Računarska grafika - vežbe

2 – Parametarske krive

Bezjeova kriva, Katmul-Rom splajn

Kubna Bezjeova kriva

$$P(t) = (1- t)\cdot P_0 + 3t(1- t)\cdot P_1 + 3t(1- t)\cdot P_2 + t\cdot P_3$$

za $t \in [0,1]$

$$P(t) = [1 \;\; t \;\; t^2 \;\; t^3] \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

Bezjeova kriva proizvoljnog reda

- U analitičkom obliku:

$$P(t) = \sum_{i=0}^{n} P_i \cdot B_{i,n}(t), t \in [0,1]$$

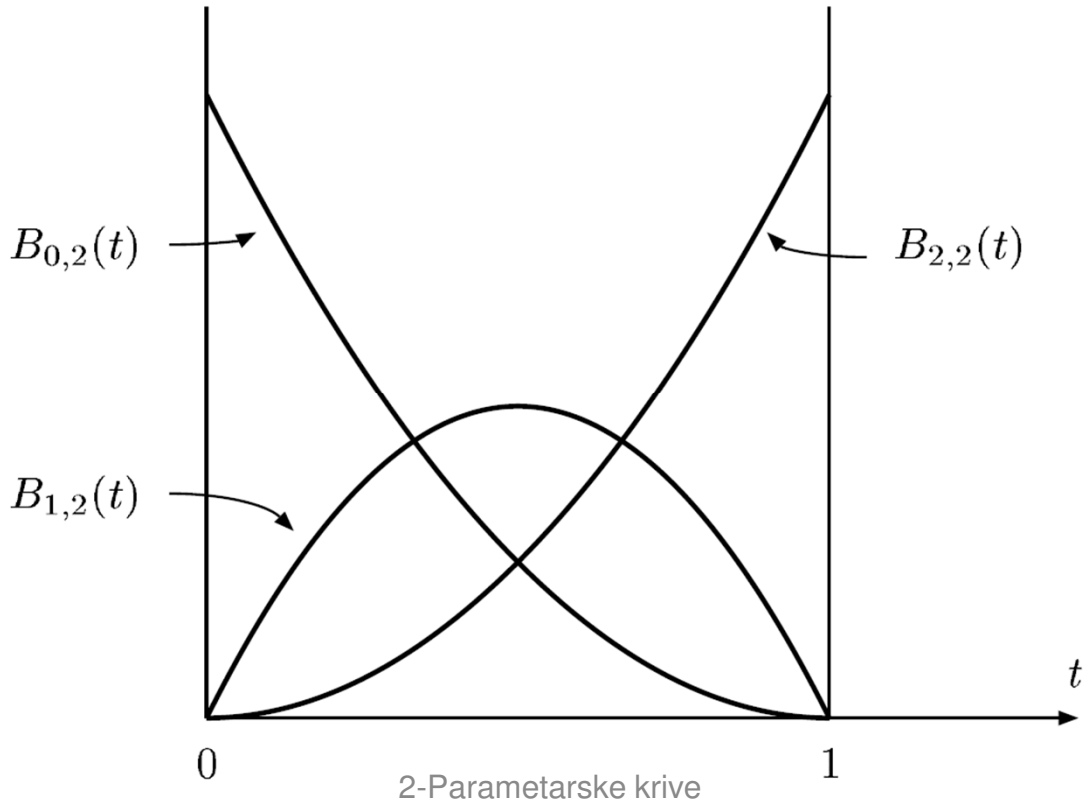
gde je $B_{i,n}(t)$ Bernštajnov polinom

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

Bernštajnov polinom

- Zanimljiva osobina Bernštajnovog polinoma

$$\sum_{i=0}^{n} B_{i,n}(t) = 1, t \in [0,1]$$



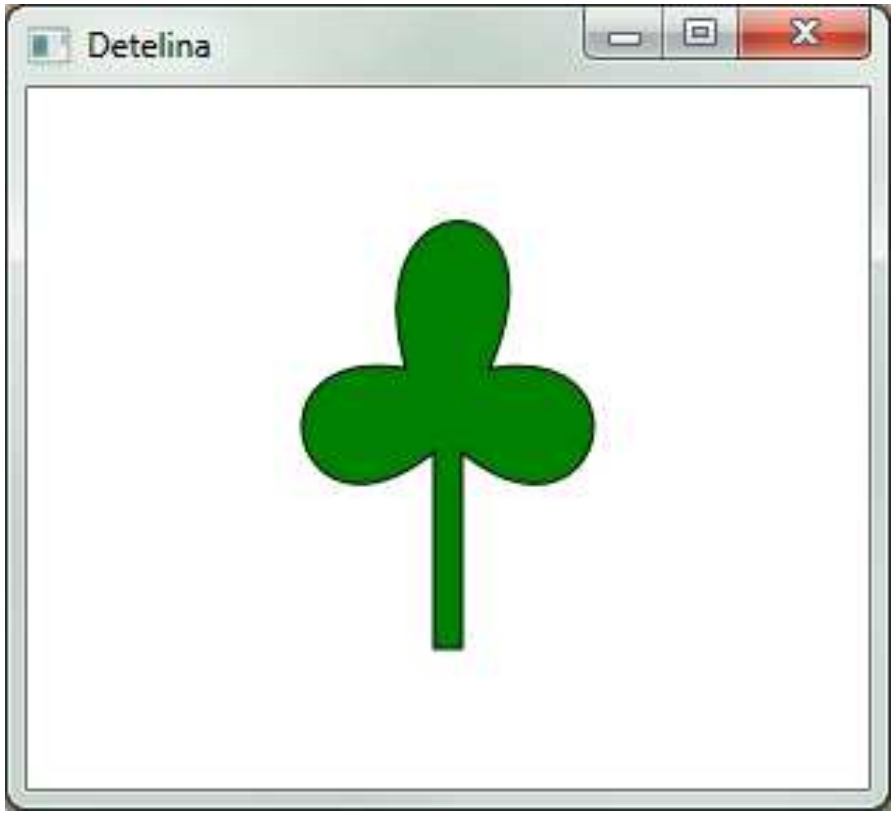
Osobine Bezjeovih krivih

- Početna i krajnja tačka (P_0 i P_n) leže na krivoj
  - u opštem slučaju ostale tačke ne leže na krivoj
- Kriva je neprekidna
  - diferencijabilna je u svakoj tački proizvoljan broj puta
- U tački P_0, tangenta krive je duž $\overline{P_0P_1}$

