МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 8**

**по дисциплине  
 «АЛГОРИТМЫ И АНАЛИЗ СЛОЖНОСТИ»**

Выполнил студент группы 25/2                                       А.А. Козин

Направление подготовки  02.03.03  Математическое обеспечение и администрирование информационных систем

Курс    2

Отчет принял доктор физико-математических наук, профессор                                                                                       А.И. Миков

Краснодар

2021 г.

**Задание:** найти сложность алгоритма решения КНФ методом резолюций

**Решение**.

Для анализа сложности алгоритма написана программа, которая случайно задает несколько массивов размера от 1 до 25 и считывает сложность их перемножения и записи в третий массив.

**Текст программы:**

#include <iostream>

#include <vector>

#include <ctime>

using namespace std;

int counter = 0;

string equationGenerator(int n)

{

int i, j;

string meow = "";

char x1, x2;

for (i = 0; i < n; i++)

{

meow += "(";

if (rand() % 2 == 0) meow += "-";

meow += "x";

x1 = rand() % 6 + 49;

meow += x1;

meow += "+x";

x2 = x1;

while (x1 == x2)x2 = rand() % 6 + 49;

meow += x2;

meow += ")\*";

}

meow.erase(meow.size() - 1, 1);

return meow;

}

string unionString(string A, string B)

{

int i, pos, posA, posNew;

string temp = "", newString = "";

for (i = 0; i < A.size(); i++)

{

if (A[i] == 'x')

{

temp += A[i];

temp += A[i + 1];

counter += 2;

pos = B.find(temp);

if (pos != -1)

{

if (pos == 0 && i > 0 && A[i - 1] != '-')

{

newString += temp;

newString += "+";

counter += 2;

}

if (pos == 0 && i == 0)

{

newString += temp;

newString += "+";

counter += 2;

}

if (pos != 0 && B[pos - 1] != '-' && i > 0 && A[i - 1] != '-')

{

newString += temp;

newString += "+";

counter += 2;

}

if (pos != 0 && B[pos - 1] == '-' && i > 0 && A[i - 1] == '-')

{

newString += "-";

newString += temp;

newString += "+";

counter += 3;

}

}

else

{

temp += "+";

if (i > 0 && A[i - 1] == '-')

{

newString += "-";

counter += 1;

}

newString += temp;

counter += 2;

}

}

temp = "";

}

for (i = 0; i < B.size(); i++)

{

if (B[i] == 'x')

{

temp += B[i];

temp += B[i + 1];

counter += 2;

posA = A.find(temp);

posNew = newString.find(temp);

if (posA == -1 && posNew == -1)

{

temp += "+";

if (i > 0 && B[i - 1] == '-')

{

newString += "-";

counter += 1;

}

newString += temp;

counter += 2;

}

}

temp = "";

}

if (newString.empty() != true)

newString.erase(newString.size() - 1, 1);

return newString;

}

int main()

{

srand(time(0));

string full, temp = "";

int i = 0, j, prov = 0, U, prov2 = 0;

for (U = 2; U <= 10; U++)

{

for (int middle = 1; middle <= 100; middle++)

{

full = equationGenerator(U);

vector<string>split;

while (i < full.size())

{

if (full[i] != '\*')

{

if (full[i] != '(' && full[i] != ')')temp += full[i];

}

else

{

split.push\_back(temp);

temp = "";

}

i++;

}

split.push\_back(temp);

temp = "";

for (i = 0; i < split.size(); i++)

{

for (j = i + 1; j < split.size(); j++)

{

temp = unionString(split[i], split[j]);

if (temp == "")break;

for (int k = 0; k < split.size(); k++) if (temp == split[k])prov = 1;

if (prov == 0)split.push\_back(temp);

}

if (temp == "")break;

prov = 0;

}

}

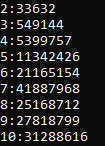
cout << U << ":" << counter << endl;

counter = 0;

}

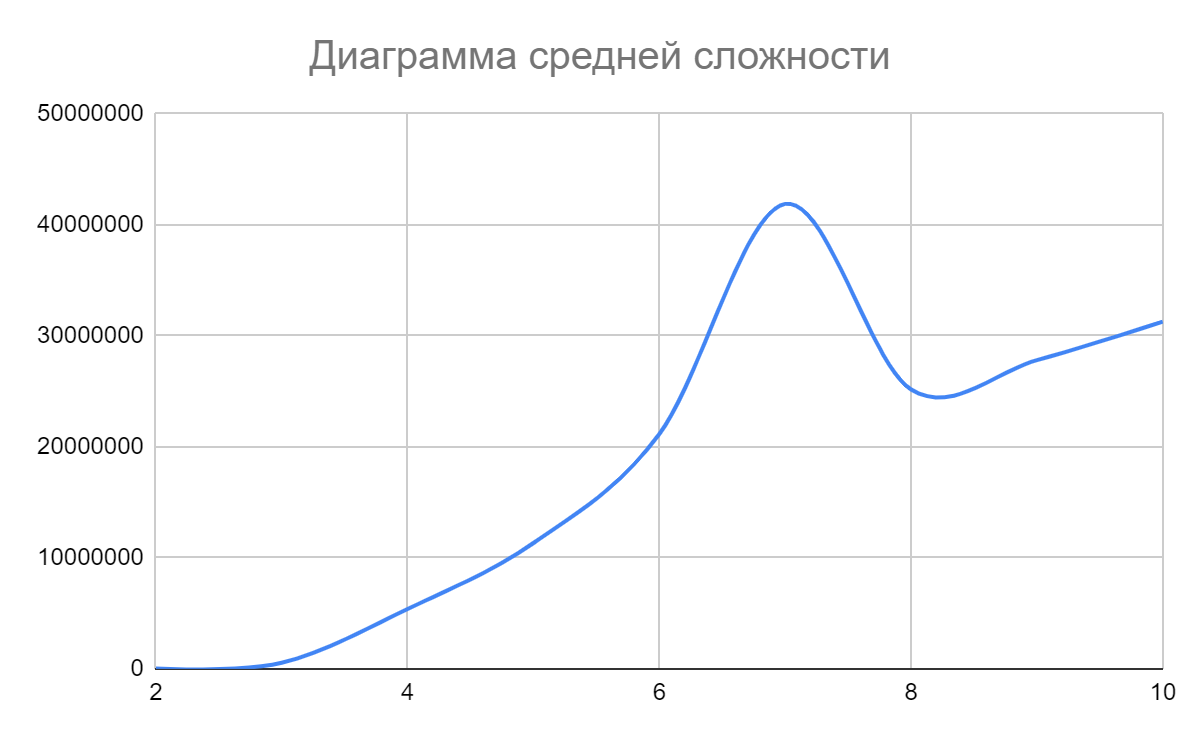
return 0;

}**Программа выдает на консоль результаты в следующем виде:**



(Слева – количество дизъюнктов, справа – сложность операции (проводится 100 раз))

График роста количества операций для сложности исходных данных *n* в диапазоне от 1 до 10 выглядит следующим образом:



**Анализ зависимости:**

Скачок на графике обусловлен тем, что на 7 дизъюнктах количество неизвестных переменных выше, а шанс вывести пустой элемент (из-за малого количества дизъюнктов) меньше, поэтому после преодоления порога в 7 график идёт на снижение (количество дизъюнктов увеличивается, а с ними увеличивается шанс вывести пустой элемент).