基礎プログラミング演習 II 教材　(#16)

■ 文法的落ち穂拾い：変数の型と精度

C 言語には必要な精度を満たすために多くの型が用意されています。
★教科書 p.22 表 2.1（整数型 short, long, int）
　教科書 p.116 表 7.1 参照（実数型 float, double）

実験：精度（有効桁数）が型によって異なることを確認しましょう。右のプログラムに精度を超えると想像できる値を指定して結果を見てください。

```
short s;
int i;
float f;
double d;

s=100;      i=200;
f=300.0f;   d=400.0;

printf("short  %d\n",s);
printf("int    %d\n",i);
printf("float  %f\n",f);
printf("double %f\n",d);
```

（型はこれ以外にも unsigned や long double などがあります。調べてみると良いでしょう。）

□ 定数における型の明記

123.456 は実数でも double 型とみなされます。float 型と明記したい場合は 123.456f と最後に f をつけて表記します。（サンプルプログラムの `f = 300.0f;` を参照。）
実数はまた `1.2e-5;` というように表記できます。（ 1.2×10^{-5} つまり `0.000012` ）
教科書 p.117 文法 7.1 も参照。
（興味のある受講生は教科書 p.24 文法 2.1、表 2.2 も参照。整数定数の 8, 16 進表記法がある。）

□ データの型に合わせた変換文字列と桁数指定

整数は %d, 実数は %f あるいは %e が利用できます。また、%10d のように、% と変換文字の間に桁数の指定などができます。以下に代表的な変換文字列を示します。

	意味	使用例	その結果
%d	整数を表示	printf("[%d]\n",10);	[10]　桁数不定
		printf("[%5d]\n",10);	[　 10]　5 桁で表示。不足分は空白。
		printf("[%05d]\n",10);	[00010]　5 桁。不足はゼロで埋める。
%f	実数を表示	printf("[%f]\n",12.345);	[12.345000]　桁数不定
		printf("[%9.5f]\n",12.345);	[12.34500]　小数点含めて全体が 9 桁、小数以下が 5 桁。
%e	実数を表示	printf("[%e]\n",12.345);	[1.234500e+01] 浮動小数点で表示

□ 誤差：if 文による判定

右のプログラムは 100 回ループすると停止するように見えますが、実際には if 文の条件は成立しません。
しかし 0.1 ではなく 1.0 ずつ加算すれば停止します。
これは 0.1 が 2 進での浮動小数点表現では無限小数（割り切れない数）になるために生じる誤差が原因です。
制御文字を %f50.45 などとして printf() すれば確認できます。

```
double d;
d=0.0;
while(1) {
  if(d == 10.0) break;
  printf("%f\n", d);
  d+=0.1;
}
```

このような場合に対処するため、実数では同一性判定は一定の範囲を定めて行うのが安全です。
`double e; e=0.1e-12;` などと非常に小さな値を用意して、
`if((d > 10.0-e)&&(d < 10.0+e)) break;` などとして判定します。

■ 数学関数

□ 数学関数の使用例

C 言語には sin( ) 関数や cos( ) 関数など、数学的な計算を行ってくれる関数がひと揃い用意されています。こういった関数を数学関数と呼んでいます。代表的な関数としては以下のようなものがあります。

sin　正弦を求める	sqrt　平方根を求める
cos　余弦を求める	pow　累乗を求める
tan　正接を求める	log　対数を求める
fabs　絶対値を求める	exp　指数を求める

他の関数、詳しい使い方などについては <<ガイド 8.2>> や参考書を参照すると良いでしょう。

y = sin(r); のようにして使えば y に角度 r の時の正弦を計算してくれます。

ただし sin 等の三角関数に与える引数は、
・ double 型の実数であること
・ 角度は、ラジアン単位（弧度）で指定することに注意して下さい。
sin 関数の戻り値は double 型で返されます。

C 言語における三角関数はどれもラジアン単位（弧度法）です。（度数法で言う 360 度を 2 πラジアンとする）
つまり sin 関数に 60 度に相当する角を渡したければ、
60 / 360 * 2 * 3.141592 ≒ 0.018278
を与えることになります。

