C プログラミング 入門

— プログラミング基礎 (1) ：変数・標準入出力 (1) —

早稲田大学

今回の目標

- C 言語でプログラムを作成する．
- **変数**が適切に使える
  - int 型
  - double 型
  - char 型
- **代入・四則演算**が使える
  - 代入「＝」，加減乗除「＋－×÷」，剰余「%」
- **標準入出力**が使える
  - 画面へ出力する：printf( );
  - キーボードから入力する：scanf( )；

プログラム作成手順

手順１

エディタ Emacs を用いてソースファイルを作成する．

↓

手順２

コンパイラ gcc を用いて実行ファイル作成する．エラー・警告が出た場合はソースファイルを修正する（手順１に戻る）

↓

手順３

プログラムを実行・検証．期待通りに動作しない場合は，ソースファイルを修正する（手順１に戻る）

例題 1

- 「hello.c」 というファイルをエディター（Emacs）で作成．
- ファイルの中身を次のように編集する：

```
#include <stdio.h>
int main(void){
    printf("Hello, world!! ¥n");
    return 0;
}
```

- printf()：文字・値の出力（”. . .”で囲まれた部分を画面上に出力）

```
printf(”Hello, world!!¥n” );
```

- 「¥n」は改行を表す.

コンパイル

コンパイラ gcc を用いて実行ファイルを作成する

```
[~/work] $ gcc hello.c
```

- **エラー・警告が出た場合は，Emacs に戻って修正する．**
- **成功すると「a.out」という名前のファイルが作られる．**
  （警告だけの場合も一応作成されるが，バグがある可能性が高い．
  警告が出なくなるまでソースファイルを見直すこと）

実行ファイルに名前をつけたい場合

```
[~/work] $ gcc hello.c -o hello
```

警告をすべて表示させたい場合

```
[~/work] $ gcc -Wall -o hello hello.c
```

よくあるエラー・警告

hello.c:1:19 error: studio.h:そのようなファイルやディレクトリはありません

- **エラーのある「ファイル名:何行目:何文字目:」が先頭に表示**
- #include<> **の中で指定されているファイル名が間違っている.**

hello.c:3: warning: character constant too long for its type
hello.c:3: warning: 文字での終端を欠いています

- **「 ”」を「 ’」などと間違えている. 括弧の対応が取れていないなど.**

hello.c:3: error: 文法エラー が ”return ” の前にあります
hello.c:3: error: 文法エラー が ’} ’ トークンの前にあります

- **「;」を忘れているなど. 問題の場所が「 の前にあります」と示されている.**

hello.c:3: error: undefined reference to ’print ’

- **「**printf**」を「**print**」と間違えている.**

プログラムの実行・検証

ファイル名を指定して，プログラムを実行する

```
[~/work] $ ./a.out

Hello, world!
```

- **今いるディレクトリにあるファイルの実行には頭に「./」が必要．**

失敗例

```
[~/work] $ ./a.out
Hello, world!/n[~/work] $
```

となった場合「¥」を「/」と間違っている．

変数

- 数値や文字を扱う場合には「変数」を利用し，この中にデータを格納する．
- 変数を利用する場合には，最初に変数の宣言を行う．

型

- 変数には「型」があり，これでデータの性格が規定される．
- int 型：整数 (integer) を１つ記憶しておける．
  （-2,147,483,648〜2,147,483,647 の範囲）
- double 型：小数を有効数字 15 桁の精度で１つ記憶しておける（0 および絶対値で 2.23×10^{-308}〜1.79×10^{308} の範囲）
- char 型：１バイト文字（半角文字）を１文字記憶しておける．

変数

変数名

- アルファベット及び「_」（アンダースコア）を使って作られる（例：ab_c）
- ２文字目以降は 0〜9 の数字も使える（例：a2）
- **予約語**（int, double など）は使えない.（【参考】参照）
- 大文字と小文字は区別される（例：abc と Abc は別物）
- 用途が分かりやすい名前をつけること.

