import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Image;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.File;

import java.util.ArrayList;

import java.util.Arrays;

import org.jfree.chart.ChartFrame;

import org.jfree.chart.JFreeChart;

import org.jfree.chart.ChartFactory;

import org.jfree.chart.ChartUtilities;

import org.jfree.chart.plot.PlotOrientation;

import org.jfree.data.category.DefaultCategoryDataset;

import java.util.Scanner;

import javax.imageio.IIOException;

import javax.imageio.ImageIO;

import javax.swing.\*;

import Codes.Complex;

import Codes.gCorr;

import Codes.gCorrChart;

import Codes.gcskew;

import Codes.gcSkewChart;

import Codes.GenomicLength;

import Codes.iCorrChart;

import Codes.Optimum;

import Codes.peak;

import Codes.Upload;

import Codes.iCorr;

import Codes.progress;

public class MainWindow

{

public static String change(String nm)

{

char[] chararray=nm.toCharArray();

String hr="";

for(int i=0;i<chararray.length;i++)

{

if(chararray[i]=='\\')

{

chararray[i]='/';

System.out.println("1");

}

hr=hr+chararray[i];

//System.out.println(2);

}

//System.out.println(hr);

return hr;

}

/\*---------------------------\*/

/\*Functions For iCorr\*/

/\*---------------------------\*/

public static void frame1(final String name,final int len) //For selecting own window size and shift size

{

final JFrame fr = new JFrame("iCorr:Choose");

fr.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

fr.setVisible(true);

fr.setSize(300,400);

fr.setLocation(40,120);

fr.setResizable(false);

final JPanel p = new JPanel();

fr.add(p);

JLabel l1= new JLabel("Enter Window Size ");

JLabel l2=new JLabel("Enter Shift Size ");

final JTextField tx1= new JTextField(10);

final JTextField tx2= new JTextField(10);

tx1.setVisible(true);

tx2.setVisible(true);

p.add(l1);

p.add(tx1);

p.add(l2);

p.add(tx2);

JButton b = new JButton("GO");

p.add(b);

//Image img =

b.setToolTipText("Click this button to evaluate for the selected window size and shift size");

b.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e)

{

try

{

final double[] list;

final int ws = Integer.parseInt(tx1.getText());

final int ss = Integer.parseInt(tx2.getText());

iCorr icorr = new iCorr(name,ws,ss,len);

list=icorr.icorrfunction();//Varuni's code linked which returns an array; this array goes into hritik's code

final peak pk = new peak(list);

JFrame f = new JFrame("iCorr");

f.setResizable(false);

f.setSize(400,400);

JPanel panel = new JPanel();

f.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

f.setVisible(true);

f.add(panel);

final JTextArea textbox = new JTextArea();

textbox.setLayout(null);

textbox.setRows(10);

textbox.setColumns(30);

//textbox.setLineWrap(true);

JScrollPane scroll = new JScrollPane(textbox);

scroll.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL\_SCROLLBAR\_ALWAYS);

scroll.setVisible(true);

//scroll.setBounds(20,20,50,50);

//panel.add(scroll);

JButton bt1= new JButton ("Graph");

panel.add(bt1);

bt1.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e)

{

//hritik's code for graphs.. call function

iCorrChart icorrchart = new iCorrChart(list);

}

});

JButton bt2= new JButton ("Show Peaks");

panel.add(bt2);

bt2.addActionListener(new ActionListener()

{

//code for showing the top peaks

public void actionPerformed(ActionEvent e)

{

textbox.setText(null);

ArrayList<Integer> arraylistsix =new ArrayList<Integer>();

arraylistsix=pk.peakfunction1();

ArrayList<Double> arraylistfive = new ArrayList<Double>();

arraylistfive=pk.peakfunction2();

// int[] arrayclub=new int[arraylistsix.size()];

//Arrays.fill(arrayclub,-1);

for(int i=0;i<arraylistsix.size()-1;i++)

{

if(arraylistsix.get(i)!=-1)

{

int count=0;

int d=0;

for(int j=i+1;j<arraylistsix.size();j++)

