Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования «Тульский государственный университет»

КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

ГЕНЕРАЦИЯ НОРМАЛЬНОГО И ЗАДАННОГО РАСПРЕДЕЛЕНИЙ

отчет о
лабораторной работе №2

по дисциплине
ТЕХНОЛОГИИ И МЕТОДЫ ПРОГРАММИРОВАНИЯ

ВАРИАНТ 6

Выполнила:	ст. гр. 230711	Павлова В.С.
Проверил:	асс. каф. ИБ	Курбаков М.Ю.

Тула, 2022 г.

ЦЕЛЬ И ЗАДАЧА РАБОТЫ

Цель: изучить генерацию случайных величин по заданному и нормальному законам распределения.

Задача: в данной работе требуется написать программы, демонстрирующие использование изученных принципов.

ЗАДАНИЕ НА РАБОТУ

1) С помощью метода обратной функции получить случайную величину с заданной по варианту плотностью распределения $f(x)$, график которой приведён на рисунке 1.
2) Построить нормальное распределение с заданными математическим ожиданием и дисперсией. Полученную в результате генерирования плотность вероятности сравнить с теоретической.

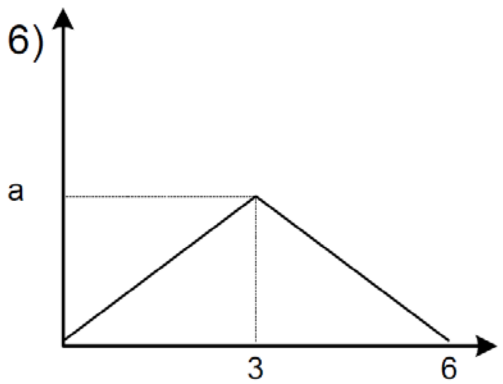


Рисунок 1 – График функции плотности распределения

СХЕМА ПРОГРАММЫ

1) Схема алгоритма, предназначенного для генерации заданного распределения, представлена на рисунке 2.

