1. まず使ってみよう

プログラムを保存する際の注意

- C 言語のプログラムには、xxx.c のように .c で終わる名前を付ける。
- xxx の部分は半角英数字で 8 文字以内が安全。
- .c で終わるファイルのみを保管すればいい。

名前を出力するプログラム例

次のプログラムをそのまま打ち込んで (左端の行番号は打たなくてよい)、実行してみよう (名前の箇所は自分の名前を書く)。ただし、 \ は ¥ のキーを打つこと。

```
1  #include <stdio.h>
2  main()
3  {
4     printf("My name is Taro Yamada\n");
5  }
```

数を 3 倍するプログラム例

次のプログラムをそのまま打ち込んで実行してみよう。

```
1  /* simple calc */
2  #include <stdio.h>
3  main()
4  {
5     int  n, kekka;
6
7     n = 11;
8     kekka = n * 3;
9     printf("%d * 3 = %d\n", n, kekka);
10 }
```

プログラムの説明

- 注釈 (1 行目)
  /* から */まではコメント (注釈) と呼んで、プログラムの実行には関係がない。プログラムを読む人の便宜のために記述する。または、プログラム作成者が後のために自分用に記すものである。/* から */の間にはどんな文字でも入れることが可能である。
- インクルードファイル (2 行目)
  これは現在の知識では説明し難いので、さしあたりおまじないと思って下さい。秋学期に説明します。

- main 関数 (3 行目)
  C 言語では 1 つ以上の関数の集まりでプログラムが構成される。最初に実行される関数は main 関数で、プログラムには必ず main という関数が必要である。関数の書き方は、

```
main()
{
        この関数内で使用する変数宣言 1 ;
        この関数内で使用する変数宣言 2 ;
        .
        .
        この関数内で使用する変数宣言 n ;

        文 1 ;
        文 2 ;
        .
        .
        文 m ;
}
```

  のように書く。上の例では関数名が main で関数 main の中で使用する変数の宣言をすべて書いてから、その下に処理する命令を書く。文 1, 文 2, ..., 文 m は順次実行され、文 m の実行が終了すると関数 main の実行も終了したことになる。

- 変数宣言 (5 行目)
  `int n, kekka;`
  これは、`n` と `kekka` という 2 つの整数型の変数を宣言している。
  C 言語では変数を使う前に必ず変数の宣言を行なう。int というデータ型は整数を表している。すなわち、変数 n には 123、0、− 3 といった値をセットすることが出来るが、4.56 といった値をセットすることはできない。ただし、整数といっても int のデータ型は −2147483648 から 2147483647 までの値しかセットすることができない。この範囲外の値をセットすると変数の値は不定になる。
  変数名を並置するときは間に , (comma) を入れる。(宣言) 文の終わりには ; (semicolon) を付ける。
  （注） **変数名**は、英数字の並びで最初の文字は英字でなければならない。また、下線 _ は英字に含まれ、大文字と小文字は区別される。`int`, `long`, `if`, `for` 等は変数名として使えない。

- 代入文 (式) と数式 (7,8 行目)
  `n = 11;`
  これは、変数 `n` に 11 を代入する文である。
  `kekka = n * 3;`
  は、変数 `n` の値に 3 を掛けてその結果を変数 `kekka` に代入する文である。

数式は、+, −, ∗, /, % の 5 つの演算子と ( ) を組み合わせて表現する。
`m % n` は、`m` を `n` で割った余りが求まる。
数式 $b^2 - 4ac$ はプログラム中では `b*b-4*a*c` と表すことになる。

- 関数 printf (9 行目)
関数 printf は画面に文字を出力する関数である。`"%d * 3 = %d\n"`はどのように画面に出力するかのフォーマットを記述する。この部分は”で両端を挟んで書く。
`printf("%d * 3 = %d\n", n, kekka);`
の例では、最初の`%d` の位置に int 型変数 `n` の値が、次の`%d` の位置に int 型変数 `kekka` の値が印刷実行時に入る。その他の文字はそのまま出力される。
`"`は二重引用符 (double quote) で、一重引用符’(single quote) 2 つではない。
`\n` は改行 (newline) を表す C 記号であるが、\ は実習機では ¥となる。(以後、プリント中で ‘\’ は ‘¥’ のことなので注意すること!)

フォーマット記述子	説明
`%d`	データ型 int の値を表示。int の値または変数名、または int 型になる式はフォーマット記述子の次に書く。
`%ld`	データ型 long int の値を表示。long int の値または変数名、または long int 型になる式はフォーマット記述子の次に書く。
`%f`	データ型 float の値を表示。float の値または変数名、または float 型になる式はフォーマット記述子の次に書く。
`%%`	%を表示する。
`\n`	改行を行なう。
`\\`	\ の文字を表示する。
`\"`	"の文字を表示する。
`\t`	tab 分だけスペースを空ける。
その他	その文字をそのまま表示する。

例えば

```
printf("a b c");
printf("abc \n");
printf(" abc %d aa %d %d\n",1,n,n*kekka7+1);
```

のように書く。

[練習 1] 8 行目の計算式をいろいろ変えて実行してみよう。
(1) $n + 3 * n$
(2) $(n + 3) * n$
(3) $n/2$
(4) $n\%2$

算術演算子の優先順位

C 言語における算術式では、日頃行なっている数式計算同様、乗除の計算は加減の計算より優先され、同じ優先度の場合には左から実行される。優先度を変えたい (あるいは明示したい) 時は、カッコ ( ) を使う。C

言語では算術式に使えるカッコは 1 種類しかなく、カッコが重なる時は ( ) を入れ子にして表す。

整数の読み込み

`n = 11;`
とすると、11 以外の時は毎回プログラムを直さなくてはならない。int 型変数 `n` に数値をキーボードから入力できるようにするには、
`scanf("%d", &n);`
という文に変更する。ただし、この場合キーボードから読み込むことが可能なのは整数だけであり、その値は int 型のデータの範囲と同じである。このときに英字、記号などを入力すると不定な値が int 型変数 n に代入される。また複数のデータを読み込むには先の命令を複数書くか、
`scanf("%d%d%d",&m,&n,&p);`
のように並べて書く。この命令の意味は難しいので当分おまじないと思って使って下さい。

改良できる点

- `printf("%d * 3 = %d\n", n, kekka);`
  の代わりに式を直接書き、
  `printf("%d * 3 = %d\n", n, n * 3);`
  としてもよい。すると、変数 `kekka` を使う必要がなくなる。

[練習 2] 二つの整数 m, n を読み込み、その和、差、積、商、および (m を n で割った) 余りを求めるプログラムを作りなさい。

[練習 3] 整数 n を読み込み、1 から n までの和を求めるプログラムを作りなさい。
$(1 + 2 + ... + n = \frac{n(n+1)}{2}$ を使うとよい)