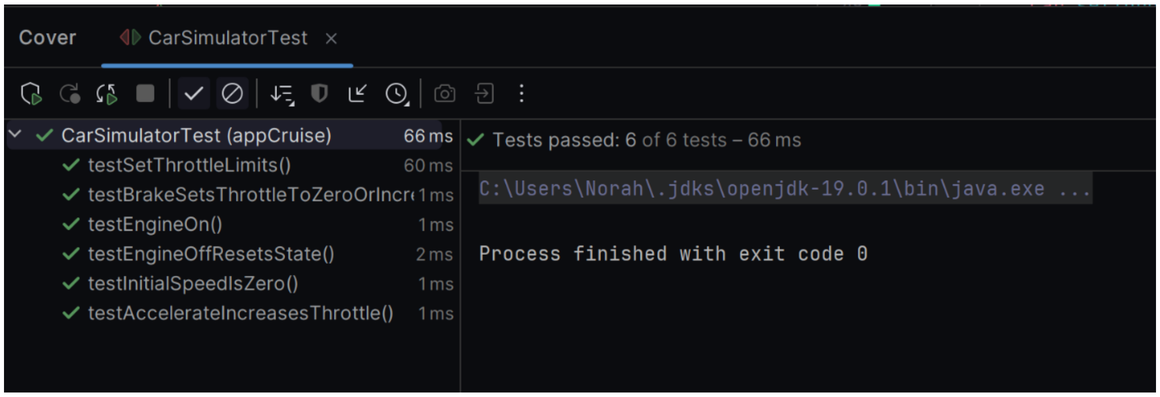
SWE326 Project - Phase 3: Unit Testing - Step 3-3

This document provides screenshots and analysis of unit test execution results and code coverage, as required for Step 3-3. IntelliJ IDEA's built-in coverage tool is used.

CarSimulator.java

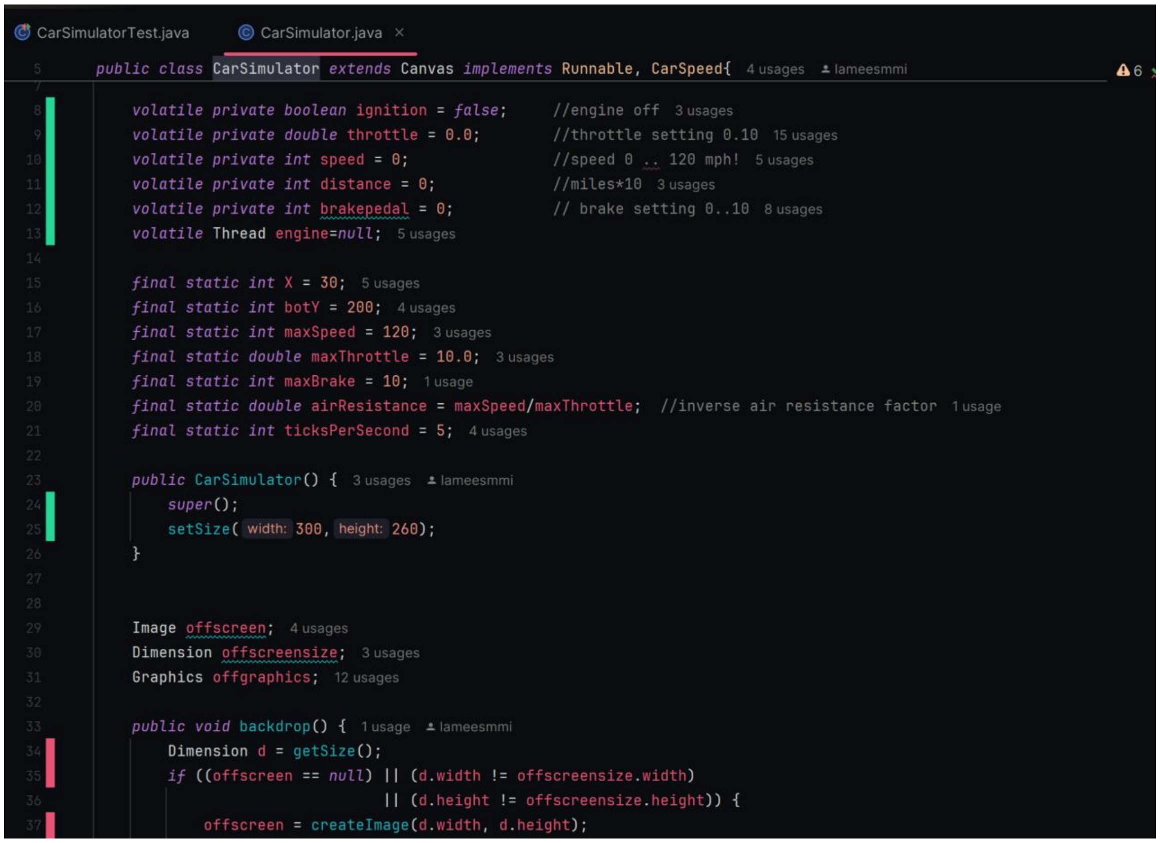
Test Results:



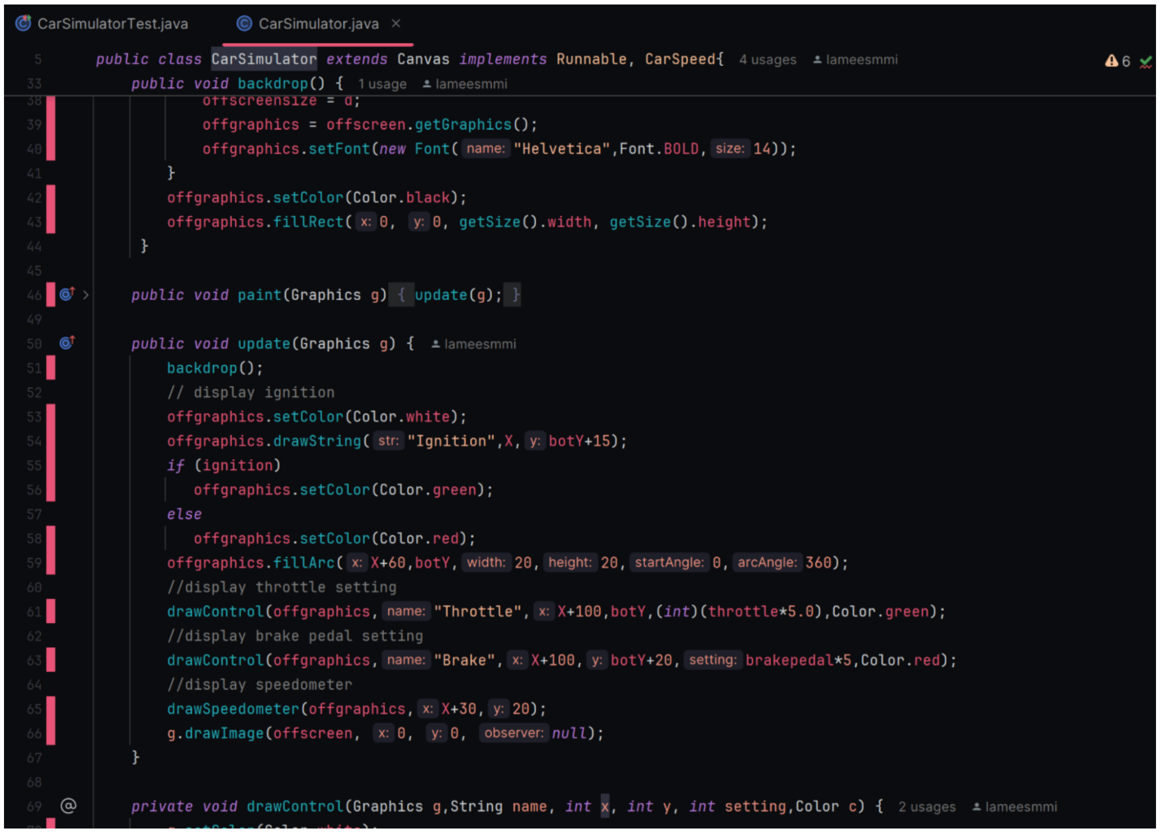
```
CarSimulatorTest (appCruise)                66 ms
  testSetThrottleLimits()                   60 ms
  testBrakeSetsThrottleToZeroOrIncr         1 ms
  testEngineOn()                            1 ms
  testEngineOffResetsState()                2 ms
  testInitialSpeedIsZero()                  1 ms
  testAccelerateIncreasesThrottle()         1 ms
```

```
Tests passed: 6 of 6 tests – 66 ms

C:\Users\Norah\.jdks\openjdk-19.0.1\bin\java.exe ...

Process finished with exit code 0
```

Code Coverage View:

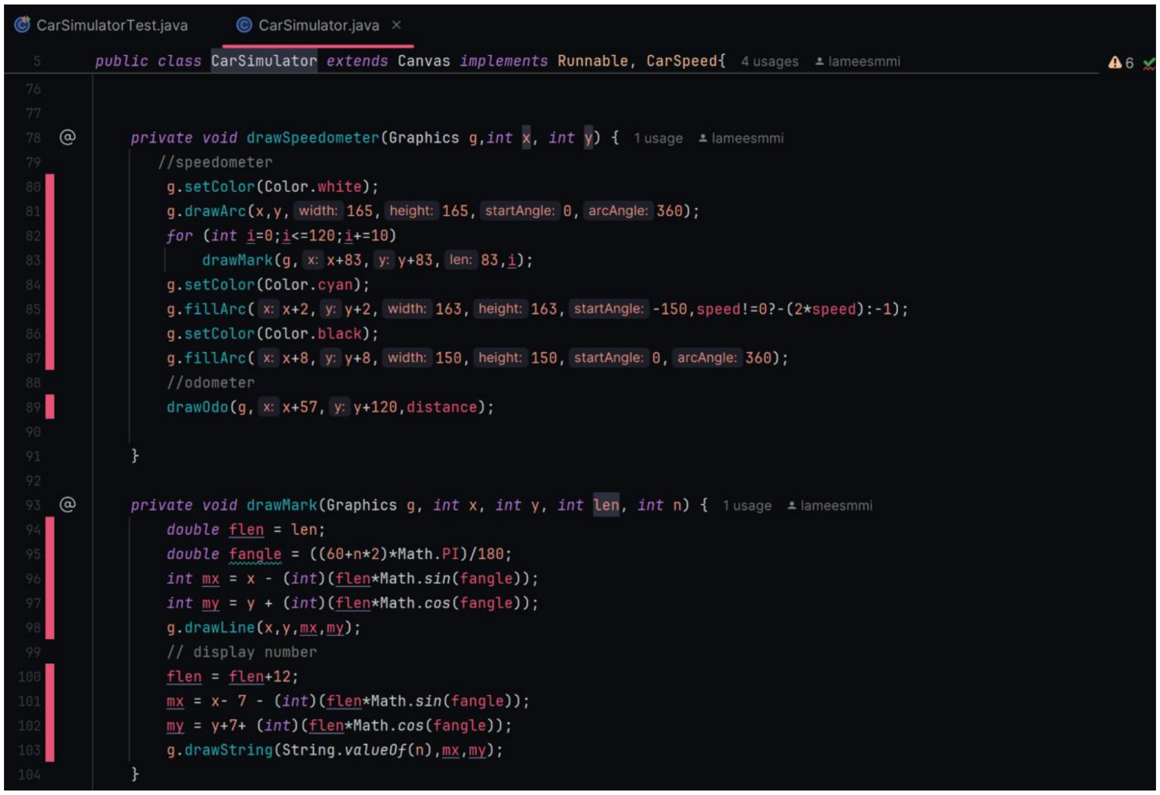


```
public class CarSimulator extends Canvas implements Runnable, CarSpeed{  4 usages  lameesmmi

    volatile private boolean ignition = false;     //engine off  3 usages
    volatile private double throttle = 0.0;        //throttle setting 0.10  15 usages
    volatile private int speed = 0;                //speed 0 .. 120 mph!  5 usages
    volatile private int distance = 0;             //miles*10  3 usages
    volatile private int brakepedal = 0;           // brake setting 0..10  8 usages
    volatile Thread engine=null;  5 usages

    final static int X = 30;  5 usages
    final static int botY = 200;  4 usages
    final static int maxSpeed = 120;  3 usages
    final static double maxThrottle = 10.0;  3 usages
    final static int maxBrake = 10;  1 usage
    final static double airResistance = maxSpeed/maxThrottle;  //inverse air resistance factor  1 usage
    final static int ticksPerSecond = 5;  4 usages

    public CarSimulator() {  3 usages  lameesmmi
        super();
        setSize( width: 300, height: 260);
    }


    Image offscreen;  4 usages
    Dimension offscreensize;  3 usages
    Graphics offgraphics;  12 usages

    public void backdrop() {  1 usage  lameesmmi
        Dimension d = getSize();
        if ((offscreen == null) || (d.width != offscreensize.width)
                                || (d.height != offscreensize.height)) {
            offscreen = createImage(d.width, d.height);
```