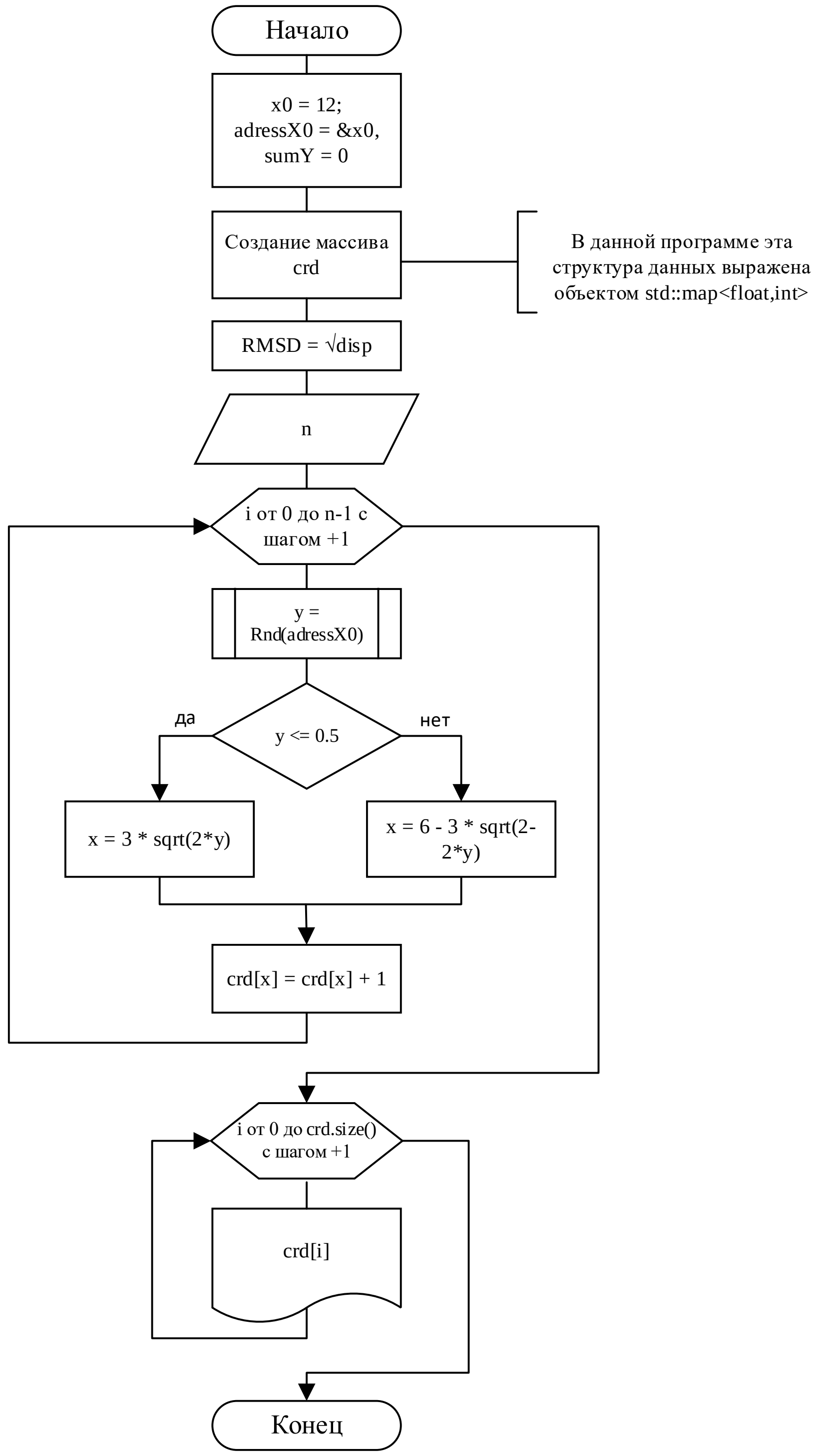


Рисунок 2 – Схема алгоритма для генерации заданного распределения

2) Схема алгоритма, предназначенного для генерации нормального распределения, представлена на рисунке 3.

