ОТЧЁТ

О ЛАБОРАТОРНОЙ РАБОТЕ №7  
по теме: Обработка исключительных ситуаций, шаблоны классов

по дисциплине: Разработка программных модулей

09.02.07 Информационные системы и программирование

Проверил:

\_\_\_\_\_\_\_\_ Антонов М.Б.

Дата: «\_\_» \_\_\_\_\_ 20\_\_ г.

Оценка: \_\_\_\_\_\_\_\_\_\_

УНИВЕРСИТЕТ ИТМО

Факультет среднего профессионального образования

Санкт-Петербург 2019

Выполнил:

студент группы Y2333

\_\_\_\_\_\_\_\_\_\_\_Харченко Д.И.

1. ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Цель работы: написать программу, реализующую двунаправленную очередь с использованием std::list. Программа должна обрабатывать исключительные ситуации и иметь меню.

Код Deque.hpp:

#ifndef \_7\_DEQUE\_HPP

#define \_7\_DEQUE\_HPP

#include <iostream>

using namespace std;

/\*\*

\* Класс двусторонней очереди

\* @tparam T

\*/

template <typename T>

class Deque {

public:

list<T> ls; ///< В задании к лабораторной работе дано, что для реализации двунаправленной очереди использовать std::list

/\*\*

\* Добалвение элемента в начало очереди

\* @param value то, что нужно добавить

\*/

void pushFront(T value) {

ls.push\_front(value);

};

/\*\*

\* Добалвение элемента в конец очереди

\* @param value то, что нужно добавить

\*/

void pushBack(T value) {

ls.push\_back(value);

};

/\*\*

\* Удаление элемента из начала

\*/

void popFront() {

ls.pop\_front();

};

/\*\*

\* Удаление элемента из конца

\*/

void popBack() {

ls.pop\_back();

};

/\*\*

\* Проверка наличия элементов в очереди

\* @return true/false – пустая очередь/нет

\*/

bool isEmpty() {

return ls.empty();

}

/\*\*

\* Поиск элемента в очереди

\* @param target элемент

\* @return true/false – есть искомый/нет

\*/

bool isInDeq(T target) {

bool result = false;

for (auto it = ls.begin(); it != ls.end(); ++it) {

if (\*it == target) {

result = true;

break;

}

}

return result;

};

/\*\*

\* Вывод очереди в консоль

\*/

void show() {

for (auto it = ls.begin(); it != ls.end(); ++it) {

cout << \*it << ' ';

}

cout << endl;

}

};

#endif //\_7\_DEQUE\_HPP

Код MyException.hpp:

#ifndef \_7\_MYEXCEPTION\_HPP

#define \_7\_MYEXCEPTION\_HPP

#include <iostream>

using namespace std;

/\*\*

\* Пользовательский класс исключений

\*/

class MyException : public exception {

public:

/\*\*

\* Метод для возврата исключения

\* @return

\*/

const char\* what() const throw ()

{

return "C++ Exception";

}

};

#endif //\_7\_MYEXCEPTION\_HPP

Код Utils.hpp:

#ifndef \_7\_UTILS\_HPP

#define \_7\_UTILS\_HPP

#include <iostream>

using namespace std;

/\*\*

\* Вывод в консоль меню первого уровня

\*/

void draw1L() {

cout << "1 – Выход" << endl;

cout << "2 – Создать интовую очередь" << endl;

cout << "3 – Создать флотовую очередь" << endl;

cout << "4 – Создать стринговую очередь" << endl;

}

/\*\*

\* Вывод в консоль меню второго уровня

\*/

void draw2L() {

cout << "1 – Назад" << endl;

cout << "2 – Добавить элемент" << endl;

cout << "3 – Удалить элемент" << endl;

cout << "4 – Найти элемент" << endl;

}

/\*\*

\* Вывод в консоль подменю добавления

\*/

void draw2LAdd() {

cout << "1 – Назад" << endl;

cout << "2 – Добавить в конец" << endl;

cout << "3 – Добавить в начало" << endl;