{

if(arraylistsix.get(j)<=arraylistsix.get(i)+((int)(ws/ss)) && arraylistsix.get(j)>=arraylistsix.get(i)-((int)(ws/ss)))

{

//arrayclub[i]=count;

//arraylistsix.set(i,-100);

//arrayclub[j]=count;

//System.out.print("Overlapping Region encountered!");

count++;

if(count==1)

{

textbox.append("Overlapping Region encountered! \n");

}

textbox.append("Window Number= "+Integer.toString(arraylistsix.get(j)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(j)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(j)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(j))+"\n");

//textbox.append(Integer.toString(arraylistsix.get(j)+1)+',');

arraylistsix.set(j,-1);

d++;

}

}

if(d!=0)

{

//System.out.println("Window Number= "+Integer.toString(arraylistsix.get(i)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(i)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(i)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(i)));

textbox.append("Window Number= "+Integer.toString(arraylistsix.get(i)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(i)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(i)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(i))+"\n\n");

//textbox.append((Integer.toString(arraylistsix.get(i)+1))+"\n");

//arraylistsix.set(i,-100);

}

else

{

//System.out.println("Window Number= "+Integer.toString(arraylistsix.get(i)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(i)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(i)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(i)));

//textbox.append((Integer.toString(arraylistsix.get(i)+1))+"\n");

textbox.append("Window Number= "+Integer.toString(arraylistsix.get(i)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(i)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(i)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(i))+"\n\n");

}

count++;

}

}

if(arraylistsix.get(arraylistsix.size()-1)!=-1)

{

//System.out.println("Window Number= " + arraylistsix.get(arraylistsix.size()-1)+1+' '+"Region: "+Integer.toString((arraylistsix.get(arraylistsix.size()-1)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(arraylistsix.size()-1)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+ Double.toString(arraylistfive.get(arraylistsix.size()-1)));

//textbox.append((arraylistsix.get(arraylistsix.size()-1))+"\n");

textbox.append("Window Number= " + Integer.toString(arraylistsix.get(arraylistsix.size()-1)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(arraylistsix.size()-1)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(arraylistsix.size()-1)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+ Double.toString(arraylistfive.get(arraylistsix.size()-1))+"\n");

}

//System.out.println(Arrays.toString(arrayclub));

//System.out.println(arraylistsix);

}

});

JButton bt3 = new JButton("Show values");

panel.add(bt3);

bt3.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e)

{

textbox.setText(null);

for (int i=0; i<list.length;i++)

{

textbox.append(Double.toString(list[i]));

textbox.append("\n");

}

//show (peak-avg)/sd result for the peak

}

});

panel.add(scroll);

}

catch(Exception e1)

{

JFrame frame = new JFrame("Error");

frame.setLocation(100,200);

frame.setResizable(false);

frame.setSize(350, 200);

JPanel panel = new JPanel();

JTextArea label = new JTextArea("An Error has occurred\nIt could be due to any of the following reasons:\n1.The window size entered is greater than the shift size\n2.One or both of the numbers is not a positive integer");

label.setSize(300,150);

label.setWrapStyleWord(true);

label.setLineWrap(true);

label.setEditable(false);

label.setFocusable(false);

frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

frame.add(panel);

panel.add(label);

}

}

});

}

public static void frame2(String name, int len) //For Automated window size and shift size

{

JFrame fr = new JFrame("iCorr: Most Optimal");

fr.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

fr.setVisible(true);

fr.setSize(400,500);

fr.setLocation(40,800);

fr.setResizable(false);

//Call for calculating the values:iCorr

final iCorr icorr1 = new iCorr(name,1000,1000,len); //arraylist 5 fetch

double[] list1;

JTabbedPane tabbed = new JTabbedPane();

fr.add(tabbed);

//TAB1: For Showing the progress

JPanel p1 = new JPanel();

JTextArea progressarea = new JTextArea();

progressarea.setText(null);

p1.add(progressarea);

tabbed.add(p1,"Progress");

//Tab2: For Showing the optimised window size and shift size

JPanel p2 = new JPanel();

