МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

МОДЕЛИРОВАНИЕ СИСТЕМ. ВАРИАНТ 2

ОТЧЕТ О ПРАКТИКЕ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Аношкина Андрея Алексеевича

Проверил
к. ф.-м. н., доцент ______________ И. Е. Тананко

Саратов 2025

СОДЕРЖАНИЕ

1 МОДЕЛИРОВАНИЕ НЕПРЕРЫВНЫХ СИСТЕМ

1.1 Дифференциальные уравнения первого порядка

1.1.1 Постановка задачи

Температура свежевыпеченного хлеба равна 150°. До отправки в магазин хлеб остывает в помещении с постоянной температурой 20°. Требуется определить длительность времени охлаждения хлеба до 40°.

Результат можно получить с использованием закона теплового излучения

$$\frac{dx}{dt} = -k(x - a),$$

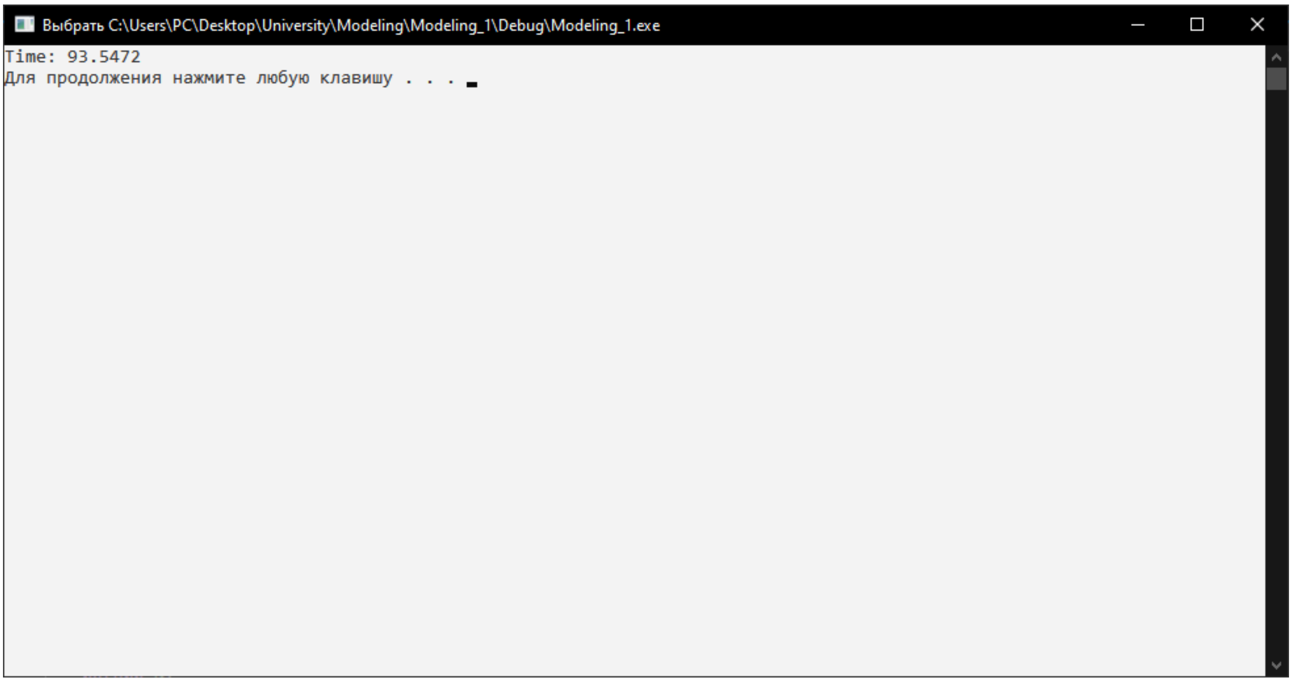
где $x(t)$ — температура хлеба в момент времени t, a — температура воздуха в помещении, $k > 0$ — коэффициент пропорциональности. Полагаем, что $k = 0,02$.