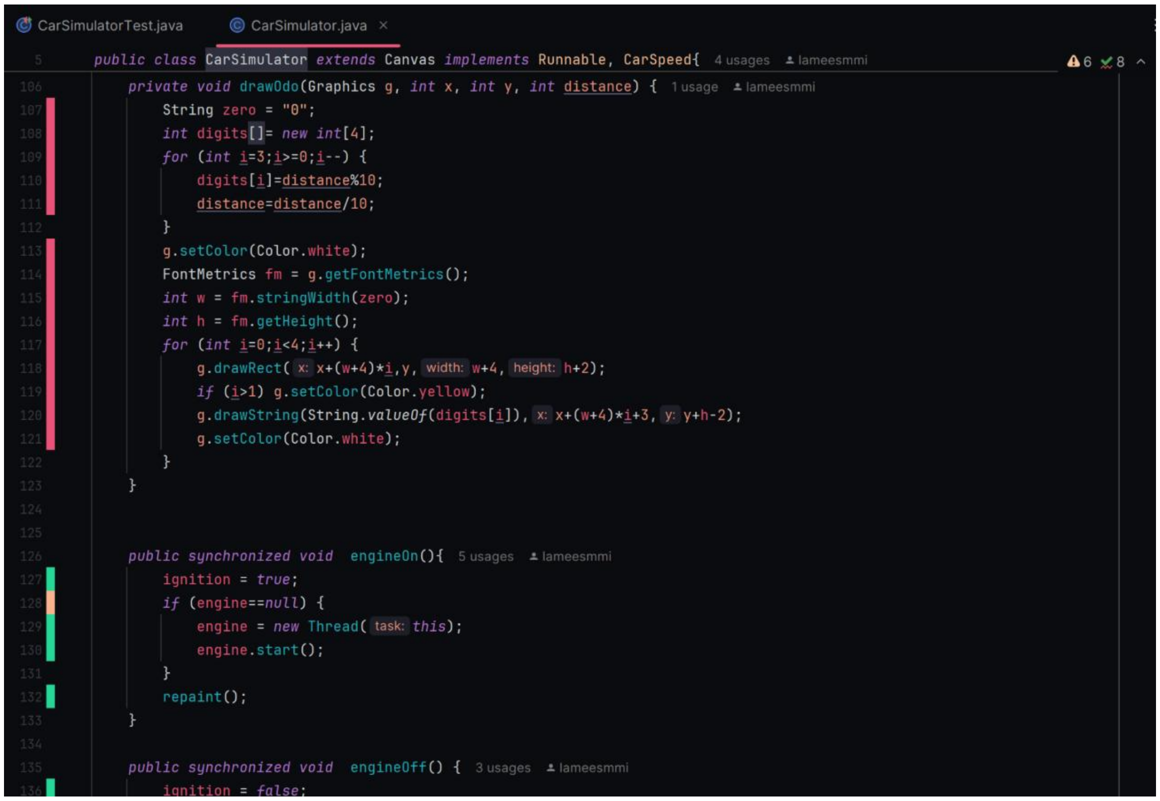
CarSimulatorTest.java   CarSimulator.java ×

```
public class CarSimulator extends Canvas implements Runnable, CarSpeed{  4 usages  lameesmmi
    public void backdrop() {  1 usage  lameesmmi
            offscreensize = d;
            offgraphics = offscreen.getGraphics();
            offgraphics.setFont(new Font( name: "Helvetica",Font.BOLD, size: 14));
        }
        offgraphics.setColor(Color.black);
        offgraphics.fillRect( x: 0,  y: 0, getSize().width, getSize().height);
    }

    public void paint(Graphics g) { update(g); }

    public void update(Graphics g) {  lameesmmi
        backdrop();
        // display ignition
        offgraphics.setColor(Color.white);
        offgraphics.drawString( str: "Ignition",X, y: botY+15);
        if (ignition)
            offgraphics.setColor(Color.green);
        else
            offgraphics.setColor(Color.red);
        offgraphics.fillArc( x: X+60,botY, width: 20, height: 20, startAngle: 0, arcAngle: 360);
        //display throttle setting
        drawControl(offgraphics, name: "Throttle", x: X+100,botY,(int)(throttle*5.0),Color.green);
        //display brake pedal setting
        drawControl(offgraphics, name: "Brake", x: X+100, y: botY+20, setting: brakepedal*5,Color.red);
        //display speedometer
        drawSpeedometer(offgraphics, x: X+30, y: 20);
        g.drawImage(offscreen,  x: 0,  y: 0,  observer: null);
    }

    private void drawControl(Graphics g,String name, int x, int y, int setting,Color c) {  2 usages  lameesmmi
```

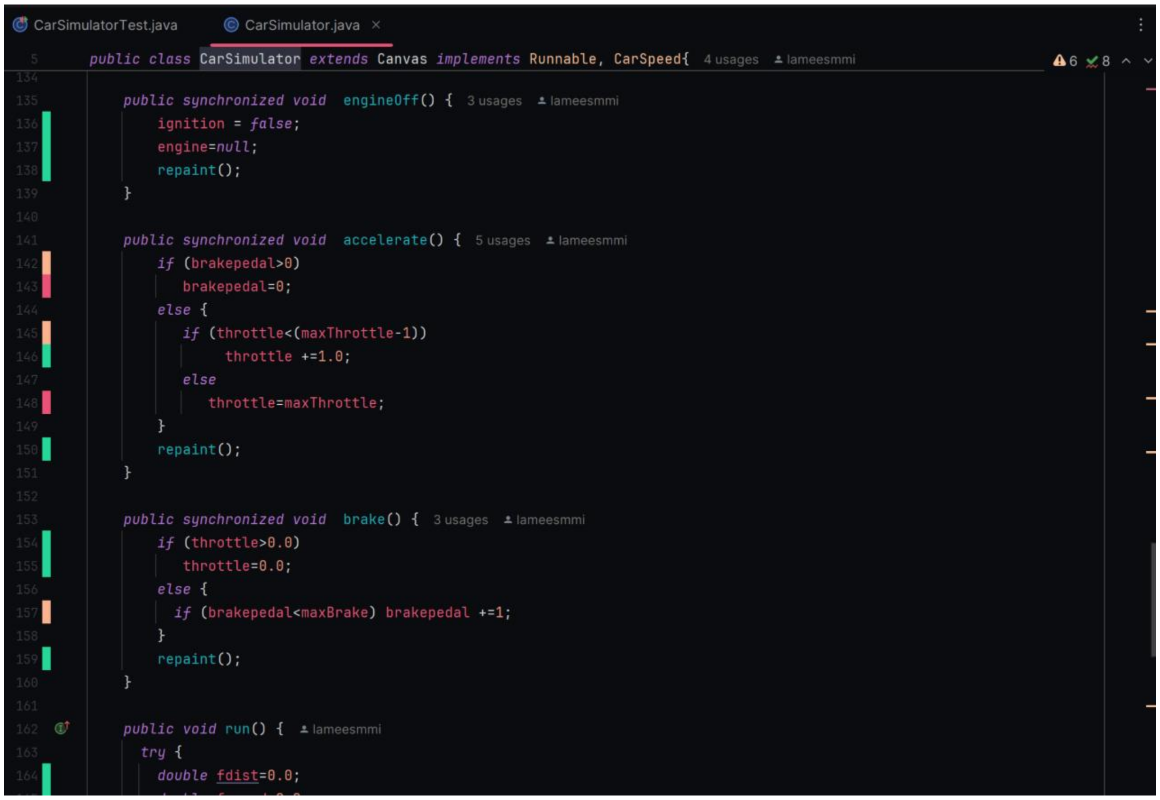


CarSimulatorTest.java   CarSimulator.java ×

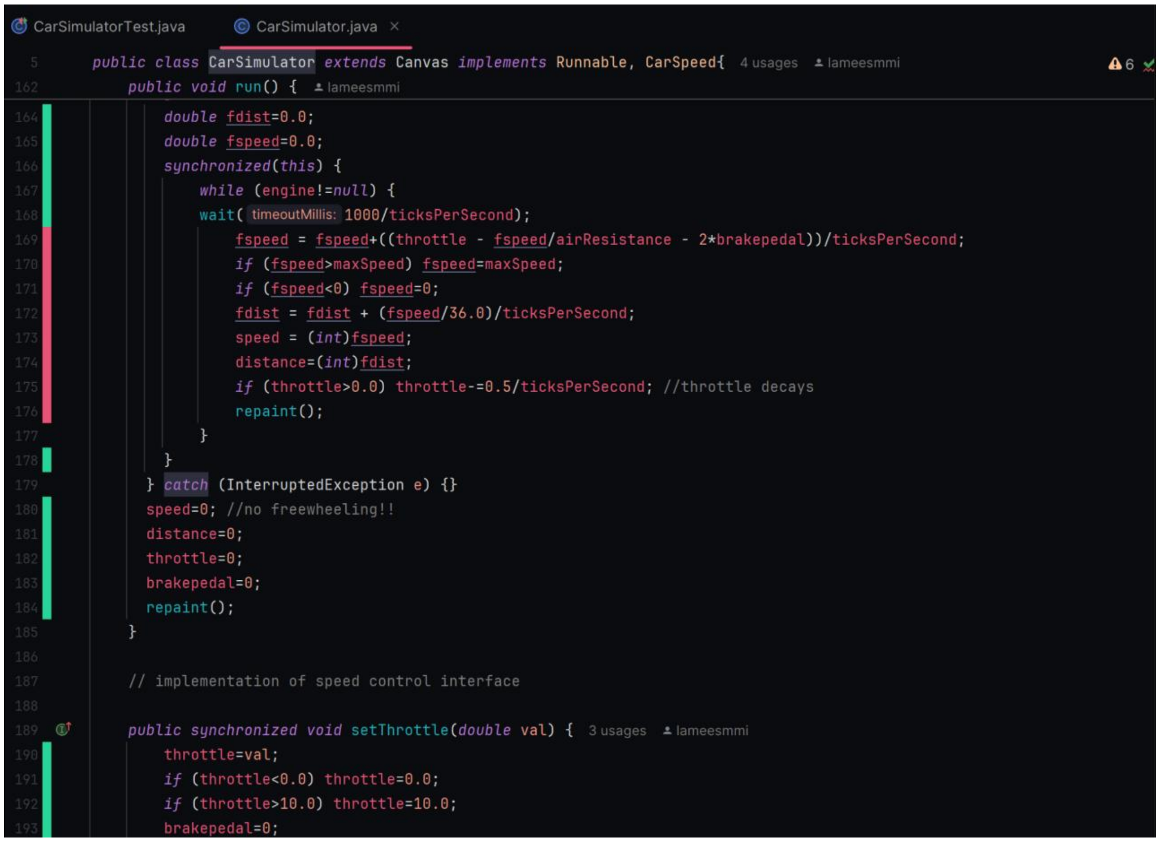
```
public class CarSimulator extends Canvas implements Runnable, CarSpeed{  4 usages  lameesmmi


    private void drawSpeedometer(Graphics g,int x, int y) {  1 usage  lameesmmi
       //speedometer
        g.setColor(Color.white);
        g.drawArc(x,y, width: 165, height: 165, startAngle: 0, arcAngle: 360);
        for (int i=0;i<=120;i+=10)
            drawMark(g, x: x+83, y: y+83, len: 83,i);
        g.setColor(Color.cyan);
        g.fillArc( x: x+2, y: y+2, width: 163, height: 163, startAngle: -150,speed!=0?-(2*speed):-1);
        g.setColor(Color.black);
        g.fillArc( x: x+8, y: y+8, width: 150, height: 150, startAngle: 0, arcAngle: 360);
        //odometer
        drawOdo(g, x: x+57, y: y+120,distance);

    }

    private void drawMark(Graphics g, int x, int y, int len, int n) {  1 usage  lameesmmi
        double flen = len;
        double fangle = ((60+n*2)*Math.PI)/180;
        int mx = x - (int)(flen*Math.sin(fangle));
        int my = y + (int)(flen*Math.cos(fangle));
        g.drawLine(x,y,mx,my);
        // display number
        flen = flen+12;
        mx = x- 7 - (int)(flen*Math.sin(fangle));
        my = y+7+ (int)(flen*Math.cos(fangle));
        g.drawString(String.valueOf(n),mx,my);
    }
```



```
CarSimulatorTest.java    CarSimulator.java ×

public class CarSimulator extends Canvas implements Runnable, CarSpeed{  4 usages  lameesmmi      ▲6 ✓8
    private void drawOdo(Graphics g, int x, int y, int distance) {  1 usage  lameesmmi
        String zero = "0";
        int digits[]= new int[4];
        for (int i=3;i>=0;i--) {
            digits[i]=distance%10;
            distance=distance/10;
        }
        g.setColor(Color.white);
        FontMetrics fm = g.getFontMetrics();
        int w = fm.stringWidth(zero);
        int h = fm.getHeight();
        for (int i=0;i<4;i++) {
            g.drawRect( x: x+(w+4)*i,y, width: w+4, height: h+2);
            if (i>1) g.setColor(Color.yellow);
            g.drawString(String.valueOf(digits[i]), x: x+(w+4)*i+3, y: y+h-2);
            g.setColor(Color.white);
        }
    }


    public synchronized void  engineOn(){  5 usages  lameesmmi
        ignition = true;
        if (engine==null) {
            engine = new Thread( task: this);
            engine.start();
        }
        repaint();
    }

    public synchronized void  engineOff() {  3 usages  lameesmmi
        ignition = false;
```

