プログラミングI
数理物理，総合理学等向け

2018年10月15日

海谷 治彦

目次

- 前回の積み残しと復習
- 演習2の解説
- 演算子の補足: べき算は無い，% 演算子

- if文 (教科書は4章)

- ブロックとは？
- ブロックの入れ子

- 演習3

復習

キーボードから変数に整数を入力

- キーボードから入力した整数値を変数に保存することができる. (まぁ, できないと困るよね.)
- まず, 保存するための変数を定義する.
- 以下に示すような, scanf という関数を用いると, キーボード入力を指定した変数に入れられる.
- 変数の前の & は忘れないように. 詳細は後日に.
- 教科書(レ)には違うやり方が書いてありますが, 無視してください.

```
#include <stdio.h>

int main(void){
  int val;
    scanf("%d", &val);
return 0;
}
```

p.14-15
(明)

(レ)では
複雑な方法を
書いてるが
お勧めしない

プログラム中の注釈 コメント文

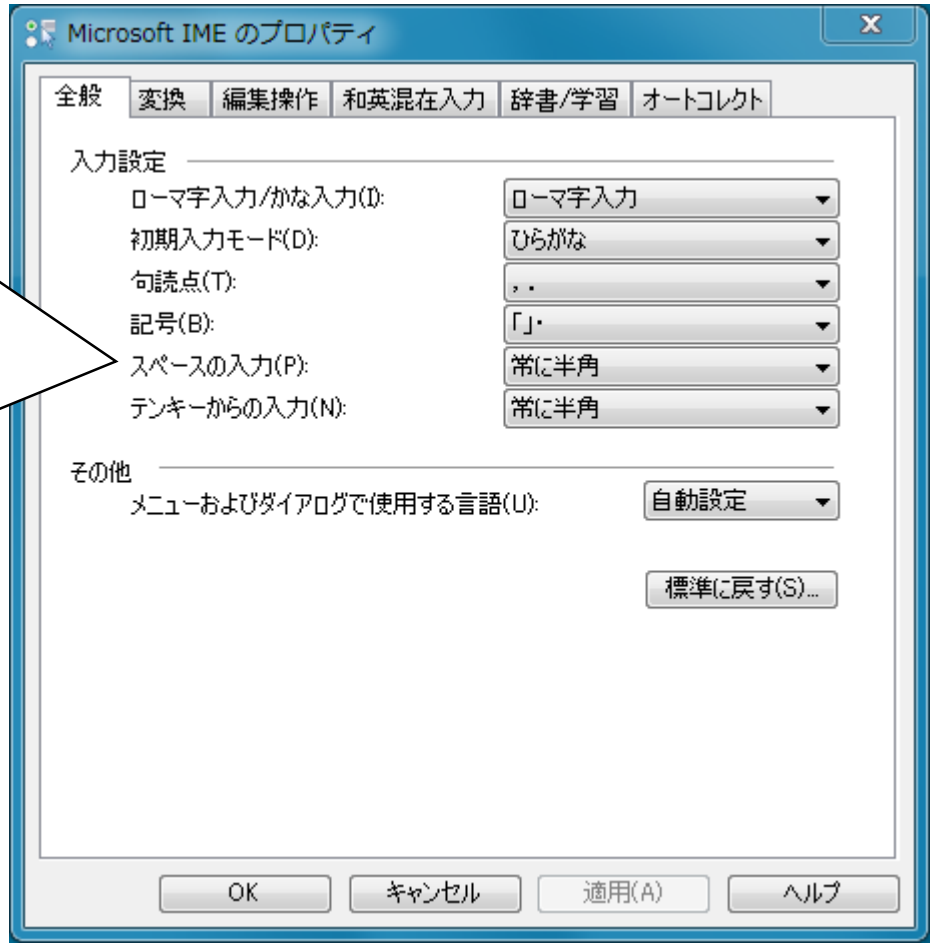
- プログラム中に，人間が後から内容を確認するためのメモを書くことができる.
- これをコメント文と呼ぶ.
- コメント文はメモなのでコンパイル等のコンピュータの動作へは影響を及ぼさない.

