```
package guidemos;



import java.awt.EventQueue;

import javax.swing.JFrame;

import java.awt.BorderLayout;
import java.awt.Component;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.Color;
import java.awt.LayoutManager;

import javax.swing.AbstractButton;
import javax.swing.DefaultCellEditor;
import javax.swing.JCheckBox;
import javax.swing.JComboBox;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JLabel;
import javax.swing.ScrollPaneConstants;
import javax.swing.UIManager;

import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

import javax.swing.JRadioButton;
import javax.swing.JButton;
import javax.swing.border.MatteBorder;
import javax.swing.table.DefaultTableCellRenderer;
import javax.swing.table.JTableHeader;
import javax.swing.table.TableCellEditor;
import javax.swing.table.TableCellRenderer;
import javax.swing.table.TableColumn;
import javax.swing.table.TableColumnModel;
```

```
import javax.swing.JEditorPane;
import javax.swing.JInternalFrame;

public class Gui33  {

	private JFrame frmLteResourceGrid;
private JTable jt;
public JComboBox<?> comboBox_1;
JComboBox comboBox_5 ;
JComboBox comboBox_2;
JComboBox comboBox;
JComboBox comboBox_3;
JComboBox comboBox_4;
	/**
	 * Launch the application.
	 */
	public static void main(String[] args) {
		EventQueue.invokeLater(new Runnable() {
			public void run() {
				try {
					Gui33 window = new Gui33();
					window.frmLteResourceGrid.setVisible(true);
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
	}

	/**
	 * Create the application.
	 */

	public Gui33() {
		initialize();
	}

	/**
	 * Initialize the contents of the frame.
	 */
```

```
	private void initialize() {
		System.out.println("comboBox :- bandwidth");
		System.out.println("comboBox_1 :- CyclicPrefix");
		System.out.println("comboBox_5 :- no.of antennas");
		System.out.println("comboBox_2 :- no.of transmission antenna ports ");
		System.out.println("comboBox_3 :- no.of cfi ");
		System.out.println("comboBox_4 :- phich duration ");
		frmLteResourceGrid = new JFrame();
		frmLteResourceGrid.getContentPane().setForeground(Color.GREEN);
		frmLteResourceGrid.getContentPane().setBackground(new Color(245, 255, 250));
		frmLteResourceGrid.getContentPane().setFont(new Font("Times New Roman", Font.BOLD,
22));
		frmLteResourceGrid.setTitle("LTE RESOURCE GRID\r\n");
		frmLteResourceGrid.setBounds(100, 100,973, 500);
		frmLteResourceGrid.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frmLteResourceGrid.getContentPane().setLayout(null);

		JLabel lblNewLabel = new JLabel("");
		lblNewLabel.setFont(new Font("Times New Roman", Font.ITALIC, 16));
		lblNewLabel.setBounds(494, 54, 432, 14);
		frmLteResourceGrid.getContentPane().add(lblNewLabel);

		JLabel lblNewLabel_1 = new JLabel("CHANNEL BANDWIDTH");
		lblNewLabel_1.setFont(new Font("Times New Roman", Font.BOLD, 15));
		lblNewLabel_1.setBounds(134, 55, 185, 14);
		frmLteResourceGrid.getContentPane().add(lblNewLabel_1);
		 comboBox = new JComboBox();
		comboBox.setBackground(Color.WHITE);
		comboBox.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent arg0) {
				if(comboBox.getSelectedItem().equals("1.4 mhz")){
					lblNewLabel.setText("6 Resource Blocks");
				}
				if(comboBox.getSelectedItem().equals("3 mhz")){
					lblNewLabel.setText("15 Resource Blocks");
				}
				for (int i=5;i<=20;i=i+5){
					if(comboBox.getSelectedItem().equals(i+" mhz")){
						lblNewLabel.setText(i*5 +" Resource Blocks");
					}
```

```
                }
            }
        });
        comboBox.setModel(new DefaultComboBoxModel(new String[] {"", "1.4 mhz", "3 mhz", "5 mhz", "10 mhz", "15 mhz", "20 mhz"}));
        comboBox.setBounds(385, 53, 72, 20);
        frmLteResourceGrid.getContentPane().add(comboBox);

        JLabel lblNewLabel_2 = new JLabel("CYCLIC PREFIX");
        lblNewLabel_2.setFont(new Font("Times New Roman", Font.BOLD, 15));
        lblNewLabel_2.setBounds(134, 121, 200, 29);
        frmLteResourceGrid.getContentPane().add(lblNewLabel_2);

        JLabel lblNewLabel_3 = new JLabel("");
        lblNewLabel_3.setFont(new Font("Times New Roman", Font.ITALIC, 16));
        lblNewLabel_3.setBounds(494, 124, 200, 20);
        frmLteResourceGrid.getContentPane().add(lblNewLabel_3);

        comboBox_1 = new JComboBox();
        comboBox_1.setBackground(Color.WHITE);
        comboBox_1.setForeground(Color.BLACK);
        comboBox_1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                if(comboBox_1.getSelectedItem().equals("NORMAL")){
                    lblNewLabel_3.setText("7 Symbols per slot");
                }
                if(comboBox_1.getSelectedItem().equals("EXTENDED")){
                    lblNewLabel_3.setText("6 Symbols per slot");
                }
            }
        });
        comboBox_1.setModel(new DefaultComboBoxModel(new String[] {"", "NORMAL", "EXTENDED"}));
        comboBox_1.setFont(new Font("Times New Roman", Font.PLAIN, 11));
        comboBox_1.setBounds(385, 126, 85, 20);
        frmLteResourceGrid.getContentPane().add(comboBox_1);

        JLabel lblNewLabel_4 = new JLabel("NO.OF TX ANTENNA PORTS");
        lblNewLabel_4.setFont(new Font("Times New Roman", Font.BOLD, 15));
        lblNewLabel_4.setBounds(134, 174, 222, 50);
```

```
frmLteResourceGrid.getContentPane().add(lblNewLabel_4);

 comboBox_2 = new JComboBox();
comboBox_2.setFont(new Font("Times New Roman", Font.ITALIC, 15));
comboBox_2.setModel(new DefaultComboBoxModel(new String[] {"", "1", "2"}));
comboBox_2.setBounds(385, 189, 37, 20);
frmLteResourceGrid.getContentPane().add(comboBox_2);

JLabel lblNewLabel_5 = new JLabel("CONTROL FORMAT INDICATOR");
lblNewLabel_5.setFont(new Font("Times New Roman", Font.BOLD, 14));
lblNewLabel_5.setBounds(134, 232, 222, 50);
frmLteResourceGrid.getContentPane().add(lblNewLabel_5);

 comboBox_3 = new JComboBox();
comboBox_3.setFont(new Font("Tahoma", Font.ITALIC, 14));
comboBox_3.setModel(new DefaultComboBoxModel(new String[] {"", "1", "2", "3"}));
comboBox_3.setBounds(385, 248, 37, 20);
frmLteResourceGrid.getContentPane().add(comboBox_3);

JLabel lblNewLabel_6 = new JLabel("PHICH DURATION");
lblNewLabel_6.setFont(new Font("Times New Roman", Font.BOLD, 15));
lblNewLabel_6.setBounds(134, 300, 200, 29);
frmLteResourceGrid.getContentPane().add(lblNewLabel_6);

 comboBox_4 = new JComboBox();
comboBox_4.setFont(new Font("Times New Roman", Font.PLAIN, 11));
comboBox_4.setModel(new DefaultComboBoxModel(new String[] {"", "NORMAL",
"EXTEND"}));
comboBox_4.setBounds(385, 300, 85, 20);
frmLteResourceGrid.getContentPane().add(comboBox_4);

JButton btnNewButton = new JButton("SUBMIT");
btnNewButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {
                if(comboBox_1.getSelectedItem().equals("NORMAL")){
                if(comboBox.getSelectedItem().equals("1.4 mhz")){

                        jt=new JTable(72,141);
                        jt.setGridColor(Color.black);
                        for(int k=0;k<=71;k++)
```

```
                    jt.setValueAt("subcarrier"+" "+ k , k, 0);
                }
                if(comboBox.getSelectedItem().equals("3 mhz")){
                    jt=new JTable(180,141);
                    jt.setGridColor(Color.black);
                    for(int k=0;k<=179;k++)
                    jt.setValueAt("subcarrier"+" "+ k , k, 0);
                }
                for(int i=5,j=300;i<=20&&j<=1200;i=i+5,j=j+300){
                if(comboBox.getSelectedItem().equals(i+" mhz")){
                    System.out.println(comboBox.getSelectedItem().equals(i+"
mhz"));
                    jt=new JTable(j,141);
                    jt.setGridColor(Color.black);
                    for(int k=0;k<j;k++)