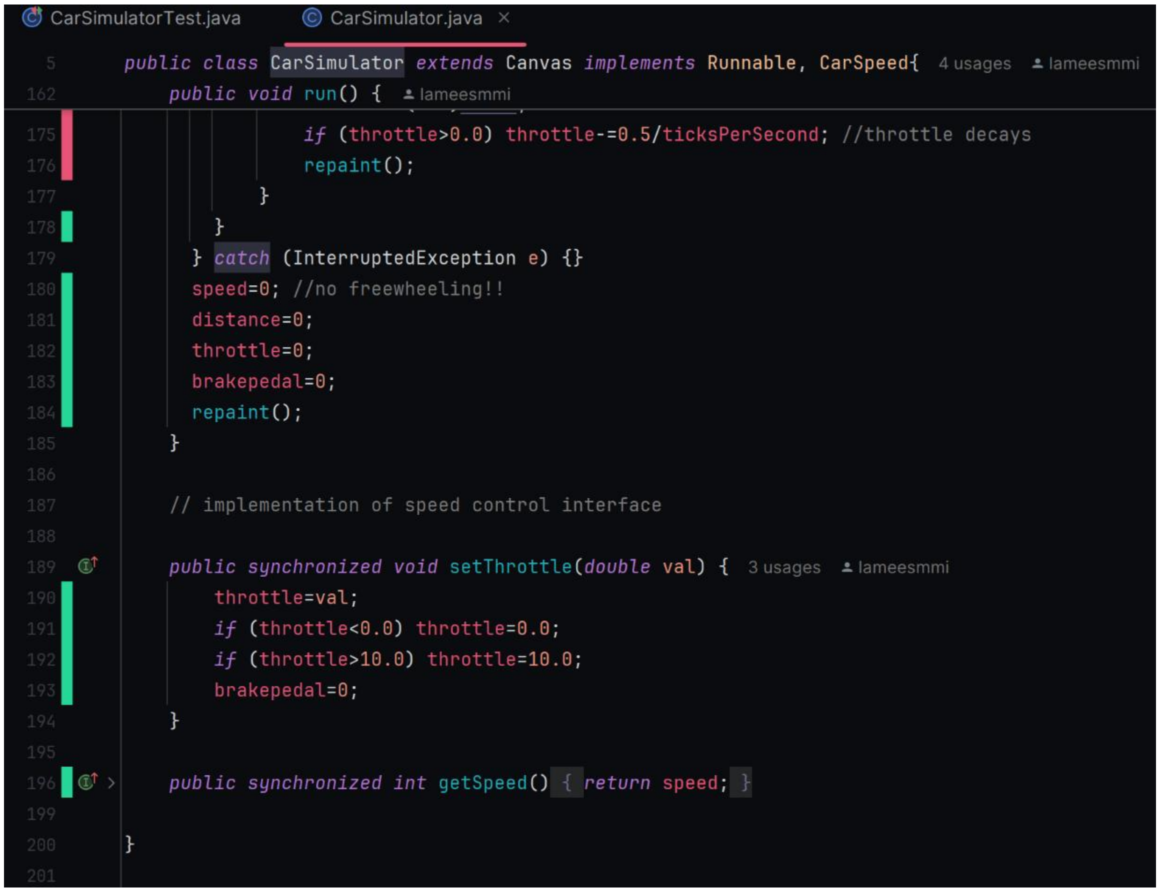


```
CarSimulatorTest.java    CarSimulator.java ×

public class CarSimulator extends Canvas implements Runnable, CarSpeed{  4 usages  lameesmmi      ▲6 ✓8

    public synchronized void  engineOff() {  3 usages  lameesmmi
        ignition = false;
        engine=null;
        repaint();
    }

    public synchronized void  accelerate() {  5 usages  lameesmmi
        if (brakepedal>0)
            brakepedal=0;
        else {
            if (throttle<(maxThrottle-1))
                throttle +=1.0;
            else
                throttle=maxThrottle;
        }
        repaint();
    }

    public synchronized void  brake() {  3 usages  lameesmmi
        if (throttle>0.0)
            throttle=0.0;
        else {
          if (brakepedal<maxBrake) brakepedal +=1;
        }
        repaint();
    }

    public void run() {  lameesmmi
      try {
        double fdist=0.0;
```



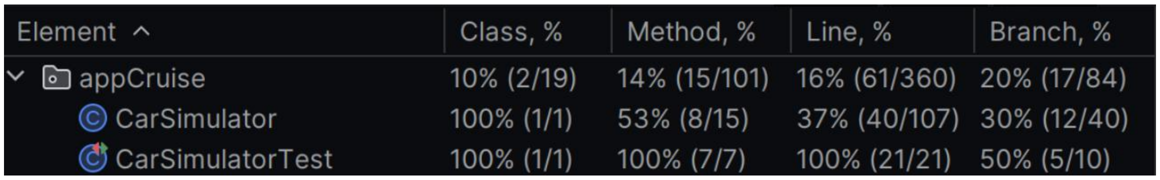
CarSimulatorTest.java   CarSimulator.java ×

```java
public class CarSimulator extends Canvas implements Runnable, CarSpeed{  4 usages  lameesmmi
    public void run() {  lameesmmi
            double fdist=0.0;
            double fspeed=0.0;
            synchronized(this) {
                while (engine!=null) {
                wait( timeoutMillis: 1000/ticksPerSecond);
                    fspeed = fspeed+((throttle - fspeed/airResistance - 2*brakepedal))/ticksPerSecond;
                    if (fspeed>maxSpeed) fspeed=maxSpeed;
                    if (fspeed<0) fspeed=0;
                    fdist = fdist + (fspeed/36.0)/ticksPerSecond;
                    speed = (int)fspeed;
                    distance=(int)fdist;
                    if (throttle>0.0) throttle-=0.5/ticksPerSecond; //throttle decays
                    repaint();
                }
            }
        } catch (InterruptedException e) {}
        speed=0; //no freewheeling!!
        distance=0;
        throttle=0;
        brakepedal=0;
        repaint();
    }

    // implementation of speed control interface

    public synchronized void setThrottle(double val) {  3 usages  lameesmmi
        throttle=val;
        if (throttle<0.0) throttle=0.0;
        if (throttle>10.0) throttle=10.0;
        brakepedal=0;
```



```java
public class CarSimulator extends Canvas implements Runnable, CarSpeed{  4 usages  lameesmmi
    public void run() {  lameesmmi
                    if (throttle>0.0) throttle-=0.5/ticksPerSecond; //throttle decays
                    repaint();
                }
            }
        } catch (InterruptedException e) {}
        speed=0; //no freewheeling!!
        distance=0;
        throttle=0;
        brakepedal=0;
        repaint();
    }

    // implementation of speed control interface

    public synchronized void setThrottle(double val) {  3 usages  lameesmmi
        throttle=val;
        if (throttle<0.0) throttle=0.0;
        if (throttle>10.0) throttle=10.0;
        brakepedal=0;
    }

    public synchronized int getSpeed() { return speed; }

}
```

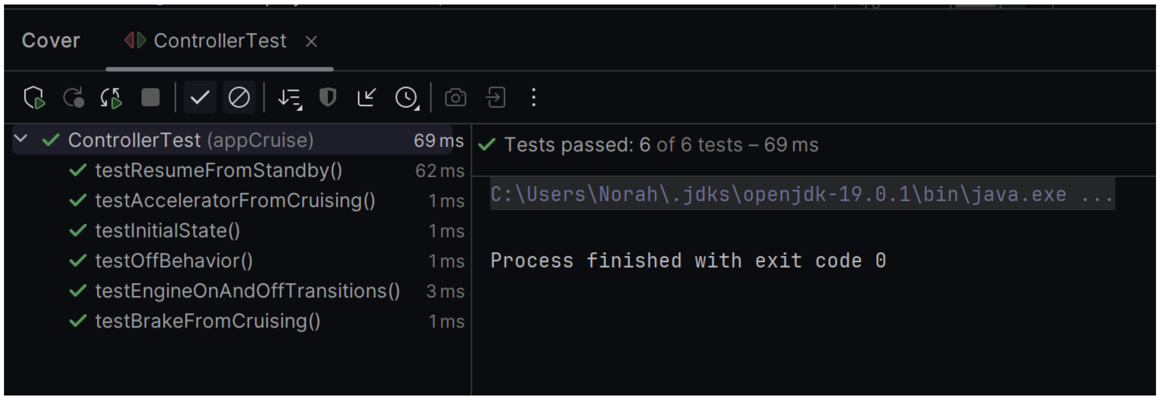
Coverage Summary Tab:



Element	Class, %	Method, %	Line, %	Branch, %
appCruise	10% (2/19)	14% (15/101)	16% (61/360)	20% (17/84)
CarSimulator	100% (1/1)	53% (8/15)	37% (40/107)	30% (12/40)
CarSimulatorTest	100% (1/1)	100% (7/7)	100% (21/21)	50% (5/10)