- C言語(C99以降)では二種類の書き方が可能
  - /* と */ の間に書く. 複数行にわたり記述可能.
  - // から行末まで書く. 一行のみ.
- 本授業ではコメント文中だけには日本語を書いてもよい. (日本語の空白文字は使わないように)

参考 全角スペースは無効化がベター

少なくとも日本語の空白文字はプログラムでは害悪しかないので，無効化するのがよいと思います．

まぁ，氏名の区切りとかで全角スペースを強要する残念なサイトとかもありますが．orz



コメントの例

```
/*
平均値を計算
Author: kaiya
Date: 18/10/2014
*/
#include <stdio.h>

int main(void){
  float x, y; // floatは精度が悪いのでお勧めしません
  x = 15.0;
  y = 32.0;
  printf(“%0.1f¥n”, (x+y)/2);
  return 0;
}
```

プログラムの空白や改行について

- # で始まる行を除き，プログラムは改行や空白は自由に入れてかまいません．
- // でコメントがある行も注意してください．
- 以下は同じプログラムですが，読みにくい右はお勧めしません．

```
#include <stdio.h>

int main(void)
{
  int x, y;
  x = 15;
  y = 32;
  printf("%d¥n", (x+y)/2);
  return 0;
}
```

```
#include <stdio.h>
int main(void){int x,y;x=15;y=32;
printf("%d¥n",(x+y)/2);return 0;}
```

前回の

演習問題2

- 半径を整数値でキーボードから入力すると，おおよその円の面積と円周の長さを表示するプログラムを作成せよ.
- 円周率は「3」を利用せよ. (「おおよそ」ゆえ)
- ソースプログラム名は circle.c としてください.
- dotcampusにアップしてね.

期待される結果の例

```
$ cc –o circle circle.c
$ ./circle
5
area = 75 circumference = 30
$
```

コレはユーザーが5と入力して，エンターを押すものとしてください.

演習2のポイント

1. 半径の値を人間から受け取る，たとえば変数 r
   - 円周率は 3 でOK
2. 面積の公式 $\pi \cdot r^2 = 3 * r * r$
3. 周長の公式 $2 \cdot \pi \cdot r = 2 * 3 * r$
4. それぞれ表示

- 二乗の演算子は C には無い.
- 必要に応じて，変数はいくつでも準備してよい.
  - pi = 3; 等，一旦変数に保存してもよい.

演算子の補足

- C言語にはべき乗の演算子は無い.
  - XのN乗は，XをN回，掛け算しないといけない.
  - 実は関数としてはある.
- ちなみに，ルート等も演算子としては無い.
  - 実は関数としてある.
- この演習とは関係ないが，C言語には剰余演算子がある.
  - 算数で MOD 等，表現されていたもの.
  - プログラムでは結構重要.
  - % の記号を使う．百分率とは関係ない.

剰余演算 %

p.24-
(明)

- 整数の上で定義される.
- 10で3を割った余りが1, 14を6で割った余りが2等を以下のように表現する.
  - 10 % 3
  - 14 % 6
- 循環する数字を扱うのにすごく便利. 例えば,
- 春=0 夏=1 秋=2 冬=3 と値を振り, 今の季節の値を変数 s に保持するとする.
- s = (s+1) % 4 を計算することで, 常に, 次の季節を表す数値を得ることができる.
- 参考 season1.c

プログラムの基本動作

- プログラムは命令文のリストで構成されてます.
- コンピュータは命令文を書いてある順番に順々に実行します. (原則)
- しかし, それだけでは込み入った処理がかけないので, 以下の二種類があります.
  1. 条件分岐: 条件によって特定の命令を実行せずに飛ばす.
  2. 繰り返し: 前に実行した命令に戻って再度実行する.
- 本日は前者の条件分岐について学びます.

