C言語のプログラムの構成

C では関数を基本単位としてプログラムを構成する

単純なプログラムは main 関数のみから成る

```
#include <stdio.h>

int main(void)
{
  文1
  文2
  文3
   ・・・
  return 0;
}
```

#include <stdio.h> などを省略する。

実際のプログラミングでは省略せずにコードして下さい。

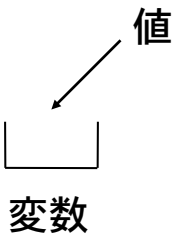
```
int main()
{
  文1
  文2
  文3
   ・・・
}
```


変数

プログラム中で、値を格納するには変数 variable を用いる

変数は、格納する値の型によって、整数型、文字型、などの型 type をもつ

変数を使うには、利用に先立って変数の宣言 declaration をしなければならない



変数とは値（データ）を格納する容器

変数の値はコンピュータのメモリ上に格納される。具体的にメモリのどの場所に格納されるかは言語処理系が自動的に扱うので、プログラマ（特に初級者）が意識する必要はない。

値には、整数値、実数値、文字などの様々な型がある

整数型

整数の値を保持する変数を整数型という

変数宣言には int を用いる

英語で整数は integer

```
int main()
{
  int x;

  x = 1;
  printf("x = %d\n",x);
}
```

← 整数型の変数 x を宣言

← 変数 x に整数値 1 を代入

← 変数 x の値を出力

変数の宣言には、変数の型（整数型の場合 int）に引き続き、変数名を書く。

変数名は原則として自由に付けられる。最初の 1 文字は英字でなければならない。また、C 言語のキーワードは使用できない。

= は、右辺の値を左辺の変数へ代入する代入演算子。

printf による変数の出力

```
printf("x = %d\n", x);
```

書式　　変数名

書式 "x = %d\n" の意味

%d は、変数を整数値として出力することを示す変換指定

変換指定 %d が無ければ printf("x = \n"); は、単に文字列リテラル "x = \n" の出力になる。

変換指定は必ず対応する変数と対になっている必要がある。

```
printf("x = %d\n",x);
```

実行結果

```
/* 変数の出力 */
#include <stdio.h>

main()
{
  int x;

  x = 1;
  printf("x = %d\n",x);
}
```

```
% cc test.c
% ./a.out
% x = 1
%
```

プログラムの出力結果

整数型の変数 x を宣言して、x に整数値 1 を代入。
そして画面に変数 x の値を出力するプログラム。

問題

下のプログラムをコンパイル・実行するとどうなるか?

```
#include <stdio.h>

main()
{
  x = 1;
  printf("x = %d\n",x);
}
```

変数 x が宣言されていないのでコンパイルエラー。

```
#include <stdio.h>

main()
{
  int x;
  printf("x = %d\n",x);
}
```

宣言しただけの変数には、でたらめな値(ゴミ)が格納されているかもしれないので無意味な値が出力される。

処理系によっては自動的に0で初期化される

変数の入力と出力

キーボードから整数値を入力し、その値を出力するプログラム

```
main()
{
   int x;

   printf("整数値を入力せよ:");
   scanf("%d", &x);
   printf("入力した整数値は %d です\n",x);
}
```

キーボードから変数の値を入力するにはライブラリ関数 scanf() を用いる

関数 scanf()

キーボードから値を整数値として読み込み、変数 x に格納する

```
scanf("%d", &x);
```

書式　　&変数名

scanf()による入力では、
変数名の前に & をつける!

