



ce-3. 式, 変数, 入力, 出力

(Cプログラミング応用, 全14回)

<https://www.kkaneko.jp/cc/c/index.html>

金子邦彦



本日の内容



- 例題 1. 自由落下距離
四則演算
- 例題 2. 三角形の面積
浮動小数の変数, 入力文, 出力文, 代入文
- 例題 3. \sin 関数による三角形の面積
ライブラリ関数

今日の到達目標



- プログラムを使って、自分の思い通りの計算ができるようになる
 - 四則演算
 - ライブラリ関数（三角関数, 対数・指数関数など）
- 見やすいプログラムを書くために、ブロック単位での字下げを行う

例題 1 . 自由落下距離



- Win32 コンソールアプリケーションを新規作成する
- その後, C++ソースファイルの編集を行い, 自由落下距離を求めるプログラムを作る
 - 地上で物を落とし始めた後の自由落下距離を求める
 - 重力加速度 g は 9.8 とする
 - 自由落下距離を求めるために, プログラム中に, 計算式 $y = (9.8 / 2.0) * x * x$ を書く

Microsoft Visual Studio C++ の画面構成



The screenshot shows the Microsoft Visual Studio C++ IDE interface. The main window displays the source code for `ConsoleApplication1.cpp`. The code includes the following lines:

```
1 // ConsoleApplication1.cpp : コンソール アプリケーションのエントリ ポイントを定義します。
2
3
4 #include "stdafx.h"
5
6
7 int main()
8 {
9
10
11
12 }
```

Annotations in green boxes highlight key components of the IDE:

- Source Code Editor:** A green box highlights the code editor area with the text "C++ソースファイルの編集はここで行う" (C++ source file editing is done here).
- Solution Explorer:** A green box highlights the Solution Explorer on the right, showing the project structure with the text "ファイルなどが表示される" (Files, etc., are displayed).
- Output Window:** A green box highlights the Output window at the bottom left, with the text "ビルド結果が現れる" (Build results appear).

Other visible elements include the menu bar (File, Edit, View, Project, Build, Debug, Team, Nsight, Tools, Architecture, Test, Analysis, Window, Help), the toolbar, and the Solution Explorer showing the project structure:

- 参照 (References)
- 外部依存関係 (External Dependencies)
- ソース ファイル (Source Files)
 - ConsoleApplication1.cpp
 - stdafx.cpp
- ヘッダー ファイル (Header Files)
 - stdafx.h
 - targetver.h
- リソース ファイル (Resource Files)
 - ReadMe.txt

The Output window shows the following information:

出力元(S):	
その他 (名前)	ConsoleApplication1
プロジェクト ファイル	d:\documents\visual studio 2015\Projects\...
プロジェクトの依存関係	
ルート名前空間	ConsoleApplication1



```
#include "stdio.h"
#include <math.h>
#pragma warning(disable:4996)
int main()
{
    double x;
    double y;
    char buf[256];
    int i;
    double start_x;
    double step_x;
    FILE* fp;
    printf( "start_x = " );
    fgets( buf, 256, stdin );
    sscanf_s( buf, "%lf%n", &start_x );
    printf( "step_x = " );
    fgets( buf, 256, stdin );
    sscanf_s( buf, "%lf%n", &step_x );
    fp = fopen( "d:¥¥data.csv", "w" );
    for( i = 0; i < 20; i++ ) {
        x = start_x + ( i * step_x );
        y = ( 9.8 / 2.0 ) * x * x;
        printf( "x= %f, y= %f\n", x, y );
        fprintf( fp, "x=, %f, y=, %f\n", x, y );
    }
    fprintf( stderr, "file d:¥¥data.csv created¥n" );
    fclose( fp );
    return 0;
}
```

**データファイル名
d:¥¥data.csv
は適切に設定すること**

自由落下距離の
計算を行っている部分



```
#include "stdio.h"
#include <math.h>
#pragma warning(disable:4996)
int main()
{
    double x;
    double y;
    char buf[256];
    int i;
    double start_x;
    double step_x;
    FILE* fp;

    printf( "start_x = " );
    fgets( buf, 256, stdin );
    sscanf_s( buf, "%lf%n", &start_x );
    printf( "step_x = " );
    fgets( buf, 256, stdin );
    sscanf_s( buf, "%lf%n", &step_x );

    fp = fopen( "d:¥¥data.csv", "w" );
    for( i = 0; i < 20; i++ ) {
        x = start_x + ( i * step_x );
        y = ( 9.8 / 2.0 ) * x * x;
        printf( "x= %f, y= %f¥n", x, y );
        fprintf( fp, "x=, %f, y=, %f¥n", x, y );
    }
    fprintf( stderr, "file d:¥¥data.csv created¥n" );
    fclose( fp );
    return 0;
}
```

