МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

КУРСОВАЯ РАБОТА

по дисциплине «Алгоритмы и структуры данных»

Тема: Хеш-таблицы с цепочками – вставка и исключение.

Студент гр. 9382 Савельев И.С.

Преподаватель _____ Фирсов М.А.

_____

Санкт-Петербург

2020

ЗАДАНИЕ
НА КУРСОВУЮ РАБОТУ

Студент Савельев И.С.

Группа 9382

Тема работы : Хеш-таблицы с цепочками – вставка и исключение.

Исходные данные:
На вход программе подается последовательность элементов разделенных пробелами.

Содержание пояснительной записки:
«Содержание», «Введение», «Задание», «Ход выполнения работы», «Заключение», «Список использованных источников»

Предполагаемый объем пояснительной записки:
Не менее 15 страниц.

Дата выдачи задания: 31.10.2020

Дата сдачи реферата: 29.11.2020

Дата защиты реферата: 29.11.2020

Студент _____ Савельев И.С.

Преподаватель _____ Фирсов М.А.

АННОТАЦИЯ

В курсовой работе реализована хеш-таблица с цепочками. Программа демонстрирует вставку и удаление из хеш-таблицы элементов введенных пользователем с консоли. В ходе выполнения программ каждый шаг сопровождается объяснениями, которые выводятся в терминал.

SUMMARY

In the course work, a hash table with chains is implemented. The program demonstrates the insertion and removal from the hash table of elements entered by the user from the console. During the execution of programs, each step is accompanied by explanations that are displayed in the terminal.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью работы являлось изучить такую структуру данных, как хеш-таблица. И реализовать ее с использованием цепочек. Для это пришлось изучить алгоритм построения хеш-таблицы, алгоритмы добавления и исключения элементов из неё. Результатом является программа, которая получает на вход последовательность элементов и создает хеш таблицу из них, также пользователь может добавить или исключить элемент из таблицы, все действия сопровождаются визуализацией.

1. ЗАДАНИЕ

Вариант 23.

Хеш-таблицы с цепочками – вставка и исключение. Демонстрация.

2. ХОД ВЫПОЛНЕНИЯ РАБОТЫ

2.1. Описание алгоритма

В начале работы программы ей на вход подается строка считанная из файла. Она разбивается на элементы, которые передаются в хэш-таблицу, где каждому с помощью хеш функции присваивается ключ и он помещается в соответствующую цепочку хеш-таблицы. Далее выводится строка считанная из файла и получившаяся хеш-таблица. Затем пользователю предлагается либо завершить программу, либо добавить элемент в таблицу, либо удалить элемент. Если пользователь выбрал первый вариант, то программа завершается и выводится сообщение сообщающее об этом. Если пользователь выбрал второй вариант, то элемент добавляется в таблицу выводится промежуточная информация, пользователю опять предлагается выбрать, что сделать. Если пользователь выбрал третий вариант, то элемент удаляется из таблицы, если он там есть, выводится промежуточная информация, пользователю опять предлагается выбрать, что сделать. Для удаления элемента из таблице создается его ключ, указывающий на определенную цепочку в хеш-таблице, которая перебирается в поисках элемента. При добавлении нового элемента в хеш-таблицу для него также создается ключ и он помещается в соответствующую цепочку.

2.2. Описание структур данных и функций

Функция string readFile(const string& file_name) получает на вход строку с названием файла создает экземпляр класса ifstream f, которому передает в качестве аргумента file_name, затем создается экземпляр класса stringstream ss

и с помощью метода rdbuf() класс ifstream, потоковый буфер передается ss. Функция возвращает строку с помощью метода str() класс stringstream.

Класс Hash использует шаблонный тип T по умолчанию равный string имеет две private переменные vector<vector<T>> table, int size=10 и следующие public методы.

Hash() конструктор класс Hash, в котором с помощью метода reserve() выделяется место под size массивов.

Метод int hashFunction(T value), принимает шаблонный тип value, внутри метода создается переменная счетчик s, в которой с помощью цикла for суммируется ASCII коды каждого символа элемента. Метод возвращает s по модулю размера таблицы или другими словами ключ элемента.

Метод void howInsert(T value) принимает шаблонный тип value, который передается в hashFunction() и возвращаемое ею значение записывается в переменную key, метод предназначен для вывода в терминал информации поясняющей работу программы.

Метод void insertElem(T value), принимает шаблонный тип value, который передается в hashFunction() и возвращаемое ею значение записывается в переменную key, после чего с помощью метода push_back в соответствующую ячейку таблицы со строкой key записывается значение value.