条件分岐の必然性

p.42-
(明)

- プログラムに限らず人間が何か処理する場合でも，条件を見て行動を変えるというのはよくあります．
  - 「雨がふりそうなら」
    - 傘をもつ，そうでなければてぶらでいく．
  - 「お金があれば」
    - タクシーにのる，そうでなければ，歩く．
  - 「体重ふえたら」
    - ダイエットする，そうでなければ何もしない．
  - 「年収が一千万以上なら」
    - 税率は30%，そうでなければ税率は20%．
- 上記のようなことを，ほとんどのプログラム言語では書くことができます．

C言語での条件とは？

- プログラム言語での条件判断のほとんどは，変数に保管している値の大小や同異です．
  - 年齢 ≠ 30
  - 年収 ≧ 10000000
  - 性別 ≡ 男
- 加えて，複数の条件を組み合わせることができます．(組み合わせ方は次回)
  - 年収 ≧ 10000000 かつ性別 ≡ 男
  - 降水確率 ≦ 30 または 昨夜の天気 ≡ 晴れ
- C言語でも文字列の比較ができますが，通常，数値に置き換えて条件判断を簡略化します．
  - “男” を 1 “女”を0とする．
  - “晴れ”を1 “曇り”を2 “雨”を3 にする等

Cでの条件の書き方 (if then)

p.42-
(明)

ある条件が成り立ては,
命令群Aを実行する.

```
// 一般形
if(条件){
   命令群A
}
```

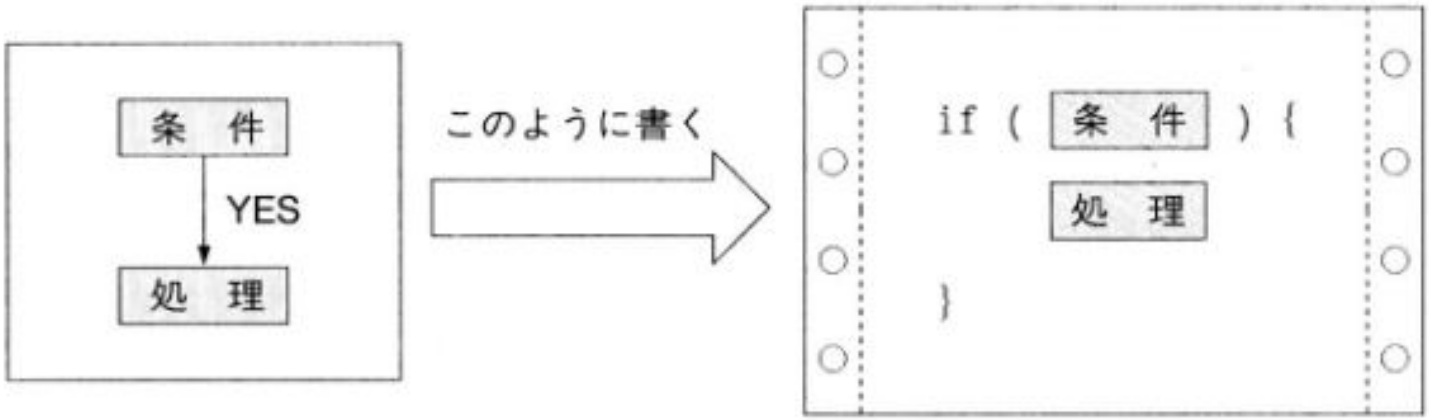
```
// 実例 tax1.c
if(income >= 10000000){ // 収入が一千万以上なら
  tax = income *30/100; // 税率は30%だ
}
printf(“income %d, tax %d¥n”, income, tax);
```

Cでの数値条件の書き方

C言語	数学
n >= 50	n ≧ 50
n <= 50	n ≦ 50
n > 50	n > 50
n < 50	n < 50
n == 50	n = 50
n != 50	n ≠ 50

== 等，独特なので注意.

