

Program Listings: Semester 2

AWT One

```
import java.awt.Toolkit;
import java.awt.Dimension;

class awtone {
    public static void main(String args[])
    {
        Toolkit defTk = Toolkit.getDefaultToolkit();

        String name = System.getProperty("awt.toolkit");
        System.out.println("Toolkit name: " + name);

        Dimension screen = defTk.getScreenSize();
        System.out.println("Screen Dimension      : " +
            screen.toString());
        System.out.println("Screen Resolution (dpi): " +
            defTk.getScreenResolution());
        System.out.println("Font List.");
        String fonts[] = defTk.getFontList();
        for (int i=0; i < fonts.length; i++)
            System.out.println(i + " " + fonts[i]);

        System.exit(0);
    }
}
```

AWT Two

```
import java.awt.Button;
import java.awt.peer.ComponentPeer;
import java.awt.Frame;

class awttwo {
    public static void main(String args[])
    {
        Button myButton = new Button("my Button");
        ComponentPeer buttonPeer = myButton.getPeer();

        if (buttonPeer == null)
        {
            System.out.println("Button peer not yet created.");
            System.out.println("Button is: " + myButton.toString());
        }
        else
        {
            System.out.println("Button Peer is Created!");
            System.out.println(buttonPeer.toString());
        }

        Frame myFrame = new Frame("my Frame");
        myFrame.add("Center", myButton);
        myFrame.pack();
        myFrame.show(); // Here is where peer will be created.

        buttonPeer = myButton.getPeer();
        if (buttonPeer == null)
        {
            System.out.println("Button peer not yet created.");
            System.out.println("Button is: " + myButton.toString());
        }
        else
        {
            System.out.println("Button Peer is Created!");
            System.out.println(buttonPeer.toString());
        }

        ComponentPeer framePeer = myFrame.getPeer();
        System.out.println("Frame Peer is also created.");
        System.out.println(framePeer.toString());
    }
}
```

AWT Three

```
import java.awt.Frame;
import java.awt.Button;
import java.awt.Label;
import java.awt.Checkbox;
import java.awt.List;
import java.awt.FlowLayout;
import java.awt.BorderLayout;
import java.awt.Panel;
import java.awt.Choice;
import java.awt.Canvas;
import java.awt.Color;
import java.awt.Event;
import java.awt.TextArea;
import java.awt.TextField;
import java.awt.Scrollbar;

class componentHolder extends Frame {
    componentHolder()
    {
        super("AWT Basic Components");
        setLayout(new BorderLayout());

        Panel topP = new Panel();
        topP.setLayout(new FlowLayout(FlowLayout.LEFT));
        topP.add(new Button("my Button"));
        topP.add(new Label("my Label"));
        topP.add(new Checkbox("my CheckBox"));
        add("North", topP);

        Panel midP = new Panel();
        midP.setLayout(new FlowLayout(FlowLayout.LEFT));
        List l = null;
        midP.add(l = new List(5,true));
        l.addItem("listItem1");
        l.addItem("listItem2");
        l.addItem("listItem3");
        Choice c = null;
        midP.add(c = new Choice());
        c.addItem("choiceItem1");
        c.addItem("choiceItem2");
        c.addItem("choiceItem3");
        Canvas can;
        midP.add(can = new Canvas());
        can.setBackground(Color.white);
        add("Center", midP);

        Panel botP = new Panel();
        botP.setLayout(new FlowLayout(FlowLayout.LEFT));
        botP.add(new TextField("my text field.));
        botP.add(new TextArea("my text Area.));
        botP.add(new Scrollbar(Scrollbar.VERTICAL));
        add("South", botP);

        pack();
        show();
    }

    public boolean handleEvent(Event evt)
    {
        switch (evt.id) {
            case Event.WINDOW_DESTROY:
                System.exit(0);
        }
        return false;
    }
}

class awtthree {
    public static void main(String args[])
    {
        new componentHolder();
    }
}
```



```

        else
            {
                System.out.println("event vector is empty.");
            }
        }
        break;
case Event.WINDOW_DESTROY:
    if (print)
        System.out.println("WINDOW_DESTROY");
    break;
case Event.WINDOW_EXPOSE:
    if (print)
        System.out.println("WINDOW_EXPOSE");
    break;
case Event.WINDOW_ICONIFY:
    if (print)
        System.out.println("WINDOW_ICONIFY");
    break;
case Event.WINDOW_DEICONIFY:
    if (print)
        System.out.println("WINDOW_DEICONIFY");
    break;
case Event.WINDOW_MOVED:
    if (print)
        System.out.println("WINDOW_MOVED");
    break;
case Event.KEY_PRESS:
    if (print)
        System.out.println("KEY_PRESS");
    break;
case Event.KEY_RELEASE:
    if (print)
        System.out.println("KEY_RELEASE");
    break;
case Event.KEY_ACTION:
    if (print)
        System.out.println("KEY_ACTION");
    break;
case Event.KEY_ACTION_RELEASE:
    if (print)
        System.out.println("KEY_ACTION_RELEASE");
    break;
case Event.MOUSE_DOWN:
    if (print)
        System.out.println("MOUSE_DOWN");
    break;
case Event.MOUSE_UP:
    if (print)
        System.out.println("MOUSE_UP");
    break;
case Event.MOUSE_MOVE:
    if (print)
    {
        System.out.print(" (" + evt.x + ", " +
            evt.y + ")");
        System.out.flush();
    }
    break;
case Event.MOUSE_ENTER:
    if (print)
        System.out.println("MOUSE_ENTER");
    break;
case Event.MOUSE_EXIT:
    if (print)
        System.out.println("MOUSE_EXIT");
    break;
case Event.MOUSE_DRAG:
    if (print)
        System.out.println("MOUSE_DRAG");
    break;
case Event.SCROLL_LINE_UP:
    if (print)
        System.out.println("SCROLL_LINE_UP");
    break;
case Event.SCROLL_LINE_DOWN:
    if (print)
        System.out.println("SCROLL_LINE_DOWN");
    break;