Метод void displayHash() выводит таблицу в консоль с помощью двух циклов for в первом(внешнем) обходятся все строки таблицы, а во втором(внутреннем) все столбцы соответствующей строки таблицы.

Метод void deleteElem(T value) принимает шаблонный тип value, который передается в hashFunction() и возвращаемое ею значение записывается в переменную key, после чего с помощью цикла в соответствующей цепочки таблицы со значением key перебираются все элементы и при нахождении элемента соответственно равного value он удаляется с помощью метода erase() цикл прерывается с помощью break.

Метод void howDelete(T value) принимает шаблонный тип value, который передается в hashFunction() и возвращаемое ею значение записывается в

переменную key, метод предназначен для вывода в терминал информации поясняющей работу программы.

2.3. Описание интерфейса пользователя

Пользователь вводит в отдельный файл через пробел элементы, которые он хочет видеть в таблице изначально, затем в начале выполнения программы в терминал выводится строка пользователя считанная из файла и хеш-таблица построенная по этой строке. После чего пользователю предлагается выбрать либо закончить программу для это надо ввести в консоль !exit, либо вставить элемент для это надо ввести в консоль !insert, либо удалить элемент из таблицы для это надо ввести в консоль !delete, если пользователь выбрал первый вариант то программа завершается с соответствующим сообщением, если 2 или 3 то пользователя предлогается ввести элемент который он хочет вставить или удалить, после считывания с консоли ввода пользователя, выводится промежуточная информация, а затем и сама хеш-таблица.

