2015/06/10

第６回 Ｃ言語勉強会

配列

丸山翔平

今回の内容

1. おさらい
2. 配列とは
3. 練習問題
4. 解答例
5. おまけ問題

１．おさらい

1．おさらい

●配列を学ぶにあたって，思い出して欲しい事

変数とデータ型

データの型 = 書式

int	整数	（%d）
double	浮動少数	（%lf）
char	文字	（%c）

変数 = 名前付きの入れ物

データ型 変数名 = 格納するデータ

```
int  x  =  32;
```

宣言・代入・参照の3ステップ

繰り返し構文

指定回数の繰り返し処理

```
for( 初期化; 条件式; 継続処理){
    繰り返したい処理
}
```

初期化　最初の状態
条件式　どうしたら終わるか
継続処理　一周したときの処理

意味がわからないよ！という場合は

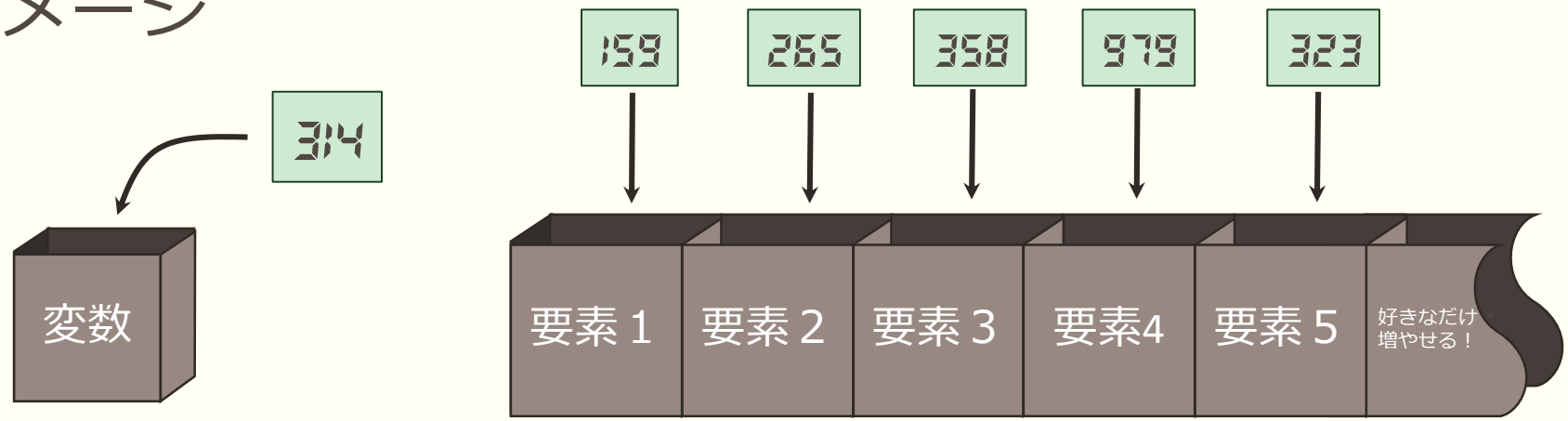
```
for(i = 0; i < n; i++){
   処理内容 ;
}
```

という並びをとりあえず覚えておこう

２．配列とは

2－1．配列とは

イメージ



変数：
　たった一つの値を収納

配列：
　任意の数の値を収納

こんなときに便利！

・毎日の出費を収納させたい

変数では…　　int day1 = 357, day2 = 410, day3 = 259;

配列を使うと…　int day[3] = {357, 410, 259};

２－２．配列の使い方

●配列も，
「宣言」「初期化（代入）」「参照（呼出）」

初期化

代入

参照

```
int array[5] = {1, 10, 100, 1000, 10000};


array[3] = 120;


printf( “%d” , array[2]);
int a = array[3] + array[4];
```

2－2．配列

●配列も，
「宣言」「初期化（代入）

```
int array[5];
array[0] = 1;
array[1] = 10;
array[2] = 100;
array[3] = 1000;
array[4] = 10000;
```