書式 "%d" の意味

%d はキーボードから入力する値を整数値として取り扱うことを指定する変換指定。

&x の意味

& をアドレス演算子という。変数名の前に付けると、その変数が格納されているメモリ上の場所を表す。

scanf の書式の変換指定と&変数は必ず対になっていなければならない

```
scanf("%d", &x);
```

printf の書式と異なり、scanf の書式には変換指定以外の文字は書かない

実行結果

```
#include <stdio.h>
main()
{
   int x;

   printf("整数値を入力せよ:");
   scanf("%d", &x);
   printf("入力した整数値は %d です\n",x);
}
```

```
% cc test2.c
% ./a.out
% 整数値を入力せよ：123
% 入力した整数値は 123 です
%
```

このプログラムは整数値を取り扱うことを前提としているので、整数値以外の値、たとえば文字を入力するとおかしな結果が表示される。

茶色はプログラムの出力結果

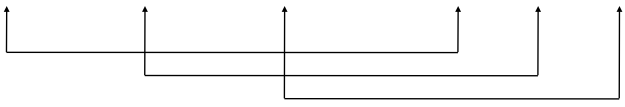
複数の変数を使う

複数の変数を宣言するには、型宣言の後に変数名をコンマで区切って書く

```
int x, y, z;
```

printf を用いて複数の変数を出力するには、書式に複数の変換指定を指定する。書式の後、コンマで変数を区切って記述。

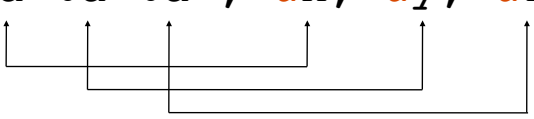
```
printf("x=%d,y=%d,z=%d\n", x, y, z);
```



変換指定と変数は必ず対を形成する。

scanf を用いて複数の変数を入力するには、書式に複数の変換指定を指定。書式のあと、アドレス演算子と変数名をコンマで区切って記述。

```
scanf("%d %d %d", &x, &y, &z);
```



キーボードから入力する値は空白（スペース）で区切って入力する。

たし算の計算

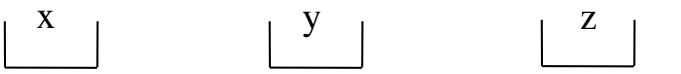
問題:

キーボードから2つの整数値を読み込んで、その和を計算して出力するプログラムを作れ。

考え方:

読み込んだ2つの整数値を格納するために、整数型の変数2つが必要

たし算の結果を格納するためにもう一つの変数が必要



入力値1　　入力値2　　たし算の結果

変数 x と y は scanf で入力する。たし算の結果 z は printf で出力する。

たし算のプログラム

```
#include <stdio.h>

main()
{
  int x, y, z;

  printf("整数値を2つ入力せよ:");
  scanf("%d %d", &x, &y);
  z = x + y;
  printf("%dと%dの和は%dである\n", x, y, z);
}
```

+ は算術演算子の一つで和を計算する。算術演算子は他には
-, *, /, % がある。それぞれ、差、積、商、余りを計算する。

たし算のプログラムその2

```
#include <stdio.h>

main()
{
  int x, y;

  printf("整数値を2つ入力せよ:");
  scanf("%d %d", &x, &y);
  printf("%dと%dの和は%dである\n", x, y, x+y);
}
```

変数を 3 つ使わなくても同じことは可能。

整数値同士の算術演算

C では、整数型 int 同士の算術演算 +, -, *, /, % の演算結果はすべて整数型になる

```
int + int
int - int
int * int
int / int
int % int
```

演算結果はすべて int となる

```
1 + 2  -> 3
5 - 10 -> -5
4 * 6  -> 24
35 / 4 -> 8
35 % 4 -> 3
```

整数同士の割り算には注意が必要！

整数型の範囲

整数型は通常 2 バイト（16ビット）で表される（言語処理系により異なる）。つまり 2 の 16乗 ＝ 65536 通りの表現しか出来ない。正と負の整数を考えると、-32768 ... 32767 の範囲の整数しか取り扱いが出来ないことになる。

取り扱い可能な整数値の最大最小値は、limits.h というヘッダファイルで、`INT_MAX, INT_MIN` として記載されている。

```
#include <stdio.h>
#include <limits.h>
main()
{
  printf("整数値の最大値は %d\n", INT_MAX);
  printf("整数値の最小値は %d\n", INT_MIN);
}
```