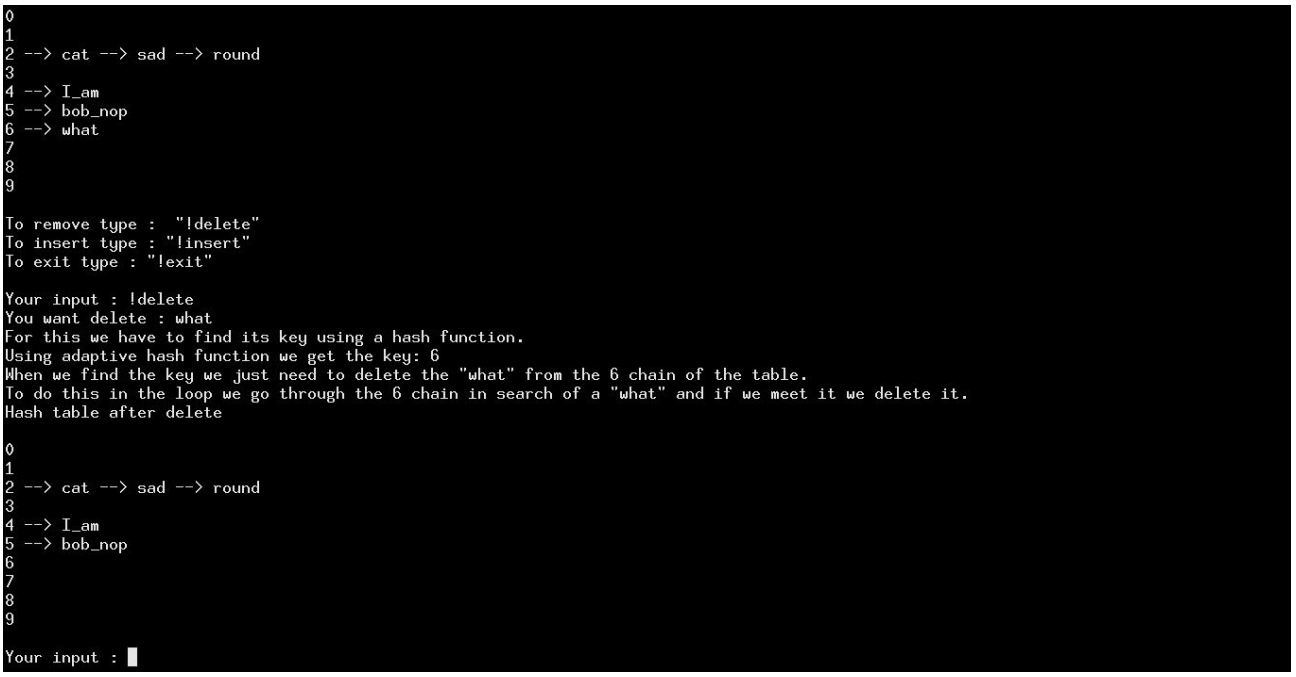
2.4. Тестирование

Результат запуска программы с входной строкой: cat sad I_am bob_nop round what

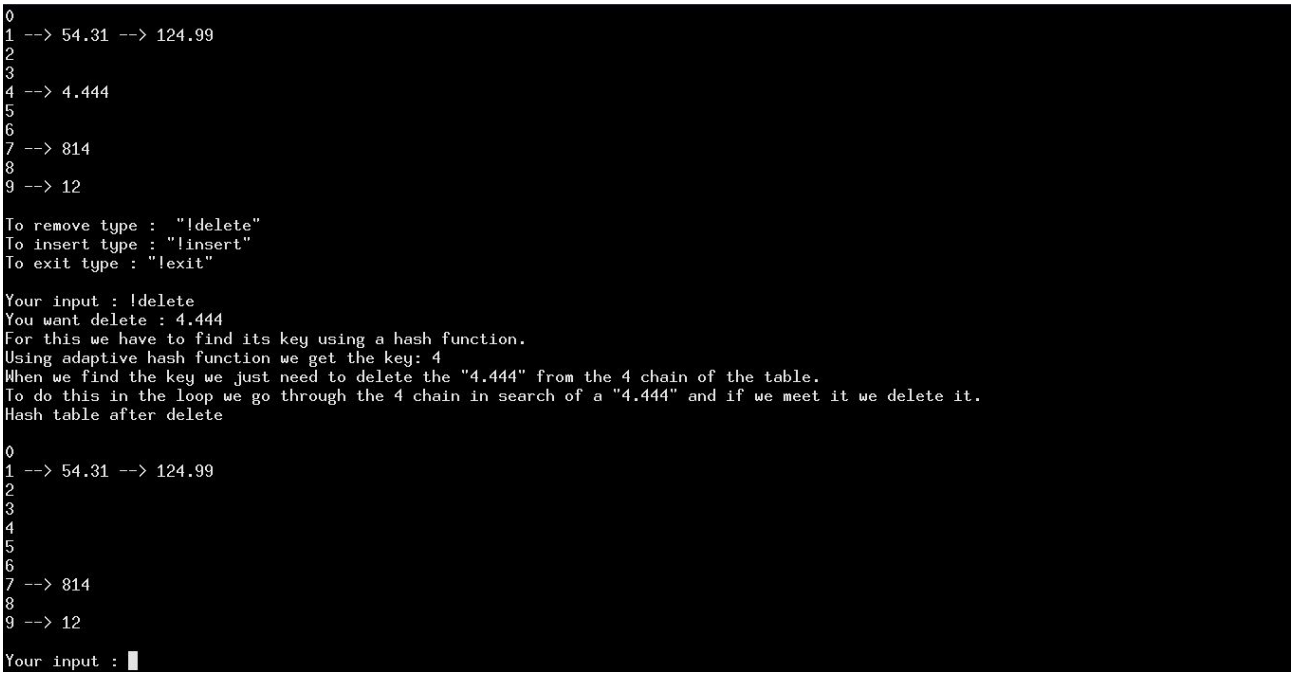
Результат запуска программы с входной строкой: 12 54.31 814 4.444 124.99

Результат работы программы с пустой входной строкой

Удаление элемента: what



Удаление элемента: 4.444



Вставка элемента: hey

```
0
1
2 --> cat --> sad --> round
3
4 --> I_am
5 --> bob_nop
6 --> what
7
8
9

To remove type :  "!delete"
To insert type : "!insert"
To exit type : "!exit"

Your input : !insert
You want insert : hey
For this we have to find its key using a hash function.
Using adaptive hash function we get the key: 6
When we find the key we just need to insert the "hey" into the 6 chain of the table.
Hash table after insert

0
1
2 --> cat --> sad --> round
3
4 --> I_am
5 --> bob_nop
6 --> what --> hey
7
8
9

Your input :
```

Вставка элемента: 999

```
0
1 --> 54.31 --> 124.99
2
3
4 --> 4.444
5
6
7 --> 814
8
9 --> 12

To remove type :  "!delete"
To insert type : "!insert"
To exit type : "!exit"

Your input : !insert
You want insert : 999
For this we have to find its key using a hash function.
Using adaptive hash function we get the key: 1
When we find the key we just need to insert the "999" into the 1 chain of the table.
Hash table after insert

0
1 --> 54.31 --> 124.99 --> 999
2
3
4 --> 4.444
5
6
7 --> 814
8
9 --> 12

Your input :
```

Завершение работы программы

ЗАКЛЮЧЕНИЕ

В ходе работы над курсовой работой была изучена такая структура данных, как хеш-таблица с цепочками. Также была реализована программа, которая позволяет продемонстрировать вставку и удаление элементов из хеш-таблицы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Справочная информация о языке программирования C++ // The C++ Resources Network // URL: https://www.cplusplus.com (дата обращения: 27.11.2020)