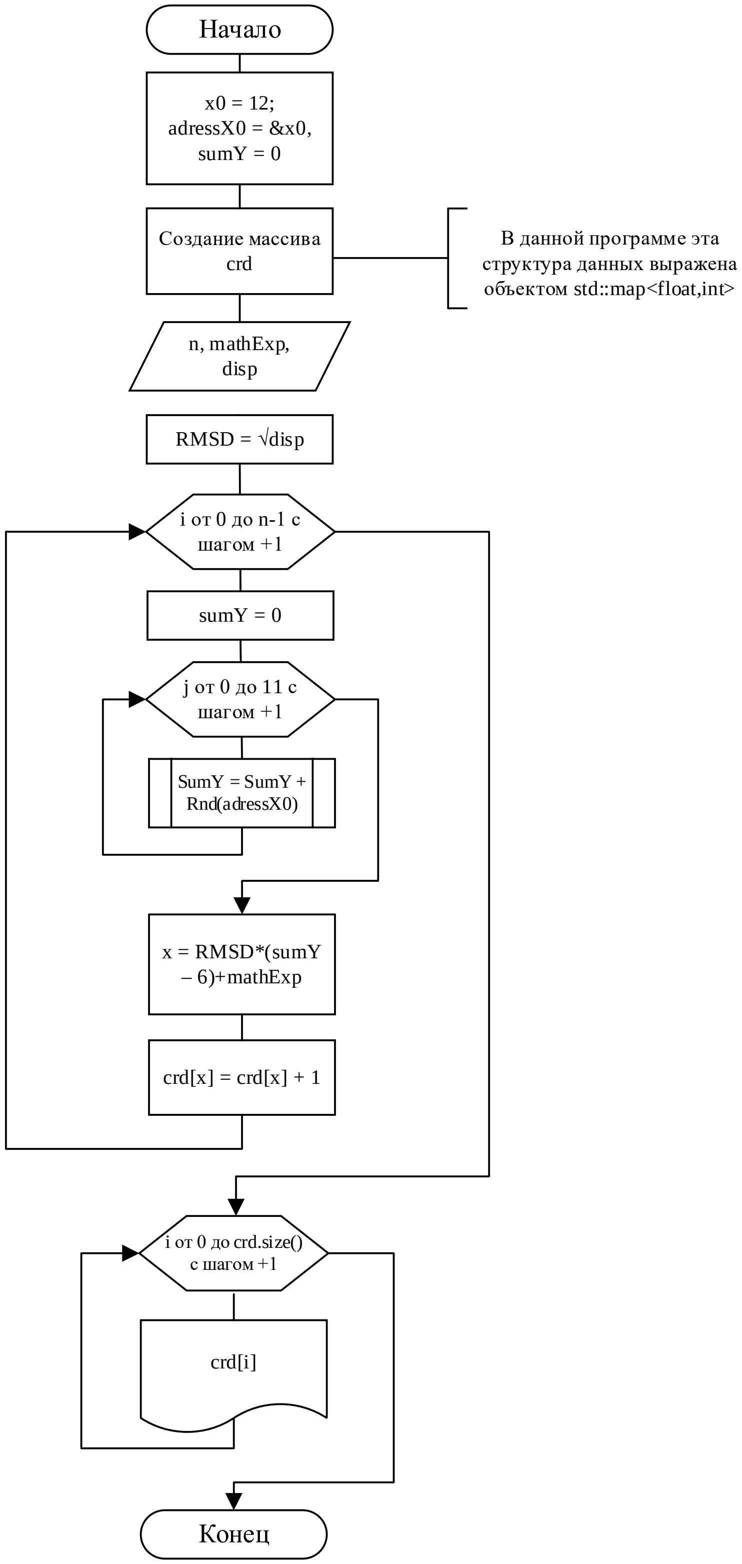


Рисунок 3 – Схема алгоритма для генерации нормального распределения

ТЕКСТ ПРОГРАММЫ

Текст программы на языке программирования C++ для генерации заданного распределения представлен в листинге 1.

Листинг 1. Текст программы

```
#include <iostream>
#include <cmath>
#include <map>
double Rnd(int* x0) //(0;1)
{
    int c = 15, m = 65536, a = 13;
    int val = (a * (*x0) + c) % m;
    *x0 = val;
    return (float)val / m;
}
int main()
{
    setlocale(LC_ALL, "Russian");
    int n, x0 = 12;
    int* adressX0 = &x0;
    float x, y, sumY = 0;
    std::map <float, int> crd;
    std::cout << "\t\tВведите длину заданного распределения: ";
    std::cin >> n;
    std::cout << "\n";
    for (int i = 0; i < n; ++i)
    {
        y = Rnd(adressX0);
        if (y <= 0.5) x = 3 * sqrt(2 * y);
        else x = 6 - 3 * sqrt(2 - 2 * y);
        crd[floor(x * 10) / 10]++;
    } std::cout << "\n";
    for (auto to : crd)
        std::cout << to.first << "\t
          << (float)to.second / n << "\n";
    return 0;
}
```

Текст программы на языке программирования C++ для генерации нормального распределения представлен в листинге 2.

Листинг 2. Текст программы

```
#include <iostream>
#include <fstream>
#include <cmath>
```

Листинг 2. Текст программы (продолжение)

```
#include <map>
double Rnd(int* x0) //(0;1)
{
    int c = 15, m = 65536, a = 13;
    int val = (a * (*x0) + c) % m;
    *x0 = val;
    return (float)val / m;
}
int main()
{
    setlocale(LC_ALL, "Russian");
    int n, m, x0 = 12;
    float mathExp, disp;
    int* adressX0 = &x0;
    float x, y, sumY = 0;
    std::map <float, int> crd;
    std::cout << "\t\tВведите длину нормального распределения: 
";
    std::cin >> n;
    std::cout << "\t\tВведите дисперсию и математическое 
ожидание: ";
    std::cin >> mathExp >> disp;
    float RMSD = sqrt(disp);
    std::cout << "\n";
    for (int i = 0; i < n; ++i)
    {
        sumY = 0;
        for (size_t i = 0; i < 12; i++)
        {
            sumY += Rnd(adressX0);
        }
        x = RMSD * (sumY - 6) + mathExp;
        crd[floor(x * 10) / 10]++;
    } std::cout << "\n";
    std::map<float, int>::iterator to = crd.begin();
    for (size_t i = 0; i < crd.size(); i++)
    {
        std::cout << to->first << "\t" << (float)to->second / n 
<< "\n";
        to++;
    } std::cout << "\n\n";
    return 0;
}
```

