

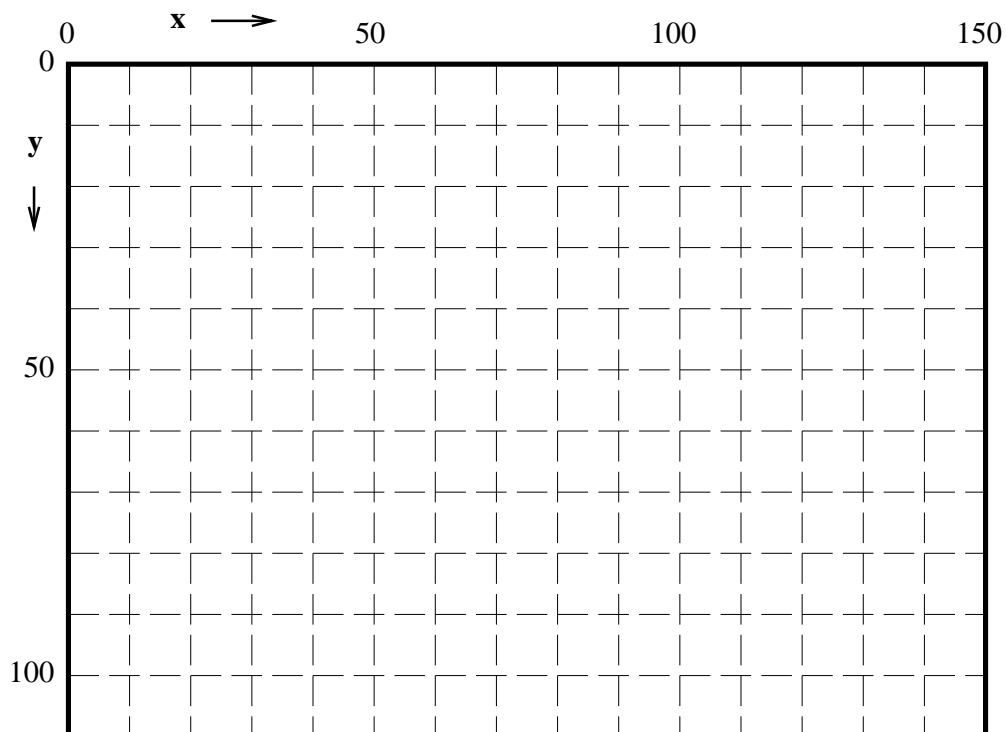
Question 1. Understanding while

[15 marks]

Consider the following program that draws some squares in the graphics pane.

```
import mcs.comp100.*;
public class DrawSomeSquares{
    public static void main(String[ ] args){
        UI window = new UI("SquareMaker");
        double x = 10;
        double y = 10;
        int s = 1;
        while ( s < 8 ) {
            x = y;
            y = y + 10;
            s++;
            window.drawRect(x, y, 5, 5);
        }
    }
}
```

(a) [8 marks] Show what the DrawSomeSquares program will draw on the screen.

