ソフトゼミ A　第 6 回

# 関数

関数と一口に言っても何か分からないと思います。今までのゼミ A で使ってきた printf や scanf などが関数と呼ばれるものです。それらの関数はすでに用意されているもので便利な関数ですが、自作することで書きたいプログラムにあった関数となるのでソースをわかりやすくできます。今回は関数の作り方と使い方を学びましょう。

## ◆ 関数とは？

関数とは、他の関数（main 文など）で呼び出されたときに、定義された手順を行うものです。例として printf は『二重引用符の中の文字列を%d,%f などの書式指定に従って画面に出力する』と定義されています。

## ◆ 関数の定義

関数を使うためにはまず定義しなければいけません。定義の仕方は、

```
返り値の型　関数名　（引数の型　引数名 1　，引数の型　引数名 2　，・・・）{
        （関数の処理）
return　（戻り値）
}
```

とします。戻り値については後で説明します。

気がつく人もいるかもしれませんが、この形式は main 文と同じです。main 文もまた関数なのです。（main 文は最初に実行されることが決まってる関数です。）

## ◆ 関数の使用

関数を使用したいときには呼び出したいところで呼び出したい関数名と引数を書きます。ここで注意しなければならないのは、コンパイラーはソース文の上からコンパイルしていくので、定義していない関数があった場合 int 型と仮定してしまうためエラーが起きてしまう場合があります。それを防ぐために、呼び出す関数より上に呼び出される関数を定義するか、もしくは呼び出す関数より上にその関数の後でどんな関数が定義されているか宣言する必要があります。その宣言を**プロトタイプ宣言**といいます。宣言の仕方は、

```
関数の型　関数名　（引数の型　引数名１　，引数の型　引数名２　，・・・）；
```

となります。

以上のことを踏まえて「受け取った値の 3 倍を画面に出力する関数 func」を作り使ってみましょう。

a06_1.c

```
#include <stdio.h>
void func(int a);   //関数プロトタイプ宣言
int main(void){    //void は「無し」という意味　main 文には引数がない
    int a=0,b=3;
    func(b);          //b を func に渡す
    printf("main a=%d¥n",a); //main 関数の a と func 関数の a とは全くの別物
    return 0;
}
//関数の定義
void func(int a){   //void 型の関数は返り値を返さない→return 文いらない
    a=a*3;            //渡された b の値が a に代入されている
    printf("func a=%d¥n",a);
 }
```

となります。プログラムの実行結果から分かるように main 文の a と func 文の a は全く違う値が入っています。(main 文は 0,func 文は 9 になっているはずです。)このように同じ変数名でもどの関数にあるかで違う変数になります。

## ◆ 引数、戻り値

違う関数どうしで変数が共有されないことを説明しましたが、関数間で数値のやり取りができないと不便です。そこで引数、戻り値を使います。

- 引数
  呼び出す関数から呼び出される関数に数値を渡す仕組みです。複数使うことができます。

- 戻り値
  呼び出された関数から呼び出した関数に数値を渡す仕組みです。引数と違いひとつの数値しか渡すことができません。

先ほどの関数 func では 3 倍にした値しか画面に出力できませんでしたが、改造して「指定した数倍した値を返す関数」に
してみましょう。

a06_2.c

```
#include <stdio.h>
int func(int a,int n);   //関数プロトタイプ宣言
int main(void){    //void は「無し」という意味　main 文には引数がない
    int a=0,b,n
    printf("b=");
    scanf("%d",&b);
    printf("x=");
    scanf("%d",&n);
    printf("before a=%d",a);
    a=func(b,n);          //b,n を func に渡す,func 関数の戻り値が a に入る
    printf("after a=%d¥n",a);
    return 0;
}
```

```
//関数の定義
int func(int a,int N){   //int 型の関数→return 文いる
    a=a*N;            //渡された b の値が a,n の値が N に代入されている;
    return (a);       //a の値を返す（a が戻り値）
 }
```

## ◆ グローバル変数

関数内で宣言された変数を**ローカル変数**といいます。a06_2. c にあるように関数内で宣言されたローカル変数は関数間では共有されず、同じ名前の異なる関数にあるローカル変数は異なるデータをもっています。かといって戻り値は一つしか使えない。（ポインタをうまく使えば解消できますが。）そこで使うのが関数外で宣言される**グローバル変数**です。グローバル変数は複数の（というよりソース全体の）関数で共有される変数です。グローバル変数を使ってプログラムを作ってみましょう。

a06_3.c

```
#include <stdio.h>
void f(void);
int a;        //グローバル変数の宣言はここ
int main(void){
    a=3;        //グローバル変数 a に 3 を入れる
    printf("before a=%d",a);
    f();          //関数 f が呼び出される
    printf("after a=%d",a);
    return (0);
}
void   f(void){
    a=10;    //グローバル変数 a に 10 を入れる
}
```

グローバル変数は初期値に 0 が入ります。グローバル変数と同じ名前の変数をローカル変数で宣言した場合はローカル変数が優先されます。

注意としてグローバル変数はどこからでも数値をいじることができてしまうので、どこで数値が変わったのかわかりづらくなります。これはバグが起きたときどの場所で不具合が起こっているのかわからないため、デバッグの際に非常に不便です。なのでどうしても必要なとき以外はグローバル変数を使わないようにしましょう。

## ◆ 練習問題

次の関数を作成してください。main 文も用意して動作確認をしてください。

1. 長方形の幅と高さを入力して面積を戻り値として返す関数
   double rectangle(double w,double h)

2. 値 a,b を入力してその最小公倍数を戻り値として返す関数
   intgetLCM(inta,int b)

3. 値 a,b,c を入力してそれらを大きい順に出力する関数
   void print_ascending_order(inta,intb,int c)