```

```

        case Event.SCROLL_PAGE_UP:
            if (print)
                System.out.println("SCROLL_PAGE_UP");
            break;
        case Event.SCROLL_ABSOLUTE:
            if (print)
                System.out.println("SCROLL_ABSOLUTE");
            break;
        case Event.LIST_SELECT:
            if (print)
                System.out.println("LIST_SELECT");
            break;
        case Event.LIST_DESELECT:
            if (print)
                System.out.println("LIST_DESELECT");
            break;
        case Event.LOAD_FILE:
            if (print)
                System.out.println("LOAD_FILE");
            break;
        case Event.SAVE_FILE:
            if (print)
                System.out.println("SAVE_FILE");
            break;
        case Event.GOT_FOCUS:
            if (print)
                System.out.println("GOT_FOCUS");
            break;
        case Event.LOST_FOCUS:
            if (print)
                System.out.println("LOST_FOCUS");
            break;
        default:
            System.out.println("*** Unknown Event ***");
            System.out.println(evt.toString());
    }
    if (capture)
    {
        Event newEvt = new Event(evt.target, evt.when,
                                evt.id, evt.x, evt.y,
                                evt.key, evt.modifiers,
                                evt.arg);
        evtVector.addElement(newEvt);
    }
    return true;
}
}

class awtfour {
    public static void main(String args[])
    {
        new eventWindow();
    }
}

```

AWT Five

```
import java.awt.Frame;
import java.awt.MenuBar;
import java.awt.Menu;
import java.awt.MenuItem;
import java.awt.Window;
import java.awt.Dialog;
import java.awt.FileDialog;
import java.awt.Event;

class topLevel extends Frame {
    topLevel()
    {
        super("AWT Windows");

        MenuBar mb = new MenuBar();
        setMenuBar(mb);

        Menu fileMenu = new Menu("File");
        mb.add(fileMenu);
        fileMenu.add(new MenuItem("Frame"));
        fileMenu.add(new MenuItem("Window"));
        fileMenu.add(new MenuItem("Modeless Dialog"));
        fileMenu.add(new MenuItem("Modal Dialog"));
        fileMenu.add(new MenuItem("FileDialog"));
        fileMenu.add(new MenuItem("Exit"));
        move(100,100);
        resize(200,200);
        show();
    }

    public boolean handleEvent(Event evt)
    {
        switch (evt.id)
        {
            case Event.ACTION_EVENT:
                if ("Exit".equals(evt.arg))
                {
                    System.exit(0);
                }
                else if ("Frame".equals(evt.arg))
                {
                    new bareFrame();
                    return true;
                }
                else if ("Window".equals(evt.arg))
                {
                    new bareWindow(this);
                    return true;
                }
                else if ("Modeless Dialog".equals(evt.arg))
                {
                    new bareDialog(this,false,"Modeless Dialog");
                    return true;
                }
                else if ("Modal Dialog".equals(evt.arg))
                {
                    new bareDialog(this,true,"Modal Dialog");
                    return true;
                }
                else if ("FileDialog".equals(evt.arg))
                {
                    new bareFileDialog(this);
                    return true;
                }
                break;
            case Event.WINDOW_DESTROY:
                System.exit(0);
                break;
            default:
                return false;
        }
        return false;
    }
}
```

```

class bareFrame extends Frame {
    bareFrame()
    {
        super("A Frame");
        move(110,150);
        resize(150,150);
        show();
    }

    public boolean handleEvent(Event evt)
    {
        switch (evt.id)
        {
            case Event.WINDOW_DESTROY:
                dispose();
                break;
            default:
                return false;
        }
        return false;
    }
}

class bareWindow extends Window {
    bareWindow(Frame parent)
    {
        super(parent);
        move(110,150);
        resize(150,150);
        show();
    }

    public boolean handleEvent(Event evt)
    {
        switch (evt.id)
        {
            case Event.MOUSE_DOWN:
                dispose();
                break;
            case Event.WINDOW_DESTROY:
                dispose();
                break;
            default:
                return false;
        }
        return false;
    }
}

class bareDialog extends Dialog {
    bareDialog(Frame parent, boolean modal, String title)
    {
        super(parent,modal);
        setTitle(title);
        move(110,150);
        resize(200,150);
        show();
    }

    public boolean handleEvent(Event evt)
    {
        switch (evt.id)
        {
            case Event.WINDOW_DESTROY:
                dispose();
                break;
            default:
                return false;
        }
        return false;
    }
}

```

```

class bareFileDialog extends FileDialog {
    bareFileDialog(Frame parent)
    {
        super(parent,"A File Dialog");
        move(110,150);
        show();
    }

    public boolean handleEvent(Event evt)
    {
        switch (evt.id)
        {
            case Event.WINDOW_DESTROY:
                dispose();
                break;
            case Event.ACTION_EVENT:
                if ("Cancel".equals(evt.arg))
                    dispose();
                break;
            default:
                return false;
        }
        return false;
    }
}

class awtfive {
    public static void main(String args[])
    {
        new topLevel();
    }
}

