［13］　関数（1）～値による呼び出し～

1．ユーザー関数の定義と引数，戻り値の使い方

・頻繁に使う計算等を１つの独立したプログラムにする．これを「関数」と呼ぶ．

☆各関数のプログラムの間では変数は独立している．（同じ変数名でも内容は異なる）

→これを局所的変数（ローカル変数）という

・関数の間での値の引き渡しは，引数（実引数，仮引数）と戻り値で行う．

○関数の定義

```
型　関数名（型　仮引数，型　仮引数,...）　　（ ）に値の受け渡しをするための
{                                          変数（仮引数）をかく．
    文
  return　式；                             戻り値（計算結果など）を式に書く．
}
```

・関数の定義は，main関数の後でも前でもよい．

○関数プロトタイプ宣言

関数を定義したら，プログラムの最初（#includeの後）に宣言文を書く．

```
型　関数名（型　仮引数，型　仮引数,...）;
```

○関数を呼び出す（関数を使う）

```
関数名（実引数，実引数,...）;
```

●関数の定義と使用例

1．引数と戻り値がある場合

例）底辺と高さを与えると，三角形の面積を求める関数triangleを定義する

```
#include<stdio.h>
double traiangle(double a, double b);    関数プロトタイプ宣言
int main(void){                          main関数
  double teihen, takasa, area;
  scanf("%lf %lf", &teihen, &takasa);
  area=triangle(teihen,takasa);          関数を呼びだす．teihenとtakasaが実引数
  printf("Menseki %lf\n",area);
  return 0;
}
double traiangle(double a, double b){    関数の定義．aとbが仮引数
  double s;                              関数内で使用する変数は定義しておく
  s=(a*b)/2.0;
  return s;                              戻り値（計算結果）
}
```

☆実引数と仮引数の型は一致しなければならない．ただし変数名を一致させる必要はない．

・引数や戻り値が無い場合は，void型で宣言する．

２．引数も戻り値もない場合

例）Hello World!と表示する関数を定義する．

```
#include <stdio.h>
void SayHello( void );                  関数プロトタイプ宣言．
int main(void){
    SayHello( );                        SayHello関数を呼び出す．引数なし．
    return 0;
}
void SayHello(void){                    void型（値を返さない），引数なし．
     printf("Hello World!\n");
}                                       戻り値がないので，return文はいらない．
```

３．戻り値がない場合

例）引数の値を表示する関数を定義する．

```
#include <stdio.h>
void func(int b);                       関数プロトタイプ宣言．
int main(void){
    int a=10;
    func(a);                            戻り値が必要でない場合は関数名だけでよい．
    return 0; }
void func(int b){
    printf("Value = %d\n",b);           Value = 10 と表示される．
}                                       戻り値がないので，return文はいらない．
```

○以上のような値の引き渡し方を，値による呼び出し(call by value)という．

・実引数の値を，仮引数に代入している．

・実引数と仮引数の間では，値は互いに影響しない．