

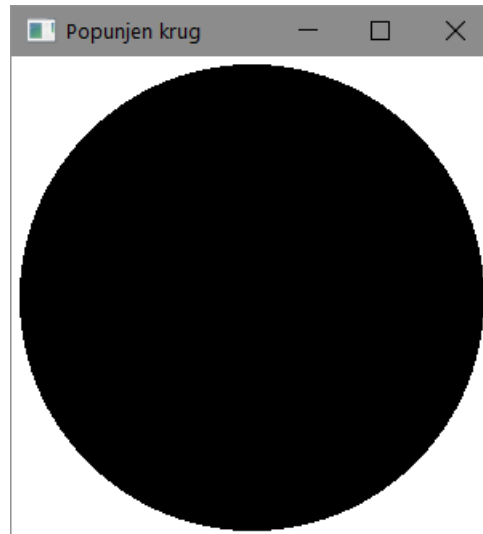
Računarska grafika - vežbe

14 – JavaFX rasterizacija

Zadatak 1: Popunjen krug

Nacrtati crni popunjen krug upisan u radnu površinu prozora, sa malom (5 piksela) marginom, bez upotrebe dodatnih struktura za pamćenje koordinata. Na raspolaganju je samo metod za crtanje pojedinačne tačke, gde su x i y koordinate tačke:

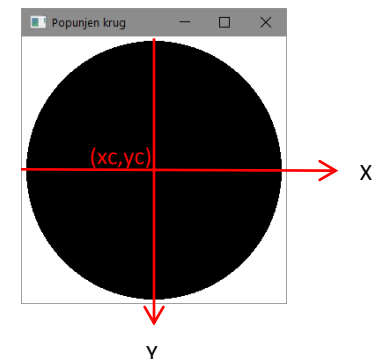
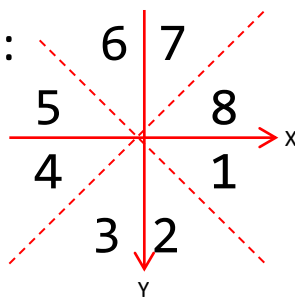
```
void tacka(int x, int y)
```



Rešenje: Popunjen krug (1/6)

- Krug se crta koristeći Bresenham-ov algoritam za kružnicu u 1. oktantu i ideju o 7 simetričnih tačaka izračunatoj tački (x,y) iz prvog oktanta
- Za svaku vrednost koordinate y treba nacrtati horizontalnu liniju u rasponu x koordinate $[x_c-x, x_c+x]$, gde su (x_c, y_c) koordinate centra
 - za jednu izračunatu tačku u 1. oktantu crtaju se 4 linije
 - tačka u 2. oktantu je simetrična tačka tački u 1. oktantu u odnosu na pravu $y=x$ (dobije se zamenu x i y koordinate)
 - tačka u 8. oktantu je simetrična tački u 1. oktantu u odnosu na pravu $y=0$ (dobije se promenom predznaka y)
 - tačka u 7. oktantu je simetrična tački u 8. oktantu u odnosu na pravu $y=-x$
- Levi koordinatni sistem

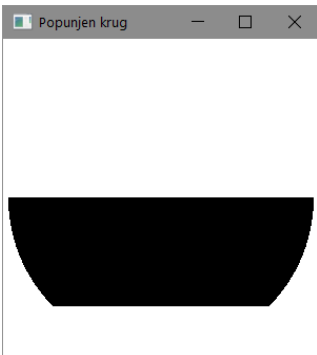
- Oktanti:



Rešenje: Popunjen krug (2/6)

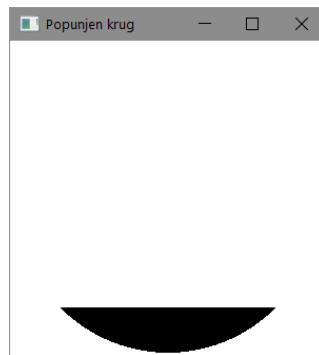
- Krug se dobija sintezom sledećih delova:
(za svaki deo su navedeni argumenti poziva funkcije za crtanje linije)

1. oktant



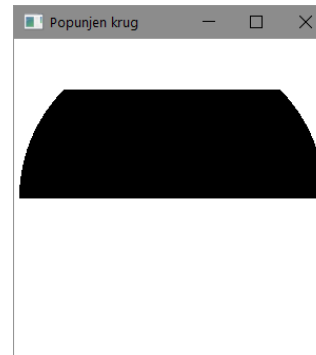
```
linija (xc-x, yc+y,  
        xc+x, yc+y);
```