条件分岐を図で書くと



Cでの条件の書き方 (if then else)

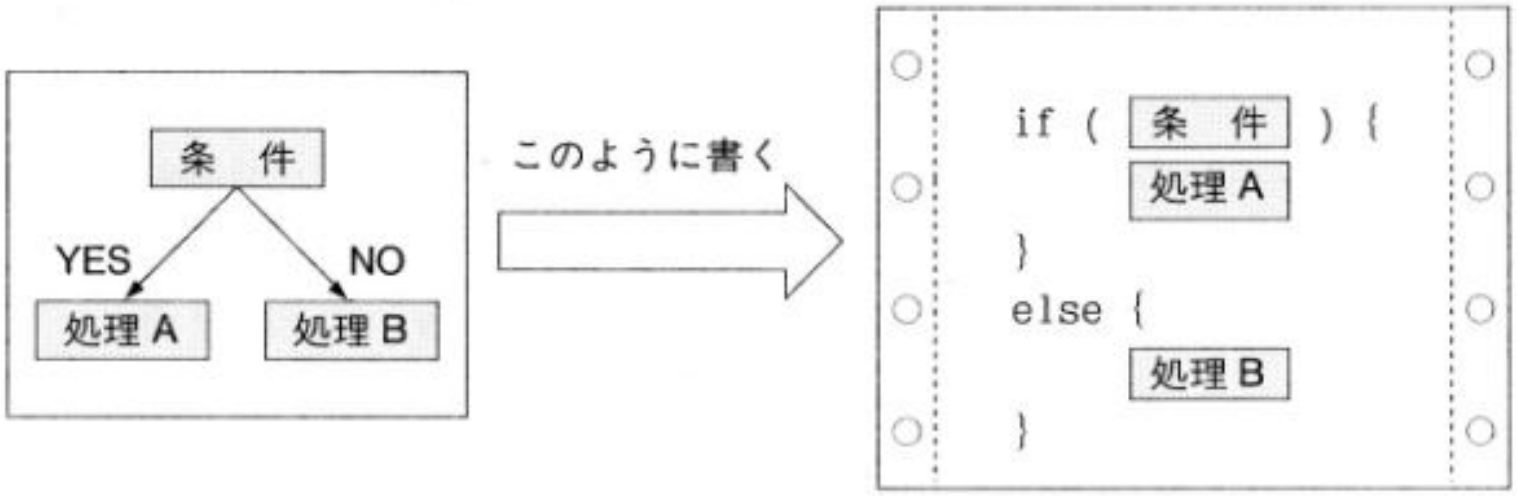
p.44-
(明)

ある条件が成り立ては,
　命令群Aを実行する.
そうでなければ,
　命令群Bを実行する.

```
// 一般形
if(条件) {
   命令群A
}else{
    命令群B
}
```

```
// 実例 tax2.c
if(income >= 10000000){ // 収入が一千万以上なら
  tax = income *30/100; // 税率は30%だ
}else{ // それ以外の場合,
  tax = income *20/100; // 税率は20%だ.
}
printf("income %d, tax %d¥n", income, tax);
```

図で書くと



List 4-1改 p.83

```
bash-3.1$ ./a
Input the probability of rain
30
You don't have to bring your umbrella.
Have a nice day!
bash-3.1$ ./a
Input the probability of rain
55
You have to bring your umbrella.
Have a nice day!
bash-3.1$
```

```c
// list3-1x.c
#include <stdio.h>

int main(void){
    int n;

    printf("Input the probability of rain¥n");
    scanf("%d", &n);

    if(n>=50){ // 降水確率 50%以上なら
        printf("You have to bring your umbrella. ¥n");
    } else { // そうじゃなければ
        printf("You don't have to bring your umbrella. ¥n");
    }
    printf("Have a nice day!¥n");
    return 0;
}
```

変数，スコープ，ブロック

p.58-59
p.145
(明)

- 変数を使うには宣言(定義)しなければならない.
- 宣言が有効なのは，宣言を行ったブロックの内側のみである.
- ブロックとは { と } で囲まれた部分である.
- 別にmainの先頭でなくても，ブロックの先頭で変数は宣言できる.
  - 実は main の外でもできるのだが.
- 変数が利用可能な範囲をスコープ (Scope)と呼ぶ.

- 最も外側のブロックで宣言すれば全体から見えるが，それはあまりお勧めしない方法である.
- 最小スコープで変数を宣言するのが良いとされる.