JTextArea text = new JTextArea(); //FOR SHOWING THE MESSAGE ABOUT WHICH IS THE BEST POSSIBLE WS AND SS

//text.setLineWrap(true);

text.setEditable(false);

p2.add(text);

tabbed.add(p2,"Optimised Sizes");

//TAB3: For showing the data values of code

JPanel p3 = new JPanel();

JTextArea textbox = new JTextArea(); //TO SHOW THE DATA VALUES OF CODE

textbox.setColumns(20);

textbox.setRows(10);

textbox.setLayout(null);

JScrollPane scroll = new JScrollPane (textbox); //ADDED TO SHOW DATA PART OF CODE

scroll.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL\_SCROLLBAR\_ALWAYS);

scroll.setVisible(true);

scroll.setBounds(20,20,50,50);

textbox.setLineWrap(true);

textbox.setEditable(false);

textbox.setVisible(true);

p3.add(textbox);

p3.add(scroll);

tabbed.add(p3,"Data");

//TAB4: To show the value of peaks

JPanel p4 = new JPanel();

JTextArea textpeaks = new JTextArea(); //TO SHOW THE VALUE OF PEAKS

textpeaks.setColumns(20);

textpeaks.setRows(10);

textpeaks.setLayout(null);

textpeaks.setEditable(false);

textpeaks.setWrapStyleWord(true);

textpeaks.setLineWrap(true);

textpeaks.setEditable(false);

textpeaks.setVisible(true);

JScrollPane scrollbar = new JScrollPane (textpeaks); //ADDED TO SHOW PEAKS

scrollbar.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL\_SCROLLBAR\_ALWAYS);

scrollbar.setVisible(true);

scrollbar.setBounds(20,20,20,10);

p4.add(textpeaks);

p4.add(scrollbar);

tabbed.add(p4,"Peaks");

//------------------

//COMPUTATION

//------------------

list1 = icorr1.icorrfunction();

peak pk1 = new peak(list1);

ArrayList<Double> arraylistfive1 = new ArrayList<Double>();

arraylistfive1 = pk1.peakfunction2();

//progress prog = null;

//prog.progressfn(1);

progressarea.setText(" 10%complete.....");

//icorr1.start();

iCorr icorr2 = new iCorr(name,2000,1000,len);

double[] list2;

list2 = icorr2.icorrfunction();

peak pk2 = new peak(list2);

ArrayList<Double> arraylistfive2 = new ArrayList<Double>();

arraylistfive2 = pk2.peakfunction2();

progressarea.setText(null);

//prog.progressfn(2);

progressarea.setText(" 20%complete.....");

//icorr2.start();

iCorr icorr3 = new iCorr(name,3000,1000,len);

double[] list3;

list3 = icorr3.icorrfunction();

peak pk3 = new peak(list3);

ArrayList<Double> arraylistfive3 = new ArrayList<Double>();

arraylistfive3 = pk3.peakfunction2();

progressarea.setText(null);

//prog.progressfn(3);

progressarea.setText(" 30%complete.....");

//icorr3.start();

iCorr icorr4 = new iCorr(name,4000,1000,len);

double[] list4;

list4 = icorr4.icorrfunction();

peak pk4 = new peak(list4);

ArrayList<Double> arraylistfive4 = new ArrayList<Double>();

arraylistfive4 = pk4.peakfunction2();

progressarea.setText(null);

//prog.progressfn(4);

progressarea.setText(" 40%complete.....");

//icorr4.start();

iCorr icorr5 = new iCorr(name,5000,1000,len);

double[] list5;

list5 = icorr5.icorrfunction();

peak pk5 = new peak(list5);

ArrayList<Double> arraylistfive5 = new ArrayList<Double>();

arraylistfive5 = pk5.peakfunction2();

progressarea.setText(null);

//prog.progressfn(5);

progressarea.setText(" 50%complete.....");

//icorr5.start();

iCorr icorr6 = new iCorr(name,1000,500,len);

double[] list6;

list6 = icorr6.icorrfunction();

peak pk6 = new peak(list6);

ArrayList<Double> arraylistfive6 = new ArrayList<Double>();

arraylistfive6 = pk6.peakfunction2();

progressarea.setText(null);

