Buổi 2 : Đồ họa trong không gian 2 chiều

1. Hướng dẫn & bài tập

1.1. Hướng dẫn

Trong buổi 2, sinh viên sẽ thực hiện những phép biến đổi cơ sở trong đồ họa 2 chiều từ cơ bản đến nâng cao.

Lưu ý: **Sinh viên phải khai báo các hàm mới tạo trong file graphics.h và định nghĩa các hàm này trong file graphics.cpp**

Ví dụ khai báo các hàm trong file **graphics.h**

```
#ifndef GRAPHICS_H
#define GRAPHICS_H

#include <QWidget>

class graphics : public QWidget
{
    Q_OBJECT
public:
    explicit graphics(QWidget *parent = 0);
    void paintEvent(QPaintEvent *);
    int mode=0;
    int h=height();
    int w=width();
    int random(int n);
    QPoint tinhtien(QPoint p,int tx,int ty);
    QPoint quay(QPoint p, QPoint c, int deta);//quay diem p quanh diem c 1 goc deta (don vi do)
    QPoint tile(QPoint p, QPoint c, int sx,int sy);
    QPoint doixungx(QPoint p);
    QPoint doixungy(QPoint p);

    void ngansao(QPainter& painter);//mode=1
    void ngugiac(QPainter& painter);//mode=2
    void bonghoa(QPainter& painter);//mode=3
    void thaicuc(QPainter& painter);//mode=4
    void thaicuc_mau(QPainter& painter);//mode=5
    void quocky(QPainter& painter);//mode=6
    void ngoilang(QPainter& painter);//mode=7


signals:

public slots:

};

#endif // GRAPHICS_H
```

Ví dụ định nghĩa hàm trong file **graphics.cpp**

```
#include "graphics.h"
#include <QPainter>
#include <math.h>

graphics::graphics(QWidget *parent) :
    QWidget(parent)
{
}
void graphics::paintEvent(QPaintEvent *){
    QPainter painter(this);
    painter.setPen(Qt::blue);
    painter.setRenderHint(QPainter::Antialiasing);

    //gọi các hàm
    if (mode==1) ngansao(painter);
    if (mode==2) {ngugiac(painter);}
    if (mode==3) bonghoa(painter);
    if (mode==4) thaicuc(painter);
    if (mode==5) thaicuc_mau(painter);
    if (mode==6) quocky(painter);
    if (mode==7) ngoilang(painter);
 }
```

Trong hình trên, để các hình vẽ được đẹp hơn, sinh viên thêm dòng code painter.setRenderHint(QPainter::Antialiasing) vào sự kiện paintEvent ()

Sinh viên cần phân biệt cách sử dụng QPen và QBrush.

+QPen, đặt màu cho nét vẽ đường viền bên ngoài.

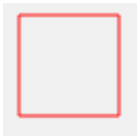
+QBrush, đặt màu tô nguyên khối hình.

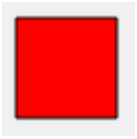
Ví dụ các câu lệnh sau:

```
painter.setPen(Qt::red);
painter.drawRect(10,10,50,50);
```

```
painter.setBrush(QBrush(Qt::red));
painter.drawRect(10,10,50,50);
```

Kết quả :





Nếu khi tô màu không muốn hiển thị đường viền thì ta thêm dòng sau :

painter.setPen(Qt::NoPen);

Kết quả :

1.2. Bài tập

1) Sinh viên khảo sát các phép biến đổi cơ bản

B1: Vẽ 2 đoạn thẳng với toạ độ 2 đầu mút (0,0) và (200,0); (0,0) và (0, 200)

B2: Vẽ hình chữ nhật với toạ độ góc trên bên trái (0,0), độ dài là 100, độ rộng là 50

B3: Thực hiện tịnh tiến với véc-tơ tịnh tiến (200,100), sử dụng hàm translate(200,100)

B4: Thực hiện xoay một góc 30^0, sử dụng hàm rotate(30)

