# Java code :

## Genetic Algorithm class :

This class has the main functions for the genetic algorithm like crossover,mutation,fitness function,and bubble sort to sort the population.

### 1-cross over function :

| public ArrayList<int[]> cross\_over() {  //creat array list to store childes  ArrayList<int[]> childes = new ArrayList<>();  //get the first and second parents  int[] p1 = arr1.get(0).clone();  int[] p2 = arr1.get(1).clone();  Set<Integer> s = new HashSet<>();  //creat childes arrays  int[] c1 = p1.clone();  int[] c2 = p2.clone();   //left and right variables which represent the part where the cross over should happened  Random random = new Random();  int l = random.ints(0, arr.size()).findFirst().getAsInt();  int r = random.ints(l, arr.size()).findFirst().getAsInt();  while (l == r) {  l = random.ints(0, arr.size()).findFirst().getAsInt();  r = random.ints(l, arr.size()).findFirst().getAsInt();  }  //if left is bigger than right swap them  if (r < l) {  int temp;  temp = l;  l = r;  r = temp;  }  //from left to right swap  for (int i = l; i <= r; i++) {  int temp;  temp = c1[i];  c1[i] = c2[i];  c2[i] = temp;  }  //mark the dublicated projects as -1  for (int i = 0; i < l; i++) {  for (int j = l; j <= r; j++) {  if (c1[i] == c1[j])  c1[j] = -1;  }  }  for (int i = r + 1; i < c1.length; i++) {  for (int j = l; j <= r; j++) {  if (c1[i] == c1[j])  c1[j] = -1;  }  }  for (int i = 0; i < l; i++) {  for (int j = l; j <= r; j++) {  if (c2[i] == c2[j])  c2[j] = -1;  }  }  for (int i = r + 1; i < c2.length; i++) {  for (int j = l; j <= r; j++) {  if (c2[i] == c2[j])  c2[j] = -1;  }  }   //creat set from all projects in the two parents  for (int i = 0; i < arr.size(); i++) {  s.add(p1[i]);  s.add(p2[i]);  }   //convert the list to two array lists  ArrayList<Integer> temp1 = new ArrayList<>(s);  ArrayList<Integer> temp2 = new ArrayList<>(s);   //creat two hash maps  HashMap<Integer, Boolean> p11 = new HashMap<>();  HashMap<Integer, Boolean> p22 = new HashMap<>();  //fill the hashmap with false for all the values of the parents  for (int i = 0; i < s.size(); i++) {  p11.put(temp1.get(i), false);  p22.put(temp2.get(i), false);  }  //mark the value from childes which does not equal -1 to true in the hashmaps  for (int i = 0; i < arr.size(); i++) {  if (c1[i] != -1) {   p11.put(c1[i], true);   }  if (c2[i] != -1) {   p22.put(c2[i], true);   }   }   for (int i = l; i <= r; i++) {  if (c2[i] == -1) {  for (Map.Entry<Integer, Boolean> u : p22.entrySet()) {  if (!u.getValue()) {  c2[i] = u.getKey();  p22.put(c2[i], true);  break;  }   }  }  }  for (int i = l; i <= r; i++) {  if (c1[i] == -1) {  for (Map.Entry<Integer, Boolean> u : p11.entrySet()) {  if (!u.getValue()) {  c1[i] = u.getKey();  p11.put(c1[i], true);  break;  }   }  }  }  HashMap<Integer, Integer> h = new HashMap<>();  for (int i = 0; i < arr.size(); i++) {  if (h.containsKey(c1[i]))  h.put(c1[i], h.get(c1[i]) + 1);  else  h.put(c1[i], 1);    }  /\*HashMap<Integer,Integer>h2 = new HashMap<>() ;  for (int i = 0 ; i < arr.size() ; i++){  if (h2.containsKey(c2[i]))  h2.put(c2[i] , h2.get(c2[i])+1) ;  else  h2.put(c2[i] , 1) ;   }  System.out.println(h +"\n"+ h2);\*/   childes.add(c1);  childes.add(c2);   return childes; } |
| --- |

### 2-mutation function :

| //mutation function public void mutation() {  ArrayList<int[]> result = new ArrayList<>();  Random random = new Random();  //generate two random numbers for chose two different indexes to swap it  int l = random.ints(0, arr.size()).findFirst().getAsInt();  int r = random.ints(0, arr.size()).findFirst().getAsInt();  //creat child 1 and do the mutation for it   int temp = arr1.get(arr1.size() - 1)[l];  arr1.get(arr1.size() - 1)[l] = arr1.get(arr1.size() - 1)[r];  arr1.get(arr1.size() - 1)[r] = temp;  //do the same thing for child 2  l = random.ints(0, arr.size()).findFirst().getAsInt();  r = random.ints(0, arr.size()).findFirst().getAsInt();  int[] c2 = arr1.get(arr1.size() - 2);  temp = arr1.get(arr1.size() - 2)[l];  arr1.get(arr1.size() - 2)[l] = arr1.get(arr1.size() - 2)[r];  arr1.get(arr1.size() - 2)[r] = temp;  } |
| --- |

### 3- fitness function :

| //fitness function public int fitness\_fun(int[] arr1) {  int sum = 0;  for (int j = 0; j < arr.size(); j++) {  if (arr1[j] == arr.get(j).getF())  sum += 111;  else if (arr1[j] == arr.get(j).getS())  sum += 74;  else if (arr1[j] == arr.get(j).getT())  sum += 37;  }  return sum; } |
| --- |

### 4- bubble sort function :

| public ArrayList<int[]> bubblesort() {  int n = arr1.size();  for (int i = 0; i < n - 1; i++)  for (int j = 0; j < n - i - 1; j++)  if (fitness\_fun(arr1.get(j)) < fitness\_fun(arr1.get(j + 1))) {  // swap arr[j+1] and arr[j]  Collections.swap(arr1, j, j + 1);   }  return arr1; } |
| --- |

### All class code :

| package sample;  import java.util.\*;  public class geniticalgorithim {  ArrayList<info> arr;  private ArrayList<int[]> arr1;   public geniticalgorithim(ArrayList<int[]> arr1, ArrayList<info> arr) {  this.arr1 = arr1;  this.arr = arr;  }   //fitness function  public int fitness\_fun(int[] arr1) {  int sum = 0;  for (int j = 0; j < arr.size(); j++) {  if (arr1[j] == arr.get(j).getF())  sum += 111;  else if (arr1[j] == arr.get(j).getS())  sum += 74;  else if (arr1[j] == arr.get(j).getT())  sum += 37;  }  return sum;  }    //for sorting the list of population based on fitness function  public ArrayList<int[]> bubblesort() {  int n = arr1.size();  for (int i = 0; i < n - 1; i++)  for (int j = 0; j < n - i - 1; j++)  if (fitness\_fun(arr1.get(j)) < fitness\_fun(arr1.get(j + 1))) {  // swap arr[j+1] and arr[j]  Collections.swap(arr1, j, j + 1);   }  return arr1;  }   //cross over function  public ArrayList<int[]> cross\_over() {  //creat array list to store childes  ArrayList<int[]> childes = new ArrayList<>();  //get the first and second parents  int[] p1 = arr1.get(0).clone();  int[] p2 = arr1.get(1).clone();  Set<Integer> s = new HashSet<>();  //creat childes arrays  int[] c1 = p1.clone();  int[] c2 = p2.clone();   //left and right variables which represent the part where the cross over should happened  Random random = new Random();  int l = random.ints(0, arr.size()).findFirst().getAsInt();  int r = random.ints(l, arr.size()).findFirst().getAsInt();  while (l == r) {  l = random.ints(0, arr.size()).findFirst().getAsInt();  r = random.ints(l, arr.size()).findFirst().getAsInt();  }  //if left is bigger than right swap them  if (r < l) {  int temp;  temp = l;  l = r;  r = temp;  }  //from left to right swap  for (int i = l; i <= r; i++) {  int temp;  temp = c1[i];  c1[i] = c2[i];  c2[i] = temp;  }  //mark the dublicated projects as -1  for (int i = 0; i < l; i++) {  for (int j = l; j <= r; j++) {  if (c1[i] == c1[j])  c1[j] = -1;  }  }  for (int i = r + 1; i < c1.length; i++) {  for (int j = l; j <= r; j++) {  if (c1[i] == c1[j])  c1[j] = -1;  }  }  for (int i = 0; i < l; i++) {  for (int j = l; j <= r; j++) {  if (c2[i] == c2[j])  c2[j] = -1;  }  }  for (int i = r + 1; i < c2.length; i++) {  for (int j = l; j <= r; j++) {  if (c2[i] == c2[j])  c2[j] = -1;  }  }   //creat set from all projects in the two parents  for (int i = 0; i < arr.size(); i++) {  s.add(p1[i]);  s.add(p2[i]);  }   //convert the list to two array lists  ArrayList<Integer> temp1 = new ArrayList<>(s);  ArrayList<Integer> temp2 = new ArrayList<>(s);   //creat two hash maps  HashMap<Integer, Boolean> p11 = new HashMap<>();  HashMap<Integer, Boolean> p22 = new HashMap<>();  //fill the hashmap with false for all the values of the parents  for (int i = 0; i < s.size(); i++) {  p11.put(temp1.get(i), false);  p22.put(temp2.get(i), false);  }  //mark the value from childes which does not equal -1 to true in the hashmaps  for (int i = 0; i < arr.size(); i++) {  if (c1[i] != -1) {   p11.put(c1[i], true);   }  if (c2[i] != -1) {   p22.put(c2[i], true);   }   }   for (int i = l; i <= r; i++) {  if (c2[i] == -1) {  for (Map.Entry<Integer, Boolean> u : p22.entrySet()) {  if (!u.getValue()) {  c2[i] = u.getKey();  p22.put(c2[i], true);  break;  }   }  }  }  for (int i = l; i <= r; i++) {  if (c1[i] == -1) {  for (Map.Entry<Integer, Boolean> u : p11.entrySet()) {  if (!u.getValue()) {  c1[i] = u.getKey();  p11.put(c1[i], true);  break;  }   }  }  }  HashMap<Integer, Integer> h = new HashMap<>();  for (int i = 0; i < arr.size(); i++) {  if (h.containsKey(c1[i]))  h.put(c1[i], h.get(c1[i]) + 1);  else  h.put(c1[i], 1);    }  /\*HashMap<Integer,Integer>h2 = new HashMap<>() ;  for (int i = 0 ; i < arr.size() ; i++){  if (h2.containsKey(c2[i]))  h2.put(c2[i] , h2.get(c2[i])+1) ;  else  h2.put(c2[i] , 1) ;   }  System.out.println(h +"\n"+ h2);\*/   childes.add(c1);  childes.add(c2);   return childes;  }   //mutation function  public void mutation() {  ArrayList<int[]> result = new ArrayList<>();  Random random = new Random();  //generate two random numbers for chose two different indexes to swap it  int l = random.ints(0, arr.size()).findFirst().getAsInt();  int r = random.ints(0, arr.size()).findFirst().getAsInt();  //creat child 1 and do the mutation for it   int temp = arr1.get(arr1.size() - 1)[l];  arr1.get(arr1.size() - 1)[l] = arr1.get(arr1.size() - 1)[r];  arr1.get(arr1.size() - 1)[r] = temp;  //do the same thing for child 2  l = random.ints(0, arr.size()).findFirst().getAsInt();  r = random.ints(0, arr.size()).findFirst().getAsInt();  int[] c2 = arr1.get(arr1.size() - 2);  temp = arr1.get(arr1.size() - 2)[l];  arr1.get(arr1.size() - 2)[l] = arr1.get(arr1.size() - 2)[r];  arr1.get(arr1.size() - 2)[r] = temp;    }    public ArrayList<int[]> getArr1() {  return arr1;  }   public void setArr1(ArrayList<int[]> arr1) {  this.arr1 = arr1;  }   public ArrayList<info> getArr() {  return arr;  }   public void setArr(ArrayList<info> arr) {  this.arr = arr;  } } |
| --- |