                    jt.setValueAt("subcarrier"+" " +k , k, 0);
                }
                }
                }
                if(comboBox_1.getSelectedItem().equals("EXTENDED")){
                    if(comboBox.getSelectedItem().equals("1.4 mhz")){
                        jt=new JTable(72,121);
                        jt.setGridColor(Color.black);
                        for(int k=0;k<=71;k++)
                        jt.setValueAt("subcarrier"+" "+ k , k, 0);
                    }
                    if(comboBox.getSelectedItem().equals("3 mhz")){
                        jt=new JTable(180,121);
                        jt.setGridColor(Color.black);
                        for(int k=0;k<=179;k++)
                        jt.setValueAt("subcarrier"+" "+ k , k, 0);
                    }
                    for(int i=5,j=300;i<=20&&j<=1200;i=i+5,j=j+300){
                    if(comboBox.getSelectedItem().equals(i+" mhz")){
                        jt=new JTable(j,121);
                        jt.setGridColor(Color.black);
                        for(int k=0;k<j;k++)
                        jt.setValueAt("subcarrier"+" " +k , k, 0);
                    }
                    }
```

```
                                }

tableSetting();
            }
});
btnNewButton.setFont(new Font("Times New Roman", Font.PLAIN, 15));
btnNewButton.setBounds(624, 411, 109, 23);
frmLteResourceGrid.getContentPane().add(btnNewButton);

JLabel lblNewLabel_7 = new JLabel("NO.OF ANTENNAS");
lblNewLabel_7.setFont(new Font("Times New Roman", Font.BOLD, 15));
lblNewLabel_7.setBounds(483, 190, 164, 18);
frmLteResourceGrid.getContentPane().add(lblNewLabel_7);

 comboBox_5 = new JComboBox();
comboBox_5.setModel(new DefaultComboBoxModel(new String[] {"", "1", "2"}));
comboBox_5.setBounds(624, 190, 37, 20);
frmLteResourceGrid.getContentPane().add(comboBox_5);

JEditorPane editorPane = new JEditorPane();
editorPane.setBackground(Color.YELLOW);
editorPane.setEditable(false);
editorPane.setBounds(791, 54, 27, 20);
frmLteResourceGrid.getContentPane().add(editorPane);

JEditorPane dtrpnPss = new JEditorPane();
dtrpnPss.setEditable(false);
dtrpnPss.setFont(new Font("Times New Roman", Font.ITALIC, 16));
dtrpnPss.setText("PSS");
dtrpnPss.setBounds(835, 53, 37, 20);
frmLteResourceGrid.getContentPane().add(dtrpnPss);

JEditorPane editorPane_1 = new JEditorPane();
editorPane_1.setEditable(false);
editorPane_1.setBackground(Color.GREEN);
editorPane_1.setBounds(791, 89, 27, 20);
frmLteResourceGrid.getContentPane().add(editorPane_1);

JEditorPane dtrpnSss = new JEditorPane();
dtrpnSss.setText("SSS");
```

```
dtrpnSss.setEditable(false);
dtrpnSss.setFont(new Font("Times New Roman", Font.ITALIC, 16));
dtrpnSss.setBounds(835, 89, 37, 20);
frmLteResourceGrid.getContentPane().add(dtrpnSss);

JEditorPane editorPane_2 = new JEditorPane();
editorPane_2.setBackground(Color.RED);
editorPane_2.setEditable(false);
editorPane_2.setBounds(791, 121, 27, 20);
frmLteResourceGrid.getContentPane().add(editorPane_2);

JEditorPane dtrpnReferenceSignal = new JEditorPane();
dtrpnReferenceSignal.setEditable(false);
dtrpnReferenceSignal.setText("Reference Signal\r\n");
dtrpnReferenceSignal.setFont(new Font("Times New Roman", Font.ITALIC, 16));
dtrpnReferenceSignal.setBounds(835, 121, 122, 29);
frmLteResourceGrid.getContentPane().add(dtrpnReferenceSignal);

JEditorPane editorPane_3 = new JEditorPane();
editorPane_3.setEditable(false);
editorPane_3.setBackground(Color.BLACK);
editorPane_3.setBounds(791, 152, 27, 20);
frmLteResourceGrid.getContentPane().add(editorPane_3);

JEditorPane dtrpnReservedSignal = new JEditorPane();
dtrpnReservedSignal.setText("Reserved Signal");
dtrpnReservedSignal.setFont(new Font("Times New Roman", Font.ITALIC, 16));
dtrpnReservedSignal.setEditable(false);
dtrpnReservedSignal.setBounds(835, 152, 112, 29);
frmLteResourceGrid.getContentPane().add(dtrpnReservedSignal);

JEditorPane editorPane_4 = new JEditorPane();
editorPane_4.setEditable(false);
editorPane_4.setBackground(Color.PINK);
editorPane_4.setBounds(791, 189, 27, 20);
frmLteResourceGrid.getContentPane().add(editorPane_4);

JEditorPane dtrpnPbch = new JEditorPane();
dtrpnPbch.setText("PBCH");
dtrpnPbch.setFont(new Font("Times New Roman", Font.ITALIC, 16));
```

```
dtrpnPbch.setEditable(false);
dtrpnPbch.setBounds(835, 189, 53, 20);
frmLteResourceGrid.getContentPane().add(dtrpnPbch);

JEditorPane editorPane_5 = new JEditorPane();
editorPane_5.setEditable(false);
editorPane_5.setBackground(Color.CYAN);
editorPane_5.setBounds(791, 220, 27, 20);
frmLteResourceGrid.getContentPane().add(editorPane_5);

JEditorPane dtrpnPdcch = new JEditorPane();
dtrpnPdcch.setText("PDCCH");
dtrpnPdcch.setFont(new Font("Times New Roman", Font.ITALIC, 16));
dtrpnPdcch.setEditable(false);
dtrpnPdcch.setBounds(835, 220, 72, 20);
frmLteResourceGrid.getContentPane().add(dtrpnPdcch);

JEditorPane editorPane_6 = new JEditorPane();
editorPane_6.setBackground(Color.MAGENTA);
editorPane_6.setEditable(false);
editorPane_6.setBounds(791, 248, 27, 20);
frmLteResourceGrid.getContentPane().add(editorPane_6);

JEditorPane dtrpnPcfich = new JEditorPane();
dtrpnPcfich.setText("PCFICH");
dtrpnPcfich.setFont(new Font("Times New Roman", Font.ITALIC, 16));
dtrpnPcfich.setEditable(false);
dtrpnPcfich.setBounds(835, 248, 72, 20);
frmLteResourceGrid.getContentPane().add(dtrpnPcfich);

JEditorPane editorPane_7 = new JEditorPane();
editorPane_7.setEditable(false);
editorPane_7.setBackground(Color.BLUE);
editorPane_7.setBounds(791, 279, 27, 20);
frmLteResourceGrid.getContentPane().add(editorPane_7);

JEditorPane dtrpnPhich = new JEditorPane();
dtrpnPhich.setText("PHICH");
dtrpnPhich.setFont(new Font("Times New Roman", Font.ITALIC, 16));
dtrpnPhich.setEditable(false);
```

```
		dtrpnPhich.setBounds(835, 279, 72, 20);
		frmLteResourceGrid.getContentPane().add(dtrpnPhich);

		JEditorPane editorPane_8 = new JEditorPane();
		editorPane_8.setBackground(new Color(240, 255, 255));
		editorPane_8.setForeground(Color.GREEN);
		editorPane_8.setEditable(false);
		editorPane_8.setBounds(791, 309, 27, 20);
		frmLteResourceGrid.getContentPane().add(editorPane_8);

		JLabel lblNewLabel_8 = new JLabel("PDSCH");
		lblNewLabel_8.setFont(new Font("Times New Roman", Font.ITALIC, 16));
		lblNewLabel_8.setBounds(835, 310, 72, 14);
		frmLteResourceGrid.getContentPane().add(lblNewLabel_8);

	}


	/*
	 * creating table size and scrollbar
	 * */

	public void tableSetting(){
		JFrame jf=new JFrame("RadioFrame                                                                                ");
		jt.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
		for (int i=0,k=0;i<=jt.getColumnCount()-2;i++,k++){
			jt.getColumnModel().getColumn(i+1).setPreferredWidth(20);
			jt.getColumnModel().getColumn(0).setPreferredWidth(90);
			jt.getColumnModel().getColumn(0).setHeaderValue(  "No.of Carriers");
			if(comboBox_1.getSelectedItem().equals("NORMAL")){
			if(k>=7){k=0;}}
			if(comboBox_1.getSelectedItem().equals("EXTENDED")){
			if(k>=6){k=0;}}
	jt.getColumnModel().getColumn(i+1).setHeaderValue("S"+ k);
		}
		System.out.println("no.of columns"+" "+jt.getColumnCount());
		jt.setRowHeight(15);
		System.out.println("no.of rows"+" "+jt.getRowCount());
		JScrollPane jsp= new JScrollPane(jt);
jf.getContentPane().add(jsp);
```

```
        JTableHeader header = jt.getTableHeader();
        header.setBackground(Color.white);
        jsp.setBackground(Color.WHITE);
        pssgen();
        sssgen();
        refgen();
        pbchgen();
        pdcchgen();
        pdccfhigen();
/*int p=0;
for(int j=0;j<jt.getColumnCount();j++){
            for(int i=0;i<jt.getRowCount();i++){
