

# 2021 計算機プログラミング演習

## 第4回

scanf() と数学関数

# 第4回の目標

- プログラムの書き方
  - 変数名の付け方
  - 定義部と実行部
- scanf() の使い方（標準入力）
  - Visual Studio での特殊事情
- 数学関数の使い方
  - #include <math.h> の付加
  - 関数の仕様と意味、使い方

# プログラムの書き方

## 定義部と実行部

- コンパイラ言語では一般にプログラムに使われる「変数」は使用する前に**あらかじめ定義されていないといけない**。
  - プログラム作成中に新しい変数を使わなければならない場合には変数型 (char, int, double...) を含めて定義をしなければならない。
  - 定義されていない変数を使った場合はビルド時にエラーが表示される。
- C 言語の場合、変数などの定義部と実行部を分けて書く必要がある。

# プログラムの書き方

## 定義部と実行部

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

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

```
    int    ivalue1;  
    int    ivalue2;  
    int    ivalue3;
```

定義部

```
    ivalue1 = 100;  
    ivalue2 = 200;  
    ivalue3 = 300;  
    :  
    :
```

実行部

ここから下が実際に動作する部分

# 変数の代入と演算について

## 基本的な書式

左辺の変数に 右辺の数(演算結果)を代入する

```
ivalue1 = 100;  
ivalue2 = ivalue1 * 2 ;
```

現在の変数の値に対して演算し、結果を変数に上書きする

```
ivalue1 = ivalue1 + 1 ;  
ivalue2 = ivalue2 * 2 ;
```

次の書き方は上の2式と同じ意味である

```
ivalue1 ++ ;  
ivalue2 *= 2 ;
```

# 定義部の書き方

```
/*  
    tax.c 消費税換算プログラム  
*/  
  
#define    _CRT_SECURE_NO_WARNINGS 1  
#include    <stdio.h>  
  
int    main(void)  
{  
    int    iZeinuki;  
    int    iZeikomi;  
    int    iZei;
```

次ページに続く

# プログラムの書き方

- 続けて実行部を記入

```
printf("価格を入力してください ");  
scanf("%d", &iZeinuki);  
  
printf("    税抜価格  %d¥n", iZeinuki);  
  
iZeikomi = iZeinuki * 1.10 ;  
iZei = iZeikomi - iZeinuki ;  
  
printf("    税込価格  %d¥n", iZeikomi);  
printf("    消費税    %d¥n", iZei);  
  
return 0 ;  
  
}
```

# 警告(warning) について

出力

出力元(S): ビルド

```
ビルドを開始しました...
1>----- ビルド開始: プロジェクト: tax, 構成: Debug Win32 -----
1>tax.cpp
1>C:\Users\Nakano\source\tax\tax\tax.cpp(20,25): warning C4244: '=': 'double' から 'int' への変換です。データが失われる可能性があります。
1>tax.vcxproj -> C:\Users\Nakano\source\tax\Debug\tax.exe
1>プロジェクト "tax.vcxproj" のビルドが終了しました。
===== ビルド: 1 正常終了、0 失敗、0 更新不要、0 スキップ =====
```

(20行目)      iZeikomi = iZeinuki \* 1.10;

警告は出るが、正常終了している。

警告はプログラムの実行に支障がない場合もあるが、常に無視してよいというわけではない。

より正しいコーディング (キャストを使う)

(20行目)      iZeikomi = (int) (iZeinuki \* 1.10);



# とりあえず実行してみる

```
C:\Users\Nakano\source\tax1\Debug>tax
価格を入力してください 100
    税抜価格    100
    税込価格    110
    消費税      10

C:\Users\Nakano\source\tax1\Debug>
```

- 正しい答えが出ているのを確認

# キーボードからの入力 scanf()

## 【重要な事項】

- Visual Studio ではキーボード入力 scanf() を使わず、セキュリティを強化した scanf\_s() を使うことを推奨している。
- そのため、Visual Studio で scanf() を使ったプログラムをビルドしようとするとき warning (警告) を発し、ビルドに失敗する。
- プログラム文頭に以下の記述を入れて警告を抑制する。