## Initial solutions class :

In this class there is the function that creates the random initial solution population that the genetic algorithm will optimize it .

### Initial solution function :

| public int[] sol() {  Random r = new Random();  //define the chromosome  int[] d = new int[size + 1];  //define visited array  boolean[] vis = new boolean[Allprojects.size() + 1];   Arrays.fill(vis, false);  Arrays.fill(d, 0);  int y = SelectedProjects.size();  //file the first part of the solution []d with projects that chosen by all groups  for (int i = 0; i < SelectedProjects.size(); i++) {  //chose random index from the selected projects  int x = r.ints(0, y).findFirst().getAsInt();  //check if that index is visited  if (!vis[SelectedProjects.get(x)])  d[i] = SelectedProjects.get(x);   else {  //if visited find un visited one  while (vis[SelectedProjects.get(x)])  x = r.ints(0, y).findFirst().getAsInt();   d[i] = SelectedProjects.get(x);   }  //mark the chosen project as visited  vis[SelectedProjects.get(x)] = true;  }  int l = Allprojects.size();  //do the same thing for the other part of the array d  for (int i = SelectedProjects.size(); i < Allprojects.size(); i++) {   int x = r.ints(0, l).findFirst().getAsInt();  if (!vis[Allprojects.get(x)])  d[i] = Allprojects.get(x);   else {  while (vis[Allprojects.get(x)])  x = r.ints(0, l).findFirst().getAsInt();   d[i] = Allprojects.get(x);   }   vis[Allprojects.get(x)] = true;  }   return d; } |
| --- |

### All class code :

| package sample; import java.util.ArrayList; import java.util.Arrays; import java.util.Random; //define the class for initial solution public class initialsol {  private final ArrayList<Integer> SelectedProjects;  private final ArrayList<Integer> Allprojects;  private final int size;   public initialsol(ArrayList<Integer> SelectedProjects, ArrayList<Integer> Allprojects, int size) {  this.SelectedProjects = SelectedProjects;  this.Allprojects = Allprojects;  this.size = size;  }   //function to generate initial solution  public int[] sol() {  Random r = new Random();  //define the chromosome  int[] d = new int[size + 1];  //define visited array  boolean[] vis = new boolean[Allprojects.size() + 1];   Arrays.fill(vis, false);  Arrays.fill(d, 0);  int y = SelectedProjects.size();  //file the first part of the solution []d with projects that chosen by all groups  for (int i = 0; i < SelectedProjects.size(); i++) {  //chose random index from the selected projects  int x = r.ints(0, y).findFirst().getAsInt();  //check if that index is visited  if (!vis[SelectedProjects.get(x)])  d[i] = SelectedProjects.get(x);   else {  //if visited find un visited one  while (vis[SelectedProjects.get(x)])  x = r.ints(0, y).findFirst().getAsInt();   d[i] = SelectedProjects.get(x);   }  //mark the chosen project as visited  vis[SelectedProjects.get(x)] = true;  }  int l = Allprojects.size();  //do the same thing for the other part of the array d  for (int i = SelectedProjects.size(); i < Allprojects.size(); i++) {   int x = r.ints(0, l).findFirst().getAsInt();  if (!vis[Allprojects.get(x)])  d[i] = Allprojects.get(x);   else {  while (vis[Allprojects.get(x)])  x = r.ints(0, l).findFirst().getAsInt();   d[i] = Allprojects.get(x);   }   vis[Allprojects.get(x)] = true;  }   return d;  }   } |
| --- |

### 

## The controller class :

In this class there is the main function which run the program.

Class code :