2. oktant



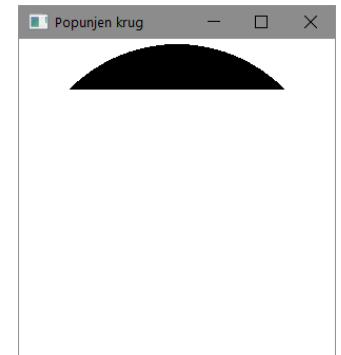
```
linija (yc-y, xc+x,  
        yc+y, xc+x);
```

8. oktant




```
linija (xc-x, yc-y,  
        xc+x, yc-y);
```

7. oktant



```
linija (yc-y, xc-x,  
        yc+y, xc-x);
```

Rešenje: Popunjen krug (3/6)



```
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.image.PixelWriter;
import javafx.scene.paint.Color;

public class PopunjenKrug extends Application {
    static final int SIRINA = 300, VISINA = SIRINA, MARGINA = 5;
    final Canvas kanvas = new Canvas(SIRINA, VISINA);
    final GraphicsContext gk = kanvas.getGraphicsContext2D();
    final PixelWriter pp = gk.getPixelWriter();
```


Rešenje: Popunjen krug (4/6)

```
void tacka(int x, int y){ pp.setColor(x, y, Color.BLACK); }
```

```
void linija(int x1, int y1, int x2, int y2) {  
    for (int x=x1; x<x2; x++) tacka(x,y1);  
};
```


```
void linije4 (int xc, int yc, int x, int y) {  
    linija(xc-x,yc+y,xc+x,yc+y); // 1. oktant  
    linija(yc-y,xc+x,yc+y,xc+x); // 2. oktant  
    linija(xc-x,yc-y,xc+x,yc-y); // 8. oktant  
    linija(yc-y,xc-x,yc+y,xc-x); // 7. oktant  
}
```

Rešenje: Popunjen krug (5/6)



```
void krugBresenham(int xc, int yc, int r) {  
    int x=r, y=0, d=3-2*r;  
    while(x>=y) {  
        linije4(xc,yc,x,y);  
        if( d < 0 ) d += 4*y+6;  
        else {  
            d += 4*(y-x)+10;  
            x--;  
        }  
        y++;  
    }  
}
```

Rešenje: Popunjen krug (6/6)



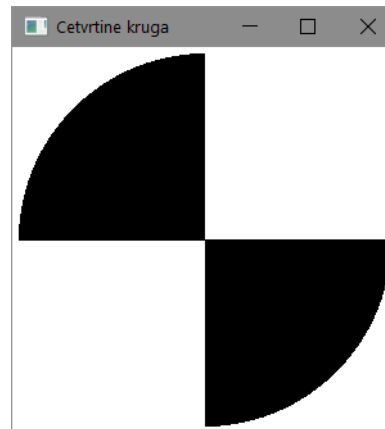
```
@Override public void start(Stage prozor) {
    Group koren = new Group();
    koren.getChildren().add(kanvas);
    final int R = Math.min(SIRINA, VISINA)/2 - MARGINA;
    krugBresenham (SIRINA/2, VISINA/2, R);
    Scene scena = new Scene(koren, SIRINA, VISINA);
    prozor.setTitle("Popunjen krug");
    prozor.setScene(scena);
    prozor.show();
}

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


Zadatak 2: Četvrtine kruga

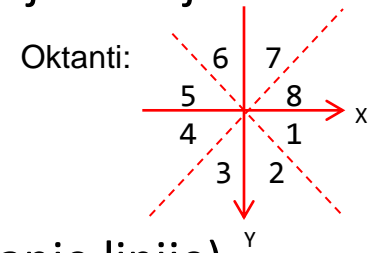
Nacrtati popunjene četvrtine kruga crnom bojom, kao na priloženoj slici, bez upotrebe dodatnih struktura za pamćenje koordinata. Krug je upisan u radnu površinu prozora, sa malom (5 piksela) marginom. Na raspolaganju je samo metod za crtanje pojedinačne tačke, gde su x i y koordinate tačke:

