On the back of this page you have a program that opens a frame and watches the mouse. You need to complete the program like you would a crossword puzzle. You can work in pairs (or in larger groups) and you can be as noisy as you want, but this won’t last more than 7 minutes and in up to 10 minutes I will have the papers collected, mainly for attendance but also to see how much you managed to reconstruct the original program. There will be a range of such programs available for you in lab today that you will need to turn in in person, on paper.

Here’s the information that you need: one usually extends a JFrame to create a free floating window (frame) that can hold components in Java. JFrame is a class defined in the package javax.swing. Three things we need to do with a frame before it can exist harmoniously in its environment are: setting the size, the visibility (to either true or false, but definitely to true if you want to see it) and telling it what the default close operation code is. To have the frame go away one uses the code ”exit on close” which is defined as a constant in the JFrame class.

To watch the mouse (or the keyboard or a button or who knows what else) one defines a listener. In this case the motion of the mouse is supposed to be watched so Java defines an interface that every mouse motion listener needs to define. The interface is composed of two methods: one that is supposed to tell us what to do when the mouse moves and the other that is supposed to tell us what to do when the mouse is dragged. They both are being called by the event generator, the one to which the listener is attached. In general you can attach the listener to only objects of the right type: panels, components etc. A JComponent is what we used in this program. The methods in the listener have one argument, of type MouseEvent, that have the ability to obtain (get) the x and y of the mouse when invoked.

A JComponent inherits a paint function. One needs to override it, so you can customize the behavior to whatever you want done. In this case we simply draw a string at a certain position in the component. Components are being added in the frame via the content pane of the frame, which is of type Container. Container is defined in a package called java.awt (abstract windowing toolkit) just like the event listener interfaces have their definitions in a package called java.awt.event. Go ahead and complete the program now.
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class Five extends JFrame {
    Five(int width, int height) {
       ContentPane c = this.getContentPane();
        Six p = new Six();
        c.addMouseMotionListener(new Broker(p));
        c.add(p);
        this.setSize(width, height);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setVisible(true);
    }

    public static void main(String[] args) {
        Five a = new Five(400, 200);
    }
}

class Six extends Component {
    String text = "Howdy.";
    public void paintComponent(Graphics g) {
        g.drawString(text, 50, 500);
    }
}

class Broker implements MouseMotionListener {
    Six client;
    Broker(Six client) {
        this.client = client;
    }

    public void mouseMoved(MouseEvent e) {
        int x = e.getX();
        int y = e.getY();
        client.text = "Mouse being moved at (" + x + ", " + y + ")";
        this.repaint();
    }

    public void mouseDragged(MouseEvent e) {
        int x = e.getX();
        int y = e.getY();
        client.text = "Mouse being dragged at (" + x + ", " + y + ")";
        this.repaint();
    }
}