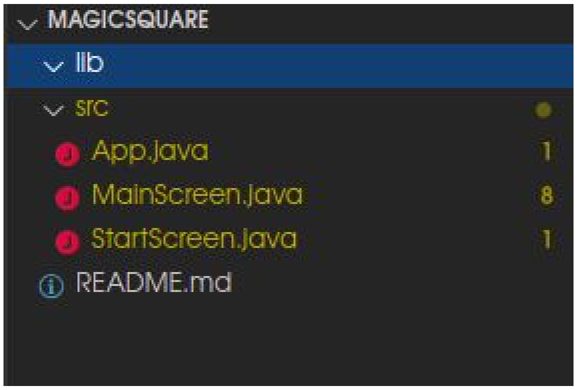
Software Tools 4
Assignment 3

Krunal Rank
U18CO081

Write a Java program to make an App for Magic Squares.

Answer:

Directory Structure:



```
MAGICSQUARE
  lib
  src
    App.java          1
    MainScreen.java   8
    StartScreen.java  1
  README.md
```

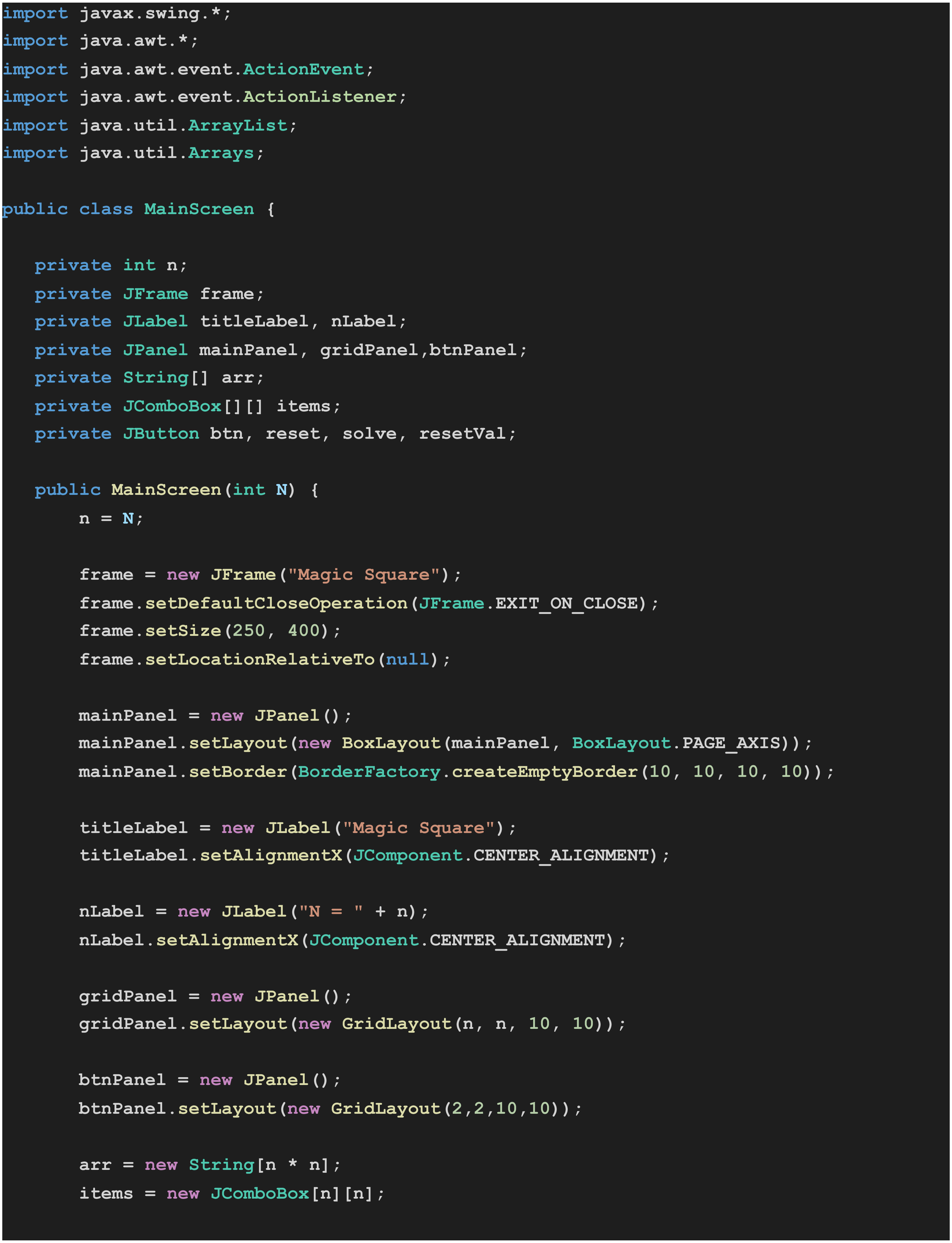
App.java:

```
public class App {
    public static void main(String[] args) throws Exception {
        StartScreen ss = new StartScreen();
    }
}
```