B5: Vẽ lại các hình trên

Thay đổi trình tự bước 3 và bước 4, xem kết quả và giải thích

```
void graphics::paintEvent(QPaintEvent *){
    QPainter painter(this);
    painter.setPen(Qt::red);
    painter.drawLine(0,0,200,0);
    painter.drawLine(0,0,0,200);
    painter.drawRect(0,0,100,50);

    painter.translate(200,100);
    painter.rotate(30);

    painter.setPen(Qt::blue);
    painter.drawLine(0,0,200,0);
    painter.drawLine(0,0,0,200);
    painter.drawRect(0,0,100,50);
}
```

2) Cho hàm Random như sau:

```
int graphics::random(int n){
    return rand() % n;
}
```

Hàm này sẽ trả về một số ngẫu nhiên trong khoảng từ 0 đến n. Ví dụ khi hàm Random(1000) được gọi thì nó sẽ trả về một số ngẫu nhiên nằm giữa 0 và 1000.

Dùng hàm Random để viết hàm vẽ 1000 điểm có tọa độ ngẫu nhiên trong cửa sổ quan sát. Những điểm vẽ rất nhỏ nên chúng ta có thể không nhìn thấy. Thay vì vẽ 1000 điểm ngẫu nhiên, sinh viên hãy vẽ 1000 hình tròn với các màu thay đổi ngẫu nhiên, bán kính vừa đủ thấy như một điểm (r=3).

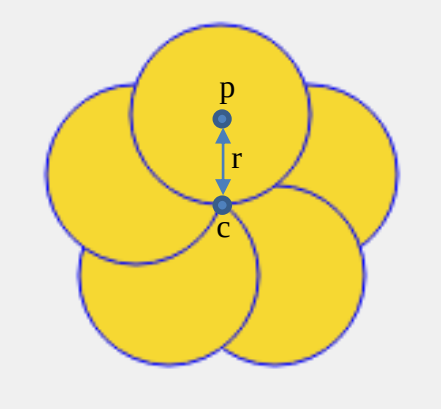
⇨Sinh viên vẽ bầu trời ngàn sao:

```
void graphics::ngansao(QPainter& painter){//mode=1
    for (int i=1;i<=1000;i++){
        painter.setPen(Qt::NoPen);
        painter.setBrush(QColor(random(255),random(255),random(255)));
        painter.drawEllipse(random(1000),random(1000),3,3);
    }
}
```

* Sinh viên viết các hàm biến đổi cơ bản:

```
QPoint graphics::tinhtien(QPoint p,int tx,int ty){
    QPoint pnew;
    pnew.setX(p.x()+tx);
     pnew.setY(p.y()+ty);
    return pnew; // w=(p.x()+tx,p.y()+ty);
}
QPoint graphics::quay(QPoint p, QPoint c, int deta){//quay điểm p quanh điểm bất kỳ c, 1 góc deta (độ)
    QPoint pnew;
    double goc=deta*3.14/180; // đổi độ về radian
    pnew.setX(c.x()+(p.x()-c.x())*cos(goc)-(p.y()-c.y())*sin(goc));
    pnew.setY(c.y()+(p.x()-c.x())*sin(goc)+(p.y()-c.y())*cos(goc));
    return pnew;
}
QPoint graphics::tile(QPoint p, QPoint c, int sx,int sy){
    QPoint pnew;
    pnew.setX(p.x()*sx+c.x()*(1-sx));
    pnew.setY(p.y()*sy+c.y()*(1-sy));
    return pnew;
}
QPoint graphics::doixungx(QPoint p){//đối xứng qua trục x
    QPoint pnew;
    pnew.setX(p.x());
    pnew.setY(-p.y());
    return pnew;
}
QPoint graphics::doixungy(QPoint p){//đối xứng qua trục y
    QPoint pnew;
    pnew.setX(-p.x());
    pnew.setY(p.y());
    return pnew;
}
```

