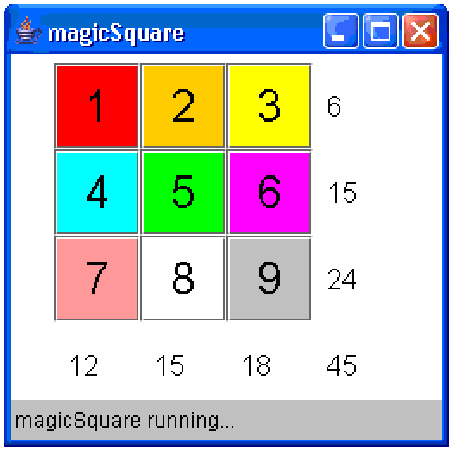
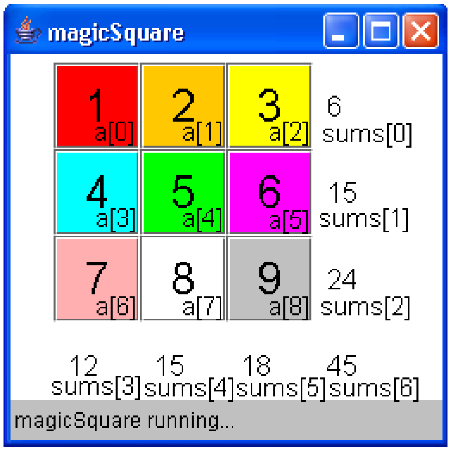
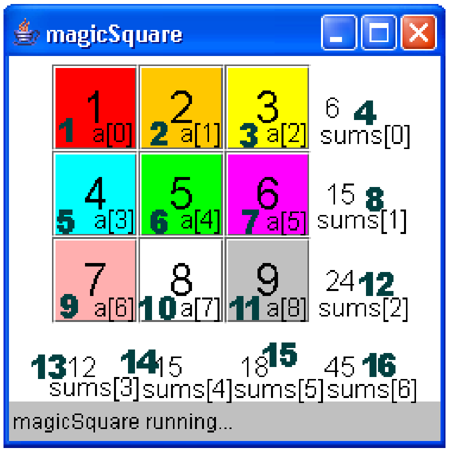
Magic Square

1. Cut and paste this code to Java.

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

public class magicSquare extends Applet implements ActionListener
{
    int row = 3;
    JButton a[] = new JButton [row * row];
    JLabel sums[] = new JLabel [row + row + 1];
    int sumNum[] = {6, 15, 24, 12, 15, 18, 45};

    public void init ()
    {
        for (int i = 0 ; i < a.length ; i++)
        {
            a [i] = new JButton ("" + (i + 1));
            a [i].addActionListener (this);
            a [i].setActionCommand ("" + i);
            a [i].setBackground (pickClr (i + 1));
            a [i].setFont (new Font ("Arial", Font.PLAIN, 26));
            a [i].setPreferredSize (new Dimension (50, 50));
        }
        for (int i = 0 ; i < sums.length ; i++)
        {
            sums [i] = new JLabel ("  " + sumNum [i]);
            sums [i].setFont (new Font ("Arial", Font.PLAIN, 16));
            sums [i].setPreferredSize (new Dimension (50, 50));
        }
        Panel g = new Panel (new GridLayout (row + 1, row + 1));
        //add the buttons here, one by one


        add (g);
        resize (250, 200);
    }

    public void actionPerformed (ActionEvent e)
    {
        if (e.getActionCommand ().equals ("reset"))
        {    //reset code here
        }
        else
        {
            int n = Integer.parseInt (e.getActionCommand ());
            int x = n / row;
            int y = n % row;
            int num = Integer.parseInt (a [n].getText ());
            //process a click code here
        }
    }

    public void updateSums ()
    {
        int n[] = new int [9];
        for (int i = 0 ; i < a.length ; i++)
        {  n [i] = Integer.parseInt (a [i].getText ());
        }
        //add in manual sums here