| package sample; import javafx.collections.FXCollections; import javafx.collections.ObservableList; import javafx.event.ActionEvent; import javafx.fxml.FXML; import javafx.fxml.FXMLLoader; import javafx.scene.Parent; import javafx.scene.Scene; import javafx.scene.control.\*; import javafx.scene.control.cell.PropertyValueFactory; import javafx.stage.FileChooser; import javafx.stage.Stage; import java.io.BufferedReader; import java.io.File; import java.io.FileReader; import java.io.IOException; import java.util.ArrayList; import java.util.HashSet; import java.util.Random; import java.util.Set; public class Controller {  public static int fi = 0, sec = 0, th = 0, other = 0;  static ArrayList<chart> charts = new ArrayList<>();  ObservableList<tabelView> list = FXCollections.observableArrayList();  Boolean checkSubmit = false;  @FXML  private Button btnSubmit;  @FXML  private TextField txtNumLoop;  @FXML  private TextField txtNumRun;  @FXML  private TextField txtPathFile;  @FXML  private Button btnBrowse;  @FXML  private TableColumn<tabelView, Integer> alg;  @FXML  private TableColumn<tabelView, Integer> first;  @FXML  private TableColumn<tabelView, String> groups;  @FXML  private TableColumn<tabelView, Integer> second;  @FXML  private TableColumn<tabelView, Integer> third;  @FXML  private TableView<tabelView> tabel;  @FXML  private TableColumn<tabelView, Integer> number;   @FXML  void showchart(ActionEvent event) {   try {  if (checkSubmit) {  Parent root = FXMLLoader.load(getClass().getResource("chart.fxml"));  Stage stage2 = new Stage();  stage2.setScene(new Scene(root));  stage2.show();    } else {  throw new IllegalAccessException("Pleas fill the information and submit");  }  } catch (IllegalArgumentException | IOException | IllegalAccessException argumentException) {   Alert alertCreat = new Alert(Alert.AlertType.WARNING);  alertCreat.setTitle("Warning");  alertCreat.setHeaderText(null);  alertCreat.setContentText(argumentException.getMessage());  alertCreat.showAndWait();   }   }    @FXML  void btnSubmitAction(ActionEvent event) throws Exception {    try {   if (txtNumLoop.getText().matches("[0-9]+")) {   } else if (txtNumLoop.getText().equals("")) {   throw new IllegalArgumentException("Pleas enter number of iteration");   } else {   throw new IllegalArgumentException("Pleas enter digits only !!");   }   if (!txtPathFile.getText().equals("")) {  charts.clear();  fi = 0;  sec = 0;  th = 0;  other = 0;   //array list to store the information from the file  ArrayList<info> arr = new ArrayList<info>();  //read the file as csv file  BufferedReader readFile = new BufferedReader(new FileReader(txtPathFile.getText()));  String readFilerow;  readFilerow = readFile.readLine();  while ((readFilerow = readFile.readLine()) != null) {  String[] data = readFilerow.split(",");  String temp = "";  if (data.length == 6) {  temp = data[0] + "," + data[1] + "," + data[2];  arr.add(new info(temp, Integer.parseInt(data[3]), Integer.parseInt(data[4]), Integer.parseInt(data[5])));  } else if (data.length == 5) {  temp = data[0] + "," + data[1];  arr.add(new info(temp, Integer.parseInt(data[2]), Integer.parseInt(data[3]), Integer.parseInt(data[4])));  }  }  //creat a Set for the projects chosen by all groups  Set<Integer> s = new HashSet<>();  for (int i = 0; i < arr.size(); i++) {  s.add(arr.get(i).getF());  s.add(arr.get(i).getS());  s.add(arr.get(i).getT());  }  //convert the set to array list  ArrayList<Integer> t = new ArrayList<>(s);  ArrayList<Integer> r = new ArrayList<>();  //creat list for all projects in the projects file  for (int i = 1; i < 39; i++)  r.add(i);  //creat an object of initial solution class and send the two lists to it  initialsol e = new initialsol(t, r, arr.size());  //creat array list for initial solutions  ArrayList<int[]> is = new ArrayList<>();  System.out.println(t.size() + " " + r.size());  //creat 10 initial solutions  for (int i = 0; i < 10; i++)  is.add(e.sol());  //creat an object from gentic algorithim class  geniticalgorithim p = new geniticalgorithim(is, arr);  //sort the initial solution based on solution fitness function  is = p.bubblesort();  //update the value for initial solution list in the genitic object  p.setArr1(is);   //define loop iteration variable which get the value from the text field in the interface  int loopiter = Integer.parseInt(txtNumLoop.getText().trim());  //start gentic algorithm loop  for (int i = 0; i < loopiter; i++) {   //creat object for random number  Random random = new Random();   //Array list to store the result for Crossover operation  ArrayList<int[]> crossresut = p.cross\_over();  //Add the result of cross over operation to the population  is.add(crossresut.get(0));  is.add(crossresut.get(1));  //update the initial solution list in genetic algorithm object  p.setArr1(is);  //generate random number between 0 and 10  int l = random.ints(0, 10).findFirst().getAsInt();  //if the random number is more than 8 make a mutation in the two children  if (l > 8)  p.mutation();   //sort the initial solution list based on fitness function  is = p.bubblesort();  //remove the lowest scored chromosomes from the population  is.remove(is.size() - 1);  is.remove(is.size() - 1);  //add the best chromosome for chart array list to use it in plot line for the algorithm performance  charts.add(new chart(i, p.fitness\_fun(is.get(0))));  }  //count the groups choices respect to order of choice  int[] f = is.get(0).clone();  for (int i = 0; i < arr.size(); i++) {   if (f[i] == arr.get(i).getF())  fi++;  else if (f[i] == arr.get(i).getS())  sec++;  else if (f[i] == arr.get(i).getT())  th++;  else  other++;  }    list.clear();  //for creating the table for comparison  for (int i = 0; i < arr.size(); i++) {  list.add(new tabelView(i, arr.get(i).getGroup(), arr.get(i).getF(), arr.get(i).getS(), arr.get(i).getT(), is.get(0)[i]));  }   number.setCellValueFactory(new PropertyValueFactory<tabelView, Integer>("number"));  groups.setCellValueFactory(new PropertyValueFactory<tabelView, String>("groups"));  first.setCellValueFactory(new PropertyValueFactory<tabelView, Integer>("first"));  second.setCellValueFactory(new PropertyValueFactory<tabelView, Integer>("second"));  third.setCellValueFactory(new PropertyValueFactory<tabelView, Integer>("third"));  alg.setCellValueFactory(new PropertyValueFactory<tabelView, Integer>("alg"));  tabel.setItems(list);   checkSubmit = true;    }//end  else if (txtPathFile.getText().equals("")) {    Alert alertCreat = new Alert(Alert.AlertType.WARNING);  alertCreat.setTitle("Warning");  alertCreat.setHeaderText(null);  alertCreat.setContentText("Please choose a file to read it!!");  alertCreat.showAndWait();    } else {    Alert alertCreat = new Alert(Alert.AlertType.WARNING);  alertCreat.setTitle("Warning");  alertCreat.setHeaderText(null);  alertCreat.setContentText("Pleas enter number of loop");  alertCreat.showAndWait();    }  } catch (IllegalArgumentException exception) {    Alert alertCreat = new Alert(Alert.AlertType.ERROR);  alertCreat.setTitle("Error");  alertCreat.setHeaderText(null);  alertCreat.setContentText(exception.getMessage());  alertCreat.showAndWait();    }    }    @FXML  void btnBrowseAction(ActionEvent event) {   FileChooser fileChooserShares = new FileChooser();  fileChooserShares.setTitle("Select project file ");  fileChooserShares.getExtensionFilters().addAll(  new FileChooser.ExtensionFilter("Text Files", "\*.csv")  );   File selectedFile = fileChooserShares.showOpenDialog(null);  if (String.valueOf(selectedFile).equals("null")) {  return;  } else {  txtPathFile.setText(selectedFile.toString());  }   }   } |
| --- |

## Table class :

This class is to create table for the result in the user interface .

### Code :

| package sample; public class tabelView {  private int number;  private String groups;  private int first;  private int second;  private int third;  private int alg;   public tabelView(int number, String groups, int first, int second, int third, int alg) {  this.number = number;  this.groups = groups;  this.first = first;  this.second = second;  this.third = third;  this.alg = alg;  }   public int getNumber() {  return number;  }   public void setNumber(int number) {  this.number = number;  }   public String getGroups() {  return groups;  }   public void setGroups(String groups) {  this.groups = groups;  }   public int getFirst() {  return first;  }   public void setFirst(int first) {  this.first = first;  }   public int getSecond() {  return second;  }   public void setSecond(int second) {  this.second = second;  }   public int getThird() {  return third;  }   public void setThird(int third) {  this.third = third;  }   public int getAlg() {  return alg;  }   public void setAlg(int alg) {  this.alg = alg;  }   @Override  public String toString() {  return "tabelView{" +  "groupNumber=" + number +  ", groups='" + groups + '\'' +  ", firstChoice=" + first +  ", secondChoice=" + second +  ", thirdChoice=" + third +  ", algChoice=" + alg +  '}';  }  } |
| --- |

## Chart class :

This class is used to create objects from it and store the needed data to plot the chart for the genetic algorithm performance .

### Code :

| package sample;  public class chart {  private int x ;  private int y ;   public chart(int x, int y) {  this.x = x;  this.y = y;  }   public int getX() {  return x;  }     public void setX(int x) {  this.x = x;  }   public int getY() {  return y;  }   public void setY(int y) {  this.y = y;  }  @Override  public String toString() {  return "chart{" +  "x=" + x +  ", y=" + y +  '}';  } } |
| --- |

## Information class :

This class is to create objects from it to store the data from the Groups file.

### Code :

| package sample;  public class info {  private String group ;  private int f ;  private int s ;  private int t ;  public info (String group , int f , int s , int t){  this.group=group ;  this.f=f ;  this.s=s;  this.t=t ;  }   public String getGroup() {  return group;  }   public void setGroup(String group) {  this.group = group;  }   public int getF() {  return f;  }   public void setF(int f) {  this.f = f;  }   public int getS() {  return s;  }   public void setS(int s) {  this.s = s;  }   public int getT() {  return t;  }   public void setT(int t) {  this.t = t;  }   @Override  public String toString() {  return group + "," + f + "," + s +"," + t ;  } } |
| --- |

## 

## 

## 

## The controller chart class :

This class is to run the code for the chart in user interface .

### Code :

| package sample;  import javafx.event.ActionEvent; import javafx.fxml.FXML; import javafx.fxml.FXMLLoader; import javafx.fxml.Initializable; import javafx.geometry.Insets; import javafx.scene.Parent; import javafx.scene.Scene; import javafx.scene.chart.LineChart; import javafx.scene.chart.NumberAxis; import javafx.scene.chart.XYChart; import javafx.scene.control.Button; import javafx.scene.control.Label; import javafx.scene.layout.AnchorPane; import java.io.IOException; import java.net.URL; import java.util.ArrayList; import java.util.ResourceBundle; public class controlerchart implements Initializable {     @FXML  private AnchorPane chartplot;   @FXML  private Label label;   @FXML  private LineChart<?, ?> ch;   @FXML  private Label first;   @FXML  private Label iteration;     @FXML  private Label other;   @FXML  private Label second;   @FXML  private Label third;   @Override  public void initialize(URL url, ResourceBundle resourceBundle) {  XYChart.Series series = new XYChart.Series();  series.setName("Score of the chromosome");  ArrayList<chart>c = Controller.charts ;  first.setText("First : "+Integer.toString(Controller.fi));   second.setText("Second : "+Integer.toString(Controller.sec));  third.setText("Third : " +Integer.toString(Controller.th));  other.setText("Other : " +Integer.toString(Controller.other));   for (int i = 0 ; i < c.size() ; i++)  series.getData().add(new XYChart.Data(Integer.toString(c.get(i).getX()) ,c.get(i).getY())) ;    ch.getData().add(series) ;    } } |
| --- |

## FXML files :

This files for user interface .