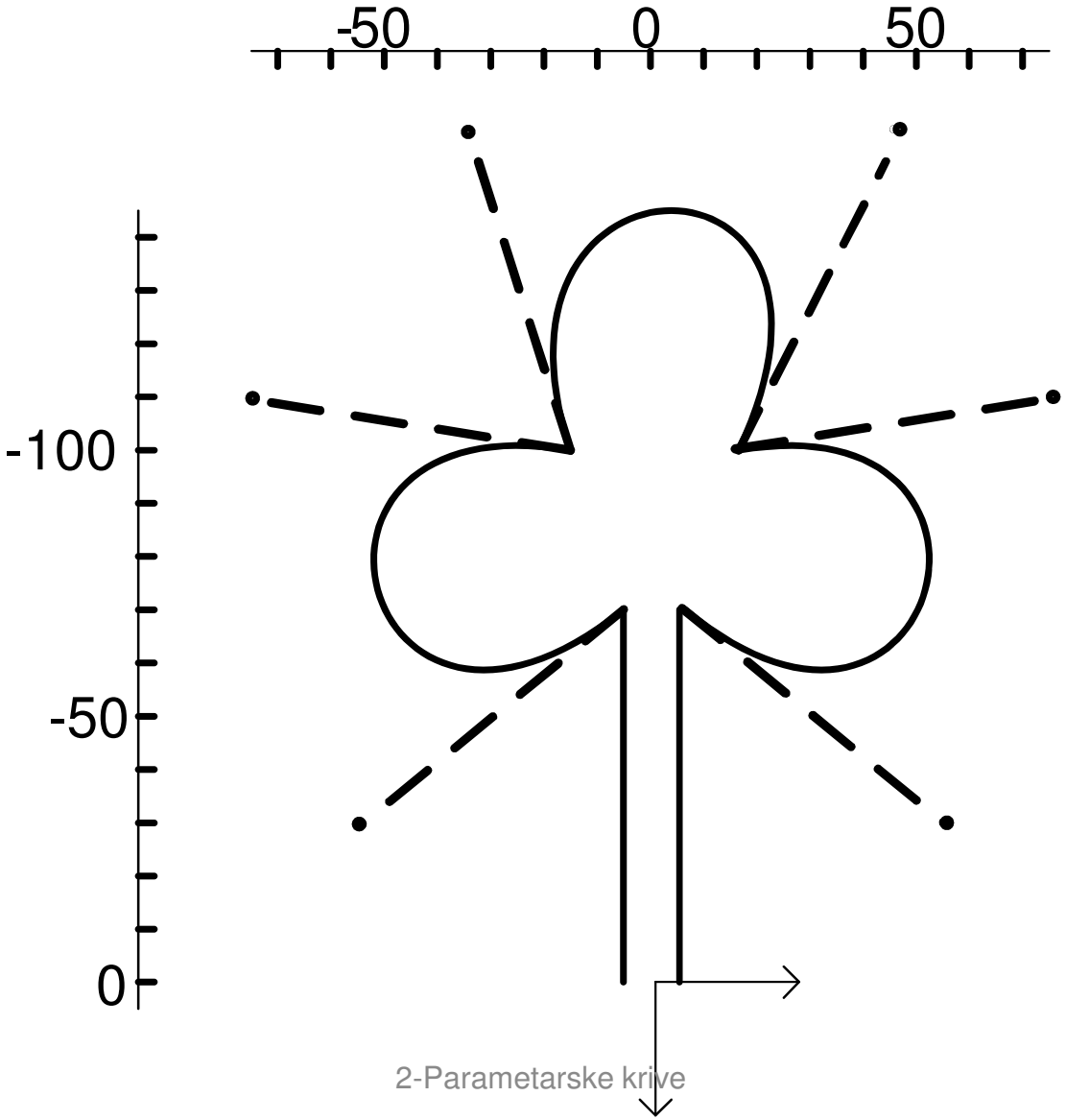
  u tački P_n, tangenta krive je duž $\overline{P_{n-1}P_n}$
- Kriva se nalazi unutar konveksnog omotača
  - omotač formiraju kontrolne tačke
- Na izgled krive utiču sve tačke iz zadatog niza
  - pomeranjem makar jedne tačke kriva menja svoj oblik

Zadatak 1: Detelina

- Napisati program koji crta trodelni list deteline koristeći grafički paket JavaFX.

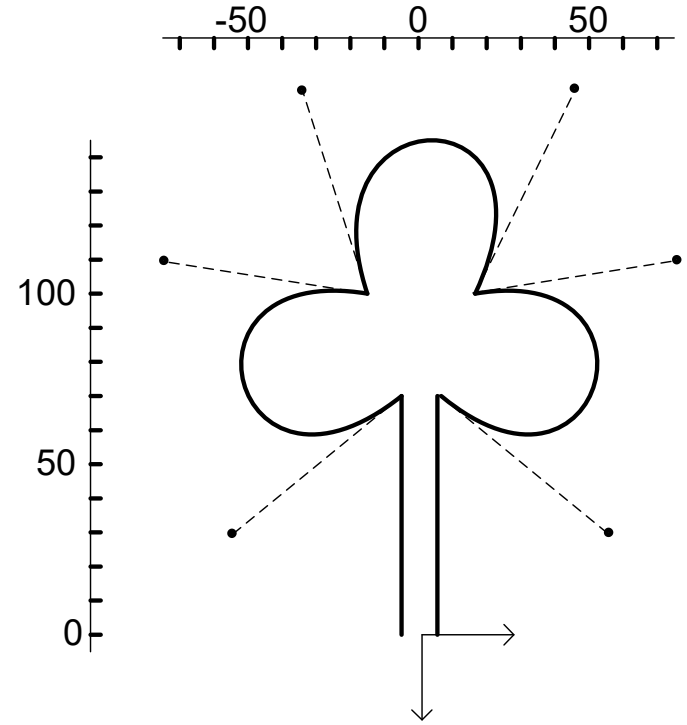


Rešenje: Detelina (1/6)