```

AWT Six

```

import java.awt.*;

class myPanel extends Panel {
    myPanel()
    {
        setBackground(Color.white);
    }

    void printGC()
    {
        Graphics GC = this.getGraphics();
        if (GC != null)
        {
            System.out.println("A Graphics Context exists for this component.");
            System.out.println("The GC is : "
                + GC.toString());
            System.out.println("OK, here's a circle for you.");
            GC.setColor(Color.red);
            GC.drawOval(25,25, 50,50);
        }
        else
            System.out.println("No Graphics context for this" +
                " component.");
    }
}

class awtsix {
    public static void main(String args[])
    {
        Frame myFrame = new Frame("Graphics Context test");
        myFrame.resize(125,125);
        myPanel aPanel = new myPanel();
        myFrame.add("Center",aPanel);
        myFrame.show(); // GC created
        aPanel.printGC();
    }
}

```

F1Driver Class

```
package java.bangor;

/* definition of a simple class for F1 drivers */

public class F1driver
{
private String name;
private int points;
private int wins;
private int champBonus;

private static int raceNumber;

public static int firstPlacePoints=10;
public static int secondPlacePoints=6;
public static int thirdPlacePoints=4;
public static int fourthPlacePoints=3;
public static int fifthPlacePoints=2;
public static int sixthPlacePoints=1;
public static int otherPlacePoints=0;

public String getName()
{
return name;
}

public void setName(String aName)
{
name = aName;
}

public int getPoints()
{
return points;
}

public void setPoints(int aPoint)
{
points = aPoint;
}

public int getWCB()
{
return champBonus;
}

public void setWCB(int aBonus)
{
champBonus = aBonus;
}

public int getWins()
{
return wins;
}

public void setWins(int aWin)
{
wins = aWin;
}
}
```

Grand Prix Program 1

```
/* A program that presents some F1 driver information.
   I was inspired by last year's disappointing result */

import java.bangor.*;

public class GrandPrix98
{
public static void main(String arg[])
{
    Fldriver mika = new Fldriver();
    Fldriver michael = new Fldriver();

    mika.setName("Mika Hakkinen");
    michael.setName("Michael Schumacher");

    System.out.print("I support ");
    System.out.print(michael.getName());
}
}
```

Grand Prix Program 2

```
/**
 *
 * Lab 11, Exercise 3
 *
 * Written by Gareth Evans
 *
 * 11/Feb/99
 */

/* A program that presents some F1 driver information.
   I was inspired by last year's disappointing result */

import java.bangor.*;

public class GrandPrix98
{
    /**
     * main
     */
    public static void main(String arg[])
    {
        Fldriver mika = new Fldriver();
        Fldriver michael = new Fldriver();

        int raceNumber;
        raceNumber = 16; /* this is where we are in the championship */

        int firstPlace = 10; /* a win scores 10 points */

        mika.setPoints(90);
        michael.setPoints(86);
        mika.setName("Mika Hakkinen");
        michael.setName("Michael Schumacher");

        System.out.print("I support ");
        System.out.println(michael.getName());
        System.out.println(); /* print out the situation */

        System.out.print("Standings - Race ");
        System.out.println(raceNumber);
        System.out.print(mika.getName());
        System.out.print(" - ");
        System.out.println(mika.getPoints());
        System.out.print(michael.getName());
        System.out.print(" - ");
        System.out.println(michael.getPoints());

        /* Update situation after the next (final) race.
           Mika Hakkinen won, so he should get the points for a win. */

        raceNumber = raceNumber + 1; /* we increase the race number by 1 */
        mika.setPoints( mika.getPoints() + firstPlace );

        /* print out the situation */

        System.out.println();
        System.out.print("Standings - Race ");
        System.out.println(raceNumber);
        System.out.print(mika.getName());
        System.out.print(" - ");
        System.out.println(mika.getPoints());
        System.out.print(michael.getName());
        System.out.print(" - ");
        System.out.println(michael.getPoints());

        /* end of code */
    }
}
```

Grand Prix Program 3

```
/**
 *
 * Lab 11, Exercise 4
 *
 * Written by Gareth Evans
 *
 * 11/Feb/99
 */

/* A program that presents some F1 driver information.
   I was inspired by last year's disappointing result */

// Exercise 3

import java.bangor.*;

public class GrandPrix98
{
    /**
     * main
     */

    public static void main(String arg[])
    {
        Fldriver mika = new Fldriver();
        Fldriver michael = new Fldriver();

        int raceNumber;
        int WCBonus;
        raceNumber = 16; /* this is where we are in the championship */
        WCBonus = 500000;

        int firstPlace = 10; /* a win scores 10 points */

        mika.setPoints(90);
        michael.setPoints(86);
        mika.setName("Mika Hakkinen");
        michael.setName("Michael Schumacher");
        mika.setWCB(0);
        michael.setWCB(0);

        System.out.print("I support ");
        System.out.println(michael.getName());
        System.out.println();

        /* print out the situation */

        System.out.print("Standings - Race ");
        System.out.println(raceNumber);
        System.out.print(mika.getName());
        System.out.print(" - ");
        System.out.println(mika.getPoints());
        System.out.print(michael.getName());
        System.out.print(" - ");
        System.out.println(michael.getPoints());

        /* Update situation after the next (final) race.
           Mika Hakkinen won, so he should get the points for a win. */

        raceNumber = raceNumber + 1; /* we increase the race number by 1 */
        mika.setPoints( mika.getPoints() + firstPlace );
    }
}
```

```

/* print out the situation */

System.out.println();
System.out.print("Standings - Race ");
System.out.println(raceNumber);
System.out.print(mika.getName());
System.out.print(" - ");
System.out.println(mika.getPoints());
System.out.print(michael.getName());
System.out.print(" - ");
System.out.println(michael.getPoints());

// Assign and Display Bonuses

if ((mika.getPoints())>(michael.getPoints()))
    mika.setWCB(1);

if ((michael.getPoints())>(mika.getPoints()))
    michael.setWCB(1);

System.out.println();
System.out.println("Bonuses:");
System.out.println();

if ((mika.getWCB())==1)
    System.out.println("Mika Hakkinen Bonus = "+WCBonus+" Pounds");
else
    System.out.println("Mika Hakkinen Bonus = 0");

if ((michael.getWCB())==1)
    System.out.println("Michael Schumacher Bonus = "+WCBonus+" Pounds");
else
    System.out.println("Michael Schumacher Bonus = 0");

/* end of code */

}
}

