

知能情報工学演習I 第12回(後半第6回)

岩村雅一

masa@cs.osakafu-u.ac.jp



前回の課題

■ 課題1

- キーボードから自然数を入力し、その合計と平均値を逐次表示するプログラムを作成しなさい。ただし、キーボードから-999が入力された時に終了するものとする。



前回の課題：課題1

```
#include<stdio.h>
```

```
int main(void){
```

```
    int a,i=0,sum=0;
```

```
    float average;
```

```
    while (1) {
```

```
        printf("自然数を入力してください。¥n");
```

```
        scanf("%d",&a);
```

```
        if (a== -999) {
```

```
            break;
```

```
        }
```

```
        i++;
```

```
        sum=sum+a;
```

```
        printf("合計: %d¥n",sum);
```

```
        average=(float)sum/i;
```

```
        printf("平均: %f¥n",average);
```

```
    }
```

```
    return(0);
```

```
}
```



前回の課題

■ 課題2

- キーボードから5つの整数を入力し、小さい順に並べ変えなさい。変数aとbの値を交換するときには以下の方法がよく使われる。

```
int a, b, tmp;
```

```
tmp=a;
```

```
a=b;
```

```
b=tmp;
```

- ヒント: for文を二重に使うとよい。

前回の課題：課題2

```
#include <stdio.h>
```

```
int main(void){  
    int a[5],tmp,i,j;
```

```
    for(i=0;i<5;i++){  
        printf("%d番目の数字: ",i+1);  
        scanf("%d",&a[i]);  
    }
```

```
    for(i=0;i<5;i++){  
        for(j=i+1;j<5;j++){  
            if(a[i]>a[j]){  
                tmp=a[i];  
                a[i]=a[j];  
                a[j]=tmp;  
            }  
        }  
    }
```

```
        printf("小さい順に並べた結果: ");  
        for(i=0;i<4;i++){  
            printf("%d,",a[i]);  
        }  
        printf("%d¥n",a[4]);  
        return(0);  
    }
```

前回の課題

■ 課題3

- キーボードから3桁の自然数を入力したとき、1の位、10の位、100の位に同じ数字がちょうど2つあるかどうかを調べよ。

```
#include<stdio.h>

int main(void){
    int a;

    printf("数字を入力してください。¥n");
    scanf("%d",&a);

    if(a%2==1){
        printf("数字は奇数です。¥n");
    }else{
        printf("数字は偶数です。¥n");
    }
    return 0;
}
```

前回の課題：課題3

```
#include<stdio.h>
```

```
int main(void){
```

```
    int a,A,B,C;
```

```
    printf("3けたの自然数を入力して下さい  
    ¥n1の位、10の位、100の位に同じ数字  
    がちょうど2つあれば真,そうでなければ  
    偽¥n");
```

```
    scanf("%d",&a);
```

```
    A=a/100;
```

```
    B=(a%100)/10;
```

```
    C=(a%10);
```

```
    if (A==B&&B==C){
```

```
        printf("偽¥n");
```

```
    } else if (A==B||B==C||C==A){
```

```
        printf("真¥n");
```

```
    } else{
```

```
        printf("偽¥n");
```

```
    }
```

```
    return(0);
```

```
}
```

前回の課題

■ 課題4

- キーボードから自然数を入力してもらい、その数を素因数分解しなさい。

```
#include<stdio.h>
int main(void){

    int m,n;
    int c = ' ';
    printf("整数を入力してください。");
    scanf("%d",&n);
    printf("%d=",n);

    if (n==1) {
        printf("1≠n");
        return(0);
    }

    for (m=2; n!=1 ;m++){
        while(n%m == 0) {
```

前回の課題：課題4

```
#include<stdio.h>
int main(void){

    int m,n;
    int c = ' ';
    printf("整数を入力してください。");
    scanf("%d",&n);
    printf("%d=",n);

    if (n==1) {
        printf("1≠n");
        return(0);
    }
```

```
for (m=2; n!=1 ;m++){
    while(n%m == 0) {
        n = n/m;
        printf("%c%d",c,m);
        c = '*';
    }
}
printf("≠n");
return(0);
}
```



後半の予定

7. 6月2日 プログラミング環境(テキスト1,2章)
8. 6月9日 変数とデータ型(3章)、演算子(4章)
9. 6月16日 コンソール入出力(6章)、配列(3章)
10. 6月23日 制御文1(テキスト5章)
11. 6月30日 制御文2(テキスト5章)
12. 7月14日 関数1(テキスト7章)
13. 7月21日 関数2(テキスト7章)