    if( getTableCellBackground(jt, 4, 5).equals(Color.white)
             p++;
     }}
     System.out.println("p"+p);*/
      jf.setSize(600,600);
                        jf.setVisible(true);

                        }


            /*
             * pss and sss signal generation*/


            public void pssgen(){
                        MyRenderer mr = new MyRenderer();
                        int c=(1)*(jt.getColumnCount()-1)/20;
                         int c1=(11)*(jt.getColumnCount()-1)/20;
                         System.out.println(c);
                        jt.getColumnModel().getColumn(c).setCellRenderer(mr);
                        jt.getColumnModel().getColumn(c1).setCellRenderer(mr);
                                    }
            public void sssgen(){
                        MyRenderer1 mr1 = new MyRenderer1();
                        MyRenderer2 mr2 = new MyRenderer2();
                        MyRenderer3 mr3 = new MyRenderer3();
                        MyRenderer4 mr4 = new MyRenderer4();
                        int c=(1)*(jt.getColumnCount()-1)/20;
```

```
            int c1=(11)*(jt.getColumnCount()-1)/20;
            if(comboBox_1.getSelectedItem().equals("NORMAL")){
           jt.getColumnModel().getColumn(c-1).setCellRenderer(mr1);
           jt.getColumnModel().getColumn(c1-1).setCellRenderer(mr1);
            }      if(comboBox_5.getSelectedItem().equals("2")){
           if(comboBox_1.getSelectedItem().equals("EXTENDED")){
                    if(comboBox_2.getSelectedItem().equals("1")){
           jt.getColumnModel().getColumn(c-1).setCellRenderer(mr3);
                                jt.getColumnModel().getColumn(c1-1).setCellRenderer(mr3);}
                    if(comboBox_2.getSelectedItem().equals("2")){
           jt.getColumnModel().getColumn(c-1).setCellRenderer(mr4);
                                jt.getColumnModel().getColumn(c1-1).setCellRenderer(mr4);}
}}
            if(comboBox_5.getSelectedItem().equals("1")){
 if(comboBox_1.getSelectedItem().equals("EXTENDED")){
                                jt.getColumnModel().getColumn(c-1).setCellRenderer(mr2);
                                jt.getColumnModel().getColumn(c1-1).setCellRenderer(mr2);
                                 }}

                    }
/*generation of reference signal and reserved signal
 * */
public void refgen(){
           MyRenderer5 mr5 = new MyRenderer5();
           MyRenderer6 mr6 = new MyRenderer6();
           MyRenderer7 mr7 = new MyRenderer7();
           MyRenderer8 mr8 = new MyRenderer8();
           int c=(1)*(jt.getColumnCount()-1)/20;
            int c1=(11)*(jt.getColumnCount()-1)/20;
            if(comboBox_5.getSelectedItem().equals("2")){
           if(comboBox_2.getSelectedItem().equals("1")){
           for(int i=1;i<jt.getColumnCount();i=i+c)if(i!=c-1&&i!=c1-1){
           jt.getColumnModel().getColumn(i).setCellRenderer(mr5);
           }
           for(int i=5;i<jt.getColumnCount();i=i+c)
            if(i!=c-1&&i!=c1-1){
           jt.getColumnModel().getColumn(i).setCellRenderer(mr6);

                    }}}
if(comboBox_5.getSelectedItem().equals("2")){
```

```
                if(comboBox_2.getSelectedItem().equals("2")){
for(int i=1;i<jt.getColumnCount();i=i+c)if(i!=c-1&&i!=c1-1){
jt.getColumnModel().getColumn(i).setCellRenderer(mr6);
}
for(int i=5;i<jt.getColumnCount();i=i+c)
 if(i!=c-1&&i!=c1-1){
jt.getColumnModel().getColumn(i).setCellRenderer(mr5);


        }}}
                if(comboBox_5.getSelectedItem().equals("1")){
                        for(int i=1;i<jt.getColumnCount();i=i+c)if(i!=c-1&&i!=c1-1){
                        jt.getColumnModel().getColumn(i).setCellRenderer(mr7);
                }}
                        if(comboBox_5.getSelectedItem().equals("1")){
                                for(int i=5;i<jt.getColumnCount();i=i+c)if(i!=c-1&&i!=c1-1){
                                jt.getColumnModel().getColumn(i).setCellRenderer(mr8);
                        }}
        }
        /*generation of physical broadcast channel*/
        public void pbchgen(){
                MyRenderer9 mr9 = new MyRenderer9();
                MyRenderer10 mr10 = new MyRenderer10();
                MyRenderer11 mr11 = new MyRenderer11();
                MyRenderer12 mr12 = new MyRenderer12();
                MyRenderer13 mr13 = new MyRenderer13();
                MyRenderer14 mr14 = new MyRenderer14();
                int c=(1)*(jt.getColumnCount()-1)/20;
                 int c1=(11)*(jt.getColumnCount()-1)/20;
                 System.out.println(c);
                 for(int i=3;i<=4;i++){
                jt.getColumnModel().getColumn(c+i).setCellRenderer(mr9);
                 }
        jt.getColumnModel().getColumn(c+1).setCellRenderer(mr10);
        jt.getColumnModel().getColumn(c+2).setCellRenderer(mr11);
        if(comboBox_5.getSelectedItem().equals("2")){
                if(comboBox_2.getSelectedItem().equals("1")){
                        jt.getColumnModel().getColumn(c+1).setCellRenderer(mr12);
                        jt.getColumnModel().getColumn(c+2).setCellRenderer(mr13);
                }
                if(comboBox_2.getSelectedItem().equals("2")){
```

```
                jt.getColumnModel().getColumn(c+1).setCellRenderer(mr14);
                jt.getColumnModel().getColumn(c+2).setCellRenderer(mr13);
        }
}
}
/*generation of physical downlink control channel*/
public void pdcchgen(){
        MyRenderer15 mr15 = new MyRenderer15();
        if(comboBox.getSelectedItem().equals("1.4 mhz")){
                for(int k=1;k<=3;k++){
                        if(comboBox_3.getSelectedItem().equals(k+"")){
                                for(int i=2;i<=k+1;i++)
                                        for(int j=2;j<jt.getColumnCount();j=j+14)
                                        for(int l=0;l<=k-1;l++)

jt.getColumnModel().getColumn(j+l).setCellRenderer(mr15);
                        }
                }
        }
        if(comboBox.getSelectedItem().equals("3 mhz")){
                for(int k=1;k<=3;k++){
                        if(comboBox_3.getSelectedItem().equals(k+"")){
                                for(int i=2;i<=k;i++)
                                for(int j=2;j<jt.getColumnCount();j=j+14)
                                for(int l=0;l<k-1;l++)
                                jt.getColumnModel().getColumn(j+l).setCellRenderer(mr15);
                        }
                }
        }
        for(int i=5,j=300;i<=20&&j<=1200;i=i+5,j=j+300){
        if(comboBox.getSelectedItem().equals(i+" mhz")){
                for(int k=1;k<=3;k++){
                        if(comboBox_3.getSelectedItem().equals(k+"")){
                                for(int m=2;m<=k;m++)
                                        for(int n=2;n<jt.getColumnCount();n=n+14)
                                        for(int l=0;l<k-1;l++)

jt.getColumnModel().getColumn(n+l).setCellRenderer(mr15);
                        }
                }
```

```
		}
		}
		}
/*generation of pdcch ,physical control format indicator channel,
 * physical hybrid Arq indicator channel*/
public void pdccfhigen(){
		MyRenderer16 mr16 = new MyRenderer16();
		MyRenderer17 mr17 = new MyRenderer17();
		MyRenderer18 mr18 = new MyRenderer18();
		MyRenderer19 mr19 = new MyRenderer19();
		MyRenderer20 mr20 = new MyRenderer20();
		MyRenderer21 mr21 = new MyRenderer21();
		MyRenderer22 mr22 = new MyRenderer22();
		MyRenderer23 mr23 = new MyRenderer23();
		MyRenderer24 mr24 = new MyRenderer24();
		MyRenderer25 mr25 = new MyRenderer25();
		MyRenderer26 mr26 = new MyRenderer26();
		MyRenderer27 mr27 = new MyRenderer27();
		if(comboBox_4.getSelectedItem().equals("NORMAL")){
