Министерство цифрового развития, связи и массовых коммуникаций

Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования «Сибирский государственный университет

телекоммуникаций и информатики» (СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа

«Вероятностное моделирование метрических характеристик программ»

Выполнил:

Студент группы ИП-017

Костин А.В.

Работу проверил:

ассистент кафедры ПМиК

Агалаков А.А.

Новосибирск 2023 г.

**Содержание**

**Оглавление**

[**1.** **Задание** 3](#_Toc151914281)

[**2.** **Расчёт значений программы** 4](#_Toc151914282)

[**3.** **Результаты выполнения** 5](#_Toc151914283)

[**4. Исходные тексты программ на языке C#** 6](#_Toc151914284)

1. **Задание**

1. Разработать программу для вероятностного моделирования процесса написания программы программистом с длиной словаря программы η =16, 32, 64, 128.

2. С помощью разработанной программы получить статистические оценки: • длины программы L,

• дисперсии длины D(Lη),

• среднеквадратического отклонения (√ D(Lη),

• 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.

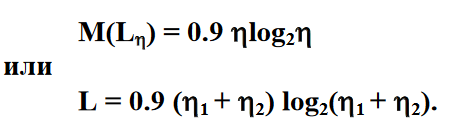
5. По первому и второму пунктам задания определить η \* 2 – число единых по смыслу входных и выходных параметров представленных в сжатой без избыточной формы. Сравнить прогнозируемую длину программы с длиной программы, рассчитанной по тексту программы.

1. **Расчёт значений программы**

Пусть n1 – число операторов, а n2 – число операндов. Величину n = n1 + n2 назовем словарем программы, она равна числу «билетов». Имя первого слова из словаря (оператора или операнда) входит в выборку при первом извлечении. Для того чтобы в выборку вошел следующий элемент, отличный от первого, может потребоваться некоторое число извлечений Lj. Вообще, пусть Lr будет числом извлечений, следующих за выбором r –того билета до выбора нового (r + 1) го билета включительно. Тогда



Является объемом выборки на тот момент, когда в нее входит r – ый новый оператор или операнд. Распределение случайной величины Lr совпадает с распределением первого успеха в последовательности испытаний Бернули. Исходя из этого, за математическое ожидание длины программы с длиной словаря n приближенно можно принять следующее выражение:



Последнее выражение и принимается за математическое ожидание длины программы, выраженное в количестве операторов и операндов. Можно показать, что дисперсия

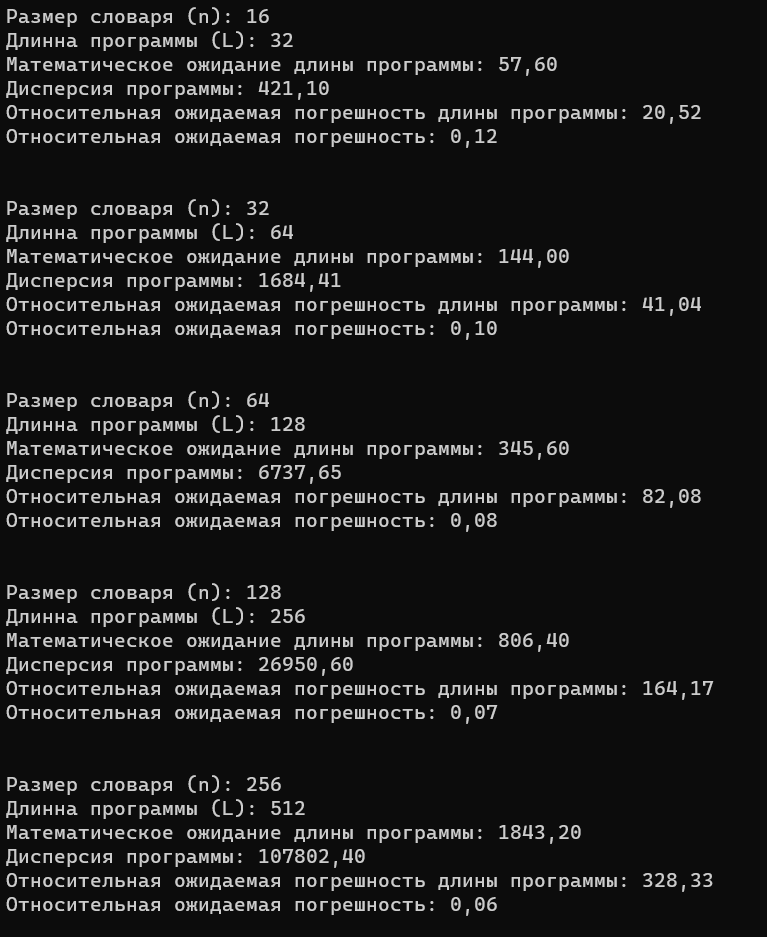
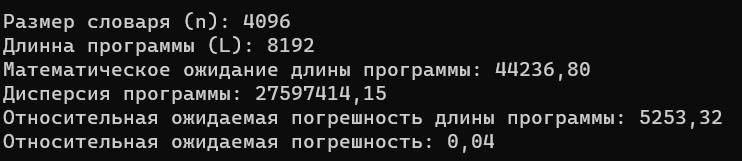