初期化

```
int array[5] = {1, 10, 100, 1000, 10000};
```

代入

```
array[3] = 120;
```

参照

```
printf( “%d” , array[2]);
int a = array[3] + array[4];
```

2－3．使用例

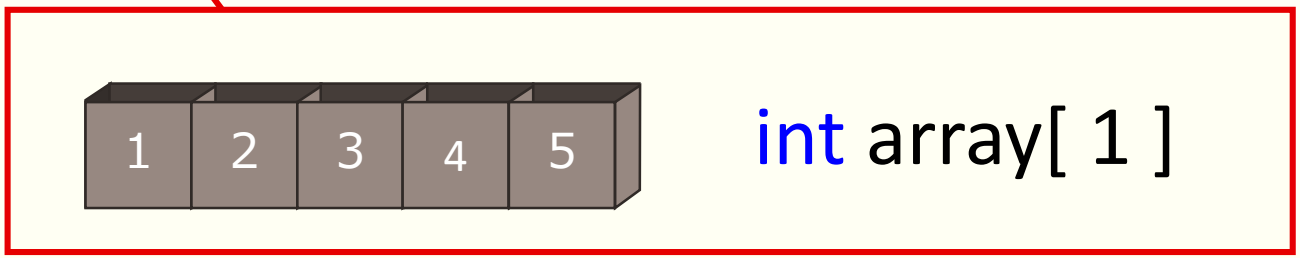
●三日間の降水量を格納し，表示する

例：1日目：15.2 mm，2日目：3.5 mm，3日目：45.3 mm

```
#include <stdio.h>

int main(){
    double prec[3] = {15.2, 3.5, 45.3};

    printf("1日目の降水量は，%.1lf mmです¥n", prec[0]);
    printf("2日目の降水量は，%.1lf mmです¥n", prec[1]);
    printf("3日目の降水量は，%.1lf mmです¥n", prec[2]);

return 0;
}
```

2－4．配列とfor文

●インデックス（要素番号）を変数で指定できる




```
#include <stdio.h>

int main(){
    double prec[3] = {15.2, 3.5, 45.3};
    int n = 2;

    printf( “3日目の降水量は, %.1lf mmです¥n”, prec[n]);

return 0;
}
```

２－４．配列とfor文

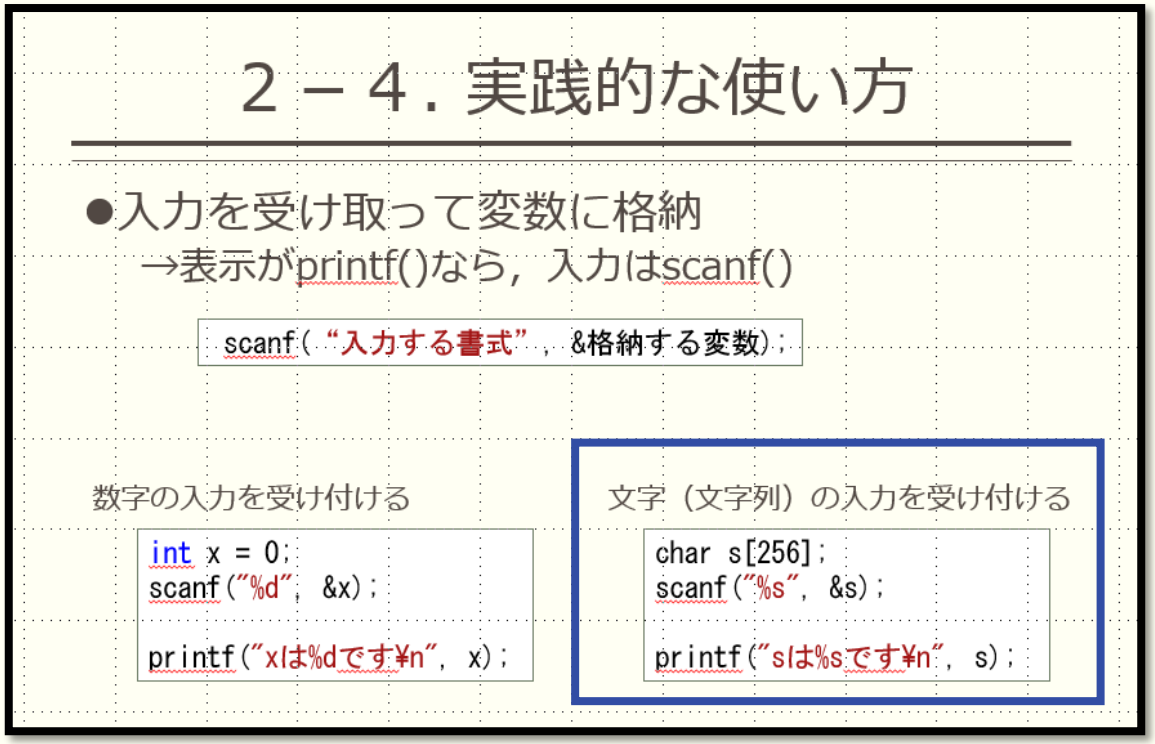
●繰り返し構文でスマートに表示・代入

```
#include <stdio.h>

int main(){
    double prec[3] = {15.2, 3.5, 45.3};
    int i, num[3];

    for(i = 0; i < 3; i++){
        printf(“%d日目の降水量は, %.1lf mmです¥n”, i+1, prec[i]);

        num[i] = i;
    }

    return 0;
}
```

