МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5

по дисциплине «Алгоритмы и структуры данных»

Тема: Хеш-таблица с цепочками

Студент гр. 9304 Сорин А.В.

Преподаватель Филатов А.Ю.

Санкт-Петербург

2020

Цель работы.

Узнать о хеш-таблицах с цепочками и их использовании в практике. Реализовать их на языке C++.

Задание.

1) По заданной последовательности элементов Elem построить структуру данных определённого типа – БДП или хеш-таблицу;

2) Для построенной структуры данных проверить, входит ли в неё элемент e типа Elem, и если входит, то в скольких экземплярах. Добавить элемент e в структуру данных. Предусмотреть возможность повторного выполнения с другим элементом.

Вариант 23 - Хеш-таблица: с цепочками.

Формат входных и выходных данных.

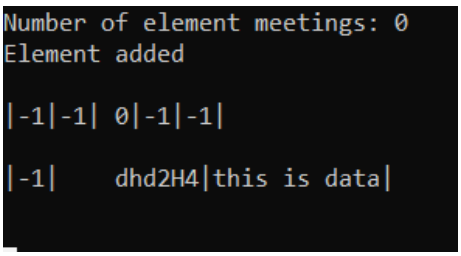
На вход подается в начале ключ затем сами данные. Например:

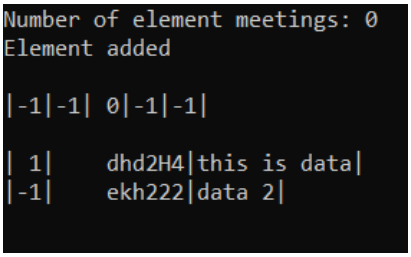
1)dgd4Fh

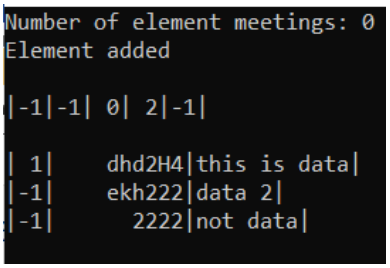
2)this is data

На выходе результат – количество элементов с таким же ключем. Также сама хеш – таблица.

Пример работы программы:







Выполнение работы.

Для выполнения задания было создано 2 класса – key_line и hesh_table.

У класса key_line есть 3 конструктора key_line() – просто создает экземпляр, key_line(std::string NewName, std::string NewData) – заполняет все поля класса кроме одного, key_line(std::string NewName, std::string NewData, int NewChain) - - заполняет все поля класса. Также у класса есть поля std::string Name – ключ, std::string Data – данные, int Chain = -1 – цепочка.

У класса hesh_table есть поля unsigned int HeshTableSize = 11 – размер таблицы индексов, unsigned int KeySize = 8 – максимальный размер ключа std::vector<key_line> KeyTable – таблица сданными и цепочками, std::unique_ptr<int[]> HeshTable – таблица с индексами, hesh_table(unsigned int HTS, unsigned int KS) – констуктор, int AddToHeshTable(std::string NewName, std::string NewData) – функция добавления элемента в таблицу, int Ord(char C) – функция, возвращающая число, соответствующее символу, int H(std::string Name) – хеш – функция, void ReadKey(std::string& Key, std::stringstream& Stream) – функция считывания ключа.

Функция добавления работает следующим образом:

Если в таблице индексов -1, то добавляет в конец списка данные и устанавливает индекс в таблице индексов. Если в таблице индексов не -1, то идем по цепочке, пока она не станет -1. Тогда ее можно заменить на нужное число и добавить данные в таблицу.

Тестирование.

Тестирование проводится при помощи скрипта на python. При этом текст из входного файла подаётся в поток на вход, а получившийся на выходе результат сравнивается в правильным и выводится, пройден тест, или нет. В таблице приведены результаты тестирования.

