Практическая работа № 9

ХЕШИРОВАНИЕ

Вариант-22Ф

Постановка задачи

Разработайте приложение, которое использует хеш-таблицу для организации прямого доступа к элементам множества, реализованного на массиве, структура элементов которого приведена в варианте.

Открытая адресация, Счет в банке: номер счета целое семизначное число, ФИО, адрес

1. Описание алгоритма

Алгоритм программы состоит из функции main и вызываемых в ней вспомогательных функций:

- **int get_hash_code** – функция генерации хеш-значения.
- **void add_cell** – функция добавления счета в таблицу.
- **void show_table** – функция вывода таблицы.
- **void find_cell** – функция поиска по владельцу.
- **void delete_cell** – функция удаления записи из таблицы.

В массиве H хранятся сами пары ключ-значение. Алгоритм вставки элемента проверяет ячейки массива H в некотором порядке до тех пор, пока не будет найдена первая свободная ячейка, в которую и будет записан новый элемент. Этот порядок вычисляется на лету, что позволяет сэкономить на памяти для указателей, требующихся в хеш-таблицах с цепочками. Последовательность, в которой просматриваются ячейки хеш-таблицы, называется последовательностью проб. В общем случае, она зависит только от ключа элемента, то есть это последовательность h0(x), h1(x), …, hn — 1(x), где x — ключ элемента, а hi(x) — произвольные функции, сопоставляющие каждому ключу ячейку в хеш-таблице. Первый элемент в последовательности, как правило, равен значению некоторой хеш-функции от ключа, а остальные считаются от него одним из приведённых ниже способов. Для успешной работы алгоритмов поиска последовательность проб должна быть такой, чтобы все ячейки хеш-таблицы оказались просмотренными ровно по одному разу. Алгоритм поиска просматривает ячейки хеш-таблицы в том же самом порядке, что и при вставке, до тех пор, пока не найдется либо элемент с искомым ключом, либо пока не будет достигнут конец списка. Ошибочно

представление, что следует искать до первой свободной ячейки, так как возможно, что элемент из этой ячейки был удален, а искомый элемент находится далее.

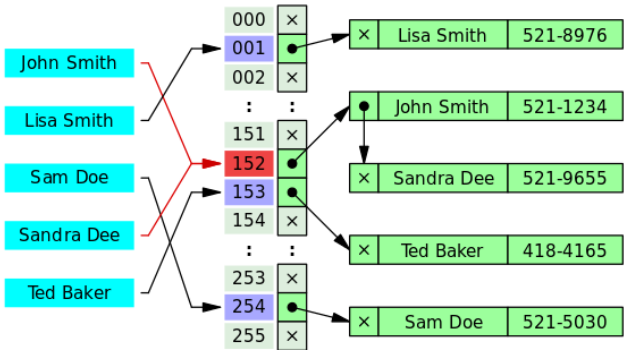

Рис.1 Схема алгоритма открытой адресации

Функция main создает объект класса Table и вызывает меню.