```
void tacka(int x, int y)
```



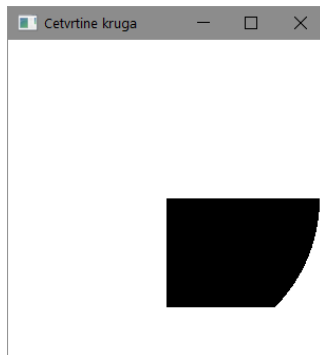
Rešenje: Četvrtine kruga (1/5)

- Za svaku izračunatu tačku (x,y) na kružnici u 1. oktantu crtaju 4 linije i to:
 - za tačke u 1. i 2. oktantu u opsegu $x=[x_c, x_c+x]$
 - za tačke u 5. i 6. oktantu u opsegu $x=[x_c-x, x_c]$



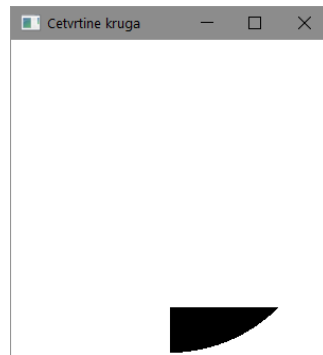
- Krug se dobija sintezom sledećih delova:
(za svaki deo su navedeni argumenti poziva funkcije za crtanje linije)

1. Oktant



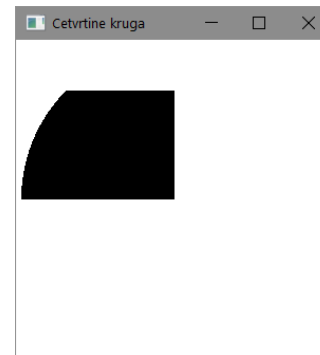
```
linija(xc, yc+y,  
       xc+x, yc+y);
```

2. oktant



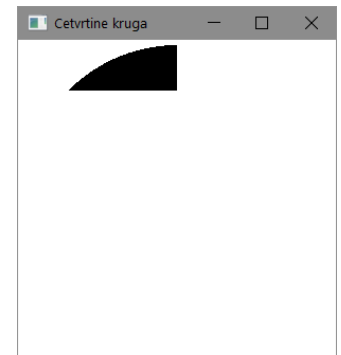
```
linija(yc, xc+x,  
       yc+y, xc+x);
```

5. oktant




```
linija(xc-x, yc-y,  
       xc, yc-y)
```

6. oktant



```
linija(yc-y, xc-x,  
       yc, xc-x);
```

Rešenje: Četvrtine kruga (2/5)




```
...
public class CetvrtineKrug extends Application {
    static final int SIRINA = 300, VISINA = SIRINA, MARGINA = 5;
    final Canvas kanvas = new Canvas(SIRINA, VISINA);
    final GraphicsContext gk = kanvas.getGraphicsContext2D();
    final PixelWriter pp = gk.getPixelWriter();

    void tacka(int x, int y) { pp.setColor(x, y, Color.BLACK); }

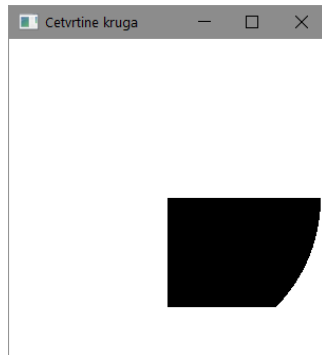
    void linija(int x1, int y1, int x2, int y2) {
        for (int x=x1; x<x2; x++) tacka(x,y1);
    };
};
```

Rešenje: Četvrtine kruga (3/5)

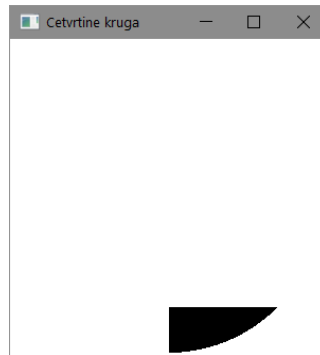


```
void linije4 (int xc, int yc, int x, int y) {  
    linija(xc,yc+y,xc+x,yc+y); // 1. oktant  
    linija(yc,xc+x,yc+y,xc+x); // 2. oktant  
    linija(xc-x,yc-y,xc,yc-y); // 5. oktant  
    linija(yc-y,xc-x,yc,xc-x); // 6. oktant  
}
```

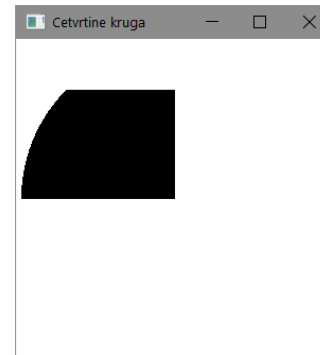
1. Oktant



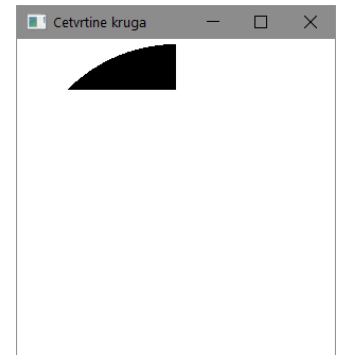
2. oktant




5. oktant



6. oktant




Rešenje: Četvrtine kruga (4/5)



