Государственное образовательное учреждение

высшего профессионального образования

«Уфимский Государственный Авиационный Технический Университет»

Кафедра «Вычислительной математики и кибернетики»

Отчет

По лабораторной работе

по дисциплине: Компьютерное моделирование

Лабораторная работа № 1.

Вычисление интеграла функции методом Монте-Карло

Проверил

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выполнил:

студент гр. ПРО-301в

Доронин С.Г.

Уфа 2014

# Задание

Вычислить интеграл функции методом Монте-Карло.

# Структура решения (этапы решения и их взаимосвязь)

Для решения этой задачи были созданы 3 класса:

1. Integral – Класс для вычисление интеграла
2. Func – Класс для подсчета координат функции
3. MonteCarlo – Основной класс и форма для вывода

Программа запускается с класса **MonteCarlo**, далее создается форма, при создании компонента отображения графика функции, происходит обращение к классу **Func** (для создания координат точек графика), который в свою очередь обращается к классу **Integral**.

# Описание интерфейса с пользователем (руководство пользователя)

Программа запускается с файла **KM-assembly-0.1-SNAPSHOT.jar** или с файла сценария **run.bat.** Для работы приложения требуется установленная JRE 1.7 (Java Runtime Environment).

После запуска программы появится окно, имеющее вид (Рисунок 1).

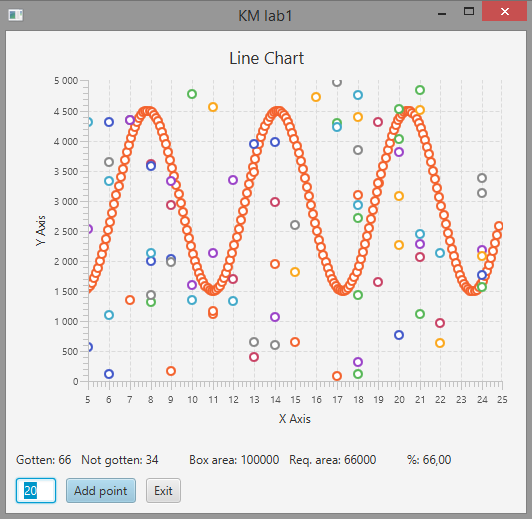


Рисунок . Главное окно программы

Сразу после запуска программы строится график интеграла и 100 точек со случайными координатами.

График интеграла имеет оранжевый цвет, случайные точки имеют разные цвета.

Количество точек попадающих / не попадающих в заданную плоскость отображается в нижней части формы. Gotten – сколько попало, Not gotten – сколько не попало (Рисунок 2).

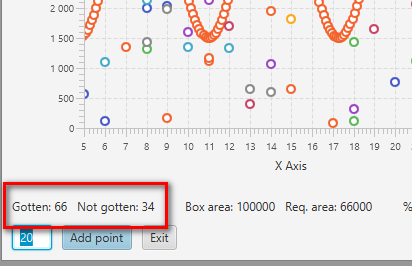


Рисунок . Количество попавших и не попавших точек в заданную плоскость

Площадь всей плоскости, рассчитанная площадь плоскости ограниченной интегралом функции, а так же % попавших точек, отображаются правее (Рисунок 3).

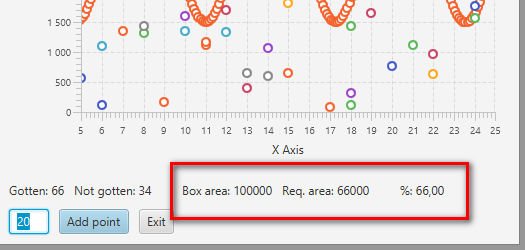


Рисунок . Площади плоскостей

Для добавления точек на график, нужно ввести количество добавляемых точек в текстовое поле в левом нижнем углу и нажать кнопку «Add point». Координаты точек будут сгенерированы случайным образом и точки будут добавлены на график (Рисунок 4).