– #include <stdio.h> よりも前！

```
#define _CRT_SECURE_NO_WARNINGS 1
```

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

# scanf() の使い方(1)

ソリューション名: scanf\_test

プロジェクト名: scanf\_test1

```
/*      scanf_test1.c      */

#define _CRT_SECURE_NO_WARNINGS 1
#include <stdio.h>

int main( void )
{
    int ivalue1, ivalue2, ireresult;

    scanf("%d", &ivalue1 ) ;
    scanf("%d", &ivalue2 ) ;

    ireresult = ivalue1 / ivalue2;
    printf("result = %d¥n", ireresult);

    return 0;

}
```

この一行を忘れないように

<stdio.h> より前に

入力する変数の前に  
必ず & を付ける

# scanf() の使い方(1) 実行結果

```
/*      scanf_test1.c      */

#define _CRT_SECURE_NO_WARNINGS 1
#include <stdio.h>

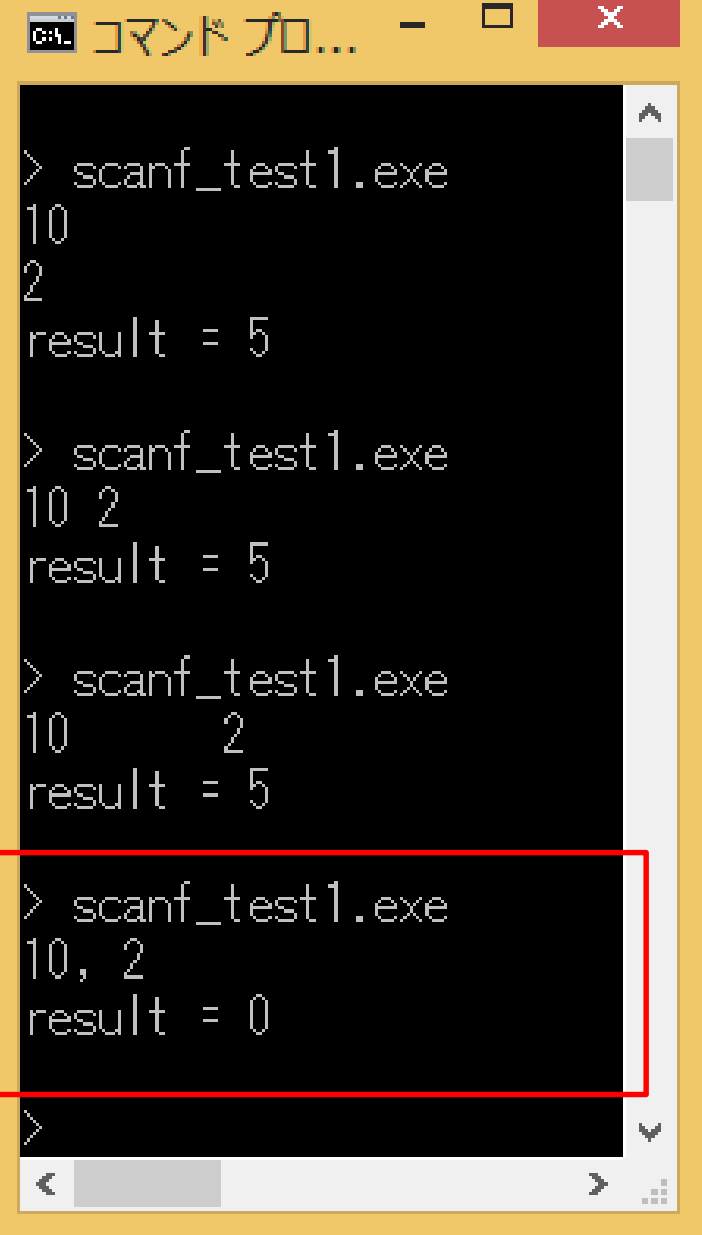
int main( void )
{
    int ivalue1, ivalue2, ireresult;

    scanf("%d", &ivalue1 ) ;
    scanf("%d", &ivalue2 ) ;

    ireresult = ivalue1 / ivalue2;
    printf("result = %d¥n", ireresult);

    return 0;
}
```

2つの数を , で区切った場合のみ、  
正しくない値が出る



```
> scanf_test1.exe
10
2
result = 5

> scanf_test1.exe
10 2
result = 5

> scanf_test1.exe
10      2
result = 5

> scanf_test1.exe
10, 2
result = 0

>
```