ИНСТРУКЦИЯ ПОЛЬЗОВАТЕЛЯ

Первая программа предназначена для генерации заданного распределения. При запуске программы пользователю предлагается ввести длину последовательности. После программа формирует последовательность чисел заданной длины, а также рассчитывает координаты точек для построения графика плотности вероятности и выводит эти координаты на экран.

Вторая программа работает аналогично первой. При запуске пользователю так же предлагается ввести длину последовательности, а ещё математическое ожидание и дисперсию. После программа формирует последовательность заданной длины, а также рассчитывает координаты точек для построения графика плотности вероятности и выводит их.

ИНСТРУКЦИЯ ПРОГРАММИСТА

Структуры данных, используемые в обеих программах, аналогичны, поэтому все они приведены в одной таблице (таблица 1).

Таблица 1 – Структуры данных в программе

Имя	**Тип (класс)**	**Предназначение**
n	int	Длина последовательности
crd	map	Структура для расчёта точек графика плотности распределения
x	float	Величина
x0	int	Начальное число для линейного конгруэнтного генератора
adressX0	*int	Адрес ячейки памяти x0
y	float	Величина, полученная линейным конгруэнтным генератором
sumY	float	Счётчик суммы чисел
mathExp	float	Математическое ожидание
disp	float	Дисперсия
RMSD	float	Среднеквадратичное отклонение (сигма)

В обеих программах имеется следующая подпрограмма:

1) `double Rnd(int *x0)` – функция, генерирующая случайное число с использованием линейного конгруэнтного метода. Структуры данных, используемые в подпрограмме, приведены в таблице 2.

Таблица 2 – Структуры данных, используемые в подпрограмме `Rnd()`

Имя	**Тип**	**Предназначение**
		формальные параметры
x0	*int	Ссылка на предыдущее полученное число
		локальные переменные
c, a, m	const int	Параметры линейного конгруэнтного генератора
val	double	Величина

ДЕМОНСТРАЦИОННЫЙ ПРИМЕР

1) Для получения случайной величины с заданной по варианту плотностью распределения методом обратной функции сперва проведём аналитические расчёты и определим эту обратную функцию. Согласно заданию варианта, плотность распределения имеет вид, представленный на рисунке 4:

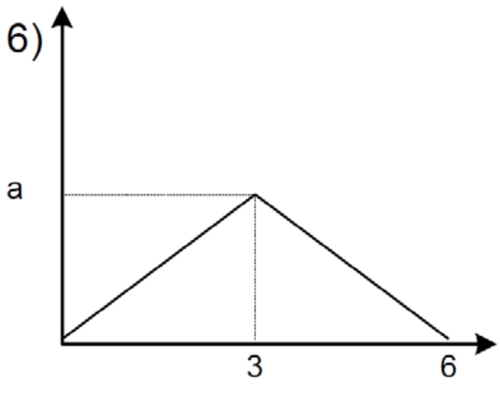


Рисунок 4 – Плотность распределения

Прежде всего, определим постоянную a, используя условие нормировки $\int_{-\infty}^{\infty} f(x)dx = 1$. Поскольку интеграл является площадью под графиком, то его можно определить как S = $\frac{1}{2}a * b = 1$. Здесь b – сторона треугольника, равная 6. Отсюда $a = \frac{2}{6} = \frac{1}{3}$. Запишем теперь функцию распределения, учтя константу а:

$$f(x) = \begin{cases} \frac{x}{9}, & x \in [0,3] \\ \frac{2}{3} - \frac{x}{9}, & x \in (3,6] \end{cases}$$

