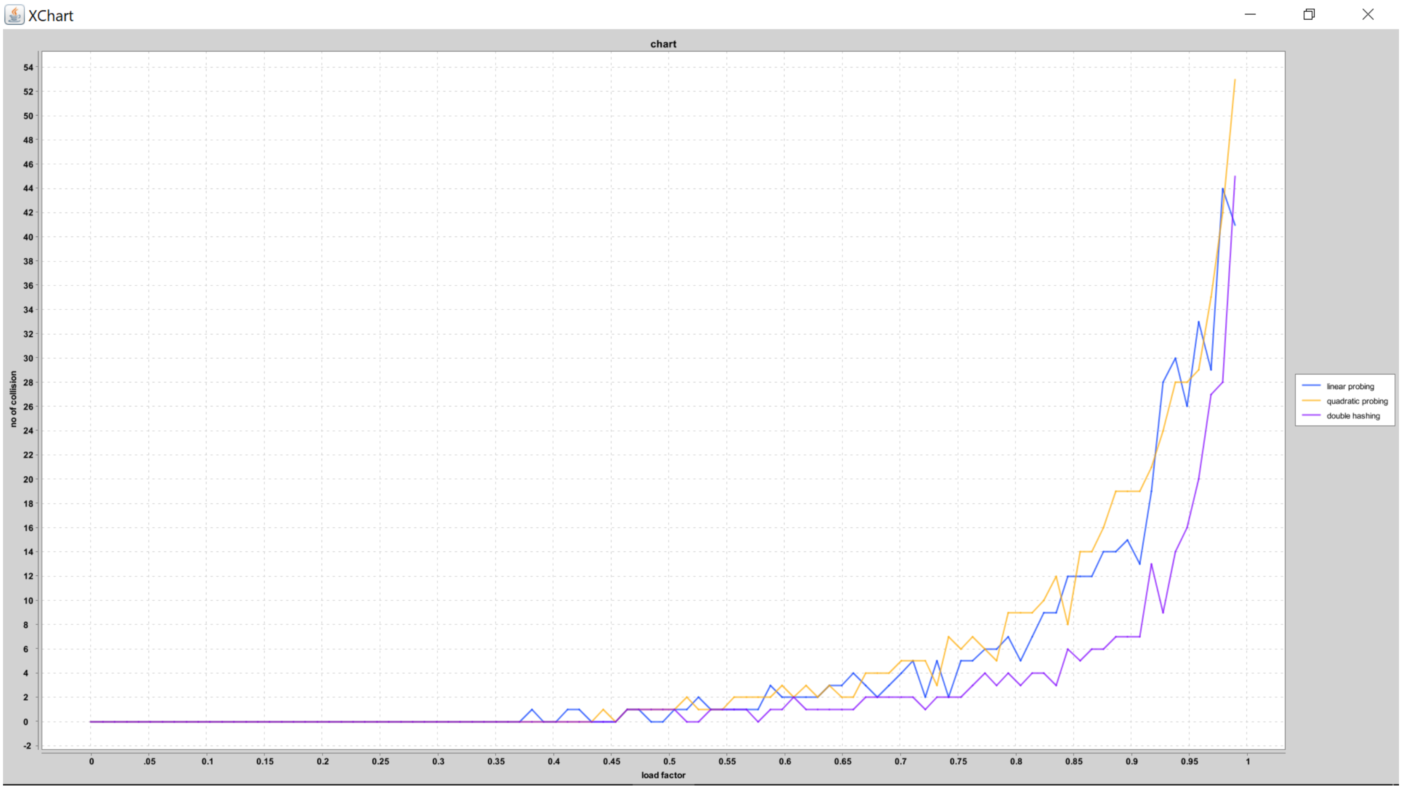
```
package adsprac2;

import static java.lang.Math.random;
import static java.lang.StrictMath.random;
import static java.lang.System.load;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Random;
import java.util.stream.IntStream;
//import javafx.scene.chart.XYChart;
import javax.swing.JPanel;
import org.knowm.xchart.QuickChart;
import org.knowm.xchart.SwingWrapper;
//import org.knowm.xchart.SwingWrapper;
import org.knowm.xchart.XChartPanel;
import org.knowm.xchart.XYChart;


public class hashing {

   public int i,b,collision,loadfac;
   int [] a;
    {
      random();
      linear();
      quadratic();
     doublehashing();
}

public void random(){
    a = new int[100];
    Random rand = new Random();
    for (int i = 0; i < 100; i++) {
      a[i] = rand.nextInt(110);
    }
    System.out.println(Arrays.toString(a));
  }
public void linear(){
        System.out.println("-------------------LINEAR PROBING-----------------");
        double load;
        int c = 0;
        ArrayList<Integer> list = new ArrayList<>();
    for(i=0;i<100;i++){
    b=a[i]%15;
    load=(i+1)/100.0;
        System.out.println("load factor is:"+load);

    for(int j=0;j<100;j++){
    boolean ans = list.contains(b);
    if(ans){
    collision++;
    c++;
    b=(b+1)%100;
    }
    else
    {

    list.add(b);
    break;
    }
     }
      System.out.println("no of collisions are:"+collision);
        collision=0;
```

```
}
        System.out.println( "Total no of collisions in linear probing is:"+c);
    }

     public void quadratic(){
         System.out.println("------------------QUADRATIC PROBING------------------");
         int collision1 = 0,c=0;
         double load;
         ArrayList<Integer> list = new ArrayList<>();
    for(i=0;i<100;i++){
    b=a[i]%15;
    load=(i+1)/100.0;
        System.out.println("Load factor is:"+load);
    int t=1;
     for(int j=0;j<100;j++){

    boolean ans = list.contains(b);
    if(ans){
    collision1++;
    c++;
      //  System.out.println(collision1);
    b=((a[i]%15)+(t*t))%100;
    t++;
    }
    else
    {
    list.add(b);
     break;
    }
     }
   System.out.println("no of collisions are:"+collision1);
        collision1=0;
    }
    System.out.println( "Total no of collisions in quadratic probing is:"+c);
     }

public void doublehashing(){
          System.out.println("---------DOUBLE HASHING---------");
        ArrayList<Integer> list = new ArrayList<>();
        int collision3=0,d=0,c=0;
    double load;
    for(i=0;i<100;i++){
    b=a[i]%15;
    load=(i+1)/100.0;
    System.out.println("Load factor is:"+load);
    c=(a[i]%(100-2))+1;

     for(int j=0;j<100;j++){
    boolean ans = list.contains(b);
    if(ans){
    collision3++;
    d++;
    b=(b+c)%100;

    }
    else
    {
    list.add(b);
       break;
    }
     }

    System.out.println("no of collisions are:"+collision3);
```

```
        collision3=0;
}
        System.out.println( "Total no of collisions in double hashing is:"+d);
    }
    public static void main(String[] args) {
        hashing j=new hashing();
        XYChart chart = QuickChart.getChart("collision Chart", "load factor", "no of collision", "y(x)", xdata, avg);
        new SwingWrapper(chart).displayChart();
    }
 }
```

OUTPUT:


XChart
chart
no of collision
54
52
50
48
46
44
42
40
38
36
34
32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
-2
0
.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
0.55
0.6
0.65
0.7
0.75
0.8
0.85
0.9
0.95
1
load factor
linear probing
quadratic probing
double hashing