Тогда относительная ожидаемая погрешность будет



Если один и тот же алгоритм поручить реализовать на каком-либо языке нескольким программистам, то написанные ими программы будут, конечно, варьировать по длине. Но, как следует из приведенных выше соотношений, а также из огромного числа статистических проверок, сделанных на основе уже написанных программ, эти отклонения невелики. Практика показывает, что б обычно не превосходит 10%. Так как функция L - выпуклая, то имеет место неравенство Иенсена, исходя из которого можно получить еще одно приближенное выражение для длины программы:

1. **Результаты выполнения**

**** ****

**4. Исходные тексты программ на языке C#**

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

namespace lab15

{

class Program

{

static Random random = new Random();

static List<string> GenerateVocabulary(int dictionarySize)

{

var vocabulary = new List<string>();

for (int i = 0; i < dictionarySize; i++)

{

vocabulary.Add("op" + i);

vocabulary.Add("operand" + i);

}

return vocabulary;

}

static Tuple<List<string>, int> SimulateProgramGeneration(int dictionarySize)

{

var vocabulary = GenerateVocabulary(dictionarySize);

var program = new List<string>();

while (vocabulary.Count > 0)

{

int index = random.Next(vocabulary.Count);

string selected = vocabulary[index];

program.Add(selected);

vocabulary.RemoveAt(index);

}

int programLength = program.Count;

return Tuple.Create(program, programLength);

}

static void Main(string[] args)

{

int[] dictionarySizes = { 16, 32, 64, 128, 256, 4096};

foreach (int dictionarySize in dictionarySizes)

{

var result = SimulateProgramGeneration(dictionarySize);

Console.WriteLine();

var program = result.Item1;

int programLength = result.Item2;

double expectedLength = 0.9 \* dictionarySize \* Math.Log2(dictionarySize);

double variance = (Math.PI \* Math.PI \* dictionarySize \* dictionarySize) / 6;

double standardDeviation = Math.Sqrt(variance);

double relativeError = 1.0 / (2 \* Math.Log2(dictionarySize));

Console.WriteLine($"Размер словаря (n): {dictionarySize}");

Console.WriteLine($"Длинна программы (L): {programLength}");

Console.WriteLine($"Математическое ожидание длины программы: {expectedLength:F2}");

Console.WriteLine($"Дисперсия программы: {variance:F2}");

Console.WriteLine($"Относительная ожидаемая погрешность длины программы: {standardDeviation:F2}");

Console.WriteLine($"Относительная ожидаемая погрешность: {relativeError:F2}\n");

}

}

}

}