Далее найдём функцию распределения $F(x) = \int_{-\infty}^{x} f(x)dx$:

$$F(x) = \begin{cases} \frac{x^2}{18}, & x \in [0,3] \\ \frac{2}{3}x - \frac{x^2}{18} - 1, & x \in (3,6] \end{cases}$$

Теперь найдем функцию, обратную к $F(x)$. Для этого выразим x через y:

1) $y = \frac{x^2}{18}$, отсюда $x = \sqrt{18\,y} = 3\sqrt{2y}$

Пересчитаем область определения, подставив значения $x = 0$, $x = 3$ в полученное уравнение $x = 3\sqrt{2y}$. Если $x = 0$, тогда $y = 0$. Если же $x = 3$, тогда имеем $y = 0.5$.

Отсюда первое уравнение из системы: $x = 3\sqrt{2y}$, $y \in (0, 0.5]$

2) $y = \frac{2}{3}x - \frac{x^2}{18} - 1$, домножим на 18 и получим

$x^2 - 12x + 18y + 18 = 0$, отсюда

$x_1 = \frac{-\sqrt{b^2-4ac}-b}{2a} = \frac{-\sqrt{144-4*(18y+18)}+12}{2} = 6 - 3\sqrt{2-2y}$,

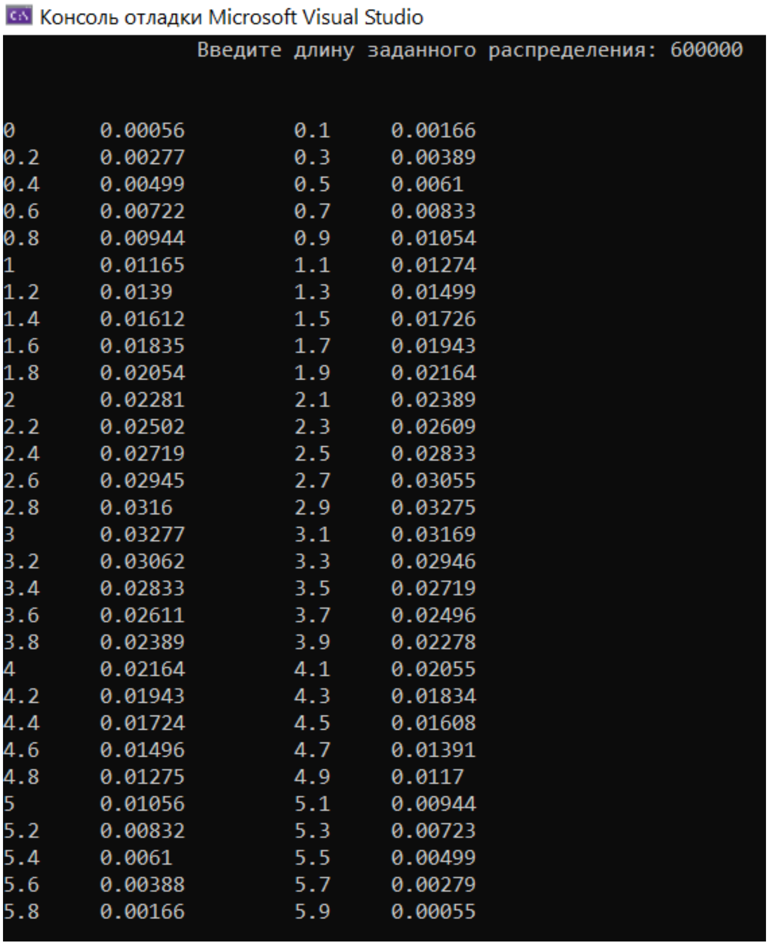
$x_2 = \frac{\sqrt{b^2-4ac}-b}{2a} = \frac{\sqrt{144-4*(18y+18)}+12}{2} = 6 + 3\sqrt{2-2y}$.