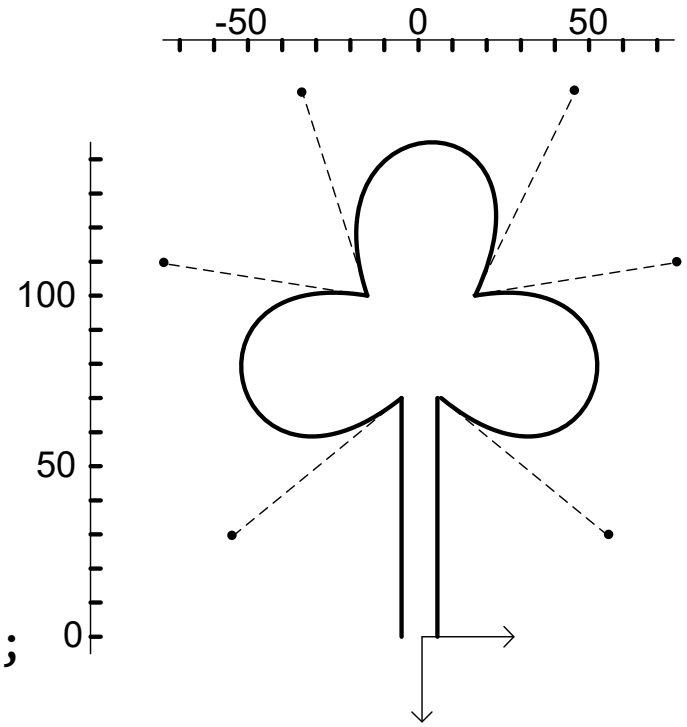
Rešenje: Detelina (2/6)

```
import ...
public class Detelina
                extends Application {

  @Override
  public void start(Stage prozor) {
    Group koren = new Group();
    koren.translateXProperty().set(150);
    koren.translateYProperty().set(200);

    Path putanja = new Path();
    putanja.setFill( Color.GREEN );
    putanja.setStroke( Color.BLACK );
```



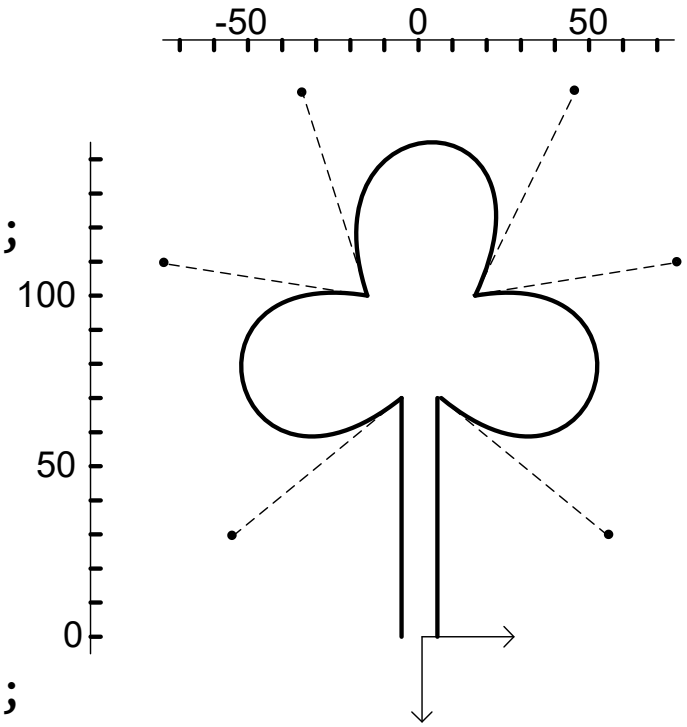
Rešenje: Detelina (3/6)

```
MoveTo pomeraj = new MoveTo();
pomeraj.setX(-5.0f);
pomeraj.setY(0.0f);

VLineTo vLinija1 = new VLineTo();
vLinija1.setY(-70.0f);

CubicCurveTo kriva1 = new CubicCurveTo();
kriva1.setX(-15.0);
kriva1.setY(-100.0);
kriva1.setControlX1(-55.0);
kriva1.setControlY1(-30.0);
kriva1.setControlX2(-75.0);
kriva1.setControlY2(-110.0);
```



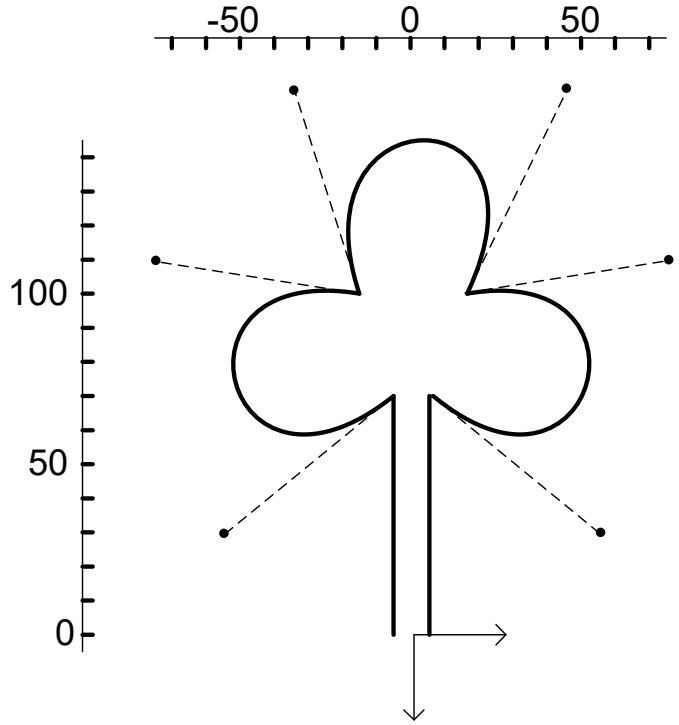
Rešenje: Detelina (4/6)

```
CubicCurveTo kriva2 = new CubicCurveTo();
kriva2.setX(15.0);
kriva2.setY(-100.0);
kriva2.setControlX1(-35.0);
kriva2.setControlY1(-170.0);
kriva2.setControlX2(45.0);
kriva2.setControlY2(-170.0);

CubicCurveTo kriva3 = new CubicCurveTo();
kriva3.setX(5.0);
kriva3.setY(-70.0);
kriva3.setControlX1(75.0);
kriva3.setControlY1(-110.0);
kriva3.setControlX2(55.0);
kriva3.setControlY2(-30.0);
```