1.1.2 Код

```
#include <iostream>

using namespace std;

const int a = 20;
const float k = 0.02;
const float dt = 1e-3;
const float T = 40;
const float eps = 1e-3;

float f(const float x) {
	return -k * (x - a);
}

int main() {
	float x = 150;
	float t = 0;

	for (; abs(x - T) >= eps; t += dt)
		x += dt * f(x);

	cout << "Time: " << t << "\n";

	system("pause");

```

```
    return 0;
}
```

1.1.3 Результат



1.2 Системы дифференциальных уравнений

1.2.1 Постановка задачи

Построить фазовый портрет системы дифференциальных уравнений

$$\begin{cases} \frac{dx}{dt} = ax - y \\ \frac{dy}{dt} = x + ay \end{cases}$$

при $-0,2 < a < 0$. Эксперимент повторить при $a > 0$.

1.2.2 Код

```
import matplotlib.pyplot as plt
import numpy as np

a = float(input())

def f(x, y):
    return a * x - y

def g(x, y):
    return x + a * y
```

```
x = 1
y = 1
t = 0
T = 100
dt = 1e-2

x_coords = [x]
y_coords = [y]

while t < T:
    xn = x + dt * f(x, y)
    yn = y + dt * g(x, y)
    x_coords.append(xn)
    y_coords.append(yn)
    x = xn
    y = yn
    t += dt

xs = np.array(x_coords, dtype=float)
ys = np.array(y_coords, dtype=float)

plt.plot(xs, ys)
plt.show()
```