Пересчитаем область определения, подставив значения $x = 3$, $x = 6$ в уравнение $x_1 = 6 - 3\sqrt{2-2y}$. Если $x_1 = 3$, тогда $y = 0.5$. Если $x_1 = 6$, тогда $y = 1$.

Попробуем подставить $x = 3$ во второе уравнение, которое имеет вид $x_2 = 6 + 3\sqrt{2-2y}$. Это уравнение не подходит, т.к. не будет иметь решений, ведь квадратный корень не может быть отрицательным. Тогда второе уравнение системы – это уравнение $x = 6 - 3\sqrt{2-2y}$, а обратная функция имеет вид:

$$F^{-1}(y) = \begin{cases} 3\sqrt{2y}, & y \in [0, 0.5] \\ 6 - 3\sqrt{2 - 2y}, & y \in (0.5, 1] \end{cases}$$

С помощью полученных формул программа, приведённая в листинге 1, производит расчёты. Рассмотрим теперь результат работы этой программы для n = 600000 чисел, он приведён на рисунке 5.



Консоль отладки Microsoft Visual Studio

Введите длину заданного распределения: 600000

0	0.00056	0.1	0.00166
0.2	0.00277	0.3	0.00389
0.4	0.00499	0.5	0.0061
0.6	0.00722	0.7	0.00833
0.8	0.00944	0.9	0.01054
1	0.01165	1.1	0.01274
1.2	0.0139	1.3	0.01499
1.4	0.01612	1.5	0.01726
1.6	0.01835	1.7	0.01943
1.8	0.02054	1.9	0.02164
2	0.02281	2.1	0.02389
2.2	0.02502	2.3	0.02609
2.4	0.02719	2.5	0.02833
2.6	0.02945	2.7	0.03055
2.8	0.0316	2.9	0.03275
3	0.03277	3.1	0.03169
3.2	0.03062	3.3	0.02946
3.4	0.02833	3.5	0.02719
3.6	0.02611	3.7	0.02496
3.8	0.02389	3.9	0.02278
4	0.02164	4.1	0.02055
4.2	0.01943	4.3	0.01834
4.4	0.01724	4.5	0.01608
4.6	0.01496	4.7	0.01391
4.8	0.01275	4.9	0.0117
5	0.01056	5.1	0.00944
5.2	0.00832	5.3	0.00723
5.4	0.0061	5.5	0.00499
5.6	0.00388	5.7	0.00279
5.8	0.00166	5.9	0.00055

Рисунок 5 – Результат работы программы

Необходимо сравнить полученную плотность распределения с заданной по варианту. Для проверки полученного результата построим график $f(x)$ плотности распределения, согласно значениям, полученным в программе.

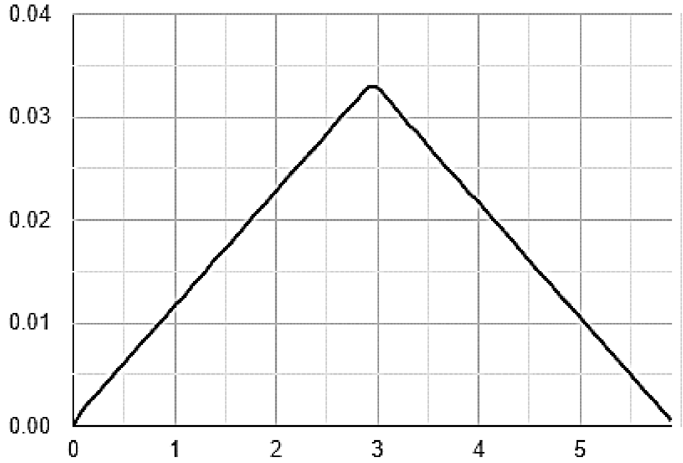


Рисунок 6 – График плотности заданного распределения

Промежуточный вывод: как видно по рисунку 6, есть незначительное отклонение от теоретического, однако график в целом соответствует теоретической плотности распределения.

