Практикум 1

Рассмотрим задачу разработки прикладной программы, демонстрирующей действие центральной предельной теоремы (Ц.П.Т.) и закона больших чисел [1], а также основные понятия информационных методов статистической обработки данных [2].

Как и во всяком проекте разработки программных средств, до начала непосредственного написания кода требуемой программы должны быть определены и специфицированы требования к результатам, проведено эскизное и техническое проектирование, составлен план разработки, тестирования и сдачи работы заказчику.

В крупных и масштабных проектах для выполнения указанных работ формируются специальные проектные команды, в состав которых кроме программистов включаются специалисты для заполнения необходимых проектных ролей [3,4,5,6,7,8]:

- менеджер продукта (руководитель разработки);
- ведущий программист (системный архитектор);
- системный аналитик;
- дизайнер;
- тестировщик;
- технический писатель и др.

В зависимости от установленных на предприятии или потребованных заказчиком правил выполнения работ могут применяться самые различные организационные схемы и технологии, самой простой из которых можно считать прямое и постоянное взаимодействие заказчика (потребителя, инициатора) со всей проектной командой. Проектирование и разработка программы при этом проходит несколько итераций, на каждой из которых уточняются и детализируются требования, рассматриваются и обсуждаются промежуточные результаты, регистрируются поступившие замечания и предложения, формируется детальный план работ на следующую итерацию.

В целях нашего практикума будем считать, что команда разработки состоит ровно из трех сотрудников:

- системного аналитика, устанавливающего основные функциональные требования к программе и отвечающего за взаимодействие с заказчиком;
- системного архитектора, устанавливающего все прочие требования;
- прикладного программиста, самостоятельно выполняющего необходимые функции всех остальных проектных ролей.

Итерация 1.

В ходе первого совещания проектной команды были зафиксированы следующие требования.

Требование 1.1 (Системный аналитик): Разрабатываемая программа должна иллюстрировать действие Центральной предельной теоремы с помощью генерации и суммирования нескольких равномерно распределенных случайных чисел.

Требование 1.2 (Системный аналитик): Пользователь должен иметь возможность задавать количество источников случайных чисел и объемы выборок.

Требование 1.3 (Системный аналитик): Результирующая выборка должна представляться пользователю на экране в табличном виде, а также в форме гистограммы с 10 интервалами.

Требование 1.4 (Системный аналитик): По запросу пользователя должно рассчитываться значение энтропии, пользователь должен иметь возможность управлять цветами отображения основных элементов экрана.

Требование 1.5 (Системный аналитик): Программа должна быть разработана с использованием языков HTML, JavaScript, CSS [9] и обеспечивать работу в автономном режиме (без подключения к сети Интернет) с помощью стандартного интернет-обозревателя.

Как мы видим, основные требования к программе заданы в достаточно общем виде. С одной стороны это хорошо, поскольку оставляет за системным архитектором и прикладным программистом значительную свободу в выборе технических решений. С другой стороны это может быть плохо, поскольку отсутствие согласованных детальных требований может вызвать множество замечаний заказчика, необходимость доделок и переделок программы.

Тем не менее, основные требования к программе понятны, поэтому можно зафиксировать ряд дополнительных архитектурных требований:

Требование 1.6 (Системный архитектор): Комплект поставки программы должен состоять из трех файлов, размещаемых при инсталляции в одной папке: gauss1.html, gauss1.css, gauss1.js.

Требование 1.7 (Системный архитектор): Для реализации статической части экранной формы, включая формирование гистограммы, должна использоваться разметка с помощью элементов <table>, при написании кода программы на языке JavaScript должны использоваться только стандартные функции, встроенные во все наиболее популярные интернет-обозреватели.

Требование 1.8 (Системный архитектор): Непосредственное управление отдельными элементами экрана из программы не допускается, для управления необходимо использовать смену стилей, заранее определенных в CSS.

Требование 1.9 (Системный архитектор): При программной реализации алгоритмов формирования статистической выборки должны использоваться раздельные массивы для хранения каждой из них, отдельный массив для результатов суммирования.

Требование 1.10 (Системный архитектор): При программной реализации каждая элементарная операция должна быть оформлена отдельной функцией. Оптимизация алгоритмов по времени выполнения не требуется, но время формирования отдельной выборки (в миллисекундах) должно выводиться на экран.

Итак, после первой же итерации у прикладного программиста появились определенные требования к программе, однако по-прежнему осталась значительная свобода в выборе конкретных технических решений и способах их реализации. В классической («водопадной») модели разработки программ эта итерация соответствует этапу «Техническое задание», по завершении которого можно приступать к эскизному проектированию.

Основной целью эскизного проектирования является сокращение издержек, связанных с недостаточной детализацией требований на начальных стадиях разработки. Эскизное проектирование может выполняться самыми разнообразными способами, наиболее эффективным и популярным из которых в настоящее время является прототипирование.

Перед началом работ освежим теоретические знания и вспомним, что центральная предельная теорема Ляпунова утверждает, что сумма достаточно большого количества слабо зависимых случайных величин, имеющих примерно одинаковые

масштабы имеет распределение, близкое к нормальному (Гаусса). Иллюстрацией этой теоремы будет служить примерно равномерное распределение попаданий случайных величин в заданные требованиями 10 интервалов при малом количестве источников и приближение гистограммы к форме колокола при увеличении количества.

