МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»

(САМАРСКИЙ УНИВЕРСИТЕТ)

Отчёт по лабораторной работе №1  
по курсу «Моделирование информационно-вычислительных систем»

Выполнил:  
Никитин А. А.

гр.6303

Проверила:  
Симонова Е .В.

Самара 2016

**ЗАДАНИЕ НА МОДЕЛИРОВАНИЕ**

***Вариант 23***

Получить N = 1000 реализаций случайной величины, распределенной по усеченно-нормальному закону с параметрами (m,σ2) и принимающей значения > 0.

Параметры m и σ выбрать самостоятельно. Статистический контроль провести с использованием 15 интервалов разбиения, α = 0.005.

**АЛГОРИТМ МОДЕЛИРОВАНИЯ**

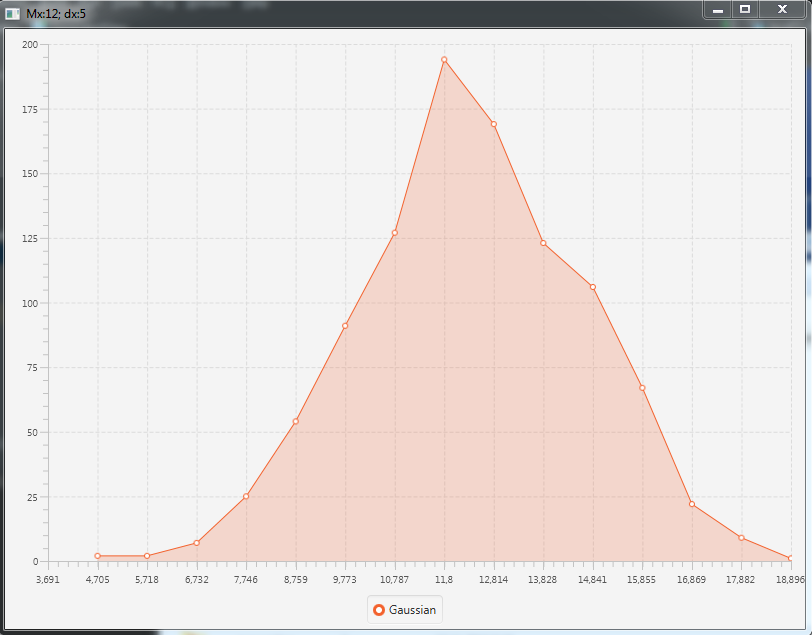
Расчет нормальной случайной величины:

public class RandomGenerator {  
 Random rand;  
  
 public RandomGenerator()  
 {  
 rand = new Random();  
 }  
 public double generate(double m, double d)  
 {  
 double result = 0;  
 for (int i = 0; i < 12; i++)  
 {  
 result += rand.nextDouble();  
 }  
  
 double res = (result - 6) \* Math.*sqrt*(d) + m;  
 if (res>=0){  
 return res;  
 }else {  
 return generate(m, d);  
 }  
 }  
}

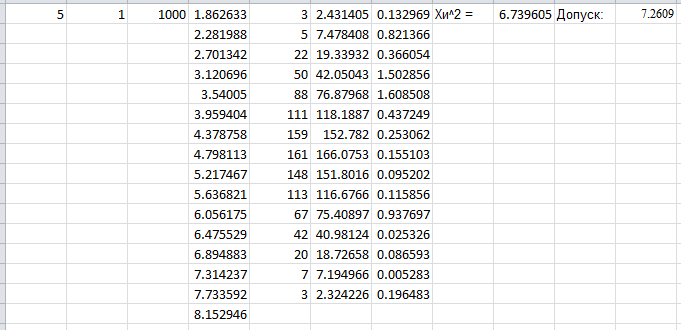
Разбиение на интервалы, анализ и вывод:

private void process(ActionEvent event) throws Exception {  
 new Thread(() -> {  
 int N = 1000;  
 double[] arr = new double[N];  
 RandomGenerator generator = new RandomGenerator();  
 for (int i = 0; i < arr.length; i++)  
 {  
 arr[i] = generator.generate(Double.*parseDouble*(mx.getText()), Double.*parseDouble*(dx.getText()));  
 }  
 Arrays.*sort*(arr);  
  
 double min = arr[0];  
 double max = arr[arr.length - 1];  
 double range = max - min;  
  
 int intervalCount = 15;//(int)(2 + 3.22 \* Math.Log10(arr.Length));  
 double dInterval = range / intervalCount;  
 int[] intervals = new int[intervalCount];  
 double[] intervalsX = new double[intervalCount];  
  
 for (int i = 0; i < arr.length; i++)  
 {  
 for (int j = 0; j < intervals.length; j++)  
 {  
 if (arr[i] >= min + dInterval \* (j) && arr[i] < min + dInterval \* (j + 1))  
 {  
 intervals[j] += 1;  
 }  
 }  
 }  
  
 for (int i = 0; i < intervalCount; i++)  
 {  
 intervalsX[i] = min + dInterval\*(i+1);  
 }  
  
 final NumberAxis xAxis = new NumberAxis(min, max, dInterval);  
 final NumberAxis yAxis = new NumberAxis();  
 final AreaChart<Number,Number> chart =  
 new AreaChart<Number,Number>(xAxis,yAxis);  
  
 XYChart.Series gaussian = new XYChart.Series();  
 gaussian.setName("Gaussian");  
  
 for (int i = 0; i < intervalCount; i++) {  
 gaussian.getData().add(new XYChart.Data(intervalsX[i], intervals[i]));  
 }  
  
 Platform.*runLater*(()->{  
 chart.getData().addAll(gaussian);  
 Stage stage = new Stage();  
 stage.setTitle("Mx:" + mx.getText() + "; dx:" + dx.getText());  
 Scene scene = new Scene(chart,800,600);  
 stage.setScene(scene);  
 stage.show();  
 });  
  
 double[] intervalBounds = new double[intervalCount+1];  
  
 for (int i = 0; i < intervalBounds.length; i++) {  
 intervalBounds[i] = min + dInterval \* i;  
 }  
  
 try {  
 ExcelService.*putValues*(  
 "pirson.xls",  
 Double.*valueOf*(mx.getText()),  
 Double.*valueOf*(dx.getText()),  
 N,  
 intervalBounds,  
 intervals);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }).start();  
}

**РЕЗУЛЬТАТ МОДЕЛИРОВАНИЯ**



**Вывод данных в Excel**

****

**ВЫВОД О КАЧЕСТВЕ МОДЕЛИ**

На основании сравнения с порогом *y0*,можно сделать вывод о том, что проверяемая гипотеза не может быть отвергнута.