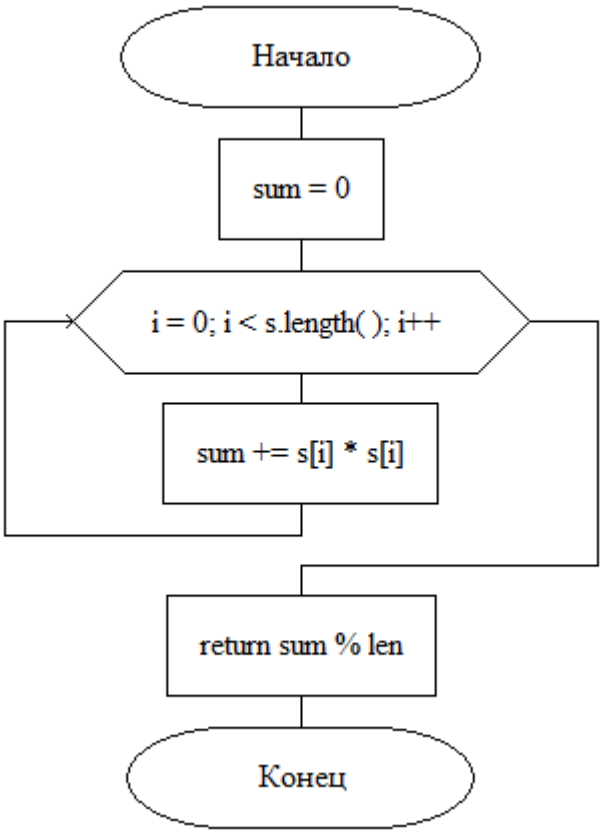

Рис.2 Схема алгоритма функции get_hash_code

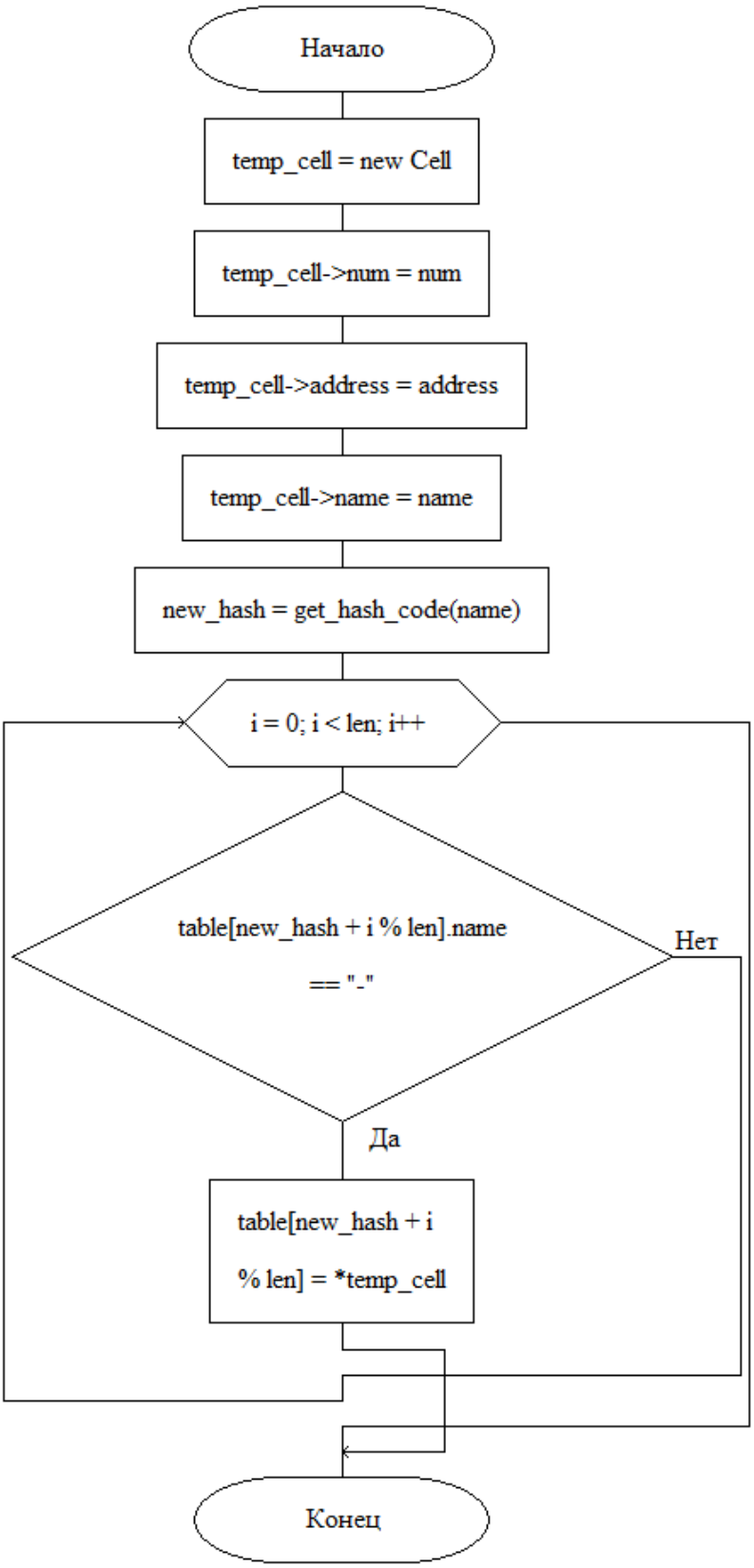

Рис.3 Схема алгоритма функции add_cell

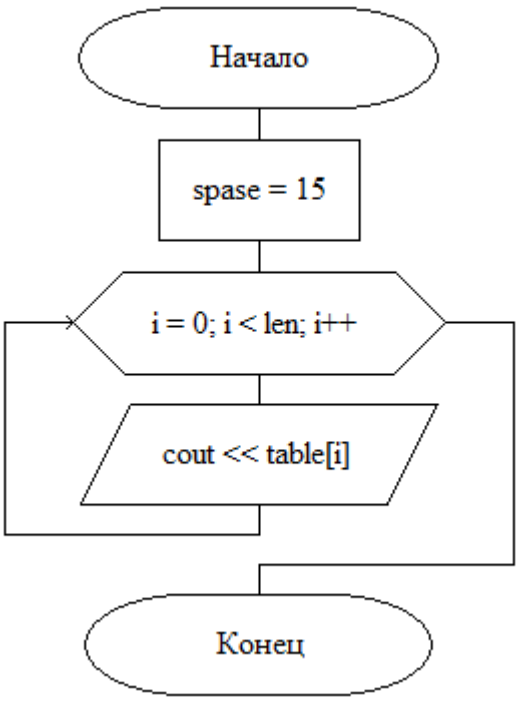

Рис.4 Схема алгоритма функции show_table

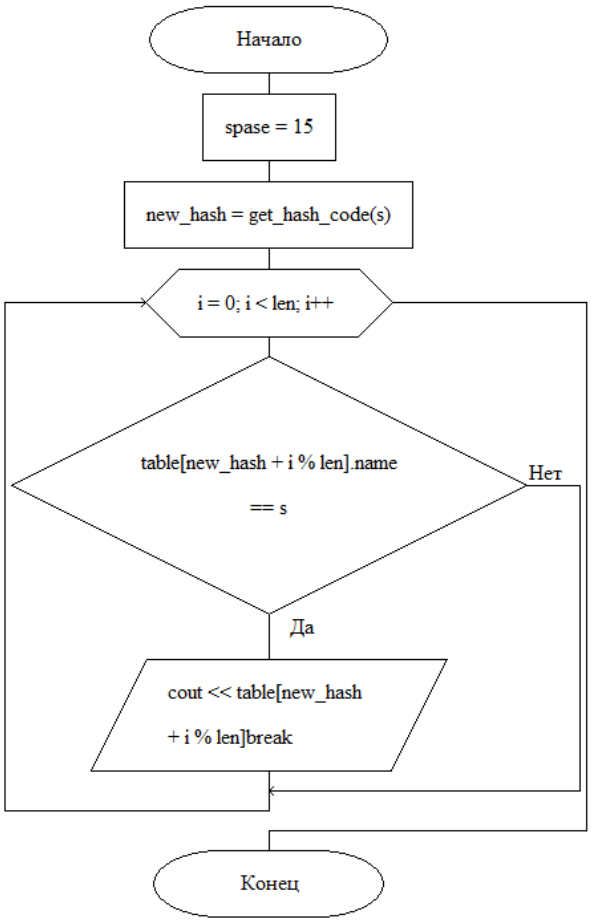

Рис.5 Схема алгоритма функции find_cell

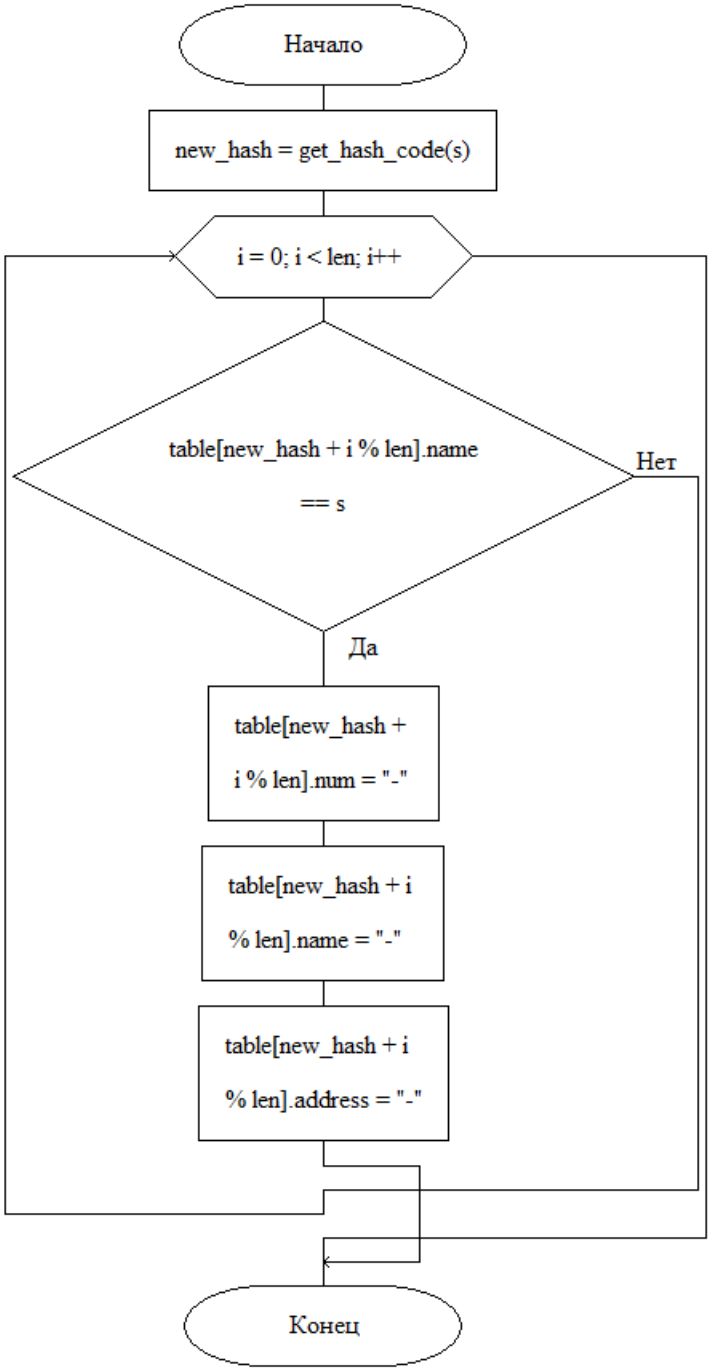

Рис.6 Схема алгоритма функции delete_cell

Реализация алгоритма

Текст исходного кода программы

main.cpp

```
#include "Table.h"
void menu(Table table) {
	cout << "Выберите команду:" << endl;
	cout << "[1] - Добавить счет." << endl;
	cout << "[2] - Удалить счет." << endl;
	cout << "[3] - Найти счет по владельцу." << endl;
