



情報処理演習II

No. 2

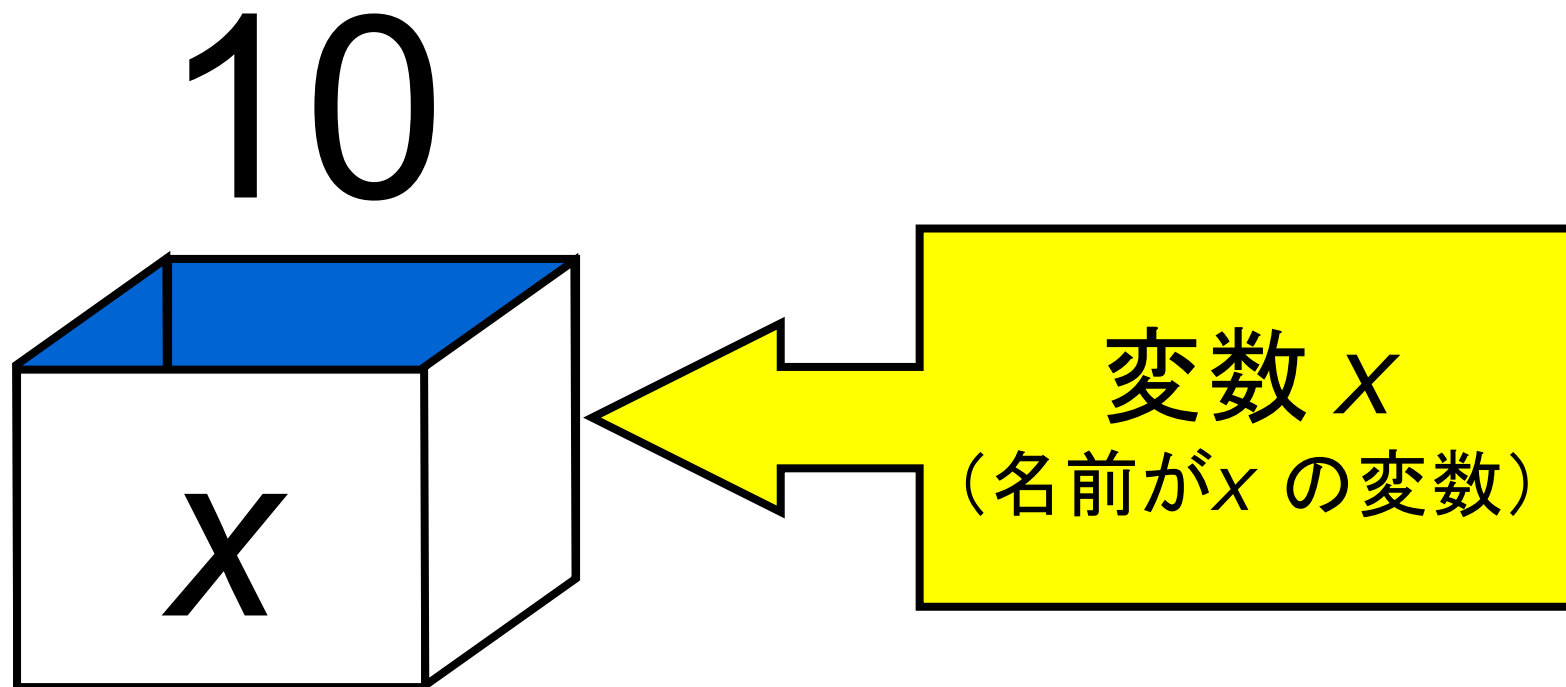
2023.09.26

芝浦工業大学 システム理工学部 機械制御システム学科

担当：桑原

変数

- 変数とは：値（データ）を入れておくための箱。
- 名前がつけられている。
- 変数の値（箱の中身）は自由に変更できる。



変数を使うためには

必ずやること：

- **変数の宣言（変数宣言）**
 - データを格納するための箱を用意する
→データ**型**（どんな種類のデータを入れるか）と**名前**を定義
- **値の代入**
 - 箱にデータを格納する

ここまで終わったら：

- **値の参照（読取り）**
 - 箱に入っているデータを読取る

変数宣言 (1)

変数宣言:

(型宣言子) (変数名);

- 31文字以内で名前を付ける。
例) x、y、z、n、m、i、j、k、num、tensuu
- 変数名の1文字目は英字（大文字, 小文字, **下線“_”**）。
大文字／小文字は区別される。
* “_”は1つの文字として認識されるので、変数名に分かり易い意味を持たせることができる。
例) number_data、price_book
- 2文字目以降には数字も使用可
- 命令や関数名との重複は不可

変数宣言 (2)

変数宣言:

(**型宣言子**) (変数名);