1.2.3 Результат

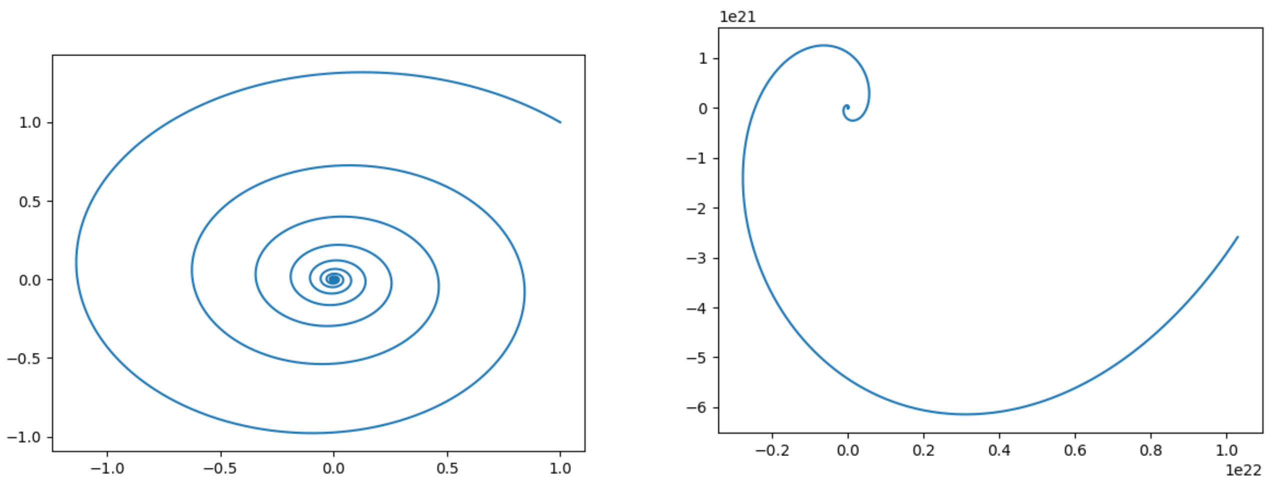


Рисунок 1 – Левый — $a = -0.1$; Правый — $a = 0.5$

2 МЕТОД СТАТИСТИЧЕКИХ ИСПЫТАНИЙ

2.1 Равномерно распределенная дискретная случайная величина

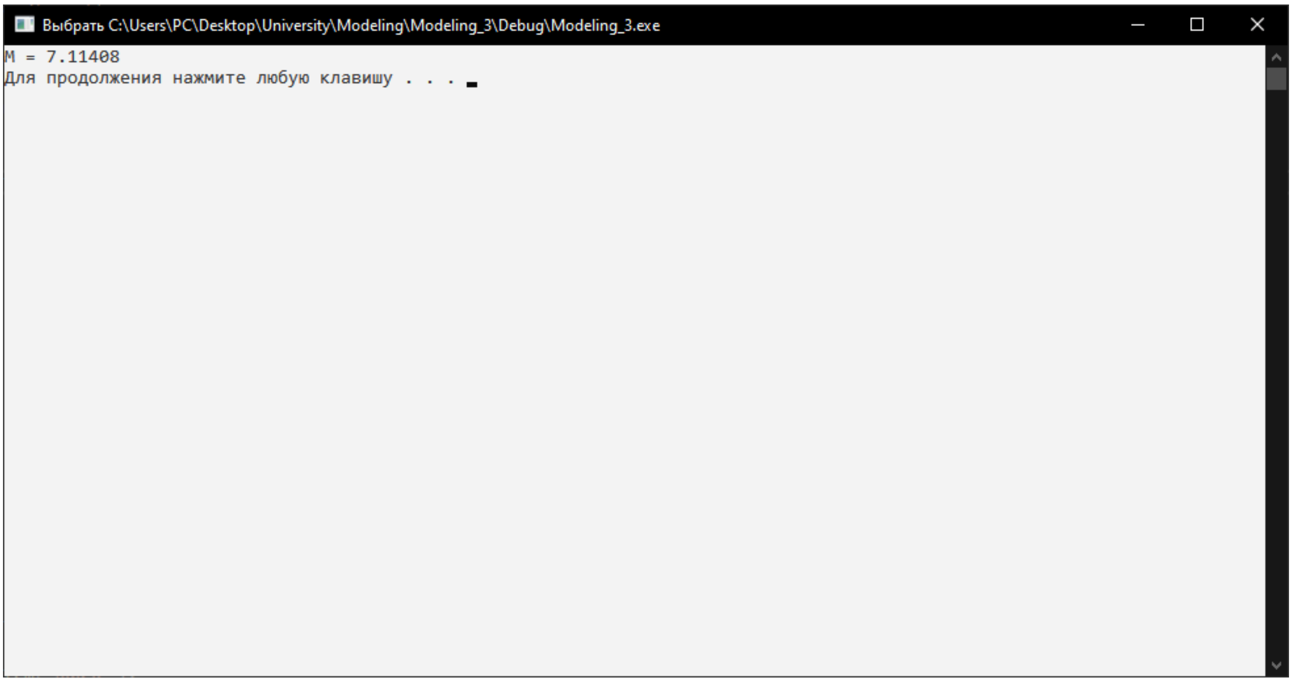
2.1.1 Постановка задачи

В мусорный бак может поместиться 10 пакетов с мусором. В день в бак с равной вероятностью может поступить 0, 1, 2 или 3 пакета. Построить модель поступления пакетов в бак. На основании 10000 испытаний оценить математическое ожидание числа дней до полного заполнения бака.

2.1.2 Код

```
#include <iostream>
#include <vector>
#include <time.h>

using namespace std;

const int n = 1e+5;

int main() {
	vector<int> experiments(1000, 0);
	srand(time(0));

	for (int i = 0; i < n; ++i) {
		int days_counter = 0;
		int space_left = 10;
		do {
			days_counter++;
			space_left -= rand() % 4;
		} while (space_left > 0);

		experiments[days_counter] += 1;
	}

	float M = 0;
	for (int i = 1; i < experiments.size(); ++i)
		M += i * 1.0f * experiments[i] / n;

	cout << "M = " << M << "\n";

	system("pause");
	return 0;
}
```

2.1.3 Результат



2.2 Равномерно распределенная непрерывная случайная величина

2.2.1 Постановка задачи

Проверить, действительно ли датчик случайных чисел можно использовать для формирования псевдослучайных чисел x_i, $i = 1, 2, \ldots,$ (на интервале $[0, 1)$), имеющих равномерную функцию распределения. Для решения задачи необходимо построить пары чисел (x_i, x_{i+1}), $i = 1, 2, \ldots,$ и координаты этих точек отобразить на единичном квадрате. Вывод сделать на основании сначала 50, затем 100 точек, расположенных на единичном квадрате.