### 1-FXML file for the main user interface :

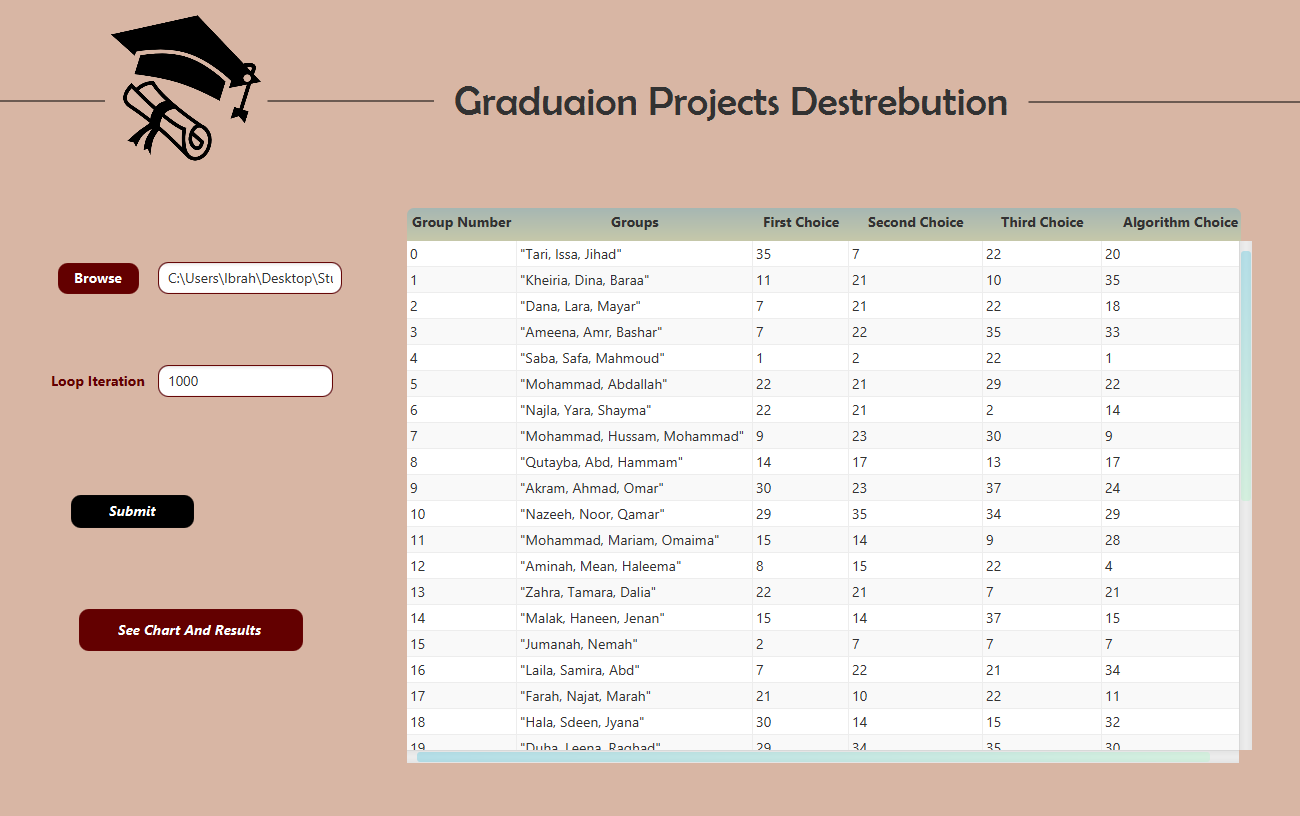
| <?xml version="1.0" encoding="UTF-8"?> <?import javafx.scene.control.Button?> <?import javafx.scene.control.Label?> <?import javafx.scene.control.TableColumn?> <?import javafx.scene.control.TableView?> <?import javafx.scene.control.TextField?> <?import javafx.scene.image.Image?> <?import javafx.scene.image.ImageView?> <?import javafx.scene.layout.AnchorPane?> <?import javafx.scene.shape.Line?> <?import javafx.scene.text.Font?>  <AnchorPane cache="true" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="824.0" prefWidth="1305.0" style="-fx-background-color: #D8B6A4;" xmlns="http://javafx.com/javafx/17" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Controller">  <children>  <TextField fx:id="txtPathFile" layoutX="161.0" layoutY="263.0" prefHeight="30.0" prefWidth="184.0" promptText="Select Path for the file" style="-fx-background-radius: 10px; -fx-border-color: #630000; -fx-border-radius: 10px;" />  <TextField fx:id="txtNumLoop" layoutX="161.0" layoutY="366.0" promptText="Enter Number of loop" style="-fx-background-radius: 10px; -fx-border-color: #630000; -fx-border-radius: 10px;" />  <Label layoutX="54.0" layoutY="372.0" style="-fx-background-radius: 10px;" text="Loop Iteration" textFill="#630000">  <font>  <Font name="System Bold" size="14.0" />  </font></Label>  <Label layoutX="457.0" layoutY="72.0" prefHeight="59.0" prefWidth="565.0" text="Graduaion Projects Destrebution">  <font>  <Font name="Berlin Sans FB" size="41.0" />  </font>  </Label>  <Button fx:id="btnSubmit" layoutX="74.0" layoutY="496.0" mnemonicParsing="false" onAction="#btnSubmitAction" prefHeight="32.0" prefWidth="123.0" style="-fx-background-color: #000000; -fx-background-radius: 10px;" text="Submit" textFill="WHITE">  <font>  <Font name="System Bold Italic" size="14.0" />  </font></Button>  <Button fx:id="btnBrowse" layoutX="61.0" layoutY="264.0" mnemonicParsing="false" onAction="#btnBrowseAction" prefHeight="30.0" prefWidth="81.0" style="-fx-background-color: #630000; -fx-background-radius: 10px;" text="Browse" textFill="WHITE">  <font>  <Font name="System Bold" size="14.0" />  </font></Button>  <TableView fx:id="tabel" layoutX="409.0" layoutY="208.0" prefHeight="557.0" prefWidth="847.0" stylesheets="@style.css">  <columns>  <TableColumn fx:id="number" prefWidth="110.0" text="Group Number" />  <TableColumn fx:id="groups" prefWidth="236.0" text="Groups" />  <TableColumn fx:id="first" prefWidth="96.0" text="First Choice" />  <TableColumn fx:id="second" prefWidth="134.0" text="Second Choice" />  <TableColumn fx:id="third" prefWidth="119.0" text="Third Choice" />  <TableColumn fx:id="alg" prefWidth="157.0" text="Algorithm Choice" />  </columns>  </TableView>  <ImageView fitHeight="150.0" fitWidth="200.0" layoutX="114.0" layoutY="14.0" pickOnBounds="true" preserveRatio="true">  <image>  <Image url="@graduation-hat-and-diploma.png" />  </image>  </ImageView>  <Line endX="-7.5" layoutX="115.0" layoutY="102.0" startX="-115.0" />  <Line endX="65.5" layoutX="371.0" layoutY="102.0" startX="-100.0" />  <Button layoutX="82.0" layoutY="610.0" mnemonicParsing="false" onAction="#showchart" prefHeight="41.0" prefWidth="224.0" style="-fx-background-color: #630000; -fx-background-radius: 10px;" text="See Chart And Results " textFill="WHITE">  <font>  <Font name="System Bold Italic" size="14.0" />  </font></Button>  <Line endX="173.0" layoutX="1132.0" layoutY="103.0" startX="-100.0" />  </children> </AnchorPane> |
| --- |

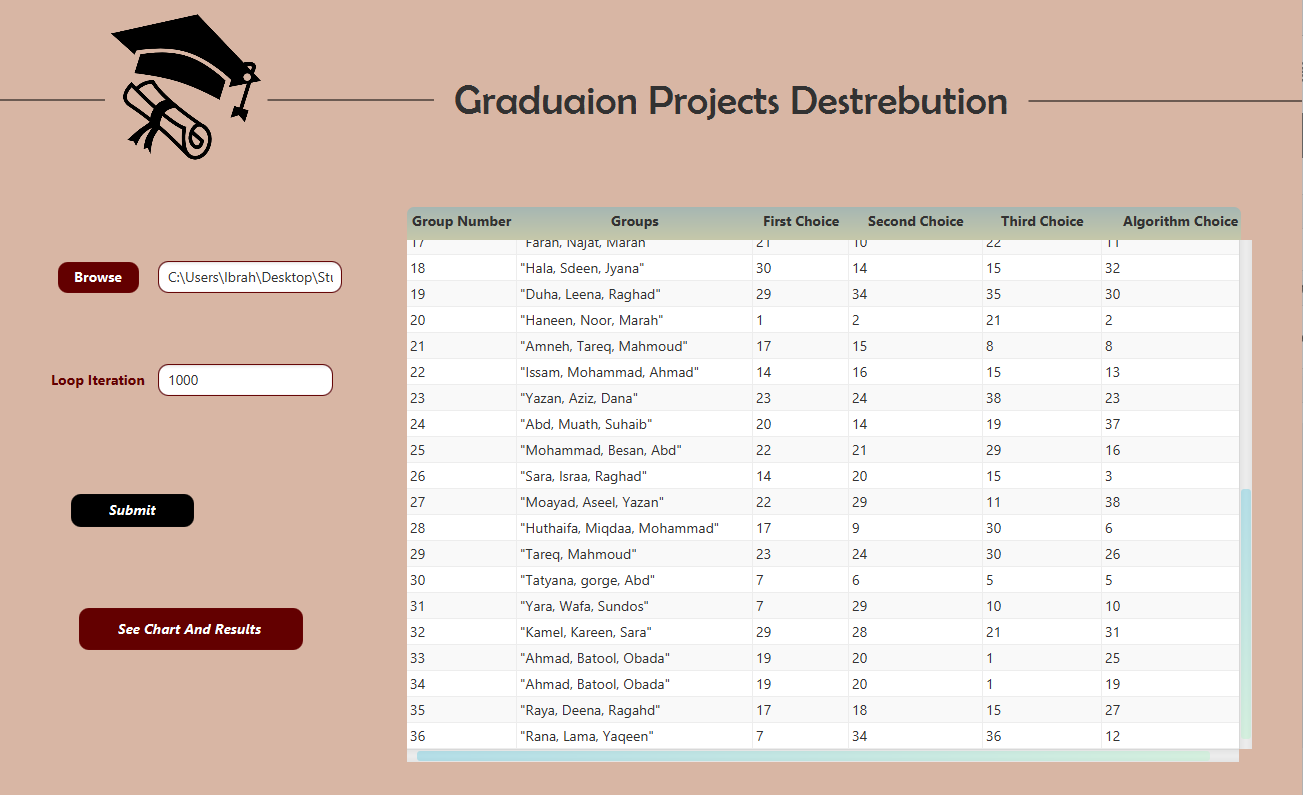
### 2- FXML file for the chart interface :