		if(comboBox_1.getSelectedItem().equals("NORMAL")){
		if(comboBox_5.getSelectedItem().equals("1")){
		for(int i=1;i<jt.getColumnCount();i=i+14)
		jt.getColumnModel().getColumn(i).setCellRenderer(mr16);
		}
		if(comboBox_5.getSelectedItem().equals("2")){
			if(comboBox_2.getSelectedItem().equals("1")){
			for(int i=1;i<jt.getColumnCount();i=i+14)
			jt.getColumnModel().getColumn(i).setCellRenderer(mr17);
			}
			if(comboBox_2.getSelectedItem().equals("2")){
				for(int i=1;i<jt.getColumnCount();i=i+14)
				jt.getColumnModel().getColumn(i).setCellRenderer(mr18);
				}

		}}
		if(comboBox_1.getSelectedItem().equals("EXTENDED")){
			if(comboBox_5.getSelectedItem().equals("1")){
			for(int i=1;i<jt.getColumnCount();i=i+14)
			jt.getColumnModel().getColumn(i).setCellRenderer(mr19);
			}
```

```
                if(comboBox_5.getSelectedItem().equals("2")){
                        if(comboBox_2.getSelectedItem().equals("1")){
                        for(int i=1;i<jt.getColumnCount();i=i+14)
                        jt.getColumnModel().getColumn(i).setCellRenderer(mr20);
                        }
                        if(comboBox_2.getSelectedItem().equals("2")){
                                for(int i=1;i<jt.getColumnCount();i=i+14)
                                jt.getColumnModel().getColumn(i).setCellRenderer(mr21);
                                }}
        }}

if(comboBox_4.getSelectedItem().equals("EXTEND")){
        if(comboBox_1.getSelectedItem().equals("NORMAL")){
        if(comboBox_5.getSelectedItem().equals("1")){
        for(int i=1;i<jt.getColumnCount();i=i+14)
                jt.getColumnModel().getColumn(i).setCellRenderer(mr22);
                for(int j=2;j<jt.getColumnCount();j=j+14)
                        jt.getColumnModel().getColumn(j).setCellRenderer(mr22);
                for(int j=3;j<jt.getColumnCount();j=j+14)
                        jt.getColumnModel().getColumn(j).setCellRenderer(mr22);
        }
        if(comboBox_5.getSelectedItem().equals("2")){
                if(comboBox_2.getSelectedItem().equals("1")){
                for(int i=1;i<jt.getColumnCount();i=i+14)
                jt.getColumnModel().getColumn(i).setCellRenderer(mr23);
                for(int j=2;j<jt.getColumnCount();j=j+14)
                        jt.getColumnModel().getColumn(j).setCellRenderer(mr23);
                for(int j=3;j<jt.getColumnCount();j=j+14)
                        jt.getColumnModel().getColumn(j).setCellRenderer(mr23);
                }
                if(comboBox_2.getSelectedItem().equals("2")){
                        for(int i=1;i<jt.getColumnCount();i=i+14)
                        jt.getColumnModel().getColumn(i).setCellRenderer(mr24);
                        for(int j=2;j<jt.getColumnCount();j=j+14)
                                jt.getColumnModel().getColumn(j).setCellRenderer(mr23);
                        for(int j=3;j<jt.getColumnCount();j=j+14)
                                jt.getColumnModel().getColumn(j).setCellRenderer(mr23);
                        }}}
        if(comboBox_1.getSelectedItem().equals("EXTENDED")){
                if(comboBox_5.getSelectedItem().equals("1")){
```

```
                        for(int i=1;i<jt.getColumnCount();i=i+14)
                                jt.getColumnModel().getColumn(i).setCellRenderer(mr25);
                                for(int j=2;j<jt.getColumnCount();j=j+14)
                                        jt.getColumnModel().getColumn(j).setCellRenderer(mr25);
                                for(int j=3;j<jt.getColumnCount();j=j+14)
                                        jt.getColumnModel().getColumn(j).setCellRenderer(mr25);
                        }
                        if(comboBox_5.getSelectedItem().equals("2")){
                                if(comboBox_2.getSelectedItem().equals("1")){
                                for(int i=1;i<jt.getColumnCount();i=i+14)
                                jt.getColumnModel().getColumn(i).setCellRenderer(mr26);
                                for(int j=2;j<jt.getColumnCount();j=j+14)
                                        jt.getColumnModel().getColumn(j).setCellRenderer(mr26);
                                for(int j=3;j<jt.getColumnCount();j=j+14)
                                        jt.getColumnModel().getColumn(j).setCellRenderer(mr26);
                                }
                                if(comboBox_2.getSelectedItem().equals("2")){
                                        for(int i=1;i<jt.getColumnCount();i=i+14)
                                        jt.getColumnModel().getColumn(i).setCellRenderer(mr27);
                                        for(int j=2;j<jt.getColumnCount();j=j+14)

        jt.getColumnModel().getColumn(j).setCellRenderer(mr26);
                                        for(int j=3;j<jt.getColumnCount();j=j+14)

        jt.getColumnModel().getColumn(j).setCellRenderer(mr26);
                                        }}}

}}
        public Color getTableCellBackground(JTable table, int row, int col) {
           DefaultTableCellRenderer renderer = (DefaultTableCellRenderer) table.getCellRenderer(row, col);
           Component component = table.prepareRenderer(renderer, row, col);
          return component.getBackground();

        }
}

/* coloring for pss */
class MyRenderer extends DefaultTableCellRenderer
{
```

```
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
            int c;
            c=table.getRowCount()/2;
   if(row>c-32&&row<=c+30){
      setBackground(Color.yellow);
return this;
            }
   if(row>c-37&&row<=c-32){
            setBackground(Color.black);
            return this;
   }
   if(row>=c+30&&row<c+36){
            setBackground(Color.black);
            return this;
   }
            return null;
   }
}
class MyRenderer1 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
            int c;
            c=table.getRowCount()/2;
   if(row>c-32&&row<=c+30){
      setBackground(Color.GREEN);
      return this;
   }
   if(row>c-37&&row<=c-32){
            setBackground(Color.black);
            return this;
   }
   if(row>=c+30&&row<c+36){
            setBackground(Color.black);
            return this;
   }
            return null;
```

```
    }
}
class MyRenderer2 extends DefaultTableCellRenderer
{
    public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
    {
            int c;
            c=table.getRowCount()/2;
            for(int i=3;i<table.getRowCount();i=i+6){
    if(row==i){
       setBackground(Color.red);
       return this;
    }
    else {
            if(row>c-37&&row<=c-32&&row!=c-33)
            {
            setBackground(Color.black);
            return this;
       }if(row>c-32&&row<=c+30){
            for(int j=1;j<=5;j++){
                    if(row!=c-32&&row==i+j&&row<=c+30){
                            setBackground(Color.green);
            return this;}}

    }
    }
            if(row>c+30&&row<c+36&&row!=c+33){
            setBackground(Color.black);
            return this;
       }
            }
return null;
            }
}
class MyRenderer3 extends DefaultTableCellRenderer
{
    public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
    {
```

```
            int c;
            c=table.getRowCount()/2;
            for(int k=0;k<table.getRowCount();k=k+6){
          if(row==k){
             setBackground(Color.black);
             return this;
          }}
            for(int i=3;i<table.getRowCount();i=i+6){
   if(row==i){
      setBackground(Color.red);
      return this;
   }
   else{
            if(row>c-37&&row<=c-32&&row!=c-33)
            {
            setBackground(Color.black);
