2006 年 10 月 19 日（木）実施

関数

main 関数

main 関数は，C 言語で記述されたプログラムが翻訳編集されて実行される際に，プログラム開始処理によって呼び出される関数である。従って，C 言語で記述されたプログラムでは，プログラムの本体として位置付けられる。

main 関数には次の 2 通りの定義がある。ここで，**仮引数**（parameter）とは関数定義に用いられる引数の呼び名で，関数の呼び出しの際に当てはめられる引数を**実引数**（actual parameter）と呼び，両者を区別する。

1）仮引数を持たない場合

```
int main(void)
{
    ・・・
}
```

前回までのプログラムは，全てこの場合である。

2）2 つの仮引数を持つ場合

```
int main(int argc,char *argv[])
{
    ・・・
}
```

このプログラムを実行する際に，プログラム名 プログラム引数 1・・・プログラム引数 n と入力したとすると，argc の値は n+1 となる。argc が 1 以上のとき，argv[0] ～ argv[argc-1]は文字列への**ポインタ**である。ここで，ポインタはメインメモリ上でデータが格納されている場所のアドレスを格納するためのものである。このとき，argv[0]はプログラム名の文字列を指し，argc が 2 以上のとき，argv[1] ～ argv[argc-1]はプログラム引数の文字列を指す。

ライブラリ関数

C 言語で予め用意されている関数群をライブラリ関数と呼び，前回までに標準入出力に関するライブラリ関数として，printf 及び scanf の簡単な使い方を学んだ。

ユーザ定義関数

プログラムを作成する側で定義する関数をユーザ定義関数と呼ぶ。ユーザ定義関数の取り扱いに関して，次のような特徴がある。

1. ユーザ定義関数には，データを受け取る仮引数とその関数を呼び出した側にデータを戻す**戻り値**とを設定できる。
2. ユーザ定義関数の利用に際しては，戻り値と仮引数のデータ型を示すプロトタイプ宣言を行う。
   例）　int wa(int,int);　　/* int 型の仮引数を 2 つ持ち，戻り値が int 型のユーザ定義関数 wa のプロトタイプ宣言 */
3. 仮引数を持たないユーザ定義関数には，それを示すために()内に void を用いる。
4. 戻り値のないユーザ定義関数には，データ型として void を用いる。

5. ユーザ定義関数の関数定義は，次の形式となる。

戻り値のデータ型 関数名(仮引数1のデータ型 仮引数1,・・・,仮引数nのデータ型 仮引数n)
{ ・・・ }

例)
```
int wa(int x,int y)
{
    int z;
    z=x+y;
    return z;
}               /* 仮引数x及びyに呼び出し側の実引数データを受け取り,それら
                   の和を求めてzに代入し，zの値を呼び出し側に戻す関数waの定義 */
```

例題1

次のプログラムを入力し，翻訳・編集して実行形式のファイルを作成し，実行せよ。ここで，ソースプログラム名は prog3-1.c とする。また，実行に際しては，prog3-1 data1 data2 と入力すること。

```
/* prog3-1.c */
#include <stdio.h>

int main(int argc,char *argv[])
{
    printf("argcの値は%dです。¥n", argc);

    printf("argv[0]が指している文字列はプログラム名「%s」です。¥n", argv[0]);

    printf("argv[1]が指している文字列は第1プログラム引数「%s」です。¥n", argv[1]);

    printf("argv[2]が指している文字列は第2プログラム引数「%s」です。¥n", argv[2]);

    return 0;
}
```

例題2

次のプログラムを入力し，翻訳・編集して実行形式のファイルを作成し，実行せよ。ここで，ソースプログラム名はprog3-2.c とする。

```
/* prog3-2.c */
#include <stdio.h>

int wa(int,int);
int sa(int,int);
int seki(int,int);
int shou(int,int);
int amari(int,int);

int main(void)
{
    int x,y;

    printf("1つ目の整数を入力してください： ");
    scanf("%d", &x);
```

```
    printf("2つ目の整数を入力してください： ");
    scanf("%d", &y);

    printf("%d + %d => %d¥n", x,y,wa(x,y));
    printf("%d - %d => %d¥n", x,y,sa(x,y));
    printf("%d * %d => %d¥n", x,y,seki(x,y));
    printf("%d / %d => %d¥n", x,y,shou(x,y));
    printf("%d %% %d => %d¥n", x,y,amari(x,y));

    return 0;
}

int wa(int a,int b)
{
    return a+b;
}

int sa(int a,int b)
{
    return a-b;
}

int seki(int a,int b)
{
    return a*b;
}

int shou(int a,int b)
{
    return a/b;
}

int amari(int a,int b)
{
    return a%b;
}
```

【解説】

1. printf 中の複数の%d には，後に続く 2 つの変数及び関数呼び出しの値を出現順に当てはめて表示される。

2. return はそれに続く式により，関数の戻り値を呼び出し側に戻す。

3. 四則演算（加減乗除）及び除算の余りを求める演算子にはそれぞれ，'+'，'-'，'*'，'/' 及び '%'を用いる。

4. 整数同士の除算の結果は整数となる。 ⇒ 例）2/3 は 0 となる。

5. printf で画面に '%'を表示させるには，%% を用いる。

<u>演習 1</u>

例題 1 を応用して，ex3-1 JIMBO Masato のように，プログラム名に続けて 2 つの文字列を入力したとき，姓：JIMBO；名：Masato と画面に表示するプログラムを作成せよ。ここで，ソースプログラム名は ex3-1.c とする。

演習 2

例題 2 を次のように書き換えよ。ここで，ソースプログラム名は ex3-2.c とする。

1. wa，sa，seki，shou，amari の関数定義を書き換え，int 型の仮引数を 2 つ持ち，戻り値のないものとする。ここに wa の定義を与える。

```
void wa(int a,int b)
{
    printf("%d + %d => %d¥n", a,b,a+b);
}
```

2. wa，sa，seki，shou，amari のプロトタイプ宣言を関数定義に合わせて書き換える。

3. main 関数で wa，sa，seki，shou，amari を呼び出しが記述されていた printf を削除し，それぞれの呼び出しの文で置き換える。ここに wa の呼び出しの文を与える。

```
    wa(x,y);
```