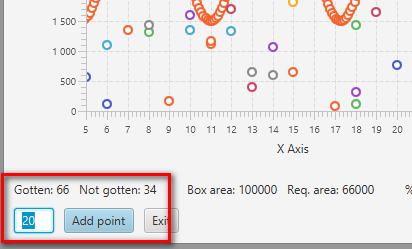


Рисунок . Добавление точек на график

Для завершения программы достаточно нажать на кнопку «Exit» или на крестик в правом верхнем углу главного окна.

# Результаты

Таблица проведения экспериментов с различным количеством сгенерированных точек:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Всего точек** | **Кол-во попавших точек** | **Кол-во не попавших точек** | **% попавших точек** | **Площадь области** |
| 1 | 50 | 29 | 21 | 58 | 58000 |
| 2 | 100 | 52 | 48 | 52 | 52000 |
| 3 | 200 | 112 | 88 | 56 | 56000 |
| 4 | 400 | 228 | 172 | 57 | 57000 |
| 5 | 800 | 456 | 344 | 57 | 57000 |

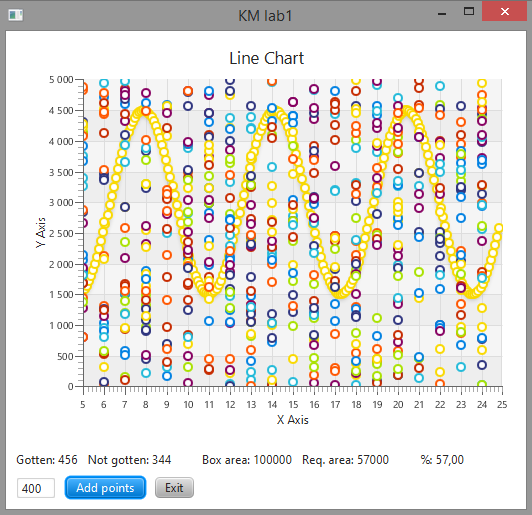


Рисунок 5. Эксперимент с 800 точками

# Код программы

…

/\*\* Добавление точек на график \*/

**private** **def** addPoints(count: Int) **=** **for** (i **<-** 1 to count) addPoint()

/\*\* Преобразование координат в точку \*/

**private** **def** toChartData **=** (xy: (Double, Double)) **=>** XYChart.Data[Number, Number](xy.\_1, xy.\_2)

/\*\* Добавление точки на график \*/

**private** **def** addPoint() {

**val** s **=** **new** XYChart.Series[Number, Number] {

**val** x **=** Random.nextInt(xUpperBounds)

**val** y **=** Random.nextInt(yUpperBounds)

data() **+=** toChartData(x, y)

**val** yCalc **=** Func.integral(x)

**if** (false) println(s"x: $x, y: $y, yCalc: $yCalc")

**if** (y **<=** yCalc) gottenPointsCount **+=** 1 **else** notGottenPointsCount **+=** 1

}

lineChart.data() **+=** s

// Обновление информации на лейблах

pointsCountLabel.text **=** s"Gotten: $gottenPointsCount Not gotten: $notGottenPointsCount"

areaLabel.text **=** s"Box area: $boxArea Req. area: $requiredArea"

partLabel.text **=** s"%: ${(requiredArea.toDouble / boxArea \* 100) formatted "**%**1.2f"}"

}

/\*\* Площадь области, в которой находится график \*/

**private** **def** boxArea **=** (xUpperBounds **-** xLowerBounds).abs \* (yUpperBounds **-** yLowerBounds).abs

/\*\* Искомая площадь фигуры \*/

**private** **def** requiredArea **=** boxArea \* gottenPointsCount / (gottenPointsCount **+** notGottenPointsCount)

/\*\* Кол-во введенных точек для добавления \*/

**private** **def** getAddPointsCount **=** addPointsCountTextField.text.value.toInt

// Обработчики событий

/\*\* Добавление точки на график \*/

addPointButton.onAction **=** (ae: ActionEvent) **=>** addPoints(getAddPointsCount)

/\*\* Выход из программы \*/

exitButton.onAction **=** (ae: ActionEvent) **=>** System.exit(0)