
プログラミング基礎演習

第2回

C言語編

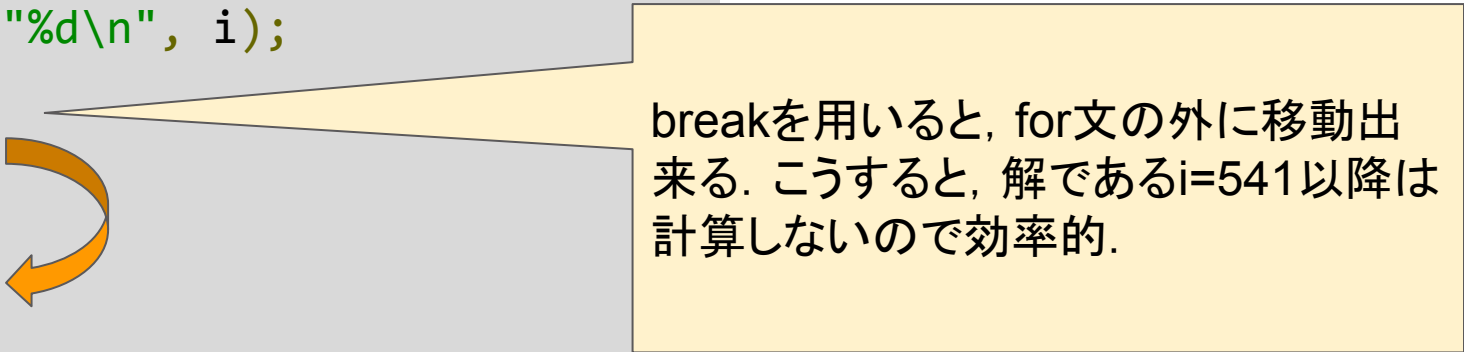
【関数】

長谷川禎彦

前回の課題2の解説

```
#include <stdio.h>

int main() {
    int n = 158340421;
    int i = 0;
    for(i = 0; i < 1000; i++) {
        if(i*i*i == n) {
            printf("%d\n", i);
            break;
        }
    }
    return 0;
}
```



breakを用いると, for文の外に移動出来る. こうすると, 解である $i=541$ 以降は計算しないので効率的.

break文の注意

- break文を使って抜けられるのはループひとつだけ。多重ループを抜けるには、複数回breakする必要がある。

前回の課題3の解説

```
#include <stdio.h>
int main() {
    int x,y,z;
    int max = 1000;
    for (x = 1; x < max; x++) {
        for (y = x; y < max; y++) {
            for (z = y; z < max; z++) {
                if (x*x + y*y == z*z) {
                    printf("%d^2 + %d^2 = %d^2\n", x, y, z);
                }
            }
        }
    }
    return 0;
}
```

y を x から開始することで、重複表示を防ぐ

while文

```
while (条件式) {  
    文;  
}
```

- 条件式が真である間, 文は繰り返し実行される
 - 条件式が偽の場合は文は実行されない
-

while文の例

```
#include <stdio.h>
int main() {
    int i = 1;
    int sum = 0;
    while(sum<1000) {
        printf("%d\n",i);
        sum = sum + i;
        i++;
    }
    return 0;
}
```

switch文

```
switch (変数) {  
    case 定数1:  
        文1;  
        break;  
    case 定数2:  
        文2;  
        break;  
    . . .  
    case 定数n:  
        文n;  
        break;  
    default:  
        default文;  
        break;  
}
```

変数の値が
定数1 と等しければ、文1 実行
定数2 と等しければ、文2 実行
定数n と等しければ、文n 実行
それ以外ならば、default文実行

switch文の例

```
#include <stdio.h>
int main() {
    int x = 1;
    switch(x) {
        case 0: printf("x is 0\n"); break;
        case 1: printf("x is 1\n"); break;
        default: printf("otherwise\n");
    }
    return 0;
}
```

xの値によって実行する文が変わる

このbreakがないと下の文もすべて実行してしまう

数学関数

- sin, cos, exp等の数学関数は, math.hというヘッダーが必要

```
#include <stdio.h>
#include <math.h>
int main() {
    printf("%f\n", sin(10));
    printf("%f\n", exp(2.5));
    return 0;
}
```

数学関数

- コンパイル時に`-lm`オプションを付ける必要がある
 - 例: `gcc -Wall -o kadai kadai.c -lm`

Warning allの略. 全てのコンパイル警告を表示させる. バグが見つかりやすくなる. 毎回つけるようにする.