実数の取り扱い

整数型の変数は整数値しか格納できない。実数を扱う型に浮動小数点型 (floating type) がある。浮動小数点型は 32ビットで表現され 10進数での有効桁数は 6 桁。変数宣言には `float` を用いる。

同じく実数を取り扱う型に、倍精度浮動型（単に倍精度型ともいう）がある。倍精度浮動型は 64ビットで表現され有効桁数は 10 桁。通常はこちらの倍精度浮動型を用いて実数を取り扱う。変数宣言には `double` を用いる。

倍数度型の変数の宣言

```
double x, y;
```
変数 x と y を倍精度として宣言

```
x = 3.1415;
```
変数 x に実数 3.1415 を代入

```
y = 5.0;
```
変数 y に実数 5 を代入

↑

値が実数であることを明示する場合は 5.0 という具合に書く

実数の出力と入力

printf を用いて倍精度型の変数の値を出力するには、変換指定 `%f` を用いる。

scanf を用いて値を倍精度型としてキーボードから入力するには変換指定 `%lf` を用いる。

```
main()
{
  double x;

  scanf("%lf", &x);
  printf("%f\n", x);
}
```


実数と整数との演算結果

倍精度型同士の算術演算結果は、倍精度型になる。

整数型と倍精度型の算術演算結果は、倍精度型になる。

整数型で宣言された変数に倍精度型の値を代入すると、少数部分が切り捨てられて代入される。

```
double x;
int y;

x = 3.1415;
y = 10;
printf("%f\n", x+y);
```

x+y の型は double になるので
変換指定は %f でなければならない

```
double x;
int seisu;

x = 3.1415;
seisu = x;
printf("%d\n", seisu);
```

算術演算子の優先度

算術演算子を複数組み合わせる場合、各演算子は数学と同じ優先順位で実行される

かけ算と割り算 (*, /) は、たし算と引き算 (+, –) よりも優先度が高い

a + b*c とは、a に、b と c の積を足すことを意味する

優先度を変えるには数学と同様、カッコ () を用いる。

(a+b)*c とは、a と b の和に c を掛けることを意味する。

以上の優先度は、整数・実数を問わず成り立つ。

型変換

明示的な型変換

int 型の値を double 型へ変換をしたい場合がある。

```
int a, b;
double c;

a = 4;
b = 22;
c = b/a;
printf("%f\n", c);

c = (double)b/a;

printf("%f\n", c);
```

c = b/a; ← a, b 共に int 型なので、b/a の値は int 5
c は double型変数なので c の値は 5.0

c = (double)b/a; ← キャスト演算子 ( ) により b の値を double 型へ変換。
22.0/4 の結果は 5.5。c の値は 5.5。

明示的な型変換のことをキャスト cast と言い、キャスト演算子を用いる

(型) 式　　式の値を型としての値に変換

算術関数

倍精度 double 型の実数に対する標準的な算術関数

標準ヘッダファイル math.h で定義されている関数

sin(x), cos(x), tan(x) : 三角関数 与える x の単位はラジアン

sqrt(x) : 平方根 負の値 x を与えると実行時エラー

log(x) : 自然対数 負の値 x を与えると実行時エラー

exp(x) : 指数関数 e^x

これらの関数を使用するに当たっては math.h をインクルードする必要がある

```
#include <math.h>
```


算術関数ライブラリのリンク

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

int main()
{
  double x, y;

  printf("実数を入力せよ:");
  scanf("%lf", &x);
  y = sin(x);
  printf("sin(%f) = %f\n", x, y);
}
```

算術関数のライブラリをリンクするため、
コンパイルオプション
-lm を付ける

```
% cc test.c -lm
% ./a.out
% 実数を入力せよ：3.1415
% sin(3.1415) = 0.0000
%
```

これはプログラムの出力結果

コンパイラの他のオプション

-o ファイル名 ：実行形式ファイルを a.out ではなく指定したファイル名で作成