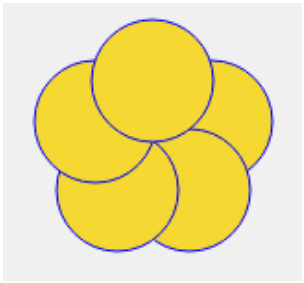
3) Vẽ bông hoa tại vị trí bất kỳ



Ý tưởng:

1. Bắt đầu từ điểm bất kỳ c(x, y), gọi là tâm quay.
2. Tạo 1 điểm mới p, cách tâm quay một khoảng r trên trục y. Điểm p(x, y - r) làm điểm bắt đầu (để quay).
3. Quay điểm p quay điểm c(x, y) một góc tương ứng (360/5, 2* 360/5…) và vẽ hình tròn tại điểm mới (pnew) với bán kính r.

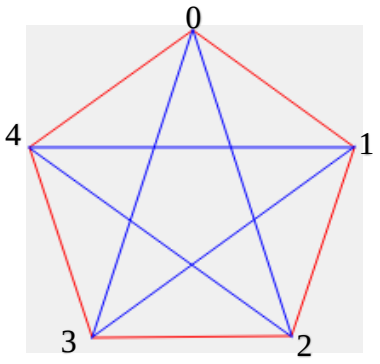
```
void graphics::bonghoa(QPainter& painter){//mode=3
    painter.setBrush(QColor(random(255),random(255),random(255)));
    int x=random(width());
    int y=random(height());

    //painter.drawLine(0,0,x,y);
    int d=100;//duong kinh;
    int r=d/3;//ban kinh;
    QPoint c(x,y);//tam quay
    QPoint p(x,y-r);//diem bat dau
    for (int i=1;i<=5;i++){
        QPoint pnew=quay(p,c,i*72);
        painter.drawEllipse(pnew,r,r);
    }
}
```



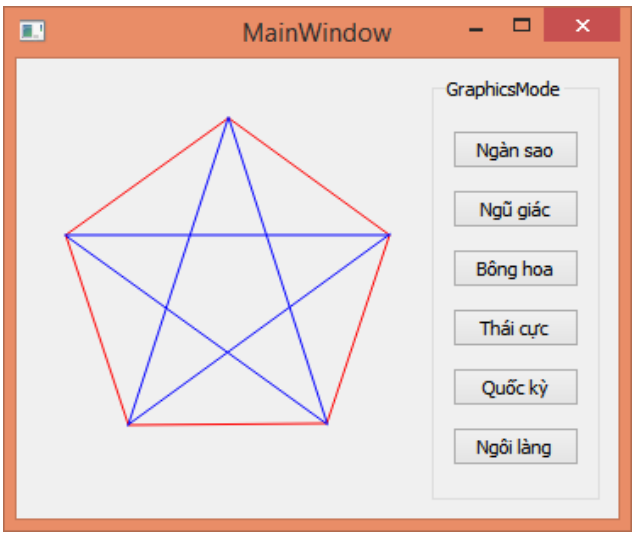
4) Viết thủ tục để vẽ hình ngôi sao trong ngũ giác đều.
   Ý tưởng vẽ ngũ giác giác đều:
   1. Cho tâm quay c(x,y)
   2. Bắt đầu từ điểm p(x, y-kc), cách tâm quay 1 khoảng cách kc. Ví dụ kc là 100 thì p(x, y-100)
   3. Đưa đỉnh p vào đa giác: polygon<<QPoint(p.x(),p.y())
   4. Tìm toạ độ 4 đỉnh tiếp theo bằng cách quay đỉnh p, quanh tâm c một góc tương ứng: 360/5=72 độ, 2*72 độ,….và lần lượt lưu toạ độ 4 đỉnh vào đa giác polygon.

