

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Институт информатики и вычислительной техники

09.03.01 "Информатика и вычислительная техника"
профиль "Программное обеспечение средств
вычислительной техники и автоматизированных систем"

Кафедра прикладной математики и кибернетики

Современные технологии программирования

Лабораторная работа №11

Вероятностное моделирование метрических характеристик программ

Выполнил:

студент гр.ИП-213

Дмитриев Антон Александрович
ФИО студента

«__» _____ 2025 г.

Проверил:

Преподаватель

ФИО преподавателя

«__» _____ 2025 г.

Оценка _____

Новосибирск 2025 г.

1. Задание

1. Разработать программу для вероятностного моделирования процесса написания программы программистом с длиной словаря программы $\eta = 16, 32, 64, 128$.
2. С помощью разработанной программы получить статистические оценки:
 - длины программы L ,
 - дисперсии длины $D(L\eta)$,
 - среднеквадратического отклонения ($\sqrt{D(L\eta)}$),
 - относительной ожидаемой погрешности δ .
3. С помощью приведенных формул получить теоретические значения и сравнить их с результатами моделирования.
4. По тексту разработанной программы посчитать длину ее словаря и длину программы. Рассчитать длину программы по размеру ее словаря с помощью приведенных формул. Сравнить посчитанное по тексту значение длины текста программы, с длиной текста программы, полученной по формуле.
5. По первому и второму пунктам задания определить η^*2 – число единых по смыслу входных и выходных параметров представленных в сжатой без избыточной форме. Сравнить прогнозируемую длину программы с длиной программы, рассчитанной по тексту программы.

2. Цель работы

Целью работы является разработка программы, имитирующей процесс написания программы программистом, и расчет статистических оценок метрических характеристик полученных программ. Для имитации используется вероятностная модель выборки с возвратом из генеральной совокупности, состоящей из η символов, пока последняя не будет исчерпана, а также проведение серий испытаний при разных значениях η .

3. Текст программы

Main.py:

```
import numpy as np
import math
import tokenize
import io
import keyword

def get_program_lexemes(code):
    """Возвращает список лексем программы (без комментариев и пробелов)"""
    lexemes = []
    try:
        tokens = tokenize.tokenize(io.BytesIO(code.encode('utf-8')).readline)
        for tok in tokens:
            if tok.type in (tokenize.NAME, tokenize.OP, tokenize.NUMBER, tokenize.STRING):
                lexemes.append(tok.string)
    except Exception as e:
        print(f"Ошибка токенизации: {e}")
    return lexemes

def analyze_program_text():
    """Анализирует текст текущей программы"""
    with open(__file__, 'r', encoding='utf-8') as f:
        code = f.read()

    lexemes = get_program_lexemes(code)
    total_length = len(lexemes)
    unique_lexemes = set(lexemes)
    eta = len(unique_lexemes)
```

```
predicted_length = 0.9 * eta * math.log2(eta) if eta > 0 else 0
```

```
return eta, total_length, predicted_length
```

```
def theoretical_values(eta1, eta2=None):
```

```
    """Теоретические значения для одного или двух словарей"""
```

```
    if eta2 is None or eta2 == 0:
```

```
        if eta1 == 0:
```

```
            return 0, 0, 0
```

```
        M_L = 0.9 * eta1 * math.log2(eta1)
```

```
        D_L = (math.pi**2 / 6) * (eta1**2)
```

```
    else:
```

```
        M_L = 0.9 * (eta1 * math.log2(eta1) + eta2 * math.log2(eta2))
```

```
        D_L = (math.pi**2 / 6) * (eta1**2 + eta2**2)
```

```
    Std_L = math.sqrt(D_L)
```

```
    delta = Std_L / M_L if M_L != 0 else 0
```

```
    return M_L, D_L, Std_L, delta
```

```
def simulate_normal(eta1, eta2=None, runs=10000):
```

```
    """Моделирование L как нормальной случайной величины"""
```

```
    if eta2 is None:
```

```
        eta2 = 0
```

```
M, D, _, _ = theoretical_values(eta1, eta2)
```

```
Std = math.sqrt(D)
```

```
lengths = np.random.normal(M, Std, runs)
```

```
lengths = np.maximum(lengths, 1)
```

```
mean_L = np.mean(lengths)
```

```

var_L = np.var(lengths, ddof=1)
std_L = np.std(lengths, ddof=1)
delta = std_L / mean_L if mean_L != 0 else 0
return mean_L, var_L, std_L, delta

def main():
    # Пункты 1-2
    etas = [16, 32, 64, 128]
    runs = 50000

    print("=*70)
    print("Пункты 1-2: Моделирование для одного словаря")
    print("=*70)
    print("eta | M(L) теор | D(L) теор | sqrt(D) теор | delta теор")
    print("-*70)
    theoretical_results = {}
    for eta in etas:
        M, D, Std, delta = theoretical_values(eta)
        theoretical_results[eta] = (M, D, Std, delta)
        print(f"\n{eta:3d} | {M:10.4f} | {D:10.4f} | {Std:10.4f} | {delta:.4f}\n")

    print("\n" + "=*70)
    print("Моделирование нормальным распределением (runs={})".format(runs))
    print("=*70)
    print("eta | M(L) модел | D(L) модел | sqrt(D) мод | delta мод")
    print("-*70)
    for eta in etas:
        M_sim, D_sim, Std_sim, delta_sim = simulate_normal(eta, runs=runs)
        print(f"\n{eta:3d} | {M_sim:10.4f} | {D_sim:10.4f} | {Std_sim:10.4f} | {delta_sim:.4f}\n")