2.2.2 Код

```
import matplotlib.pyplot as plt
import numpy as np
import random

xs = []
ys = []
N = int(input())

for i in range(N):
    xs.append(random.random())
    ys.append(random.random())

```

```
plt.scatter(np.array(xs, dtype=float), np.array(ys, dtype=float))
plt.show()
```

2.2.3 Результат

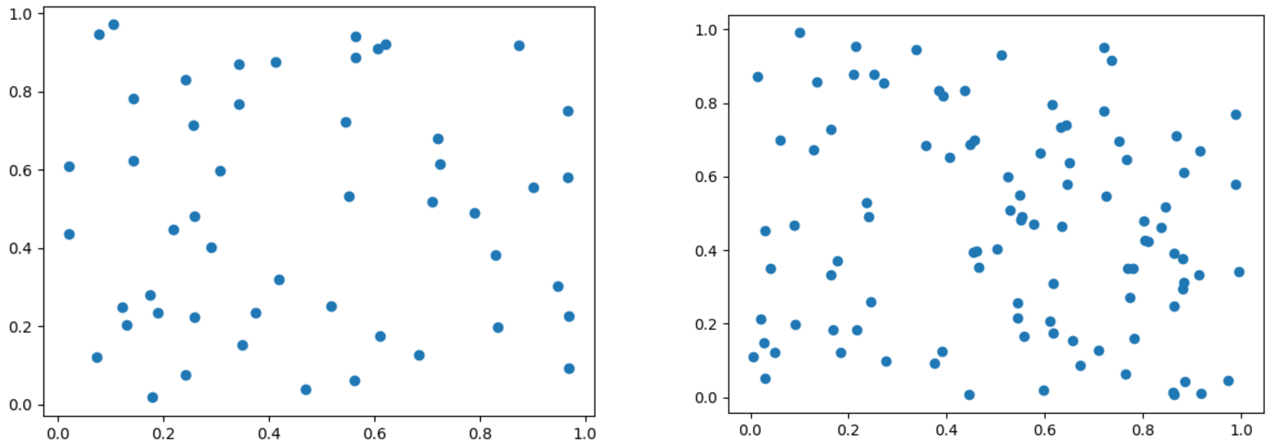


Рисунок 2 – Левый — 50 точек; Правый — 100 точек

2.3 Нормально распределенная случайная величина

2.3.1 Постановка задачи

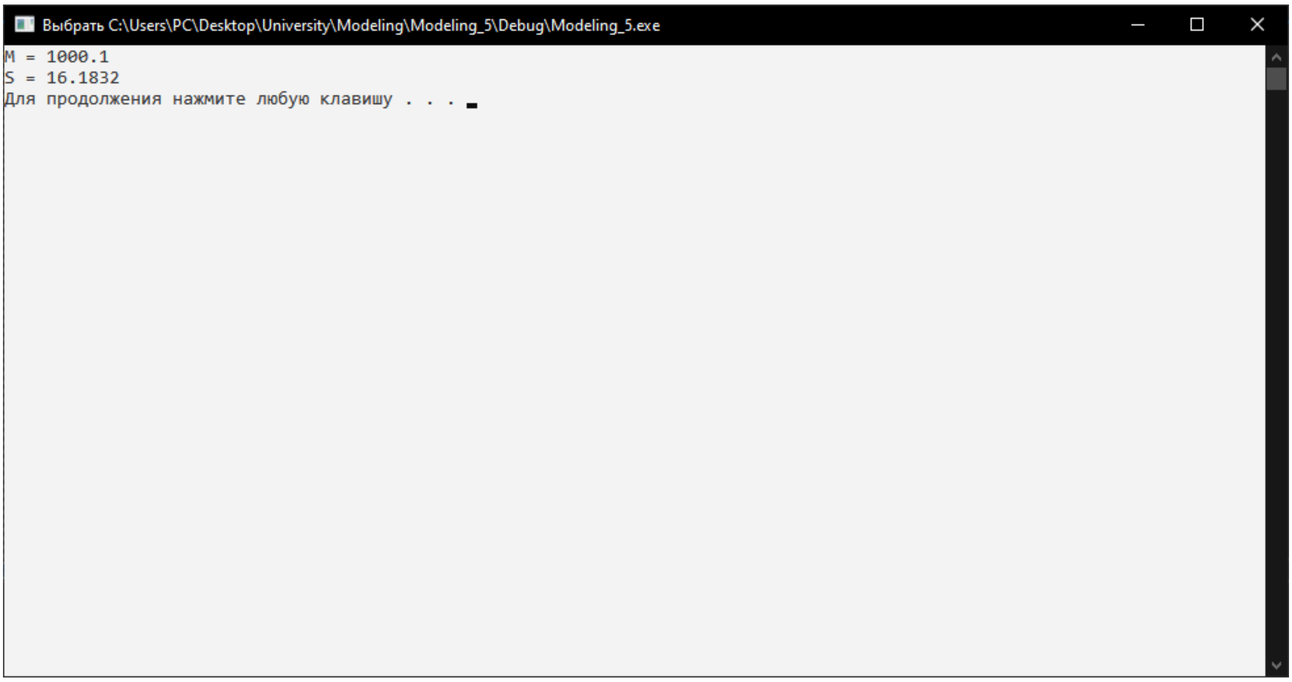
Вес коробки с печеньями есть нормально распределенная случайная величина с параметрами $\mu = 100$ гр. и $\sigma = 5$ гр. Одна упаковка содержит 10 коробок с печеньями. Построить модель формирования веса одной упаковки. Оценить математическое ожидание и среднее квадратическое отклонение веса одной упаковки на основании 1000 испытаний.

2.3.2 Код

```
#include <iostream>
#include <vector>
#include <time.h>

using namespace std;

const int n = 1000;
const int Mu = 100;
const int sigma = 5;

float generate() {
        float R = 0;
```

```
	for (int i = 0; i < 12; ++i)
		R += rand() * 1.f / RAND_MAX;

	return Mu + sigma * (R - 6);
}

int main() {
	srand(time(0));
	vector<float> weights;

	float M = 0;
	for (int i = 0; i < n; ++i) {
		float weight = 0;
		for (int j = 0; j < 10; ++j)
			weight += generate();

		M += weight * 1.0f / n;
		weights.push_back(weight);
	}


	float sqr_sum = 0;
	for (int i = 0; i < n; ++i)
		sqr_sum += (weights[i] - M) * (weights[i] - M);