Таблица Б.1 – Результаты тестирования

№	Входные данные	Выходные данные	Комментарии
1	1)abc0 2)ex0 1)abc1 2)ex1 1)abc2 2)ex2 1)abc3 2)ex3 1)abc4 2)ex4	\x1b[2J\x1b[HNumber of element meetings: 0\nElement added\n\n|-1|-1| 0|-1|\n\n|-1| abc0|ex0|\n\n\x1b[2J\x1b[HNumber of element meetings: 0\nElement added\n\n|-1|-1|-1| 0| 1|\n\n|-1| abc0|ex0|\n|-1| abc1|ex1|\n\n\x1b[2J\x1b[HNumber of element meetings: 0\nElement added\n\n| 2|-1|-1| 0| 1|\n\n|-1| abc0|ex0|\n|-1| abc1|ex1|\n|-1| abc2|ex2|\n\n\x1b[2J\x1b[HNumber of element meetings: 0\nElement added\n\n| 2| 3|-1| 0| 1|\n\n|-1| abc0|ex0|\n|-1| abc1|ex1|\n|-1| abc2|ex2|\n|-1| abc3|ex3|\n\n\x1b[2J\x1b[HNumber of element meetings: 0\nElement added\n\n| 2| 3| 4| 0| 1|\n\n|-1| abc0|ex0|\n|-1| abc1|ex1|\n|-1| abc2|ex2|\n|-1| abc3|ex3|\n|-1| abc4|ex4|\n\n	Пять элементов, с различным значением хеш - функции
2	2	Error while entering expression	Не верно построенное выражение
3	1)5 2)s0 1)5 2)s1 1)5	\x1b[2J\x1b[HNumber of element meetings: 0\nElement added\n\n| 0|-1|-1|-1|-1|\n\n|-1| 5|s0|\n\n\x1b[2J\x1b[HNumber of element meetings: 1\nElement	Три элемента с одинаковым ключем.

	2)s2	added\n\n| 0|-1|-1|-1|-1|\n\n| 1| 5|s0|\n|-1| 5|s1|\n\n\x1b[2J\x1b[HNumber of element meetings: 2\nElement added\n\n| 0|-1|-1|-1|-1|\n\n| 1| 5|s0|\n| 2| 5|s1|\n|-1| 5|s2|\n\n	

Пример тестирования:

```
aleksey@aleksey-VirtualBox:~/Загрузки/lab5-20201210T055617Z-001/lab5$ python3 tests.py

enter ['1)abc0\n', '2)ex0\n', '1)abc1\n', '2)ex1\n', '1)abc2\n', '2)ex2\n', '1)abc3\n', '2)ex3\n', '1)abc4\n', '2)ex4\n', '\n']
1)abc0
2)ex0
1)abc1
2)ex1
1)abc2
2)ex2
1)abc3
2)ex3
1)abc4
2)ex4


Number of element meetings: 0
Element added

|-1|-1|-1| 0|-1|

|-1|      abc0|ex0|
```

```
Number of element meetings: 0
Element added

|-1|-1|-1| 0| 1|

|-1|      abc0|ex0|
|-1|      abc1|ex1|
```

```
Number of element meetings: 0
Element added

| 2|-1|-1| 0| 1|

|-1|      abc0|ex0|
|-1|      abc1|ex1|
|-1|      abc2|ex2|
```

```
Number of element meetings: 0
Element added

| 2| 3|-1| 0| 1|

|-1|      abc0|ex0|
|-1|      abc1|ex1|
|-1|      abc2|ex2|
|-1|      abc3|ex3|









Number of element meetings: 0
Element added

| 2| 3| 4| 0| 1|

|-1|      abc0|ex0|
|-1|      abc1|ex1|
|-1|      abc2|ex2|
|-1|      abc3|ex3|
|-1|      abc4|ex4|
```

```
.
enter ['2\n', '\n']
2
Error while entering expression
.
enter ['1)5\n', '2)s0\n', '1)5\n', '2)s1\n', '1)5\n', '2)s2\n', '\n']
1)5
2)s0
1)5
2)s1
1)5
2)s2


Number of element meetings: 0
Element added

| 0|-1|-1|-1|-1|

|-1|         5|s0|
```

```
Number of element meetings: 1
Element added

| 0|-1|-1|-1|-1|

| 1|         5|s0|
|-1|         5|s1|
```

```
Number of element meetings: 2
Element added

| 0|-1|-1|-1|-1|

| 1|         5|s0|
| 2|         5|s1|
|-1|         5|s2|


.
----------------------------------------------------------------------
Ran 3 tests in 109.549s

OK
```

Выводы.

Стало известно о хеш-таблицах с цепочками и их использовании в практике. Они были реализованы на языке C++. Была реализована хеш таблица, добавление в нее элемента. Хеш таблица нужна для того, чтобы находить нужный элемент по его ключу. Хеш таблица с цепочками лучше хеш таблицы прямого доступа лучшим разрешением конфликтов. Также был реализован вариант с таблицей индексов. Он позволяет избежать перевыделения памяти, когда элементов слишком много.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include <string>
#include "hesh.h"

std::stringstream GetLine(void) {
std::string Str;
if (!std::getline(std::cin, Str))
     throw std::runtime_error("Error while reading from stream");
std::stringstream Stream(Str);
return Stream;
}

std::ostream& operator<<(std::ostream& os, hesh_table& HT) {
os << '|';
for (int it = 0; it < HT.HeshTableSize; it++) {
     os.width(2);
     os << HT.HeshTable[it] << '|';
}
os << "\n\n";
for (int it = 0; it < HT.KeyTable.size(); it++) {
     os << '|';
     os.width(2);
     os << HT.KeyTable[it].Chain;
     os << '|';
```