            return this;
          }if(row>c-32&&row<=c+30){
            for(int j=1;j<=5;j++){
                        if(row!=c-32&&row==i+j&&row<=c+30){
                                    setBackground(Color.green);
            return this;}}
      }

      }
            if(row>c+30&&row<c+36&&row!=c+33){
            setBackground(Color.black);
            return this;
          }
            }
            return null;}}

class MyRenderer4 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
            int c;
            c=table.getRowCount()/2;
            for(int k=0;k<table.getRowCount();k=k+6){
```

```
        if(row==k){
           setBackground(Color.red);
           return this;
        }}
          for(int i=3;i<table.getRowCount();i=i+6){
   if(row==i){
      setBackground(Color.black);
      return this;
   }
   else{
          if(row>c-37&&row<=c-32&&row!=c-33)
          {
          setBackground(Color.black);
          return this;
        }if(row>c-32&&row<=c+30){
          for(int j=1;j<=5;j++){
                    if(row!=c-32&&row==i+j&&row<=c+30){
                              setBackground(Color.green);
          return this;}}
      }

      }
          if(row>c+30&&row<c+36&&row!=c+33){
          setBackground(Color.black);
          return this;
        }
          }
          return null;}}
class MyRenderer5 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
          int c;
          c=table.getRowCount()/2;
          for(int k=0;k<table.getRowCount();k=k+6){
        if(row==k){
           setBackground(Color.red);
           return this;
        }}for(int k=3;k<table.getRowCount();k=k+6){
```

```
            if(row==k){
               setBackground(Color.black);
               return this;
            }}return null;}}
class MyRenderer6 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
            int c;
            c=table.getRowCount()/2;
            for(int k=3;k<table.getRowCount();k=k+6){
          if(row==k){
             setBackground(Color.red);
             return this;
          }}for(int k=0;k<table.getRowCount();k=k+6){
             if(row==k){
                setBackground(Color.black);
                return this;
             }}return null;}}
class MyRenderer7 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
             for(int k=0;k<table.getRowCount();k=k+6){
          if(row==k){
             setBackground(Color.red);
             return this;
          }}return null;}}
class MyRenderer8 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
             for(int k=3;k<table.getRowCount();k=k+6){
          if(row==k){
             setBackground(Color.red);
             return this;
          }}return null;}}
```

```
class MyRenderer9 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
            int c;
            c=table.getRowCount()/2;
   if(row>=c-36&&row<c+36){
      setBackground(Color.pink);
      return this;
            }return null;}}
class MyRenderer10 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
            int c;
            c=table.getRowCount()/2;
            for(int i=0;i<table.getRowCount();i=i+6){
   if(row==i){
      setBackground(Color.red);
      return this;
   }
   else {
            if(row>=c-36&&row<c+36){
            for(int j=1;j<=5;j++){
                     if(row==i+j){
                              setBackground(Color.pink);
            return this;}}}}



   }
return null;
            }
            }
class MyRenderer11 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
```

```
            int c;
            c=table.getRowCount()/2;
            for(int i=0;i<table.getRowCount();i=i+6){

    if(row==i){if(row>=c-36&&row<c+36){
        setBackground(Color.black);
        return this;
    }}
    else {
            if(row>=c-36&&row<c+36){
            for(int j=1;j<=5;j++){
                        if(row==i+j){
                                    setBackground(Color.pink);
            return this;}}}}



    }
return null;
            }
            }
class MyRenderer12 extends DefaultTableCellRenderer
{
    public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
    {
            int c;
            c=table.getRowCount()/2;
            for(int k=3;k<table.getRowCount();k=k+6){
           if(row==k){
              setBackground(Color.black);
              return this;
           }}
            for(int i=0;i<table.getRowCount();i=i+6){
    if(row==i){
        setBackground(Color.red);
        return this;
    }
    else{
            if(row>=c-36&&row<c+36){
            for(int j=1;j<=5;j++){
```

```
                    if(row==i+j){
                            setBackground(Color.pink);
            return this;}}
            }
        }
            }
            return null;}}
class MyRenderer13 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {
            int c;
            c=table.getRowCount()/2;
            for(int k=3;k<table.getRowCount();k=k+6){
          if(row==k){
            if(row>=c-36&&row<c+36){
             setBackground(Color.black);
             return this;
          }}}
            for(int i=0;i<table.getRowCount();i=i+6){
   if(row==i){
            if(row>=c-36&&row<c+36){
      setBackground(Color.black);
      return this;
   }}
   else{
            if(row>=c-36&&row<c+36){
            for(int j=1;j<=5;j++){
                    if(row==i+j){
                            setBackground(Color.pink);
            return this;}}
            }
        }
            }
            return null;}}
class MyRenderer14 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
```

```
    {
            int c;
            c=table.getRowCount()/2;
            for(int k=3;k<table.getRowCount();k=k+6){
          if(row==k){
             setBackground(Color.red);
             return this;
          }}
            for(int i=0;i<table.getRowCount();i=i+6){
    if(row==i){
       setBackground(Color.black);
       return this;
    }
    else{
            if(row>=c-36&&row<c+36){
            for(int j=1;j<=5;j++){
                        if(row==i+j){
                                    setBackground(Color.pink);
            return this;}}
            }
       }
            }
            return null;}}
class MyRenderer15 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
   {
  setBackground(Color.cyan);
  return this;
   }
}
class MyRenderer16 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
   {       int n=table.getRowCount()/12;
   int c=table.getRowCount()/(12*8);
   if(table.getRowCount()%(12*8)!=0)
            c=c+1;
```

```
   for(int k=0;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.red);
         return this;}}
   for(int s=0;s<=3;s++){
          int j=(s*n)/2;
for(int i=6*j+1;i<=6*j+4;i++){
           if(row==i){
           setBackground(Color.magenta);
           return this;
           }
           }}

   for(int m=0;m<c;m++)
   for(int j=0;j<=2;j++ ){
          int i1=(m+j*n)*4;
          int p=0;
           int k1=c-1;
          int k=c;
                  if(j==0){
                         int i=m;
                         if(table.getRowCount()==600)
                                k=c-1;
                         if(table.getRowCount()>=900)
                                k=c-2;
                          for(;i<5+4*c+k;i++){
                                if(row==i){
                                       setBackground(Color.blue);
                                return this;}
                                }
                  }
                  else{
                         if(i1%6==0){
                         i1=i1+1;}
                  else k1--;