2－5．配列と文字列

array[ ] = … という形，前にも見たことが…？！

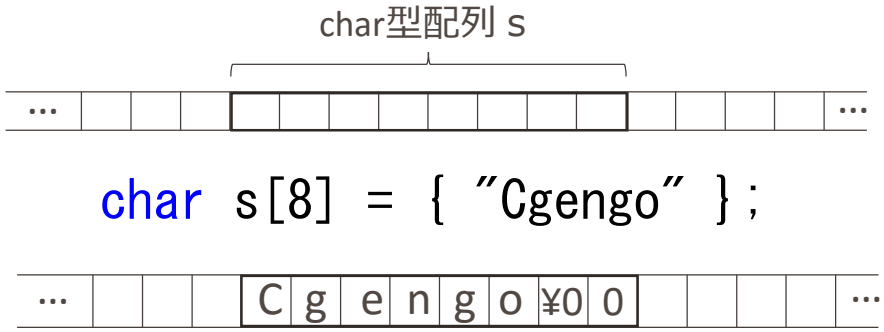
文字列の場合も，値と一緒で
指定されなかった場所には
0が格納される



2 – 5．配列と文字列

```
char s[8]
```

→要素数8でchar型の配列 s を宣言



文字列の場合も，値と一緒で
初期化時に指定されなかった
場所には０が格納される

使い方

数字の入力を受け付ける

```
int x = 0;
scanf("%d", &x);

printf("xは%dです¥n", x);
```

文字（文字列）の入力を受け付ける

```
char s[256];
scanf("%s", &s);

printf("sは%sです¥n", s);
```

2－6．配列と関数

```c
#include <stdio.h>

double average(double x[]);

int main(){
    double prec[3] = {15.2, 3.5, 45.3};
    int i;

    for(i = 0; i < 3; i++)
        printf("%d日目の降水量は，%.1lf mmです¥n", i, prec[i]);

    printf("三日間の平均降水量は，%.2lf mmです¥n", average(prec));
    return 0;
}

double average(double x[]){
    int i;
    double sum = 0.0;
    for(i = 0; i < 3; i++)
        sum += x[i];

    return sum / 3;
}
```

関数に配列の要素をまとめて渡すことが出来る

関数に配列を渡すとき → 引数は［］をつけない
引数をつけると要素（値）を渡す

2－7．配列でできること

- 二次元配列
  - 表やグラフの作成に便利
  - int array2d[5][5]というように宣言すると，
    5×5の25要素を持った配列を用意出来る．

```
int array2d[5][5];
   for(x = 0; x < 5; x++){
      for(y = 0; y < 5; y++){
         array2d[x][y] = (x+1) * (y+1);
   }
}
```

	1	2	3	4	5
1	1	2	3	4	5
2	2	4	6	8	10
3	3	6	9	12	15
4	4	8	12	16	20
5	5	10	15	20	25

- 配列とファイル操作
  - 膨大な数のデータ処理
  - .csvファイルへ出力してグラフを描かせる
    - たとえば，一年間の降水量を一気に処理させたりできる

3．練習問題

3．練習問題

- $f_{(x)} = x^3 - 4x^2 + 6x + 2$ の計算結果を，$1 \leqq x \leqq 10$の範囲で求めよう

出力例

```
x    |  1   2   3   4   5   6   7   8   9  10
f(x) |  5   6  11  26  57 110 191 306 461 662
```