キーボードからの
データ読み込みを
行っている部分

計算を行っている部分

ファイルへの書き出し
を行っている部分

```
#include "stdio.h"
#include <math.h>
#pragma warning(disable:4996)
int main()
```

Cプログラムはメイン関数から
実行開始



```
double x;
double y;
char buf[256];
int i;
double start_x;
double step_x;
FILE* fp;
```

変数 x , y , buf , i , $start_x$,
 $step_x$, fp をメモリエリア中に確保

プログラムは順次実行

```
printf( "start_x =" );
fgets( buf, 256, stdin );
sscanf_s( buf, "%lf%n", &start_x );
printf( "step_x =" );
fgets( buf, 256, stdin );
sscanf_s( buf, "%lf%n", &step_x );
fp = fopen( "d:¥¥data.csv", "w" );
for( i = 0; i < 20; i++ ) {
```

printf でメッセージを表示
fgets でキーボードから1行を読み込み
sscanf で数値を読み取って変数に格納

printf でメッセージを表示
fgets でキーボードから1行を読み込み
sscanf で数値を読み取って変数に格納

```
    x = start_x + ( i * step_x );
    y = ( 9.8 / 2.0 ) * x * x;
    printf( "x= %f, y= %f%n", x, y );
    fprintf( fp, "x=, %f, y=, %f%n", x, y );
```

20回の繰り返し ($i = 0, 1, \dots, 19$)

```
    }
    fprintf( stderr, "file d:¥¥data.csv created¥n" );
    fclose( fp );
    return 0;
}
```

x の値から
 $(9.8 / 2.0) * x * x$
を求め、 y に書き込む

C++ソースファイルの 書き換えが終わった後の手順



- ビルド

 - 「ビルド」 → 「ソリューションのビルド」

 - ビルドが終了し

 - 「ビルド： 1 正常終了, 0 失敗, 0 スキップ」

 - のように表示されていることを確認

 - さもなければ, プログラム中のミスを疑う

- 実行

 - 「デバック」 → 「デバッグ無しで開始」

 - すると, 新しいウィンドウが開く

実行手順



- 実行すると、新しいウィンドウが現れるので、start_x, step_x の値をキーボードから与える

例えば

start_x = 0

step_x = 0.1

- ウィンドウは消えるが、d: ドライブに data.csv (データファイル) が作成されるので、Excel 等で開き確認する

```
C:\WINDOWS\system32\cmd.exe
start_x = 0
step_x = 0.1
x= 0.000000, y= 0.000000
x= 0.100000, y= 0.043000
x= 0.200000, y= 0.169000
x= 0.300000, y= 0.441000
x= 0.400000, y= 0.784000
x= 0.500000, y= 1.250000
x= 0.600000, y= 1.784000
x= 0.700000, y= 2.401000
x= 0.800000, y= 3.150000
x= 0.900000, y= 3.989000
x= 1.000000, y= 4.880000
x= 1.100000, y= 5.870000
x= 1.200000, y= 6.910000
x= 1.300000, y= 8.040000
x= 1.400000, y= 9.290000
x= 1.500000, y= 10.620000
x= 1.600000, y= 12.040000
x= 1.700000, y= 13.540000
x= 1.800000, y= 15.180000
x= 1.900000, y= 17.888000
file d:\data.csv created
実行するには何かキーを押してください...
```

Excelでデータファイルを開いたとき



	A	B	C	D	E	F	G
1	x=	0	y=	0			
2	x=	0.1	y=	0.049			
3	x=	0.2	y=	0.196			
4	x=	0.3	y=	0.441			
5	x=	0.4	y=	0.784			
6	x=	0.5	y=	1.225			
7	x=	0.6	y=	1.764			
8	x=	0.7	y=	2.401			
9	x=	0.8	y=	3.136			
10	x=	0.9	y=	3.969			
11	x=	1	y=	4.9			
12	x=	1.1	y=	5.929			
13	x=	1.2	y=	7.056			
14	x=	1.3	y=	8.281			
15	x=	1.4	y=	9.604			
16	x=	1.5	y=	11.025			
17	x=	1.6	y=	12.544			
18	x=	1.7	y=	14.161			
19	x=	1.8	y=	15.876			
20	x=	1.9	y=	17.689			
21							
22							
23							
24							