                  if(table.getRowCount()==180)
                         k1=k1+1;
                         if(table.getRowCount()>=900)
                                k1=k1-2;
   for(;i1<=(m+j*n)*4+4+k1;i1++){
```

```
            if(row==i1){
                    setBackground(Color.blue);
            return this;}
            }
            }
    }
 for(int k=0;k<table.getRowCount();k=k+6){
                            if(row!=k){
                               setBackground(Color.cyan);
                               return this;}
                            }
    return null;
    }
}
class MyRenderer17 extends DefaultTableCellRenderer
{
    public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
    {       int n=table.getRowCount()/12;
    int c=table.getRowCount()/(12*8);
    if(table.getRowCount()%(12*8)!=0)
            c=c+1;
    for(int k=0;k<table.getRowCount();k=k+6){
       if(row==k){
          setBackground(Color.red);
          return this;}}
    for(int k=3;k<table.getRowCount();k=k+6){
       if(row==k){
          setBackground(Color.black);
          return this;}}
    for(int s=0;s<=3;s++){
             int j=(s*n)/2;
              for(int i1=6*j+1;i1<=6*j+5;i1++){
                                  if(row==i1){
                                  setBackground(Color.magenta);
                                 return this;
                                  }
                                  } }

    for(int m=0;m<c;m++)
```

```
   for(int j=0;j<=2;j++ ){
            int i1=(m+j*n)*4;
             int k1=2*c-1;
            int k=2*c;
            if(j==2){
                     if(table.getRowCount()==300)
                 k1=2*c+5;
            }

                     if(j==0){
                              int i=m;
                               for(;i<6+4*c+k;i++){
                                       if(row==i){
                                                setBackground(Color.blue);
                                       return this;}
                                       }
                     }
                     else{
                              if(i1%6==0){
                              i1=i1+1;}
   for(;i1<=(m+j*n)*4+4+k1;i1++){
            if(row==i1){
                     setBackground(Color.blue);
            return this;}
            }
            }
   }
 for(int k=0,j=3;j<table.getRowCount()&&k<table.getRowCount();k=k+6,j=j+6){
                          if(row!=k&&row!=j){
                             setBackground(Color.cyan);
                             return this;}
                          }
   return null;
   }
}
class MyRenderer18 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
   {        int n=table.getRowCount()/12;
```

```
   int c=table.getRowCount()/(12*8);
   if(table.getRowCount()%(12*8)!=0)
            c=c+1;
   for(int k=0;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.black);
         return this;}}
   for(int k=3;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.red);
         return this;}}
   for(int s=0;s<=3;s++){
            int j=(s*n)/2;
for(int i=6*j+1;i<=6*j+5;i++){
            if(row==i&&row!=i+3){
            setBackground(Color.magenta);
            return this;
            }}}
   for(int m=0;m<c;m++)
      for(int j=0;j<=2;j++ ){
         int i1=(m+j*n)*4;
          int k1=2*c-1;
         int k=2*c;
         if(j==2){
                  if(table.getRowCount()==300)
               k1=2*c+5;
         }

                  if(j==0){
                           int i=m;
                            for(;i<6+4*c+k;i++){
                                    if(row==i){
                                             setBackground(Color.blue);
                                    return this;}
                                    }
                  }
                  else{
                           if(i1%6==0){
                           i1=i1+1;}
      for(;i1<=(m+j*n)*4+4+k1;i1++){
```

```
            if(row==i1){
                    setBackground(Color.blue);
            return this;}
            }
            }
        }
 for(int k=0,j=3;j<table.getRowCount()&&k<table.getRowCount();k=k+6,j=j+6){
                            if(row!=k&&row!=j){
                               setBackground(Color.cyan);
                               return this;}
                            }
   return null;
   }
}
class MyRenderer19 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
   {       int n=table.getRowCount()/12;
   int c1=table.getRowCount()/(12*8);
   if(table.getRowCount()%(12*8)!=0)
           c1=c1+1;
   int c=2*c1;
   for(int k=0;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.red);
         return this;}}
   for(int s=0;s<=3;s++){
           int j=(s*n)/2;
 for(int i=6*j+1;i<=6*j+4;i++){
            if(row==i){
            setBackground(Color.magenta);
            return this;
            }
            }}

   for(int m=0;m<c;m++)
   for(int j=0;j<=2;j++ ){
           int i1=(m+j*n)*4;
           int p=0;
```

```
 int k1=c-1;
int k=c;
int b=table.getRowCount()/60;
        if(j==0){
                if(table.getRowCount()>300)
                        k=b+1;
                if(table.getRowCount()==300)
                        k=c-1;

                 for(int i=m;i<5+4*c+k;i++){
                        if(row==i){
                                setBackground(Color.blue);
                        return this;}
                        }
        }
        else{
                if(i1%6==0){
                i1=i1+1;}
        else k1--;
                if(table.getRowCount()==72)
                        k1=k1+1;
                if(table.getRowCount()>=600)
                        k1=b;
                if(j==2)
                {
                        if(table.getRowCount()>=600)
                        k1=k1+5;
                   if(table.getRowCount()==300)
                           k1=k1-2;
                }

  for(;i1<=(m+j*n)*4+4+k1;i1++){
        if(row==i1){
                setBackground(Color.blue);
        return this;}
        }
        }
  }
for(int k=0;k<table.getRowCount();k=k+6){
                        if(row!=k){
```

```
                                setBackground(Color.cyan);
                                return this;}
                            }
    return null;
    }
}
class MyRenderer20 extends DefaultTableCellRenderer
{
    public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
    {       int n=table.getRowCount()/12;
    int c1=table.getRowCount()/(12*8);
    if(table.getRowCount()%(12*8)!=0)
            c1=c1+1;
    int c=2*c1;
    for(int k=0;k<table.getRowCount();k=k+6){
       if(row==k){
          setBackground(Color.red);
          return this;}}
    for(int k=3;k<table.getRowCount();k=k+6){
       if(row==k){
          setBackground(Color.black);
          return this;}}
    for(int s=0;s<=3;s++){
            int j=(s*n)/2;
             for(int i1=6*j+1;i1<=6*j+5;i1++){
                                  if(row==i1){
                                  setBackground(Color.magenta);
                                 return this;
                                  }
                                  } }

    for(int m=0;m<c;m++)
    for(int j=0;j<=2;j++ ){
            int i1=(m+j*n)*4;
             int k1=2*c-1;
            int k=2*c;
            if(j==2){
                     if(table.getRowCount()>=72)
             k1=2*c+5;
```

```
        }

                if(j==0){
                        int i=m;
                         for(;i<6+4*c+k;i++){
                                if(row==i){
                                        setBackground(Color.blue);
                                return this;}
                                }
