Ver.1　2012.3.26

Processing プログラミング入門

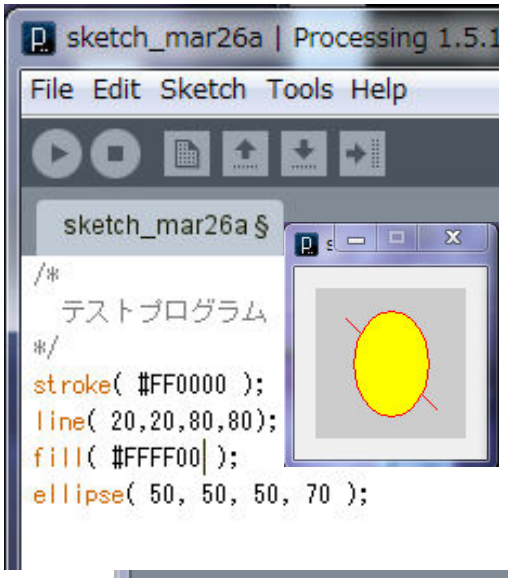
鳥取環境大学　足利裕人

Processing は Java という言語をベースとしたオープンソースのプログラミング言語です。フリーで利用でき，開発環境が整っています。画像やアニメーションを用いた視聴的な表現を得意としていますので，大変とっつきやすいです。

では，デスクトップに作った Processing.exe へのショートカットアイコンをクリックして，プログラムを入力しましょう。

1　線と楕円を描く

```
/*
  テストプログラム
*/
stroke( # FF 0000 );  // 線の色を赤に
line(20, 20, 80, 80);  //(x,y)が(20, 20)-(80, 80)へ線を引く
fill( #FFFF00 );              //次の図形を黄で塗りつぶす
ellipse(50, 50, 50, 70);   //(50,50)を中心に，横縦 50,70 の直径の楕円を描く
```

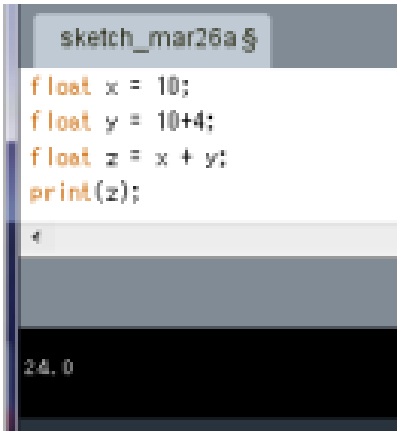


プログラムの各行を文と呼び，文の最後には必ずセミコロン（;）を付けます。
コメントは // を書いたところから行末まで，もしくは，/* から */ までの間に書くことができます。
これを プログラムソースコードまたはスケッチ と呼びます．

2　変数と和の計算

変数には int（整数型），float（実数型），color（色型）などがあります．変数を利用する際には，
型名　変数名，変数名，・・・；
のように宣言する必要があります。宣言と同時に値を代入（初期値を設定）することも可能です．変数への値の格納には代入演算子 = を用います．では，次のプログラムを入力してみましょう。

```
float x = 10;          //変数 x を実数型に定義し，10 を代入する
float y = 10+4;        //変数 y を実数型に定義し，10+4 を代入する
float z = x + y;       //変数 z を実数型に定義し，x+y の値を代入する
print(z);              //z の値を画面下に表示する
```