   Ý tưởng vẽ ngôi sao trong ngũ giác giác đều: vẽ các đoạn thẳng có toạ độ 2 điểm đầu mút lưu trong đa giác polygon. Ví dụ đoạn thẳng có 2 đỉnh đầu mút là đỉnh 0 và đỉnh 3.

```
painter.drawLine(polygon.value(0),polygon.value(3));
```



```
void graphics::ngugiac(QPainter& painter){//mode=2

    painter.setPen(Qt::red);
    QPolygon polygon;
    int x=width()/2;
    int y=height()/2;
    QPoint p(x,y-100);//dinh 1
    QPoint c(x,y);//tam quay
    polygon <<QPoint(p.x(),p.y());
    for (int i=1;i<5;i++){
        QPoint pnew=quay(p,c,i*72);
        polygon <<QPoint(pnew.x(),pnew.y());
    }
     painter.drawPolygon( polygon);
    //vẽ ngôi sao trong ngũ giác
     painter.setPen(Qt::blue);
     painter.drawLine(polygon.value(0),polygon.value(3));
     painter.drawLine(polygon.value(0),polygon.value(2));
     painter.drawLine(polygon.value(1),polygon.value(3));
     painter.drawLine(polygon.value(2),polygon.value(4));
     painter.drawLine(polygon.value(1),polygon.value(4));
}
```

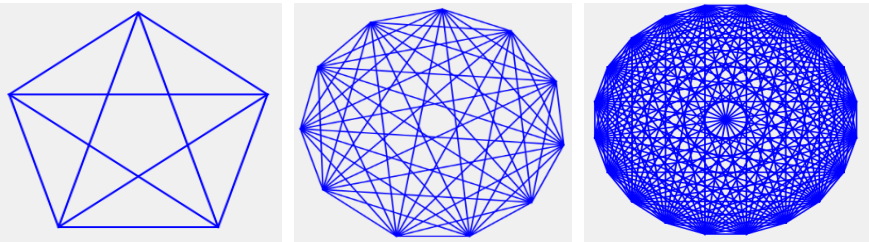


5) Viết thủ tục để vẽ hình thái cực, sử dụng các phép biến đổi cơ bản.
Hình thái cực được tạo ra từ 1 hình tròn lớn (bán kính r=200), 2 hình tròn nhỏ (bán kính khoảng r/10) và 2 nữa cung tròn (đối xứng với nhau qua 2 trục x, y - đối xứng qua trục x và tịnh tiến theo trục x khoảng r)

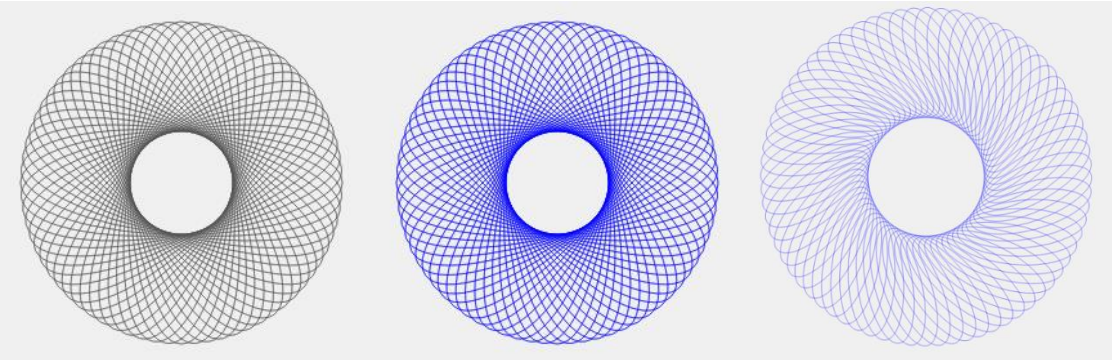
```
    //su dung phuong phap doi xung, tinh tien
void graphics::thaicuc(QPainter& painter){
        painter.setPen(Qt::red);
        int x=width()/2;
        int y=height()/2;
        int r=200;//ban kinh duong tron lon
        int r1=r/2;//ban kinh duong tron nho
        int xl=x-r;//toa do truc x duong tron ben trai
        int yl=y-r1;//toa do truc y duong tron ben trai
        int xr=x;//toa do truc x duong tron ben trai
        int yr=y-r1;//toa do truc y duong tron ben trai
        QPoint p(x,y);//tam duong tron lon

        painter.drawEllipse(p,r,r);
        painter.drawArc(xl,yl,r,r,0,-(180*16));

        xr=doixungx(QPoint(xl,yl)).x();
        yr=doixungy(QPoint(xl,yl)).y();
        xr=tinhtien(QPoint(xr,yr),r,0).x();

        painter.drawArc(xr,yr,r,r,0,(180*16));
        QPoint pl(x-r/2,y);//tam duong tron ben trai
        QPoint pr(x+r/2,y);//tam duong tron ben phai
        painter.drawEllipse(pl,r/10,r/10);
        painter.drawEllipse(pr,r/10,r/10);
}
```