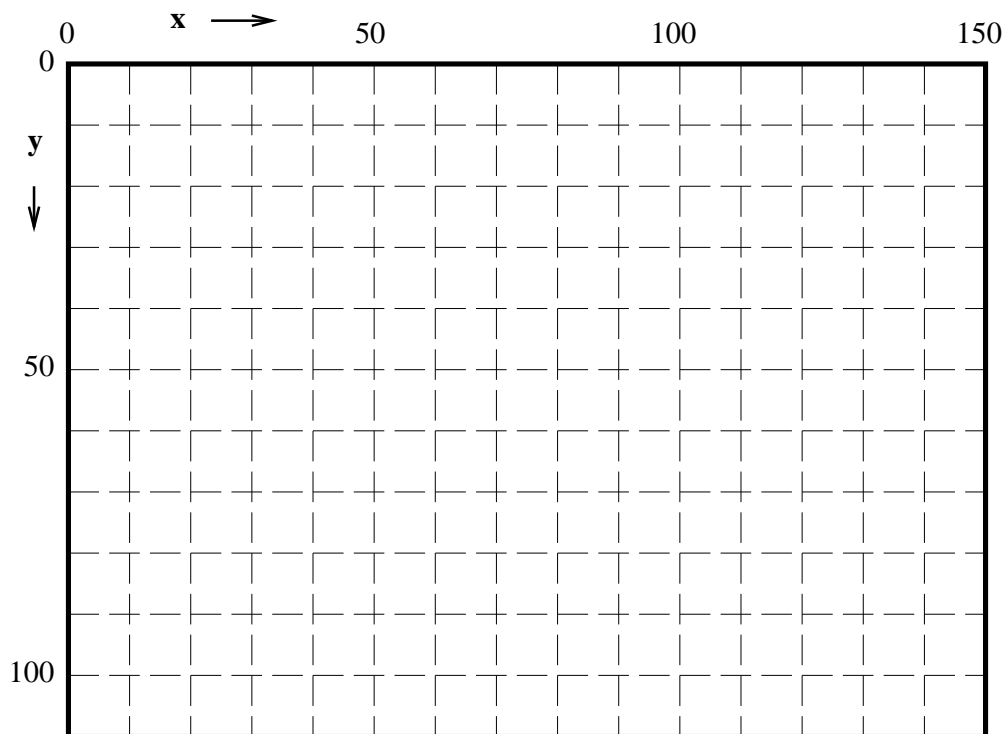


Note: the first two arguments of `drawRect` specify the coordinates of the top left corner of the rectangle; the last two arguments specify its width and its height.

Consider the following program that also draws some squares in the graphics pane.

```
import mcs.comp100.*;
public class DrawManySquares{
    public static void main(String[] args){
        UI window = new UI("SquareMaker");
        double x = 10;
        double y = 10;
        double d = 10;
        while ( d < 45 ) {
            double z = 80-x;
            x = y;
            y = z;
            window.drawRect(x, y, d, d);
            d = d+5;
        }
    }
}
```

(b) [7 marks] Show what the DrawManySquares program will draw on the screen.



Question 2. Understanding Text-Field Input

[10 marks]

Consider the following program that provides two text-fields into which the user can enter values.

```
import mcs.comp100.*;

public class Stories{
    public static void main(String[ ] args){
        UI window = new UI();
        StoryTeller s = new StoryTeller(window);
    }
}

class StoryTeller implements TextFieldListener{
    private UI win;
    private String story;
    private String person;

    public StoryTeller(UI w){
        this.win = w;
        this.win.addTextField("Story", this);
        this.win.addTextField("Hearer", this);
    }

    public void textFieldPerformed(String f, String val){
        if ( f.equals("Story") ){
            this.person = "Someone";
            this.story = val;
            this.win.println(this.person + " heard that " + this.story);
        }
        else if ( f.equals("Hearer") ){
            this.win.println(this.person + " told " + val + " that " + this.story);
            this.person = val;
        }
    }
}
```

(Question 2 continued on next page)

(Question 2 continued)

(a) [4 marks] What would the program print on the screen if the user entered the following data?

Into the "Story" text-field: `The fox stole the hen`
Into the "Hearer" text-field: `Jonathan`
Into the "Hearer" text-field: `Jane`
Into the "Hearer" text-field: `Jeremy`

(b) [3 marks] What would the program print on the screen if, when the program was first started, the user entered the following data?

Into the "Hearer" text-field: `Jonathan`
Into the "Story" text-field: `The fox stole the hen`
Into the "Hearer" text-field: `Jane`

(c) [3 marks] This behaviour (part (b)) is not particularly nice. Make a small change to the Stories program on the opposite page so that the program would have reasonable behaviour with input like that in part (b).