}

/\*\*

\* Вывод в консоль подменю удаления

\*/

void draw2LDel() {

cout << "1 – Назад" << endl;

cout << "2 – Удалить из конца" << endl;

cout << "3 – Удалить из начала" << endl;

}

#endif //\_7\_UTILS\_HPP

Код main.cpp:

#include <iostream>

#include <list>

#include <string>

#include "Deque.hpp"

#include "MyException.hpp"

#include "Utils.hpp"

using namespace std;

int main() {

string ch = "2";

while (ch != "1") {

cout << "Добро пожаловать!" << endl;

draw1L();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" || ch == "4") ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< работа с интовой очередью

Deque<int> deq;

while (ch != "1") {

cout << "Интовая очередь:" << endl;

deq.show();

draw2L();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" || ch == "4") ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Подменю добавления

while (ch != "1") {

draw2LAdd();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" ) ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Добаление в конец

cout << "Введите число?" << endl;

string boof;

cin >> boof;

try {

float fl = stof(boof); ///< Попытка преобразовать строку в число

if ( (float) ( (int) fl ) == fl ) { ///< Проверка на целочисленность

int target = fl;

deq.pushBack(target);

} else {

throw MyException();

}

} catch (...) {

cout << "Некорректное число" << endl;

}

}

if (ch == "3") { ///< Добавление в начало

cout << "Введите число?" << endl;

string boof;

cin >> boof;

try {

float fl = stof(boof);

if ( (float) ( (int) fl ) == fl ) {

int target = fl;

deq.pushFront(target);

} else {

throw MyException();

}

} catch (...) {

cout << "Некорректное число" << endl;

}

}

}

ch = "2";

}

if (ch == "3") { ///< Подменю удаления

while (ch != "1") {

draw2LDel();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" ) ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Удаление из конца

try {

if (!deq.isEmpty()) {

deq.popBack();

} else {

throw MyException();

}

} catch (...) {

cout << "Произошла ошибка удаления" << endl;

}

}

if (ch == "3") { ///< Удаление из начала

try {

if (!deq.isEmpty()) {

deq.popFront();

} else {

throw MyException();

}

} catch (...) {

cout << "Произошла ошибка удаления" << endl;

}

}

}

ch = "3";

}

if (ch == "4") { ///< Поиск элемента в очереди

cout << "Что ищем?" << endl;

string boof;

cin >> boof;

try {

float fl = stof(boof);

if ( (float) ( (int) fl ) == fl ) {

int target = fl;

if (deq.isInDeq(target)) {

cout << "Есть такое" << endl;

} else {

cout << "Нет такого" << endl;

}

} else {

throw MyException();

}

} catch (...) {

cout << "Некорректное число" << endl;

}

}

}

ch = "2";

}

if (ch == "3") { ///< Работа с флотовой очередью

Deque<float> deq;

while (ch != "1") {

cout << "Флотовая очередь:" << endl;

deq.show();

draw2L();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" || ch == "4") ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Подменю добавления

while (ch != "1") {

draw2LAdd();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" ) ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Добавление в конец

cout << "Введите число?" << endl;

string boof;

cin >> boof;

try {

float fl = stof(boof);

deq.pushBack(fl);

} catch (...) {

cout << "Некорректное число" << endl;

}

}

if (ch == "3") { ///< Добавление в начало

cout << "Введите число?" << endl;

string boof;

cin >> boof;

try {

float fl = stof(boof);

deq.pushFront(fl);

} catch (...) {

cout << "Некорректное число" << endl;

}

}

}

ch = "2";

}

if (ch == "3") { ///< Подменю удаления

while (ch != "1") {

draw2LDel();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" ) ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Удаление из конца

try {

if (!deq.isEmpty()) {

deq.popBack();

} else {

throw MyException();

}

} catch (...) {

cout << "Произошла ошибка удаления" << endl;

}

}

if (ch == "3") { ///< Удаление из начала

try {

if (!deq.isEmpty()) {

deq.popFront();

} else {

throw MyException();

}

} catch (...) {

cout << "Произошла ошибка удаления" << endl;

}

}

}

ch = "3";