2. Bài tập nâng cao:

6) Một trong những biến thể đặc biệt của đa giác đều đó là hình Rosette. Hình Rosette là một đa giác đều với mỗi đỉnh được nối với tất cả các đỉnh còn lại. Hãy vẽ:
   a) Các đa giác đều có n cạnh, với n = 3, 4, 5, 6, 12, 16, 40. Rút ra nhận xét khi ta tăng số cạnh n?
   b) Các Rosette có 5 đỉnh, 11 đỉnh, 20 đỉnh.



7) Viết thủ tục để vẽ các hình sau:



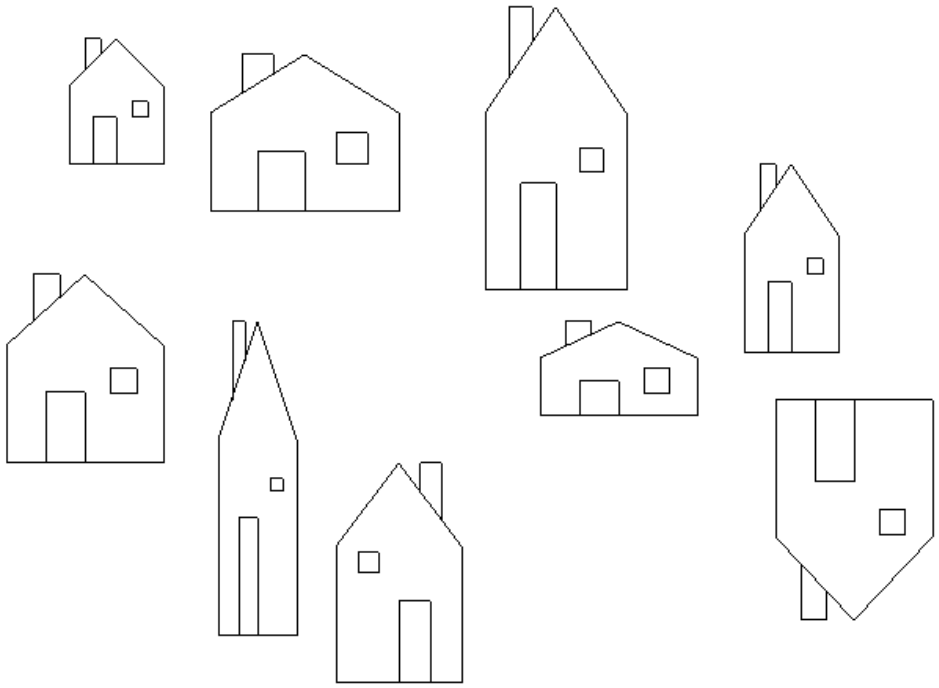
Những hình này có thể vẽ một cách dễ dàng bằng cách vẽ một loạt các Ellipse xoay xung quanh tâm điểm của hình. Gợi ý: Sinh viên sử dụng hàm painter.translate() và hàm painter.rotate().