//prog.progressfn(6);

progressarea.setText(" 60%complete.....");

//icorr6.start();

iCorr icorr7 = new iCorr(name,2000,500,len);

double[] list7;

list7 = icorr7.icorrfunction();

peak pk7 = new peak(list7);

ArrayList<Double> arraylistfive7 = new ArrayList<Double>();

arraylistfive7 = pk7.peakfunction2();

progressarea.setText(null);

//prog.progressfn(7);

progressarea.setText(" 70%complete.....");

//icorr7.start();

iCorr icorr8 = new iCorr(name,3000,500,len);

double[] list8;

list8 = icorr8.icorrfunction();

peak pk8 = new peak(list8);

ArrayList<Double> arraylistfive8 = new ArrayList<Double>();

arraylistfive8 = pk8.peakfunction2();

progressarea.setText(null);

//prog.progressfn(8);

progressarea.setText(" 80%complete.....");

//icorr8.start();

iCorr icorr9 = new iCorr(name,4000,500,len);

double[] list9;

list9 = icorr9.icorrfunction();

peak pk9 = new peak(list9);

ArrayList<Double> arraylistfive9 = new ArrayList<Double>();

arraylistfive9 = pk9.peakfunction2();

progressarea.setText(null);

//prog.progressfn(9);

progressarea.setText(" 90%complete.....");

//icorr9.start();

iCorr icorr10 = new iCorr(name,5000,500,len);

double[] list10;

list10 = icorr10.icorrfunction();

peak pk10 = new peak(list10);

ArrayList<Double> arraylistfive10 = new ArrayList<Double>();

arraylistfive10 = pk10.peakfunction2();

progressarea.setText(null);

//prog.progressfn(10);

progressarea.setText(" 100%complete !");

//icorr10.start();

Optimum optimum = new Optimum(arraylistfive1,arraylistfive2,arraylistfive3,arraylistfive4,arraylistfive5,arraylistfive6,arraylistfive7,arraylistfive8,arraylistfive9,arraylistfive10);

int num = optimum.optimumfunction();

System.out.println("num value is "+num);

double[] array = null;

ArrayList<Integer> arraylistsix=null;

ArrayList<Double> arraylistfive=null;

int ws=0,ss=0;

switch (num)

{

case 1: text.append("1000,1000 is the best possible window size");

array=icorr1.icorrfunction();

arraylistsix=pk1.peakfunction1();

arraylistfive=pk1.peakfunction2();

ws=1000;

ss=1000;

break;

case 2: text.append("2000,1000 is the best possible window size");

array=icorr2.icorrfunction();

arraylistsix=pk2.peakfunction1();

arraylistfive=pk2.peakfunction2();

ws=2000;

ss=1000;

break;

case 3: text.append("3000,1000 is the best possible window size");

array=icorr3.icorrfunction();

arraylistsix=pk3.peakfunction1();

arraylistfive=pk3.peakfunction2();

ws=3000;

ss=1000;

break;

case 4: text.append("4000,1000 is the best possible window size");

array=icorr4.icorrfunction();

arraylistsix=pk4.peakfunction1();

arraylistfive=pk4.peakfunction2();

ws=4000;

ss=1000;

break;

case 5: text.append("5000,1000 is the best possible window size");

array=icorr5.icorrfunction();

arraylistsix=pk5.peakfunction1();

arraylistfive=pk5.peakfunction2();

ws=5000;

ss=1000;

break;

case 6: text.append("1000,500 is the best possible window size");

array=icorr6.icorrfunction();

arraylistsix=pk6.peakfunction1();

arraylistfive=pk6.peakfunction2();

ws=1000;

ss=500;

break;

case 7: text.append("2000,500 is the best possible window size");

array=icorr7.icorrfunction();

arraylistsix=pk7.peakfunction1();

arraylistfive=pk7.peakfunction2();

ws=2000;

ss=500;

break;

case 8: text.append("3000,500 is the best possible window size");

array=icorr8.icorrfunction();

arraylistsix=pk8.peakfunction1();

arraylistfive=pk8.peakfunction2();

ws=3000;

ss=500;

break;

case 9: text.append("4000,500 is the best possible window size");

array=icorr9.icorrfunction();

arraylistsix=pk9.peakfunction1();

arraylistfive=pk9.peakfunction2();

ws=4000;

ss=500;

break;

case 10: text.append("5000,500 is the best possible window size");

array=icorr10.icorrfunction();

arraylistsix=pk10.peakfunction1();

arraylistfive=pk10.peakfunction2();

ws=5000;

ss=500;

break;