StartScreen.java:

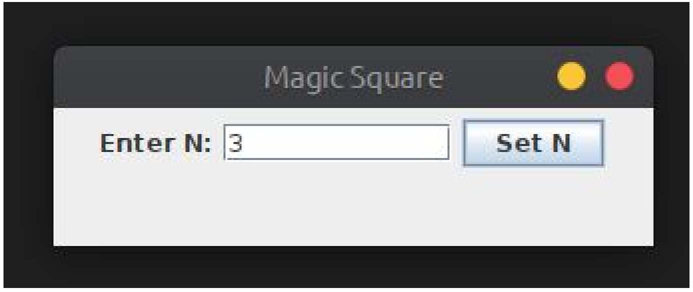
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

class StartScreen {
    private int n = 3;
    private JFrame frame;
    private JPanel panel;
    private JLabel label;
    private JTextField tf;
    private JButton btn;
    public StartScreen() {
        frame = new JFrame("Magic Square");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300, 100);
        frame.setResizable(false);
        frame.setLocationRelativeTo(null);

        panel = new JPanel();

        label = new JLabel("Enter N:");
        tf = new JTextField(n+"",10);
        btn = new JButton("Set N");

        btn.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent evt) {
                try{
                    n = Integer.parseInt(tf.getText());
                    if(n<1 || n>20) throw new Exception("Please enter N between 1 and
20!");
                    MainScreen ms = new MainScreen(n);
                    frame.dispose();
                }catch(Exception e){
                    System.out.println(e);
                    JOptionPane.showMessageDialog(frame,e.getMessage(),"Magic Square -
Error",JOptionPane.ERROR_MESSAGE);
                }
            }


        });

        panel.add(label);
        panel.add(tf);
        panel.add(btn);

        frame.getContentPane().add(BorderLayout.CENTER,panel);
        frame.setVisible(true);

    }

}
```

MainScreen.java:

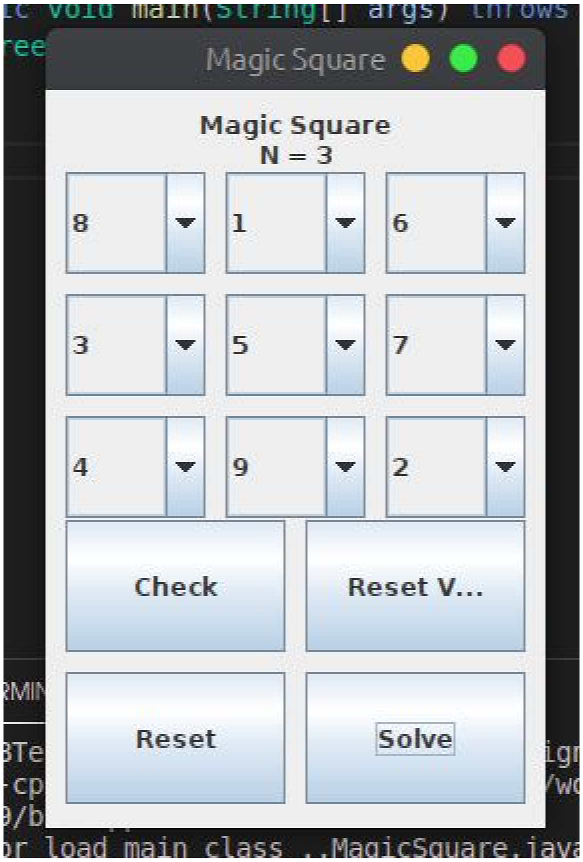


```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.Arrays;

