UNIVERSITETI I PRISHTINËS

DEPARTAMENTI I MATEMATIKËS

Dega – Shkenca Kompjuterike

Programim Dhe Algoritme

Punim seminarik

Studentët / et :

Festina Klinaku

Altin Gjonbalaj

Prishtinë, 2020

Kërkesat e parashtruara për punimin seminarik

1. Lokacioni në GitHub
2. Problemi që duhet zgjidhur (Problemi i detyrës)
3. Si e keni zgjidhur?
4. Arkitektura e aplikacionit
5. Pjesa kryesore (E shkruar në trajtë të kodit dhe e spjeguar, vetëm pjesa kryesore)
6. Reflektimin (Çfarë do të ndryshonit nëse do ta krijonit edhe njëherë nga fillimi aplikacionin e parashtruar).

1. Lokacioni në GitHub

https://github.com/altingjonbalaj18/Programimi-dhe-Algoritmet/tree/Bachelor

2. Problemi që duhet zgjidhur

Detyra 6. 11. 5

Një program i cili shfaq distancën e kryer nga një automjet, pasi që përdoruesi i jep informatat hyrëse, shpejtësinë fillestare dhe nxitimin me anë të një kornize dialogu.

Pasi që shfrytëzuesi jep informatat hyrëse, aplikacioni duhet të shfaq distancën e kryer për **t** sekonda pasi që përdoruesi shtyp një buton i cili inkrementon kohën për një sekond njësi dhe i cili shfaq distancën aktuale të automjetit.

3. Si e keni zgjedhur?

Informatat përdoruesi duhet të i japë përmes një driatare dialogjesh(JFrame.showInputDialog()).

Këto të dhëna ruhen, dhe më pas shfrytëzohen nga një metodë publike e cila kthen vlera primitive të tipit double.

Klasa AutoOutputFrame, krijon dritaren **Auto** në të cilën shfaqen rezultatet e fituara gjatë kohës së përdorimit të aplikacionit. Dritarja ka madhësinë 400x400 pixel. Në të janë vendosur dy butona, **Next Second** dhe **Quit.**

Butoni Next Second, inkrementon kohën për një sekond dhe thirr metodën **repaint()**.

Butoni Quit, mbyll aplikacionin përmes **System.exit(0)**.

Metoda paintComponent(Graphics g), përdor pendën grafike dhe vizaton në dritaren Auto. Për të vizatuar në dritaren Auto kemi përdorur një **switch**, dhe varësisht se sa është vlera aktuale e kohës kjo metodë do të afishojë në dritaren Auto stringjet e paracaktuara nga secili bllok i urdhërave brenda **switch**-it.

Kjo metodë do të ekzekutohet sa herë që metoda **repaint()** përdoret si urdhër i një blloku. Pra, sa herë që përdoruesi shtyp butonin **Next Second** kjo metodë do të thirret.

4. Arkitektura e aplikacionit

Për ndërtimin e këtij aplikacioni kemi përdorur arkitekturë **MVC**(Model – View – Controller).

Ndërtohet nga tri klasa:

- **Auto**(Controller),
- **AutoInputFrame**(Model dhe InputView) dhe
- **AutoOutputFrame**(OutputView).

Vërejtje: Klasa AutoInputFrame është Model dhe InputView për arsye sepse përmban metodën returnDistance (int time) dhe njëkohësisht merr të dhëna nga shfrytëzuesi. Arsyeja pse ne kemi vendosur që kjo klasë të shërbejë si të dyjat njëkohësisht është për thjeshtim kodi dhe që të mos ketë ndërthurje të mbingarkuar të klasave.

Klasa Auto bën instancimin e klasës AutoOutputFrame.

Klasa AutoOutputFrame së pari bën instancimin e klasës AutoInputFrame dhe ekzekuton konstruktorin e saj AutoInputFrame(). Konstruktori AutoInputFrame() ekzekuton një bllok urdhërash të cilët marrin informata nga përdoruesi dhe i ruajnë brenda klasës.

Kjo klasë ka një metodë e cila kthen vlerë primitive të tipit double. Kjo metodë shfrytëzohet nga klasa AutoOutputFrame për llogaritje të distancës.

Pasi ekzekutohet konstuktori i klasës AutoInputFrame në terësi, fillon ekzekutimi i konstruktorit të klasës AutoOutputFrame. Konstruktori AutoOutputFrame cakton madhësinë e dritares, emrin e saj dhe njëkohësisht përdor disa metoda të cilat krijojnë butona në dritaren e krijuar.

Këtyre butonave iu shtohet funksioni i tyre përmes ActionListener dhe ActionEvent. Këto klasë janë klasë abstrakte, pra nuk mund të instancohen por duhet të përdoren brenda metodave, njësoj sikur klasa Graphics.

Rreshtat afishohen në dritare përmes metodës paintComponent(Graphics g), e cila përdor pendën grafike për të vizatuar në dritare. Kjo metodë rithirret përmes metodës repaint(), e cila është urdhër brenda butonit Next Second i krijuar brenda dritares.

5. Pjesa Kryesore

- Klasa Auto

```
public class Auto{
  public static void main(String[] a){
    AutoOutputFrame output = new AutoOutputFrame();
  }
}
```

Klasa Auto bën instancimin e klasës AutoOutputFrame dhe thirr konstruktorin e saj.

