変数 (variable) と演算子 (operator) (1)

変数 (variable)

変数とは，計算の途中経過や結果の値を保存しておく「場所」のことである。変数名とはその場所につけられた変数の名前で，プログラム中で変数名を使うと，その場所に保存されている値を参照できる。

変数名の命名ルール

1) 変数名には，半角のアルファベットと数字，アンダースコア _ を使うことができる。
2) 変数名は数字で始めることはできない。
3) C 言語で予約されている単語等は使えない。

（例）

○ a, xyz, tate, yoko, suu1, w_x_, ichi2san

× 3x, san+yon, main （なぜ×なのか，上記1)〜3)で当てはまる項目を考えなさい）

変数名とデータ型

C 言語で扱うことのできるデータには様々な型があるが，変数は格納するデータ型に合わせて確保（宣言）する必要がある。 C 言語で扱えるデータ型の一部を以下に示す。

タイプ	型名	大きさ (bit)	意味	範囲	printf の指示
整数型	short	16	単長整数	-32,768 〜 +32,767	%d
	int	32	整数 （大きさは処理系依存）	-2,147,483,648 〜 +2,147,483,647	%d
	long	32	倍長整数	-2,147,483,648 〜 +2,147,483,647	%d
浮動小数点型	float	32	単精度浮動小数点数	有効桁数7桁の実数	%f
	double	64	倍精度浮動小数点数	有効桁数16桁の実数	%f
文字型	char	8	文字	8bit で表せる文字	%c

変数の宣言

C 言語では，変数は宣言してから使用する。宣言の形式は以下の通り。

変数の型名　変数名（, 変数名...）;

（例）

```
int i;                      /* 整数型の変数 i を宣言する */
double tate, yoko;          /* 倍精度浮動小数点型の変数 tate と yoko を宣言する */
```

変数への代入

変数に値を代入するには = 記号を使う。変数の型に合わない値は正しく代入できない（代入する変数の型に強制的に変換されるか，文法エラーになる）。
同じ変数に代入すると，前の値は上書きされる。
宣言しただけで代入されない変数の値は「不定」とる。

```
変数 = 値;
（例）
i = 5;                      /* 変数 i に整数値 5 を代入する */
yoko = 10.3;                /* 変数 yoko に実数値 10.3 を代入する */
```

宣言と同時に代入をすることもできる。

```
（例）
int i = 5;            /* 整数型の変数 i を宣言し，整数値 5 を代入する */
double yoko = 10.3;   /* 倍精度浮動小数点型の変数 yoko を宣言し，実数値 10.3 を代入 */
```

変数の値の表示（出力）

変数の値を表示するには，**フォーマット指示子**を含んだ printf 関数を使う。

```
（例）
int i = 5;
printf( “%d¥n”, i );       /* 変数 i の値を 10進数の整数表示形式で出力する */


double yoko = 10.3;
printf( “%f¥n”, yoko );    /* 変数 yoko の値を 浮動小数点数表示形式で出力する */
```

演算子 (operator)

演算子とは計算に使う記号のことである。 C 言語での加減乗除は， +, -, *, / を使う。 % 演算子は剰余（割った余り）を計算する演算子である。また，計算の順序を変えるために括弧 () を使うことができる。

（練習）長方形の面積を計算する以下のプログラム rectangle.c を入力し，実行してみなさい。（行番号は入力しないこと）

```
 1  /* rectangle.c */
 2
 3  #include <stdio.h>
 4
 5  int main()
 6  {
 7          int tate, yoko;
 8          int menseki;
 9
10          tate = 5;
11          yoko = 3;
12          menseki = tate * yoko;
13
14          printf( "tate %d * yoko %d = ", tate, yoko );
```

```
15          printf( "%d¥n", menseki );
16
17          return 0;
18  }
```

（練習2）長さが実数の値でも正しく計算できるように倍長浮動小数点型に変更した以下のプログラム rectangle2.c を作成し，実行結果を確かめなさい〔ヒント： rectangle.c とはアンダーライン部のみが異なるので，「名前をつけてで保存」してからアンダーライン部を書き直せば良い。〕

```
 1  /* rectangle2.c */
 2
 3  #include <stdio.h>
 4
 5  int main()
 6  {
 7          double tate, yoko;
 8          double menseki;
 9
10          tate = 5.5;
11          yoko = 3.0;
12          menseki = tate * yoko;
13
14          printf( "tate %f * yoko %f = ", tate, yoko );
15          printf( "%f¥n", menseki );
16
17          return 0;
18  }
```

（練習3）底辺 teihen と高さ takasa の値を代入して三角形の面積を計算するプログラム triangle.c を作成し，実行結果を確かめなさい。

（練習4）上底 joutei と下底 katei と高さ takasa の値を代入して台形の面積を計算するプログラム trapezoid.c を作成し，実行結果を確かめなさい。

（*練習5）分数 minute を代入して，時間と分に分けて表示するプログラム h-m.c を作成し，実行結果を確かめなさい。（ヒント， 「90 分は 1 時間と 30 分」のように表示する。分数を 60 で割った整数部が時間，余りが分になる。余りを求める演算子は %。）

整数型・浮動小数点数型の混在と演算

除算の対象（割る数と割られる数）の型が整数のみから構成されている場合，演算結果は整数になる（小数点以下切り捨て）。除算に限らず，演算結果が実数になることが予想される場合，変数であれば浮動小数点型，値そのものであれば小数点付きの表現（ 9 ではなく， 9.0 など）を使う。または，型変換（キャスト）を利用する。型変換とは，変数や式の直前に変換したい型（ int や double など）を括弧 () 付きで指定する方法である。たとえば，整数変数 seisuu を浮動小数点型で計算したい場合は (double)seisuu と表記する。

