2013 年 5 月 30 日（木）実施

2 次元配列

これまで argv[1][0]のように具体例では出していたが，添え字を 2 個持つ配列を **2 次元配列**と呼ぶ。2 次元配列を次のように宣言することにより，

```
char moji[5][11];
```

最大 10 文字(及び NULL 文字'¥0'１個分)の文字列を 5 個用意することができる。この場合のメモリ上の配置は,moji[0][0],moji[0][1], ... ,moji[0][10],moji[1][0], ... ,moji[1][10], ... , moji[4][10]の順となる。なお，2 次元配列に対して moji[j]と記述すると，&moji[j][0]を意味する。この例では，これは添え字の 0 番を 1 番目と数えれば，j + 1 番目の文字列に対応する。

0 回以上の繰り返しのプログラム

for 文

C 言語で 0 回以上の繰り返しのプログラムを実現するための文として，**for 文**と **while 文**とが用意されている。for 文の構文は次のようになる。

```
for（ 初期設定式; 継続条件式; 再設定式 ） 文
```

先ず，初期設定式を実行し，続いて継続条件式を評価する。ここで，継続条件式が真であれば，繰り返しの対象となる文を実行する。次に再設定式を実行した上で，再度，継続条件式を評価する，という繰り返しを行う。継続条件式が真でなければ，繰り返しの対象となる文を実行せず，for 文から抜け出す。最初から継続条件式が真でなければ，全く文を実行しないので，0 回以上の繰り返しと呼ばれる。

for 文は通常，**特定の処理を決まった回数繰り返す目的**で用いられる。

while 文

while 文の構文は次のようになる。

```
while（ 継続条件式 ） 文
```

先ず，継続条件式を評価して，真であれば，繰り返しの対象となる文を実行し，再度，継続条件式を評価する，という繰り返しを行う。継続条件式が真でなければ，繰り返しの対象となる文を実行せず，while 文から抜け出す。最初から継続条件式が真でなければ，全く文を実行しないので，0 回以上の繰り返しと呼ばれる。

while 文は通常，**処理を特定の状態になるまで繰り返す目的**で用いられる。例えば，ファイルを終わりまで読む場合や，キーの入力待ちを行う場合がその典型的な例である。

例題１（for 文を用いた特定の処理の決まった回数の繰り返し）

次のソースプログラムをテキストエディタで入力して，prog7-1.c の名前を付けて保存する。それを翻訳・編集して実行形式のファイルを作成し，実行せよ。

```
/* prog7-1.c */
#include <stdio.h>
#define KOSUU 5

int main(void)
{
    int i;
    double data[KOSUU], goukei=0.0, heikin;

    for (i=0; i<KOSUU; i++)
    {
        printf("%d 番目の実数データを入力してください： ", i+1);
        scanf("%lf", &data[i]);

        goukei = goukei + data[i];
    }

    heikin = goukei / KOSUU;

    for (i=0; i<KOSUU; i++)
        printf("%d 番目のデータ ： %f¥n", i+1, data[i]);

    printf("データの合計： %f¥n", goukei);
    printf("データの平均： %f¥n", heikin);

    return 0;
}
```

【解説】

1. #define は**マクロ定義**を行うための**プリプロセッサ命令**である。このプログラムで書かれている #define KOSUU 5 は，プログラムが翻訳される際に，プログラム中の KOSUU を全て 5 で置き換えることを要求している。

2. ++ は**増分演算子**と呼ばれ，i++ は i の値を 1 だけ増加させる。

3. %lf は，scanf によってキーボードから入力したデータを倍精度の浮動小数点数に変換して格納するための変換指定である。ここで，l(エル)は long を意味する。

＊ printf では，浮動小数点数に対応する変換指定は，単精度，倍精度共に%f であるので，混同しないように！

例題２（while 文を用いた特定の状態になるまでの処理の繰り返し）

次のソースプログラムをテキストエディタで入力して，prog7-2.c の名前を付けて保存する。

それを翻訳・編集して実行形式のファイルを作成し，実行せよ。なお，このプログラムは quit から始まる文字列を入力しない限り，次の文字列の入力を促して処理を続ける。

```
/* prog7-2.c */
#include <stdio.h>
#include <string.h>
#define NAGASA 81
#define KAISUU 5

int main(void)
{
    int i,j;
    char buf[NAGASA], buf2[NAGASA];

    printf("半角の英数字で文字列(80 文字以内) ¥n を入力してください： ");
    scanf("%s", buf);

    while (strncmp(buf, "quit", 4) != 0)
    {
        for (j=0; j<KAISUU; j++)
        {
            i=0;
            while (buf[i+1]!='¥0')
            {
                buf2[i]=buf[i+1];
                i++;
            }

            buf2[i]=buf[0];
            buf2[i+1]='¥0';
            printf("buf2：%s¥n", buf2);

            strncpy(buf, buf2, i+1);
        }

        printf("次の文字列を入力してください： ");
        scanf("%s", buf);
    }

    return 0;
}
```