`math.h`を使うために必要.

良く使うmath.hの関数

- `double pow(double x, double y)`
 - `x` の `y` 乗を計算する。
- `double sqrt(double x)`
 - `x` の平方根
- `double cos(double x)`
- `double sin(double x)`
- `double tan(double x)`
- `double exp(double x)`
- `double log(double x)`
- `double fabs(double x)`
 - `x` の絶対値

課題1 (必須課題)

以下の級数から円周率を計算せよ

$$\frac{\pi^2}{6} = \sum_{n=1}^{\infty} \frac{1}{n^2 2^{n-1}} + (\ln 2)^2$$

math.hの関数を用いる必要がある

関数

- `sin`も`pow`も全て関数.
- プログラム言語では自作の関数を作成出来る.
- 関数を用いることで, 良く用いる機能をまとめることが可能

```
戻り値の型 関数名 (引数リスト) {  
    ステートメント...  
}
```

関数

関数は引数のデータになんらかの処理をして戻り値を返す.

引数

```
int add(int a, int b) {
```

```
    int x;
```

```
    x = a + b;
```

```
    return x;
```

```
}
```

この場合
引数の型は(int, int)

関数が返す値の型

関数の例

```
int gcd(int a, int b) {  
    ...  
}
```

gcdは整数をとって、整数を返す

```
void print_data() {  
    ...  
}
```

返値がない場合はvoidという型

関数のプロトタイプ宣言

- プロトタイプ宣言とは、「関数の仕様にあたる引数と返り値の型だけを抜き出したもの



- コンパイル時のチェックのため
- 関数を使う場所が、関数の定義より前にくる場合がある

プロトタイプ宣言

```
#include <stdio.h>
```

```
int add(int, int);
```

```
int main() {  
    int n;  
    n = add(2,3);  
    return 0;  
}
```

```
int add(int a, int b) {  
    int x;  
    x = a + b;  
    return x;  
}
```

引数の型と戻り値の型だけあらかじめ宣言（プロトタイプ宣言）

関数の定義は、使う場所より後でもよい

プロトタイプ宣言の注意

```
void func(void) ;
```

引数を取らない場合は, 引数のところにちゃんと
voidと書く

課題2(必須課題)

Zellerの公式で曜日を計算せよ. h 百 y 年 m 月 d 日
(2016年11月3日の場合
, $h=20$, $y=16$, $m=11$, $d=3$)としたとき, 曜日は

$$w = y + \left\lfloor \frac{y}{4} \right\rfloor + \left\lfloor \frac{h}{4} \right\rfloor - 2h + \left\lfloor \frac{13(m+1)}{5} \right\rfloor + d$$

w を7で割った余りが曜日である(1を日曜, 2を月曜
.....とする). 但し, 1月及び2月の場合は前年の
13月及び14月と考える.

```
#include <stdio.h>
#include <math.h>
int zeller(int, int, int);
int main() {
    int year = 2016, month = 10, day = 5;
    int w;
    w = zeller(year, month, day);
    printf("%d/%d/%d is ", month, day, year);
    switch(w) {
        case 0: printf("Sat.\n"); break;
        case 1: printf("Sun.\n"); break;
        case 2: printf("Mon.\n"); break;
        case 3: printf("Tue.\n"); break;
        case 4: printf("Wed.\n"); break;
        case 5: printf("Thurs.\n"); break;
        case 6: printf("Fri.\n"); break;
    }
    return 0;
}
```

ひな型不要の人は使わなくてもOK

```
int zeller(int year, int month, int day) { /*この関数を埋める*/ }
```

キャスト

- 強制的に別の型に変換したいときに、キャストという操作を用いる.
- doubleをintにキャストすると、0方向に丸める
 - つまり小数部分を捨てる
- 例

```
double w = 3.9;  
int x = (int) w;
```

この時 x は 3 となる

キャストしている

キャスト

- 逆にintからdoubleの場合には, 自動的に行われる

```
int x = 10;
```

```
double y = x;
```

この時 y は10.0となっている.