public class MainScreen {

    private int n;
    private JFrame frame;
    private JLabel titleLabel, nLabel;
    private JPanel mainPanel, gridPanel,btnPanel;
    private String[] arr;
    private JComboBox[][] items;
    private JButton btn, reset, solve, resetVal;

    public MainScreen(int N) {
        n = N;

        frame = new JFrame("Magic Square");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(250, 400);
        frame.setLocationRelativeTo(null);

        mainPanel = new JPanel();
        mainPanel.setLayout(new BoxLayout(mainPanel, BoxLayout.PAGE_AXIS));
        mainPanel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));

        titleLabel = new JLabel("Magic Square");
        titleLabel.setAlignmentX(JComponent.CENTER_ALIGNMENT);

        nLabel = new JLabel("N = " + n);
        nLabel.setAlignmentX(JComponent.CENTER_ALIGNMENT);

        gridPanel = new JPanel();
        gridPanel.setLayout(new GridLayout(n, n, 10, 10));

        btnPanel = new JPanel();
        btnPanel.setLayout(new GridLayout(2,2,10,10));

        arr = new String[n * n];
        items = new JComboBox[n][n];
        for (int i = 1; i <= n * n; i++)
            arr[i - 1] = "" + i;

        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                items[i][j] = new JComboBox(arr);
                items[i][j].setSelectedIndex(0);
                gridPanel.add(items[i][j]);
            }
        }

        btn = new JButton("Check");
        btn.setAlignmentX(JComponent.CENTER_ALIGNMENT);

        btn.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent arg0) {
                int[][] arr = new int[n][n];
                for (int i = 0; i < n; i++) {
                    for (int j = 0; j < n; j++) {
                        arr[i][j] = items[i][j].getSelectedIndex() + 1;
                    }
                }
                if (checkCorrectness(arr)) {
                    JOptionPane.showMessageDialog(frame,
                            "Congratulations! You have solved the Magic Square
successfully!",
                            "Magic Square - Congratulations",
JOptionPane.INFORMATION_MESSAGE);
                } else {
                    JOptionPane.showMessageDialog(frame, "Sorry! Please try again to
solve the Magic Square!",
                            "Magic Square - Try Again",
JOptionPane.INFORMATION_MESSAGE);

                }

            }

            boolean checkCorrectness(int[][] arr) {

                int[] marked = new int[n * n];
                for (int i = 0; i < n * n; i++)
                    marked[i] = 0;
                for (int[] p : arr)
                    for (int q : p)
                        marked[q - 1] = 1;
                int allMarked = 0;
                for (int i : marked)
                    allMarked += i;

                if (allMarked != n * n)
                    return false;

                int sum = 0;
                for (int i = 0; i < n; i++)
                    sum += arr[0][i];

                for (int i = 1; i < n; i++) {
                    int sum1 = 0;
                    for (int j = 0; j < n; j++)
                        sum1 += arr[i][j];
                    if (sum1 != sum)
                        return false;
                }
                for (int i = 0; i < n; i++) {
                    int sum1 = 0;
                    for (int j = 0; j < n; j++)
                        sum1 += arr[j][i];
                    if (sum1 != sum)
                        return false;
                }
                int sum1 = 0;
                for (int i = 0; i < n; i++)
                    sum1 += arr[i][i];
                if (sum != sum1)
                    return false;

                sum1 = 0;
                for (int i = 0; i < n; i++)
                    sum1 += arr[n - i - 1][i];
                if (sum != sum1)
                    return false;
                return true;
            }

        });
```

```
        reset = new JButton("Reset");
        reset.setAlignmentX(JComponent.CENTER_ALIGNMENT);

        reset.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent arg0) {
                StartScreen ss = new StartScreen();
                frame.dispose();
            }

        });

        solve = new JButton("Solve");
        solve.setAlignmentX(JComponent.CENTER_ALIGNMENT);

        solve.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent arg0) {
                if (n % 2 == 0) {
                    JOptionPane.showMessageDialog(frame,
                            "Solution is currently available only for odd sided Magic
Squares!",
                            "Magic Squre - Solution", JOptionPane.INFORMATION_MESSAGE);
                    return;
                }
                int[][] arr = new int[n][n];
                for (int i = 0; i < n; i++) {
                    for (int j = 0; j < n; j++)
                        arr[i][j] = 0;
                }

                int i = 0;
                int j = n / 2;
                int count = 1;
                while (count <= n * n) {
                    if (i < 0 && j >= n) {
                        i = 1;
                        j = n - 1;
                    } else if (i < 0) {
                        i = n - 1;
                    } else if (j >= n) {
                        j = 0;
                    } else if (arr[i][j] != 0) {
                        i += 2;
                        j--;
                    } else {
                        arr[i][j] = count++;
                        i--;
                        j++;
                    }
                }
                for (int p = 0; p < n; p++) {
                    for (int q = 0; q < n; q++) {
                        items[p][q].setSelectedIndex(arr[p][q] - 1);
                    }
                }