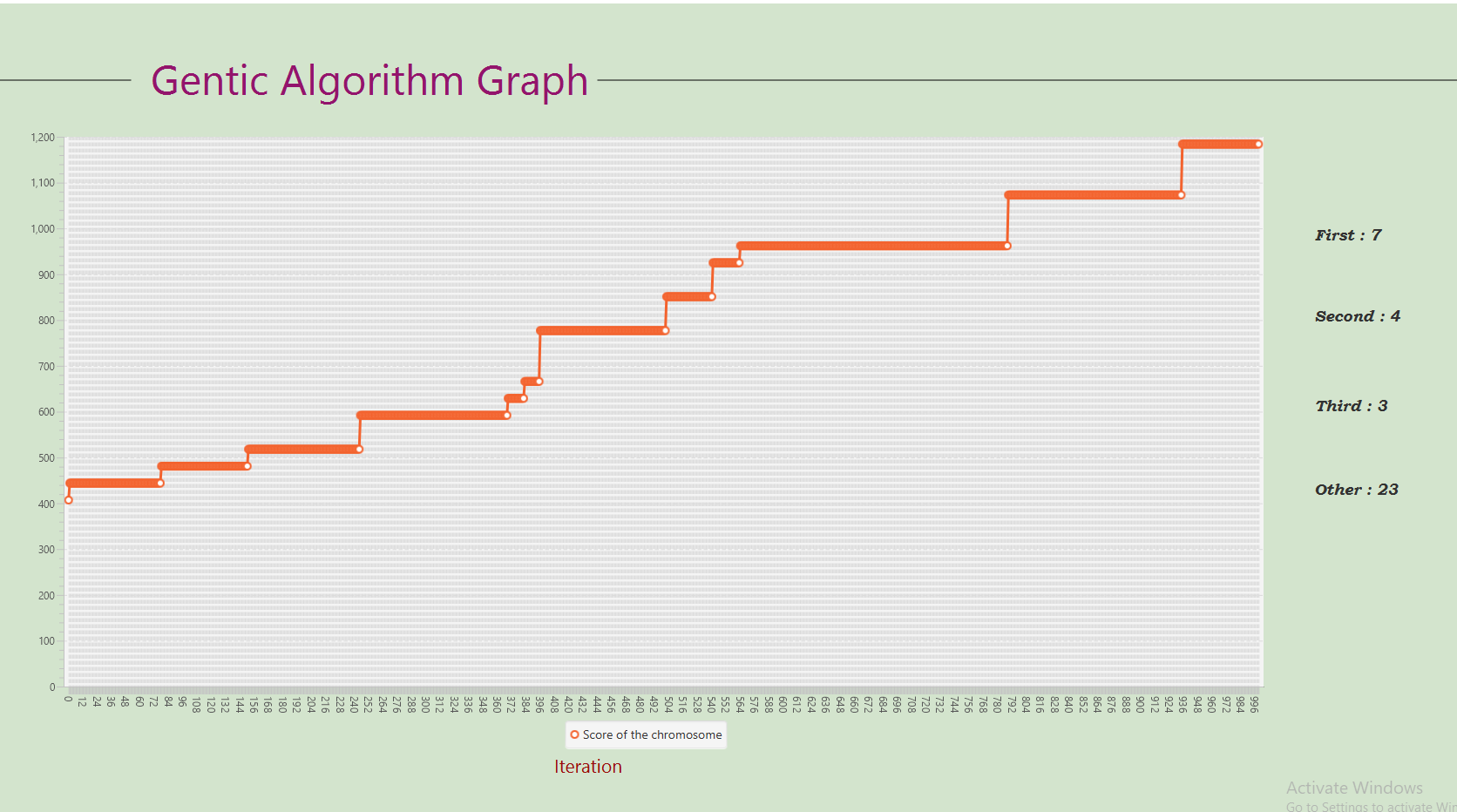
| <?xml version="1.0" encoding="UTF-8"?> <?import javafx.scene.chart.CategoryAxis?> <?import javafx.scene.chart.LineChart?> <?import javafx.scene.chart.NumberAxis?> <?import javafx.scene.control.Label?> <?import javafx.scene.layout.AnchorPane?> <?import javafx.scene.shape.Line?> <?import javafx.scene.text.Font?>  <AnchorPane fx:id="chartplot" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="936.0" prefWidth="1681.0" style="-fx-background-color: #D3E4CD;" xmlns="http://javafx.com/javafx/17" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.controlerchart">  <children>  <Label fx:id="label" layoutX="176.0" layoutY="35.0" prefHeight="105.0" prefWidth="533.0" text="Gentic Algorithm Graph" textFill="#94146d">  <font>  <Font size="48.0" />  </font>  </Label>  <LineChart fx:id="ch" layoutX="22.0" layoutY="138.0" prefHeight="722.0" prefWidth="1445.0">  <xAxis>  <CategoryAxis side="BOTTOM" />  </xAxis>  <yAxis>  <NumberAxis side="LEFT" />  </yAxis>  </LineChart>  <Label fx:id="iteration" layoutX="639.0" layoutY="860.0" prefHeight="20.0" prefWidth="105.0" text="Iteration" textFill="#9a0b0b">  <font>  <Font size="21.0" />  </font>  </Label>  <Label fx:id="first" layoutX="1511.0" layoutY="252.0" prefHeight="27.0" prefWidth="126.0" style="-fx-background-radius: 10px;" text="Label">  <font>  <Font name="Bookman Old Style Bold Italic" size="18.0" />  </font>  </Label>  <Label fx:id="second" layoutX="1511.0" layoutY="343.0" prefHeight="31.0" prefWidth="119.0" style="-fx-background-radius: 10px;" text="Label">  <font>  <Font name="Bookman Old Style Bold Italic" size="18.0" />  </font>  </Label>  <Label fx:id="third" layoutX="1511.0" layoutY="448.0" prefHeight="27.0" prefWidth="119.0" text="Label">  <font>  <Font name="Bookman Old Style Bold Italic" size="18.0" />  </font>  </Label>  <Label fx:id="other" layoutX="1511.0" layoutY="544.0" prefHeight="27.0" prefWidth="105.0" text="Label">  <font>  <Font name="Bookman Old Style Bold Italic" size="18.0" />  </font>  </Label>  <Line endX="686.0" layoutX="995.0" layoutY="88.0" startX="-306.0" />  <Line endX="-78.0" layoutX="231.0" layoutY="88.0" startX="-240.5" />  </children> </AnchorPane> |
| --- |

## Sample tests :

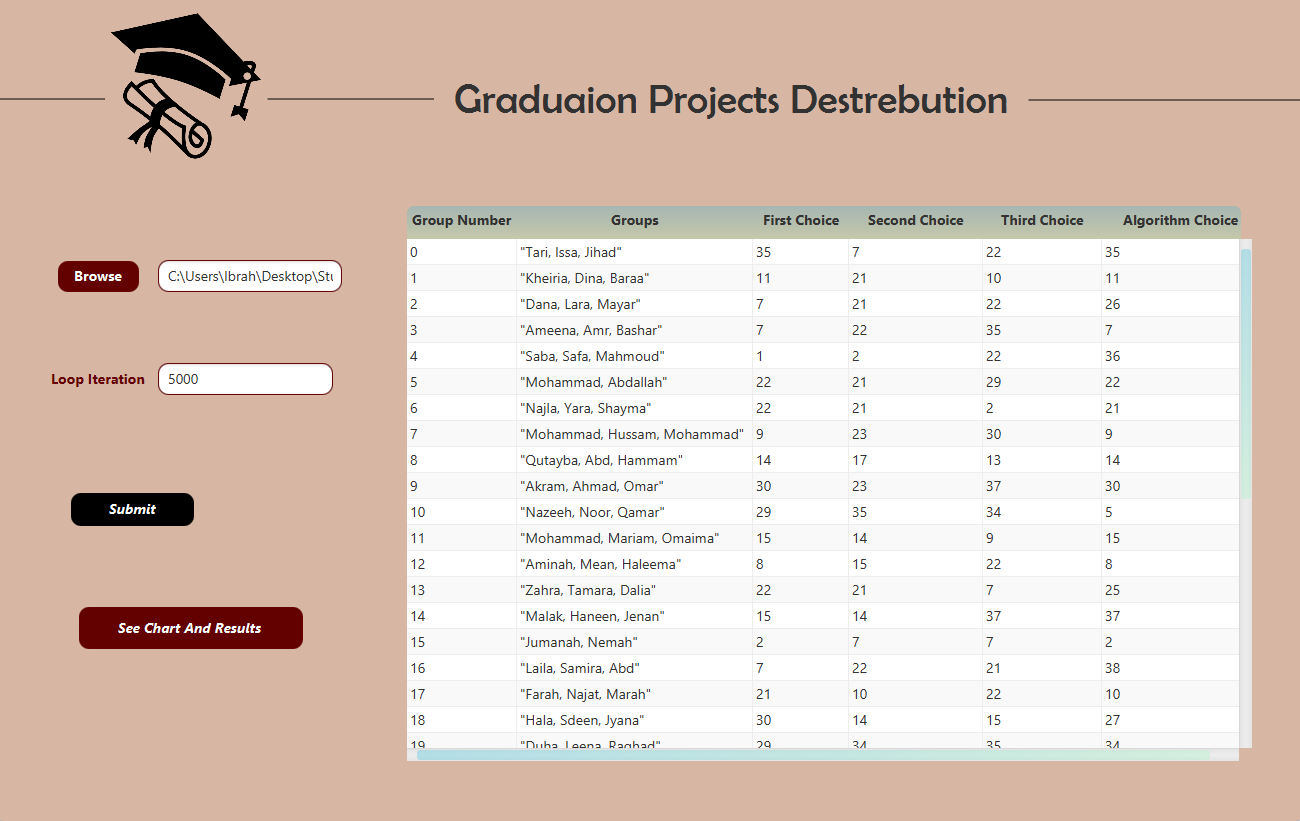
### Run the program with 1000 iteration for the genetic loop :