}

if (ch == "4") { ///< Поиск элемента в очереди

cout << "Что ищем?" << endl;

string boof;

cin >> boof;

try {

float fl = stof(boof);

if (deq.isInDeq(fl)) {

cout << "Есть такое" << endl;

} else {

cout << "Нет такого" << endl;

}

} catch (...) {

cout << "Некорректное число" << endl;

}

}

}

ch = "3";

}

if (ch == "4") { ///< Работа со стринговой очередью

Deque<string> deq;

while (ch != "1") {

cout << "Стринговая очередь:" << endl;

deq.show();

draw2L();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" || ch == "4") ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Подменю добавления

while (ch != "1") {

draw2LAdd();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" ) ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Добавление в конец

cout << "Введите строку" << endl;

string boof;

cin >> boof;

deq.pushBack(boof);

}

if (ch == "3") { ///< Добавление в начало

cout << "Введите строку" << endl;

string boof;

cin >> boof;

deq.pushFront(boof);

}

}

ch = "2";

}

if (ch == "3") { ///< Подменю удаления

while (ch != "1") {

draw2LDel();

try {

cin >> ch;

if (!( ch == "1" || ch == "2" || ch == "3" ) ) {

throw MyException();

}

} catch (MyException& e) {

cout << "Вы некорректно выбрали!" << endl;

} catch (exception& e) {

cout << "umm" << endl;

}

if (ch == "2") { ///< Удаление из конца

try {

if (!deq.isEmpty()) {

deq.popBack();

} else {

throw MyException();

}

} catch (...) {

cout << "Произошла ошибка удаления" << endl;

}

}

if (ch == "3") { ///< Удаление из начала

try {

if (!deq.isEmpty()) {

deq.popFront();

} else {

throw MyException();

}

} catch (...) {

cout << "Произошла ошибка удаления" << endl;

}

}

}

ch = "3";

}

if (ch == "4") { ///< Поиск элемента в очереди

cout << "Что ищем?" << endl;

string boof;

cin >> boof;

if (deq.isInDeq(boof)) {

cout << "Есть такое" << endl;

} else {

cout << "Нет такого" << endl;

}

}

}

ch = "4";

}

}

return 0;

}

Вывод программы:

|  |  |
| --- | --- |
| Добро пожаловать!  1 – Выход  2 – Создать интовую очередь  3 – Создать флотовую очередь  4 – Создать стринговую очередь  2  Интовая очередь:  1 – Назад  2 – Добавить элемент  3 – Удалить элемент  4 – Найти элемент  e  Вы некорректно выбрали!  Интовая очередь:  1 – Назад  2 – Добавить элемент  3 – Удалить элемент  4 – Найти элемент  2  1 – Назад  2 – Добавить в конец  3 – Добавить в начало  3  Введите число?  1.6  Некорректное число  1 – Назад  2 – Добавить в конец  3 – Добавить в начало  3  Введите число?  4  1 – Назад  2 – Добавить в конец  3 – Добавить в начало  3  Введите число?  5  1 – Назад  2 – Добавить в конец  3 – Добавить в начало  3  Введите число?  10  1 – Назад  2 – Добавить в конец  3 – Добавить в начало  1 | Интовая очередь:  10 5 4  1 – Назад  2 – Добавить элемент  3 – Удалить элемент  4 – Найти элемент  3  1 – Назад  2 – Удалить из конца  3 – Удалить из начала  2  1 – Назад  2 – Удалить из конца  3 – Удалить из начала  1  Интовая очередь:  10 5  1 – Назад  2 – Добавить элемент  3 – Удалить элемент  4 – Найти элемент  4  Что ищем?  q  Некорректное число  Интовая очередь:  10 5  1 – Назад  2 – Добавить элемент  3 – Удалить элемент  4 – Найти элемент  4  Что ищем?  5  Есть такое  Интовая очередь:  10 5  1 – Назад  2 – Добавить элемент  3 – Удалить элемент  4 – Найти элемент  1  Добро пожаловать!  1 – Выход  2 – Создать интовую очередь  3 – Создать флотовую очередь  4 – Создать стринговую очередь  1  Process finished with exit code 0 |