            }

        });

        resetVal = new JButton("Reset Values");
        resetVal.setAlignmentX(JComponent.CENTER_ALIGNMENT);

        resetVal.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent arg0) {
                for (JComboBox[] i : items)
                    for (JComboBox j : i)
                        j.setSelectedIndex(0);
            }

        });



        btnPanel.add(btn);
        btnPanel.add(resetVal);
        btnPanel.add(reset);
        btnPanel.add(solve);

        mainPanel.add(titleLabel);
        mainPanel.add(nLabel);
        mainPanel.add(gridPanel);
        mainPanel.add(btnPanel);

        frame.getContentPane().add(BorderLayout.CENTER, mainPanel);
        frame.setVisible(true);

    }

}
```



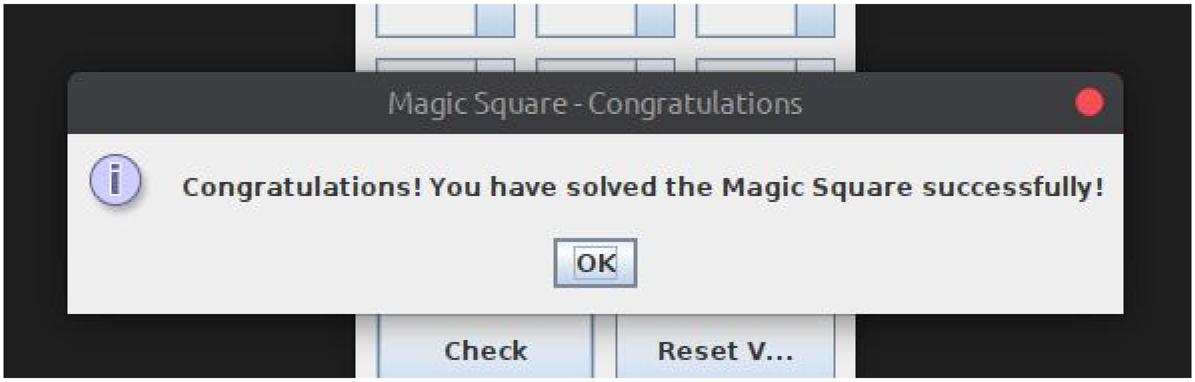
Enter N: 3



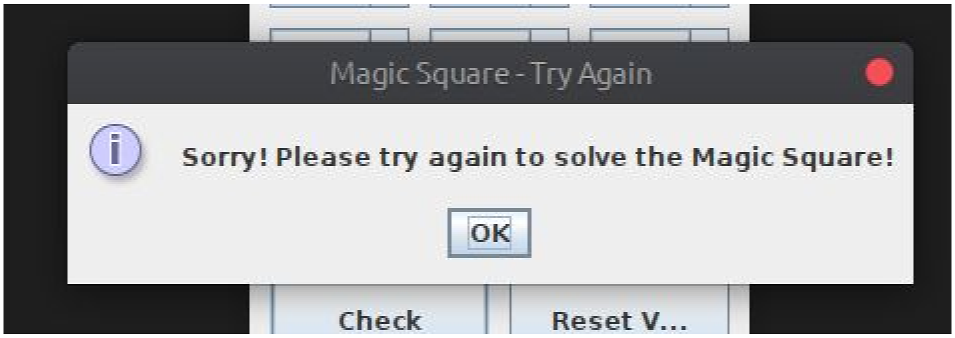
```
ic void main(String[] args) throws
ree
RMIN
3Te                                     igr
-cp                                     /wo
9/b
or load main class ..MagicSquare.java
```

Magic Square
N = 3

```
8 1 6
3 5 7
4 9 2
```



Congratulations! You have solved the Magic Square successfully!



Sorry! Please try again to solve the Magic Square!