	cout << "[4] - Вывести тыблицу." << endl;
	cout << "[5] - Завершить программу." << endl;
	cout << "---->";
	int ch = 0;
	cin >> ch;
	if (ch == 1) {
		cout << "Введите номер счета, имя владельца и адресс через пробелы" <<
endl;
		string num, name, address;
		cin >> num >> name >> address;
		table.add_cell(num, name, address);
		menu(table);
	}
	else if (ch == 2) {
		cout << "Введите имя владельца счета" << endl;
		string name;
		cin >> name;
		table.delete_cell(name);
		menu(table);
	}

	else if (ch == 3) {
		cout << "Введите имя владельца счета" << endl;
		string name;
		cin >> name;
		table.find_cell(name);
		menu(table);
	}
	else if (ch == 4) {
		table.show_table();
		menu(table);
	}
	else if (ch == 5) {
		cout << "Программа завершена" << endl;
	}

}
int main()
{
	setlocale(LC_ALL, "Russian");
	int size;
```

```
	cout << "Введите размер таблицы" << endl;
	cin >> size;
	Table table(size);
	menu(table);}
```

Table.h

```
#pragma once
#include <iostream>
#include <iomanip>
using namespace std;
struct Cell {
	string num = "-";
	string name = "-";
	string address = "-";
};
class Table
{
private:

	Cell* table;
	int len;
public:
	Table(int size);
	int get_hash_code(string s);
	void find_cell(string s);
	void add_cell(string num, string name, string adress);
	void show_table();
	void delete_cell(string s);
};
```