```

Grand Prix Program 4

```
/*
 *
 * Gareth Evans
 *
 * 18th February 1999
 *
 * Exercise 4
 *
 * Simulation of a 5 race season
 *
 */

/* "A program that presents some F1 driver information.
   I was inspired by last year's disappointing result" */

import java.util.* ;
import java.bangor.*;

public class GrandPrix98
{
    /**
     *
     * main
     *
     */

    public static void main(String arg[]) throws Exception
    {

        Fldriver mika = new Fldriver();
        Fldriver michael = new Fldriver();

        int WCBonus;
        WCBonus = 500000;

        int racenumber;
        int input;
        int mikatemp;
        int michaeltemp;
        int mikasponsor=0;
        int michaelsponsor=0;
        int winstemp;
        int championship=length=5;

        mika.setPoints(0);
        michael.setPoints(0);
        mika.setName("Mika Hakkinen");
        michael.setName("Michael Schumacher");
        mika.setWCB(0);
        michael.setWCB(0);

        /* simulate the 5 race season */

        racenumber=1;
        System.out.println();

        for (int count=0; count<championship; count++)
        {
            System.out.println("This is race number " + racenumber);
            System.out.println();

            mikatemp = mika.getPoints();
            michaeltemp = michael.getPoints();

            BasicIo.prompt("What was Michael Schumacher's Position in the race? ");
            input = BasicIo.readInteger();
        }
    }
}
```

```

switch (input)
{
case 1:  michael.setPoints(michaeltemp + Fldriver.firstPlacePoints);
        winstemp = michael.getWins();
        winstemp++;
        michael.setWins(winstemp);
        break;
case 2:  michael.setPoints(michaeltemp + Fldriver.secondPlacePoints);
        break;
case 3:  michael.setPoints(michaeltemp + Fldriver.thirdPlacePoints);
        break;
case 4:  michael.setPoints(michaeltemp + Fldriver.fourthPlacePoints);
        break;
case 5:  michael.setPoints(michaeltemp + Fldriver.fifthPlacePoints);
        break;
case 6:  michael.setPoints(michaeltemp + Fldriver.sixthPlacePoints);
        break;
default: break;
} ;

System.out.println();

BasicIo.prompt("What was Mika Hakkinen's Position in the race? ");
input = BasicIo.readInteger();

switch (input)
{
case 1:  mika.setPoints(mikatemp + Fldriver.firstPlacePoints);
        winstemp = mika.getWins();
        winstemp++;
        mika.setWins(winstemp);
        break;
case 2:  mika.setPoints(mikatemp + Fldriver.secondPlacePoints);
        break;
case 3:  mika.setPoints(mikatemp + Fldriver.thirdPlacePoints);
        break;
case 4:  mika.setPoints(mikatemp + Fldriver.fourthPlacePoints);
        break;
case 5:  mika.setPoints(mikatemp + Fldriver.fifthPlacePoints);
        break;
case 6:  mika.setPoints(mikatemp + Fldriver.sixthPlacePoints);
        break;
default: break;
} ;

// Show the situation

System.out.println();
System.out.println("Standings - Race " + racenumber);
System.out.println();
System.out.print(mika.getName());
System.out.print(" - ");
System.out.println(mika.getPoints() + " pts");
System.out.print(michael.getName());
System.out.print(" - ");
System.out.println(michael.getPoints() + " pts");
System.out.println();

mikatemps=mika.getPoints();
michaeltemp=michael.getPoints();

if (mikatemps>michaeltemp)
{
    System.out.println("Mika Hakkinen is winning the championship");
}

if (michaeltemp>mikatemps)
{
    System.out.println("Michael Schumacher is winning the championship");
}

