**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Математической кибернетики и информационных технологий

**Отчет по лабораторной работе №2**

по дисциплине «ТЯП»

на тему:

«Таблицы идентификаторов»

Выполнил: студент группы БВТ1801

Клюшкин Дмитрий Алексеевич

Руководитель:

Кутейников Иван Алексеевич

Москва 2019

**Цель работы:**

Написать программу, которая получает на входе набор идентификаторов, организует таблицу по заданному методу и позволяет осуществить многократный поиск и добавление идентификатора в этой таблице. Список идентификаторов задан в виде текстового файла. Длина идентификаторов ограничена 32 символами.

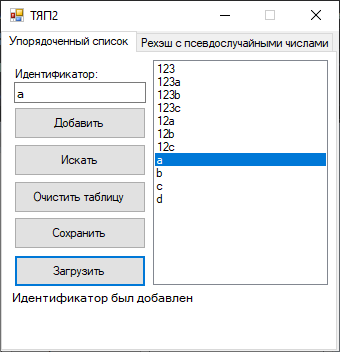
Будем считать, что прописные и строчные буквы в идентификаторах различны. В качестве кодов символов возьмём коды таблицы ASCII. Тогда, если положить, что строка из области определения хэш-функции содержит только цифры и буквы английского алфавита, то минимальным значением хэшфункции будет сумма трёх кодов цифры «0», а максимальным значением – сумма трёх кодов литеры «z». Таким образом, область значений выбранной хэш-функции может быть описана как: '0' + '0' + '0' … 'z' + 'z' + 'z' Диапазон области значений составляет 223 элемента. Длина входных идентификаторов в данном случае ничем не ограничена. Для удобства пользования опишем две константы, задающие границы области значений хэшфункции: HASH\_MIN = '0' + '0' + '0'; HASH\_MAX = 'z' + 'z' + 'z'. Сама хэш-функция без учёта рехэширования будет вычислять следующее выражение: HASH = sName[0] + sName[Length(sName) / 2]) + sName[Length(sName) – 1], где sName – это входная строка (идентификатор). Для рехэширования возьмём простейший генератор последовательности псевдослучайных чисел, построенный на основе формулы F= i · H1 mod H2, где H1 и H2 – простые числа, выбранные таким образом, чтобы H1 было в диапазоне от Н2 / 2 до Н2. Причём, чтобы этот генератор выдавал максимально длинную последовательность во всём диапазоне от HASH\_MIN до HASH\_MAX, Н2 должно быть максимально близко к величине HASH\_MAX – HASH\_MIN + 1. В данном случае диапазон содержит 223 элемента, и поскольку 223 – простое число, то возьмём Н2 = 223 (если бы размер диапазона не был простым числом, то в качестве Н2 нужно было бы взять ближайшее к нему меньшее простое число). В качестве Н1 возьмём 127. Опишем соответствующие константы:

REHASH1 = 127; REHASH2 = 223. Тогда хэш-функция с учётом рехэширования будет иметь следующий

вид: Result = (HASH – HASH\_MIN + iNum \* REHASH1 mod REHASH2) mod (HASH\_MAX – HASH\_MIN + l) + HASH\_MIN; if Result < HASH\_MIN Result = HASH\_MIN; Входные параметры этой функции: iNum – индекс рехэширования (если iNum = 0, то рехэширование отсутствует). Строка проверки величины результата (Result < HASH\_MIN) добавлена, чтобы исключить ошибки в тех случаях, когда на вход функции подаётся строка, содержащая символы вне диапазона '0'...'z' (если контроль входных идентификаторов отсутствует, это имеет смысл)

**Ход работы:**

*Упорядоченный список:*



*Код программы:*

* Добавить

String^ Input = textBox2->Text;

int length = Input->Length;

if (length == 0)return;

int i = 0;

bool flag;

for (i = 0; i < listBox3->Items->Count; i++) {

if (String::Compare(Convert::ToString(listBox3->Items[i]), Input)==0) {

label4->Text = "Идентификатор уже существует";

listBox3->SelectedIndex = i;

return;

}

if (String::Compare(Convert::ToString(listBox3->Items[i]), Input) > 0) {

listBox3->Items->Insert(i, Input);

flag = true;

break;

}

}

if (!flag) listBox3->Items->Insert(i, Input);

listBox3->SelectedIndex = i;

label4->Text = "Идентификатор был добавлен";

* Поиск

String^ Input = textBox2->Text;

int length = Input->Length;

if (length == 0)return;

int a=0;

int b=listBox3->Items->Count-1;

bool flag;

do{

int pas = String::Compare(Convert::ToString(listBox3->Items[(b + a) / 2]), Input);

if (b - a == 1) {

if (pas >= 0)b = a;

else a = b;

}

pas = String::Compare(Convert::ToString(listBox3->Items[(b + a) / 2]), Input);

if (pas == 0) {

label4->Text = "Индентификатор был найден";

listBox3->SelectedIndex = (b + a) / 2;

flag = true;

break;

}

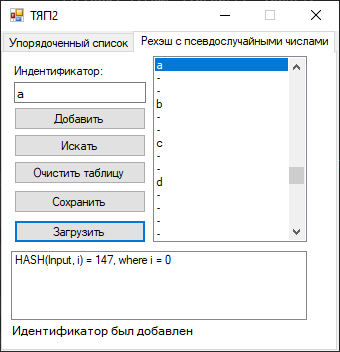
if (pas > 0) b = (b + a) / 2;

else a = (b + a) / 2;

} while (b-a > 0);

if (!flag)label4->Text = "Такого индентификатора нет";

*Рехэш с псевдослучайными числами:*



*Код программы:*

* Добавить

listBox2->Items->Clear();

String^ Input = textBox1->Text;

int length = Input->Length;

if (length == 0)return;

for (int i = 0; i < SIZE; i++) {

int temp = Hash(Input[0], Input[int(length / 2)], Input[length - 1], i);

listBox2->Items->Add("HASH(Input, i) = " + temp + ", where i = " + i);

if (Convert::ToString(listBox1->Items[temp]) == "-") {

listBox1->Items[temp] = Input;

label1->Text = "Идентификатор был добавлен";

listBox1->SelectedIndex = temp;

break;

}

else if (Convert::ToString(listBox1->Items[temp]) == Input) {

label1->Text = "Идентификатор уже существует";

listBox1->SelectedIndex = temp;

break;

}

else if (i == SIZE - 1)label1->Text = "Таблица уже полная";

}

* Поиск

listBox2->Items->Clear();

String^ Input = textBox1->Text;

int length = Input->Length;

if (length == 0)return;

for (int i = 0; i < SIZE; i++) {

int temp = Hash(Input[0], Input[int(length / 2)], Input[length - 1], i);

listBox2->Items->Add("HASH(Input, i) = " + temp + ", where i = " + i);

if (Convert::ToString(listBox1->Items[temp]) == Input) {

label1->Text = "Идентификатор был найден";

listBox1->SelectedIndex = temp;

break;

}

else if (Convert::ToString(listBox1->Items[temp]) == "-" || i == SIZE - 1) {

label1->Text = "Такого идентификатора нет";

break;

}

}

**Вывод:**

Написали программу, которая получает на входе набор идентификаторов, организует таблицу по заданному методу и позволяет осуществить многократный поиск и добавление идентификатора в этой таблице. Список идентификаторов задан в виде текстового файла. Длина идентификаторов ограничена 32 символами.