8) Viết thủ tục để vẽ lại ngôi nhà ở buổi 1 dưới dạng tham số hóa

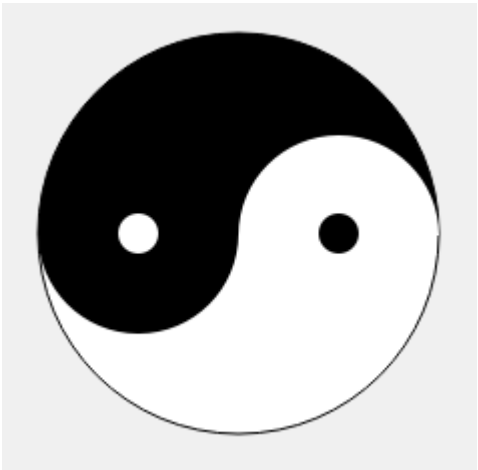
Ở buổi 1, SV vẽ ngôi nhà bằng cách gán cứng các tọa độ vào các đỉnh của ngôi nhà. Nếu muốn thay đổi kích thước sẽ rất khó khăn.

Hãy viết lại hàm vẽ ngôi nhà dưới dạng tham số hóa. Các tham số truyền vào sẽ là tọa độ đỉnh của nóc nhà, chiều ngang và chiều cao của ngôi nhà.

Dùng hàm đó để vẽ một ngôi làng như sau :



9) Viết thủ tục để vẽ & tô màu cho hình thái cực

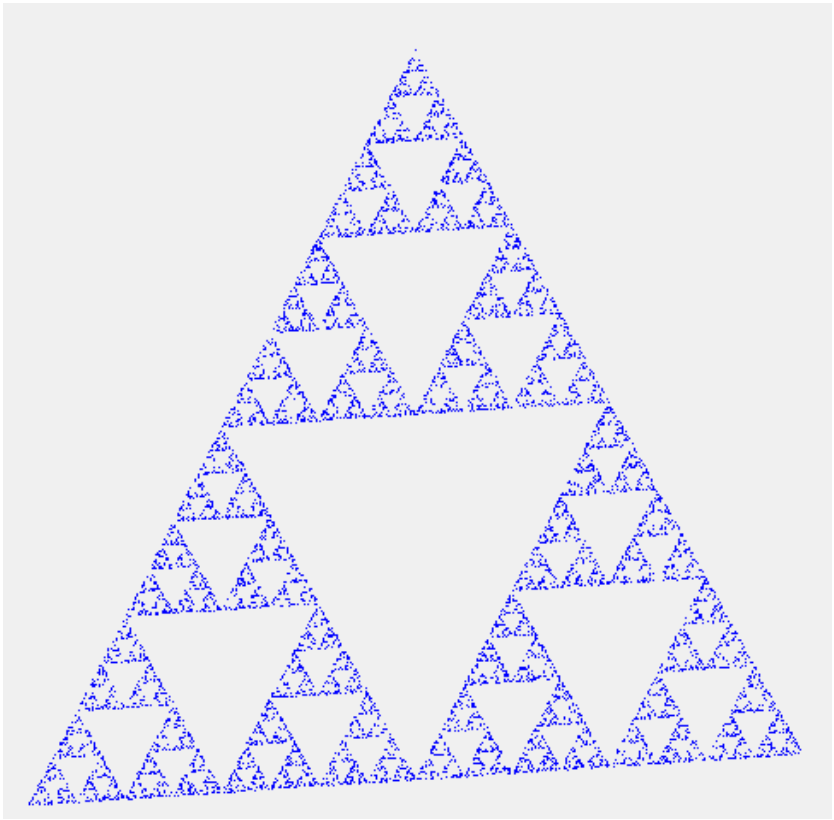


Gợi ý: chia hình tròn lớn ra làm 2 nữa vòng tròn nhỏ, nữa trên tô màu đen, nữa dưới tô màu trắng; sau đó vẽ thêm 2 nữa cung tròn nhỏ nữa nằm chồng lên 2 cung tròn lớn; cuối cùng là vẽ 2 hình tròn nhỏ.

Vẽ và tô màu cho các cung tròn bằng hàm drawChord() thay vì drawArc(), cách sử dụng cũng giống như drawArc().

10) Viết thủ tục để vẽ và tô màu cho Lá cờ tổ quốc.