Controller.java

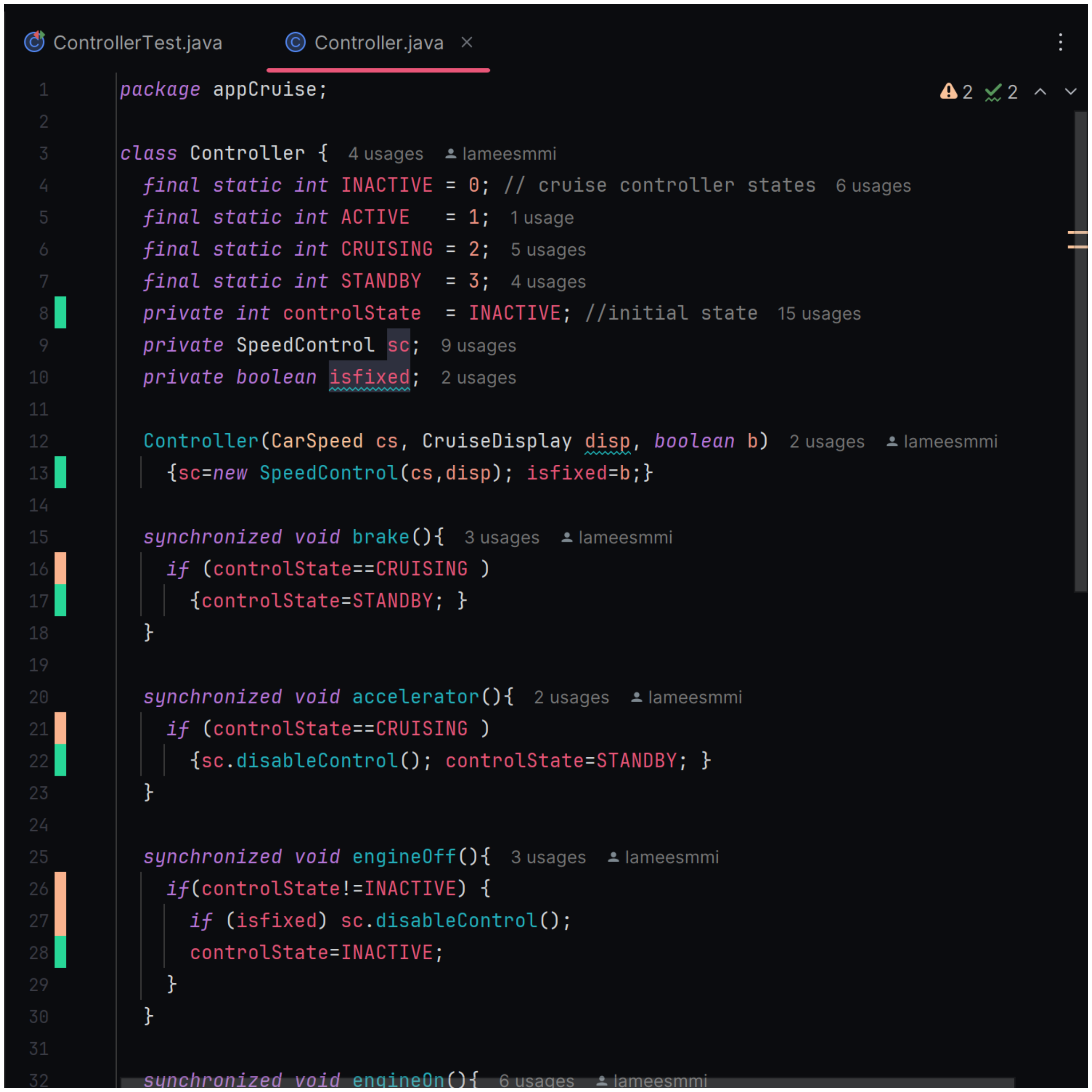
Test Results:



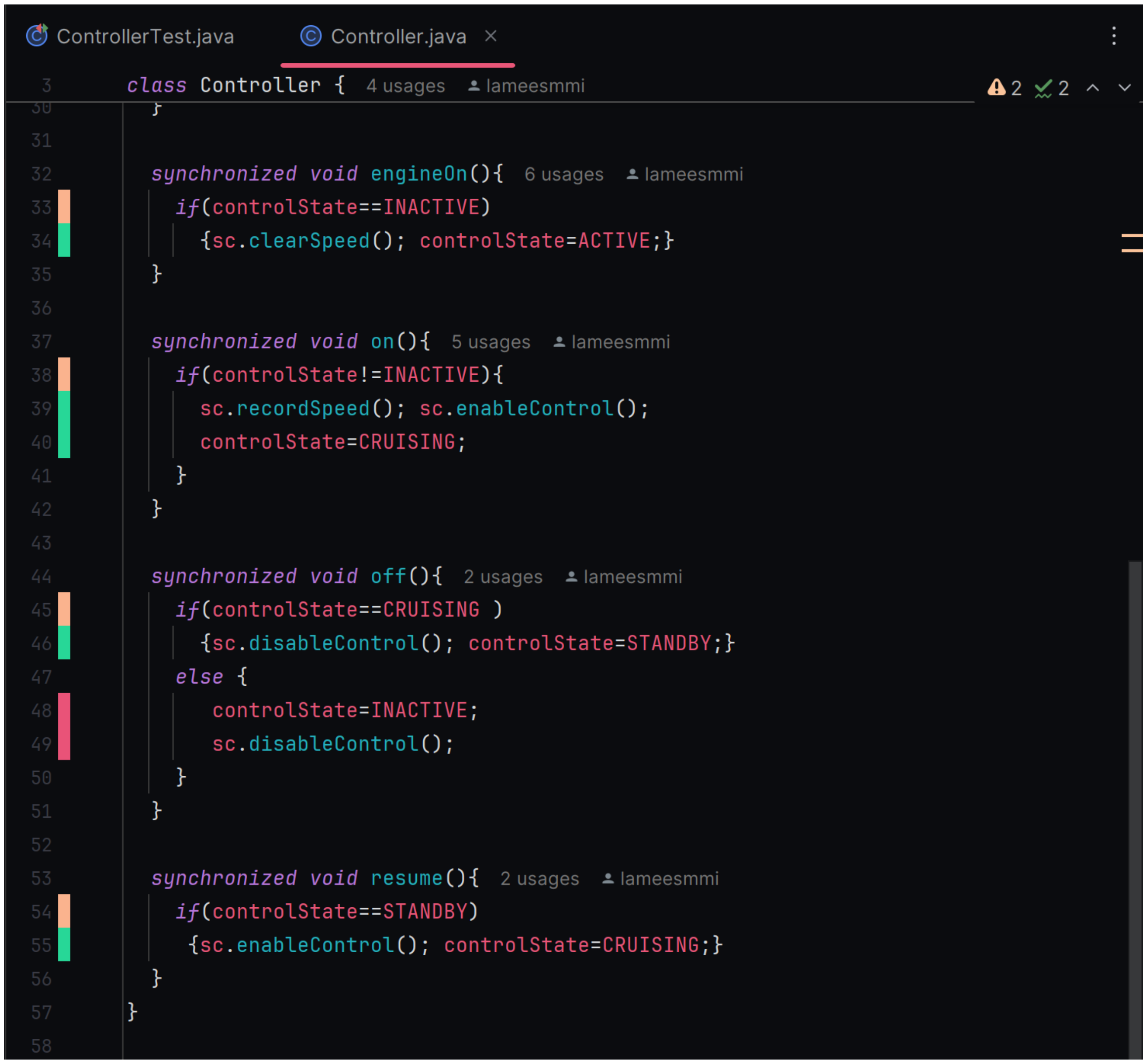
```
✓ ControllerTest (appCruise)            69 ms
    ✓ testResumeFromStandby()           62 ms
    ✓ testAcceleratorFromCruising()      1 ms
    ✓ testInitialState()                 1 ms
    ✓ testOffBehavior()                  1 ms
    ✓ testEngineOnAndOffTransitions()    3 ms
    ✓ testBrakeFromCruising()            1 ms
```

```
✓ Tests passed: 6 of 6 tests – 69 ms

C:\Users\Norah\.jdks\openjdk-19.0.1\bin\java.exe ...

Process finished with exit code 0
```

Code Coverage View:

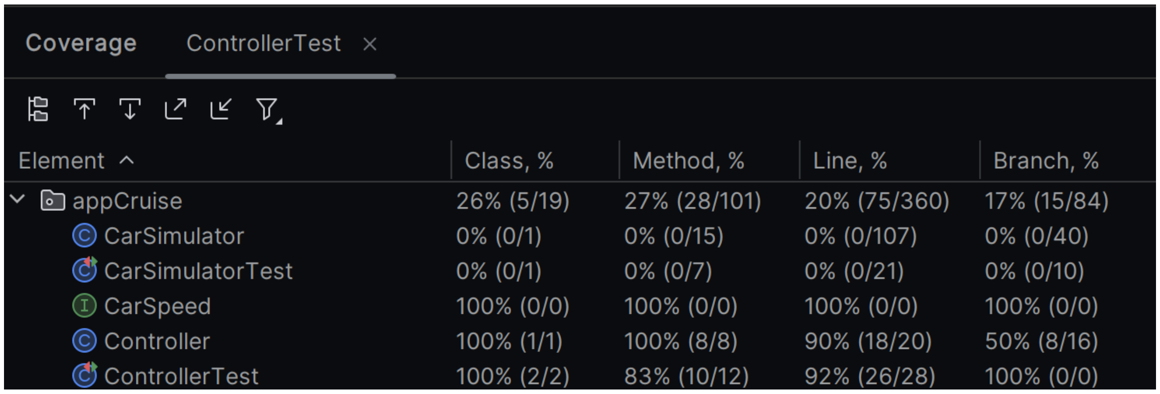


```java
package appCruise;

class Controller {  4 usages  lameesmmi
  final static int INACTIVE = 0; // cruise controller states  6 usages
  final static int ACTIVE   = 1;  1 usage
  final static int CRUISING = 2;  5 usages
  final static int STANDBY  = 3;  4 usages
  private int controlState  = INACTIVE; //initial state  15 usages
  private SpeedControl sc;  9 usages
  private boolean isfixed;  2 usages

  Controller(CarSpeed cs, CruiseDisplay disp, boolean b)  2 usages  lameesmmi
    {sc=new SpeedControl(cs,disp); isfixed=b;}

  synchronized void brake(){  3 usages  lameesmmi
    if (controlState==CRUISING )
      {controlState=STANDBY; }
  }

  synchronized void accelerator(){  2 usages  lameesmmi
    if (controlState==CRUISING )
      {sc.disableControl(); controlState=STANDBY; }
  }

  synchronized void engineOff(){  3 usages  lameesmmi
    if(controlState!=INACTIVE) {
      if (isfixed) sc.disableControl();
      controlState=INACTIVE;
    }
  }

  synchronized void engineOn(){  6 usages  lameesmmi
```


ControllerTest.java Controller.java

```
class Controller {  4 usages  lameesmmi
    }

    synchronized void engineOn(){  6 usages  lameesmmi
      if(controlState==INACTIVE)
        {sc.clearSpeed(); controlState=ACTIVE;}
    }

    synchronized void on(){  5 usages  lameesmmi
      if(controlState!=INACTIVE){
        sc.recordSpeed(); sc.enableControl();
        controlState=CRUISING;
      }
    }

    synchronized void off(){  2 usages  lameesmmi
      if(controlState==CRUISING )
        {sc.disableControl(); controlState=STANDBY;}
      else {
         controlState=INACTIVE;
         sc.disableControl();
      }
    }

    synchronized void resume(){  2 usages  lameesmmi
      if(controlState==STANDBY)
       {sc.enableControl(); controlState=CRUISING;}
    }
}
```

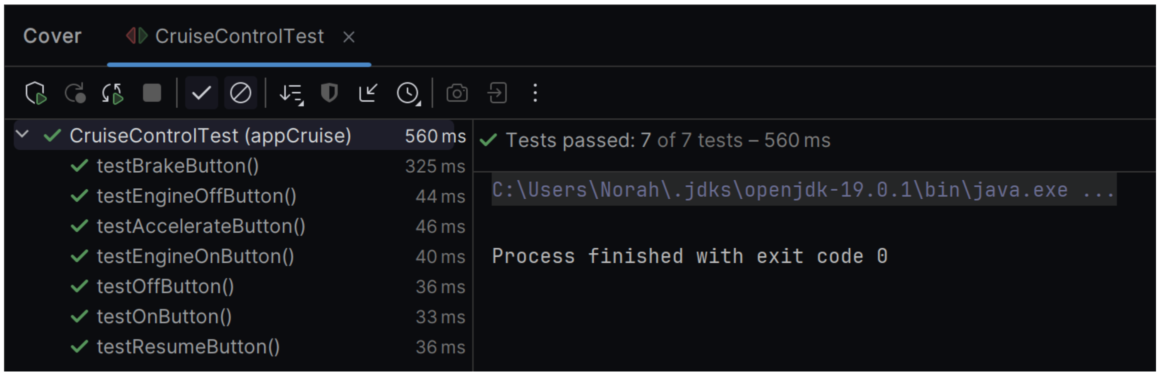
Coverage Summary Tab:


Coverage ControllerTest

Element	Class, %	Method, %	Line, %	Branch, %
appCruise	26% (5/19)	27% (28/101)	20% (75/360)	17% (15/84)
CarSimulator	0% (0/1)	0% (0/15)	0% (0/107)	0% (0/40)
CarSimulatorTest	0% (0/1)	0% (0/7)	0% (0/21)	0% (0/10)
CarSpeed	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
Controller	100% (1/1)	100% (8/8)	90% (18/20)	50% (8/16)
ControllerTest	100% (2/2)	83% (10/12)	92% (26/28)	100% (0/0)