                }
                else{
                        if(i1%6==0){
                        i1=i1+1;}
   for(;i1<=(m+j*n)*4+4+k1;i1++){
          if(row==i1){
                setBackground(Color.blue);
          return this;}
          }
          }
   }
 for(int k=0,j=3;j<table.getRowCount()&&k<table.getRowCount();k=k+6,j=j+6){
                        if(row!=k&&row!=j){
                           setBackground(Color.cyan);
                           return this;}
                        }
   return null;
   }
}
class MyRenderer21 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
   {       int n=table.getRowCount()/12;
   int c1=table.getRowCount()/(12*8);
   if(table.getRowCount()%(12*8)!=0)
          c1=c1+1;
   int c=2*c1;
   for(int k=0;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.black);
```

```
        return this;}}
   for(int k=3;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.red);
         return this;}}
   for(int s=0;s<=3;s++){
          int j=(s*n)/2;
for(int i=6*j+1;i<=6*j+5;i++){
          if(row==i&&row!=i+3){
          setBackground(Color.magenta);
          return this;
          }}}
   for(int m=0;m<c;m++)
      for(int j=0;j<=2;j++ ){
         int i1=(m+j*n)*4;
          int k1=2*c-1;
         int k=2*c;
         if(j==2){
                  if(table.getRowCount()>=72)
             k1=2*c+5;

         }

                  if(j==0){
                           int i=m;
                            for(;i<6+4*c+k;i++){
                                    if(row==i){
                                             setBackground(Color.blue);
                                    return this;}
                                    }
                  }
                  else{
                           if(i1%6==0){
                           i1=i1+1;}
      for(;i1<=(m+j*n)*4+4+k1;i1++){
         if(row==i1){
                  setBackground(Color.blue);
         return this;}
         }
         }
```

```
        }
 for(int k=0,j=3;j<table.getRowCount()&&k<table.getRowCount();k=k+6,j=j+6){
                                if(row!=k&&row!=j){
                                   setBackground(Color.cyan);
                                   return this;}
                                }
   return null;
   }
}
class MyRenderer22 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
   {       int n=table.getRowCount()/12;
   int c=table.getRowCount()/(12*8);
   int d=1;
   if(table.getRowCount()%(12*8)!=0)
            c=c+1;
   for(d=1;d<table.getColumnCount();d=d+14){
            if(column==d){
   for(int k=0;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.red);
         return this;}}
   for(int s=0;s<=3;s++){
            int j=(s*n)/2;
 for(int i=6*j+1;i<=6*j+4;i++){
             if(row==i){
             setBackground(Color.magenta);
             return this;
             }
             }}
   for(int m=0;m<c;m++)
   for(int j=0;j<=2;j++ ){
            int i1=(m+j*n)*4;
            int p=0;
             int k1=c-1;
            int k=c;
                     if(j==0){
                              int i=m;
```

```
                    if(table.getRowCount()==600)
                        k=c-1;
                    if(table.getRowCount()>=900)
                        k=c-2;
                     for(;i<5+4*c+k;i++){
                        if(row==i){
                            setBackground(Color.blue);
                        return this;}
                        }
                }}}
        for(int e=2,f=3;e<table.getColumnCount()&&f<table.getColumnCount();f=f+14,e=e+14){
   for(int m=0;m<c;m++)
      for(int j=1;j<=2;j++ ){
        if(column==e&&j==1||column==f&&j==2){
        int i1=(m+j*n)*4;
        int p=0;
         int k1=c-1;
        int k=c;
                    if(i1%6==0){
                    i1=i1+1;}
                else k1--;
                if(table.getRowCount()==180)
                    k1=k1+1;
                    if(table.getRowCount()>=900)
                        k1=k1-2;
   for(;i1<=(m+j*n)*4+4+k1;i1++){
        if(row==i1){
            setBackground(Color.blue);
        return this;}}}}
        }
        }
 for(int k=0;k<table.getRowCount();k=k+6){
                    if(row!=k){
                       setBackground(Color.cyan);
                       return this;}
                    }
   return null;
   }
}
class MyRenderer23 extends DefaultTableCellRenderer
```

```
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {       int n=table.getRowCount()/12;
   int c=table.getRowCount()/(12*8);
   if(table.getRowCount()%(12*8)!=0)
           c=c+1;
   for(int d=1;d<table.getColumnCount();d=d+14){
           if(column==d){
   for(int k=0;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.red);
         return this;}}
   for(int k=3;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.black);
         return this;}}
   for(int s=0;s<=3;s++){
           int j=(s*n)/2;
            for(int i1=6*j+1;i1<=6*j+5;i1++){
                                 if(row==i1){
                                 setBackground(Color.magenta);
                                return this;
                                 }
                                 } }

   for(int m=0;m<c;m++)
   for(int j=0;j<=2;j++ ){
           int i1=(m+j*n)*4;
            int k1=2*c-1;
           int k=2*c;


                        if(j==0){
                                int i=m;
                                 for(;i<6+4*c+k;i++){
                                         if(row==i){
                                                 setBackground(Color.blue);
                                         return this;}
                                         }
```

```
                }}}        for(int
e=2,f=3;e<table.getColumnCount()&&f<table.getColumnCount();f=f+14,e=e+14){
                        for(int m=0;m<c;m++)
                           for(int j=0;j<=2;j++ ){
                                if(column==e&&j==1||column==f&&j==2){

                                int i1=(m+j*n)*4;
                                 int k1=2*c-1;
                                int k=2*c;
                                if(j==2){
                                if(table.getRowCount()==300)
                                  k1=2*c+5;
                                }
                                if(i1%6==0){
                                i1=i1+1;}
   for(;i1<=(m+j*n)*4+4+k1;i1++){
          if(row==i1){
                setBackground(Color.blue);
          return this;}
          }
          }
   }}}
 for(int k=0,j=3;j<table.getRowCount()&&k<table.getRowCount();k=k+6,j=j+6){
                        if(row!=k&&row!=j){
                           setBackground(Color.cyan);
                           return this;}
                        }
   return null;
   }
}
class MyRenderer24 extends DefaultTableCellRenderer
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
   {       int n=table.getRowCount()/12;
   int c=table.getRowCount()/(12*8);
   if(table.getRowCount()%(12*8)!=0)
          c=c+1;
   for(int d=1;d<table.getColumnCount();d=d+14){
          if(column==d){
```

```
    for(int k=0;k<table.getRowCount();k=k+6){
        if(row==k){
            setBackground(Color.black);
            return this;}}
    for(int k=3;k<table.getRowCount();k=k+6){
        if(row==k){
            setBackground(Color.red);
            return this;}}
    for(int s=0;s<=3;s++){
              int j=(s*n)/2;
               for(int i1=6*j+1;i1<=6*j+5;i1++){
                                        if(row==i1){
                                        setBackground(Color.magenta);
                                       return this;