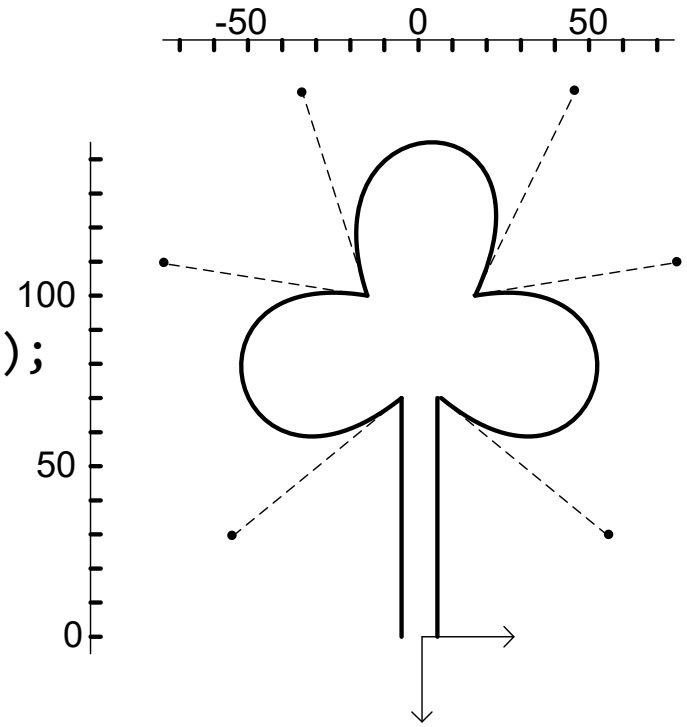
Rešenje: Detelina (5/6)

```
VLineTo vLinija2 = new VLineTo();
vLinija2.setY(0.0f);

ClosePath zatvaranje = new ClosePath();

putanja.getElements().add(pomeraj);
putanja.getElements().add(vLinija1);
putanja.getElements().add(kriva1);
putanja.getElements().add(kriva2);
putanja.getElements().add(kriva3);
putanja.getElements().add(vLinija2);
putanja.getElements().add(zatvaranje);
```



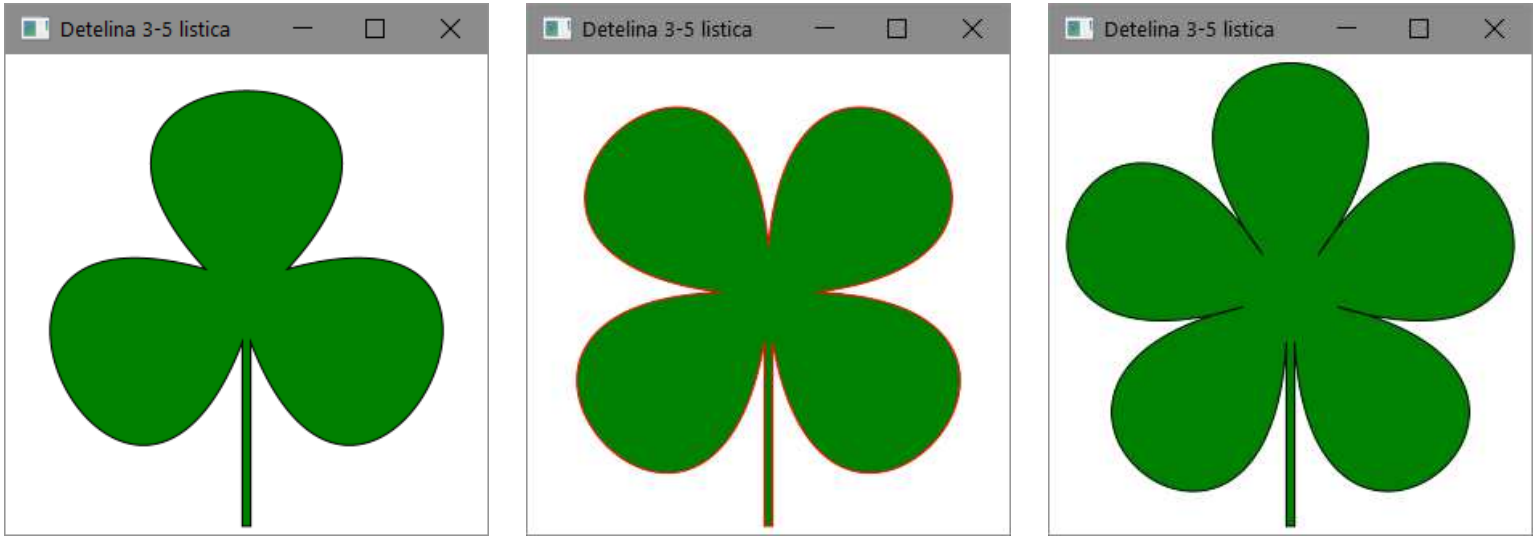
Rešenje: Detelina (6/6)

```
    koren.getChildren().add(putanja);

    Scene scena = new Scene(koren, 300, 250);

    prozor.setTitle("Detelina");
    prozor.setScene(scena);
    prozor.setResizable(false);
    prozor.show();
  }

  public static void main(String[] arg) {
    launch(arg);
  }
}
```