CruiseControl.java

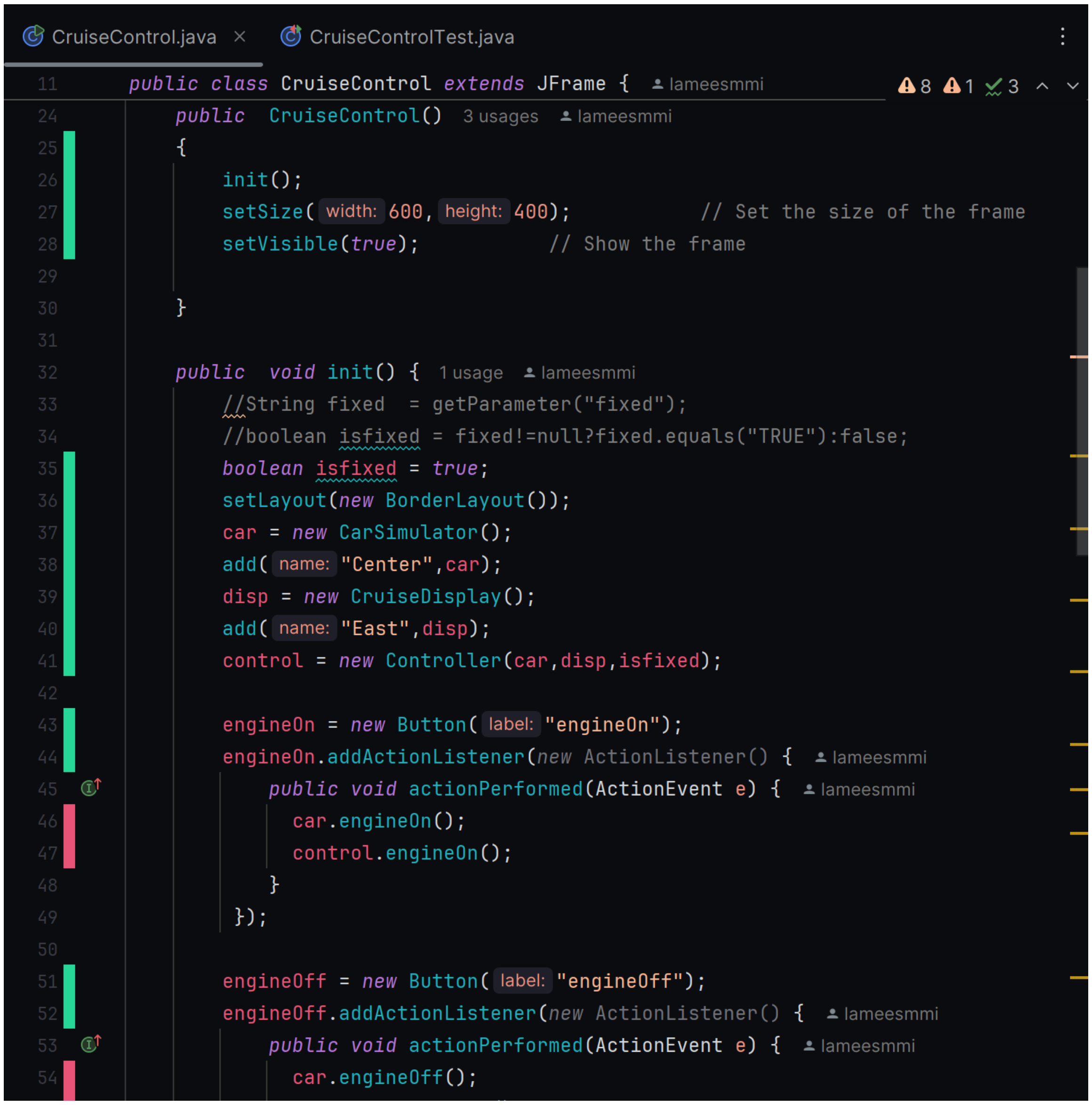
Test Results:



```
CruiseControlTest (appCruise)   560 ms
  testBrakeButton()             325 ms
  testEngineOffButton()          44 ms
  testAccelerateButton()         46 ms
  testEngineOnButton()           40 ms
  testOffButton()                36 ms
  testOnButton()                 33 ms
  testResumeButton()             36 ms
```

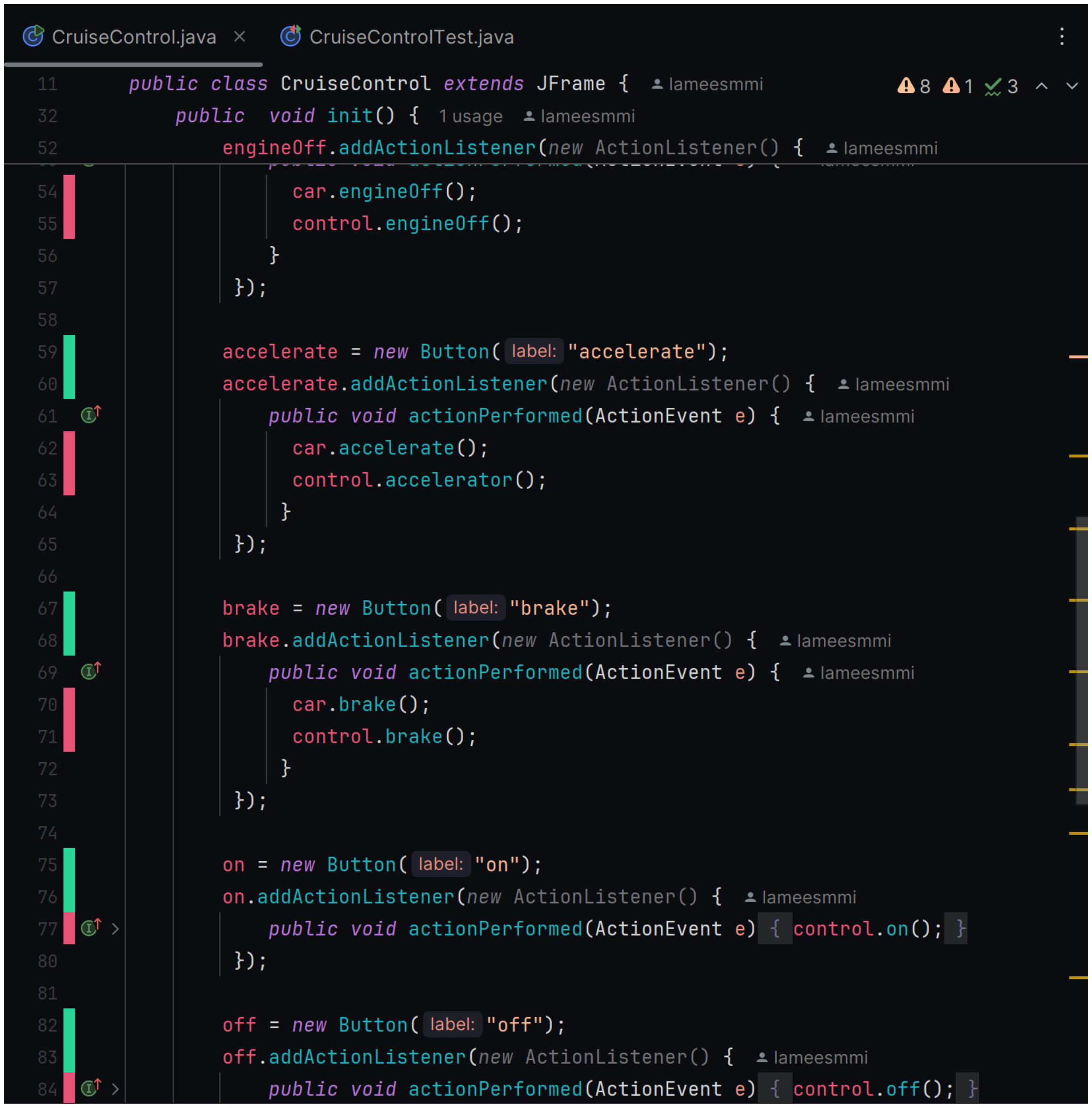
```
Tests passed: 7 of 7 tests – 560 ms

C:\Users\Norah\.jdks\openjdk-19.0.1\bin\java.exe ...

Process finished with exit code 0
```

Code Coverage View:

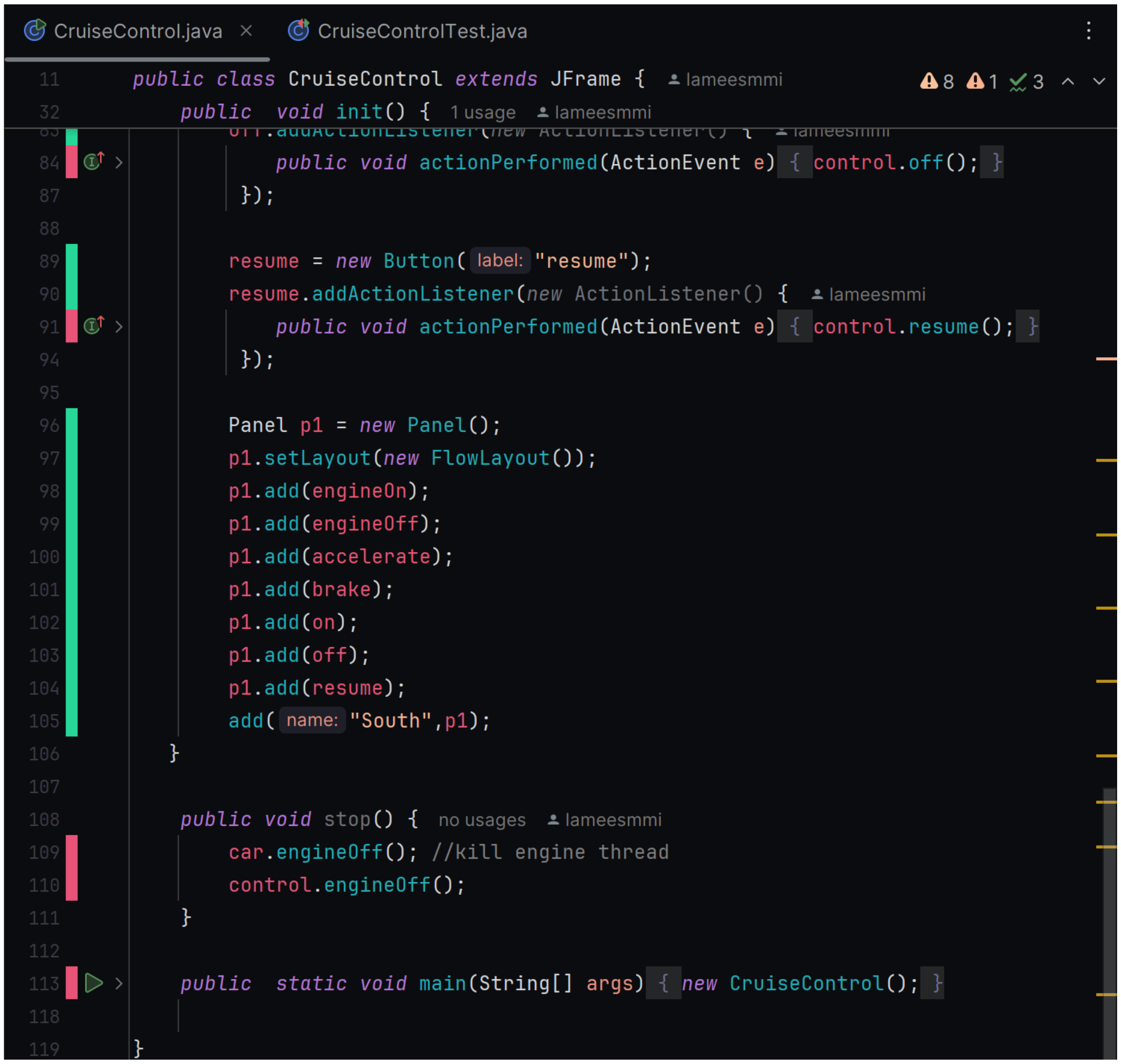


CruiseControl.java × CruiseControlTest.java

```java
public class CruiseControl extends JFrame {
    public  CruiseControl()
    {
        init();
        setSize(600, 400);              // Set the size of the frame
        setVisible(true);           // Show the frame

    }

    public  void init() {
        //String fixed  = getParameter("fixed");
        //boolean isfixed = fixed!=null?fixed.equals("TRUE"):false;
        boolean isfixed = true;
        setLayout(new BorderLayout());
        car = new CarSimulator();
        add("Center",car);
        disp = new CruiseDisplay();
        add("East",disp);
        control = new Controller(car,disp,isfixed);

        engineOn = new Button("engineOn");
        engineOn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
              car.engineOn();
              control.engineOn();
            }
         });

        engineOff = new Button("engineOff");
        engineOff.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
              car.engineOff();
```

