Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Кафедра ИС

Отчет

по лабораторной работе № 7

«STL. Контейнеры»

Выполнил

ст. гр. ИТб22

Серегин А.В.

Проверил:

Тимофеев И.С.

Севастополь

2015

1. Цель работы

Приобретение практических навыков в написание объектно-ориентированных программ с использованием контейнеров STL. Освоение особенностей отладки объектно-ориентированных программ.

2. Варианты заданий

Вариант 5:

Написать программу “Моя записная книжка”. Предусмотреть возможность работы с произвольным числом записей, поиска записи по какому-либо признаку (например, по фамилии, дате рождения или номеру телефона), добавления и удаление записей, сортировки по разным полям.

3. Текст программы

Bookmark.h

#pragma once

#ifndef \_BOOKMARK\_H\_

#define \_BOOKMARK\_H\_

#include <ostream>

#include <exception>

#include <iomanip>

#include <map>

#include <string>

#include <algorithm>

#define record std::string, long long

class Bookmark

{

private:

std::multimap<record> records\_;

size\_t size\_;

public:

Bookmark();

Bookmark(const Bookmark& \_bookmark);

void add(const std::pair<record>& \_newRecord);

std::multimap<record>::iterator search(const long long& \_phone);

std::pair<std::multimap<record>::iterator,

std::multimap<record>::iterator> search(const std::string& \_account);

std::multimap<record>::iterator search(const std::pair<record>& \_record);

void del(const std::pair<record>& \_record);

void del(const size\_t& \_number);

void del(const std::multimap<record>::iterator& \_iter);

friend std::ostream& operator<<(std::ostream& os, const Bookmark& \_bookmark);

};

#endif

Bookmark.cpp

#include "Bookmark.h"

std::ostream& operator<<(std::ostream& os, const std::pair<record>& \_record)

{

os.setf(std::ios::left);

os << std::setw(35) << \_record.first << std::setw(10) << \_record.second;

return os;

}

std::ostream& operator<<(std::ostream& os, const Bookmark& \_bookmark)

{

os.setf(std::ios::left);

os << std::setw(4) << "N";

os << std::setw(35) << "FIO";

os << std::setw(10) << "Phone" << std::endl;

size\_t rowCounter = 0;

for (auto i : \_bookmark.records\_)

{

os << std::setw(4) << ++rowCounter << i << std::endl;

}

return os;

}

Bookmark::Bookmark()

{

size\_ = 0;

}

Bookmark::Bookmark(const Bookmark & \_bookmark)

{

records\_ = std::multimap<record>(\_bookmark.records\_);

size\_ = \_bookmark.size\_;

}

void Bookmark::add(const std::pair<record>& \_newRecord)

{

records\_.insert(\_newRecord);

size\_++;

}

std::multimap<record>::iterator Bookmark::search(const long long& \_phoneNumber)

{

for (std::multimap<record>::iterator i = records\_.begin(); i != records\_.end(); ++i)

if (i->second == \_phoneNumber)

return i;

throw std::exception("Phone nubmer not enable in bookmark");

}

std::pair<std::multimap<record>::iterator,

std::multimap<record>::iterator> Bookmark::search(const std::string& \_account)

{

for (std::multimap<record>::iterator i = records\_.begin(); i != records\_.end(); ++i)

if (i->first == \_account)

for (auto j = i; (j->first == \_account) && (j != records\_.end()); ++j)

return std::make\_pair(i, j);

throw std::exception("Account not enable in bookmark");

}

std::multimap<record>::iterator Bookmark::search(const std::pair<record>& \_record)

{

for (std::multimap<record>::iterator i = records\_.begin(); i != records\_.end(); ++i)

if (i->second == \_record.second && i->first == \_record.first)

return i;

throw std::exception("Phone nubmer not enable in bookmark");

}

void Bookmark::del(const std::pair<record>& \_record)

{

std::multimap<record>::iterator iter = this->search(\_record);

std::multimap<record>::iterator iter2 = iter;

iter++;

records\_.erase(iter2, iter);

size\_--;

}

void Bookmark::del(const size\_t & \_number)

{

if (\_number > this->size\_ - 1)

throw std::exception("Size error");

std::multimap<record>::iterator iter = records\_.begin();

std::multimap<record>::iterator iter2;

for (size\_t i = 0; i < \_number; ++i)

iter++;

iter2 = iter;

iter2++;

records\_.erase(iter,iter2);

size\_--;

}

main.cpp

#include <iostream>

#include <conio.h>

#include <set>

#include "Bookmark.h"

int main()

{

try

{

Bookmark bookmark;

std::string name;

long long phone;

std::set<char> keyset;

keyset.insert(27);

keyset.insert('1');

keyset.insert('2');

keyset.insert('3');

char c;

bool flag = false;

while (true)

{

system("cls");

c = 111;

printf("1.Add\n2.Del\n3.Print\nEsc.Quit\n");

while (!keyset.count(c))

c = \_getch();

switch (c)

{

case '1':

{

system("cls");

std::cout << "Input FIO:" << std::endl;

if (flag) std::cin.get();

std::getline(std::cin, name);

std::cout << "Input phone number:" << std::endl;

std::cin >> phone;

bookmark.add(std::make\_pair(name, phone));

flag = true;

system("pause");

} break;

case '2':

{

system("cls");

size\_t pos;

std::cout << "Input position on bookmark:" << std::endl;

std::cin >> pos;

bookmark.del(pos - 1);

system("pause");

} break;

case '3':

{

system("cls");

std::cout << bookmark;

std::cout.flush();

system("pause");

}break;

case 27:

{

return 0;

}break;

}

}

}

catch (const std::exception& \_e)

{

std::cerr << \_e.what() << std::endl;

}

system("pause");

return 0;

}

4. Тестовые примеры.

На рисунке ниже приведен результат выполнения программы.

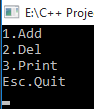


Рисунок 1 – Главное меню.

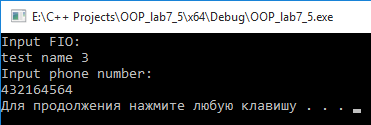


Рисунок 2 – Меню добавления.

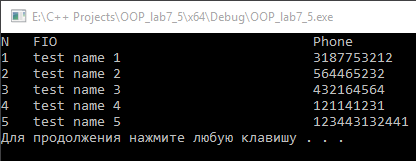


Рисунок 3 – Меню списка контактов.

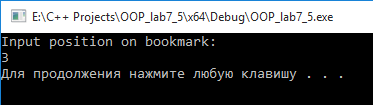


Рисунок 4 – Меню удаления контакта.

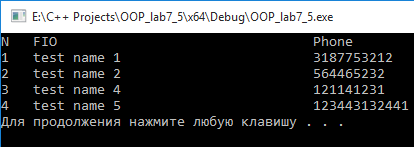


Рисунок 5 – Список, после удаления контакта.

Выводы

В ходе лабораторной работы были приобретены навыки написания программ с использованием стандартной библиотеки шаблонов языка C++. Были изучены основные контейнеры, итераторы и использование алгоритмов стандартной библиотеки.