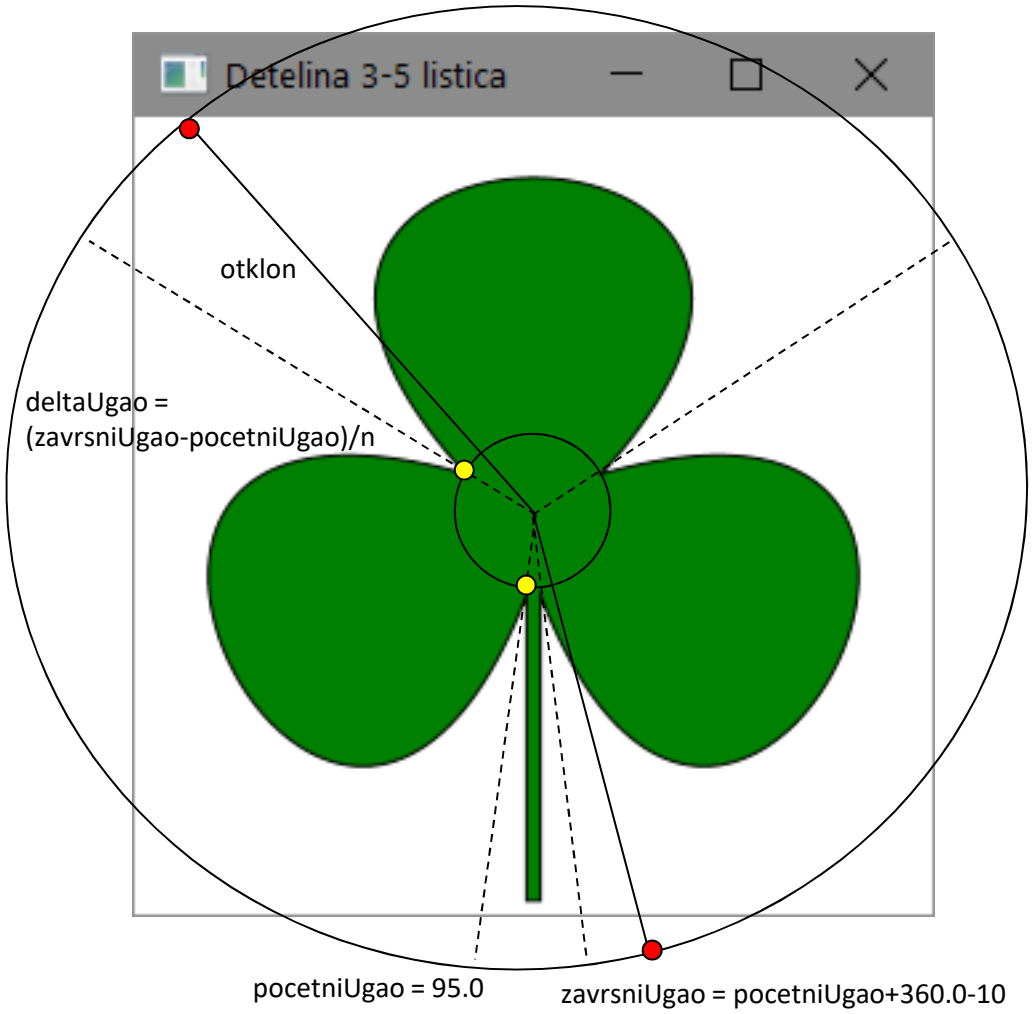


Zadatak 2 – Detelina 3-5

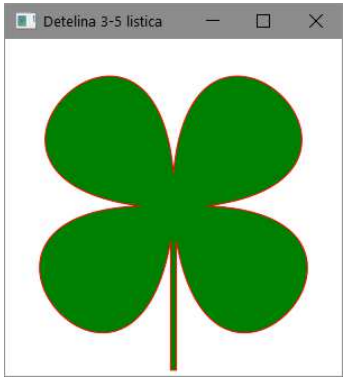
- Napisati program koji crta slučajno izabranu detelinu sa od 3 do 5 listića



Rešenje: Detelina 3-5 (1/8)

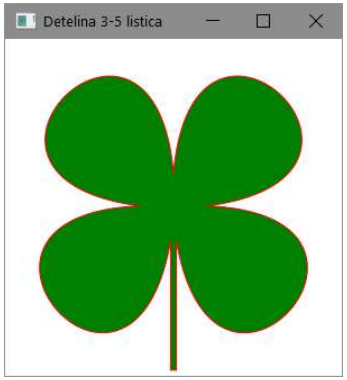


Rešenje: Detelina 3-5 (2/8)



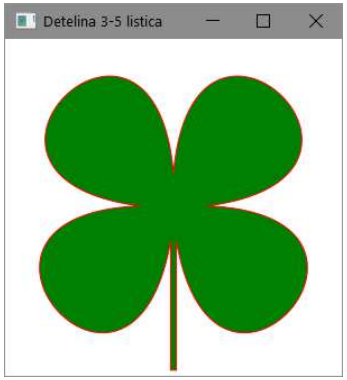

```
package detelina;
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.Group;
import javafx.scene.shape.MoveTo;
import javafx.scene.shape.VLineTo;
import javafx.scene.shape.CubicCurveTo;
import javafx.scene.shape.ClosePath;
import javafx.scene.shape.Path;
import javafx.scene.paint.Color;
import java.util.Random;
```

Rešenje: Detelina 3-5 (3/8)




```
public class Detelina3_5 extends Application {

  static final int SIRINA=300;
  static final int VISINA=300;
  final double unutrasnjiR = 30.0;
  final double spoljasnjiR = VISINA-70;
  final double pocetniUgao = 95.0;
  final double zavrsniUgao = pocetniUgao+360.0-10;
  final double otklon = 25;

  double kosinus(double ugao){return Math.cos(2*Math.PI*ugao/360);}
  double sinus(double ugao){return Math.sin(2*Math.PI*ugao/360);}
```