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <string>
#include <iterator>

#define FILE "/home/indiora/C++/Aisd/read.txt"

using namespace std;

string readFile(const string& file_name) {
	ifstream f(file_name); // Экземпляр класса
	stringstream ss;
	ss << f.rdbuf(); // Возвращает указатель на потоковый буфер
	return ss.str(); // Возвращает строку
}


template <typename T=string>
class Hash {
private:
	vector<vector<T>> table;
	// размер таблицы
	int size=10;
public:
	// Конструктор
	Hash() {
	table.reserve(size);
	}

	// Хеш функция реализованная адаптивным методом
	int hashFunction(T value) {
	int s = 0;
	for(int i = 0 ; i < value.length(); i++)
		s += value[i];
	// возвращает ключ
	return s % size;
	}

	// Вставка elem
	void insertElem(T value) {
	// Расчитываем ключ от значения
	int key = hashFunction(value);
	// Добавляем в соответствующую ячейку таблицы
	table[key].push_back(value);
	}

	// Пояснения для вставки
	void howInsert(T value) {
	int key = hashFunction(value);
	std::cout << "For this we have to find its key using a hash function." <<
'\n';
	std::cout << "Using adaptive hash function we get the key: " << key << '\n';
	std::cout << "When we find the key we just need to insert the \"" << value
<< "\" into the " << key << " chain of the table." << '\n';
	}
	// Пояснения для удаления
	void howDelete(T value) {
```

```
      int key = hashFunction(value);
      std::cout << "For this we have to find its key using a hash function." <<
'\n';
      std::cout << "Using adaptive hash function we get the key: " << key << '\n';
      std::cout << "When we find the key we just need to delete the \"" << value
<< "\" from the " << key << " chain of the table." << '\n';
      std::cout << "To do this in the loop we go through the " << key << " chain
in search of a \"" << value << "\" and if we meet it we delete it." << '\n';
      }

   // Удаление элемента
   void deleteElem(T value) {
      // получаем хеш-индекс ключа
      int key = hashFunction(value);
      // В цикле обходим цепочку с соответствующим ключом
      for   (typename   vector<T>::iterator   i   =   begin(table[key]);   i   !=
end(table[key]); ++i) {
            // При нахождении elem удаляем его
            if (*i == value){
            table[key].erase(i);
            break;
            }
      }
   }

      void displayHash() {
      // Обходим всю таблицу
      cout << "\n";
      for (int i = 0; i < size; i++) {
            cout << i;
            // Выводим элементы цепочки
            for (auto x : table[i]) {
            cout << " --> " << x;
            }
            cout << "\n";
      }
      cout << "\n";
   }

};



int main(int argc, char const *argv[]) {
      // Создаем экземпляр класса Hash
      Hash table;
      // Создаем экземпляр класс is.. и передаем ему строку считанную из файла
      istringstream iss(readFile(FILE));
      // Разбиваем считанную строку на Elem
      vector<string>                          tokens{istream_iterator<string>{iss},
istream_iterator<string>{}};
      // Заносим Elem в хэш табоицу
      for(int i = 0; i < tokens.size(); i++){
      table.insertElem(tokens[i]);
      }
      // Выводим строку считанную из файла
      cout << "You entered: ";
      cout << iss.str() << '\n';
      // Выводим хэш таблицу
      cout << "Hash table" << "\n";
      table.displayHash();

      string input;
```

```
	cout << "To remove type :  \"!delete\" \n"
		"To insert type : \"!insert\" \n"
		"To exit type : \"!exit\" \n\n";

	while(true){
	// Считываем значение введенное пользователем
	std::cout << "Your input : ";
	cin >> input;
	// Для удаления элеммента
	if (input == "!delete") {
		std::cout << "You want delete : ";
		cin >> input;
		table.howDelete(input);
		table.deleteElem(input);
		std::cout << "Hash table after delete" << '\n';
		table.displayHash();
	}
	// Для добавления элемента
	else if (input == "!insert") {
		std::cout << "You want insert : ";
		cin >> input;
		table.howInsert(input);
		table.insertElem(input);
		std::cout << "Hash table after insert" << '\n';
		table.displayHash();
	}
	// Для выхода
	else if (input == "!exit") {
		break;
	}
	// Иначе
	else {
		std::cout << "Incorrect input please try again." << '\n';
	}
	}
	std::cout << "Program end !" << '\n';
	return 0;
}
```