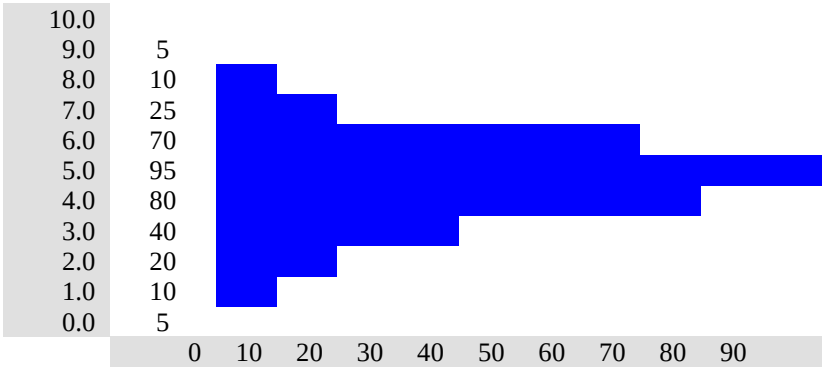
Закон больших чисел в теории вероятностей утверждает, что среднее арифметическое достаточно большой конечной выборки из известного распределения близко к математическому ожиданию этого распределения. Иллюстрацией этого закона будет служить наличие большого числа выбросов при малых выборках и приближение гистограммы к ожидаемой форме при больших выборках.

Итак, у нас определились еще и предварительные критерии оценки правильности работы программы, которые можно будет применить при сдаче работ заказчику.

Теперь начнем составлять иерархическую структуру работ нашего проекта с задачи проектирования пользовательского интерфейса. Из требований к программе следует, что для их полной реализации будет достаточно ровно одного окна (формы), разделенной на три визуальные области, одна из которых, в свою очередь, разделена еще на две области:

1. Входные данные и управление	
2.1 Таблица	2.2 Гистограмма
3. Результаты	

В области таблицы можно разместить информацию о границах 10 требуемых интервалов, а также о количестве попаданий случайных величин в каждый интервал. В области гистограммы можно визуально представить количество попаданий в каждый интервал, выделив для этого по 10 ячеек в каждой строке и закрашивая каким-то темным цветом все «полностью заполненные» ячейки:



Вернемся к области 1 и определим, какие элементы в ней должны находиться. Очевидно, что должны быть поля ввода данных (input type=text) о количестве источников и размере выборки, должна быть кнопка (input type=button) для запуска алгоритма расчета.

Для управления расчетом энтропии сделаем флажок (input type=checkbox), а также еще раз вернемся к теории и вспомним, что энтропией системы называется сумма произведений вероятностей различных состояний системы на логарифмы этих вероятностей, взятая с обратным знаком. Энтропия растет при равномерном распределении вероятностей и уменьшается при наличии закономерностей в их распределении. Иллюстрацией этого свойства может стать уменьшение энтропии при увеличении количества источников.

Осталось разобраться с требованиями к набору стилей. С их помощью можно управлять цветом фона и цветом текста всего окна, цветом и фоном отдельных областей, цветом, фоном, размерами и другими параметрами отдельных элементов. Короче говоря, текущие требования к набору стилей очень размытые.

Предположим, что пользователю будет достаточно комплекта из трех предопределенных стилей, каждый из которых задает общие параметры одновременно для всех областей, а также для визуализации гистограммы. Для выбора одного из стилей в области 1 можно разместить выпадающий список.

Итерация 2.

Эскизное проектирование заказанной программы в общих чертах закончено, и можно в принципе переходить к следующим этапам – техническому и рабочему проектированию. В классической модели разработки предполагается, что результаты эскизного проектирования документируются и согласовываются с заказчиком, при этом должно формироваться технико-экономическое обоснование всех последующих работ. Однако для достаточно простых случаев часто используется уже упоминавшийся ранее метод прототипирования, заключающийся в разработке некоторого экспериментального образца программы, с помощью которого можно как уточнить требования (в т.ч. внести необходимые изменения в Техническое задание), так и проверить на практике определенные пункты детального плана дальнейших работ.

Будем считать, что программист не стал торопиться с прототипом, а на следующем рабочем совещании произошло обсуждение эскизного проекта программы, в результате которого большинство результатов было одобрено, но при этом появились и дополнительные требования:

Требование 2.1 (Системный аналитик): Пользовательский интерфейс программы должен состоять из одного окна, разделенного на три горизонтально расположенных области: области 1 исходных данных и управления, области 2 табличной части и гистограммы, области 3 результатов.

Требование 2.2 (Системный аналитик): В области табличной части и гистограммы числа, указывающие значения границ интервалов, должны визуально располагаться на уровне соответствующих штрихов на линии координат.

Требование 2.3 (Системный аналитик): Пользователь должен имеет возможность выбора одного из предопределенных стилей для цвета фона всех областей и для цвета диаграммы, а также управления наличием сетки в области гистограммы

Требование 2.4 (Системный архитектор): Для правильной визуализации таблицы и гистограммы должны использоваться атрибуты colspan и rowspan элементов <tr> и <td>.

Все требования для программиста понятны, поэтому приступим к техническому проектированию, немедленно реализуя в прототипе все принятые проектные решения.

