
```
import("glib3.rr");
Glib_math_coordinate=1;
glib_window(-2,-2, 2,2);
glib_line(-2,0,2,0 | color=0x0000ff);
glib_line(0,-2,0,2 | color=0x0000ff);
for (X=-2.0; X< 2.0; X = X+0.1) {
  Y = X^2-1;
  X1 = X+0.1;
  Y1 = X1^2-1;
  glib_line(X,Y, X1,Y1);
}
glib_flush();

end$
```

```
#include <stdio.h>
#include "glib4.h" // かならず書く
/*
講義のページから glib4.c, glib4.h をダウンロードしてこのファイルと同じフォルダに
置いておく.
cc sample4.c glib4.c -lX11 -lm
でコンパイル.
./a.out で実行
*/

int main() {
    /** main は変更しない. 描画の手続きは下記の glib_draw (call back 関数となる)に書く.
    **/
    glib_open();
    glib_loop();
}

/*
glib_line(X1,Y1,X2,Y2 | color=C) は
myglib_line(X1,Y2,X2,Y2,C);
これはこのまま使う.
*/
void myglib_line(double x,double y,double x2, double y2, int color) {
    glib_color(NULL,color);
    glib_line(x,y,x2,y2);
}

/* 描画の手続きは sample3.rr に従いこの中に書く.
   ここを自由に修正. glib_draw の名前を変えてはいけない.
*/
void glib_draw() {
    double X,Y,X1,Y1;
    glib_window(-2,-2, 2,2);
    myglib_line(-2,0,2,0 ,0x0000ff);
    myglib_line(0,-2,0,2 ,0x0000ff);
    for (X=-2.0; X< 2.0; X = X+0.1) {
        Y = X*X-1;
        X1 = X+0.1;
        Y1 = X1*X1-1;
        glib_line(X,Y, X1,Y1);
    }
    glib_flush();
}
```

```
/* sample2.c への書き換え方 */
import("glib3.rr"); //これは不要
Glib_math_coordinate=1; //これは不要

/* circle は C の関数として書き換える.
   超入門 p.29
   sample2.c を参照
*/
def circle(X,Y,R,CC) {
  E=0.1;

  for (T=0; T<=6.28; T = T+E) {
    Px = X+deval(R*cos(T));
    Py = Y+deval(R*sin(T));
    Qx = X+deval(R*cos(T+E));
    Qy = Y+deval(R*sin(T+E));
    glib_line(Px,Py,Qx,Qy | color=CC);
  }
}

/*
   以下の
   関数の外にある命令を順番に sample2.c の glib_draw() 関数に書き込む.
*/
glib_window(-1,-1,1,1);
glib_window(-1,-1,1,1);
CC = 0xff0000;
for (P=0.4; P<0.5; P = P+0.01) {
  circle(0,0,P,CC);
  CC = random()%0x1000000;
}
glib_flush();

end$
```

```
#include <stdio.h> // かならず書く
#include <stdlib.h> // random を使う時.
#include <math.h> // exp, log, cos, sin などを使う時
#include "glib4.h" // かならず書く
/*
circle-sample2.rr を C 言語で書いた.
(超入門の circle.rr を C 言語で書いた例.)
circle-sample2.rr と比較しながら読む.
講義のページから glib4.c, glib4.h をダウンロードしてこのファイルと同じフォルダに
置いておく.
```

```
cc sample2.c glib4.c -lX11 -lm
でコンパイル.
./a.out で実行
```

矢印キーでターミナル(unix shell)に以前に入力したコマンドを呼び出せる.
ls -l コマンドでファイル一覧と作成時間を確認.
emacs sample2.c & 編集.

```
*/
int main() {
    /** main は変更しない. 描画の手続きは下記の glib_draw (call back 関数となる)に書く.
    **/
    glib_open();
    glib_loop();
}
/*
関数宣言は glib_draw での描画の前に.
*/

/*
おまけ
 $X^A$  は mypower(X,A) とする.
*/
double mypower(double x,double a) {
    /*  $x^a = \exp(a \cdot \log(x))$  */
    return( exp(a*log(x)) );
}

/*
glib_line(X1,Y1,X2,Y2 | color=C) は
myglib_line(X1,Y2,X2,Y2,C);
*/
void myglib_line(double x,double y,double x2, double y2, int color) {
    glib_color(NULL,color);
    glib_line(x,y,x2,y2);
}

/*
circle-sample2.rr を書き換え.
*/
void circle(double X,double Y,double R,int Color) {
    double E,T;
    double Px,Py,Qx,Qy;
    E=0.1;
    glib_color(NULL,Color);
    for (T=0; T<=2*3.14; T = T+E) {
        Px = X+R*cos(T);
        Py = Y+R*sin(T);
        Qx = X+R*cos(T+E);
        Qy = Y+R*sin(T+E);
        glib_line(Px,Py,Qx,Qy);
    }
}
```

```
    glib_flush();
}

/* 描画の手続きはこの中に書く．
   ここを自由に修正．
*/
void glib_draw() {
    double y;
    double CC,P;

    glib_clear();
    glib_window(-1,-1,1,1);
    CC = 0xff0000;
    for (P=0.4; P<0.5; P = P+0.01) {
        circle(0,0,P,CC);
        CC = random()%0x1000000;
    }
    glib_flush();
}
```