        for (int i = 0 ; i < sums.length ; i++)
        {  sums [i].setText ("   " + sumNum [i]);
        }
    }
    public Color pickClr (int n)
    {
        Color clr[] = {Color.red, Color.orange, Color.yellow, Color.cyan,
            Color.green, Color.magenta, Color.pink, Color.white, Color.lightGray};
        return (clr [n - 1]);
    }
}
```




magicSquare
1 a[0] 2 a[1] 3 a[2] 6 sums[0]
4 a[3] 5 a[4] 6 a[5] 15 sums[1]
7 a[6] 8 a[7] 9 a[8] 24 sums[2]
12 sums[3] 15 sums[4] 18 sums[5] 45 sums[6]
magicSquare running...


magicSquare
1 1 a[0] 2 2 a[1] 3 3 a[2] 6 4 sums[0]
4 5 a[3] 5 6 a[4] 6 7 a[5] 15 8 sums[1]
7 9 a[6] 8 10 a[7] 9 11 a[8] 24 12 sums[2]
13 12 14 15 18 15 45 16
sums[3] sums[4] sums[5] sums[6]
magicSquare running...

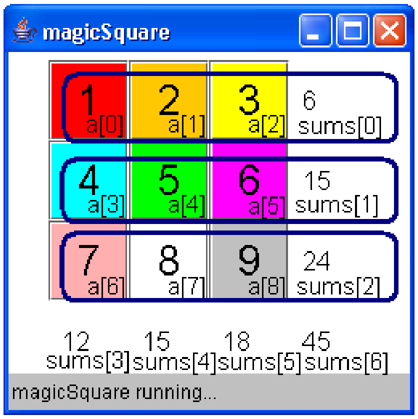
2. Finish adding the buttons to the screen.

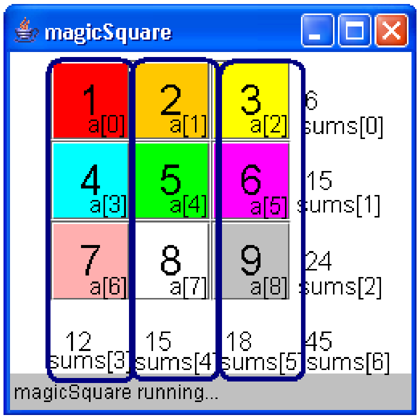
- Because they are in an odd order, they will need to be added one by one.
- The order is in shown in the last diagram above.
- For example the first button will be added like this: `g.add(a[0]);`

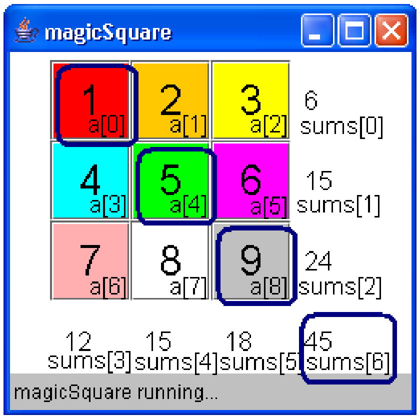
3. In the actionPerformed method’s else: process a click

- As the user clicks on the buttons, they are increased by one and their colour is changed. As well, the sums in the labels update.
- Add the code by following these steps:

```
        Add one to num
        If num is greater than or equal to 10, set num back to one.
        Set a[n]’s text to be num.
        Set a[n]’s background to be whatever the pickClr (num) method returns.
        Take a look in init to see how this works up there.
        Call the updateSums() method.
```


magicSquare
1 a[0] 2 a[1] 3 a[2] 6 sums[0]
4 a[3] 5 a[4] 6 a[5] 15 sums[1]
7 a[6] 8 a[7] 9 a[8] 24 sums[2]
12 sums[3] 15 sums[4] 18 sums[5] 45 sums[6]
magicSquare running...


magicSquare
1 a[0] 2 a[1] 3 a[2] 6 sums[0]
4 a[3] 5 a[4] 6 a[5] 15 sums[1]
7 a[6] 8 a[7] 9 a[8] 24 sums[2]
12 sums[3] 15 sums[4] 18 sums[5] 45 sums[6]
magicSquare running...


magicSquare
1 a[0] 2 a[1] 3 a[2] 6 sums[0]
4 a[3] 5 a[4] 6 a[5] 15 sums[1]
7 a[6] 8 a[7] 9 a[8] 24 sums[2]
12 sums[3] 15 sums[4] 18 sums[5] 45 sums[6]
magicSquare running...

4. In the update sums method, we will need to update the following 7 sums.

- At the comment, add in manual sums. The first is done for you, the others are shown in the diagrams.