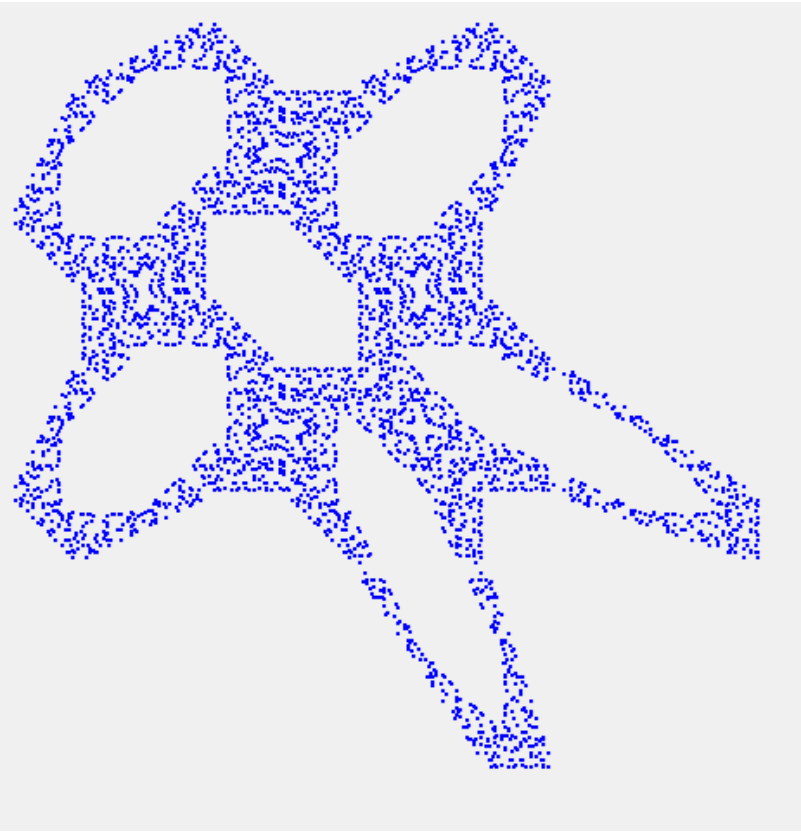
11) Viết thủ tục để vẽ Sierpinski gasket.



Cách vẽ :

- Chọn 3 điểm cố định P_1, P_2, P_3 làm 3 đỉnh của tam giác lớn. Lưu ý không vẽ 3 điểm này lên màn hình.
- Chọn một trong 3 điểm làm đỉnh gốc G_0 bằng cách chọn ngẫu nhiên 1 trong 3 đỉnh (dùng hàm Random), vẽ đỉnh gốc này.
- Lặp lại 2 bước sau đến khi nào vừa ý (Khoảng 10.000 lần)
    - Chọn một điểm bất kỳ trong số 3 điểm P_1, P_2, P_3, Gọi đó là điểm P;(hàm Random())
    - Tạo điểm tiếp theo (G_n) bằng cách lấy trung điểm của đoạn thẳng nối điểm P và điểm G trước đó (G_{n-1}). Tức là G_n = trung điểm của G_{n-1} và P
    - Vẽ điểm G_n bằng hàm drawPoint();

12) Viết thủ tục để vẽ "chiếc bánh hình người" (The Gingerbread Man)



Chiếc bánh hình người được vẽ bởi tập hợp các điểm. Đây làm ột đối tượng quen thuộc của lý thuyết hỗn loạn (Chaos Theory). Các bước để vẽ như sau :

- Chọn điểm bắt đầu là điểm có tọa độ P(115,121).
- Lặp đi lặp lại các bước sau 10000 lần :
  - Tính điểm vẽ Q(x, y) được theo công thức :
    Q.x = 40(1 + 2*3) – P.y + |P.x – 40*3|
    Q.y = P.x
  - Vẽ điểm Q(x,y).
  - Coi điểm Q là điểm P và tiếp tục tính điểm Q mới theo công thức