Итерация 2.1 - создаем файл gauss1.html

Создаем заголовок HTML, сразу же подключая файл стилей CSS и файл скрипта JavaScript:

```
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8">
    <title>Пример ЦПТ</title>
    <link rel="stylesheet" type="text/css" href="gauss1.css"/>
    <script type="text/javascript" src="gauss1.js"></script>
</head>
```

```
<body>
…
```

В теле страницы размещаем таблицу с тремя строками (<tr>), разделяющими ее на три области с именами a, b, c:

```
<table>
      <tbody>
            <tr> a </tr>
            <tr> b </tr>
            <tr> c </tr>
      </tbody>
</table>
```

В область a помещаем два прямоугольника ( ), в первом из которых помещаем элементы для ввода исходных данных и кнопки запуска алгоритма расчета (вызова функции count, которую нужно будет создать в файле gauss1.js):

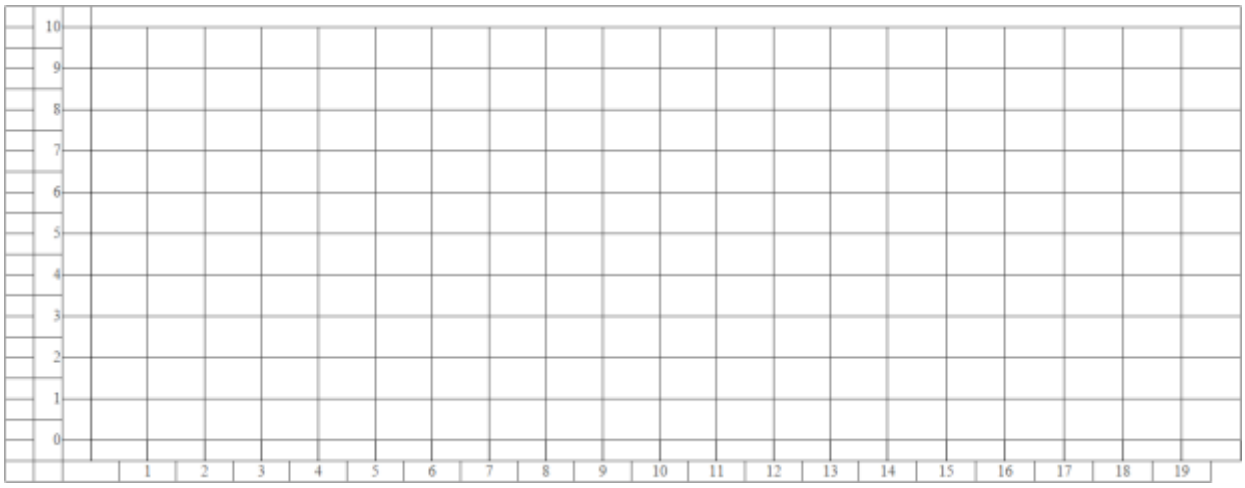
```
          <td class="form grey">
                <div class="base">
                    <h3>Основные параметры</h3>
                    <span>
                        <label for="number_of_sources">Количество источников:
</label>
                        <input type="text" id="number_of_sources"/>
                    </span>
                    <span>
                        <label for="number_of_elements">Количество элементов
в выборке: </label>
                        <input type="text" id="number_of_elements"/>
                    </span>
                    <input type="button" onclick="count(); return false;"
value="Посчитать"/>
                </div>
              <div class="additional">
                    <h3>Дополнительно</h3>
```

Второй прямоугольник делим еще на две части и размещаем в них необходимые элементы управления (включая вызов функции showHideLines, которую нужно будет создать в файле gauss1.js):

```
                    <div>
                        <span>
                            <label><input type="checkbox"
onchange="showHideLines(); return false;" id="show_lines">Показать
сетку</label>
                        </span>
                        <span>
                            <label><input type="checkbox" checked
id="show_entropy">Посчитать энтропию</label>
                        </span>
                    </div>
                    <div>
                        <span>
                            <label for="background_color">Цвет фона: </label>
                            <select name="background"
onchange="changeBackgroundColor(); return false;" id="background_color">
                                <option value="orange">Оранжевый</option>
                                <option value="blue">Синий</option>
                                <option value="grey" selected>Серый</option>
                            </select>
                        </span>
                        <span>
                            <label for="diagram_color">Цвет диаграммы:
</label>
```

```
                            <select name="diagram"
onchange="changeDiagramColor(); return false;" id="diagram_color">
                                <option value="blue" selected>Синий</option>
                                <option value="orange">Оранжевый</option>
                                <option value="grey">Серый</option>
                            </select>
                        </span>
                    </div>
```

В области b помещаем еще одну таблицу (<table>) и с помощью атрибутов colspan, rowspan делаем ее разметку показанной ниже:



Как мы видим, эта таблица имеет 23 строки и 23 столбца, при этом каждая ячейка (<td>) диаграммы создается с атрибутами colspan=2 и rowspan=2:

```
                <table>
                    <tr>
                        <td> </td>
                        <td rowspan="2" class="yAxisCategory">10</td>
                        <td> </td>
                        <td colspan="40"></td>
                    </tr>

                    <tr id="tr_10">
                        <td> </td>
                        <td class="yLine count" rowspan="2"></td>
                        <td class="item" colspan="2" rowspan="2"></td>
                        <td class="item" colspan="2" rowspan="2"></td>
                        <td class="item" colspan="2" rowspan="2"></td>
…
```