# scanf() の使い方(2)

```
/*      scanf_test1.c      */

#define _CRT_SECURE_NO_WARNINGS 1
#include <stdio.h>

int main( void )
{
    int ivaluel, ivalue2, ireresult;

    scanf("%d%d", &ivaluel, &ivalue2 ) ;

    ireresult = ivaluel / ivalue2;
    printf("result = %d¥n", ireresult);

    return 0;
}
```

1行で2つの数を入力する

# scanf() の使い方(2) 実行結果

```
/*      scanf_test1.c      */

#define _CRT_SECURE_NO_WARNINGS 1
#include <stdio.h>

int main( void )
{
    int ivalue1, ivalue2, ireresult;

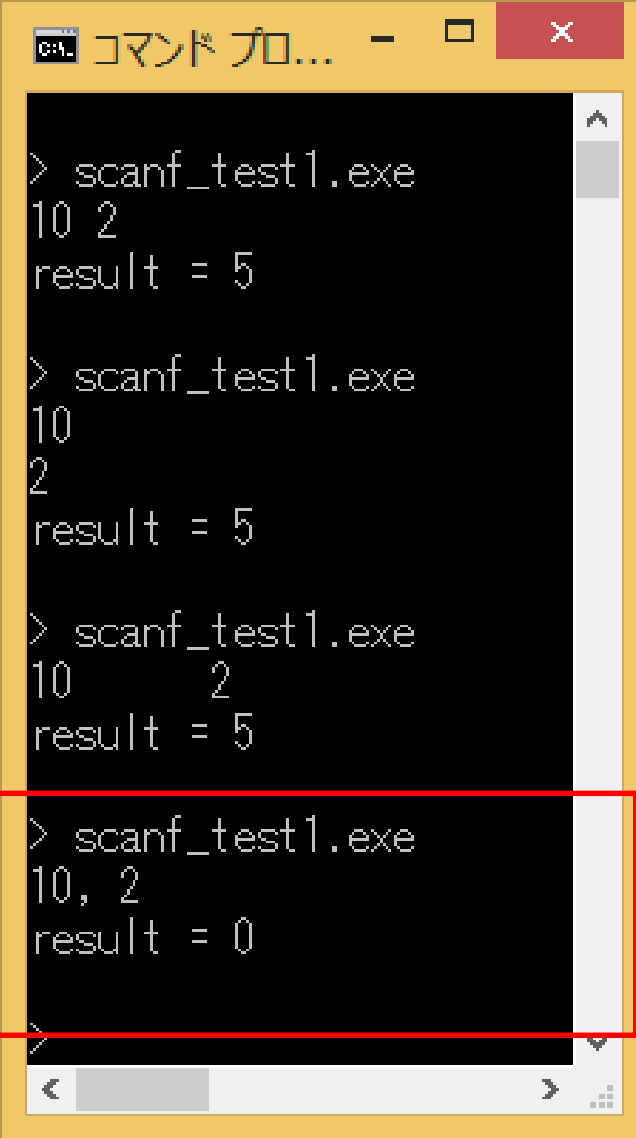
    scanf("%d%d", &ivalue1, &ivalue2 ) ;

    ireresult = ivalue1 / ivalue2;
    printf("result = %d¥n", ireresult);

    return 0;

}
```

この場合も同様。  
2つの数を , で区切った場合のみ、  
正しくない値。



```
cmd コマンドプロンプト

> scanf_test1.exe
10 2
result = 5

> scanf_test1.exe
10
2
result = 5

> scanf_test1.exe
10, 2
result = 5

> scanf_test1.exe
10, 2
result = 0
```

# scanf() の使い方(3)

```
/*      scanf_test1.c      */

#define _CRT_SECURE_NO_WARNINGS 1
#include <stdio.h>

int main( void )
{
    int ivalue1, ivalue2, ireresult;

    scanf("%d,%d", &ivalue1, &ivalue2 ) ;

    ireresult = ivalue1 / ivalue2;
    printf("result = %d¥n", ireresult);

    return 0;
}
```