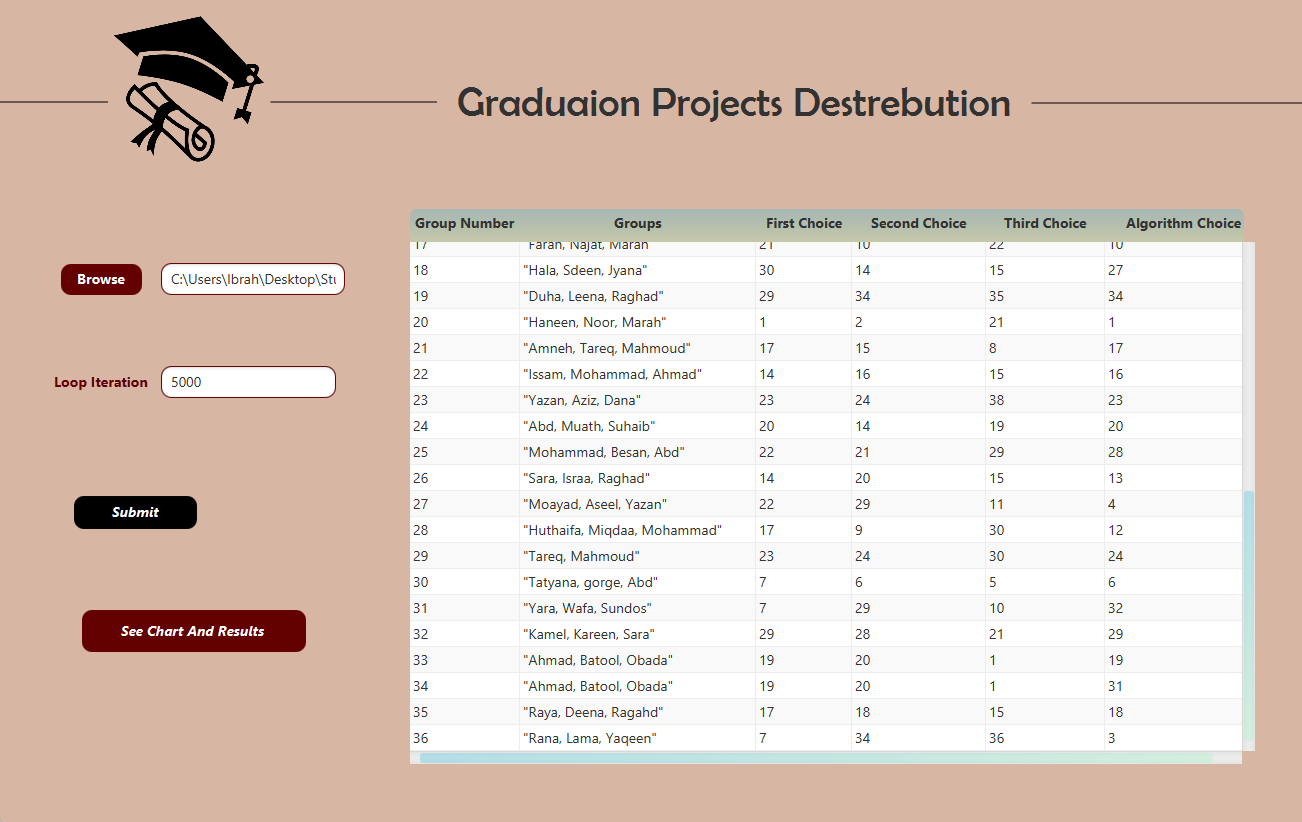


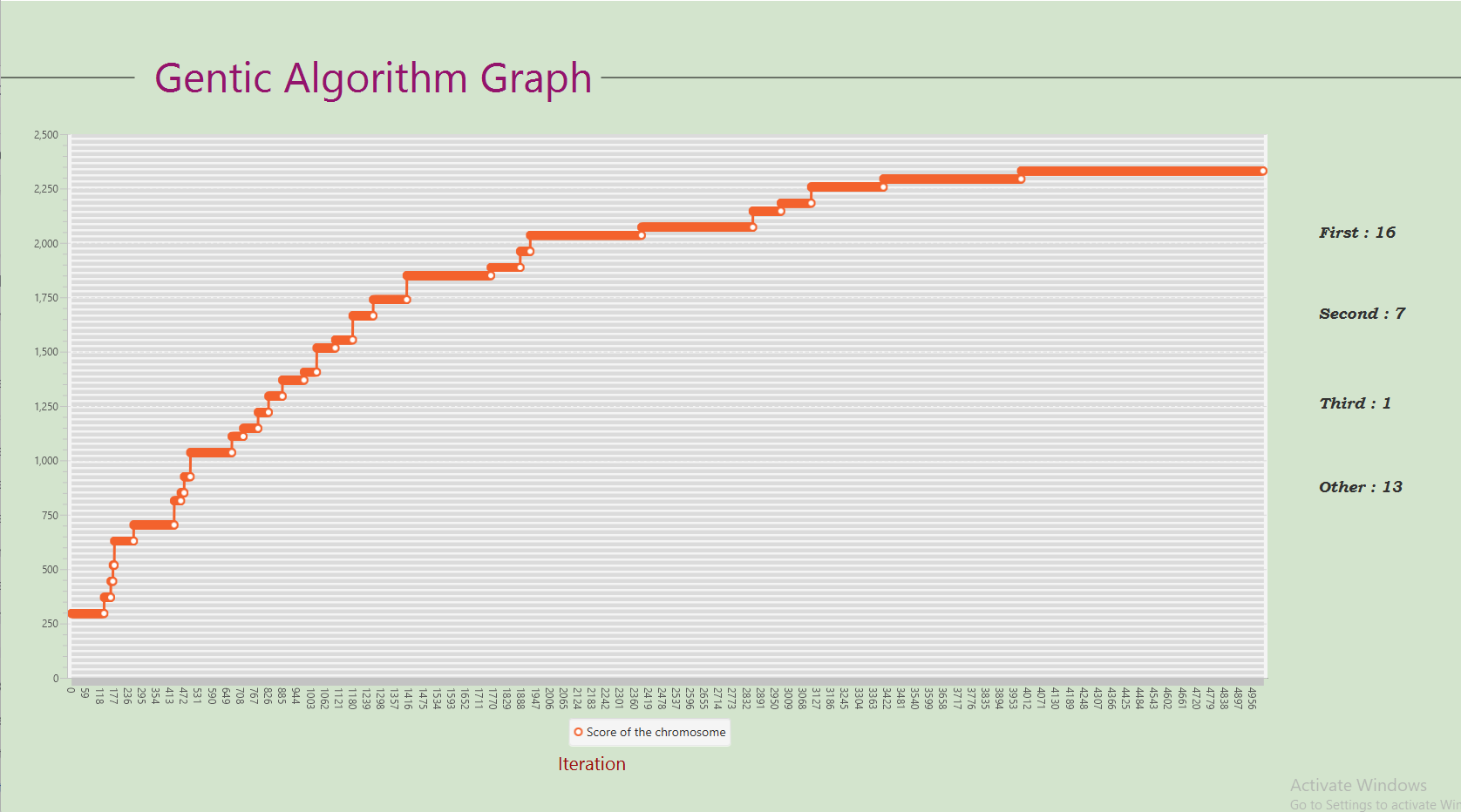




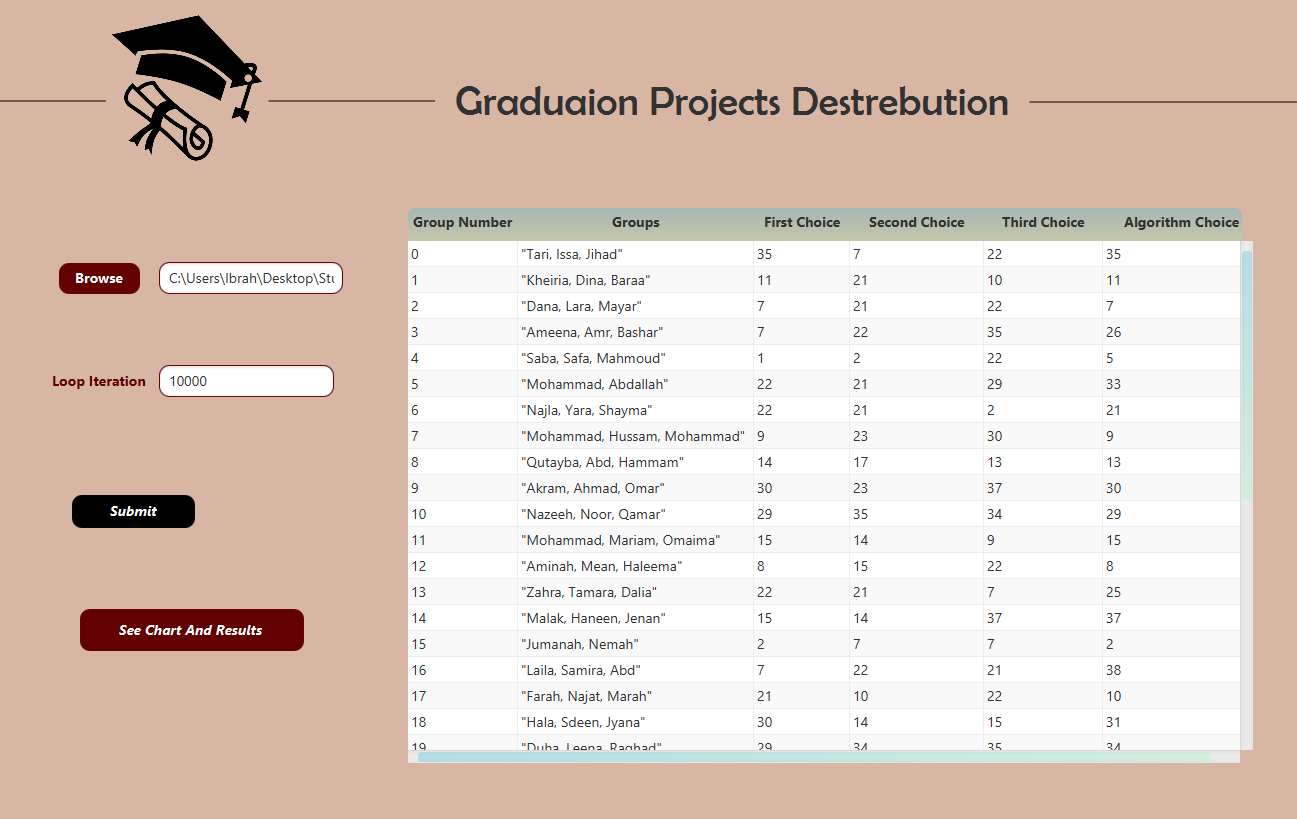
### Run the program with 5000 iteration for the genetic loop :