Table.cpp

```
#include "Table.h"
Table::Table(int size) {
	len = size;
	table = new Cell[size];
}

int Table::get_hash_code(string s) {
	int sum = 0;
	for (int i = 0; i < s.length(); i++) {
		sum += s[i]*s[i];
	}
	return sum % len;
}

void Table::add_cell(string num, string name, string address) {
	Cell* temp_cell = new Cell;
	temp_cell->num = num;
	temp_cell->address = address;
	temp_cell->name = name;

	int new_hash = get_hash_code(name);
	for (int i = 0; i < len; i++) {
		if (table[new_hash + i % len].name == "-") {
			table[new_hash + i % len] = *temp_cell;

			break;
		}
		if (i == len - 1)	cout << "Свободных ячеек больше нет" << endl;
	}
```

```
}

void Table::show_table() {
	int spase = 15;
	cout << setw(spase) << "№ ячейки" << setw(spase)
		<< "Номер счета" << setw(spase)
		<< "Имя" << setw(spase)
		<< "Адресс" << endl;
	for (int i = 0; i < len; i++) {
		cout << setw(spase) <<i<<setw(spase)
			<< table[i].num << setw(spase)
			<< table[i].name << setw(spase)
			<< table[i].address << endl;
	}
}

void Table::find_cell(string s) {
	int spase = 15;
	int new_hash = get_hash_code(s);
	for (int i = 0; i < len; i++) {
		if (table[new_hash + i % len].name == s) {
			cout << setw(spase) << "№ ячейки" << setw(spase)
				<< "Номер счета" << setw(spase)
				<< "Имя" << setw(spase)
				<< "Адресс" << endl;
			cout << setw(spase) << new_hash + i % len << setw(spase)
				<< table[new_hash + i % len].num << setw(spase)
				<< table[new_hash + i % len].name << setw(spase)
				<< table[new_hash + i % len].address << endl;

			break;
		}
		if (i == len - 1)	cout << "Счет не найден" << endl;
	}


}

void Table::delete_cell(string s) {

	int new_hash = get_hash_code(s);
	for (int i = 0; i < len; i++) {
		if (table[new_hash + i % len].name == s) {
			table[new_hash + i % len].num = "-";
			table[new_hash + i % len].name = "-";
			table[new_hash + i % len].address = "-";
			break;
		}
		if (i == len - 1)	cout << "Счет не найден" << endl;
	}

}
```