В области c размещаем одну ячейку <td> для показа результатов:

```
        <tr>
            <td class="result grey">
            </td>
        </tr>
```

Итерация 2.2 - создаем файл gauss1.css

Оформляем главную таблицу для трех областей без рамки (границы), задаем ширину 100%, и устанавливаем table-layout: fixed чтобы все ячейки были одной ширины, независимо от их заполненности:

```
table {
    border-collapse: collapse;
    width: 100%;
    table-layout: fixed;
}
```

Для каждого преопределенного стиля создаем класс, который задает цвет текста, цвет фона, и цвет границы:

```
.blue {
    color: rgba(255,255,255,.9);
    text-shadow: #2e7ebd 0 1px 2px;
    border-color: #60a3d8 #2970a9 #2970a9 #60a3d8;
    background: #60a3d8 linear-gradient(#89bbe2, #60a3d8 50%, #378bce);
    box-shadow: inset rgba(255,255,255,.5) 1px 1px;
}
.orange {
    color: rgba(255,255,255,.9);
    border-width: 2px 0;
    border-style: solid none;
    border-color: #FDBE33 #000 #ff8e06;
    background: linear-gradient(#F3AE0F, #E38916) #E38916;
}
.grey {
    border-color: rgba(0,0,0,.1);
    color: rgb(68,68,68);
    background: rgb(245,245,245) linear-gradient(#f4f4f4, #f1f1f1);
}
```

Аналогично создаем классы для стилей цвета самой диаграммы:

```
.blueDiagram {
    background-color: cornflowerblue;
}
.orangeDiagram {
    background-color: orange;
}
.greyDiagram {
    background-color: lightgrey;
}
```

Цвета фона диаграммы мы переопределяем, чтобы они не сливались с цветом самой диаграммы:

```
.diagram.orange {
    background: linear-gradient(#fce25b, #ecc92b);
}
.diagram.grey {
    background: white;
}
.diagram.blue {
    background: linear-gradient(#89bbe2, #60a3d8);
}
```

Создаем классы для меток координат, определяющее выравнивание текста метки и скрытие границ:

```
.xAxisCategory {
    text-align: center;
    border: none;
}
.yAxisCategory {
    text-align: right;
    border: none;
}
```

Для отображения чисел попаданий в каждый интервал и для штрихов на линиях координат создаем еще три класса:

```
.xLine, .yLine {
    border-right: 1px solid black;
    border-top: 1px solid black;
}
.count {
    text-align: center;
    font-weight: bold;
}
```

Для показа и скрытия сетки создаем класс, который задает параметры границы при отображении сетки.

```
.borderDiagram {
    border: 1px solid black;
}
```

Для оформления области результатов и области ввода данных и управления, создаем несколько дополнительных классов:

```
.result {
    padding: 10px;
    border: 1px solid;
}
.result span {
    display: block;
    margin-left: 20px;
}
.form {
    padding-top: 0;
    padding-right: 10px;
    padding-left: 10px;
    padding-bottom: 0;
    border: 1px solid;
}
.form div {
    padding: 0 10px 10px;
    display: inline-block;
    vertical-align: top;
}
.form div span {
    display: block;
    margin-bottom: 10px;
}
.form div h3 {
    text-align: center;
    margin-top: 5px;
    margin-bottom: 10px;
}
#number_of_sources {
    margin-left: 64px;
}
#background_color {
    margin-left: 42px;
}
.base {
    text-align: right;
}
```

Итерация 2.3 - создаем файл gauss1.js

Создаем главную функцию, которая срабатывает при нажатии кнопки «Посчитать»:

```
function count() {
    startTime = new Date().getTime();

    var n = parseInt(document.getElementById("number_of_sources").value) ||
0;
    var k = parseInt(document.getElementById("number_of_elements").value) ||
0;

    clearHistogram();
    clearResult();

    if(n === 0 || k === 0) {
        alert("Пожалуйста введите данные!");
        return;
    }

    var generatedNumbers = generate(n, k);

    var array = generatedNumbers.array;
    var min = generatedNumbers.min;
    var gradY = generatedNumbers.gradY;

    showHistogram(array, min, gradY);
    showResult(k, array);
}
```

Из элементов ввода получаем значения (количество источников и объем выборки), потом вызываем функции для очистки гистограммы и области результатов.

Если все требуемые исходные данные заданы, то вызываем функцию, которая возвращает объект из трех полей (сгенерированный массив, минимальное значение в выборке и шаг диапазонов для оси Y), потом вызываем функции для показа гистограммы и результатов соответственно.

Теперь рассмотрим подробно, как генерируется массив в функции generate:

```
function generate(numberOfSources, numberOfElements) {
    var uniformDistributionNumbersArray =
generateUniformDistributionNumbers(numberOfSources, numberOfElements);

    var normalDistributedNumbers = [];
    for(var p = 0; p < numberOfElements; p++) {
        var sum = 0;
        for(var q = 0; q < numberOfSources; q++) {
            sum += uniformDistributionNumbersArray[q][p];
        }
        var average = sum / numberOfSources;
        normalDistributedNumbers.push(average);
    }

    var max = normalDistributedNumbers[0];
    var min = normalDistributedNumbers[0];
    for(var i = 0; i < numberOfElements; i++) {
        max = normalDistributedNumbers[i] > max ? normalDistributedNumbers[i]
: max;
        min = normalDistributedNumbers[i] < min ? normalDistributedNumbers[i]
: min;
    }
    var gradY = (max - min) / 10;

    var countList = [0,0,0,0,0,0,0,0,0,0];
    for(var k = 0; k < numberOfElements; k++) {
        var round = Math.floor(normalDistributedNumbers[k] / gradY - (min /
gradY));
```

```
        countList[(round == 10) ? 9 : round] ++; // [1,2)-[2,3)-[3,4)-[4,5)-
[5,6)-[6,7)-[7,8)-[8,9)-[9,10]
    }

    return {
        array   : countList,
        min     : min,
        gradY   : gradY
    };
}
```

Сначала вызываем функцию, которая принимает в качестве параметров количество источников и объем выборки и возвращает массив из массивов равномерно распределенных чисел.