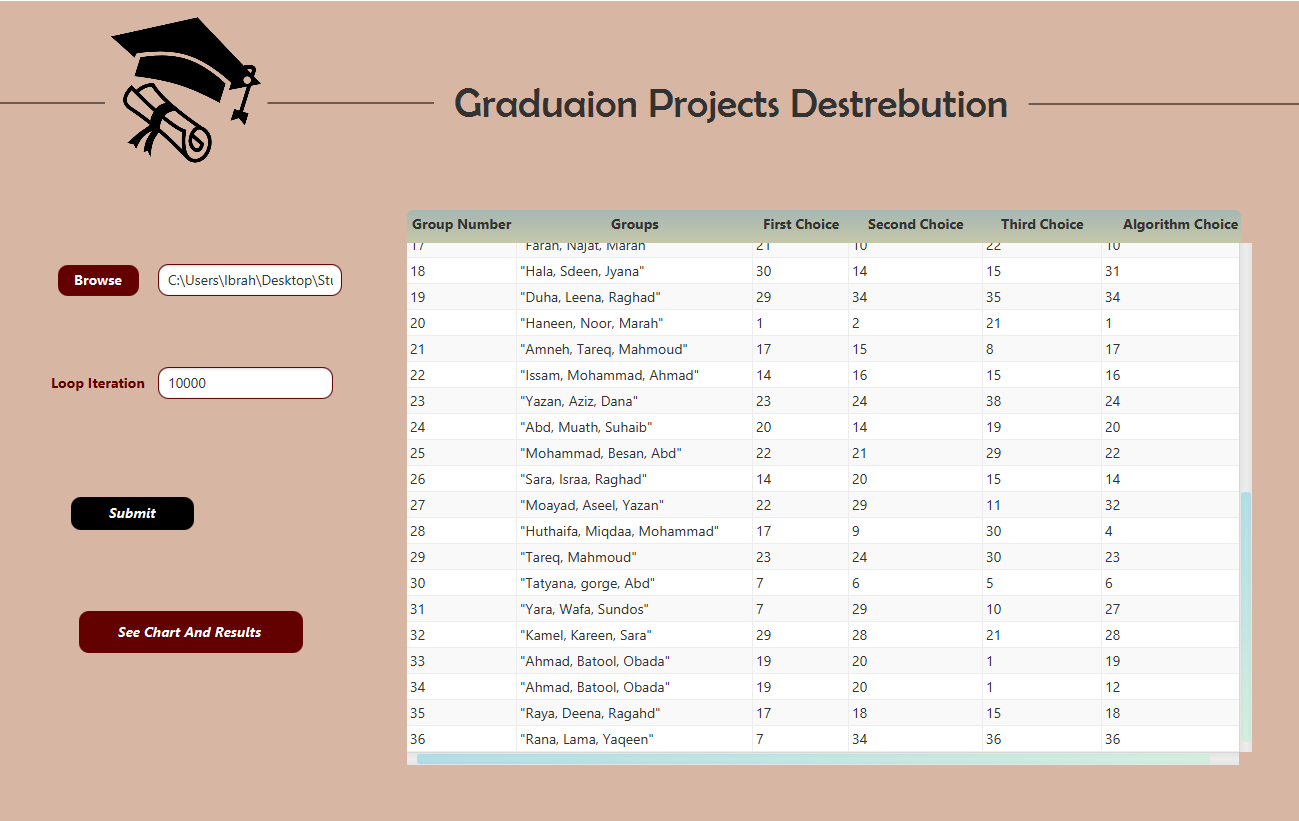


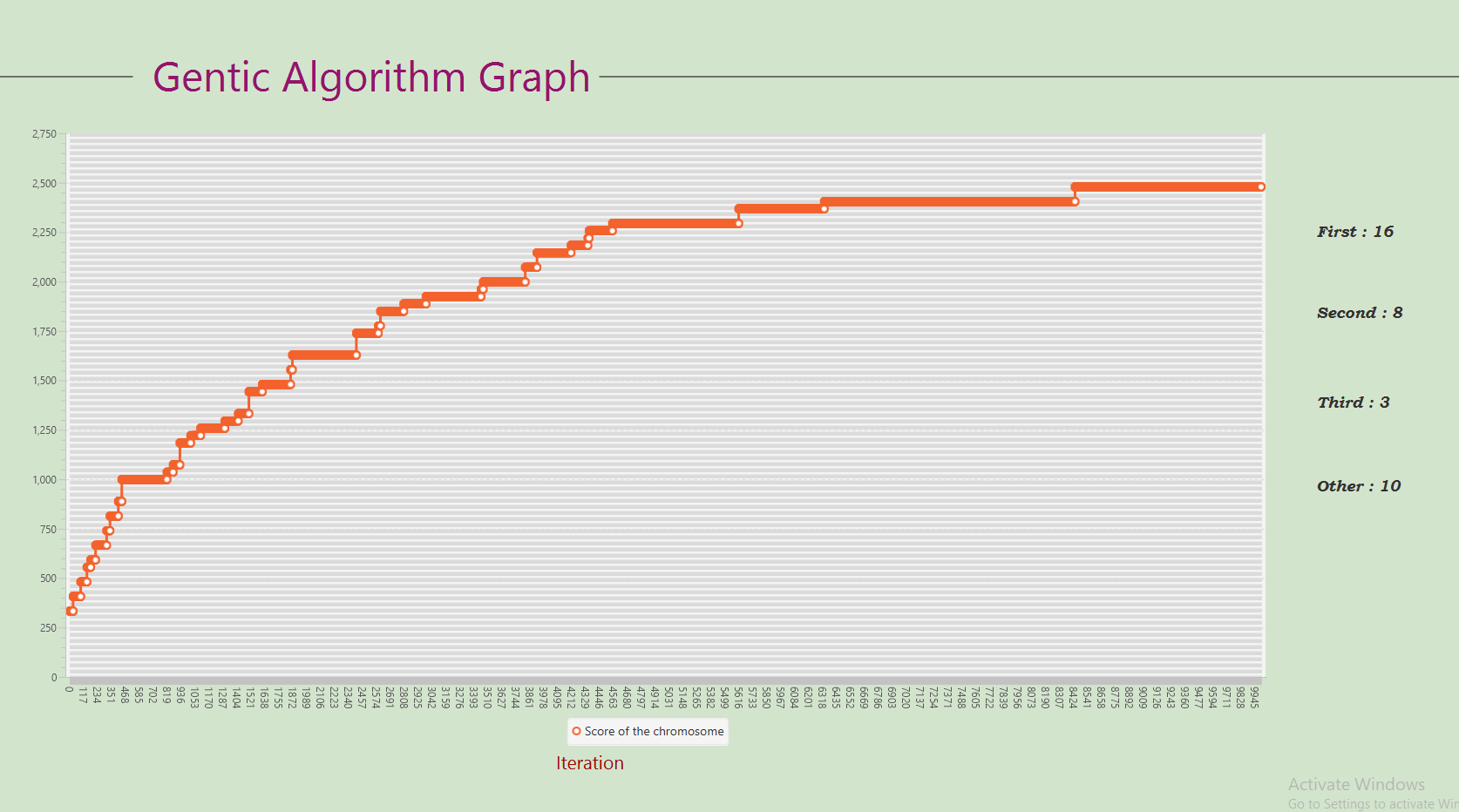




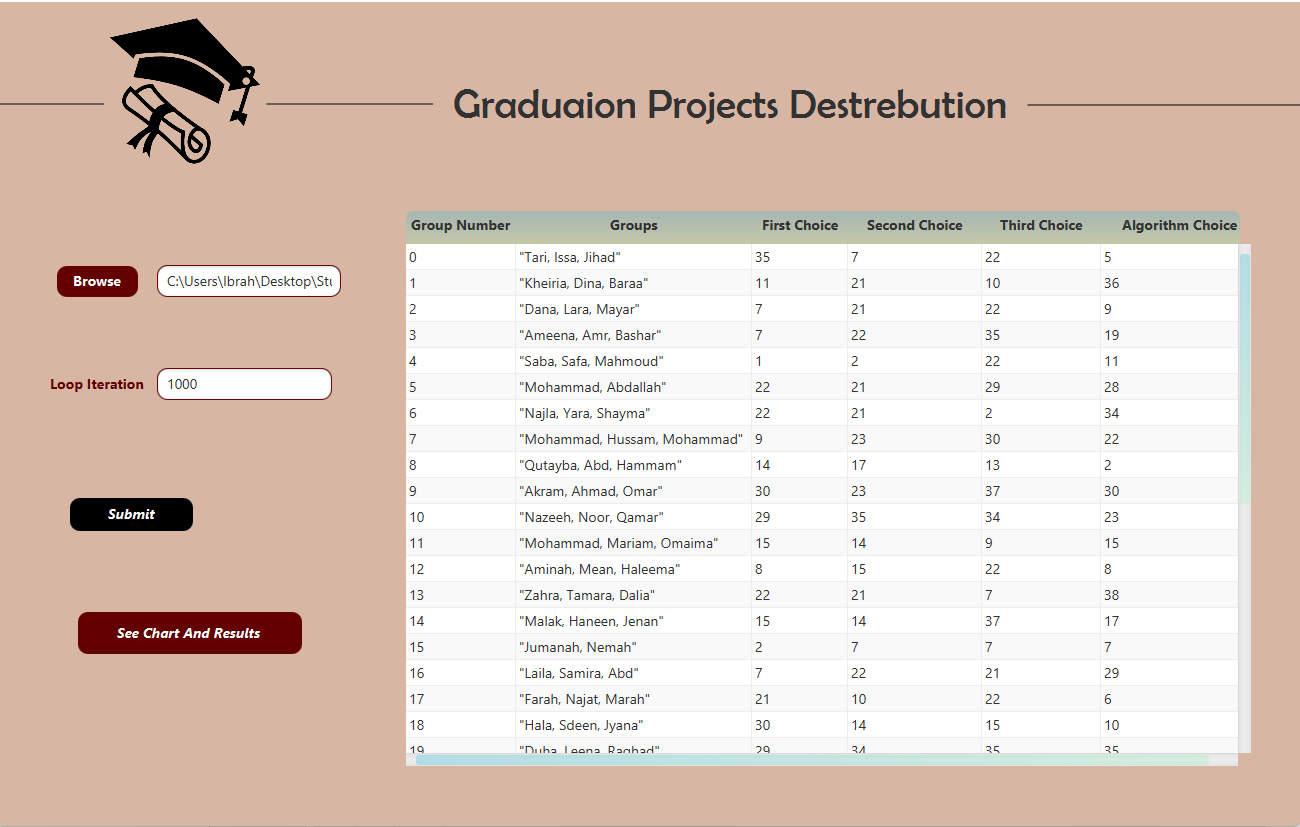
### Run the program with 10000 iteration for the genetic loop :