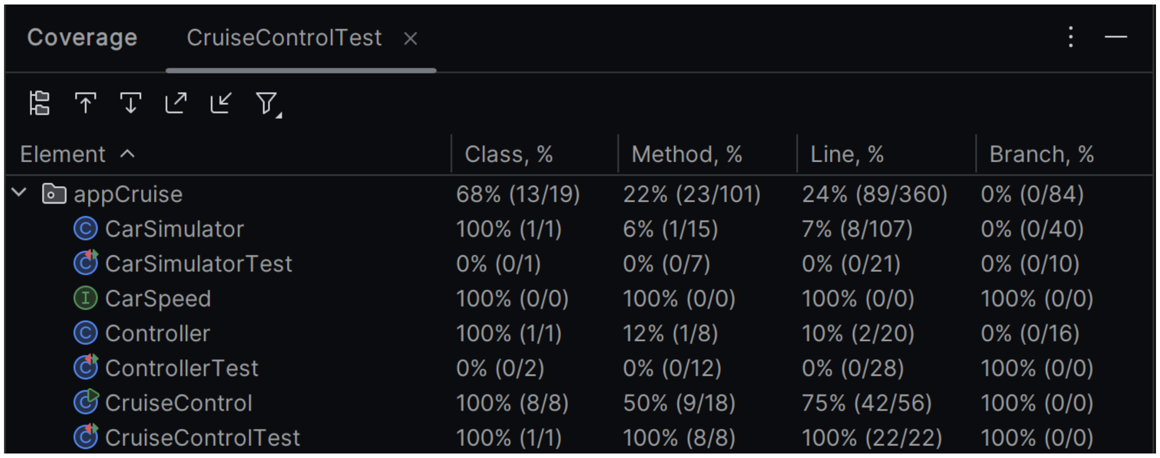


```
    public class CruiseControl extends JFrame {  lameesmmi
        public  void init() {  1 usage  lameesmmi
            engineOff.addActionListener(new ActionListener() {  lameesmmi
                    car.engineOff();
                    control.engineOff();
                }
            });

            accelerate = new Button( label: "accelerate");
            accelerate.addActionListener(new ActionListener() {  lameesmmi
                public void actionPerformed(ActionEvent e) {  lameesmmi
                    car.accelerate();
                    control.accelerator();
                }
            });

            brake = new Button( label: "brake");
            brake.addActionListener(new ActionListener() {  lameesmmi
                public void actionPerformed(ActionEvent e) {  lameesmmi
                    car.brake();
                    control.brake();
                }
            });

            on = new Button( label: "on");
            on.addActionListener(new ActionListener() {  lameesmmi
                public void actionPerformed(ActionEvent e) { control.on(); }
            });

            off = new Button( label: "off");
            off.addActionListener(new ActionListener() {  lameesmmi
                public void actionPerformed(ActionEvent e) { control.off(); }
```



```
public class CruiseControl extends JFrame {
    public  void init() {
            public void actionPerformed(ActionEvent e) { control.off(); }
        });

        resume = new Button( label: "resume");
        resume.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) { control.resume(); }
        });

        Panel p1 = new Panel();
        p1.setLayout(new FlowLayout());
        p1.add(engineOn);
        p1.add(engineOff);
        p1.add(accelerate);
        p1.add(brake);
        p1.add(on);
        p1.add(off);
        p1.add(resume);
        add( name: "South",p1);
    }

    public void stop() {
        car.engineOff(); //kill engine thread
        control.engineOff();
    }

    public  static void main(String[] args) { new CruiseControl(); }

}
```

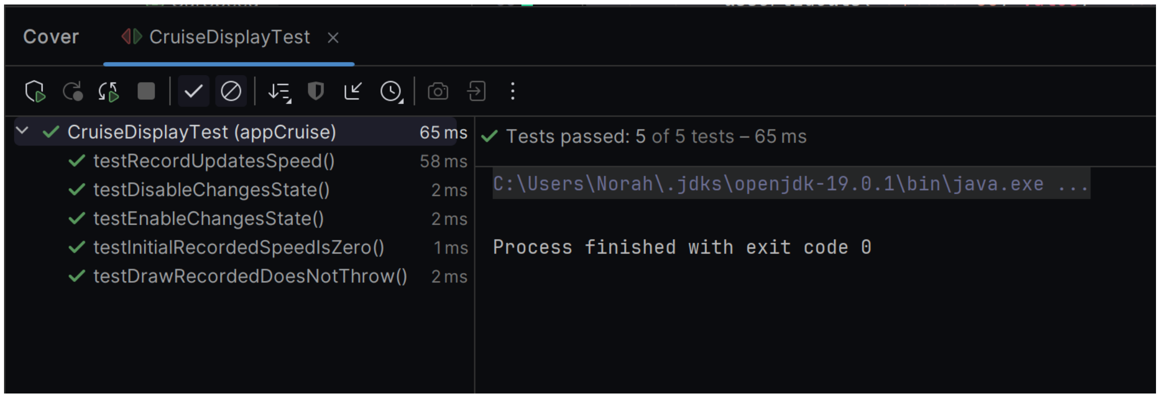
Coverage Summary Tab:



Element	Class, %	Method, %	Line, %	Branch, %
appCruise	68% (13/19)	22% (23/101)	24% (89/360)	0% (0/84)
CarSimulator	100% (1/1)	6% (1/15)	7% (8/107)	0% (0/40)
CarSimulatorTest	0% (0/1)	0% (0/7)	0% (0/21)	0% (0/10)
CarSpeed	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
Controller	100% (1/1)	12% (1/8)	10% (2/20)	0% (0/16)
ControllerTest	0% (0/2)	0% (0/12)	0% (0/28)	100% (0/0)
CruiseControl	100% (8/8)	50% (9/18)	75% (42/56)	100% (0/0)
CruiseControlTest	100% (1/1)	100% (8/8)	100% (22/22)	100% (0/0)

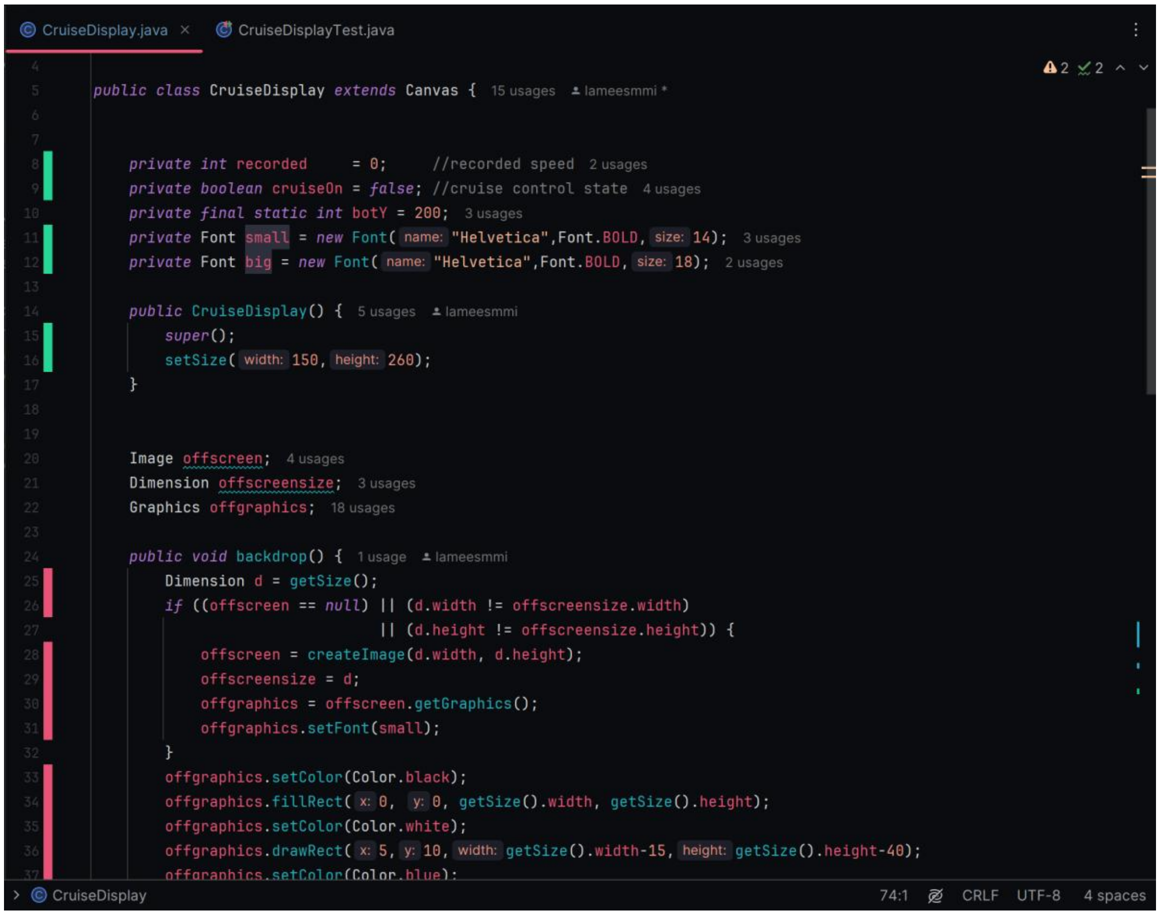
CruiseDisplay.java

Test Results:

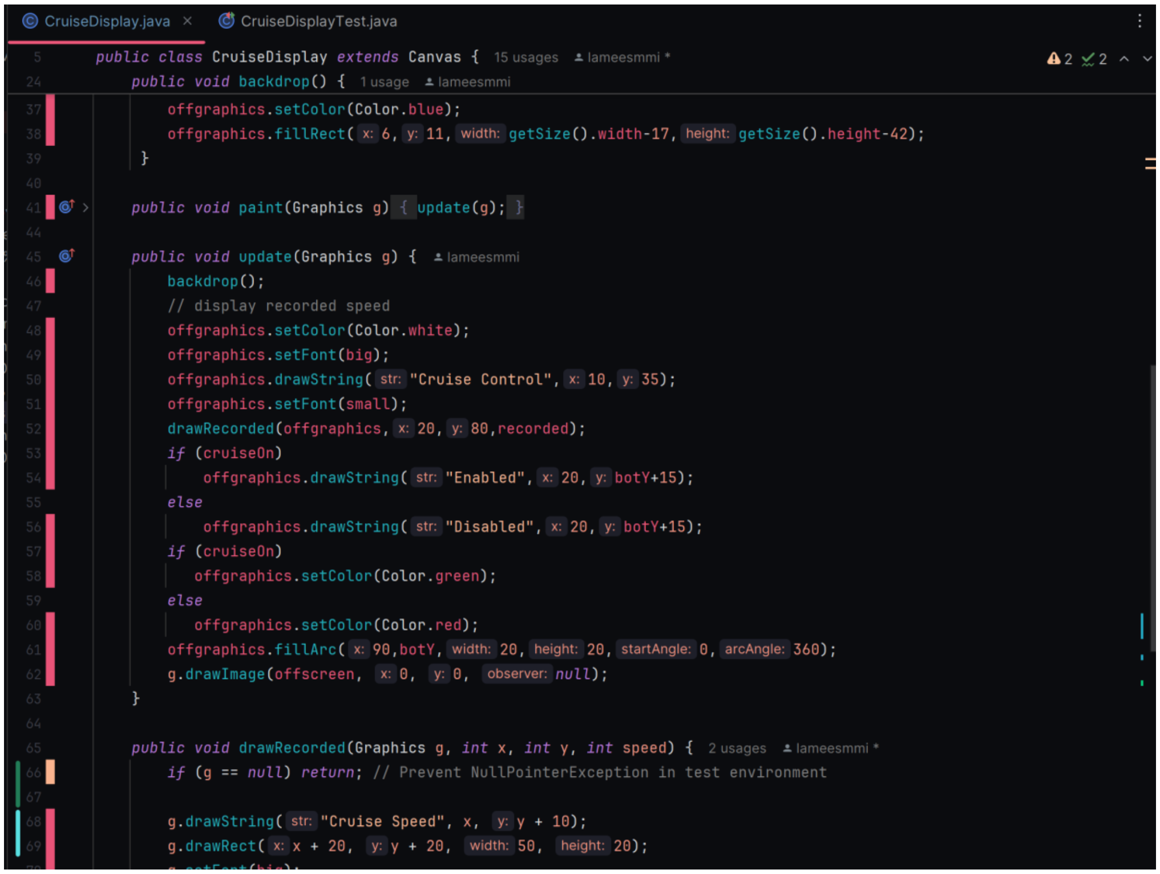


```
Cover    CruiseDisplayTest ×

CruiseDisplayTest (appCruise)        65 ms    Tests passed: 5 of 5 tests – 65 ms
  testRecordUpdatesSpeed()           58 ms
  testDisableChangesState()           2 ms    C:\Users\Norah\.jdks\openjdk-19.0.1\bin\java.exe ...
  testEnableChangesState()            2 ms
  testInitialRecordedSpeedIsZero()    1 ms    Process finished with exit code 0
  testDrawRecordedDoesNotThrow()      2 ms
```