Суммируем соответствующие по индексу элементы всех массивов и вычисляем выборочное среднее этой суммы, которое согласно Ц.П.Т. будет иметь распределение , стремящееся к нормальному).

Для наглядности отображения данных в таблице и гистограмме вычисляем максимальное и минимальное значение в выборке, а так же шаг диапазонов для оси Y.

Для генерации равномерно распределенных чисел используем стандартную функцию Math.random():

```
function generateUniformDistributionNumbers(numberOfSources,
numberOfElements) {
    var uniformDistributionNumbersArray = [];
    for(var i = 0; i < numberOfSources; i++) {
        var uniformDistributionNumbers = [];
        for(var j = 0; j < numberOfElements; j++) {
            uniformDistributionNumbers.push(Math.floor(Math.random() * 10 +
1));
        }
        uniformDistributionNumbersArray.push(uniformDistributionNumbers);
    }

    return uniformDistributionNumbersArray;
}
```

Для вычисления энтропии создаем функцию, вычисляющую ее через двоичный логарифм (в битах):

```
function countEntropy(count, array) {
    var entropy = 0;
    for (var i = 0; i < array.length; i++) {
        var p = array[i] / count;
        entropy += (p !== 0) ? (-1) * p * Math.log(p) / Math.LN2 : 0;
    }
    return entropy.toFixed(2);
```

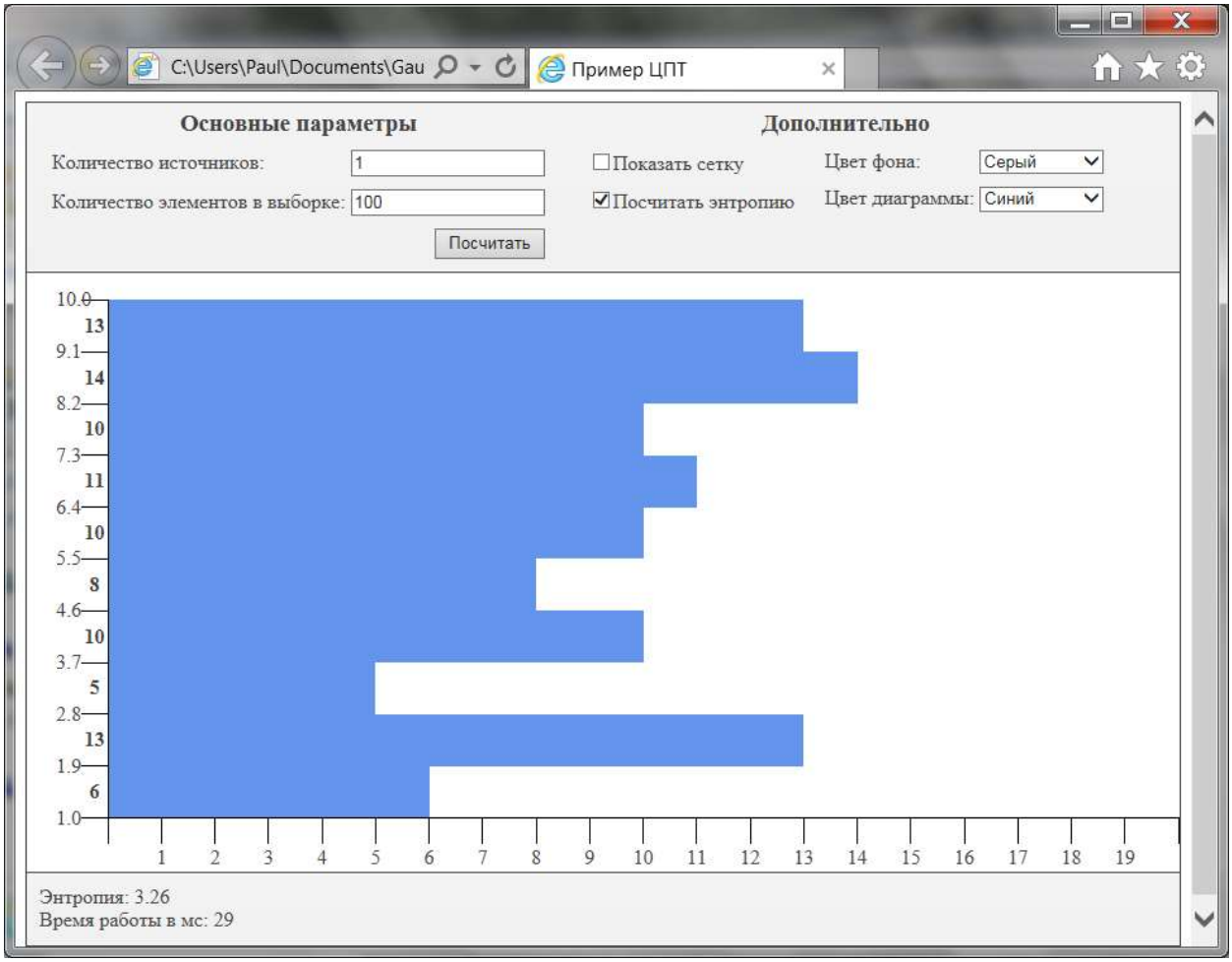
Остальные функции имеют технических характер, их назначение очевидно из их имен, алгоритмы работы – из их текста.