- 「2－7．配列でできること」の二次元配列を参考にして，掛け算九九表を作ってみよう

```
   |  1  2  3  4  5  6  7  8  9
---+--------------------------
0  |  1  2  3  4  5  6  7  8  9
1  |  2  4  6  8 10 12 14 16 18
2  |  3  6  9 12 15 18 21 24 27
3  |  4  8 12 16 20 24 28 32 36
4  |  5 10 15 20 25 30 35 40 45
5  |  6 12 18 24 30 36 42 48 54
6  |  7 14 21 28 35 42 49 56 63
7  |  8 16 24 32 40 48 56 64 72
8  |  9 18 27 36 45 54 63 72 81
```

- for文は，配列に格納するためのものと，配列の要素を表示するためのものの二つ用意する必要があるよ
- 九九表の出力例のような表示が難しかったら，赤く囲った部分だけ表示させてもok

４．解答例

４．解答例

- $f(x) = x^3 - 4x^2 + 6x + 2$ の計算結果を，$1 \leqq x \leqq 10$の範囲で求めよう

```
#include <stdio.h>

int main(){
    int x, fx[10], i;

    for(x = 1; x <= 10; x++){
        fx[x-1] = x * x * x - 4 * x * x + 6 * x + 2;
    }

    printf("x  |");
    for(i = 0; i < 10; i++){
        printf("%4d", i+1);
    }
    printf("¥nfx |");
    for(i = 0; i < 10; i++){
        printf("%4d", fx[i]);
    }

    return 0;
}
```

４．解答例

- 「２－７．配列でできること」の二次元配列を参考にして，掛け算九九表を作ってみよう

```c
#include <stdio.h>

int main(){
    int m[9][9], i, j;

    for(i = 0; i < 9; i++){
        for(j = 0; j < 9; j++){
            m[i][j] = (i + 1) * (j + 1);
        }
    }

    printf("  |  1  2  3  4  5  6  7  8  9¥n");
    printf("--+--------------------------¥n");
    for(i = 0; i < 9; i++){
        printf("%d |", i);
        for(j = 0; j < 9; j++){
            printf("%3d", m[i][j]);
        }
        printf("¥n");
    }
    return 0;
}
```

５．おまけ問題

5．おまけ問題

- 2 – 5のコードを参考にして，3日間の降水量を表示し，さらに平均降水量との差も表示するプログラムを考えてみよう．

- ２つの配列prec[3], differ[3]を用意する
  prec[3] には降水量を，differ[3] には全て0.0を代入しておく．
- 関数calc_dif(double prec[], double dif[])内では，
  降水量の平均値を計算し，平均との差をdif[]に格納していく．

出力例

```
１日目の降水量は，15.2 mmで，平均との差は–6.13 mmです．
２日目の降水量は，  3.5 mmで，平均との差は-17.83 mmです．
３日目の降水量は，15.2 mmで，平均との差は 23.97 mmです．
```

コード例

main()関数

```c
#include <stdio.h>

void calc_dif(double x[], double y[]);

int main(){
    double prec[3] = {15.2, 3.5, 45.3}, differ[3] = {0.0, 0.0, 0.0};
    int i;

    dif(prec, differ);
    for(i = 0; i < 3; i++){
        printf("%d日目の降水量は, %3.1lf mmで,
                平均との差は%4.2lf mmです¥n", i, prec[i], differ[i]);

    }
    return 0;
}
```

calc_dif()関数

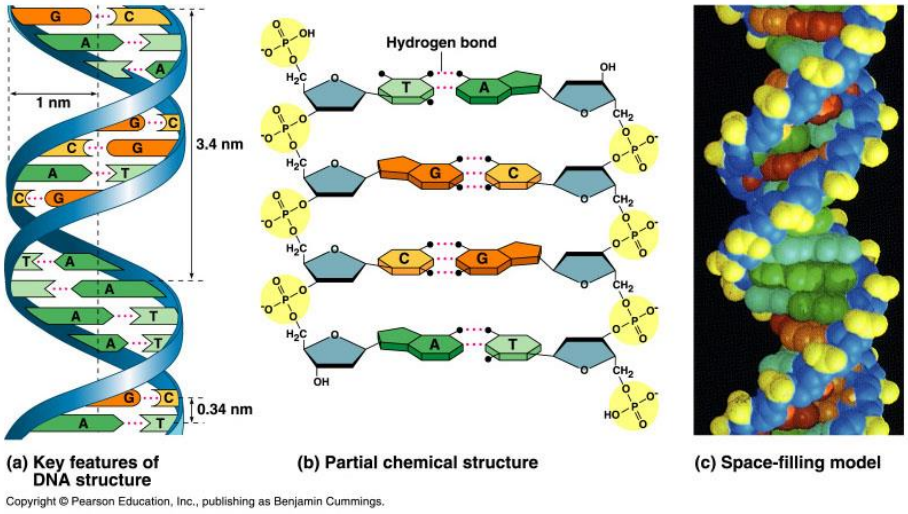
```c
void calc_dif(double x[], double y[]){
    int i;
    double sum = 0.0, average;
    for(i = 0; i < 3; i++)
        sum += x[i];

    average = sum / 3;

    for(i = 0; i < 3; i++)
        y[i] = x[i] - average;
}
```

