ソフトゼミ A 2015 第 6 回

関数

関数と一口に言っても何かわからないと思いますが、今まで出てきた 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なので便利ですが、自作することで書きたいプログラムに合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)
}
```

ここで思い出してほしいのが main 文です。

```
int main (void){
        (プログラムの中身)
return (0)
}
```

形式が同じですね。つまり、main 文も関数の一種です。(main 文は最初に実行されるのが決まっている関数です。)

➢ 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は 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 倍の値を画面に出力する関数 func」を作ってみましょう。

```
#include<stdio.h>
void func(int a); //関数プロトタイプ宣言
int main(void){   //void は引数がないことを示す
        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 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;
        printf("func a=%d¥n",a);
}
```

プログラムの実行結果からわかるように、main 文の a と func 文の a に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。

➢ 引数と戻り値

違う関数同士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、一つしか数値を渡すことができません。

…言葉で説明しても分かりにくいと思うので、実際のプログラム例を見てみましょう。

```
#include<stdio.h>
int func(int a,int n);   //関数プロトタイプ宣言
int main(void){
        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 型の関数なので return 文がいる
        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)
}
```

先ほどの関数 func を改造して、「指定した数倍した値を返す関数」にしました。main 文内の変数 b,n の値を func 文でも使えるようにしています。

➢ 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。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持っています。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。

```
#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;
        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;
}
```

グローバル変数は初期値に 0 が入ります。グローバル変数と同じ名前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ため、どこで数値が変わったのかわかりづらくなります。これはバグが起こった時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。どうしても必要な時以外はグローバル変数は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変数
 ingetLCM(int a,int b)
3.値 a,b,c を入力してをそれらを大きい順に出力する関数
 void print_ascending_order(int a,int b ,int c)