```

```

if (michaeltemp == mikatemp)
{
    michaeltemp = michael.getWins();
    mikatemp = mika.getWins();

    if (mikatempmichaeltemp)
    {
        System.out.println("Mika has won more races than Michael, ");
        System.out.println("Therefore Mika in leading the championship");
    }

    if (michaeltempmikatempm)
    {
        System.out.println("Michael has won more races than Mika, ");
        System.out.println("Therefore Michael in leading the championship");
    }

    if (mikatempmichaeltemp)
    {
        System.out.println("It is a draw at this stage!");
    }
}

racenumber++;
System.out.println();
System.out.println();
}

System.out.println();
System.out.println("Standings - After the end of our championship ");
System.out.println();
System.out.print(mika.getName());
System.out.print(" - ");
System.out.println(mika.getPoints() + " pts");
System.out.print(michael.getName());
System.out.print(" - ");
System.out.println(michael.getPoints() + " pts");

// Assign and Display Bonuses

if (mika.getPoints()michael.getPoints())
{
    mika.setWCB(1);
    System.out.println("Mika Hakkinen wins the championship");
}

if (michael.getPoints()mika.getPoints())
{
    michael.setWCB(1);
    System.out.println("Michael Schumacher wins the championship");
}

if (michael.getPoints() == mika.getPoints())
{
    if (mika.getWins()michael.getWins())
    {
        mika.setWCB(1);
        System.out.println("Mika has won more races than Michael, ");
        System.out.println("Therefore Mika wins the championship");
    }

    if (mika.getWins()michael.getWins())
    {
        michael.setWCB(1);
        System.out.println("Michael has won more races than Mika, ");
        System.out.println("Therefore Michael wins the championship");
    }

    if (mika.getWins()michael.getWins())
    {
        System.out.println("It is a draw!");
    }
}

System.out.println();
System.out.println("Bonuses:");
System.out.println();

```

```
if ((mika.getWCB()==1)
    System.out.println("Mika Hakkinen Bonus = "+WCBonus+" Pounds");
else
    System.out.println("Mika Hakkinen Bonus = 0");

if ((michael.getWCB()==1)
    System.out.println("Michael Schumacher Bonus = "+WCBonus+" Pounds");
else
    System.out.println("Michael Schumacher Bonus = 0");

/* end of code */

}
}
```

Grand Prix Program 5

```
/*
 *
 * Gareth Evans
 *
 * 18th February 1999
 *
 * Exercise 5
 *
 * Simulation of a 5 race season
 * And assigns "Sponsorship" bonuses
 *
 */

/* "A program that presents some F1 driver information.
   I was inspired by last year's disappointing result" */

import java.util.* ;
import java.bangor.*;

public class GrandPrix98
{
    /**
     *
     * main
     *
     */

    public static void main(String arg[]) throws Exception
    {
        Fldriver mika = new Fldriver();
        Fldriver michael = new Fldriver();

        int WCBonus;
        WCBonus = 500000;

        int racenumber;
        int input;
        int mikatemp;
        int michaeltemp;
        int mikasponsor=0;
        int michaelsponsor=0;
        int winstemp;
        int championshiplength=5;

        mika.setPoints(0);
        michael.setPoints(0);
        mika.setName("Mika Hakkinen");
        michael.setName("Michael Schumacher");
        mika.setWCB(0);
        michael.setWCB(0);

        /* simulate the 5 race season */

        racenumber=1;
        System.out.println();

        for (int count=0; count<championshiplength; count++)
        {
            System.out.println("This is race number " + racenumber);
            System.out.println();

            mikatemp = mika.getPoints();
            michaeltemp = michael.getPoints();

            BasicIo.prompt("What was Michael Schumacher's Position in the race? ");
            input = BasicIo.readInteger();
        }
    }
}
```

```

switch (input)
{
case 1:  michael.setPoints(michaeltemp + Fldriver.firstPlacePoints);
        winstemp = michael.getWins();
        winstemp++;
        michael.setWins(winstemp);
        michael.sponsor=michael.sponsor+10000;
        break;
case 2:  michael.setPoints(michaeltemp + Fldriver.secondPlacePoints);
        michael.sponsor=michael.sponsor+10000;
        break;
case 3:  michael.setPoints(michaeltemp + Fldriver.thirdPlacePoints);
        michael.sponsor=michael.sponsor+10000;
        break;
case 4:  michael.setPoints(michaeltemp + Fldriver.fourthPlacePoints);
        break;
case 5:  michael.setPoints(michaeltemp + Fldriver.fifthPlacePoints);
        break;
case 6:  michael.setPoints(michaeltemp + Fldriver.sixthPlacePoints);
        break;
default: break;
} ;

System.out.println();

BasicIo.prompt("What was Mika Hakkinen's Position in the race? ");
input = BasicIo.readInteger();

switch (input)
{
case 1:  mika.setPoints(mikatemp + Fldriver.firstPlacePoints);
        winstemp = mika.getWins();
        winstemp++;
        mika.setWins(winstemp);
        mika.sponsor=mika.sponsor+10000;
        break;
case 2:  mika.setPoints(mikatemp + Fldriver.secondPlacePoints);
        mika.sponsor=mika.sponsor+10000;
        break;
case 3:  mika.setPoints(mikatemp + Fldriver.thirdPlacePoints);
        mika.sponsor=mika.sponsor+10000;
        break;
case 4:  mika.setPoints(mikatemp + Fldriver.fourthPlacePoints);
        break;
case 5:  mika.setPoints(mikatemp + Fldriver.fifthPlacePoints);
        break;
case 6:  mika.setPoints(mikatemp + Fldriver.sixthPlacePoints);
        break;
default: break;
} ;

// Show the situation

System.out.println();
System.out.println("Standings - Race " + racenumber);
System.out.println();
System.out.print(mika.getName());
System.out.print(" - ");
System.out.println(mika.getPoints() + " pts");
System.out.print(michael.getName());
System.out.print(" - ");
System.out.println(michael.getPoints() + " pts");
System.out.println();

mikatemps=mika.getPoints();
michaeltemp=michael.getPoints();

if (mikatemps>michaeltemp)
{
    System.out.println("Mika Hakkinen is winning the championship");
}

if (michaeltemp>mikatemps)
{
    System.out.println("Michael Schumacher is winning the championship");
}