四則演算のための演算子



+	和
-	差
*	積
/	商

例題 2. 三角形の面積



- 底辺と高さを読み込んで、面積を計算するプログラムを作る

例) 底辺が 2. 5, 高さが 5 のとき,

面積 : 6. 2 5

- 底辺, 高さ, 面積を扱うために, 浮動小数の変数を 3 つ使う



```
#include "stdio.h"
#include <math.h>
#pragma warning(disable:4996)
int main()
{
    double teihen;
    double takasa;
    double menseki;
    int ch;
    printf("teihen=");
    scanf("%lf", &teihen);
    printf("takasa=");
    scanf("%lf", &takasa);
    menseki = teihen*takasa*0.5;
    printf("menseki=%f¥n", menseki);
    ch = getchar();
    ch = getchar();
    return 0;
}
```

キーボードからの
読み込み部分

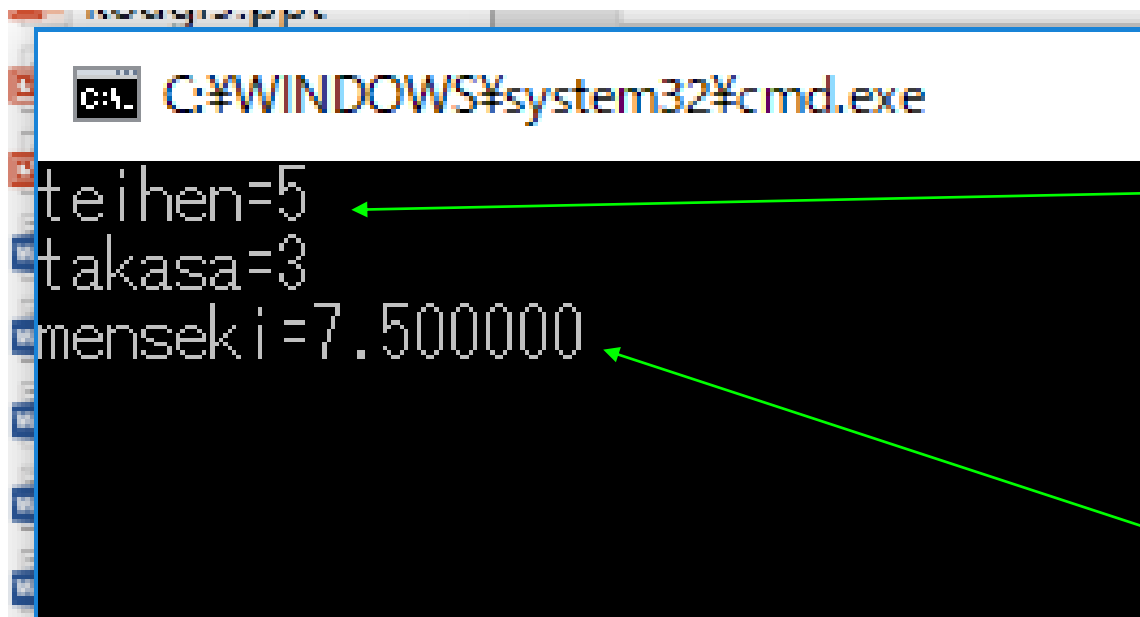
計算部分
出力部分

終了確認のため、
キーボードからの読み込み

実行手順



- 実行すると、新しいウィンドウが現れるので、teihen, takasa の値をキーボードから与える



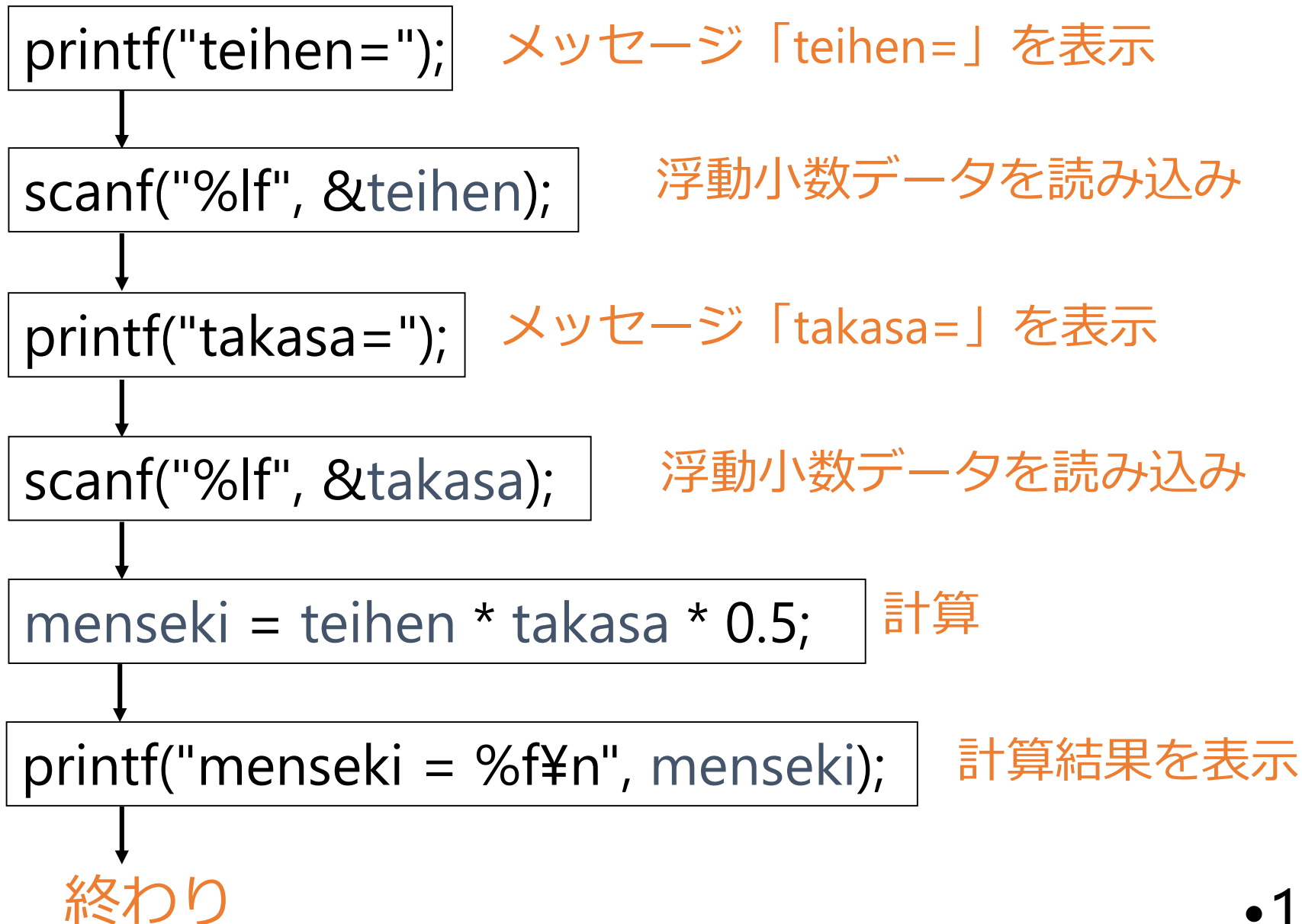
```
C:\WINDOWS\system32\cmd.exe
teihen=5
takasa=3
menseki=7.500000
```