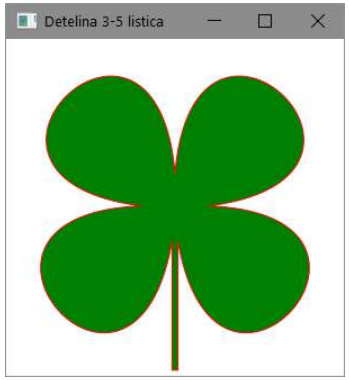
Rešenje: Detelina 3-5 (4/8)



```
@Override public void start(Stage prozor) {
    Random rnd = new Random();
    Group koren = new Group();
    koren.translateXProperty().set(SIRINA/2);
    koren.translateYProperty().set(VISINA/2);
    Path putanja = new Path();
    putanja.setFill( Color.GREEN );

    int n=rnd.nextInt(3)+3;
    if (n==4) putanja.setStroke( Color.RED );
    else putanja.setStroke( Color.BLACK );
```

Rešenje: Detelina 3-5 (5/8)

```
double deltaUgao = (zavrsniUgao-pocetniUgao)/n;
double ugao = pocetniUgao;

double xPocetno = unutrasnjiR*kosinus(ugao);
double yPocetno = unutrasnjiR*sinus(ugao);
MoveTo pomeraj = new MoveTo();
pomeraj.setX(xPocetno);
pomeraj.setY(yPocetno+spoljasnjiR/2);
putanja.getElements().add(pomeraj);
VLineTo vLinija1 = new VLineTo();
vLinija1.setY(yPocetno);
putanja.getElements().add(vLinija1);
```

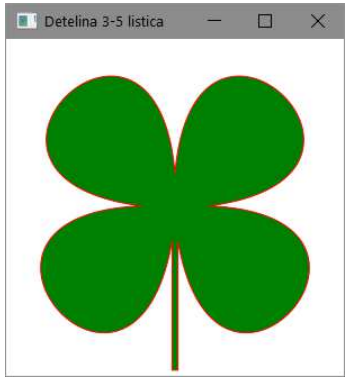


Rešenje: Detelina 3-5 (6/8)

```
double xz=0; double yz=0;
for (int i=1; i<=n; i++){

  double xk1 = spoljasnjiR*kosinus(ugao+deltaUgao/3-otklon);
  double yk1 = spoljasnjiR*sinus(ugao+deltaUgao/3-otklon);
  double xk2 = spoljasnjiR*kosinus(ugao+2*deltaUgao/3+otklon);
  double yk2 = spoljasnjiR*sinus(ugao+2*deltaUgao/3+otklon);

  ugao += deltaUgao;
  xz = unutrasnjiR*kosinus(ugao);
  yz = unutrasnjiR*sinus(ugao);
```

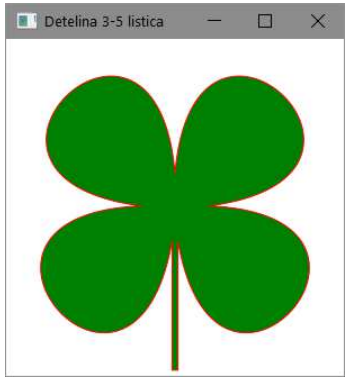
Rešenje: Detelina 3-5 (7/8)

```
    CubicCurveTo kriva = new CubicCurveTo();
    kriva.setControlX1(xk1);
    kriva.setControlY1(yk1);
    kriva.setControlX2(xk2);
    kriva.setControlY2(yk2);
    kriva.setX(xz);
    kriva.setY(yz);
    putanja.getElements().add(kriva);
}

VLineTo vLinija2 = new VLineTo();
vLinija2.setY(yz+spoljasnjiR/2);
putanja.getElements().add(vLinija2);
```



Rešenje: Detelina 3-5 (8/8)

```
        ClosePath zatvaranje = new ClosePath();
        putanja.getElements().add(zatvaranje);

        koren.getChildren().add(putanja);
        Scene scena = new Scene(koren, SIRINA, VISINA);

        prozor.setTitle("Detelina 3-5 listica");
        prozor.setScene(scena);
        prozor.setResizable(false);
        prozor.show();
    }
    public static void main(String[] arg) {launch(arg);}
}
```



Katmul-Romova kriva - uvod

- Objavljena 10 godina nakon Bezjeove (1974)
- Interpolativna lokalna kriva
  - interpolativna:
    kriva prolazi kroz sve kontrolne tačke
  - lokalna:
    izgled zavisi samo od nekoliko susednih kontrolnih tačaka
- Oslanja se na Fergusonove parametarske kubne krive
  - date su tačke P_0 i P_1
  - dati su nagibi tangenta krive u tim tačkama (P_0' i P_1' respektivno)

Fergusonova kriva (1)

- U zavisnosti od parametra t, pozicija tačke na krivoj se određuje polinomom trećeg stepena:

$$P(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

- Parametar t uzima vrednost u opsegu [0,1]
- Potrebno je odrediti koeficijente a_0 do a_3

$$P(0) = a_0 \qquad P'(0) = a_1$$

$$P(1) = a_0 + a_1 + a_2 + a_3 \qquad P'(1) = a_1 + 2a_2 + 3a_3$$

- Poznato: $P(0)=\mathrm{P}_0$, $P(1)=\mathrm{P}_1$, $P'(0)=\mathrm{P}_0'$ i $P'(1)=\mathrm{P}_1'$

Fergusonova kriva (2)

- Rešavanjem datog sistema jednačina, dobija se:

$$a_0 = P(0) \qquad a_2 = 3[P(1) - P(0)] - 2P'(0) - P'(1)$$

$$a_1 = P'(0) \qquad a_3 = 2[P(0) - P(1)] + P'(0) + P'(1)$$

- U analitičkom obliku:

$$\boxed{P(t) = (1 - 3t^2 + 2t^3)P(0) + (3t^2 - 2t^3)P(1) + (t - 2t^2 + t^3)P'(0) + (-t^2 + t^3)P'(1)}$$