以下の矩形がそれぞれブロック

```
// list3-1x.c 説明のため改行を増やしている
#include <stdio.h>
int main(void)
{
    int n;

    printf("Input the probability of rain¥n");
    scanf("%d", &n);

    if(n>=50)
    { // 降水確率 50%以上なら
        printf("You have to bring your umbrella. ¥n");
    }
    else
    { // そうじゃなければ
        printf("You don't have to bring your umbrella. ¥n");
    }
    printf("Have a nice day!¥n");
    return 0;
}
```

ブロックに注目した例1

- 収入が1000万円以上の場合は税率が30%
  - ただし子供がいたら，25%
- それ以外は，20%

- 1000万円未満なら子供の有無は聞かなくて良い.

```
bash-3.1$
bash-3.1$ ./a.exe
10000000
Do you have your child(ren)? Yes=1 No=0: 1
income 10000000, tax 2500000
bash-3.1$ ./a.exe
10000000
Do you have your child(ren)? Yes=1 No=0: 0
income 10000000, tax 3000000
bash-3.1$ ./a.exe
8000000
income 8000000, tax 1600000
bash-3.1$
```

```
int main(void)
{ // 実例 tax3.c
int income, tax;
  scanf("%d", &income);
  if(income >= 10000000)
  { // 収入が一千万以上なら
  int hasChild=0;
    printf("Do you have your child(ren)? Yes=1 No=0: ");
    scanf("%d", &hasChild);
    if(hasChild==1)
    {
      tax = income *25/100; // 子供のおかげで税金安い
    }
    else
    {
      tax = income *30/100; // 税率は30%だ
    }
  }
  else
  {
    tax = income *20/100; // 税率は20%だ.
  }
  printf("income %d, tax %d¥n", income, tax);
  return 0;
}
```

ブロックに注目した例2

- 降水確率50%以上なら，有無をいわさず「傘もってけ」と助言．
- それ以外でも，
  - 昨日，雨だったら「傘もってけ」と助言．
  - そうでなければ，「傘は不要」と助言．

```
bash-3.1$ cc list4-1y.c
bash-3.1$ ./a
Input the probability of rain
55
You have to bring your umbrella.
Have a nice day!
bash-3.1$ ./a
Input the probability of rain
30
Did it rain yesterday? Yes=1, No=0: 1
You have to bring your umbrella.
Have a nice day!
```

```
bash-3.1$ ./a
Input the probability of rain
35
Did it rain yesterday? Yes=1, No=0: 0
You don't have to bring your umbrella.
Have a nice day!
bash-3.1$
```

```
#include <stdio.h>

int main(void){ // list4-1y.c
  int n;

  printf("Input the probability of rain¥n");
  scanf("%d", &n);

  if(n>=50){ // 降水確率 50%以上なら
    printf("You have to bring your umbrella. ¥n");
  }
  else { // そうじゃなければ
  int isRained=0;
    printf("Did it rain yesterday? Yes=1, No=0: ");
    scanf("%d", &isRained);
    if(isRained==1){
      printf("You have to bring your umbrella. ¥n");
    }
    else{
      printf("You don't have to bring your umbrella. ¥n");
    }
  }
  printf("Have a nice day!¥n");
  return 0;
}
```

ブロックのメリット

- 条件判断を複数組み合わせることで，複雑な判断を行なうことができる.
  - 新幹線の切符の自動販売機等も，先に入力された情報によって，次に聞いてくることが違うでしょ？
  - 所謂，ウィザードを構築する基本的な仕組み.
  - 前述の例では，1000万円未満の収入では，子供の有無は聞かれない.
- 必要な変数の有効範囲を明確にできる.
  - 前述の例では，1000万円以上の場合のみ，子供の有無を記録する変数が定義(宣言)される.
  - 大規模なプログラムでは，変数の有効範囲をできるだけ狭くしないと修正が大変になる.
    - 1000万行先で代入間違えてても気づかないでしょ？

本日の演習4

- 身長と体重を入力し，身長から体重を引いた値が，
  - 100以下の場合，「やせろ」 yasero でも可
  - それ以外は「ふとれ」 futore でも可

と生意気に助言するプログラムを作成せよ.

期待される結果の例

```
$ cc –o sbmi sbmi.c
$ ./sbmi
175
80
height = 175 weight = 80
Lose your weight!
$
```

コレはユーザーが
175と80を入力して，
それぞれエンターを押す
ものとしてください.

本日は以上