Question 3. Understanding Mouse Input

[10 marks]

Consider the following MouseInput program that allows the user to draw lines on the screen using the mouse.

```
import mcs.comp100.*;

public class MouseInput{
    public static void main(String[ ] args){
        UI window = new UI();
        Mouser m = new Mouser(window);
    }
}

class Mouser implements MouseListener{
    private UI win;
    private double u = 10;
    private double v = 10;

    public Mouser(UI w){
        this.win = w;
        w.addMouseListener(this);
    }

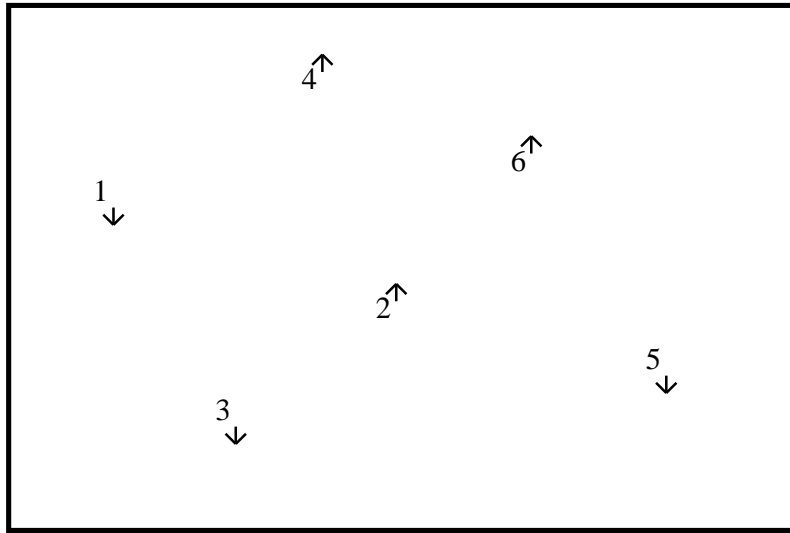
    public void mousePerformed(String act, double x, double y, double lastx, double lasty){
        if ( act.equals("Pressed") ){
            this.win.drawCircle(x, y, 10);
            this.u = x;
        }
        else if ( act.equals("Released") ){
            this.win.drawLine(this.u, this.v, x, y);
            this.v = y;
        }
    }
}
```

Note: The mousePerformed method takes five arguments. The first is the kind of mouse action. The second and third are the coordinates at which the mouse action happened. The fourth and fifth are the coordinates of the point at which the user last pressed the mouse.

(Question 3 continued on next page)

(Question 3 continued)

On the diagram below, show what the MouseInput program will draw, assuming that the user presses and releases the mouse at the points shown below. Each mouse press is shown by a down arrow; each mouse release is shown by an up arrow; the numbers show the order of the events.



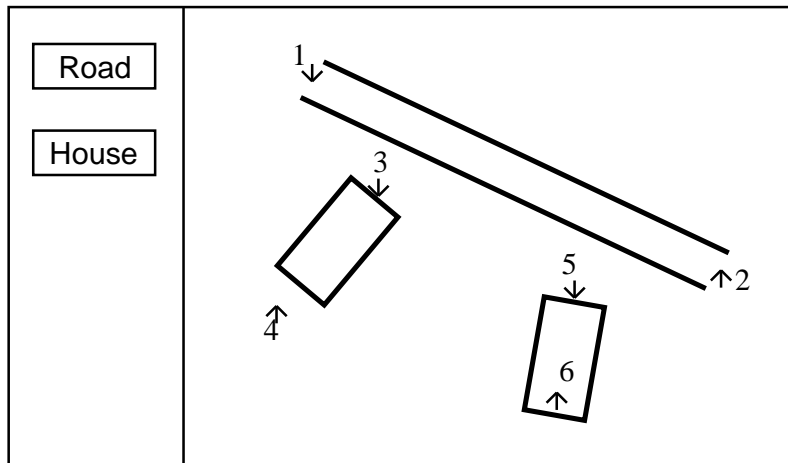
└─ 10 pixels

Question 4. Designing with Input

[17 marks]

This question concerns a program that allows the user to plan a housing subdivision by drawing a plan of roads and houses. The program provides two buttons by which the user can select whether to draw roads or houses next. The user can draw roads by dragging the mouse from one end of the road to the other. The user can draw a house by dragging the mouse from the front of the house in the direction of the middle of the house.