```

```

if (michaeltemp == mikatemp)
{
    michaeltemp = michael.getWins();
    mikatemp = mika.getWins();

    if (mikatemp>michaeltemp)
    {
        System.out.println("Mika has won more races than Michael, ");
        System.out.println("Therefore Mika in leading the championship");
    }

    if (michaeltemp>mikatemp)
    {
        System.out.println("Michael has won more races than Mika, ");
        System.out.println("Therefore Michael in leading the championship");
    }

    if (mikatemp==michaeltemp)
    {
        System.out.println("It is a draw at this stage!");
    }
}

racenumber++;
System.out.println();
System.out.println();
}

System.out.println();
System.out.println("Standings - After the end of our championship ");
System.out.println();
System.out.print(mika.getName());
System.out.print(" - ");
System.out.println(mika.getPoints() + " pts");
System.out.print(michael.getName());
System.out.print(" - ");
System.out.println(michael.getPoints() + " pts");

// Assign and Display Bonuses

if (mika.getPoints()>michael.getPoints())
{
    mika.setWCB(1);
    System.out.println("Mika Hakkinen wins the championship");
}

if (michael.getPoints()>mika.getPoints())
{
    michael.setWCB(1);
    System.out.println("Michael Schumacher wins the championship");
}

if (michael.getPoints() == mika.getPoints())
{
    if (mika.getWins()>michael.getWins())
    {
        mika.setWCB(1);
        System.out.println("Mika has won more races than Michael, ");
        System.out.println("Therefore Mika wins the championship");
    }

    if (michael.getWins()>mika.getWins())
    {
        michael.setWCB(1);
        System.out.println("Michael has won more races than Mika, ");
        System.out.println("Therefore Michael wins the championship");
    }

    if (mika.getWins()==michael.getWins())
    {
        System.out.println("It is a draw!");
    }
}
}

```

```

System.out.println();
System.out.println("Bonuses:");
System.out.println();

if ((mika.getWCB()==1)
    System.out.println("Mika Hakkinen Bonus = "+WCBonus+" Pounds");
else
    System.out.println("Mika Hakkinen Bonus = 0");

if ((michael.getWCB()==1)
    System.out.println("Michael Schumacher Bonus = "+WCBonus+" Pounds");
else
    System.out.println("Michael Schumacher Bonus = 0");

// Set and Show Sponsorship Bonuses

mikatemps = mika.getPoints();
michaeltemp = michael.getPoints();

if ((mika.getWCB()==1)
{
    if (mikatemps>69)
    {
        if ((mikatemps-michaeltemp)>5)
        {
            int temp;
            temp = mikatemps-michaeltemp;
            temp = temp*5000;
            mikasponsor=mikasponsor+temp;
        }
    }
}

if ((michael.getWCB()==1)
{
    if (michaeltemp>69)
    {
        if ((michaeltemp-mikatemps)>5)
        {
            int temp;
            temp = michaeltemp-mikatemps;
            temp = temp*5000;
            michaelssponsor=michaelssponsor+temp;
        }
    }
}

System.out.println();
System.out.println("Sponsorship Bonuses:");
System.out.println();
System.out.println("Michael Schumacher: " + michaelssponsor + " pounds");
System.out.println("Mika Hakkinen: " + mikasponsor + " pounds");

/* end of code */

}
}

```

Grand Prix Program 6

See the Project Handed In...

Exam Results Program (Arrays)

```
/**
 *
 * Lab 12, Exercise 1
 *
 * Written by Gareth Evans
 *
 * 25/Feb/99
 *
 */

import java.util.* ;
import java.bangor.*;

public class ExamResults
{
    /**
     *
     * main
     *
     */

    public static void main(String arg[]) throws Exception
    {

        // Enter the Marks

        int[] examMarks;
        examMarks = new int[12];

        for (int i = 0; i < 12; i++)
        {
            do
            {
                System.out.println("Please enter Exam Mark for exam number " + (i+1));
                examMarks[i] = BasicIo.readInteger();
            }
            while (examMarks[i] < 0 && examMarks[i] > 101);
        }

        // Display the Highest Mark

        int Highest;
        Highest = 1;

        for (int k = 1; k < 12; k++)
        {
            if (examMarks[k] > examMarks[(Highest-1)])
            { Highest = k+1; }
        }

        System.out.println();
        System.out.println("The Highest mark was mark " + Highest + ", this was " +
examMarks[Highest-1]);

        System.out.println();

        // Display the lowest mark

        int Lowest;
        Lowest = 1;

        for (int l = 1; l < 12; l++)
        {
            if (examMarks[l] < examMarks[(Lowest-1)])
            { Lowest = l+1; }
        }

        System.out.println();
        System.out.println("The Lowest mark was mark " + Lowest + ", this was " +
examMarks[Lowest-1]);
    }
}
```

```
System.out.println();

// Display the average mark

int Sum;
Sum = 0;

for (int m = 0; m < 12; m++)
{
    Sum = Sum + examMarks[m];
}

Sum = Sum/12;

System.out.println();
System.out.println("The average mark was " + Sum);
System.out.println();

// Display how many marks were above 60

int Count;
Count = 0;

for (int n = 0; n < 12; n++)
{
    if (examMarks[n]>60)
    { Count++; }
}

System.out.println();
System.out.println("There were " + Count + " Mark(s) over 60");

/* end of code */
}
}
```