- Klasa AutoOutputFrame

Klasa AutoOutputFrame trashëgon klasën Jpanel dhe implementon klasën ActionListener. ActionListener është klasë abstrakte dhe duhet të implementohet dhe të mbishkruhet me anë të metodave që ajo përmban.

```
public class AutoOutputFrame extends JPanel implements ActionListener {

//Instancimi i klasës AutoInputFrame dhe thirret Konstruktori i saj.

//Klasa AutoInputFrame është instancuar si “objekt” i fushës pasi që do të përdoret edhe në metodat e mëposhtme. Në momentin e instancimit thirret konstruktori i saj.
  AutoInputFrame input = new AutoInputFrame();
  public static int time;
  private double distance;

// Krijimi i butonave Next Seconds dhe Quit
  JButton nextSecond = new JButton("Next Second");
  JButton quit = new JButton("Quit");
  //Konstruktori AutoOutputFrame
  public AutoOutputFrame() {
    //Krijimi i dritares
    JFrame frame = new JFrame("Auto");
```

```
 //Pozicioni  dhe madhësia e butonit
    nextSecond.setBounds(0,0,400,30);

    quit.setBounds(0,30,400,30);
    //Caktimi i funksionit të butonit
    nextSecond.addActionListener(new ActionListener() {
        //Funksioni i butonit
        public void actionPerformed(ActionEvent e) {

 //Butoni inkrementon kohën për një njësi dhe thirr metodën paintComponent(Graphics g).
            time++;
            repaint();
        }
      });

    quit.addActionListener(new ActionListener() {

        public void actionPerformed(ActionEvent e) {

//Butoni Quit mbyll aplikacionin.
            System.exit(0);
        }
      });
    //Shtimi i butonave në dritare (kornizë)
    frame.add(nextSecond);
    frame.add(quit);
    //Madhësia e dritares
    frame.setSize(400, 400);
    frame.getContentPane().add(this);
    frame.setLocationRelativeTo(null);
    frame.setVisible(true);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  }
  //Mbishkrimi (Override) i klasës ActionListener pasi që klasa jonë e implementon. Klasa
ActionListener është klasë abstrakte.
  @Override
  public void actionPerformed(ActionEvent a) {
  }
```

```
//Metoda paintComponent(Graphics g), vizaton në dritaren grafike.
  public void paintComponent(Graphics g) {
    g.setColor(Color.black);

   //Switch Statement tek i cili blloqet e urdhërave janë të bazuara varësisht nga ndryshorja time.
    switch(time) {

        //Nëse time është 1
      case 1 : Urdhërat për afishim në dritare ...

               break;

      ...

      ...

        //Nëse time është 14

      case 14 : Urdhërat për afishim në dritare ...

               break;
    }
  }
}
```

- Klasa AutoInputFrame

```
public class AutoInputFrame{
  public static double initialVelocity;
  public static double accerelation;
  public AutoInputFrame(){
    //Përderisa initialVelocity është <=0, përsërit.
    //Pasi që initialVelocity është 0 (Vlera default), programi hyn brenda lakut(loop).
    while(initialVelocity <= 0){
      String input1 = JOptionPane.showInputDialog("What's the Initial Velocity");
      //Nëse përdoruesi shtyp Cancel ose Close.
      if(input1 == null){
        System.out.println("Application closed, for later use restart application.");
        System.exit(0);
      }
      else
        initialVelocity = new Double(input1).doubleValue();
    }
```

```
    //Përderisa accerelation është <=0, përsërit.
    //Pasi që accerelation është 0 (Vlera default), programi hyn brenda lakut(loop).
    while(accerelation <= 0){
      String input2 = JOptionPane.showInputDialog("What's the Accerelation");
      //Nëse përdoruesi shtyp Cancel ose Close.
      if(input2 == null){
        System.out.println("Application closed, for later use restart application.");
        System.exit(0);
      }
      else
        accerelation = new Double(input2).doubleValue();
    }
  }
  //Metoda returnDistance kthen distancën aktuale në kohën time.
  public double returnDistance(int time){

    // s = v0 * t + (1.0/2.0)  *  a  *  t^2 ose s = t * (v0 + (1.0/2.0) * a * t).
    double distance = time * (initialVelocity + (1.0/2.0) * accerelation * time);
    return distance;
  }
}
```

6. Reflektimi

Aplikacioni ynë mund ti parashtrohet ndryshimeve të shumta si brenda kodit, ashtu edhe nga pamja e dritares.

Njëra ndër ndryshimet e shumta do të ishte, krijimi i disa butonave të rinj. Butoni Start Over, do ta ristartonte aplikacionin në terësi ose do të kthente vlerat fillestare të aplikacionit. Butoni Stop do të bënte të pa mundshme klikimin e butonit Next Second.

Të dhënat do të ishte e mundshme që të mirreshin përmes JText, pra brenda aplikacionit dhe njëra ndër avancimet e mundshme do të ishte Real-Time Informations (Informatat do të ndërroheshin vazhdimisht kurdoherë që të jetë nevoja).

Insertimi i ndonjë imazhi, do ta bënte paksa më të improvizuar distancën e kryer nga automjeti.