Matrični zapis Fergusonove krive

$$P(t) = [1 \quad t \quad t^2 \quad t^3] * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} * \begin{bmatrix} P(0) \\ P(1) \\ P'(0) \\ P'(1) \end{bmatrix}$$

$$P(t) = T * F * \begin{bmatrix} P(0) \\ P(1) \\ P'(0) \\ P'(1) \end{bmatrix}$$

Katmul-Romova kriva (1)

- Specijalan slučaj kardinalne (kanoničke) krive
- Kardinalna kriva je specijalan slučaj Hermitove krive
- Katmul-Romova (Catmull-Rom) kriva (splajn)
  - razvija se nad nizom od n+1 tačaka (u oznaci P_0 do P_n)
- Kriva se lokalno definiše, deo po deo, nad lukom P_iP_{i+1}
- Nagib tangente u tačkama P_i i P_{i+1} :

$$P_i' = \frac{P_{i+1} - P_{i-1}}{2} \qquad P_{i+1}' = \frac{P_{i+2} - P_i}{2}$$

Katmul-Romova kriva (2)

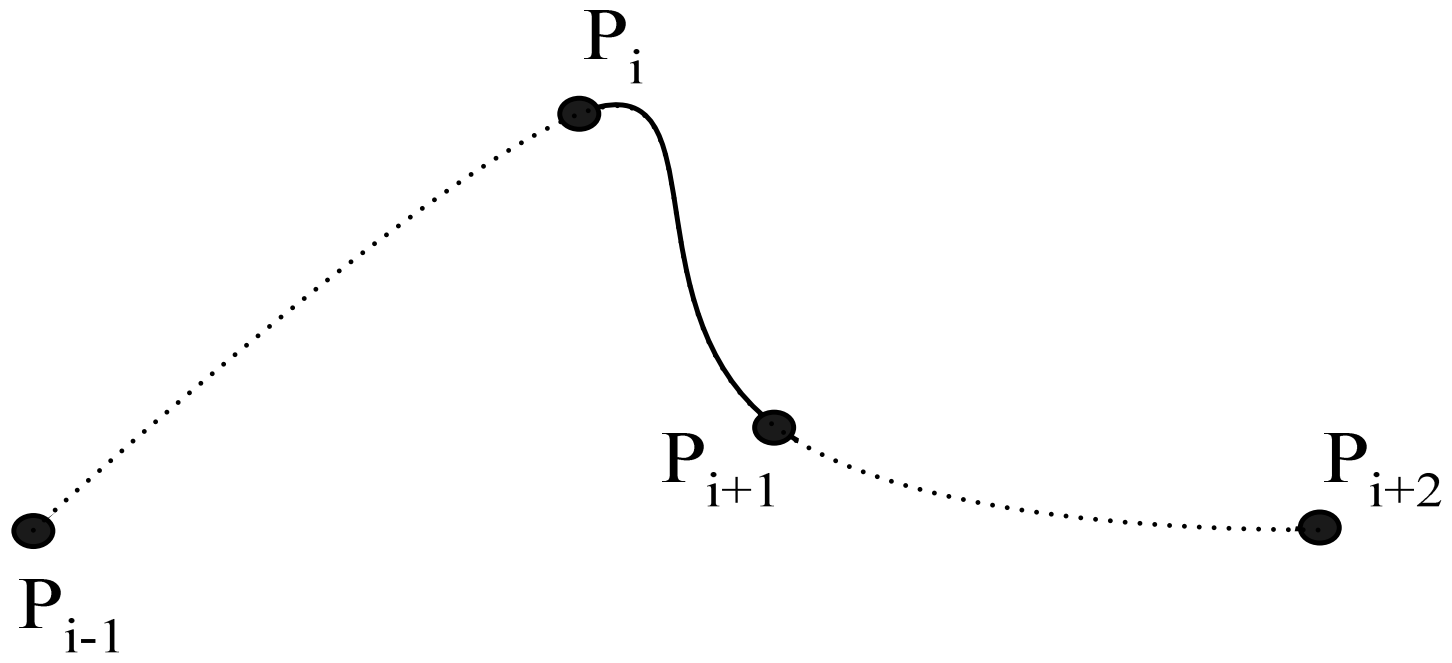
$P(0) \rightarrow P_i$, $P(1) \rightarrow P_{i+1}$

$$P(t) = T * F * \begin{bmatrix} P_i \\ P_{i+1} \\ \frac{P_{i+1} - P_{i-1}}{2} \\ \frac{P_{i+2} - P_i}{2} \end{bmatrix}$$

$$P(t) = T * F * \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1/2 & 0 & 1/2 & 0 \\ 0 & -1/2 & 0 & 1/2 \end{bmatrix} * \begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i+2} \end{bmatrix}$$

Matrični zapis Katmul-Romove krive

$$P(t) = [1 \quad t \quad t^2 \quad t^3] * \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1/2 & 0 & 1/2 & 0 \\ 1 & -5/2 & 2 & -1/2 \\ -1/2 & 3/2 & -3/2 & 1/2 \end{bmatrix} * \begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i+2} \end{bmatrix}$$