, で区切ることを指定

# scanf() の使い方(3) 実行例

```
/*      scanf_test1.c      */

#define _CRT_SECURE_NO_WARNINGS 1
#include <stdio.h>

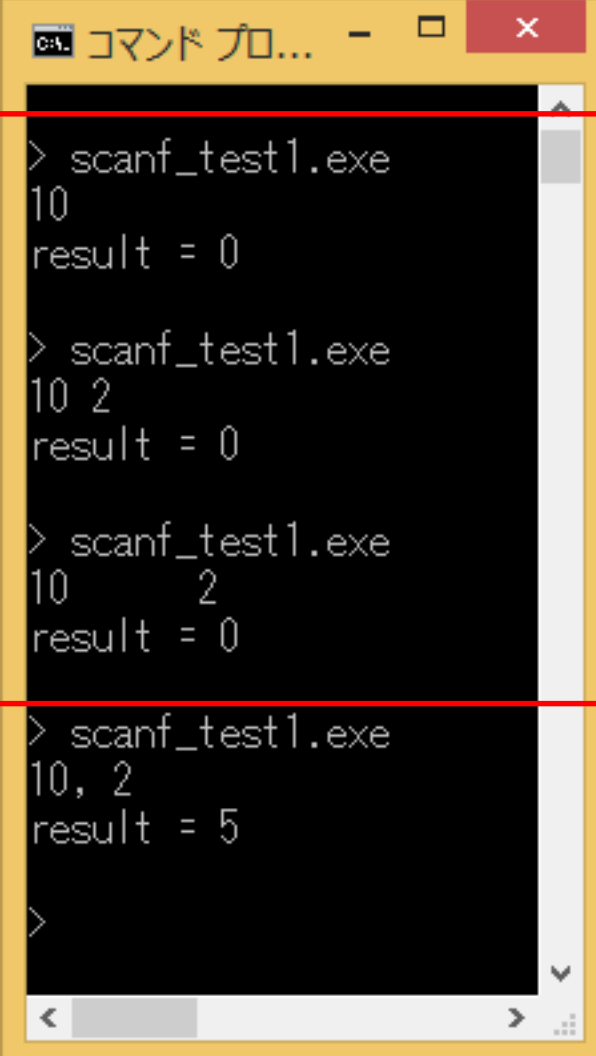
int main( void )
{
    int ivalue1, ivalue2, ireresult;

    scanf("%d,%d", &ivalue1, &ivalue2 ) ;

    ireresult = ivalue1 / ivalue2;
    printf("result = %d¥n", ireresult);

    return 0;
}
```

今度は逆に、一行に2つの数を  
, で区切った場合以外は正しくない。



The screenshot shows a Windows Command Prompt window titled "コマンドプロ..." (Command Prompt). The window contains the following text:

```
> scanf_test1.exe
10
result = 0

> scanf_test1.exe
10 2
result = 0

> scanf_test1.exe
10      2
result = 0

> scanf_test1.exe
10, 2
result = 5

>
```

A red rectangular box highlights the first three execution examples, where the input is either a single number (10) or two numbers separated by a space (10 2). In these cases, the output is "result = 0".



# 区切り記号( , 等) について

- scanf() の区切り記号は慣れないと問題を起こしやすい。
  - この問題は教科書 p.123 に詳しい
- 同様な使い方をする関数 fscanf() でも区切り記号の問題は発生する。
  - fscanf() で " , " 区切りの csv ファイル等を読み込む時などに同様の問題が生じるので、よく理解しておくように。

# scanf() の使い方(4)

## 倍精度実数 double の場合

```
/*      scanf_test2.c      */

#define _CRT_SECURE_NO_WARNINGS 1
#include <stdio.h>

int main( void )
{
    double dvalue1, dvalue2, dresult;

    scanf("%lf", &dvalue1 ) ;
    scanf("%lf", &dvalue2 ) ;

    dresult = dvalue1 / dvalue2;
    printf("result = %f¥n", dresult);

    return 0;
}
```