例えば

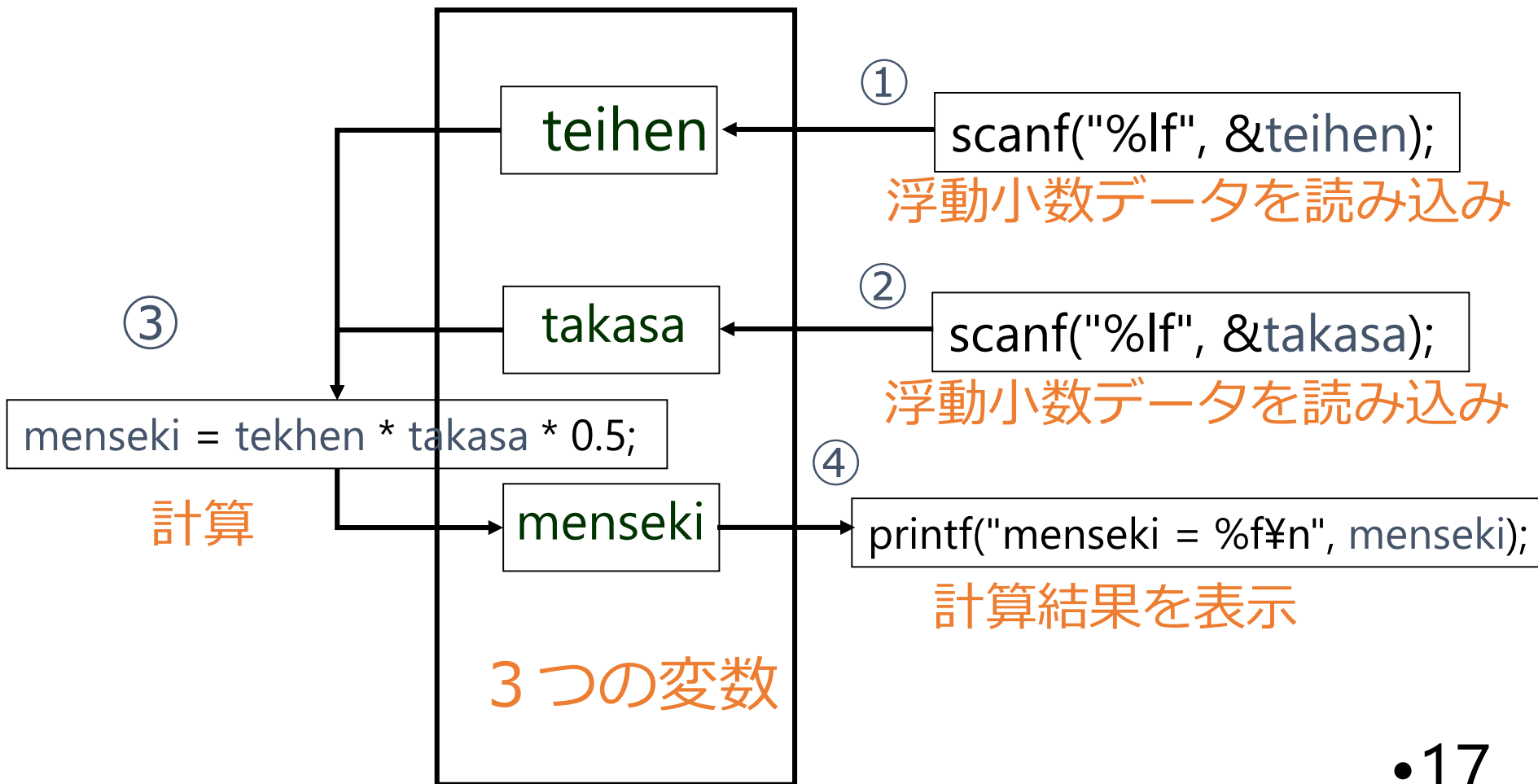
teihen = 5
takasa = 3

menseki の値が
表示されるので、
確認したら
Enter キーを押す

プログラム実行順



メモリ



変数宣言



- 変数は、データを入れるための容器
- 変数宣言とは、変数を使うために、名前と型を書いて、変数の使用をコンピュータに伝えること

```
double teihen;
```

```
double takasa;
```

```
double menseki;
```

} 浮動小数データで、変数名は「teihen」

} 浮動小数データで、変数名は「takasa」

} 浮動小数データで、変数名は「menseki」

「double」とは、浮動小数データという意味。

代入文



- 計算結果 ($\text{teihen} * \text{takasa} * 0.5$) を, 変数 `menseki` に格納する (このことを, 代入という)
- 「=」は, 変数に計算結果等を格納するという意味. 「両辺が等しい」という意味ではない

```
menseki = teihen*takasa*0.5;
```

入力, 出力とは



- 入力

 - データの読み込み

 - (読み込まれたデータは変数に格納される)

- 出力

 - メッセージの表示

 - データの表示

 - (変数に格納されたデータが表示される)

入力文



- 入力文とは, データを読み込むための文
- 書式と読み込むべき変数名を書く
 - 書式 : 浮動小数データを読み込む場合, 書式は「%lf」と書くことになっている
 - 変数名 : 変数名の前には「&」を付けること

```
scanf("%lf", &teihen);
```