Итерация 3.

В общих чертах программист закончил технорабочее проектирование, остается провести приемо-сдаточные испытания, внести исправления по полученным замечаниям, закончить разработку необходимой программной документации.

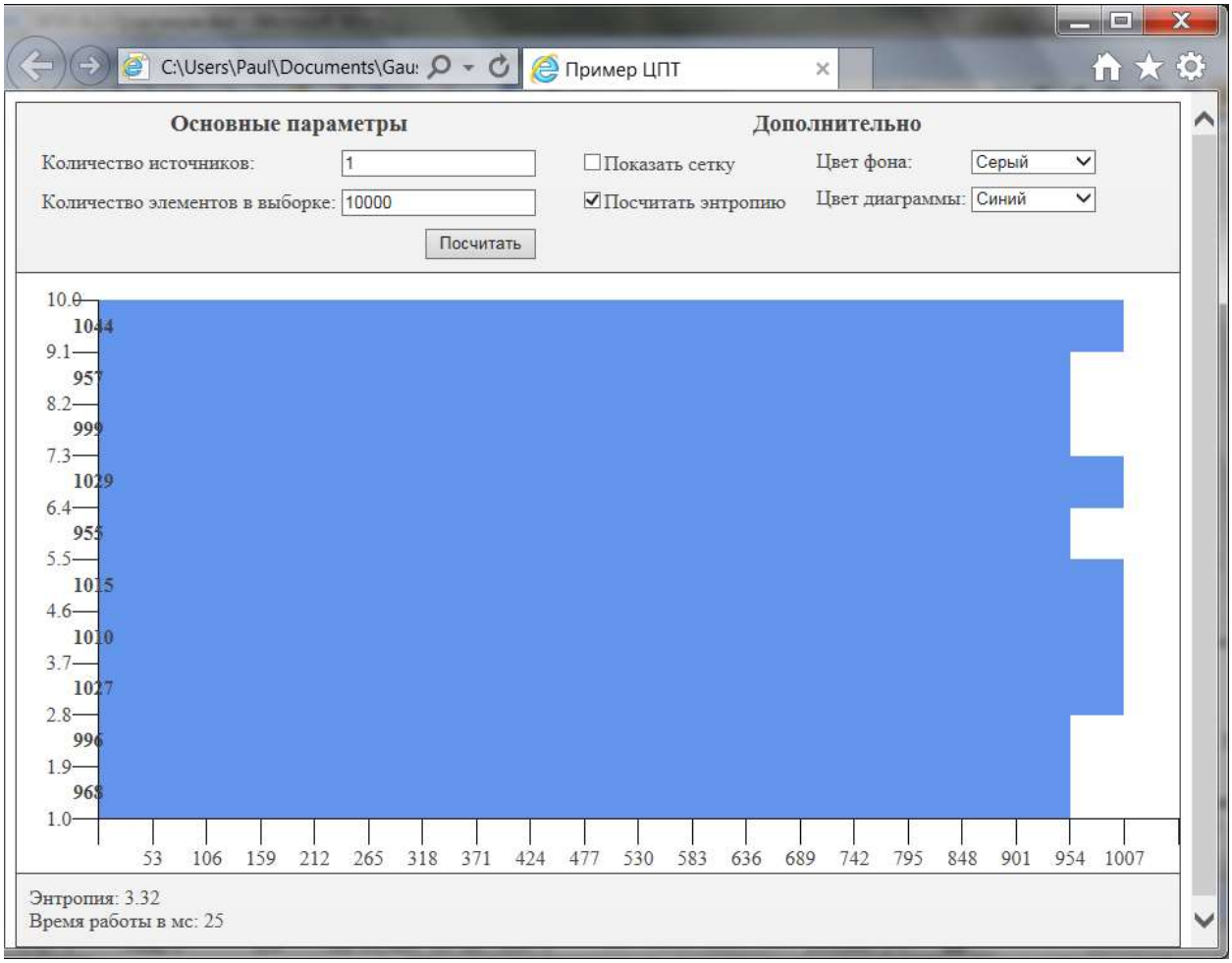
Проведем испытания программы в форме демонстрации, для чего достаточно скачать файлы программы [файлы программы] на компьютер пользователя и открыть файл gauss1.html в стандартном интернет-обозревателе.