2) Исходя из рисунка 1, приведённого в задании на работу, найдем математическое ожидание и дисперсию для заданного распределения:

$$m = \int_0^3 x\frac{x}{9}dx + \int_3^6 x\left(\frac{2}{3} - \frac{x}{9}\right)dx = 3.$$

$$D = \int_0^3 (x-3)\frac{x}{9}dx + \int_3^6 (x-3)^2(\frac{2}{3} - \frac{x}{9})dx = 1.5.$$

Для генерации нормального распределения воспользуемся этими параметрами и формулой $X = \frac{12\sigma}{n}\left(\sum_{i=1}^{n} Y_i - \frac{n}{2}\right) + m,$ где $\sigma = \sqrt{D}, n = 12,$ а Y_i – независимая случайная равномерная величина.

```
Выбрать Консоль отладки Microsoft Visual Studio
                Введите длину нормального распределения: 600000
                Введите дисперсию и математическое ожидание: 1.5 3


-6.1    6e-05           -5.2    6e-05           -5.1    6e-05
-4.8    6e-05           -4.7    6e-05           -4.6    0.00012
-4.5    0.00012         -4.4    6e-05           -4.3    0.00012
-4.2    0.00012         -4.1    0.00018         -4      0.0003
-3.9    0.00018         -3.8    0.0003          -3.7    0.00036
-3.6    0.00036         -3.5    0.00061         -3.4    0.00055
-3.3    0.00097         -3.2    0.00092         -3.1    0.00122
-3      0.00116         -2.9    0.00121         -2.8    0.00164
-2.7    0.00183         -2.6    0.0025          -2.5    0.00226
-2.4    0.00286         -2.3    0.00305         -2.2    0.00385
-2.1    0.00396         -2      0.00415         -1.9    0.00487
-1.8    0.00427         -1.7    0.00542         -1.6    0.00488
-1.5    0.00599         -1.4    0.00622         -1.3    0.00695
-1.2    0.00764         -1.1    0.00951         -1      0.00892
-0.9    0.00969         -0.8    0.01116         -0.7    0.01042
-0.6    0.01155         -0.5    0.01152         -0.4    0.0128
-0.3    0.01311         -0.2    0.01464         -0.1    0.01594
0       0.01593         0.1     0.01763         0.2     0.01752
0.3     0.01721         0.4     0.01886         0.5     0.01911
0.6     0.01801         0.7     0.01922         0.8     0.01972
0.9     0.02019         1       0.02001         1.1     0.02104
1.2     0.0224          1.3     0.02001         1.4     0.02048
1.5     0.02105         1.6     0.02074         1.7     0.02056
1.8     0.02046         1.9     0.02166         2       0.02055
2.1     0.01971         2.2     0.02057         2.3     0.01982
2.4     0.01794         2.5     0.01872         2.6     0.01745
2.7     0.01825         2.8     0.01528         2.9     0.01622
3       0.01612         3.1     0.01643         3.2     0.01441
3.3     0.01368         3.4     0.01244         3.5     0.01129
3.6     0.00999         3.7     0.01037         3.8     0.01028
3.9     0.0094          4       0.00812         4.1     0.00807
4.2     0.00682         4.3     0.00739         4.4     0.00555
4.5     0.00563         4.6     0.00488         4.7     0.0039
4.8     0.00409         4.9     0.00354         5       0.0036
5.1     0.00322         5.2     0.00305         5.3     0.00274
5.4     0.00268         5.5     0.0014          5.6     0.00146
5.7     0.00146         5.8     0.00134         5.9     0.00134
6       0.00109         6.1     0.00145         6.2     0.00091
6.3     0.00048         6.4     0.00061         6.5     0.00036
6.6     0.00054         6.7     0.00024         6.8     0.00018
6.9     0.00024         7       0.00012         7.1     0.00018
7.2     0.00024         7.3     0.00018         7.4     0.00018
7.5     0.00018         7.7     0.00018         7.8     6e-05
7.9     6e-05           8       6e-05           8.1     0.00012
8.2     6e-05
```

Рисунок 7 – Результат работы программы

Для проверки полученного результата построим график $f(x)$ для нормального распределения, согласно значениям, полученным в программе. Этот график приведён на рисунке 8.