```
void krugBresenham(int xc, int yc, int r) {
    int x=r, y=0, d=3-2*r;
    while(x>=y) {
        linije4(xc,yc,x,y);
        if( d < 0 ) d += 4*y+6;
        else {
            d += 4*(y-x)+10;
            x--;
        }
        y++;
    }
}
```

Rešenje: Četvrtine kruga (5/5)



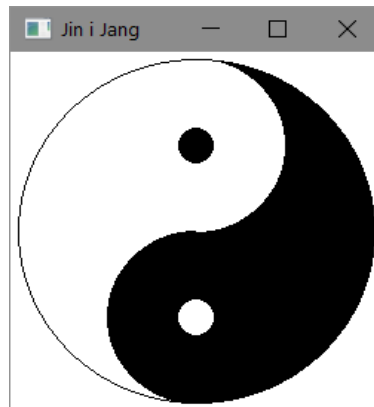
```
@Override public void start(Stage prozor) {
    Group koren = new Group();
    koren.getChildren().add(kanvas);
    final int R = Math.min(SIRINA, VISINA)/2 - MARGINA;
    krugBresenham(SIRINA/2, VISINA/2, R);
    Scene scena = new Scene(koren, SIRINA, VISINA);
    prozor.setTitle("Cetvrtine kruga");
    prozor.setScene(scena);
    prozor.show();
}

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

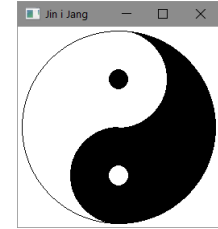
Zadatak 3: Jin i Jang

Nacrtati Jin i Jang (Taijitu) simbol sa slike, bez dodatnih struktura za pamćenje koordinata. Krug je upisan u radnu površinu prozora, sa malom (5 piksela) marginom. Na raspolaganju je samo metod za crtanje pojedinačne tačke, gde su x i y koordinate tačke, a b boja koja može da bude samo crno (`Color.BLACK`) ili belo (`Color.WHITE`):

```
void tacka(int x, int y, Color b)
```



Rešenje: Jin i Jang (1/8)




- Simbol se može dekomponovati na sledeće delove, koji se crtaju sledećim redosledom:
 - crni desni polukrug sa centrom u (x_c, y_c) i poluprečnikom R
 - beli desni gornji polukrug sa centrom u $(x_c, y_c - R/2)$ i poluprečnikom $R/2$
 - crni levi donji polukrug sa centrom u $(x_c, y_c + R/2)$ i poluprečnikom $R/2$
 - crni gornji kružić sa centrom u $(x_c, y_c - R/2)$ i poluprečnikom $R/10$
 - beli donji kružić sa centrom u $(x_c, y_c + R/2)$ i poluprečnikom $R/10$
 - crna kružnica sa centrom u (x_c, y_c) i poluprečnikom R
- Napomena: umesto kružnice bi mogla da se crta leva polukružnica, ali bi beli desni gornji polukrug išao do ivice crnog desnog polukruga i prekrio par piksela na gornjoj ivici za $x > x_c$
 - zato se na kraju crta cela kružnica sa centrom u (x_c, y_c) i poluprečnikom R

Rešenje: Jin i Jang (2/8)

- Za svaki od od navedenih delova poziva se metod za rasterizaciju kruznice
- Za svaku izračunatu tačku u prvom oktantu ($\text{ugao} < 45^\circ$) poziva se odgovarajući metod za crtanje polukruga, kruga ili kružnice
- Metod za crtanje kružnice neposredno poziva metod za crtanje tačke i koristi ideju simetrije 7 tačaka u odnosu na izračunatu tačku (*Michener*)
- Metod za popunjeni polukrug ili kružić koristi metod za crtanje linije
 - koji koristi metod za crtanje tačke


Rešenje: Jin i Jang (3/8)



```
...
public class JinJang extends Application {
    static final int SIRINA = 250, VISINA = SIRINA, MARGINA = 5;
    final Canvas kanvas = new Canvas(SIRINA, VISINA);
    final GraphicsContext gk = kanvas.getGraphicsContext2D();
    final PixelWriter pp = gk.getPixelWriter();
    enum DeoKruga { KRUZNICA, DESNI_VELIKI_POLUKRUG,
                    DESNI_MALI_POLUKRUG, LEVI_MALI_POLUKRUG,
                    GORNJI_MALI_KRUG, DONJI_MALI_KRUG }
    void tacka(int x, int y, Color b) { pp.setColor(x, y, b); }


    void linija(int x1, int y1, int x2, int y2, Color b) {
        for (int x=x1; x<x2; x++) tacka(x, y1, b);
    }
}
```