}

iCorrChart chart = new iCorrChart(array);

//int[] arrayclub=new int[arraylistsix.size()];

//Arrays.fill(arrayclub,-1);

//int count=0;

for(int i=0;i<arraylistsix.size()-1;i++)

{

if(arraylistsix.get(i)!=-1)

{

int count=0;

int d=0;

for(int j=i+1;j<arraylistsix.size();j++)

{

if(arraylistsix.get(j)<=arraylistsix.get(i)+((int)(ws/ss)) && arraylistsix.get(j)>=arraylistsix.get(i)-((int)(ws/ss)))

{

//arrayclub[i]=count;

//arraylistsix.set(i,-100);

//arrayclub[j]=count;

count++;

if(count==1)

{

textpeaks.append("Overlapping Region encountered! \n");

}

textpeaks.append("Window Number= "+Integer.toString(arraylistsix.get(j)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(j)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(j)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(j))+' ');

//textbox.append(Integer.toString(arraylistsix.get(j)+1)+',');

arraylistsix.set(j,-1);

d++;

}

}

if(d!=0)

{

//System.out.println("Window Number= "+Integer.toString(arraylistsix.get(i)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(i)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(i)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(i)));

textpeaks.append("Window Number= "+Integer.toString(arraylistsix.get(i)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(i)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(i)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(i))+"\n\n");

//textbox.append((Integer.toString(arraylistsix.get(i)+1))+"\n");

//arraylistsix.set(i,-100);

}

else

{

//System.out.println("Window Number= "+Integer.toString(arraylistsix.get(i)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(i)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(i)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(i)));

//textbox.append((Integer.toString(arraylistsix.get(i)+1))+"\n");

textpeaks.append("Window Number= "+Integer.toString(arraylistsix.get(i)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(i)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(i)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+Double.toString(arraylistfive.get(i))+"\n\n");

}

//count++;

}

}

if(arraylistsix.get(arraylistsix.size()-1)!=-1)

{

//System.out.println("Window Number= " + arraylistsix.get(arraylistsix.size()-1)+1+' '+"Region: "+Integer.toString((arraylistsix.get(arraylistsix.size()-1)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(arraylistsix.size()-1)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+ Double.toString(arraylistfive.get(arraylistsix.size()-1)));

//textbox.append((arraylistsix.get(arraylistsix.size()-1))+"\n");

textpeaks.append("Window Number= " + Integer.toString(arraylistsix.get(arraylistsix.size()-1)+1)+' '+"Region: "+Integer.toString((arraylistsix.get(arraylistsix.size()-1)+1)\*ss)+'-'+Integer.toString((arraylistsix.get(arraylistsix.size()-1)+1)\*ss+ws)+' '+"Ratio(peak-avg/stdev): "+ Double.toString(arraylistfive.get(arraylistsix.size()-1))+"\n");

}

//System.out.println(Arrays.toString(arrayclub));

//System.out.println(arraylistsix);

for (int i=0;i<array.length;i++)

{

textbox.append(Double.toString(array[i]));

textbox.append("\n");

}

}

/\*-----------------------------------------------------------------------\*/

/\*-----------------------------------------------------------------------\*/

/\* MAIN FUNCTION STARTS \*/

/\*-----------------------------------------------------------------------\*/

/\*-----------------------------------------------------------------------\*/

public static void main(String[] args)

{

String name=null;

final double[] geneLength;

JFrame frame = new JFrame("DISA Software");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(350,400);

frame.setVisible(true);

frame.setResizable(false);

JTabbedPane tab = new JTabbedPane();

frame.add(tab);

JPanel panel = new JPanel();