2. Тестирование программы

Ниже представлен результат работы программы с бинарным файлом

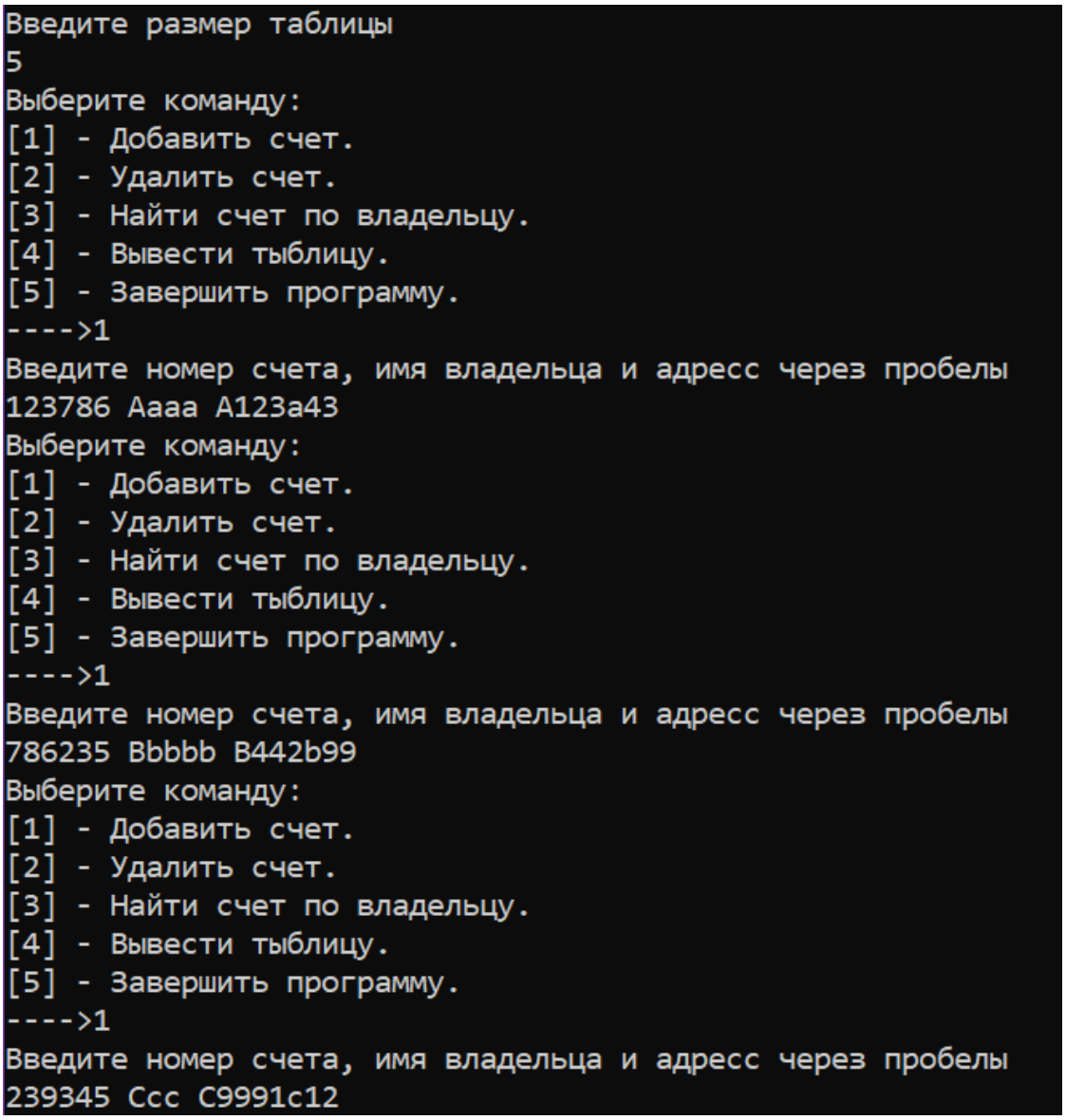

Рис.7 Скриншот добавления счетов в таблицу

```
Выберите команду:
[1] - Добавить счет.
[2] - Удалить счет.
[3] - Найти счет по владельцу.
[4] - Вывести тыблицу.
[5] - Завершить программу.
---->4
      № ячейки    Номер счета           Имя          Адресс
              0              -             -               -
              1         239345           Ccc        C9991c12
              2         123786          Aaaa        A123a43
              3         786235         Bbbbb        B442b99
              4              -             -               -
```

Рис.8 Скриншот вывода таблицы

```
Выберите команду:
[1] - Добавить счет.
[2] - Удалить счет.
[3] - Найти счет по владельцу.
[4] - Вывести тыблицу.
[5] - Завершить программу.
---->3
Введите имя владельца счета
Ccc
      № ячейки    Номер счета           Имя          Адресс
              1         239345           Ccc        C9991c12
```