Rešenje: Jin i Jang (4/8)



```
void kruznica(int xc, int yc, int x, int y, Color b) {  
    pp.setColor(xc+x, yc+y, b); // 1.oktant  
    pp.setColor(xc+y, yc+x, b); // 2.oktant  
    pp.setColor(xc-y, yc+x, b); // 3. oktant  
    pp.setColor(xc-x, yc+y, b); // 4. oktant  
    pp.setColor(xc-x, yc-y, b); // 5. oktant  
    pp.setColor(xc-y, yc-x, b); // 6. oktant  
    pp.setColor(xc+y, yc-x, b); // 7. oktant  
    pp.setColor(xc+x, yc-y, b); // 8. oktant  
}
```


Rešenje: Jin i Jang (5/8)



```
void desniPolukrug(int xc, int yc, int x, int y, Color b){  
    linija(xc,yc+y,xc+x,yc+y,b); // 1. oktant  
    linija(xc,yc+x,xc+y,yc+x,b); // 2. oktant  
    linija(xc,yc-y,xc+x,yc-y,b); // 8. oktant  
    linija(xc,yc-x,xc+y,yc-x,b); // 7. oktant  
}
```

```
void leviPolukrug(int xc, int yc, int x, int y, Color b){  
    linija(xc-x,yc+y,xc,yc+y,b); // 4. oktant  
    linija(xc-y,yc+x,xc,yc+x,b); // 3. oktant  
    linija(xc-x,yc-y,xc,yc-y,b); // 5. oktant  
    linija(xc-y,yc-x,xc,yc-x,b); // 6. oktant  
}
```


Rešenje: Jin i Jang (6/8)



```
void maliKrug(int xc, int yc, int x, int y, Color b) {  
    linija(xc-x,yc+y,xc+x,yc+y,b); // 1. i 4. oktant  
    linija(xc-y,yc+x,xc+y,yc+x,b); // 2. i 3. oktant  
    linija(xc-x,yc-y,xc+x,yc-y,b); // 5. i 8. oktant  
    linija(xc-y,yc-x,xc+y,yc-x,b); // 6. i 7. oktant  
}
```


```
void figura(int xc, int yc, int R) {  
    krugBresenham(xc,yc,R,DeoKruga.DESNI_VELIKI_POLUKRUG,Color.BLACK);  
    krugBresenham(xc,yc-R/2,R/2,DeoKruga.DESNI_MALI_POLUKRUG,Color.WHITE);  
    krugBresenham(xc,yc+R/2,R/2,DeoKruga.LEVI_MALI_POLUKRUG,Color.BLACK);  
    krugBresenham(xc,yc-R/2,R/10,DeoKruga.GORNJI_MALI_KRUG,Color.BLACK);  
    krugBresenham(xc,yc+R/2,R/10,DeoKruga.DONJI_MALI_KRUG,Color.WHITE);  
    krugBresenham(xc,yc,R,DeoKruga.KRUZNICA,Color.BLACK);  
}
```

Rešenje: Jin i Jang (7/8)



```
void krugBresenham(int xc, int yc, int r, DeoKrug a deo, Color b) {
    int x=r, y=0, d=3-2*r;
    while(x>=y) {
        switch (deo) {
            case DESNI_VELIKI_POLUKRUG: desniPolukrug(xc,yc,x,y,b); break;
            case DESNI_MALI_POLUKRUG: desniPolukrug(xc,yc,x,y,b); break;
            case LEVI_MALI_POLUKRUG: leviPolukrug(xc,yc,x,y,b); break;
            case GORNJI_MALI_KRUG: maliKrug(xc,yc,x,y,b); break;
            case DONJI_MALI_KRUG: maliKrug(xc,yc,x,y,b); break;
            case KRUIZNICA: kruznica(xc,yc,x,y,b); break;
        }
        if( d < 0 ) d += 4*y+6;
        else { d += 4*(y-x)+10; x--; }
        y++;
    }
}
```

Rešenje: Jin i Jang (8/8)



```
@Override public void start(Stage prozor) {
    Group koren = new Group();
    koren.getChildren().add(kanvas);
    final int R = Math.min(SIRINA, VISINA)/2 - MARGINA;
    figura(SIRINA/2, VISINA/2, R);
    Scene scena = new Scene(koren, SIRINA, VISINA);
    prozor.setTitle("Jin i Jang");
    prozor.setScene(scena);
    prozor.show();
}

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