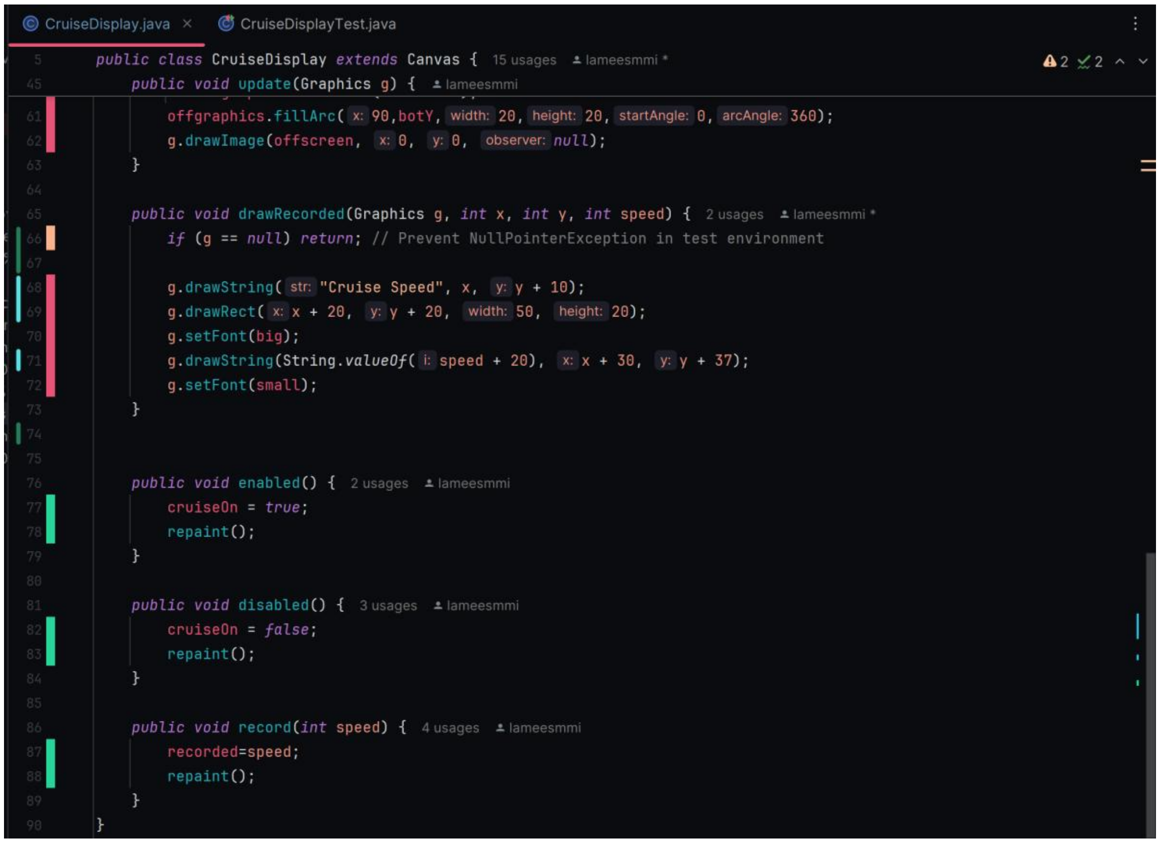
Code Coverage View:



```java
public class CruiseDisplay extends Canvas {  15 usages  lameesmmi *


    private int recorded     = 0;     //recorded speed  2 usages
    private boolean cruiseOn = false; //cruise control state  4 usages
    private final static int botY = 200;  3 usages
    private Font small = new Font( name: "Helvetica",Font.BOLD, size: 14);  3 usages
    private Font big = new Font( name: "Helvetica",Font.BOLD, size: 18);  2 usages

    public CruiseDisplay() {  5 usages  lameesmmi
        super();
        setSize( width: 150, height: 260);
    }


    Image offscreen;  4 usages
    Dimension offscreensize;  3 usages
    Graphics offgraphics;  18 usages

    public void backdrop() {  1 usage  lameesmmi
        Dimension d = getSize();
        if ((offscreen == null) || (d.width != offscreensize.width)
                                || (d.height != offscreensize.height)) {
            offscreen = createImage(d.width, d.height);
            offscreensize = d;
            offgraphics = offscreen.getGraphics();
            offgraphics.setFont(small);
        }
        offgraphics.setColor(Color.black);
        offgraphics.fillRect( x: 0,  y: 0, getSize().width, getSize().height);
        offgraphics.setColor(Color.white);
        offgraphics.drawRect( x: 5,  y: 10, width: getSize().width-15, height: getSize().height-40);
        offgraphics.setColor(Color.blue);
```

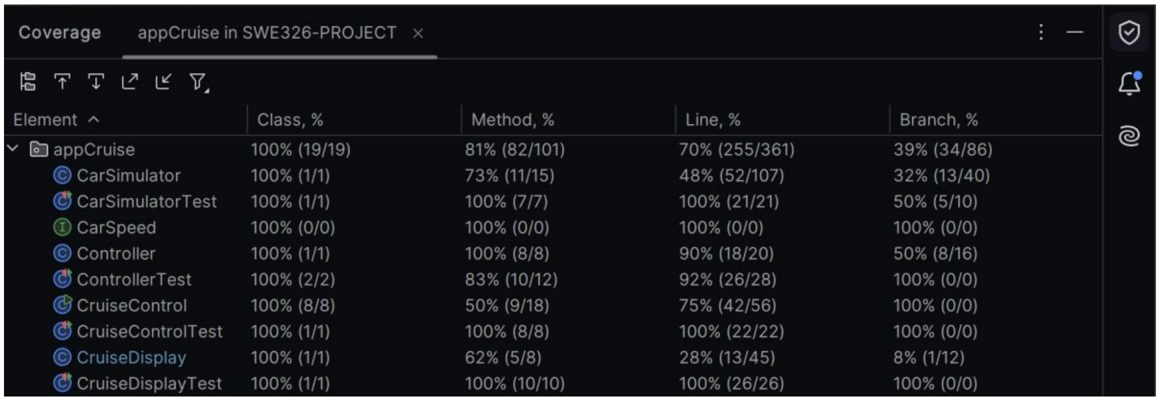


```
    public class CruiseDisplay extends Canvas {  15 usages  lameesmmi *
        public void backdrop() {  1 usage  lameesmmi
            offgraphics.setColor(Color.blue);
            offgraphics.fillRect( x: 6, y: 11, width: getSize().width-17, height: getSize().height-42);
        }

        public void paint(Graphics g) { update(g); }

        public void update(Graphics g) {  lameesmmi
            backdrop();
            // display recorded speed
            offgraphics.setColor(Color.white);
            offgraphics.setFont(big);
            offgraphics.drawString( str: "Cruise Control", x: 10, y: 35);
            offgraphics.setFont(small);
            drawRecorded(offgraphics, x: 20, y: 80,recorded);
            if (cruiseOn)
                offgraphics.drawString( str: "Enabled", x: 20, y: botY+15);
            else
                offgraphics.drawString( str: "Disabled", x: 20, y: botY+15);
            if (cruiseOn)
               offgraphics.setColor(Color.green);
            else
               offgraphics.setColor(Color.red);
            offgraphics.fillArc( x: 90,botY, width: 20, height: 20, startAngle: 0, arcAngle: 360);
            g.drawImage(offscreen,  x: 0,  y: 0,  observer: null);
        }

        public void drawRecorded(Graphics g, int x, int y, int speed) {  2 usages  lameesmmi *
            if (g == null) return; // Prevent NullPointerException in test environment

            g.drawString( str: "Cruise Speed", x,  y: y + 10);
            g.drawRect( x: x + 20,  y: y + 20,  width: 50,  height: 20);
```



```
CruiseDisplay.java ×    CruiseDisplayTest.java
public class CruiseDisplay extends Canvas {  15 usages  lameesmmi *
    public void update(Graphics g) {  lameesmmi
            offgraphics.fillArc( x: 90,botY, width: 20, height: 20, startAngle: 0, arcAngle: 360);
            g.drawImage(offscreen,  x: 0,  y: 0,  observer: null);
        }

        public void drawRecorded(Graphics g, int x, int y, int speed) {  2 usages  lameesmmi *
            if (g == null) return; // Prevent NullPointerException in test environment

            g.drawString( str: "Cruise Speed", x,  y: y + 10);
            g.drawRect( x: x + 20,  y: y + 20,  width: 50,  height: 20);
            g.setFont(big);
            g.drawString(String.valueOf( i: speed + 20),  x: x + 30,  y: y + 37);
            g.setFont(small);
        }


        public void enabled() {  2 usages  lameesmmi
            cruiseOn = true;
            repaint();
        }

        public void disabled() {  3 usages  lameesmmi
            cruiseOn = false;
            repaint();
        }

        public void record(int speed) {  4 usages  lameesmmi
            recorded=speed;
            repaint();
        }
    }
```

Coverage Summary Tab:

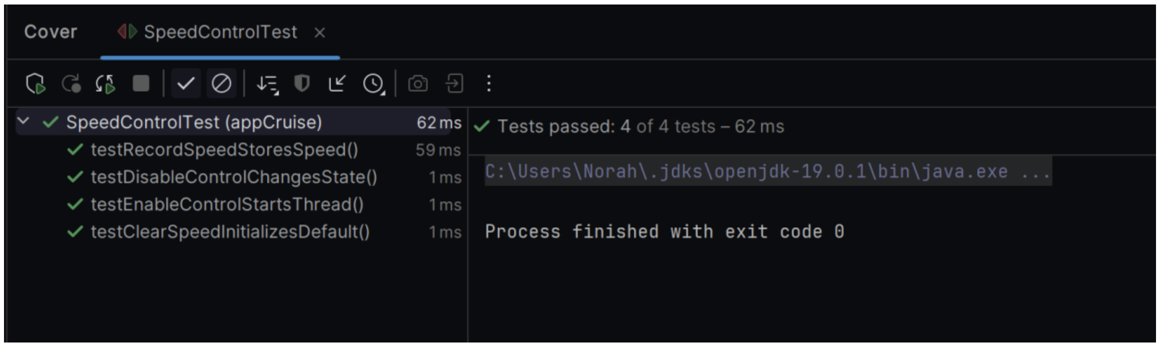


Coverage   appCruise in SWE326-PROJECT ×

Element	Class, %	Method, %	Line, %	Branch, %
appCruise	100% (19/19)	81% (82/101)	70% (255/361)	39% (34/86)
CarSimulator	100% (1/1)	73% (11/15)	48% (52/107)	32% (13/40)
CarSimulatorTest	100% (1/1)	100% (7/7)	100% (21/21)	50% (5/10)
CarSpeed	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
Controller	100% (1/1)	100% (8/8)	90% (18/20)	50% (8/16)
ControllerTest	100% (2/2)	83% (10/12)	92% (26/28)	100% (0/0)
CruiseControl	100% (8/8)	50% (9/18)	75% (42/56)	100% (0/0)
CruiseControlTest	100% (1/1)	100% (8/8)	100% (22/22)	100% (0/0)
CruiseDisplay	100% (1/1)	62% (5/8)	28% (13/45)	8% (1/12)
CruiseDisplayTest	100% (1/1)	100% (10/10)	100% (26/26)	100% (0/0)