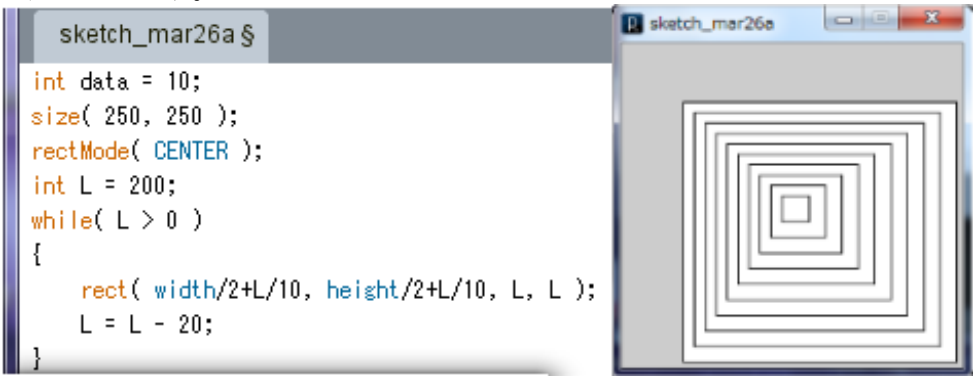


実行すると下の欄に 24 が表示されます。

3　繰り返し図形を描く

以下を実行すると右図の図形が描かれます。手順の流れを確認しながら理解しましょう。

```
int data = 10;
//data を整数型に定義
size( 250, 250 )
```



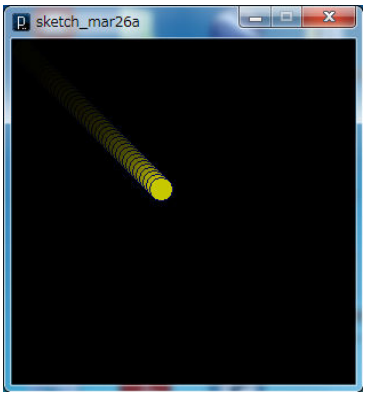
```
//枠の大きさ;
rectMode( CENTER );    //四角の中心位置を指定するモード
int L = 200;
while( L > 0 )                          //L の値が正の間以下の｛｝内を繰り返せ
{
     rect( width/2, height/2, L, L );   //width,height は画面の横，縦のサイズを収めたシステム変数
     L = L - 20;                        //L を 20 ずつ減らす
}
```

4　ボールを動かす

動いていくボールを表すには，１つ前に描いたボールを消して，新たに少し離れた位置にボールを描くことを繰り返します。Processing には次のような連続モードが用意されています.

```
void setup(){   初期化コード   }        //最初に一度だけ実行させたいコードを書く
void draw(){   画面描画コード   }        //繰り返し呼び出されるコードを書く
```

では，アニメーションのプログラムを描いてみましょう。画面左上から右下へ向かって流れ星のようにボールが移動します。数値をいろいろ変えて，動きの変化を確認しましょう。




```
int r = 10;
float x0 = r, y0 = r;    //float：実数を定義　座標初期値
float v = 30;            //等速度値
float t = 0;             //時刻 t を 0 に
float x = x0, y = y0;    //(x,y)を初期値に
void setup()
{
   size(300,300);
   colorMode(RGB,255,255,255,100);     //RGB モードに, 各色の最大値 255, 不透明度 100
   ellipseMode(RADIUS);                //中心の(x,y)と縦・横半径指定モード
   background(0,0,0);                  //黒
   frameRate(30);                      //画面の切り替え時間(1/30s)を設定
}
void draw()                            //メインの繰り返し関数
{
  stroke( 0, 0, 0, 10 );               //輪郭線の不透明度が 10
  fill( 0, 0, 0, 10 );                 //図形の中を塗りつぶす。不透明度が 10
  rect( 0, 0, width, height );         //四角を描く
  x = x0 + v* t;                       //座標値(等速度運動)
  y = y0 + v* t;
  stroke( 0, 0, 100, 100 );
  fill( 0, 200, 255, 100 );
  ellipse( x, y, 10, 10 );             //楕円を描く
  t = t + 0.1;                         //時間の刻みを 0.1 に
}
```