```
	std::string s = HT.KeyTable[it].Name;
	os.width(HT.KeySize);
	os << s;
	os << '|';
	os << HT.KeyTable[it].Data;
	os << '|';
	os << '\n';
}
return os << '\n';
}

int main() {
try
{
	unsigned int HTS = 5, MKS = 10;
	/*std::cout << "Enter hesh table size: ";
	std::cin >> HTS;
	std::cout << "\n";
	std::cout << "Enter max key size: ";
	std::cin >> MKS;
	std::cout << "\n";*/

	hesh_table HT(HTS, MKS);
	int Res = 0;
	std::string Key;
	while (1) {
		Key = "";
```

```
            std::stringstream Stream = GetLine();
            char C = 0;
            if (Stream.get(C).eof())
                  break;
            if (C != '1')
                  throw    std::invalid_argument("Error    while    entering
expression");
            if ((C = Stream.get()) != ')')
                  throw    std::invalid_argument("Error    while    entering
expression");
            HT.ReadKey(Key, Stream);
            std::stringstream Stream2 = GetLine();
            std::string Data;
            if ((C = Stream2.get()) != '2')
                  throw    std::invalid_argument("Error    while    entering
expression");
            if ((C = Stream2.get()) != ')')
                  throw    std::invalid_argument("Error    while    entering
expression");
            Data = Stream2.str().erase(0, 2);
            Res = HT.AddToHeshTable(Key, Data);
            std::cout << "\x1B[2J\x1B[H";
            std::cout << "Number of element meetings: " << Res;
            std::cout << "\nElement added\n\n";
            std::cout << HT;
      }

}
catch (const std::exception& Error)
```

```
{
     std::cout << Error.what();
}
return 0;
}
```

Название файла: hesh.h

```
#ifndef __HESH_H
#define __HESH_H

#include <iostream>
#include <stdexcept>
#include <sstream>
#include <string>
#include <vector>
#include <chrono>
#include <thread>

/*template <typename base>
std::ostream&      operator<<(std::ostream&      os,
std::list<base>& res) {
for (auto it = res.begin(); it != res.end(); it++)
{
     os << '{' << *it << '}';
}
return os << '\n';
}*/

class key_line {
public:
key_line();
key_line(std::string NewName, std::string NewData);
key_line(std::string NewName, std::string NewData, int
NewChain);
std::string Name;
std::string Data;
int Chain = -1;
};

class hesh_table {
public:
```

```
unsigned int HeshTableSize = 11;
unsigned int KeySize = 8;
std::vector<key_line> KeyTable;
std::unique_ptr<int[]> HeshTable;
hesh_table(unsigned int HTS, unsigned int KS);
int  AddToHeshTable(std::string  NewName,  std::string
NewData);
int Ord(char C);
int H(std::string Name);
void  ReadKey(std::string&  Key,  std::stringstream&
Stream);
};

#endif // __HESH_H
```

Название файла: hesh.cpp

```
#include "hesh.h"

key_line::key_line() {

}
key_line::key_line(std::string  NewName,  std::string
NewData) : Name(NewName), Data(NewData) {

}
key_line::key_line(std::string  NewName,  std::string
NewData, int NewChain) : Name(NewName), Data(NewData),
Chain(NewChain) {

}

hesh_table::hesh_table(unsigned int HTS, unsigned int
KS) : HeshTableSize(HTS), KeySize(KS) {
     std::unique_ptr<int[]>                    Fi(new
int[HeshTableSize]);
     HeshTable = std::move(Fi);
     for (int i = 0; i < HeshTableSize; i++)
          HeshTable[i] = -1;
}
int  hesh_table::AddToHeshTable(std::string  NewName,
std::string NewData) {
     int N = H(NewName);
     if (HeshTable[N] == -1) {
          key_line KL(NewName, NewData);
          KeyTable.push_back(KL);
          HeshTable[N] = KeyTable.size() - 1;
          return 0;
     }
     else {
```