C 言語の予約語（参考）

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

int 型使用プログラム例

int.c

```
#include <stdio.h>
int main(void){
    int Age ;                                   /*変数を宣言*/
    printf("Input your age: ");                 /*表示*/
    scanf("%d", &Age);                          /*入力受付*/
    Age = (Age - 18) * 3+2;                     /*演算・代入*/
    printf("Lucky number is %d.¥n", Age);
    return 0;
}
```

例題 2. 上のプログラムを写し，コンパイル，実行してみよ

- **ファイル名：**int.c **とせよ．**
- [~/work] $ gcc int.c
- [~/work] $ ./a.out
- Input your age : 19 【Enter】
- Lucky number is 5.

プログラムの説明

- 各行について（各行の最後にセミコロン";"を忘れずに！）
  - １～２行目はおまじない（詳細は今後！）
  - ３行目：　int 型変数 Age を用意
  - ４行目：　printf("～") の～を表示．
  - ５行目：　scanf は変数に値を入力する．%d は整数を表し，入力された数値を int 型変数 Age に入力する．
  - ６行目：　「＝」は方程式ではなく，代入を表す．
  - ７行目：　「%d」の箇所に"..."の後に指定した int 型変数 Age の中身を表示してくれる．「¥n」は改行を表す．
  - ８行目以降：　これも決まり文句（詳細は第８回目以降で）
- その他
  - 四則演算の優先順位は数学での優先順位と同じ．
  - 「/*」と「*/」で挟まれた部分はコメントとなり，コンパイラは無視する．(コメントアウト)
  - 「&」は今はおまじない．scanf は&が必要，printf は不要．

画面への出力

printf():画面上に文字列や変数値を出力する関数

- **文字の出力（”...”で囲まれた部分を画面上に出力）**

```
printf(”Input your age: ”);
```

- **変数値の出力（ここでは Age の値の出力）**

```
printf(”Lucky number is %d.¥n”, Age):
```

- int **型の変数** Age **の値を画面上に出力**
- %**で始まる部分を変数文字列という. 変数文字列にはそれぞれ対応する型がある.** int **型は「**%d**」.**
- %d **の部分に変数** Age **の値が表示される.**

キーボードからの入力

scanf():キーボードから変数の値を入力する関数

- **文字の入力**

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

- int **型の変数** Age **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**. int **型は「**%d**」**
- **変数名の前に「**&**」をつける**.

四則演算，代入

算術演算子【+, -, *, /, %】

- 数値の計算を行うために，**算術演算子**【+, -, *, /, %】がある．
- 演算子の優先順位は，数学と同じ．
- int 同士の除算は結果も int になり**小数点以下が切り捨てられる**

- **プログラム例** (operator.c)

```
#include <stdio.h>

int main(void) {
  int a=5, b=3;
  printf("a+b=%d¥n", a+b);/*加算*/
  printf("a-b=%d¥n", a-b);/*減算*/
  printf("a*b=%d¥n", a*b);/*乗算*/
  printf("a/b=%d¥n", a/b);/*除算*/
  printf("a%b=%d¥n", a%b);/*剰余算*/
  return 0;
}
```

- **実行結果**

```
[~ /work] $ gcc operator.c
[~ /work] $ ./a.out
a+b=8
a-b=2
a*b=15
a/b=1
a%b=2
```

四則演算，代入

代入演算子【=】

- 変数の値を代入するには，「=」を用いる．
- 定数の代入：a = 3;
- 変数の代入：a = b;
- 式の代入：Age = (Age - 18) * 3+2;　　/*演算・代入*/
  - 「=」の右側の値を，左側の変数に代入する．
  - 数式によって計算された値が，変数 Age に上書きされる．
- 自身を更新：a = a + 2;
  - a を 2 増やすことを意味する．

例題 3

３つの整数値を入力し，合計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.

- **プログラムの流れは以下の通り：**
  - 4 **つの変数（**a,b,c,Sum**）を宣言する．**
  - **表示，入力を３回繰り返す．(**a,b,c **に数値を入力）**
  - **和** (Sum) **を計算する．**
  - **最後に，和の表示を行う．**

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．
- **次回，**double **型を用いた小数点数の計算を行う．**

まとめ

- C 言語でプログラムを作成．
- **変数**
  - int 型
  - double 型
  - char 型
- **代入・四則演算**
  - 代入「＝」，加減乗除「＋－×÷」，剰余「%」
- **標準入出力**
  - 画面へ出力する：printf( );
  - キーボードから入力する：scanf( )；