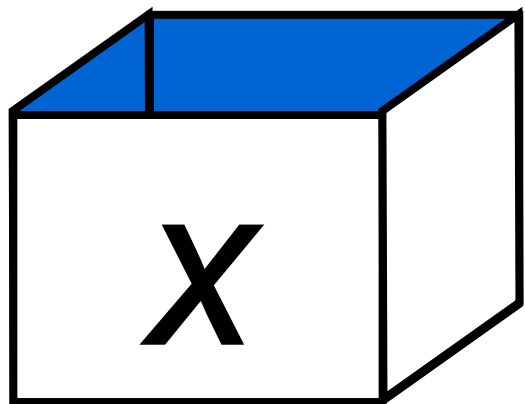
型宣言子は変数のデータ型によって決まる

プログラムを
作る人が
目的に
応じて選ぶ

型宣言子	データ型(データの種類)
char	文字型
int	整数型
float	単精度浮動小数点数型
double	倍精度浮動小数点数型

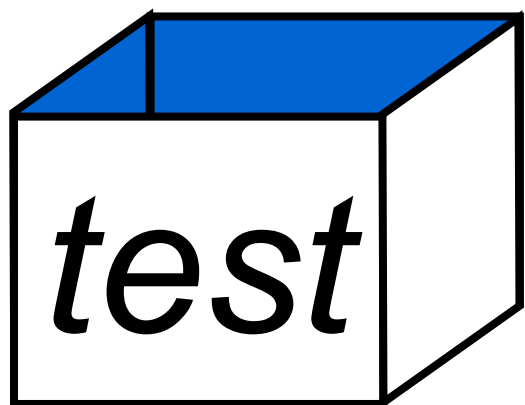
変数宣言 (3)

- 格納する値の種類に応じてデータ型を選ぶ



変数宣言: `int x;`

整数を格納する
変数 `x` (`int`型)



変数宣言: `double test;`

倍精度浮動小数点数を格納する
変数 `test` (`double`型)

なぜ変数にデータ型があるか？

- 変数は、物理的にはメモリ上の領域（ビットの集まり）である
- データ型によって割り当てられるビット数が異なる
- 「ビットをどのように用いてデータを表すか？」はデータ型によって異なる
- よって「変数がどのデータ型を格納するか？」を明確にする必要がある

例) 32bit コンピュータの int 型変数 = 4byte (=32bit) 分のメモリ領域



符号部

31ビット分で数値を指定

例) 32bit コンピュータの double 型変数 = 8byte (=64bit) 分のメモリ領域

IEEE754形式: $(-1)^{(\text{符号部})} \times 2^{(\text{指数部})-1023} \times [1 + (\text{仮数部})]$



符号部

11ビット分で指数部を指定

52ビット分で仮数部を指定

変数へ値を代入：

(変数名) = (値);

- 「=」は「右側を左側に代入する」という意味を持つ。
数学の等号のように「等しい」という意味はない。

- (値) と「変数のデータ型」を一致させる。

- 値の代入は変数宣言の後に行うこと。

例) `int n;` (整数型変数 n を宣言する)

`n=1;` (変数 n に 1 を代入する)

- 変数宣言と値の代入を1行でまとめて書くこともできる

例) `int n = 1;` (整数型変数 n を宣言して 1 を代入)

値の参照

- 値の参照（読取り）の方法は様々

例1) 別の変数の値を代入

int n = 1; （整数型変数 n を宣言し, 1を代入）

int m; （整数型変数 m を宣言）

m = n; （変数 m に変数 n の値を代入. すなわち m に 1 を代入）

例2) printf関数で変数の値を引用してスクリーンに表示する

printf("answer = %d.¥n",x);

（変数 x の値を %d の場所に入れて, 「" "」内を画面に表示させる）

printf関数による値の参照

- 例) `printf("answer = %d.¥n", x);`
- 画面に表示する部分は「" "」内に書き、変数の値を表示する場所に**変換仕様**
(どのデータ型で表示するか)を書く

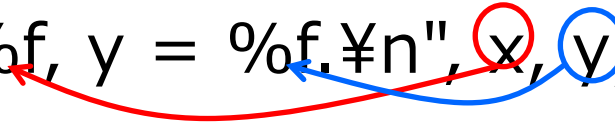
変換仕様	データ型
%d	整数型 → intに対応
%f	単精度・倍精度浮動小数点数 → float/doubleに対応
%c	文字（1文字） → charに対応
%s	文字 列

引用する変数は「" "」の後に「,」を入れて書く

引用が複数ある時は変換仕様と同じ順で列挙。

その場合、変数間も「,」で区切る

例) `printf("x = %f, y = %f.¥n", x, y);`



並べた順で対応

サンプルプログラム

サンプルプログラム 2.1

```
/*
 * 変数宣言と値の代入
 */
#include <stdio.h>

main()
{
    int x;          /* 変数宣言 */
    int y;

    x = 1;          /* 値の代入 */
    y = 2;

    printf("x = %d, y = %d.¥n", x, y);
    printf("x + y = %d.¥n", x+y);
}
```

← まとめて「int x, y;」としてもよい

← 変数宣言のあとには空行を書く

(コーディングスタンダード)

← 変数x,yの値を参照し, その和 x+y を引用する

scanf関数による値の読み込み

例) scanf("%lf", &x);

- double 型変数 x の値をキーボード入力から読み取る。
- printf の文法に似ているが、変数 x の前に「&」がついて「&x」となる。

サンプルプログラム 2

```
(前略)
main()
{
    double x;

    printf("Please input: x =¥n");
    scanf("%lf", &x);
    printf("x = %f.¥n", x);
}
```

注意1:「&」が抜けてもコンパイルエラーは出ないが、実行すると予想通りの動作をしない

注意2: 変換仕様はprintf と若干異なるので注意

変換仕様	データ型
%d	整数型
%f	単精度浮動小数点数
%lf (エルエフ)	倍精度浮動小数点数
%s	文字列