書式

&読み込むべき変数名

いろいろな入力



```
double x;  
scanf( "%lf¥n", &x );
```

浮動小数の変数
x への入力

```
double a;  
double b;  
scanf( "%lf¥n", &a );  
scanf( "%lf¥n", &b );
```

浮動小数の変数
a と b への入力

出力文



- 出力文とは、データとメッセージを表示するための文
- 書式と表示すべき変数名を書く
 - 書式： 浮動小数データを表示する場合、書式は「%f」と書くことになっている
 - 変数名： 変数名の前には「&」を付けない (scanf とは違う)

```
printf("menseki=%f¥n", menseki);
```

書式

表示すべき変数名

いろいろな出力



```
printf( "x= ?" );
```

メッセージ 「x= ?」
の表示

```
printf( "x= %f" );  
printf( "y= %f" );
```

「x= 10.0000
y= 20.0000」
のように、メッセージ
と変数の中身を並べて
画面に表示

¥n



- 次の行に進め（改行）という指示
- printf 文などの中で用いる

```
printf("menseki=%f¥n",  
menseki);
```

浮動小数データの使い方



- 変数宣言 :

```
double teihen;
```

```
double takasa;
```

```
double menseki;
```

- 書式 :

```
%lf  – scanf (入力) での書式
```

```
%f   – printf (出力) での書式
```

例題 3 . sin 関数による三角形の面積



- 三角形の 2 辺の長さ a, b とその挟角 θ を読み込んで、面積 S を計算するプログラムを作る
 - 面積を求めるために、 \sin 関数を使う
 - 円周率 $\pi = 3.14159$ とする

$$S = \frac{1}{2} ab \sin \theta$$

```
C:\WINDOWS\system32\cmd
a=4
b=4
theta=90
S = 8.000000
```



```
#include "stdio.h"
#include <math.h>
#pragma warning(disable:4996)
int main()
{
    double a;
    double b;
    double theta;
    double S;
    int ch;
    printf("a=");
    scanf("%lf", &a);
    printf("b=");
    scanf("%lf", &b);
    printf("theta=");
    scanf("%lf", &theta);
    S = 0.5 * a * b * sin( theta * 3.14159 / 180.0 );
    printf("S = %f¥n ", S );
    ch = getchar();
    ch = getchar();
    return 0;
}
```

キーボードからの
データ読み込みを
行っている部分

計算

実行ウィンドウへの表示

終了確認のため、
キーボードからの読み込み

標準ライブラリ関数



- 指数, 対数, 平方根
 - exp 指数関数 (eを底とする指数 z の累乗, e の z 乗)
 - log 対数関数 (底を e とする自然対数の計算)
 - sqrt 平方根
- 三角関数
 - acos 逆コサイン
 - asin 逆サイン
 - atan 逆タンジェント
 - cos コサイン
 - sin サイン
 - tan タンジェント
- その他
 - fabs 絶対値
 - fmod(x,y) 浮動小数データの剰余
 - pow(x,y) べき乗 (x の y 乗)

いろいろな計算



- $y = \sin(x);$

$\sin x$ を計算し, y に格納

```
y = sqrt(x);
```

\sqrt{x} を計算し, y に格納

```
d = sqrt((x * x) + (y * y));
```

$\sqrt{x^2 + y^2}$ を計算し, d に格納

ライブラリ関数の利用



- 計算に関するライブラリ関数を利用するには,

- 次のように, プログラムの先頭部分に書くこと

#include <math.h>



```
#include "stdio.h"
#include <math.h>
#pragma warning(disable:4996)
int main()
{
    double a;
    double b;
    double theta;
    double S;
    int ch;
    printf("a=");
    scanf("%lf", &a);
    printf("b=");
    scanf("%lf", &b);
    printf("theta=");
    scanf("%lf", &theta);
    S = 0.5 * a * b * sin(theta * 3.14159 / 180.0);
    printf("S = %f¥n ", S);
    ch = getchar();
    ch = getchar();
    return 0;
}
```

「度」から「ラジアン」への変換

三角関数ではラジアンを
単位とする

180.0 の「.0」には意味がある
(浮動小数での計算を行うべき
であることをコンピュータに教えている)