Рис.9 Скриншот поиска счета по владельцу

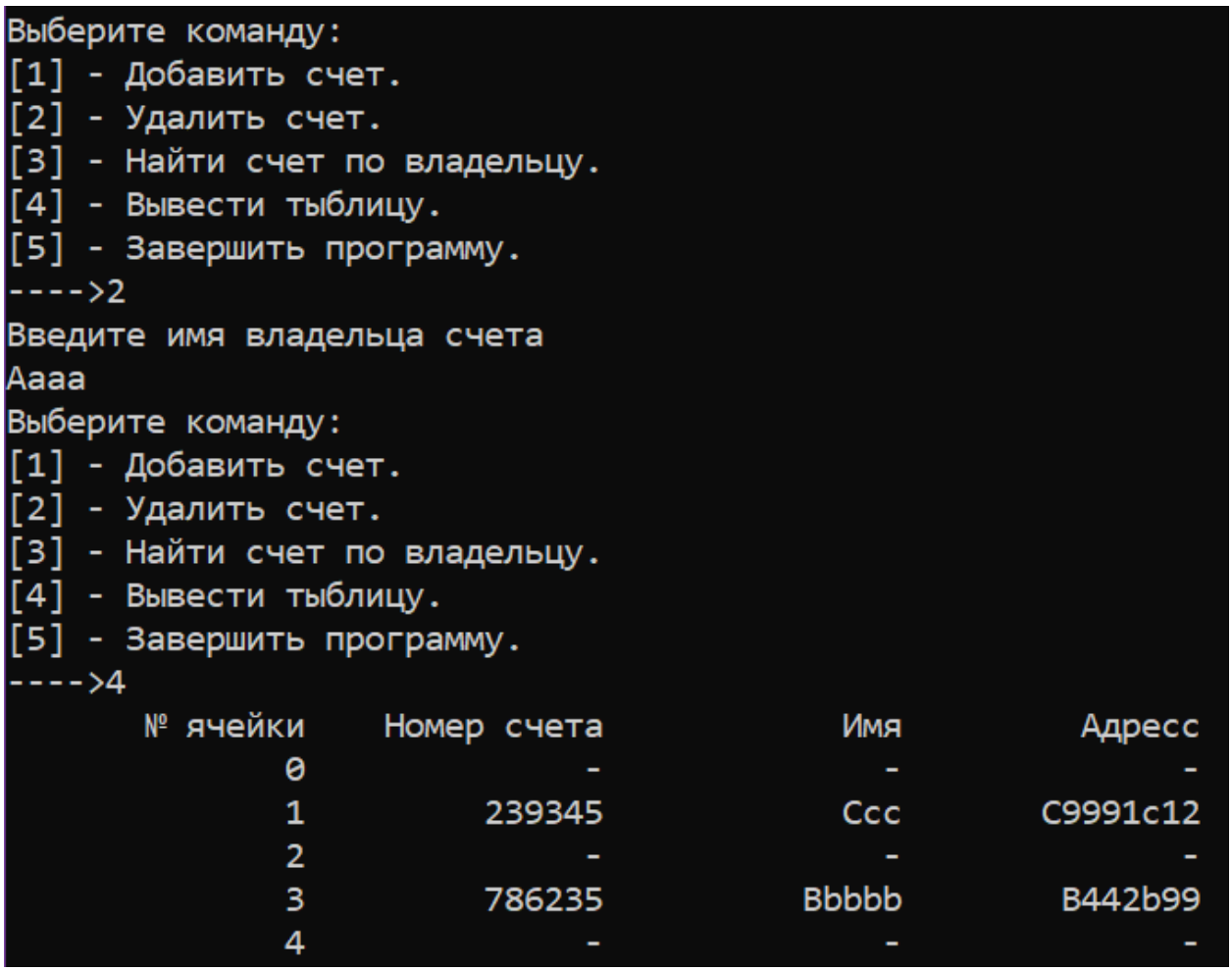

Рис.10 Скриншот удаления счета по владельцу

Выводы

1. В ходе работы была создана программа для работы с хеш-таблицами.
2. Также были реализованы функции записи, удаления, поиска и вывода данных в таблице.
3. Были изучены преимущества и недостатки хранения данных в хеш-таблице:
4. Преимущества: хеш-таблица обладает невероятной скоростью, что позволяет взаимодействовать с данными даже в очень больших таблицах практически моментально. Это возможно благодаря сложности O(1).
5. Недостатки: требует немного больше памяти чем обычные массивы или списки.
6. Таким образом, была изучена работа алгоритмов хеширования.

Список используемых информационных источников

1. Сыромятников В.П. Структуры и алгоритмы обработки данных, лекции, РТУ МИРЭА, Москва, 2020/2021 уч./год.
2. Документация по языку программирования C++, интернет-ресурс: https://en.cppreference.com/w/ (Дата обращения – 02.11.2020)
3. Интегрированная среда разработки для языков программирования C и C++, разработанная компанией JetBrains - CLion / Copyright © 2000-2020 JetBrains s.r.o., интернет-ресурс: https://www.jetbrains.com/clion/learning-center/ (Дата обращения – 02.11.2020).
4. ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения. Интернет-ресурс: http://docs.cntd.ru/document/gost-19-701-90-espd (Дата обращения – 02.11.2020).
5. Описание хеш-функций. интернет-ресурс: https://ru.wikipedia.org/wiki/Хеш-функция (Дата обращения – 02.11.2020).