課題3(必須課題)

1-1000まで素数かどうかをチェックするプログラムを書け.

```
#include <stdio.h>
int is_prime(int); /*引数が素数の場合1, それ以外の場合0を返す*/
int main() {
    int i;
    for (i = 2; i <= 1000; i++) {
        if (is_prime(i) == 1) {
            printf("%d is prime\n", i);
        }
    }
}
int is_prime(int n) {
    //ここにコードを書く
}
```

is_prime関数の実体を作る

ひな型不要の人は使わなくてもOK

課題3のヒント

- 自然数 n の素数判定
 - n より小さい2以上自然数全てで割り切れない
 - ただしこれは無駄も多い

出力例

```
2 is prime
3 is prime
5 is prime
7 is prime
11 is prime
13 is prime
17 is prime
...
```


課題4(自由課題)

● オイラー積と円周率

- 円周率は素数に関する積によって求めることが出来る
- オイラー積を用いて, 円周率を計算するプログラムを書いてみよ

$$\prod_{p \in \text{prime}} \left(1 - \frac{1}{p^2}\right) = \frac{6}{\pi^2}$$

オイラー積

$$\left(1 - \frac{1}{2^2}\right) \left(1 - \frac{1}{3^2}\right) \left(1 - \frac{1}{5^2}\right) \left(1 - \frac{1}{7^2}\right) \left(1 - \frac{1}{11^2}\right) \cdots = \frac{6}{\pi^2}$$

素数

素数

素数

素数

素数

課題4(自由課題)

- 2から10000以下の素数に対してオイラー積を計算してみる.
- 1 / 3などのint割るintに注意
- 平方根は`sqrt()` 関数.
 - `sqrt`関数は`math.h`をインクルードする必要がある.
 - コンパイル時に`-lm`オプションを付ける必要がある.
`gcc -Wall -o kadai02 kadai02.c -lm`
 - 表示は
`pi = xxxx`
のように表示させよ. `xxxx`部分が求めた円周率.

課題5(自由課題)

ユークリッドの互除法を実装せよ(最大公約数gcdを求めるアルゴリズム).

```
#include <stdio.h>
int gcd(int, int);
int main() {
    int r;
    int a = 8733;
    int b = 64681;
    r = gcd(a,b);
    printf("gcd(%d, %d) = %d\n", a, b, r);
    return 0;
}
int gcd(int a, int b) {
    /*この関数を埋める*/
}
```

ひな型不要の人は使わなくてもOK

ユークリッドの互除法

[pseudo code]

```
gcd(a, b)
  while b  $\neq$  0
    t := b
    b := a mod b
    a := t
  return a
```

課題6(自由課題)

前回のピタゴラス数を表示するプログラムでは、互いに素ではない結果も表示された.

例えば

$$3^2 + 4^2 = 5^2$$
$$6^2 + 8^2 = 10^2$$

互いに素な場合のみを表示するように改良せよ

課題7(自由課題)

- ゴールドバッハ予想を数値的に確かめてみる.
ゴールドバッハ予想＝「4以上の全ての偶数は、二つの素数の和で表すことができる」

$$4 = 2 + 2$$

$$6 = 3 + 3$$

$$8 = 3 + 5$$

$$10 = 7 + 3$$

...

課題7(自由課題)

以下のように1000までの偶数について表示させよ

4 = 2 + 2
6 = 3 + 3
8 = 3 + 5
10 = 3 + 7
12 = 5 + 7
14 = 3 + 11
16 = 3 + 13
18 = 5 + 13
20 = 3 + 17
22 = 3 + 19
24 = 5 + 19
26 = 3 + 23
...
98 = 19 + 79
100 = 3 + 97

複数の表し方あるものもあるが、一つで良い。もちろん複数表示するように改良しても良い。

課題一覽

- 課題1：必須課題
 - 課題2：必須課題
 - 課題3：必須課題
 - 課題4：自由課題
 - 課題5：自由課題
 - 課題6：自由課題
 - 課題7：自由課題
-

課題提出方法

- 課題xの答えをkadai0x.cファイルに書く.
 - xは1,2,3,...
- 締め切りは日曜日の23:59
- ファイルをzipファイルにまとめる
 - ファイル名: 学籍番号.zip
 - 例: 学籍番号が641234の場合, 641234.zipとする.
 - 圧縮方法はホームページに記載してある
- 学籍番号.zipファイルのみを提出する
 - 提出は <https://goo.gl/QJ3LMP>

課題提出方法

- 時間内に終わらない場合は、途中まででも期限内に提出する
 - 努力の跡が確認できれば出席点は付く