|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

**ОТЧЕТ**

**по лабораторной работе № 1**

**по курсу: «Моделирование»**

|  |  |
| --- | --- |
| **Тема Изучение функций распределения и функций плотности распределения случайных чисел**  **Вариант 20 (4)**  **Студент Якуба Д. В.**  **Группа ИУ7-73Б**  **Оценка (баллы)**  **Преподаватель Рудаков И.В.** |  |

Москва, 2021

1. Задание

Для равномерного и распределения и распределения Гаусса построить графики функций распределения и плотности.

Разработать интерфейс для взаимодействия с программным обеспечением.

2. Выполнение

На рисунке 2.1-2.2 предоставлен интерфейс разработанного приложения для ввода параметров равномерного распределения и распределения Гаусса.

**2.1 Равномерное распределение**

На рисунке 2.3 предоставлен пример работы приложения при построении графика функции распределения и плотности распределения для равномерного распределения.

**2.2 Распределение Гаусса**

На рисунке 2.4 предоставлен пример работы приложения при построении графика функции распределения и плотности распределения для распределения Гаусса.

3. Листинг

В данном разделе предоставлены используемые методы для построения требуемых графиков.

|  |
| --- |
| private *fun* makeAndAddPlotsForEvenDistribution(parameterA\_: Float, parameterB\_: Float)  {  *var* parameterA = parameterA\_  *var* parameterB = parameterB\_      if (parameterA > parameterB)      {          parameterA = parameterB.also { parameterB = parameterA }      }  *val* sizeOfPlate = parameterB - parameterA  *val* stepOnPlate = sizeOfPlate / 1e3  *var* currentX = parameterA - sizeOfPlate / 3  *val* distributionSeries = XYSeries("Функция распределения")  *val* densitySeries = XYSeries("Плотность распределения")      for (i in 0 until (5 \* sizeOfPlate / (3 \* stepOnPlate)).toInt())      {  *val* curDistr = when          {          currentX < parameterA -> 0          currentX > parameterB -> 1          else -> (currentX - parameterA) / (parameterB - parameterA)          }          distributionSeries.add(currentX, curDistr)  *val* curDensity = when (currentX)          {          in parameterA..parameterB -> 1 / (parameterB - parameterA)          else              -> 0          }          densitySeries.add(currentX, curDensity)          currentX += stepOnPlate.toFloat()      }      createWindowWithPlot(          ChartPanel(          ChartFactory.createXYLineChart(              "Функция распределения",              "x",              "F(x)",              XYSeriesCollection(distributionSeries),              PlotOrientation.VERTICAL, true, true, false          )          ), "Функция распределения"      )      createWindowWithPlot(          ChartPanel(          ChartFactory.createXYLineChart(              "Функция плотности распределения",              "x",              "f(x)",              XYSeriesCollection(densitySeries),              PlotOrientation.VERTICAL, true, true, false          )          ), "Функция плотности распределения"      )  }  private *fun* makeAndAddPlotsForGaussianDistribution(parameterMu: Float, parameterSigma: Float)  {  *val* startPosition = parameterMu - 2 - parameterSigma \* 4  *val* endPosition = parameterMu + 2 + parameterSigma \* 4  *val* step = 1e-3  *val* distributionSeries = XYSeries("Функция распределения")  *val* densitySeries = XYSeries("Плотность распределения")  *var* currentX = startPosition.toDouble()      for (i in 0 until ((endPosition - startPosition) / step).toInt())      {  *val* curDistribution = 0.5 \* (1 + org.apache.commons.math.special.Erf.erf(          (currentX - parameterMu) / sqrt(2 \* parameterSigma \* parameterSigma)          ))  *val* curDensity =          (1 / (parameterSigma \* sqrt(2.0 \* PI))) \* exp(-(((currentX - parameterMu) \* (currentX - parameterMu)) / (2 \* parameterSigma \* parameterSigma)))          distributionSeries.add(currentX, curDistribution)          densitySeries.add(currentX, curDensity)          currentX += step      }      createWindowWithPlot(          ChartPanel(          ChartFactory.createXYLineChart(              "Функция распределения",              "x",              "F(x)",              XYSeriesCollection(distributionSeries),              PlotOrientation.VERTICAL, true, true, false          )          ), "Функция распределения"      )      createWindowWithPlot(          ChartPanel(          ChartFactory.createXYLineChart(              "Функция плотности распределения",              "x",              "f(x)",              XYSeriesCollection(densitySeries),              PlotOrientation.VERTICAL, true, true, false          )          ), "Функция плотности распределения"      )  } |