本日のメニュー

■ 関数

- 関数とは何か
- 引数や戻り値が無い関数
- return文
- 関数の再帰的呼び出し
- ローカル変数とグローバル変数
- プロトタイプ宣言

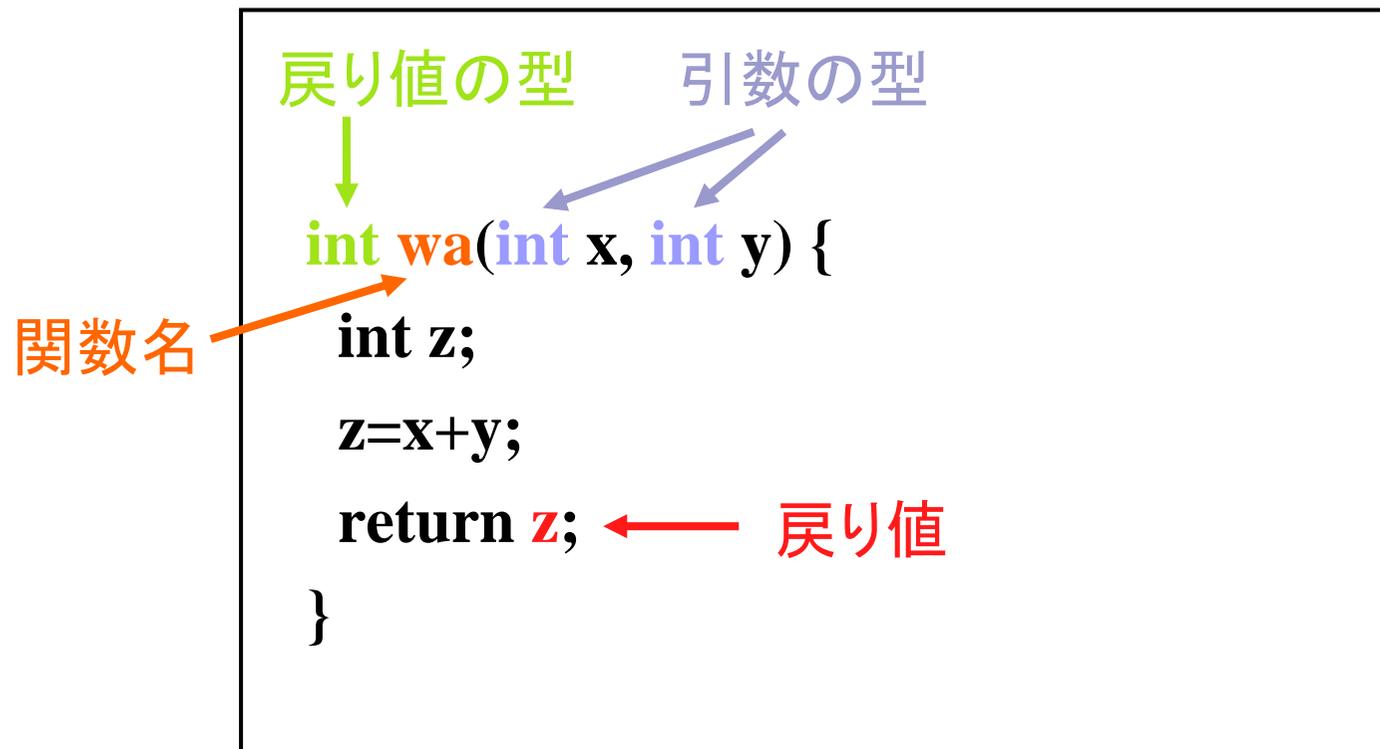


関数とは

- 引数を取り、処理の後、戻り値を返すもの
 - 数学の関数("y=f(x)")のようなもの
 - printf()やscanf()は関数
 - main()も関数

関数の例(定義)

- 関数waは、2つの引数xとyを元にzの値を計算し、戻り値として返す



サンプルプログラム

```
#include <stdio.h>

int wa(int x, int y) {
    int z;
    z=x+y;
    return z;
}

int main(void) {
    int a, b, sum;
    printf("a: "); scanf("%d", &a);
    printf("b: "); scanf("%d", &b);
    sum=wa(a,b);
    printf("a + b = %d\n", sum);

    return 0;
}
```

サンプルプログラム

```
#include <stdio.h>
```

```
int wa(int x, int y) {  
    int z;  
    z=x+y;  
    return z;  
}
```

```
int x=a;  
int y=b;
```

関数waの呼び出し

```
int main(void) {  
    int a, b, sum;  
    printf("a: "); scanf("%d", &a);  
    printf("b: "); scanf("%d", &b);  
    sum=wa(a,b);  
    printf("a + b = %d\n", sum);  
  
    return 0;  
}
```



引数や戻り値が無い関数

■ 引数が無い関数

```
int func(void) {  
    ...  
}
```

■ 戻り値が無い関数

```
void func(int a) {  
    ...  
}
```

return文の役割

- return文があると、関数の処理を打ち切って呼び出し元に戻る

```
int waru(int x, int y) {
```

```
    int z;
```

```
    if (y==0) {
```

```
        return -999;
```

```
    }
```

```
    z=x/y;
```

```
    return z;
```

```
}
```

yが0ならば-999を返して
関数の計算は終了

階乗を計算する関数

■ 階乗を計算する関数を考えてみよう

□ $3! = 1 * 2 * 3 = 6$

□ $0! = 1$

```
#include <stdio.h>
```

```
int fact(int x) {
```

```
    int fact = 1;
```

```
    ...
```

```
    return(fact);
```

```
}
```

```
int main (void){
```

```
    int n;
```

```
    printf("n: "); scanf("%d", &n);
```

```
    printf("fact(%d) = %d\n", n,  
          fact(n));
```

```
    return(0);
```

```
}
```