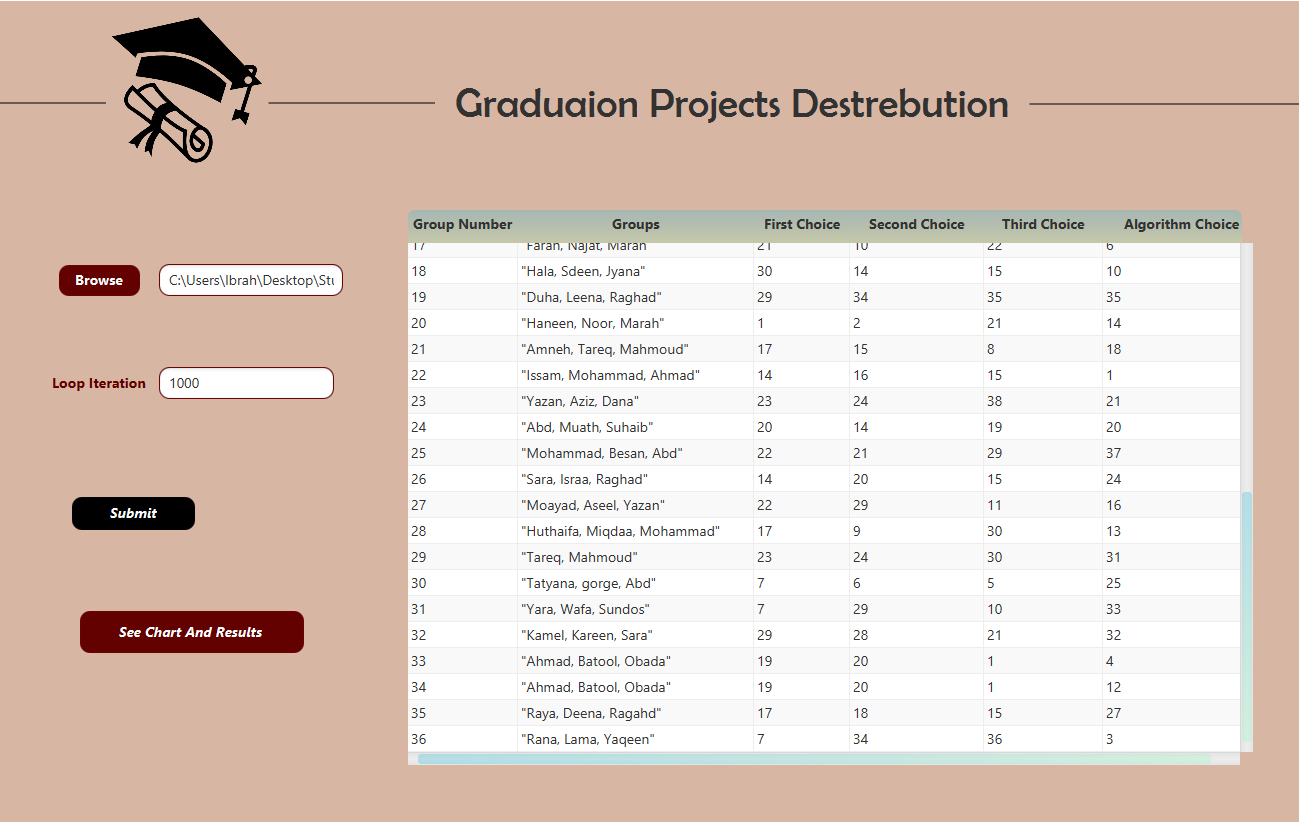


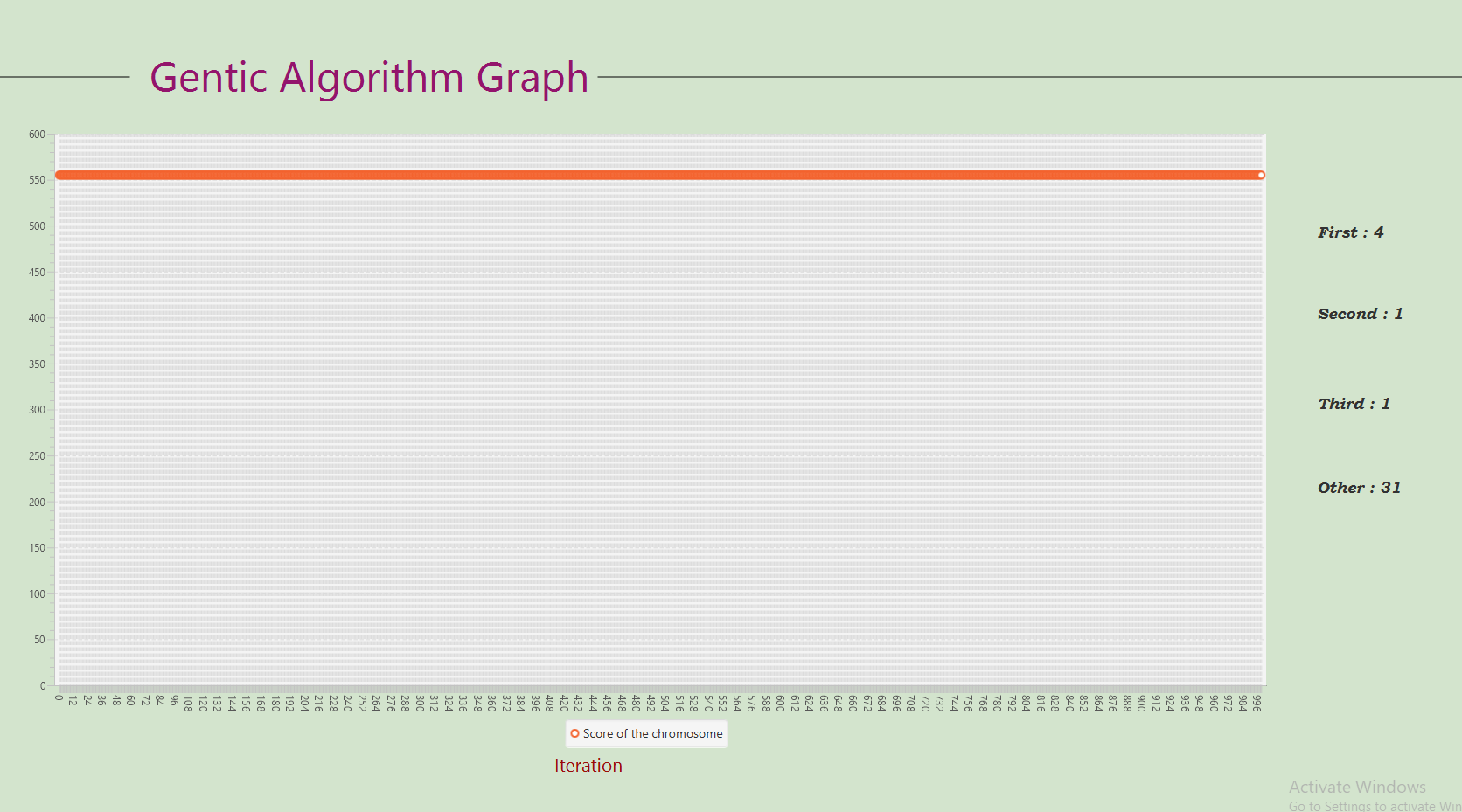




Run without mutation to see the importance of it :







We note that the algorithm is stuck and can not optimize the solution .