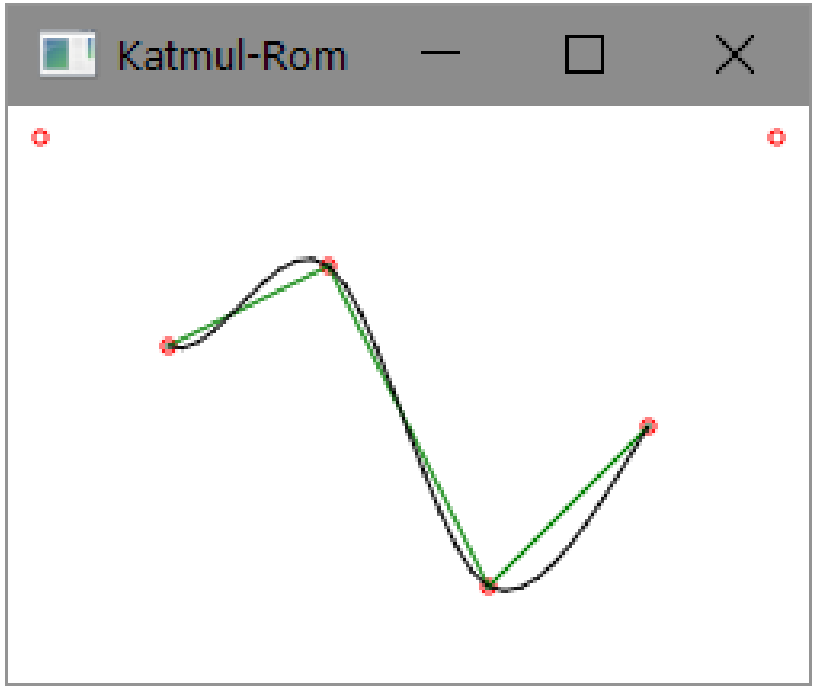


Katmul-Romova kriva - osobine

- Kriva je interpolativna:
  - prolazi kroz sve kontrolne tačke
- Kriva je lokalna:
  - promenom pozicije jedne kontrolne tačke lokalno se menja oblik krive
- Kriva je kontinualna (kontinualnost C_1)
- Kriva na segmentu je diferencijabilna u svakoj tački proizvoljan broj puta
- Kriva izlazi izvan kontrolnog mnogougla
- Kako se crta kriva na segmentima P_0-P_1 i P_{n-1}-P_n ?

Zadatak 2: CR kriva

- Nacrtati Katmul-Romovu krivu na kanvasu, na osnovu 6 kontrolnih tačaka



Rešenje: CR kriva (1/9)

```
package katmul_rom;

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.Group;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
```

Rešenje: CR kriva (2/9)

```
class Tacka {
    double x;
    double y;
    Tacka(){x=0;y=0;}
    Tacka(double x,double y){this.x=x; this.y=y;}

    static Tacka zbirT(Tacka p1, Tacka p2){
        return new Tacka(p1.x+p2.x,p1.y+p2.y);
    }

    static Tacka vXvT (double[] v, Tacka[] p){
        Tacka t=new Tacka();
        for (int i=0; i<v.length; i++)
            t=zbirT(t,new Tacka(v[i]*p[i].x,v[i]*p[i].y));
        return t;
    }
}
```

Rešenje: CR kriva (3/9)

```
class KatmulRom {
    double[][] KR= {{ 0.0, 1.0, 0.0, 0.0},
                    {-0.5, 0.0, 0.5, 0.0},
                    { 1.0,-2.5, 2.0,-0.5},
                    {-0.5, 1.5, -1.5,0.5}};
    double [] T4= new double [4];
    Tacka [] P4 = new Tacka[4];

    void postavi4Tacke(Tacka[] P){
        for (int i=0; i<P.length; i++) P4[i]=P[i];
    }
```

Rešenje: CR kriva (4/9)

```
    double [] vXm(double[] v, double[][] m) {
        double r[] = new double[m[0].length];
        for (int i=0; i<m[0].length; i++)
            for (int j=0; j<m.length; j++)
                r[i]+=v[j]*m[j][i];
        return r;
    }

    Tacka p(double t){
        for (int i=0; i<T4.length; i++) T4[i]=Math.pow(t, i);
        return Tacka.vXvT(vXm(T4,KR), P4);
    }
} // KatmulRom
```

Rešenje: CR kriva (5/9)

```
public class KatmulRomDemo extends Application {

    final double sirina = 250, visina = 180;
    final double[] xT = {10.0, 50.0, 100.0, 150.0, 200.0, 240.0};
    final double[] yT = {10.0, 75.0, 050.0, 150.0, 100.0, 010.0};
    Tacka [] kT = new Tacka[xT.length]; // kontrolne tačke

    void postaviKontrolneTacke(){
        for (int i=0; i<xT.length; i++)
          kT[i]=new Tacka(xT[i],yT[i]);
    }
```

Rešenje: CR kriva (6/9)

```
void nacrtajKontrolneTacke(GraphicsContext gk){
    gk.setStroke(Color.RED);
    for(int i=0; i<xT.length; i++)
      gk.strokeOval(xT[i]-2,yT[i]-2, 4, 4);
    gk.setStroke(Color.BLACK);
}
void crtajPoligon(int i, GraphicsContext gk){
    // i-ta ivica kontrolne polilinije
    double xi=kT[i].x;     double yi=kT[i].y;
    double xi1=kT[i+1].x;  double yi1=kT[i+1].y;
    gk.setStroke(Color.GREEN);
    gk.strokeLine(xi, yi, xi1, yi1);
    gk.setStroke(Color.BLACK);
}
```

Rešenje: CR kriva (7/9)

```
@Override
public void start(Stage prozor) {
  Canvas kanvas = new Canvas(sirina, visina);
  GraphicsContext gk = kanvas.getGraphicsContext2D();

  nacrtajKontrolneTacke(gk);
  postaviKontrolneTacke();

  KatmulRom kr = new KatmulRom();
```

Rešenje: CR kriva (8/9)

```
for (int i=1; i<kT.length-2; i++) {
   crtajPoligon(i, gk);
   Tacka[] P4 = new Tacka[4];
   for (int j=0; j<4; j++) P4[j]=kT[i-1+j];
   kr.postavi4Tacke(P4);

   double tx=kT[i].x, ty=kT[i].y;
   for (double t=0.1; t<1.0; t+=0.1) {
      double tx1=kr.p(t).x, ty1=kr.p(t).y;
      gk.strokeLine(tx, ty, tx1, ty1);
      tx=tx1; ty=ty1;
   }
 }
```

Rešenje: CR kriva (9/9)

```
        Group koren = new Group();
        koren.getChildren().add(kanvas);
        Scene scena = new Scene(koren, sirina, visina);
        prozor.setTitle("Katmul-Rom");
        prozor.setScene(scena);
        prozor.setResizable(false);
        prozor.show();
    }

    public static void main(String[] args) { launch(args); }
} // KatmulRomDemo
```