5　音を鳴らす

Processing で音を扱うためには，標準でついてくる Minim ライブラリを利用します。

下のスケッチは，440Hz の正弦波が作る音を発生するものです.

```
import ddf.minim.*;                    //Minim ライブラリのインポート
import ddf.minim.signals.*;
Minim minim;                           //Minim 本体のオブジェクト変数
```

```
AudioOutput out;                          //音声を出力する
SineWave sine;                            //sin 波を発生する
void setup()
{
  minim = new Minim(this);                 //Minim の本体を生成
  out = minim.getLineOut(Minim.STEREO);    //ステレオのライン出力に音声を出力するオブジェクト生成
  sine = new SineWave(440, 0.5, out.sampleRate());  //周波数，振幅，サンプルレートでサイン波を発振するオブジェクト生成
  out.addSignal(sine);                     //音声出力
}
void draw(){}
void stop()                                //Minim の停止
{
  out.close();
  minim.stop();
  super.stop();                            //他のプログラム停止
}
```

周波数を倍にすると 1 オクターブ上，半分にすると 1 オクターブ下になるので， sine = new SineWave(440, 0.5, out.sampleRate()); の 440 の部分を 880 や 220 に変えると，1 オクターブ上と下のラの音が聞こえるはずです.

〇正弦波を合成して，いろんな楽器の音を作る

楽器によって音色が異なるのは，複数の周波数の正弦波を含む割合が異なるからです。複数の高さのラの音を同時に鳴らせてみましょう。どんな楽器に似ていますか。

Processing で複数の周波数の正弦波を合成して出力するには次のようにします。これらを上のスケッチのどこに配置したらよいか考え，実行してみましょう。なお，SineWave メソッドの 2 番目のパラメータは音の強さを指示するもので 0.0 から 1.0 の値を設定することができます。

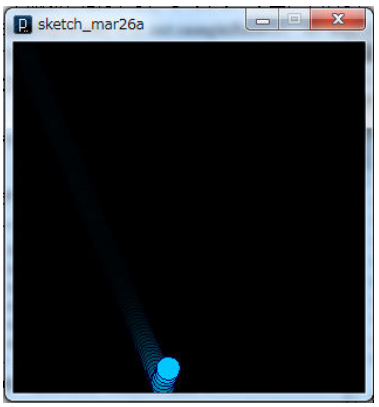
```
sine1 = new SineWave(440, 0.5, out.sampleRate());
sine2 = new SineWave(880, 0.2, out.sampleRate());
sine3 = new SineWave(1320, 0.1, out.sampleRate());
out.addSignal(sine1);
out.addSignal(sine2);
out.addSignal(sine3);
```

6　条件付きで図形を動かす

現実の世界では壁や地面にぶつかったボールは，跳ね返ります。壁にぶつかったかどうかの判定をしたいときには，条件分岐制御を行うための if 文を利用します.

if(条件式){　条件が成り立った際に実行する文;}

では入力しましょう。




```
int r = 10;
float x0 = r, y0 = r;
float v = 30;
float theta = (PI/8)*3;          // θ =3*π/8
float t = 0;
float x = x0, y = y0;
float vx = v * cos(theta), vy = v * sin(theta);
void setup()
{
  size(300,300);
  colorMode(RGB,255,255,255,100);
  ellipseMode(RADIUS);
```

```
   background(0,0,0);
   frameRate(30);
}
void draw()
{
  stroke( 0, 0, 0, 10 );
  fill( 0, 0, 0, 10 );
  rect( 0, 0, width, height );
  x = x0 + vx*t;
  y = y0 + vy*t;
  stroke( 0, 0, 100, 100 );
  fill( 0, 200, 255, 100 );
  ellipse( x, y, 10, 10 );
  t = t + 0.1;
    if( y > height-r )                    //床の中にボールが来たら
  {
      vy = -vy;                           //y 方向の速度を逆向きに
      x0 = x;
      y0 = height-r;
      t = 0;
  }
}
```

○上の天井で跳ね返るようにしてみましょう。また，左右の壁でも反射するようにしてみましょう。

【参考】

・神戸電子専門学校 Web ページ

・Built with Processing[Ver. 1.x 対応版] -デザイン/アートのためのプログラミング入門 , 田中 孝太郎, 前川 峻志, ビー・エヌ・エヌ新社