```
% cc test.c -lm -o test
```

よく使う変換指定

`%d`　int 型を10進数表記に変換

```
printf("%5d\n",x);
```
int 型の変数 x の値を右詰め 5 桁で表示（10進法）

```
scanf("%d", &x);
```
キーボードから入力された値を整数値として変数 x に格納

`%f`　double 型を 10進数表記に変換（scanf では使用不可）

```
printf("%f\n",x);
```
変数 x の値を表示

```
printf("%10.5f\n",x);
```
変数 x の値を右詰め 10桁、小数点以下 5桁で表示

`%lf`　double 型を 10進数表記に変換（printfの書式変換としても使用可能）

```
scanf("%lf", &x);
```
キーボードから入力された値を double型として変数 xに格納

`%e`　実数を指数部付き10進数表記に変換

```
printf("%10.4e\n", x);
```
変数 x の値を右詰め 10桁、精度 4桁で表示

変換指定は他にもたくさんある。詳細は C 言語の教科書を参照

C言語のキーワード

C 言語で標準的に使い方が決まっている言葉をキーワードという。
キーワードは変数名として使用することが出来ない。

キーワード一覧

```
auto, double, int, struct, break, else, long
switch, case, enum, register, typedef, char
extern, return, union, const, float, short
unsigned, continue, for, signed, void, default
goto, sizeof, volatine, do, if, static, while
```

本講義ではこの色のキーワードを習得する

変数名について：キーワード以外の自由な名前を付けることが可能。
プログラム中で分かりやすい名前を付けるのが望ましい。

```
int input, output;
double ondo_C, ondo_F;
```

課題 1

キーボードから整数値を 1つ読み込み、そのまま出力するプログラムを作れ。

動作例:

```
% ./a.out
% 整数値を 1 つ入力せよ：123
% 入力した整数値は 123 です
%
```

茶色はプログラムの出力結果

課題 2

キーボードから 2つの整数値を読み込み、加減乗除と余りを計算する
プログラムを作れ。

動作例:

```
% ./a.out
% 整数値を 2 つ入力せよ：24 5
% 入力した整数値は 24 と 5 です
% 24 + 5 = 29
% 24 - 5 = -19
% 24 * 5 = 120
% 24 / 5 = 4
% 24 % 4 = 4
%
```

空白文字(スペース)で区切って入力

茶色はプログラムの出力結果

課題 3

キーボードから気温（華氏）を入力し、摂氏に変換するプログラムを作れ。

摂氏 C と華氏 F の関係は以下の通り。

$$F = 1.8*C + 32$$

動作例：

```
% ./a.out
% 気温を華氏で入力せよ：74.5
% 華氏 74.5 度は摂氏 23.6 度ですがな
%
```

茶色はプログラムの出力結果

摂氏：スウェーデンのセルシウス(Celsius, Anders)が水の氷点を 0℃、沸点を100℃と定めた温度の単位。ほとんどの国で用いられる温度の単位系。

華氏：ドイツのファーレンハイト(Fahrenheit, Gabriel Daniel)が、氷と食塩の混合物の温度を0F、人間の体温を96Fと定めた温度の単位。一部の国はいまだにこの単位系を使い続けている。

あるウェブサイト www.cnn.com では世界各地の気温（摂氏・華氏）が得られる。

課題 4

1） 計算機システムの C 言語処理系で取り扱い可能な整数の最大・最小値を出力するプログラムを作れ。

整数の最大・最小値はヘッダファイル limits.h の中で、
`INT_MAX`, `INT_MIN` として記載されている。

2） 最大整数値に整数値 1 を足した結果はどうなるか確認せよ。
最小整数値から整数値 1 を引いた結果はどうなるか確認せよ。

3） 浮動小数点型 `float` と倍精度型 `double` が取り扱い可能な最大値を調べよ。それぞれの最大値はヘッダファイル float.h の中で、`FLT_MAX`, `DBL_MAX`で定義されている。この際、printf の書式の変換指定で `%f` の代わりに `%e` を使用せよ。