Video Shop Program 1

```
/**
 *
 * Lab 12, Exercise 2
 *
 * Written by Gareth Evans
 *
 * 25/Feb/99
 */

import java.util.* ;
import java.bangor.*;

public class VideoShop
{

    /**
     *
     * main
     *
     */

    public static void main(String arg[]) throws Exception
    {

        // Define the Variables

        int[] Videos;
        Videos = new int[20];

        int Choice;
        int HireVideo;
        int HireCustomer;
        int ReturnVideo;
        int ReturnCustomer;
        int CheckVideo;
        int CheckCustomer;

        // Note: The video variable will store the number of videos in stock

        int[] Customers;
        Customers = new int[50];

        // Note: The Customer Variable will store which video the customer
        //         Currently has ( the value -1 = no video)

        // Set the Variable Values

        for (int i = 0; i < 10; i++)
        { Videos[i] = 4; } // We have 4 copies of Videos 0 to 9

        for (int j = 10; j < 20; j++)
        { Videos[j] = 6; } // We have 6 copies of Videos 10 to 19

        for (int k = 0; k < 50; k++)
        { Customers[k] = -1; } // No videos are hired out at the start
    }
}
```

```

// MENU SYSTEM

do
{
    System.out.println();
    System.out.println("MAIN MENU - PLEASE ENTER YOUR CHOICE");
    System.out.println("-----");
    System.out.println();
    System.out.println("1. Video Hire");
    System.out.println("2. Video Return");
    System.out.println("3: Tapes in Stock");
    System.out.println("4. Customer Information");
    System.out.println();
    System.out.println("5. Exit");
    System.out.println();
    Choice = BasicIo.readInteger();

    // 1. VIDEO HIRE

    if (Choice==1)
    {
        System.out.println("Which video do you want to hire?");
        System.out.println("Enter a number between 0 and 19");
        HireVideo = BasicIo.readInteger();

        if (Videos[HireVideo]>0)
        {
            System.out.println();
            System.out.println("Which customer wants to hire this video?");
            System.out.println("Enter a number between 0 and 49");
            HireCustomer = BasicIo.readInteger();

            if ((Customers[HireCustomer]) == (-1))
            {
                Customers[HireCustomer] = HireVideo;
                Videos[HireVideo] = (Videos[HireVideo] - 1);
            }
            else
            {
                System.out.println();
                System.out.println("The Customer selected has already hired a video");
                System.out.println("Please return this video before attempting to hire another video");
            }
        }
        else
        {
            System.out.println();
            System.out.println("There are no copies of this video in stock. Sorry");
        }
    }

    // 2. VIDEO RETURN

    if (Choice==2)
    {
        System.out.println();
        System.out.println("Which Customer is Returning a video?");
        ReturnCustomer = BasicIo.readInteger();
        if (Customers[ReturnCustomer]>-1)
        {
            ReturnVideo = Customers[ReturnCustomer];
            Videos[ReturnVideo] = (Videos[ReturnVideo] + 1);
            Customers[ReturnCustomer] = -1;
        }
        else
        {
            System.out.println();
            System.out.println("The selected customer has no videos to return");
        }
    }
}

```

```

// 3. TAPES IN STOCK

if (Choice==3)
{
    System.out.println();
    for (int i = 0; i < 20; i++)
    {
        System.out.println("Movie " + i + ": " + Videos[i] + " copies");
    }
}

// 4. CUSTOMER INFORMATION

if (Choice==4)
{
    System.out.println();
    System.out.println("Please enter a Customer Number between 0 and 49");
    CheckCustomer = BasicIo.readInteger();
    System.out.println();

    if ((CheckCustomer>-1) && (CheckCustomer<50))
    {
        if (Customers[CheckCustomer] > -1)
        {
            System.out.println("Customer " + CheckCustomer + " currently has Tape " +
                Customers[CheckCustomer]);
        }
        else
        {
            System.out.println("Customer " + CheckCustomer + " currently has No
                Tape");
        }
    }
}

}
while ((Choice<5) && (Choice>0));

/* end of code */
}
}