The figure below shows what the program would draw if the user clicked the “Road” button, then dragged the mouse from position 1 to position 2, then clicked the House” button and dragged the mouse from 3 to 4 and then from 5 to 6.



You are to complete the `TownPlanner` program on the facing page. You will need to define additional fields, complete the constructor and the two methods to respond to the buttons and the mouse. Your code may use the two methods `drawRoad` and `drawHouse` for drawing roads and houses. You may assume that these methods have been defined and you do not have to complete them. `drawRoad` and `drawHouse` are described below:

- `public void drawRoad(double x1, double y1, double x2, double y2)`
Draws a road from the position $(x1, y1)$ to the position $(x2, y2)$
- `public void drawHouse(double x1, double y1, double x2, double y2)`
Draws a house. The position $(x1, y1)$ will be in the middle of the front of the house, and the house will be aligned in the direction of a line from $(x1, y1)$ to $(x2, y2)$

```

import mcs.comp100.*;
public class TownPlanner{
    public static void main(String[ ] args){
        UI window = new UI("Town Planner");
        Planner p = new Planner(window);
    }
}

class Planner implements ButtonListener, MouseListener{
    private UI win;

    public Planner(UI w){

    }

    public void buttonPerformed(String but){

    }

    public void mousePerformed(String act, double x, double y, double lastx, double lasty){

    }

    public void drawRoad(double x1, double y1, double x2, double y2){
        Documented on the facing page. You do not need to define this method.
    }

    public void drawHouse(double x1, double y1, double x2, double y2){
        Documented on the facing page. You do not need to define this method.
    }
}

```

Question 5. Designing with while

[20 marks]

For this question, you are to complete one method of a program for predicting the number of customers of two telephone companies. Saturn has just started competing with Telecom in the Wellington region. We will assume that each year, some fraction of Telecom's customers will become unhappy with Telecom and will switch to Saturn for the following year. Similarly, each year some (different) fraction of Saturn's customers will become unhappy with Saturn and will switch to Telecom for the following year. To begin with, more customers will go from Telecom to Saturn than the other way. Eventually, the number of customers switching from each company will be the same and the number of customers for each company will become constant.

The program will assume that Telecom has 100,000 customers in the first year, and Saturn has none. The program should provide number-fields into which the user can enter the fraction of customers that Telecom will lose each year and the fraction of customers that Saturn will lose each year. After the user has entered a value, it should print out a header line followed by a line for each year showing the number of customers of each company, until the number of Saturn customers will not increase any more.

For example, if the user entered the value 0.4 into the "Saturn Loss Rate" number-field, the program should print out the information on the facing page.

Part of the program is given below. The last line calls the `displayCustomers()` method. You are to complete the `displayCustomers()` method on the facing page.

```
import mcs.comp100.*;

public class SaturnCalculator{
    public static void main(String[] args){
        UI window = new UI("Calculator");
        Calculator c = new Calculator(window);
    }
}

class Calculator implements NumberFieldListener{
    private UI win;
    private double telecomLossRate = 0.20;
    private double saturnLossRate = 0.10;
    private int totalCustomers = 100000;

    public Calculator(UI w){
        this.win = w;
        this.win.addNumberField("Telecom Loss Rate", this);
        this.win.addNumberField("Saturn Loss Rate", this);
    }

    public void numberFieldPerformed(String f, double val){
        if ( f.equals("Telecom Loss Rate") )    this.telecomLossRate = val;
        else if ( f.equals("Saturn loss Rate") ) this.saturnLossRate = val;

        this.win.clear();
        this.displayCustomers();
    }
}
```