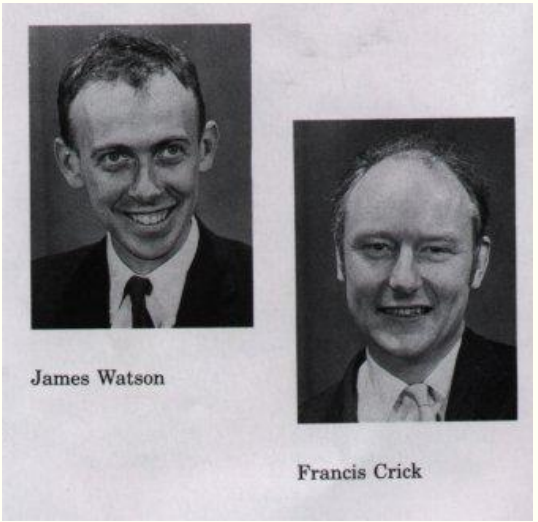
変数ではできなかった,
関数間での複数の値のやりとりが可能
=実質的にいくつかのデータを関数が返せる

5．おまけ問題そのに

- Watson-Crick base pairメーカー

  遺伝情報を記憶しているDNAは，ふつう二本鎖の形状をとっている．遺伝情報は4種類の塩基(base；A,T,G,C)によって記述されている．二本鎖の形成はAとT, GとCという決まったペアどうしの結合によってでき，この対になる法則の発見者の名前をとってワトソン-クリック塩基対（Watson-Crick base pair）と呼ばれている．





5．おまけ問題そのに

- Watson-Crick base pairメーカー

今，100塩基からなるDNA断片

```
TATACGACTCACTATAGATA
GATAGGGCTGGTTAATCGTT
TTAGAGCTAGAAATAGCAAG
TTAAAATAAGGCTAGTCCGT
TATCAACTTAGAAAAATCAG
```

●もとの配列を格納するsense[101]と，相補鎖を格納するantisense[101]を宣言する．
●sense[101]には左の塩基情報を格納し，antisense[101]には何も格納しない
●繰返しと条件分岐をうまく用いて，AならT，GならC…というようにantisense[101]に反対の文字を格納していく

が解析により得られた．遺伝情報の正確な理解には，このDNAの塩基情報とともに，相補対のDNAの塩基情報も不可欠である．そこで，断片に対応する塩基情報を出力するプログラムを作製したい．出力例とヒントをもとに，塩基対メーカーをつくってみよう

出力例

```
sense     : TATACGACTCACTATAGATAGATAGGGCTGGTTAATCGTTTTAGAGCTAGAAATAGCAAGTTAAAATAAGGCTAGTCCGTTATCAACTTAGAAAAA
antisense : ATATGCTGAGTGATATCTATCTATCCCGACCAATTAGCAAAATCTCGATCTTTATCGTTCAATTTTATTCCGATCAGGCAATAGTTGAATCTTTTT
```

コード例

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


int main(){
    char sense[101] =
{"TATACGACTCACTATAGATAGATAGGGCTGGTTAATCGTTTTAGAGCTAGAAATAGCAAGTTAAAATAAGGCTAGTCCGTTATCAACT
TAGAAAAA"};
    char antisense[101];
    int i;


    for(i = 0; i <= 100; i++){
        if(sense[i] == 'A')
            antisense[i] = 'T';
        else if(sense[i] == 'T')
            antisense[i] = 'A';
        else if(sense[i] == 'G')
            antisense[i] = 'C';
        else if(sense[i] == 'C')
            antisense[i] = 'G';
        else
            antisense[i] = '¥0';
    }

    printf("sense     : %s¥n", sense);
    printf("antisense : %s¥n", antisense);

    return 0;
}
```