□ 数学関数を使うために

数学関数を使うために先頭に以下の一行を書いて下さい。

#include <math.h>

これによって数学関数を定義したヘッダファイル math.h がプリプロセッサによって取り込まれます。書かなかった場合のエラーを一度見ておくと良いでしょう。
Termnal で man 3 sin などとすると、どの関数がどのヘッダを必要とするか分かります。

また、MacOSX では不要ですが、幾つかのコンパイラでは -lm オプションが必要になります。

```
$ cc sample.c -lm
```

これは数学関数のためのライブラリを使う指示で、これがないと下記のように「sin という名前は未定義 (undefined) である」といったエラーが出る場合があります。

```
$ cc –o sample sample.c
/tmp/ccHZhcFU.o: In function `main':
/tmp/ccHZhcFU.o(.text+0x3f): undefined reference to `sin'
collect2: ld returned 1 exit status
$
```

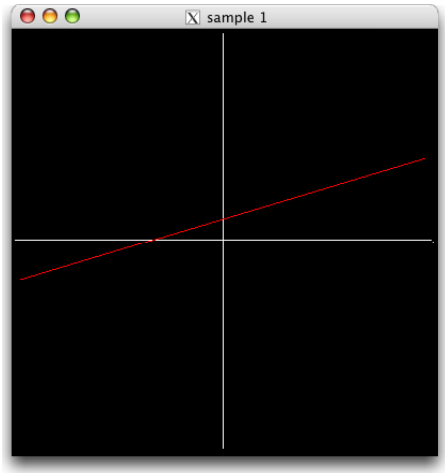
なお egg コマンドでコンパイルするときは –lm は追加しなくてかまいません。理由は egg コマンドが自動的に –lm もつけてくれているからです。
（ egg コマンド実行時のメッセージをよく見ると –lm と付いているのがわかるでしょう。）

☐ 前準備（一次関数のグラフを描く）

以下に $y = ax + b$ $(a=0.3, b=20.0)$ の一次関数グラフを描くプログラムを示します。

押さえて欲しいポイント：
- line( ) 関数で座標軸を描いています。第四引数に PENUP と書くことで始点まで移動し、そこから PENDOWN と書いて与えた座標位置までを線引きします。
- pset( ) 関数で、計算した座標位置に点を打っています。
- 原点をグラフィクスウィンドウの真ん中（200, 200）に置いているので、pset( ) 関数の x, y 座標位置指定にはそれぞれ 200.0 を加算しています。

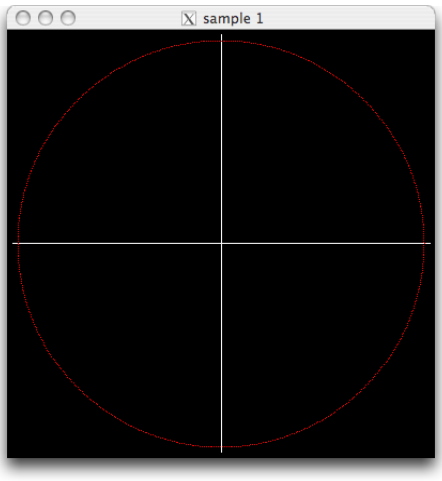
プログラムと実行結果




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

int main() {
  int x, win;
  double y, a, b;

  win=gopen(400,400);  /* 描画ウィンドウを開く */
  winname(win, "sample 1"); /* 名前をつける */
  newpen(win, 1);
  line(win, 5.0, 200.0, PENUP);
  line(win, 395.0, 200.0, PENDOWN);
  line(win, 200.0, 5.0, PENUP);
  line(win, 200.0, 395.0, PENDOWN);

  a=0.3;
  b=20.0;
  newpen(win, 2);
  for(x=-190; x<190; x++) {
    y=a * x + b;
    pset(win, x + 200.0, y + 200.0);
  }

  ggetch(); /* キー入力を待つ */
  gclose(win); /* 描画ウィンドウを閉じる */
  return 0;
}
```

☐ 課題 1.

左図のような円を描いてください。ただし circle 関数を使うのではなく、円周上に多くの点を打って円に見えるようにします。



☐ 課題 2.

右図のような sin カーブを描いてください。