/\*--------------------------------------------------------------------\*/

/\*Tab1: To Upload A file which will be subsequently used for all calculations in icorr, gcorr\*/

/\*--------------------------------------------------------------------\*/

//tab.add("File Up",panel);

Upload up = new Upload(tab);

//System.out.println(10);

while(name==null)

{

String s = up.getName();

if(s!=null)

name =s;

}

//the included new code

//System.out.println(1000);

final String hr=change(name);

GenomicLength gl = new GenomicLength(hr);

geneLength = gl.getGL();

/\*--------------------------------------------------------------------\*/

//Tab2:For icorr

/\*--------------------------------------------------------------------\*/

panel = new JPanel();

panel.setBackground(Color.PINK);

//Frame1: For Self choice of ws and ss

JButton b1 = new JButton("Choose your own window size");

b1.setLocation(2,20);

b1.setSize(50,20);

b1.addActionListener(new ActionListener()

{

public void actionPerformed (ActionEvent e)

{

frame1(hr,(int)geneLength[4]);

}

});

/\*--------------------------------------------------------------------\*/

//Frame2: For Automatic Optimisation

/\*--------------------------------------------------------------------\*/

JButton b2 = new JButton("Automatically Optimise");

b2.setLocation(20,20);

b2.setSize(50,20);

b2.addActionListener(new ActionListener()

{

public void actionPerformed (ActionEvent e)

{

frame2(hr,(int)geneLength[4]);

}

});

panel.add(b1);

panel.add(b2);

tab.add("iCorr", panel);

//Tab3:gCorr

panel=new JPanel();

//String str1,str2;

JLabel l1= new JLabel("Enter The Window Size ");

JLabel l2=new JLabel("Enter The Shift Size ");

final JTextField tx1= new JTextField(10);

final JTextField tx2= new JTextField(10);

tx1.setVisible(true);

tx2.setVisible(true);

JButton button = new JButton("Okay");

button.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e)

{

int ws = Integer.parseInt(tx1.getText());

int ss = Integer.parseInt(tx2.getText());

//System.out.println(hr);

//System.out.println((int)geneLength[4]);

gCorrChart gcorrchart = new gCorrChart(hr,ws,ss,(int)geneLength[4]);

}

});

panel.add(l1);

panel.add(tx1);

panel.add(l2);

panel.add(tx2);

panel.add(button, BorderLayout.CENTER);

panel.setBackground(Color.PINK);

tab.add("gCorr", panel);

//Tab 4:GCSkew

panel=new JPanel();

//String str1,str2;

JLabel l11= new JLabel("Enter The Window Size ");

JLabel l22=new JLabel("Enter The Shift Size ");

final JTextField tx11= new JTextField(10);

final JTextField tx22= new JTextField(10);

tx11.setVisible(true);

tx22.setVisible(true);

JButton button1 = new JButton("Okay");

button1.addActionListener(new ActionListener()

{

public void actionPerformed(ActionEvent e1)

{

int ws = Integer.parseInt(tx11.getText());

int ss = Integer.parseInt(tx22.getText());

//System.out.println(hr);

//System.out.println((int)geneLength[4]);

gcSkewChart gcskewchart = new gcSkewChart(hr,ws,ss,(int)geneLength[4]);

}

});

panel.add(l11);

panel.add(tx11);

panel.add(l22);

panel.add(tx22);

panel.add(button1);

panel.setBackground(Color.PINK);

tab.add("GC Skew", panel);

//Tab5

panel=new JPanel();

JTextArea area=new JTextArea("This Software is developed by:\n1.Shreya Johri\n2.Hritik Bansal\n3.Varuni Sarwal\nstudents of IIT Delhi, under the guidance of Prof. Kushal K. Shah. This software is for the scientific community to explore and give feedback so that we are able to further develop the iCorr Method and make our predictions more accurate.\nContact us at: icorr@gmail.com");

area.setSize(300,100);

area.setWrapStyleWord(true);

area.setLineWrap(true);

area.setOpaque(false);

area.setEditable(false);

area.setFocusable(false);

panel.add(area);

tab.add("About Us",panel);

}

}