```

Video Shop Program 2

```
/**
 *
 * Lab 12, Exercise 3
 *
 * Written by Gareth Evans
 *
 * 02/Mar/99
 */

import java.util.* ;
import java.bangor.*;

public class VideoShop
{

    /**
     *
     * main
     *
     */

    public static void main(String arg[]) throws Exception
    {

        // Define the Variables

        int[] Videos;
        Videos = new int[20];

        String[] VideoNames;
        VideoNames = new String[20];
        for (int tempcount1 = 0; tempcount1 < 20; tempcount1++)
        {
            VideoNames[tempcount1] = new String();
        }

        int Choice;
        int HireVideo;
        String HireCustomer = new String();
        int ReturnVideo;
        String ReturnCustomer = new String();
        int CheckVideo;
        String CheckCustomer = new String();

        int VideoHireTemp;
        int VideoReturnTemp;
        int InfoTemp;

        // Note: The video variable will store the number of videos in stock

        int[] Customers;
        Customers = new int[50];

        String[] CustomerNames;
        CustomerNames = new String[50];
        for (int tempcount2 = 0; tempcount2 < CustomerNames.length; tempcount2++)
        {
            CustomerNames[tempcount2] = new String();
        }

        // Note: The Customer Variable will store which video the customer
        //         Currently has ( the value -1 = no video)
```

```

// Set the Variable Values

for (int i = 0; i < 10; i++)
{ Videos[i] = 4; } // We have 4 copies of Videos 0 to 9

for (int j = 10; j < 20; j++)
{ Videos[j] = 6; } // We have 6 copies of Videos 10 to 19

for (int k = 0; k < 50; k++)
{ Customers[k] = -1; } // No videos are hired out at the start

// Set the video names

VideoNames[0] = "Braveheart";
VideoNames[1] = "Apollo 13";
VideoNames[2] = "Aliens 2";
VideoNames[3] = "Predator";
VideoNames[4] = "Sleepless in Seattle";
VideoNames[5] = "Trainspotting";
VideoNames[6] = "The Jungle Book";
VideoNames[7] = "True Lies";
VideoNames[8] = "Tomorrow Never Dies";
VideoNames[9] = "Speed";
VideoNames[10] = "G.I. Jane";
VideoNames[11] = "Ants";
VideoNames[12] = "Lock, Stock & 2 Smoking Barrels";
VideoNames[13] = "Titanic";
VideoNames[14] = "Independence Day";
VideoNames[15] = "Men In Black";
VideoNames[16] = "X Files The Movie";
VideoNames[17] = "A Bug's Life";
VideoNames[18] = "Saving Private Ryan" ;
VideoNames[19] = "RoboCop";

// Set the Customer names; Note - only 10 included; rest could
// be entered via another option in the menu?

CustomerNames[0] = "Gareth Evans";
CustomerNames[1] = "Jim Smith";
CustomerNames[2] = "Alex Ferguson";
CustomerNames[3] = "Kenny Dalglish";
CustomerNames[4] = "Glenn Hoddle";
CustomerNames[5] = "Bobby Gould";
CustomerNames[6] = "Kevin Keegan";
CustomerNames[7] = "Roy Evans";
CustomerNames[8] = "Walter Smith";
CustomerNames[9] = "Ruud Gullit";

// MENU SYSTEM

do
{
    System.out.println();
    System.out.println("MAIN MENU - PLEASE ENTER YOUR CHOICE");
    System.out.println("-----");
    System.out.println();
    System.out.println("1. Video Hire - (Choose option 3 if unsure which video you
    want)");
    System.out.println("2. Video Return");
    System.out.println("3: Tapes in Stock");
    System.out.println("4. Customer Information");
    System.out.println();
    System.out.println("5. Exit");
    System.out.println();
    Choice = BasicIo.readInteger();

    // 1. VIDEO HIRE

    if (Choice==1)
    {
        System.out.println("Which video do you want to hire?");
        System.out.println("Enter a number between 0 and 19");
        HireVideo = BasicIo.readInteger();
    }
}

```

```

if (Videos[HireVideo]>0)
{
    System.out.println();
    System.out.println("Which customer wants to hire this video?");
    System.out.println("Enter the Customer's Full Name");
    HireCustomer = BasicIo.readString();

    VideoHireTemp = -1;

    for (int i = 0; i <= 49; i++)
    {
        if ( CustomerNames[i].equals(HireCustomer) )
        { VideoHireTemp = i; }
    }

    if (VideoHireTemp == -1)
    { System.out.println("Sorry. There is no customer with that name."); }

    if (VideoHireTemp > -1)
    {
        if ((Customers[VideoHireTemp]) == (-1))
        {
            Customers[VideoHireTemp] = HireVideo;
            Videos[HireVideo] = (Videos[HireVideo] - 1);
        }
        else
        {
            System.out.println();
            System.out.println("The Customer selected has already hired a video");
            System.out.println("Please return this video before attempting to hire
            another video");
        }
    }
}
else
{
    System.out.println();
    System.out.println("There are no copies of this video in stock. Sorry");
}
}

// 2. VIDEO RETURN

if (Choice==2)
{
    System.out.println();
    System.out.println("Which Customer is Returning a video?");
    System.out.println("Please type in the Full Name");
    ReturnCustomer = BasicIo.readString();

    VideoReturnTemp = -1;

    for (int i = 0; i <= 49; i++)
    {
        if ( CustomerNames[i].equals(ReturnCustomer) )
        { VideoReturnTemp = i; }
    }

    if (VideoReturnTemp == -1)
    { System.out.println("Sorry. There is no customer with that name."); }

    if (VideoReturnTemp > -1)
    {
        if (Customers[VideoReturnTemp]>-1)
        {
            ReturnVideo = Customers[VideoReturnTemp];
            Videos[ReturnVideo] = (Videos[ReturnVideo] + 1);
            Customers[VideoReturnTemp] = -1;
        }
        else
        {
            System.out.println();
            System.out.println("The selected customer has no videos to return");
        }
    }
}
}

```

```

// 3. TAPES IN STOCK

if (Choice==3)
{
    System.out.println();
    for (int i = 0; i < 20; i++)
    {
        System.out.println(i + ": " + VideoNames[i] + ": " + Videos[i] + " copies");
    }
}

// 4. CUSTOMER INFORMATION

if (Choice==4)
{
    System.out.println();
    System.out.println("Please enter a Customer Name (Full Name)");
    CheckCustomer = BasicIo.readString();
    System.out.println();

    InfoTemp = -1;

    for (int i = 0; i <= 49; i++)
    {
        if ( CustomerNames[i].equals(CheckCustomer) )
        { InfoTemp = i; }
    }

    if (InfoTemp == -1)
    { System.out.println("Sorry. There is no customer with that name."); }

    if ((InfoTemp>-1) && (InfoTemp<50))
    {
        if (Customers[InfoTemp] > -1)
        {
            System.out.println(CheckCustomer + " currently has " +
                VideoNames[Customers[InfoTemp]]);
        }
        else
        {
            System.out.println(CheckCustomer + " currently has No Tape");
        }
    }
}

while ((Choice<5) && (Choice>0));

/* end of code */
}
}

```