                                        }
                                        } }

    for(int m=0;m<c;m++)
    for(int j=0;j<=2;j++ ){
              int i1=(m+j*n)*4;
               int k1=2*c-1;
              int k=2*c;
                        if(j==0){
                                    int i=m;
                                     for(;i<6+4*c+k;i++){
                                                if(row==i){
                                                            setBackground(Color.blue);
                                                return this;}
                                                }}}
              }}
    for(int k=0,j=3;j<table.getRowCount()&&k<table.getRowCount();k=k+6,j=j+6){
        if(row!=k&&row!=j){
            setBackground(Color.cyan);
            return this;}
        }
return null;
}
}
class MyRenderer25 extends DefaultTableCellRenderer
{
```

```
public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
{       int n=table.getRowCount()/12;
int c1=table.getRowCount()/(12*8);
if(table.getRowCount()%(12*8)!=0)
        c1=c1+1;
int c=2*c1;
for(int d=1;d<table.getColumnCount();d=d+14){
        if(column==d){
for(int k=0;k<table.getRowCount();k=k+6){
   if(row==k){
      setBackground(Color.red);
      return this;}}
for(int s=0;s<=3;s++){
        int j=(s*n)/2;
 for(int i=6*j+1;i<=6*j+4;i++){
         if(row==i){
         setBackground(Color.magenta);
         return this;
         }
         }}

for(int m=0;m<c;m++)
for(int j=0;j<=2;j++ ){
        int i1=(m+j*n)*4;
        int p=0;
         int k1=c-1;
        int k=c;
        int b=table.getRowCount()/60;
                if(j==0){
                        if(table.getRowCount()>300)
                                k=b+1;
                        if(table.getRowCount()==300)
                                k=c-1;

                         for(int i=m;i<5+4*c+k;i++){
                                if(row==i){
                                        setBackground(Color.blue);
                                return this;}
                                }
```

```
                }}}
        for(int e=2,f=3;e<table.getColumnCount()&&f<table.getColumnCount();f=f+14,e=e+14){
          for(int m=0;m<c;m++)
                for(int j=0;j<=2;j++ ){
                        if(column==e&&j==1||column==f&&j==2){
                        int i1=(m+j*n)*4;
                        int p=0;
                        int k1=c-1;
                        int k=c;
                        int b=table.getRowCount()/60;
                        if(i1%6==0){
                        i1=i1+1;}
                else k1--;
                        if(table.getRowCount()==72)
                                k1=k1+1;
                        if(table.getRowCount()>=600)
                                k1=b;
                        if(j==2)
                        {
                                if(table.getRowCount()>=600)
                                k1=k1+5;
                           if(table.getRowCount()==300)
                                   k1=k1-2;
                        }
   for(;i1<=(m+j*n)*4+4+k1;i1++){
          if(row==i1){
                setBackground(Color.blue);
          return this;}
          }
          }
   }}}
 for(int k=0;k<table.getRowCount();k=k+6){
                        if(row!=k){
                           setBackground(Color.cyan);
                           return this;}
                        }
   return null;
   }
}
class MyRenderer26 extends DefaultTableCellRenderer
```

```
{
   public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row, int column)
   {       int n=table.getRowCount()/12;
   int c1=table.getRowCount()/(12*8);
   if(table.getRowCount()%(12*8)!=0)
           c1=c1+1;
   int c=2*c1;
   for(int d=1;d<table.getColumnCount();d=d+14){
           if(column==d){
   for(int k=0;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.red);
         return this;}}
   for(int k=3;k<table.getRowCount();k=k+6){
      if(row==k){
         setBackground(Color.black);
         return this;}}
   for(int s=0;s<=3;s++){
           int j=(s*n)/2;
            for(int i1=6*j+1;i1<=6*j+5;i1++){
                                 if(row==i1){
                                 setBackground(Color.magenta);
                                return this;
                                 }
                                 } }

   for(int m=0;m<c;m++)
   for(int j=0;j<=2;j++ ){
           int i1=(m+j*n)*4;
            int k1=2*c-1;
           int k=2*c;


                        if(j==0){
                                int i=m;
                                 for(;i<6+4*c+k;i++){
                                         if(row==i){
                                                 setBackground(Color.blue);
                                         return this;}
```

```
                                        }
                    }}}
            for(int e=2,f=3;e<table.getColumnCount()&&f<table.getColumnCount();f=f+14,e=e+14){
             for(int m=0;m<c;m++)
                        for(int j=0;j<=2;j++ ){
                                if(column==e&&j==1||column==f&&j==2){

                                int i1=(m+j*n)*4;
                                 int k1=2*c-1;
                                int k=2*c;
            if(j==2){
                        if(table.getRowCount()>=72)
             k1=2*c+5;
            }
                                if(i1%6==0){
                                i1=i1+1;}
    for(;i1<=(m+j*n)*4+4+k1;i1++){
            if(row==i1){
                        setBackground(Color.blue);
            return this;}
            }
            }
    }}}
 for(int k=0,j=3;j<table.getRowCount()&&k<table.getRowCount();k=k+6,j=j+6){
                            if(row!=k&&row!=j){
                               setBackground(Color.cyan);
                               return this;}
                            }
    return null;
    }
}
class MyRenderer27 extends DefaultTableCellRenderer
{
    public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean
hasFocus, int row, int column)
    {       int n=table.getRowCount()/12;
    int c1=table.getRowCount()/(12*8);
    if(table.getRowCount()%(12*8)!=0)
            c1=c1+1;
    int c=2*c1;
```

```
for(int d=1;d<table.getColumnCount();d=d+14){
        if(column==d){
for(int k=0;k<table.getRowCount();k=k+6){
   if(row==k){
      setBackground(Color.black);
      return this;}}
for(int k=3;k<table.getRowCount();k=k+6){
   if(row==k){
      setBackground(Color.red);
      return this;}}
for(int s=0;s<=3;s++){
        int j=(s*n)/2;
         for(int i1=6*j+1;i1<=6*j+5;i1++){
                              if(row==i1){
                              setBackground(Color.magenta);
                             return this;
                              }
                              } }

for(int m=0;m<c;m++)
for(int j=0;j<=2;j++ ){
        int i1=(m+j*n)*4;
         int k1=2*c-1;
        int k=2*c;
                if(j==0){
                        int i=m;
                         for(;i<6+4*c+k;i++){
                                 if(row==i){
                                         setBackground(Color.blue);
                                 return this;}
                                 }
                }}}}
        for(int k=0,j=3;j<table.getRowCount()&&k<table.getRowCount();k=k+6,j=j+6){
             if(row!=k&&row!=j){
                setBackground(Color.cyan);
                return this;}}
               int p=0;
                if(table.getBackground().equals(Color.white))
                  p++;
                System.out.println("p  "+p);
```

```
return null;
}
}
```