Example output:

```
Telecom Loss Rate:0.2 Saturn Loss Rate:0.4
Yr:1 Telecom:100000 Saturn:0
Yr:2 Telecom:80000 Saturn:20000
Yr:3 Telecom:72000 Saturn:28000
Yr:4 Telecom:68800 Saturn:31200
Yr:5 Telecom:67520 Saturn:32480
Yr:6 Telecom:67008 Saturn:32992
Yr:7 Telecom:66803 Saturn:33197
Yr:8 Telecom:66721 Saturn:33279
Yr:9 Telecom:66688 Saturn:33312
Yr:10 Telecom:66675 Saturn:33325
Yr:11 Telecom:66670 Saturn:33330
Yr:12 Telecom:66668 Saturn:33332
Yr:13 Telecom:66667 Saturn:33333
Yr:14 Telecom:66666 Saturn:33334
```

Complete the `displayCustomers()` method below.

```
public void displayCustomers() {
    this.win.print("Telecom Loss Rate:" + this.telecomLossRate);
    this.win.println(" Saturn Loss Rate:" + this.saturnLossRate);

}
}
```

Question 6. Designing with Objects

[18 marks]

This question concerns a program for displaying a weather forecast for the Wellington region. The program should display three icons showing the morning weather forecast (a picture of sun, cloud, or rain and the temperature) for three different cities. After a brief pause, it should then change the weather displayed to the forecast for the middle of the day, and then change them again to the forecast for the afternoon.

The program contains two classes:

- WeatherForecast, which contains a main method, and
- WeatherIcon, which represents the weather and temperature at a particular location.

The main method creates three WeatherIcon objects, passing the location (coordinates on the screen), initial weather, and initial temperature to the constructor. The WeatherIcon objects should draw themselves on the screen. The main method then changes the weather and temperature of the icons for mid-day and then for the afternoon, which should redraw themselves at each change.

The WeatherForecast class is shown below. You are to complete the WeatherIcon class on the facing page. You need to define the fields, the constructor, and two methods: changeTemp and changeWeather. Your code should call the draw() method that re-displays the current weather and temperature of the icon on the screen. You may assume that the draw() method is already defined for you — you do **not** have to define it.

```
import mcs.comp100.*;

public class WeatherForecast{
    public static void main(String[ ] args){
        UI win = new UI();

        // Create WeatherIcon objects with morning forecast
        WeatherIcon wellington = new WeatherIcon(win, 200, 300, "Cloud", 13);
        WeatherIcon huttValley = new WeatherIcon(win, 250, 200, "Rain", 12);
        WeatherIcon waikanae = new WeatherIcon(win, 100, 50, "Sun", 17);
        win.sleep(2000);

        // change to mid-day forecast
        wellington.changeTemp(14);
        huttValley.changeWeather("Cloud");
        waikanae.changeTemp(19);
        win.sleep(2000);

        // change to afternoon forecast
        wellington.changeTemp(16);
        huttValley.changeWeather("Sun");
        huttValley.changeTemp(17);
        win.sleep(2000);
    }
}
```

```

class WeatherIcon {
    /* Fields for window, location, weather type, and temperature */
    private UI win;

    /* Constructor */
    public WeatherIcon(UI w,

}

/* changeWeather: changes the kind of weather, and redisplay the WeatherIcon. */
    public void changeWeather(

}

/* changeTemp: changes the temperature, and redisplay the WeatherIcon. */
    public void changeTemp(

}

/* draw: redisplay the WeatherIcon, drawing the kind of weather and the temperature. */
    public void draw() {
        You do NOT need to define this method - assume it is defined.
    }
}

```