（練習６）以下のプログラム vartest.c を入力，コンパイルし，実行結果を書きなさい。また，なぜそのような結果になったのか，説明しなさい。

```
1     /* vartest.c */
2
3     #include <stdio.h>
4
5     int main()
6     {
7        double jissuu1, jissuu2;
8
9        jissuu1 = 9 / 2;
10       jissuu2 = 9.0 / 2.0;
11
12       printf( "jissuu1 = %f¥n", jissuu1 );
13       printf( "jissuu2 = %f¥n", jissuu2 );
14
15       return 0;
16    }
```

値の表示 - 関数 printf()

printf() 関数は，表示する値を整形（小数点第何位まで等）して出力することができる。

printf の構文：

printf(フォーマット指定文字列 [, 式１ [,式2 [...]]]);

例）

printf(“i = %d\n”, i); // 変数 i の値が 5 なら， i = 5 と表示される

フォーマット指定文字列のルール（抜粋）：

%[フラグ][フィールド幅][.精度]変換文字 // []内は省略可能

変換文字	動作	例（i=100，x=314.15とする）	出力
d，l	10進数で出力する	printf("i=:%d:\n", i);	i=:100:
o	8進数で出力する	printf("i=:%o:\n", i);	i=:144:
x，X	16進数で出力する	printf("i=:%x:\n", i);	i=:64:
f	浮動小数点として出力する	printf("x=:%f:\n", x);	x=:314.150000:
e，E	指数形式で出力する	printf("x=:%e:\n", x);	x=:3.141500e+02:
c	文字として出力する （文字自体は'で括る）	printf("x=:%c:\n", 'd'); printf("x=:%c:\n", i);	x=:d:（dの文字コードは100）
s	文字列として出力する （文字列自体は"で括る）	printf(":OHA%sSAN!:\n", "YOU");	:OHAYOUSAN!:
%	文字%を出力する	printf(":%%n:\n");	:%n:

※例中のコロン記号:は，表示範囲がわかるようにするためにつけいているので，通常はなくても良い。

フィールド幅	動作	例（x=314.15とする）	出力
数値	最低の文字幅を指定する。右詰で表示される。	printf("x=:%10f:\n", x); 小数点も1文字に数える	x=:314.150000:

精度	動作	例（x=314.15とする）	出力
数値	小数点以下の文字幅を指定する。	printf("x=:%10.1f:\n", x); この例では整数部8桁.小数部1桁	x=: 314.1:

フラグ	動作	例（x=314.15とする）	出力
-	左詰で表示する	printf("x=:%-10.1f:\n", x);	x=:314.1 :
+	+/-符号をつけて表示する	printf("x=:%+10.1f:\n", x);	x=: +314.1:
#	8進数に0，16進数に0xを先頭に付加する	printf("i=:%#x:\n", i);	i=:0x64:

参考：http://www.mm2d.net/c/c-01.shtml等

数値計算では，最小文字幅と小数点以下の指定を組み合わせて使うことが多い。

例）文字幅 10 桁, 小数点以下 3 桁で浮動小数点数を表示させる（4位以下四捨五入） `%10.3f`
文字幅には小数点も含むので，整数部分は符号も含めて 6 桁になる。
ただし整数部が 6 桁に収まらない場合は，自動的に伸張される。

（練習）以下のプログラム， `printftest.c` の実行結果は，小数点がそろわず，いささか見にくい。小数点の表示位置をそろえ，小数点以下二位まで表示できるように，文字幅15桁，小数点以下2桁のフォーマット文字列に変更したプログラム `printftest2.c` を作成し，コンパイル，実行してみなさい。

```
 1       /* printftest.c */
 2
 3       #include <stdio.h>
 4
 5       int main()
 6       {
 7         double value = -0.2;
 8
 9         printf( "value = %f¥n", value );
10         value = value + 123456789.56789;
11         printf( "value = %f¥n", value );
12
13         return 0;
14       }
```

（printftest.c の実行結果）

```
value = -0.200000
value = 123456789.367890
```

（printftest2.c の実行結果）

```
value =           -0.20
value =    123456789.37
```

（練習７）以下の `pftest1.c` は `printf()` 関数のフォーマット指定と出力の関係を確認するためのプログラムである。入力，コンパイル，実行して，実行結果を確認しなさい。

```
 1   /* pftest1.c */
 2
 3   #include <stdio.h>
 4
 5   int main()
 6   {
 7     int i = 100;
 8     double x = 314.156;
 9
10     printf( "i=:%d:\n", i );
11     printf( "i=:%o:\n", i );
12     printf( "i=:%x:\n", i );
13
14     printf( "x=:%f:\n", x );
15     printf( "x=:%e:\n", x );
16
17     printf( ":OHA%sSAN!:\n", "YOU" );
18
19     printf( "x=:%c:\n", 'd' );
20     printf( "x=:%c:\n", i );
21
22     printf( "i=:%#x:\n", i );
23   }
```

（練習８）上記 pftest1.c のフォーマット指定 % を %10.2 と変更したプログラム pftest2.c を作成し，コンパイル，実行して結果を確認しなさい。ただし， 19,20,22 行目は変更しないこと。
10.2 という指定でそれぞれ出力がどのように変化するか。

（*追加練習）フォーマット指定フラグ +, - を使って浮動小数点数の表示がどう変化するか，プログラム pftest3.c を作成して確かめなさい。