【解説】

1. string.h は文字列操作のためのライブラリ関数 strncmp 及び strncpy を利用するために組み込んでおく必要がある。

2. strncmp(文字列 1, 文字列 2, 比較文字数)は文字列 1 と文字列 2 とを先頭からの比較文字数分だけ比較するライブラリ関数である。strncmp(buf, "quit", 4)であれば，buf に quit から始まる文字列が格納されているとき，0 が返される。

3. strncpy(複写先配列, 複写元文字列, 複写文字数)は複写元文字列から複写文字数文だけの文字列を複写先配列に複写するライブラリ関数である。

演習１

for 文を用いて 10 人分の成績判定を行うプログラムの空欄 1)　　, 2)　　, 3)　　を埋めてソースプログラムを完成させ，テキストエディタで入力して，ex7-1.c の名前を付けて保存する。それを翻訳・編集して実行形式のファイルを作成し，実行せよ。なお，成績判定の仕方は prog6-1.c に準ずるものとする。

```
/* ex7-1.c */
#include <stdio.h>
#define 1)　　　　 10

int main(void)
{
  int 2)　　, tensuu[NINZUU], rank;

  for (i=0; i<NINZUU; i++)
  {
    printf("¥n %d 人目¥n", i+1);
    printf("点数を 0 点から 100 点の範囲の整数で入力してください： ");
    scanf("%d", &tensuu[i]);

    if (tensuu[i] < 0 || tensuu[i] > 100)
    {
      printf("入力された点数は範囲外です。入力し直してください¥n");
      i--;
    }
    else
    {
      rank = 3)　　　　　/10;

      switch(rank)
      {
        case  6: printf("点数が%d 点なので，成績評価は[可]です。¥n", tensuu[i]); break;
        case  7: printf("点数が%d 点なので，成績評価は[良]です。¥n", tensuu[i]); break;
        case  8: printf("点数が%d 点なので，成績評価は[優]です。¥n", tensuu[i]); break;
        case  9:
        case 10: printf("点数が%d 点なので，成績評価は[秀]です。¥n", tensuu[i]); break;
        default: printf("点数が%d 点なので，成績評価は[不可]です。¥n", tensuu[i]);
      }
    }
  }

  return 0;
}
```

＊ -- は**減分演算子**と呼ばれ，i-- は i の値を 1 だけ減少させる。

演習 2

while 文を用いてある人数分の名前と電話番号とを入力し，一覧を表示するプログラムの空欄 1)　　，2)　　，3)　　を埋めてソースプログラムを完成させ，テキストエディタで入力して，ex7-2.c の名前を付けて保存する。それを翻訳・編集して実行形式のファイルを作成し，実行せよ。なお，プログラムの仕様は次のとおりである。

> 名前と電話番号とはそれぞれ別の char 型の 2 次元配列（[人数]×[文字数＋1]）に格納するものとし，名前として quit から始まる文字列を入力すると終了するものとする。ただし，人数は最大限 10 名までとし，#define で定義して，配列の要素数には NINZUU を用いる。while 文の中では配列の添え字を i とし，i が NINZUU に達した場合には，break 文を実行して，while 文から抜け出すものとする。

```
/* ex7-2.c */
#include <stdio.h>
#include <string.h>
#define 1)　　 15
#define 2)　　 10

int main(void)
{
    int i=0,j;
    char namae[NINZUU][NAGASA], denwa[NINZUU][NAGASA];

    printf("%d 番目の人の名前を入力してください：", i+1);
    scanf("%s", namae[i]);

    while (strncmp(namae[i], "quit", 4) != 0)
    {
        printf("%d 番目の人の電話番号を入力してください：", i+1);
        scanf("%s", denwa[i]);

        i++;
        if (i == 3)　　) break;

        printf("%d 番目の人の名前を入力してください：", i+1);
        scanf("%s", namae[i]);
    }

    if (i > 0) printf("¥n 電話番号簿¥n");

    for (j=0; j<i; j++)
        printf("%2d) 名前：%-14s  電話番号：%-14s¥n", j+1, namae[j], denwa[j]);

    return 0;
}
```

提出物：

1) 例題 1，2 の出力結果をコピーして貼り付けたテキストファイル res7.txt

2) 演習 1 のソースプログラムのファイル ex7-1.c の完成版

3) 演習 2 のソースプログラムのファイル ex7-2.c の完成版