

C プログラミング 入門

プログラミング基礎 (2) : 変数・標準入出力 (2)

早稲田大学

今回の目標

- 変数が適切に使える
 - double 型
- int 型 , double 型が混ざった計算が使える
- double 型の変数を画面へ出力する : `printf( );`
- double 型の変数をキーボードから入力する : `scanf( );`
- キャストがわかる

プログラミングの基礎 (1) の例題 4

3つの整数値を入力し，合計値，平均値を表示するプログラム．

- 表示は以下のようにする：
（数字は入力例）

Input three integers:

a: 15 【Enter】

b: 23 【Enter】

c: 32 【Enter】

Sum is 70, Average is 23.3.

- 平均を求める際に除算が行われるが，整数型の除算は小数点以下が打ち切られる．
⇒ 小数点型の数を扱う必要がある．
⇒ double 型の利用．

プログラミングの基礎 (1) の例題 4 :

sumave.c

```
#include <stdio.h>
int main(void){
    double a, b, c, Sum, Ave;           /*変数を宣言*/
    printf("Input three integers:¥n");  /*表示*/
    printf("a:");                       /*入力*/
    scanf("%lf",&a);
    printf("b:");
    scanf("%lf",&b);
    printf("c:");
    scanf("%lf",&c);
    q
    Sum = a + b + c;                    /* 演算・代入*/
    Ave = Sum/3;

    printf("Sum is %.0f, Average is %.1f.¥n",Sum, Ave);
    return 0;
}
```

画面への出力（double の場合）

printf():画面上に文字列や変数値を出力する関数

- 変数値の出力

```
printf(" Sum is %.0f, Average is %.1f.¥n", Sum, Ave);
```

- double 型の変数 Sum, Ave の値を画面上に出力
- double 型は「%f」.
- %.0f は有効数字 0 桁, %.1f は有効数字 1 桁で表示させる .
例えば, %9.2f では, 小数点をいれて 9 桁 (小数点以下 2 桁) で表示させる .

変数の入力（double の場合）

scanf(): キーボードから変数の値を入力する関数（double の場合）

- 文字の入力

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

- double 型の変数 **a** にキーボードから値を入力
- キーボードから入力される値を, 変数文字列に従って変数の中に格納する. double 型は「%lf」
【注意】「%lf」は「エルエフ」. 「イチエフ」ではない.
- 変数名の前に「&」をつける.

int 型と double 型

double 型

- double 型が記憶できるのは約 15 桁の**近似値**。正確な値ではない。
 - (コンピューターは 2 進数で数字を理解するので) 2 進数で (52 桁以内で) 表現しきれない場合は近似値となる。
 - 例) $0.25 = 0.01_{(2)}$, $0.1 = 0.000110\cdots_{(2)}$ 。
- int 型は double 型で代用できるか？
 - int 型 (32bit) の整数は , double 型で正確に表せる。逆はできない。
 - int 型はビット演算・剰余演算可能 , double 型はできない。
 - (扱える範囲が違うため) オーバーフロー , アンダーフローの処理などが異なる。
 - 使用メモリ量が int 型 (32bit) は 4byte , double 型は 8byte (64bit) 。

暗黙の型変換

- 代入時に型が異なると、左側の型に変換される。

```
int a;  
double x = 3.14;  
a = x; /*int 型に変換されるため小数点が切り捨てられる*/
```

- プログラム例

```
#include <stdio.h>  
  
int main(void) {  
    int a;  
    double x = 3.14;  
    a = x;  
    printf("a=%d¥n", a);  
    return 0;  
}
```

- 実行結果

```
[~/work] $ gcc cast1.c  
[~/work] $ ./a.out  
a=3
```

暗黙の型変換

- int 型同士 , double 型同士の演算が基本 . 2 つの型が混在する計算は double 型 (精度の高い型) に変換されて計算される .

$\text{char} < \text{int} < \text{double}$

- プログラム例

```
#include <stdio.h>

int main(void) {
    int a=175, b=100;
    double x = 100.0;
    printf("a/b=%d¥n", a/b);
    printf("a/x=%f¥n", a/x);
    return 0;
}
```

- 実行結果

```
[~/work] $ gcc cast2.c
[~/work] $ ./a.out
a/b=1
a/x=1.75
```

明示的な型変換（キャスト演算子）

除算については以下の注意が必要

- `int / int = int`
例： `5/2 = 2`
- `double / double = double`
例： `5.0/2.0 = 2.5`
- `5/2=2.5` と表示させるためには，キャスト演算子（明示的な型変換）が必要．

`(double) 5/2 = 2.5      /* （型）式 で使う */`
- 上記のようにすれば，計算結果を `double` 型にすることができる．

キャスト演算子使用プログラム例

cast3.c

```
#include <stdio.h>
int main(void){
    int a=10, b=4;
    double c, d;
    c = a/b;
    d =(double) a/b;
    printf("a/b=%d, c=%f, d=%f¥n", a/b, c, d);
    return 0;
}
```

出力結果

- [~/work] \$ gcc cast3.c
- [~/work] \$./a.out
- a/b=2, c=2.000000, d=2.500000

- a,b は int 型なので , キャストしないと a/b も int 型になってしまう .

まとめ

- 変数
 - double 型
- int 型 , double 型が混ざった計算
- double 型の変数を画面へ出力する : printf();
- double 型の変数をキーボードから入力する : scanf();
- キャスト