```
          int Count = 0;
          int SaveInd = HeshTable[N];
          if (KeyTable[SaveInd].Name == NewName)
               Count++;
          while (KeyTable[SaveInd].Chain != -1) {
               SaveInd = KeyTable[SaveInd].Chain;
               if     (KeyTable[SaveInd].Name     ==
NewName)
                    Count++;
          }
          KeyTable[SaveInd].Chain = KeyTable.size();
          key_line KL(NewName, NewData);
          KeyTable.push_back(KL);
          return Count;
     }
}
int hesh_table::Ord(char C) {
     if (std::isdigit(C))
          return C - '0';
     else if (C >= 'a' && C <= 'z') {
          return C - 'a' + 10;
     }
     else if (C >= 'A' && C <= 'Z') {
          return C - 'A' + 10 + 'z' - 'a' + 1;
     }
     else
          throw  std::invalid_argument("Error  while
entering expression");
}
int hesh_table::H(std::string Name) {
     int Sum = 0;
     for (int i = 0; i < Name.size(); i++)
          Sum += Ord(Name[i]);
     int Mod = HeshTableSize;
     Sum %= Mod;
     return Sum;
}
void      hesh_table::ReadKey(std::string&      Key,
std::stringstream& Stream) {
     char C = 0;
     if (!Stream.get(C))
          throw  std::invalid_argument("Error  while
entering expression");
     for (int i = 0; (std::isdigit(C) || (C >= 'a' &&
C <= 'z') || (C >= 'A' && C <= 'Z')) && (i < KeySize);
i++) {
          Key += C;
          if (Stream.get(C).eof())
               return;
          if (Stream.fail())
```

```
                throw    std::invalid_argument("Error
while entering expression");
      }
      throw     std::invalid_argument("Error     while
entering expression");
}
```

Название файла тестирующей программы: tests.py

```
import unittest
import subprocess


class tester(unittest.TestCase):

def test1(self):
      with open('./Tests/test_1.txt', 'r') as file:
            print('\nenter', file.readlines())
            res  =  '\x1b[2J\x1b[HNumber  of  element
meetings: 0\nElement added\n\n|-1|-1|-1| 0|-1|\n\n|-
1|       abc0|ex0|\n\n\x1b[2J\x1b[HNumber of element
meetings: 0\nElement added\n\n|-1|-1|-1| 0| 1|\n\n|-
1|                                    abc0|ex0|\n|-1|
abc1|ex1|\n\n\x1b[2J\x1b[HNumber of element meetings:
0\nElement   added\n\n|   2|-1|-1|   0|   1|\n\n|-1|
abc0|ex0|\n|-1|                        abc1|ex1|\n|-1|
abc2|ex2|\n\n\x1b[2J\x1b[HNumber of element meetings:
0\nElement   added\n\n|   2|   3|-1|   0|   1|\n\n|-1|
abc0|ex0|\n|-1|     abc1|ex1|\n|-1|     abc2|ex2|\n|-
1|       abc3|ex3|\n\n\x1b[2J\x1b[HNumber of element
meetings: 0\nElement added\n\n| 2| 3| 4| 0| 1|\n\n|-
1|          abc0|ex0|\n|-1|          abc1|ex1|\n|-1|
abc2|ex2|\n|-1|    abc3|ex3|\n|-1|    abc4|ex4|\n\n'




      self.assertEqual(subprocess.check_output(["./la
b5"], universal_newlines=True), res)
            print(res)
def test2(self):
      with open('./Tests/test_2.txt', 'r') as file:
            print('\nenter', file.readlines())
            res = 'Error while entering expression'
```

```
     self.assertEqual(subprocess.check_output(["./la
b5"], universal_newlines=True), res)
          print(res)
def test3(self):
     with open('./Tests/test_3.txt', 'r') as file:
          print('\nenter', file.readlines())
          res  =  '\x1b[2J\x1b[HNumber  of  element
meetings: 0\nElement added\n\n| 0|-1|-1|-1|-1|\n\n|-
1|           5|s0|\n\n\x1b[2J\x1b[HNumber of element
meetings: 1\nElement added\n\n| 0|-1|-1|-1|-1|\n\n| 1|
5|s0|\n|-1|           5|s1|\n\n\x1b[2J\x1b[HNumber of
element meetings: 2\nElement added\n\n| 0|-1|-1|-1|-
1|\n\n| 1|          5|s0|\n| 2|          5|s1|\n|-1|
5|s2|\n\n'
```

```
     self.assertEqual(subprocess.check_output(["./la
b5"], universal_newlines=True), res)
          print(res)
```

```
if __name__ == '__main__':
unittest.main()
```

Название файла: Makefile `lab2`:

```
 all: lab5

lab5: ./src/main.o ./src/hesh.o
 g++  src/main.o  src/hesh.o  -o
 lab5
 rm -rf *.o
main.o:          ./src/main.cpp
 ./src/hesh.h
 g++ -std=c++17 ./src/main.cpp
hesh.o:          ./src/hesh.cpp
 ./src/hesh.h
 g++ -std=c++17 ./src/hesh.cpp

clear:
```

```
rm -rf *.o lab5
rm -rf ./src/*.o
```