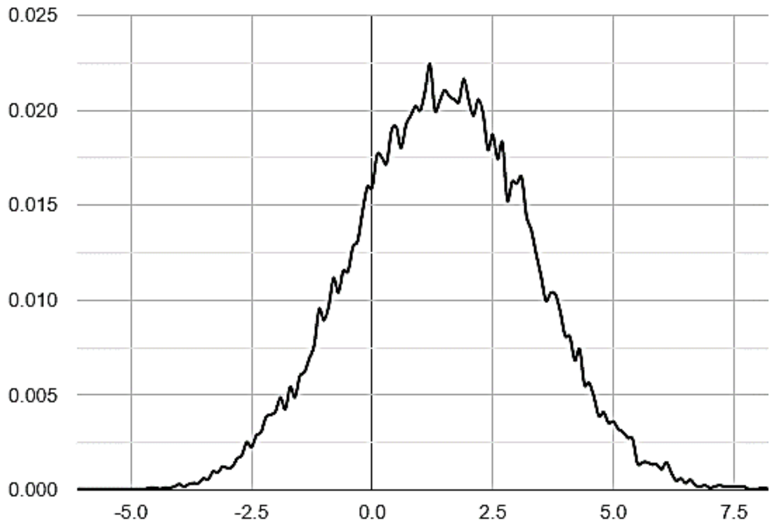


Рисунок 8 – Графики плотности полученного распределения

Теперь необходимо построить кривую распределения Гаусса, рассчитав значения функции для заданных параметров математического ожидания и дисперсии. Полученный график приведен на рисунке 9.

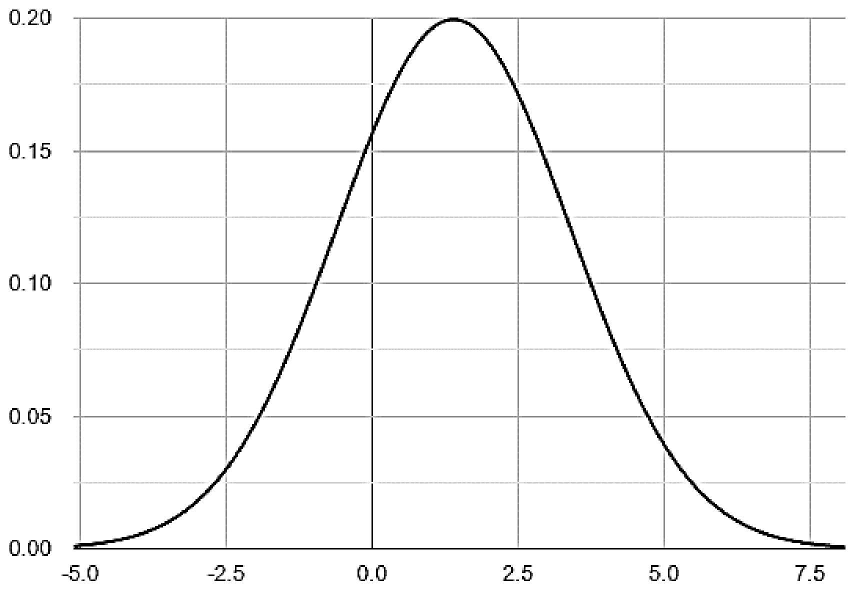


Рисунок 9 – Кривая распределения Гаусса для заданных параметров математического ожидания и дисперсии

Промежуточный вывод: как видно по рисункам 8 и 9, общий вид функции плотности полученного нормального распределения соответствует кривой распределения Гаусса с некоторым отклонением.

ВЫВОДЫ

В ходе данной лабораторной работы был изучен принцип генерации случайных величин по заданному и нормальному законам распределения. Для демонстрации полученных знаний была написаны программы для генерации соответствующих распределений, результат работы которой был проверен аналитически. В ходе проверки с помощью плотностей распределения обнаружилось, что полученные распределения соответствуют теоретическим с незначительным отклонением.