Проверка 1. Сформируем выборку из 1 источника со 100 элементами:



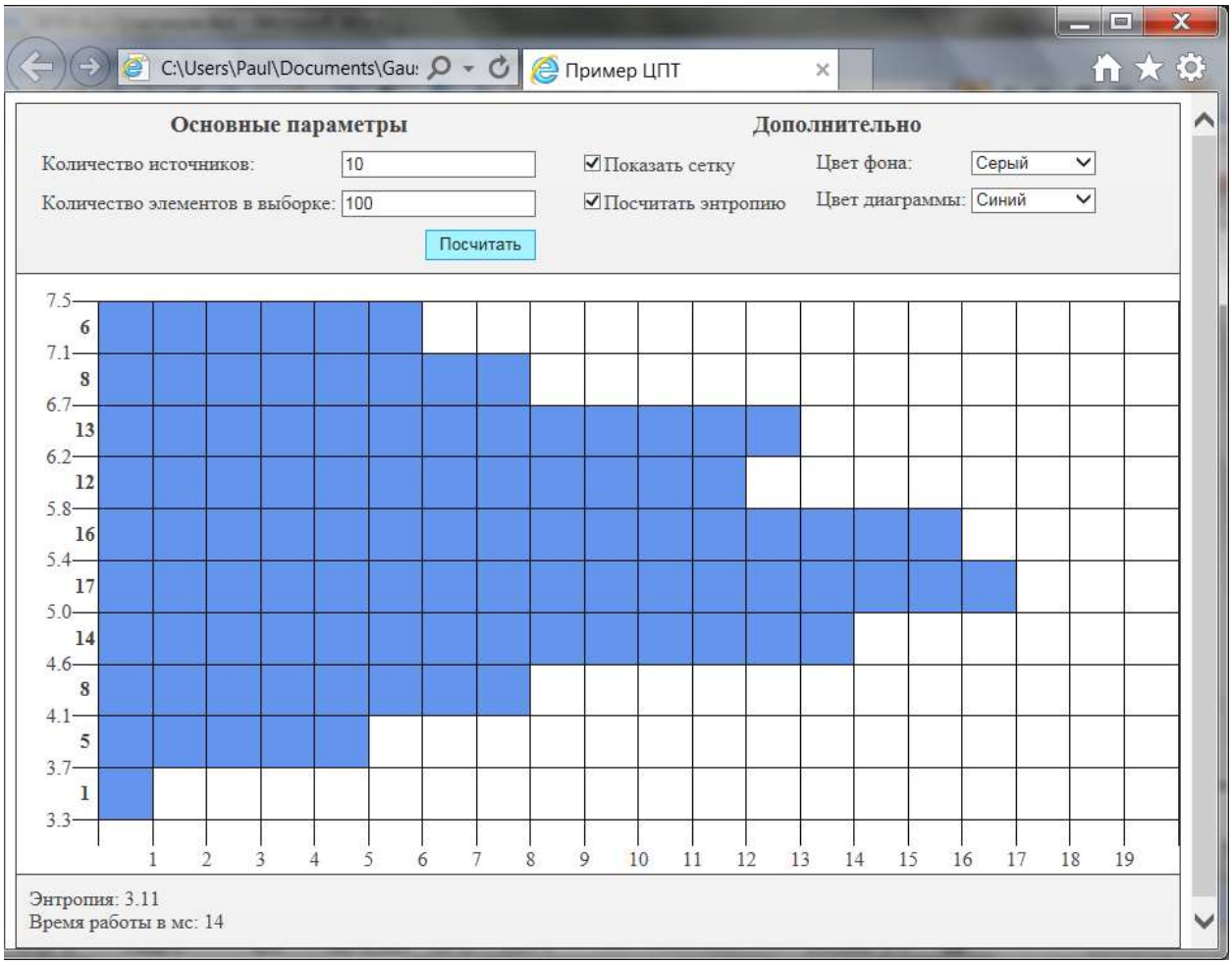
Как мы видим, распределение близко к равномерному, но имеется большое число выбросов.

Проверка 2. Увеличим число элементов в выборке до 1000:



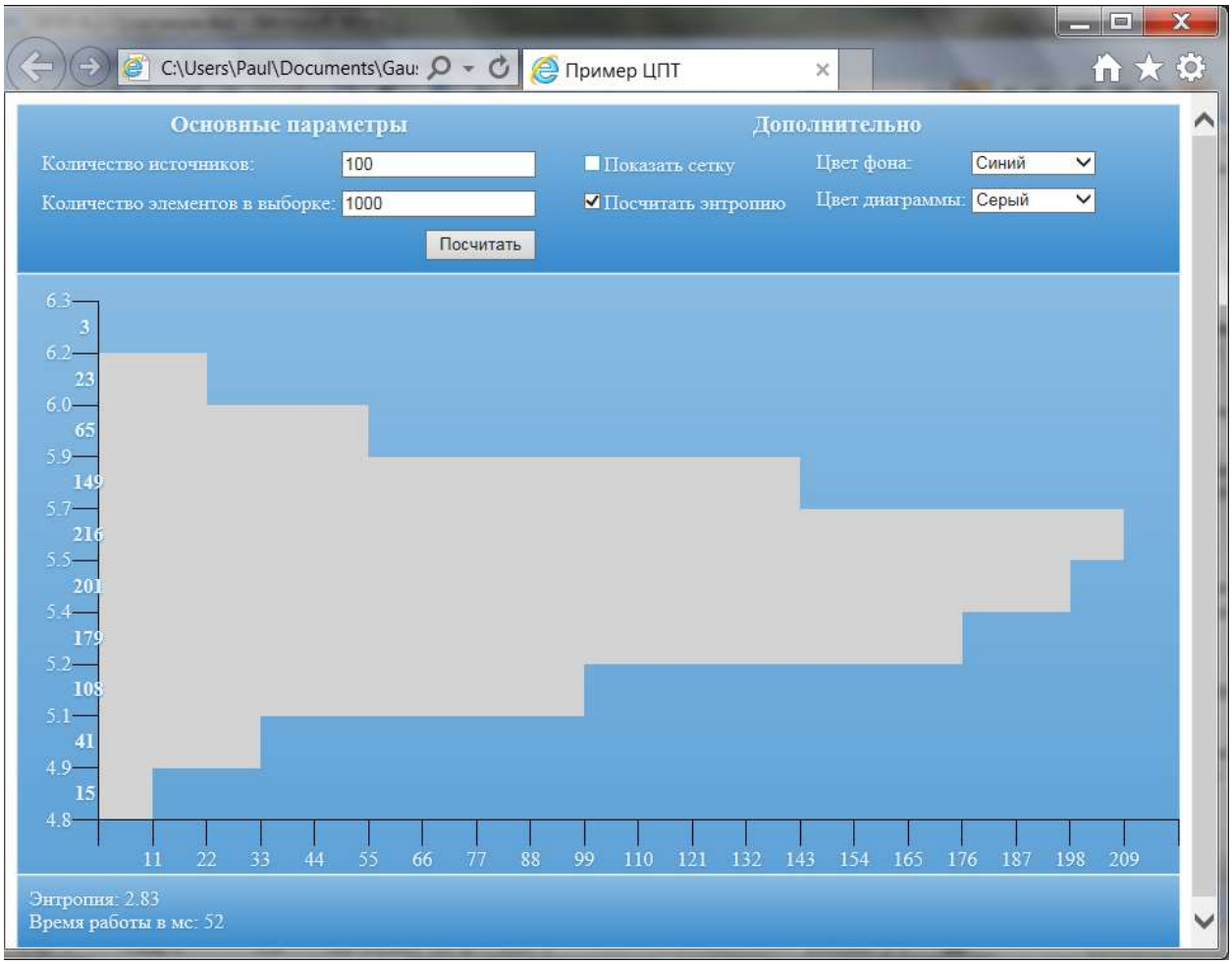
Как мы видим, распределение почти неотличимо от равномерного, а энтропия немного увеличилась. Закон больших чисел работает, и программа это наглядно демонстрирует!

Проверка 3. Увеличим число источников до 10 при размере выборки 100. Заодно проверим управление появлением линий сетки в гистограмме:



Как мы видим, распределение имеет явно выраженное приближение к нормальному, а энтропия уменьшилась. Линии сетки показаны, что повышает наглядность.

Проверка 4. Увеличим число источников до 100, а размеры выборки до 1000. Заодно проверим управление цветом фона окна и диаграммы:



Как мы видим, приближение распределения к нормальному стало еще более явным, а энтропия еще уменьшилась. Центральная предельная теорема работает, программа это наглядно демонстрирует! Цветами фона и диаграммы пользователь может управлять без необходимости повторного вызова алгоритма расчетов.

Задачи для самостоятельной работы

Задача 1. Составьте иерархический список всех элементов пользовательского интерфейса (в порядке Z-order), иерархический список структуры программы и иерархический список использованных структур данных (включая список стилей).

Задача 2. В большинстве инженерных задач используются расчеты энтропии и информации не в битах (через двоичный логарифм), а в натах (через натуральный логарифм). Исправьте этот недостаток программы и в скобках после слова «Энтропия» укажите использованные единицы измерения.

Задача 3. При больших значениях числа попаданий в интервал это число не помещается в ячейке (Проверка 2, Проверка 4). Исправьте этот недостаток программы.

Задача 4. При малых значениях размера интервала отметки границ начинают сливаться при отображении одного числа после запятой (Проверка 4). Исправьте этот недостаток программы.

Задача 5. Дополните список допустимых цветов фона черным и белым, допустимых цветов диаграммы – черным, белым и желтым.

Задача 6. Дополните результаты расчетов значениями математического ожидания (выборочного среднего) и среднеквадратичного отклонения полученной выборки.

Литература:

1. *Вентцель Е.С.* Теория вероятностей: Учеб. для вузов. — 6-е изд. стер. — М.: Высш. шк., 1999.— 576 с.
2. *Григорович В.Г., Юдин С.В., Козлова Н.О., Шильдин В.В.* Информационные методы в управлении качеством. — М.: РИА "Стандарты и качество", 2001. — 208 с.
3. Профессиональный стандарт «Менеджер продуктов в области информационных технологий»
4. Профессиональный стандарт «Руководитель разработки программного обеспечения», версия 5.140917, утвержден Приказом Минтруда России №645н от 17.09.2014
5. Профессиональный стандарт «Системный аналитик»
6. Профессиональный стандарт «Специалист по тестированию в области информационных технологий», версия 5.131214, утвержден Приказом Минтруда России №225н от 11.04.2014
7. Профессиональный стандарт «Технический писатель (Специалист по технической документации в области ИТ)», версия 5.140908, утвержден Приказом Минтруда России №612н от 8.09.2014
8. Профессиональный стандарт «Специалист по информационным ресурсам», версия 5.140908, утвержден Приказом Минтруда России №629н от 8.09.2014
9. Электронный учебник по JavaScript