Week 10, Lecture 1

Hunting the Treasure

Announcements

- **Lab Assessments 6 and 7:**

  deadline this week!

- **Assignment 3:**

  deadline next week

  4pm Friday October 15th

Practical Exam in Week 13

- **Thursday 28th** and **Friday 29th**
- 2 sessions each day starting at 10 am and 1 pm; sessions are of 2.5 hours duration
- see the course web-page for a provisional timetable and information about your session
- you can change your session if you have a timetable clash (see the web-page for details)

Week 10

Lecture 1: Hunting the treasure

Lecture 2: Building a calculator

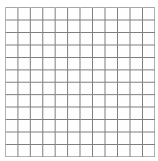
***Java Genesis*:**

–Ch10: Graphical components

Lab Assessment 8 **(deadline week 12)**

Describing the treasure hunt

Treasure is hidden at one of the squares in a 12 by 12 board of squares.



At the start of the hunt all the squares on the board are white. If we click with the mouse on a square it turns an appropriate colour to indicate its distance from the treasure.

Denoting distance by colour

If the distance from the selected square to the treasure is

5 or more the square turns **gray**

4 the square turns **purple**

3 the square turns **blue**

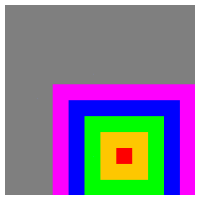
2 the square turns **green**

1 the square turns **orange**

0 the square turns **red** (treasure found)

Measuring distance to treasure

distance to treasure: the smallest number of steps to reach the treasure, where a step is a move to an adjacent horizontal, vertical or diagonal square.



So squares on the board can be referenced, number them from 0 to 143 as shown here:

0	1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31	32	33	34	35
36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79	80	81	82	83
84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107
108	109	110	111	112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127	128	129	130	131
132	133	134	135	136	137	138	139	140	141	142	143

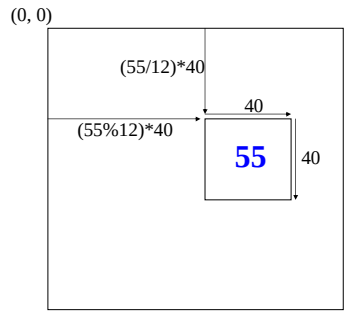
n%12 = 0 55%12 = 7 *n*%12 = 11

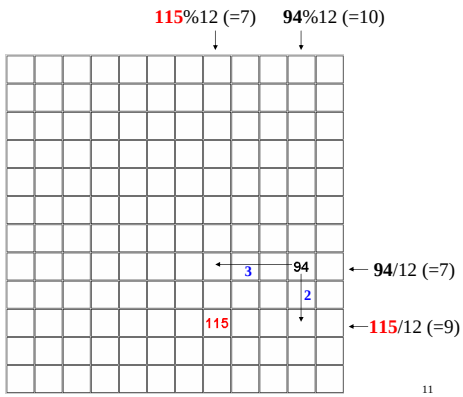
0	1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31	32	33	34	35
36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79	80	81	82	83
84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107
108	109	110	111	112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127	128	129	130	131
132	133	134	135	136	137	138	139	140	141	142	143

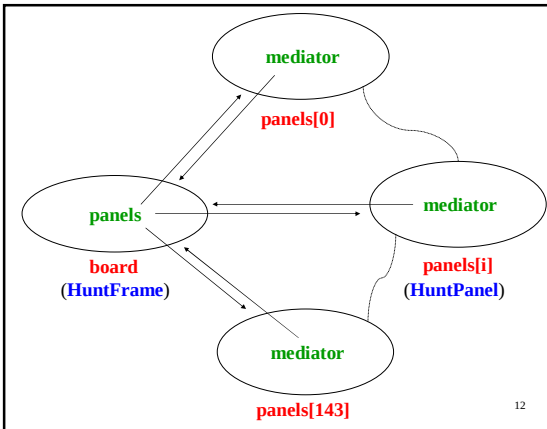
← *n*/12 = 0

← 55/12 = 4

← *n*/12 =11







```
public class Main {

   public static void main (String [ ] args) {
     HuntFrame board = new HuntFrame( );
     board.setVisible(true);
   }
}
```



```
import java.awt.*;
import genesis.*;

public class HuntFrame extends ExitFrame {

   // instance variables
   private HuntPanel [ ] panels =
                              new HuntPanel [144];
   private int treasurePosn =
                        (int)(144*Math.random( ));
   // constructor
   public HuntFrame ( ) {
     setTitle("Treasure Hunt");
     setBounds(300, 100, 12*40+10, 12*40+35);
```



```
    Container c = getContentPane( );
    c.setLayout(null);
    for (int i=0; i<144; i++) {
       panels[i] =
          new HuntPanel(i, treasurePosn, this);
       panels[i].setBounds
                ((i%12)*40, (i/12)*40, 40, 40);
       c.add(panels[i]);
  }
  public void showColourOfRest ( ) {
    for (int i=0; i<144; i++)
      if (i != treasurePosn)
         panels[i].showColour( );
  }
}
```



```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class HuntPanel extends JPanel {

   // instance variables
   private Color [ ] colours =
                   {Color.red, Color.orange,
                    Color.green, Color.blue,
                    Color.magenta, Color.gray};
   private int colourNum;
   private int index;
   private int treasurePosn;
   private HuntFrame mediator;
```



```
  // constructor
  public HuntPanel
           (int in, int posn, HuntFrame med) {
    index = in;
    treasurePosn = posn;
    mediator = med;
    setBackground(Color.white);
    addMouseListener(new MouseAdapter( ) {
       public void mousePressed(MouseEvent e) {
         showColour( );
       }
    });
  }
```



```
  public int showColour ( ) {
    int horizDist =
            Math.abs(index%12-treasurePosn%12);
    int vertDist =
            Math.abs(index/12-treasurePosn/12);
    colourNum = Math.max(horizDist, vertDist);
    if (colourNum > 5) colourNum = 5;
    setBackground(colours[colourNum]);
    if (colourNum == 0)
                  mediator.showColourOfRest( );
  }
}
```