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 № 5**

**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 среднее количество коллизий при хешировании линейным способом и случайным.

**Решение**.

Для нахождения количества коллизий напишем программу хеширования:

**Текст программы:**

#include <iostream>

#include <ctime>

#include <string>

using namespace std;

int counter = 0;

struct HeshTable

{

string word = "#";

int indexOfNext;

};

string new\_word(int n)

{

string res, q;

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

{

q = 97 + (rand() % 26);

res += q;

}

return res;

}

void hesh\_linear\_add(HeshTable\* table, string word, int n)

{

int index = 0, next = table->indexOfNext;

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

index += word[i] < 100 ? word[i] % 10 : word[i] % 100;

if (table[index].word == "#")

table[index].word = word;

else {

while (table[index].word != "#")

{

if (index + 1 >= 299)

index = 0;

counter++;

index++;

}

table[index].word = word;

}

}

void hesh\_random\_add(HeshTable\* table, string word, int n)

{

int index = 0, next = table->indexOfNext, newIndex = 0;

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

index += word[i] < 100 ? word[i] % 10 : word[i] % 100;

if (table[index].word == "#")

table[index].word = word;

else {

while (table[index].word != "#")

{

newIndex = rand() % 300;

table[index].indexOfNext = newIndex;

index = newIndex;

counter++;

}

table[index].word = word;

}

}

void empty\_table(HeshTable\* table) {

for (int i = 0; i < 300; i++) table[i].word = "#";

}

void main()

{

srand(time(0));

setlocale(LC\_ALL, "RU");

int n, i = 0, j = 0;

double mean = 0;

string word;

HeshTable\* table\_linear = new HeshTable[300];

HeshTable\* table\_random = new HeshTable[300];

cout << "Cреднее линейное значение коллизий: " << endl;

for (int k = 10; k < 300; k += 10)

{

for (int j = 1; j <= 1000; j++)

{

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

{

n = rand() % 14 + 2;

word = new\_word(n);

hesh\_linear\_add(table\_linear, word, n);

}

counter = 0;

n = rand() % 14 + 2;

word = new\_word(n);

hesh\_linear\_add(table\_linear, word, n);

mean += counter;

empty\_table(table\_linear);

}

cout << k << " : " << mean / 1000 << endl;

mean = 0;

}

cout << "Cреднее случайное значение коллизий: " << endl;

for (int k = 10; k < 300; k += 10)

{

for (int j = 1; j <= 1000; j++) {

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

{

n = rand() % 14 + 2;

word = new\_word(n);

hesh\_random\_add(table\_random, word, n);

}

counter = 0;

n = rand() % 14 + 2;

word = new\_word(n);

hesh\_random\_add(table\_random, word, n);

mean += counter;

empty\_table(table\_random);

}

cout << k << " : " << mean / 1000 << endl;

mean = 0;

}

system("pause>nul");

}

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

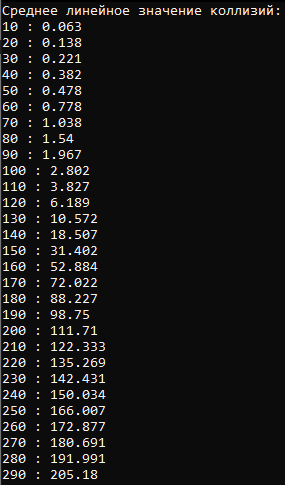
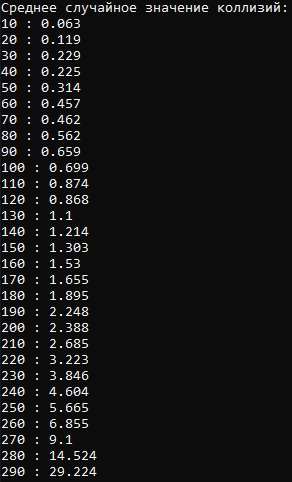


График роста количества операций для сложности линейной функции и случайной функции:

**Вывод**: Линейная функция в среднем возрастает пропорционально значению самой величины или имеет экспоненциальную зависимость, а случайная имеет степенную, поэтому алгоритм случайного задания эффективнее чем алгоритм линейного задания.