```

```

# Пункт 4

print("\n" + "="*70)

eta_prog, L_real, L_pred = analyze_program_text()

print(f"Длина словаря программы η = {eta_prog}")

print(f"Реальная длина программы L = {L_real} лексем")

print(f"Прогнозируемая длина по формуле M(L) = 0.9η log2 η = {L_pred:.2f}")

print(f"Отклонение: {abs(L_real - L_pred) / L_real * 100:.2f}%")

```

```

# Пункт 5

print("\n" + "="*70)

# Возможно, n2 = η (или n2 = η1 + η2) — уточнить из контекста

# Если n2 — число входных/выходных параметров, то можно связать с η через D(L)

# D(L) = π2η2/6 => η = sqrt(6*D/π2)

# Но в нашем моделировании D известна теоретически

# Для программы:

M_prog, D_prog, Std_prog, delta_prog = theoretical_values(eta_prog)

print(f"Для программы с η={eta_prog}:")

print(f"M(L)={M_prog:.2f}, D(L)={D_prog:.2f}")

# Если n2 = η (гипотеза), то:

n_squared = eta_prog

print(f"n2 (предположительно размер словаря) = {n_squared}")

print(f"Прогнозируемая длина по модели: {M_prog:.2f}")

print(f"Реальная длина: {L_real}")

print(f"Сравнение: прогноз/реальность = {M_prog/L_real:.2%}")

if __name__ == "__main__":
    main()

```

4. Статистические и расчетные характеристики длин программ для заданных размеров словарей

Моделирование для одного словаря					
eta	M(L) теор	D(L) теор	sqrt(D) теор	delta теор	
16	57.6000	421.1031	20.5208	0.3563	
32	144.0000	1684.4125	41.0416	0.2850	
64	345.6000	6737.6499	82.0832	0.2375	
128	806.4000	26950.5998	164.1664	0.2036	

Моделирование нормальным распределением (runs=50000)					
eta	M(L) модел	D(L) модел	sqrt(D) мод	delta мод	
16	57.5335	423.6256	20.5822	0.3577	
32	144.0879	1692.1264	41.1355	0.2855	
64	345.6152	6712.3329	81.9288	0.2371	
128	807.2386	26985.5912	164.2729	0.2035	

5. Рассчитанную и полученную по тексту длину разработанной программы.

=====

Длина словаря программы $\eta = 150$
Реальная длина программы $L = 652$ лексем
Прогнозируемая длина по формуле $M(L) = 0.9\eta \log_2 \eta = 975.89$
Отклонение: 49.68%

=====

Для программы с $\eta=150$:
 $M(L)=975.89$, $D(L)=37011.02$
 n^2 (предположительно размер словаря) = 150
Прогнозируемая длина по модели: 975.89
Реальная длина: 652
Сравнение: прогноз/реальность = 149.68%

=====