入力変数が double の  
場合は "%lf" で受ける

# scanf() の使い方(4)

```
/*      scanf_test2.c      */

#define _CRT_SECURE_NO_WARNINGS 1
#include <stdio.h>

int main( void )
{
    double dvalue1, dvalue2, dresult;

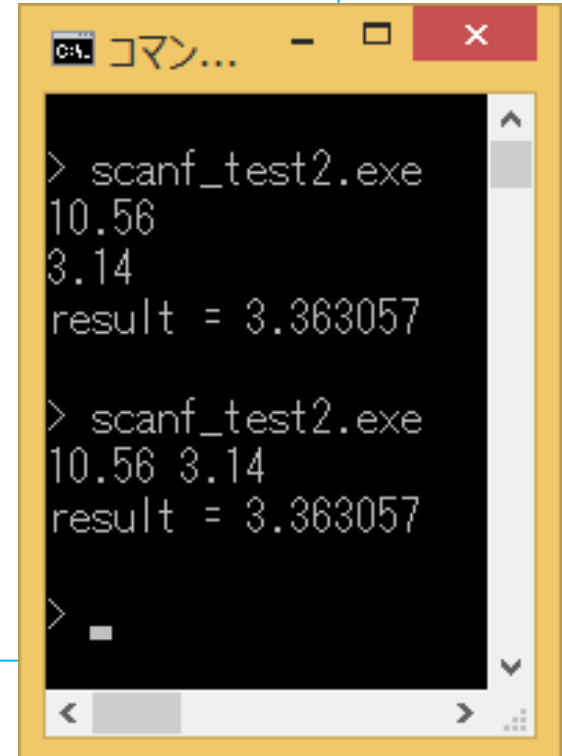
    scanf("%lf", &dvalue1 ) ;
    scanf("%lf", &dvalue2 ) ;

    dresult = dvalue1 / dvalue2;
    printf("result = %f¥n", dresult);

    return 0;
}
```

scanf() の入力変数が double の場合は "%lf" (long float) で受ける

printf() の場合は "%f" でも構わない



```
C:\> scanf_test2.exe
10.56
3.14
result = 3.363057

C:\> scanf_test2.exe
10.56 3.14
result = 3.363057

C:\>
```

# 数学関数の使い方(1)

## 基本

- 戻り値と引数(ひきすう)について
  - 戻り値(計算結果)、引数(入力値)

戻り値

```
double x, y ;
```

```
x = 3.14159265 / 4 ;
```

引数

戻り値を y に  
代入

```
y = sin( x ) ;
```

② 計算結果が変数  
y に代入される

① 変数 x に代入された入力値に  
ついてのサイン関数を計算して

- 今回紹介する数学関数については 戻り値、引数とも double(倍精度実数) で取り扱う。

# 数学関数の使い方(2)

## 三角関数(1)

- 三角関数(1)

- 以下の関数の**引数**の単位はラジアン(rad)である。
  - 角度(deg) ではないので注意。

- 正弦関数    `double        sin( double arg )`
- 余弦関数    `double        cos( double arg )`
- 正接関数    `double        tan( double arg )`

`arg` は「引数 (argument)」の変数(変数名は `arg` である必要はないが、一般的に引数変数をこのように書く)

# 数学関数の使い方(3)

## 三角関数(2)

- 逆三角関数

- 以下の関数の戻り値の単位はラジアン(rad)である。
- 角度(deg) ではないので注意。

• アークサイン	<code>double asin( double arg )</code>
• アークコサイン	<code>double acos( double arg )</code>
• アークタンジェント	<code>double atan( double arg )</code>

- `asin( )`, `acos( )` の引数の範囲は $\pm 1.0$  であることを注意。
- これらの関数の戻り値は  $\pm \pi/2$  であることにも注意 ( $\pm 90$ 度の範囲。第1象限と第4象限の角のみ)

# 数学関数の使い方(4)

## 良く使う数学関数

- 数学関数

- 以下の関数の戻り値も引数も double

• 絶対値	double	<code>fabs( double arg )</code>
• 平方根	double	<code>sqrt( double arg )</code>
• べき乗 ( $p^q$ )	double	<code>pow( double p, double q )</code>
• 指数関数 ( $e^q$ )	double	<code>exp( double q )</code>
• 自然対数	double	<code>log( double arg )</code>
• 切り上げ	double	<code>ceil( double arg )</code>
• 切り捨て	double	<code>floor( double arg )</code>

- `ceil( )`, `floor( )` は整数値への切り上げ・切り捨てであるが、計算結果の入れ物は倍精度実数のままであることに注意。整数変数に代入するには `(int)` でのキャストが必要。

# 数学関数の使い方(5)

## これらの関数を使う場合には (重要)

```
/*  
    math_test1.c  
    正弦関数を表示するプログラム  
*/  
  
#include <stdio.h>  
#include <math.h>  
  
int main( void )  
{  
    double pi = 3.141592653590 ;  
  
    printf( "cos( PI ) = %18.15f¥n", cos( pi ) ) ;  
  
    return 0 ;  
}
```

#include <stdio.h>の次の行に  
この一行を必ず加える！

cos(PI) = -1.0000000000000000