SpeedControl.java

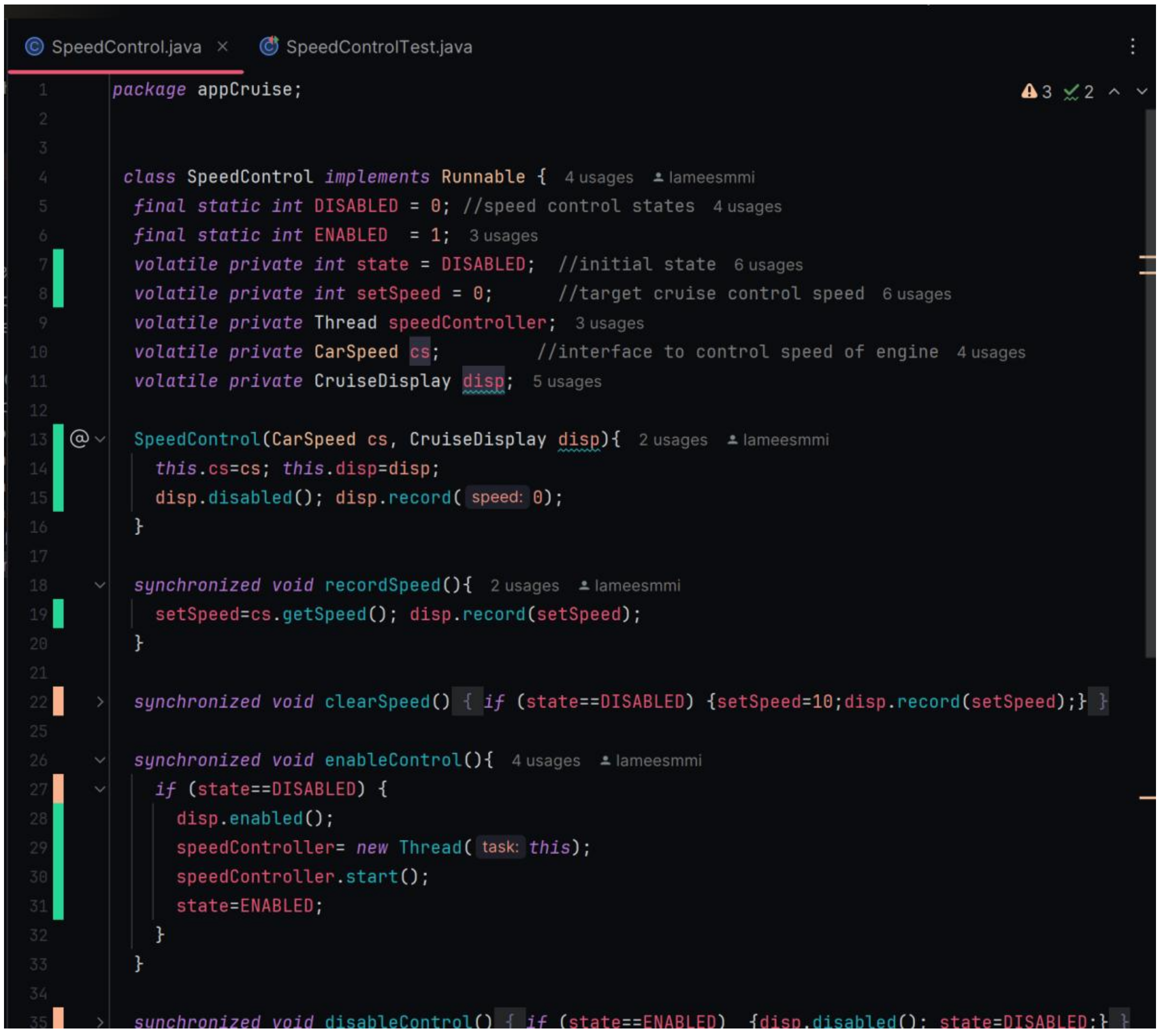
Test Results:



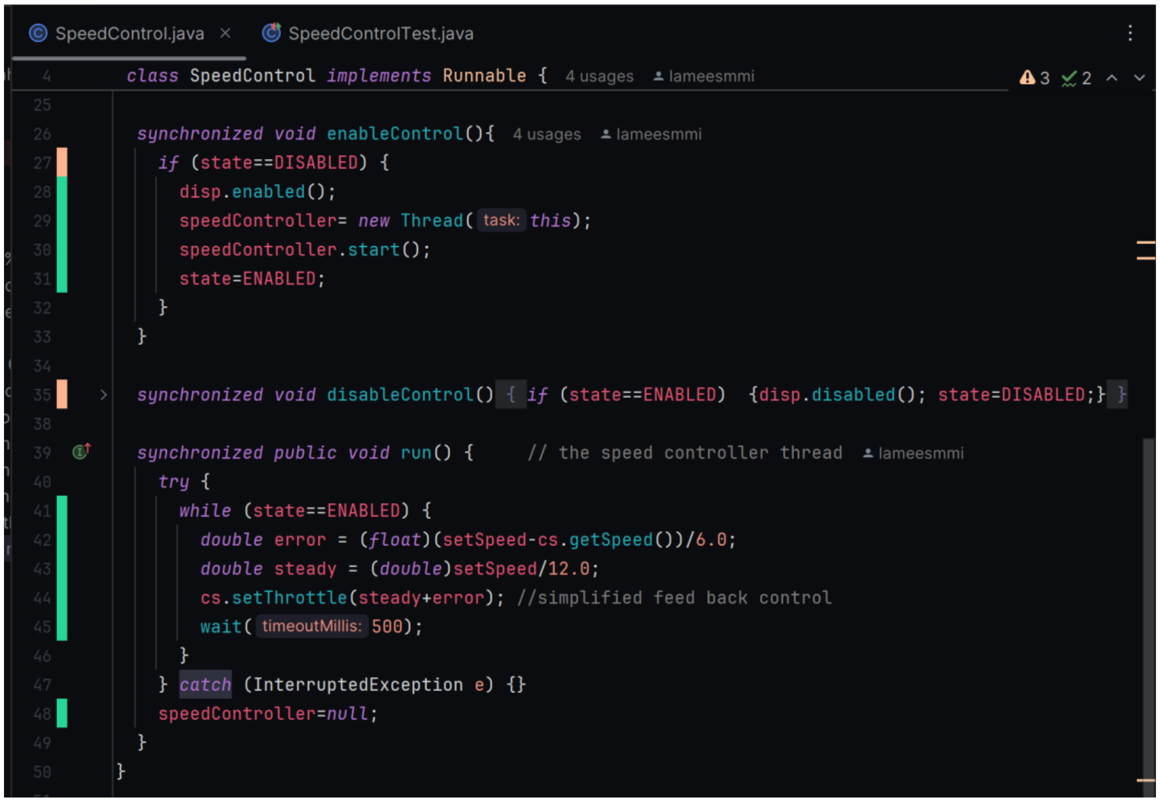
```
SpeedControlTest (appCruise)                 62 ms
    testRecordSpeedStoresSpeed()             59 ms
    testDisableControlChangesState()          1 ms
    testEnableControlStartsThread()           1 ms
    testClearSpeedInitializesDefault()        1 ms
```

```
Tests passed: 4 of 4 tests – 62 ms

C:\Users\Norah\.jdks\openjdk-19.0.1\bin\java.exe ...

Process finished with exit code 0
```

Code Coverage View:



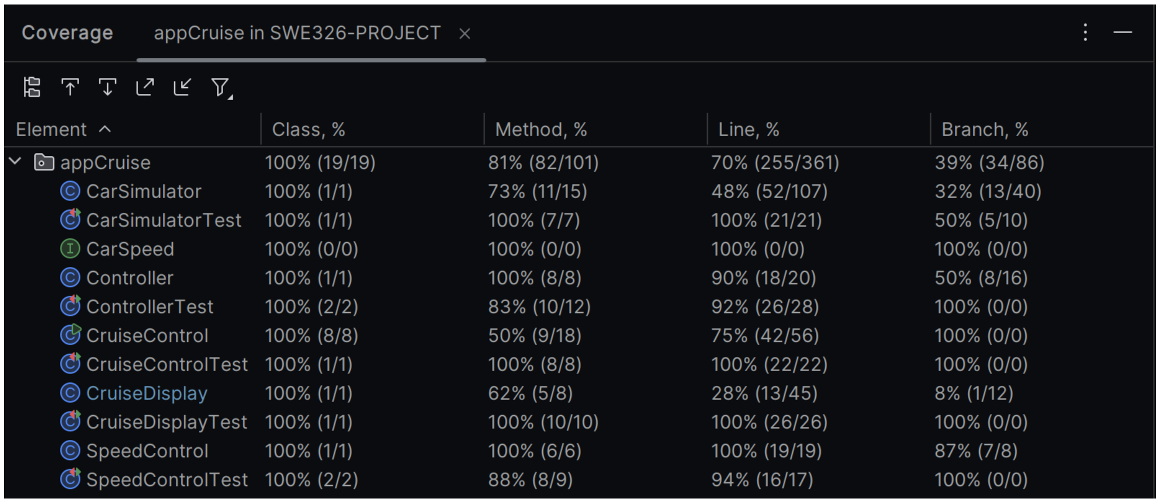
```
package appCruise;


 class SpeedControl implements Runnable {  4 usages  lameesmmi
  final static int DISABLED = 0; //speed control states  4 usages
  final static int ENABLED  = 1;  3 usages
  volatile private int state = DISABLED;  //initial state  6 usages
  volatile private int setSpeed = 0;      //target cruise control speed  6 usages
  volatile private Thread speedController;  3 usages
  volatile private CarSpeed cs;      //interface to control speed of engine  4 usages
  volatile private CruiseDisplay disp;  5 usages

  SpeedControl(CarSpeed cs, CruiseDisplay disp){  2 usages  lameesmmi
    this.cs=cs; this.disp=disp;
    disp.disabled(); disp.record( speed: 0);
  }

  synchronized void recordSpeed(){  2 usages  lameesmmi
    setSpeed=cs.getSpeed(); disp.record(setSpeed);
  }

  synchronized void clearSpeed() { if (state==DISABLED) {setSpeed=10;disp.record(setSpeed);} }

  synchronized void enableControl(){  4 usages  lameesmmi
    if (state==DISABLED) {
      disp.enabled();
      speedController= new Thread( task: this);
      speedController.start();
      state=ENABLED;
    }
  }

  synchronized void disableControl() { if (state==ENABLED)  {disp.disabled(); state=DISABLED;} }
```



SpeedControl.java × SpeedControlTest.java

```
class SpeedControl implements Runnable {  4 usages  lameesmmi

    synchronized void enableControl(){  4 usages  lameesmmi
      if (state==DISABLED) {
        disp.enabled();
        speedController= new Thread( task: this);
        speedController.start();
        state=ENABLED;
      }
    }

    synchronized void disableControl() { if (state==ENABLED)  {disp.disabled(); state=DISABLED;} }

    synchronized public void run() {     // the speed controller thread  lameesmmi
      try {
        while (state==ENABLED) {
          double error = (float)(setSpeed-cs.getSpeed())/6.0;
          double steady = (double)setSpeed/12.0;
          cs.setThrottle(steady+error); //simplified feed back control
          wait( timeoutMillis: 500);
        }
      } catch (InterruptedException e) {}
      speedController=null;
    }
}
```

Coverage Summary Tab:



Coverage appCruise in SWE326-PROJECT ×

Element	Class, %	Method, %	Line, %	Branch, %
appCruise	100% (19/19)	81% (82/101)	70% (255/361)	39% (34/86)
CarSimulator	100% (1/1)	73% (11/15)	48% (52/107)	32% (13/40)
CarSimulatorTest	100% (1/1)	100% (7/7)	100% (21/21)	50% (5/10)
CarSpeed	100% (0/0)	100% (0/0)	100% (0/0)	100% (0/0)
Controller	100% (1/1)	100% (8/8)	90% (18/20)	50% (8/16)
ControllerTest	100% (2/2)	83% (10/12)	92% (26/28)	100% (0/0)
CruiseControl	100% (8/8)	50% (9/18)	75% (42/56)	100% (0/0)
CruiseControlTest	100% (1/1)	100% (8/8)	100% (22/22)	100% (0/0)
CruiseDisplay	100% (1/1)	62% (5/8)	28% (13/45)	8% (1/12)
CruiseDisplayTest	100% (1/1)	100% (10/10)	100% (26/26)	100% (0/0)
SpeedControl	100% (1/1)	100% (6/6)	100% (19/19)	87% (7/8)
SpeedControlTest	100% (2/2)	88% (8/9)	94% (16/17)	100% (0/0)