```
sumNum [0] = n [0] + n [1] + n [2];
sumNum [1] =
sumNum [2] =
sumNum [3] =
sumNum [4] =
sumNum [5] =
sumNum [6] =
```

5. Run your code, verify everything is working.

6. Paste in the winner method:

```
  public boolean winner ()
    {
        boolean win = true;
        //all sums the same
        //- search to see if all are equal to first spot
        int check = sumNum [0];
        for (int i = 1 ; i < sumNum.length ; i++)
        {
            if (sumNum [i] != check)
                win = false;
        }
        //all numbers different
        //- bin sort approach for each label, if equal, set to 0
        int nums[] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
        int n[] = new int [9];
        for (int i = 0 ; i < a.length ; i++)
        {
            n [i] = Integer.parseInt (a [i].getText ());
        }
        for (int i = 0 ; i < n.length ; i++)
        {
            nums [n [i]]++;
        }
        //- if all 0, then all are different
        for (int i = 1 ; i < nums.length ; i++)
        {
            if (nums [i] == 0)
                win = false;
        }
        return win;
    }
```

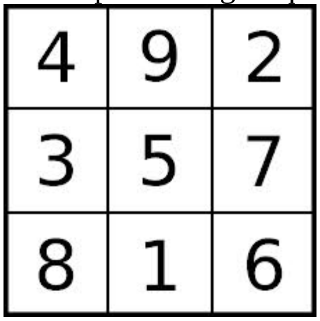
7. Call the winner method in actionPerformed:

- Add to bottom of else condition in AP:

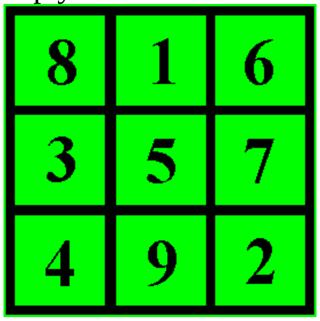
```
showStatus ("Have you won? " + winner ());
```

8. Verify it works.

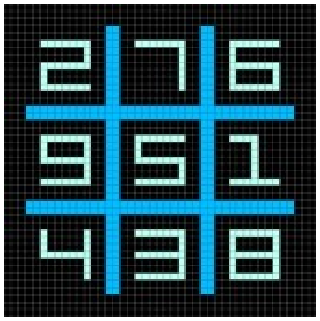
- Some completed magic squares to help you test:



4	9	2
3	5	7
8	1	6

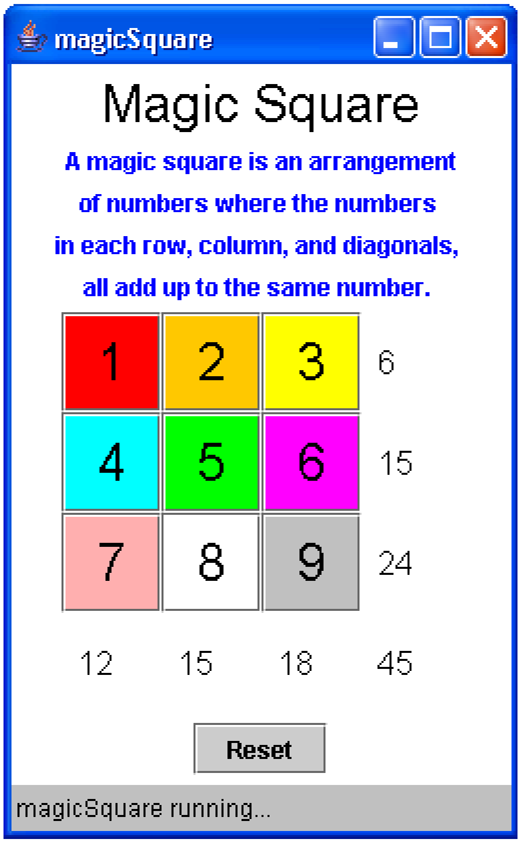


8	1	6
3	5	7
4	9	2



2	7	6
9	5	1
4	3	8

9. Set up the screen nicely:
   - Add in the title
   - Add in the instructions
   - Add in the reset button
   - Resize the screen



Magic Square

A magic square is an arrangement of numbers where the numbers in each row, column, and diagonals, all add up to the same number.

1	2	3	6
4	5	6	15
7	8	9	24
12	15	18	45

magicSquare running...

10. Code the Reset Button:
    - Make a for loop to go through the a array.
      - In it, set the Text of a back to `(i+1).`
      - Also, set the Background to `(pickClr (i + 1)`
    - After the loop, call the `updateSums ()` method
    - As well, add in the code to test if someone has won:

```
showStatus ("Have you won? " + winner ());
```