階乗を計算する関数

```
#include <stdio.h>

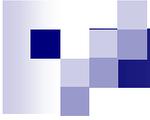
/* 階乗を計算する関数 */
int fact(int x) {
    int i, fact = 1;
    for(i = 2; i <= x; i++) {
        fact *= i;
    } fact = 1 * 2 * 3 * 4 * ...
    return(fact);
} 0!や1!もok
```

```
int main (void){
    int n;

    printf("n: "); scanf("%d", &n);

    printf("fact(%d) = %d¥n", n,
        fact(n));

    return(0);
}
```



関数の再帰的呼び出し

```
#include <stdio.h>
```

```
/* 階乗を計算する関数 */
```

```
int fact(int x) {  
    if (x==1 || x==0) {  
        return(1);  
    } else {  
        return(x*fact(x-1));  
    }  
}
```

```
int main (void){  
    int n;  
  
    printf("n: "); scanf("%d", &n);  
  
    printf("fact(%d) = %d¥n", n,  
        fact(n));  
  
    return(0);  
}
```

関数の再帰的呼び出し

```
#include <stdio.h>
```

```
/* 階乗を計算する関数 */
```

```
int fact(int x) {  
    if (x==1 || x==0) {  
        return(1);  
    } else {  
        return(x*fact(x-1));  
    }  
}
```

||
fact(1)
||
1

```
/* 階乗を計算する関数 */
```

```
int fact(int x) {  
    if (x==1 || x==0) {  
        return(1);  
    } else {  
        return(x*fact(x-1));  
    }  
}
```

関数の再帰的呼び出し

```
#include <stdio.h>
```

```
/* 階乗を計算する関数 */
```

```
int fact(int x) {  
    if (x==1 || x==0) {  
        return(1);  
    } else {  
        return(x*fact(x-1));  
    }  
}
```

fact(4)
= 4 * fact(3)
= 4 * 3 * fact(2)
= 4 * 3 * 2 * fact(1)
= 4 * 3 * 2 * 1

ローカル変数とグローバル変数

```
#include <stdio.h>
```

```
int global;
```

グローバル変数

```
int func1(int x, int y) {
```

```
    int local;
```

```
    ...
```

```
}
```

ローカル変数

```
int func2(void) {
```

```
    int local, global;
```

```
    ...
```

```
}
```

ローカル変数とグローバル変数

```
#include <stdio.h>
```

```
int global;
```

```
int func1(int x, int y) {  
    int local;  
    ...  
}
```

```
int func2(void) {  
    int local, global;  
    ...  
}
```

変数が有効な範囲

global

local

local

global

優先

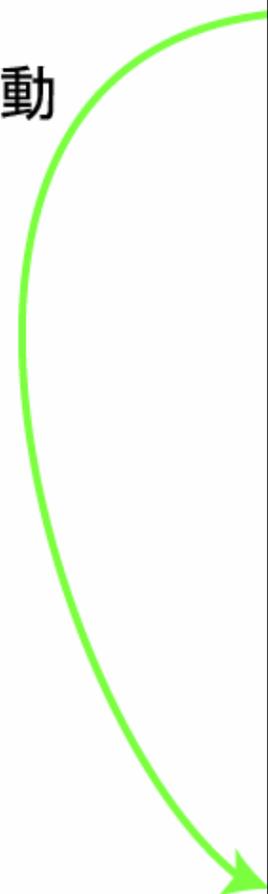


プロトタイプ宣言

- 関数の「定義」を先にできない場合は「宣言」だけを先に書く

プロトタイプ宣言

移動



```
#include <stdio.h>
```

```
int main(void) {  
    int a, b, sum;  
    printf("a: "); scanf("%d", &a);  
    printf("b: "); scanf("%d", &b);  
    sum=wa(a,b); ???  
    printf("a + b = %d\n", sum);  
  
    return 0;  
}
```

```
int wa(int x, int y) {  
    int z;  
    z=x+y;  
    return z;  
}
```

プロトタイプ宣言

```
#include <stdio.h>
```

```
int wa(int x, int y);
```

```
int main(void) {
```

```
    int a, b, sum;
```

```
    printf("a: "); scanf("%d", &a);
```

```
    printf("b: "); scanf("%d", &b);
```

```
    sum=wa(a,b);
```

```
    printf("a + b = %d\n", sum);
```

```
    return 0;
```

```
}
```

```
int wa(int x, int y) {
```

```
    int z;
```

```
    z=x+y;
```

```
    return z;
```

```
}
```