrilateral.cpp × main.cpp

(グローバル スコープ) quadrilateral(double height, double wi

```
1  #include "original.h"
2
3  /*--- 四角形の面積を求める関数 ---*/
4  double quadrilateral( double height , double width )
5  {
6      double ans;
7
8      ans = height * width;
9
10     return ans;
11 }
```

メイン関数

original.h quadrilateral.cpp main.cpp ×

(グローバル スコープ) main()

```
1  #include "original.h"
2
3  int main()
4  {
5      double quad_area;
6
7      double tmp_height;
8      double tmp_width;
9
10     printf( "四角形の面積を計算します¥n" );
11     printf( "縦の長さを入力してください¥n" );
12     scanf( "%lf" , &tmp_height );
13     printf( "横の長さを入力してください¥n" );
14     scanf( "%lf" , &tmp_width );
15
16     quad_area = quadrilateral( tmp_height , tmp_width );
17
18     printf( "四角形の面積は %.4lf です" , quad_area );
19
20     return 0;
21 }
```

main()関数を含めた、main.cpp と、circle()関数を含めた circle.cpp、プロトタイプ宣言を含めたヘッダーファイルの３つを作って用意すればコンパイル出来ます。
ソースファイルのフォルダに cpp ファイルを、ヘッダーファイルのフォルダに.h ファイルを入れればいいよ。
ヘッダーファイルのフォルダに入ってる.h ファイルをインクルードする場合は、#include “〜〜”のように、””でくくってやればお k。

ファイル分割に関しては、慣れてからの方がよくわかるかもしれない。
どの単位ごとに分ければいいかっていうのも、各々違うから、いくつかゲームを作ってみながら練習してね。

＊オマケ
関数内の変数について。
自作関数内で定義した変数は、その関数内でしか読み書き出来ません。
つまり、その関数内でしか使えません。
さらに、自作関数内で定義した変数は、自作関数を使い終わったあとに消去されます。

……どういうこと？

実は、関数内のものは関数内でしか扱えないため、関数が終わった後、この関数内の変数は必要なくなります。残っていてもメモリにゴミが溜まるので消されてしまうわけです。
前使った時の値をそのまま使いたいとかは出来ないわけです。

さらに言うと、自作関数から main()関数や、他の自作関数の変数へ読み書きすることは出来ません。
（特殊な方法を使えば可能ですが、それは配列やポインタのときに）
なので、自作関数外の値を使いたければ引数を使って読み込まなければならないわけです。

＊オマケ２
戻り値は１個しか作れません。
多数の値を返したくても返せません、無理です。
配列やポインタ、構造体をやるまで諦めててください。

＊オマケ３
戻り値のいらない関数について。
printf()関数や scanf()関数って、「x = printf()」のようにしなくてもいいよね。
これらの関数は戻り値が無くてもいいよね。
（実際にはあるのですが）
こんなふうに、関数に戻り値が無くてもいいものがあります。
数回後に配列や、ポインタといったものを扱うのですが、このときはじつは戻り値がなくてもなんとかなったりします。
戻り値というのはデータのことであり、画面への表示ではないわけです。
ゲームを作る上で、ウィンドウ上に画像を表示することがありますが、この画像表示を関数にまとめた場合、戻り値のない関数で画像描画することが出来ます。

因みに、printf()や scanf()では、エラーだとか、表示した文字数やらを返す戻り値が存在します。
ぶっちゃけ使った憶えないけど。

＊オマケ４
関数は再帰的に使えます。
再帰的とは、関数の中で自分のことを呼ぶことが出来る、というもの。
ただ、自分のことと言いましたが、関数名が同一なだけで、別ものと判断されます。
変数などは各関数ごとに用意されます。
練習問題にあるので試してみてください。

＊練習問題

1．５つの値から最大値を求める関数を作りなさい。

2．落下させる高さを引数に、落下にかかる時間を戻り値とする自作関数を作りなさい。

3．組み合わせ（コンビネーション）の計算をする自作関数を作りなさい。
（再帰関数で実装できるよ）

１～３について、main()関数でprintf()を使って確認出来るといいでしょう。
scanf()で数値を入力出来るとさらにいいね。

＊ヒント

2．x = ½ gt2 でしたよね？ 憶えてます？？
g = 9.8 でいいんじゃないでしょうか？？

3．nCr = n! / ( n – r )! * r!　でしたよね。