	float S = sqrt(sqr_sum / (n - 1));

	cout << "M = " << M << "\n";
	cout << "S = " << S << "\n";

	system("pause");
	return 0;
}
```

2.3.3 Результат



2.4 Экспоненциально распределенная случайная величина

2.4.1 Постановка задачи

С момента появления первых признаков неисправности до полного отказа элемента проходит экспоненциально распределенный интервал времени. Математическое ожидание этого интервала равно 1. Через равные интервалы времени (равные 0,5) элемент проходит техническое обслуживание и с вероятностью 0,7 признаки неисправности будут обнаружены и элемент будет заменен. На основании 10000 исходов (замена элемента или его отказ) оценить вероятность того, что неисправность элемента не будет обнаружена при проведении технического обслуживания и этот элемент выйдет из строя.

2.4.2 Код

```
#include <iostream>
#include <time.h>
#include <cmath>

using namespace std;

const float lambda = 1;
const int n = 1e+4;

float generate() {
	return -1.f / lambda * log(rand() * 1.f / RAND_MAX);
}
```

```
int main() {
    srand(time(0));
    float chance = 0;
    float T = 0;

    for (int i = 0; i < n; ++i) {
        float t = generate();
        T += t;
        while (t > 0) {
            t -= 0.5;
            if (rand() % 10 < 7)
                break;
            else if (t < 0)
                chance++;
        }
    }

    chance /= n;

    cout << "Chance = " << chance * 100 << "%\n";

    system("pause");
    return 0;
}
```

2.4.3 Результат