**Московский государственный технический университет им. Н.Э.Баумана**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| Утверждаю: |  |
| Правдина А.Д. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г. |

**Комплексная лабораторная работа по дисциплине ПКШ**

**“**Система классов словарей и гнезд**”**

Листинг программы

(вид документа)

писчая бумага

(вид носителя)

12

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-41Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Рябкин А.В. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г. |

Москва - 2019

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Содержание**

[Файл Header.h: 3](#__RefHeading___Toc3145_1642094031)

[Файл Source.cpp: 11](#__RefHeading___Toc3147_1642094031)

## Файл Header.h:

#pragma once

#include <iostream>

#include <cstdlib>

#include <algorithm>

#include <fstream>

#include <iomanip>

#include <cmath>

#include <vector>

#include <string>

using namespace std;

char YNflag();

class Nidus {

public:

//Номер гнезда

//int sNumb;

//Значения гнезда

string en, ru;

//Конструктор по умолчанию

Nidus() {

en = "";

ru = "";

}

//Конструктор с параметрами

Nidus(string a, string b) {

en = a;

ru = b;

}

//Деструктор

~Nidus() {};

//Методы

//Вывод полной информации про гнездо

void printNiduses() {

cout << en << " " << ru << endl;

}

string getRu() const {

return ru;

}

string getEn() const {

return en;

}

};

int CheckNumNiduses(vector <Nidus> Paras);

bool SortCompareNidusEn(const Nidus &a, const Nidus &b) {

return a.getEn() < b.getEn();

}

bool SortCompareNidusRu(const Nidus &a, const Nidus &b) {

return a.getRu() < b.getRu();

}

class Dictionary {

public:

//Фамилия автора

string autor;

//Название словаря

string name;

//Год выпуска

int date;

//Количество пар

int amount;

//сами гнезда

vector <Nidus> niduses;

//Конструктор по умолчанию

Dictionary()

{

autor = "";

name = "";

date = 0;

amount = 0;

}

//Конструктор с параметрами1

Dictionary(string autorn, string namen, int daten, int amountn)

{

autor = autorn;

name = namen;

date = daten;

amount = amountn;

}

//Конструктор с параметрами2

Dictionary(string autorn, string namen, int daten, int amountn, vector <Nidus> defdicts)

{

autor = autorn;

name = namen;

date = daten;

amount = amountn;

niduses = defdicts;

}

//Деструктор

~Dictionary() {};

//Вывод одного словаря

void print(){

cout << " Автор: " << autor << "Название: " << name << " Дата выпуска: " << date << " г." << " Количество планируемых пар:" << amount << endl;

cout << "Пары в словаре: " << endl;

if (niduses.size() > 0)

{

for (int i = 0; i < niduses.size(); i++)

{

cout << " Пара №" << i+1 << " ";

niduses[i].printNiduses();

}

}

else cout << "Словарь пуст!" << endl;

cout << endl;

}

//Вывод ТОЛЬКО информации о словаре, без пар

void printS()

{

cout << " Автор: " << autor << " Название: " << name << " Дата выпуска: " << date << " г." << " Количество планируемых пар:" << amount << endl;

cout << endl;

}

//Вывод списка пар в словаре

void printPars()

{

if (niduses.size() > 0)

{

for (int i = 0; i < niduses.size(); i++)

{

cout << " Пара №" << i+1 << " ";

niduses[i].printNiduses();

}

}

else cout << "Словарь пуст!" << endl;

cout << endl;

}

//Добавление гнезда в словарь

void AddPars()

{

cout << endl << "Добавление новой пары в словарь" << endl;

Nidus p;

string e, r;

cout << "Введите первое значение на Английском: ";

//e = getchar();

//getline(cin, p.en);

cin >> p.en;

cout << "Введите второе значение на Русском: ";

//r = getchar();

//getline(cin, p.ru);

cin >> p.ru;

int flag = 1;

for (int i = 0; i < niduses.size(); i++)

{

if ((niduses[i].en == p.en) or (niduses[i].ru == p.ru))

{

cout << "У вас уже имеется данное значение под номером " << i+1 << " - " << niduses[i].en << " = "<< niduses[i].ru<<". Хотите изменить пару? (Y/N)";

if (YNflag() == '1') {

cout << "Введите новое значение на Английском: ";

cin >> p.en;

niduses[i].en = p.en;

cout << "Введите новое значение на Русском: ";

cin >> p.ru;

niduses[i].ru = p.ru;

flag = 0;

break;

}

}

}

if (flag == 1)

{

niduses.push\_back(p);

}

cout << "Пара добавлена в словарь!" << endl;

cout << endl;

}

//Удаление гнезда в словаре

void DeletePars()

{

cout << endl << "Удаление пар из словаря" << endl;

cout << "Вывести список пар? <Y/N>: ";

if (YNflag() == '1')

{

printPars();

}

Nidus p;

char c;

//Флаг проверки

char flag = '1';

//Номер пары в сорваре

int num;

while (flag == '1')

{

c = getchar();

cout << "Введите номер пары, которую хотите удалить: ";

num = CheckNumNiduses(niduses);

for (int i = 0; i < niduses.size(); i++)

{

if (i == num)

{

flag = '0';

niduses.erase(niduses.begin() + i);

vector<Nidus>(niduses).swap(niduses);

cout << "Пара успешно удалена!" << endl;

break;

}

}

if (flag != '0')

cout << "Пара не найдена! Попробуйте ещё раз!" << endl;

}

cout << endl;

}

//Поиск слова в гнездах словаря

int SearchWord()

{

string naidi;

cout << "Введите значение, которое хотите найти: ";

cin >> naidi;

for (int i = 0; i < niduses.size(); i++)

{

if (naidi == niduses[i].ru)

{

cout << "Значение найдено на русском языке, номер гнезда: " << i+1 << endl;

return i;

break;

}

if (naidi == niduses[i].en)

{

cout << "Значение найдено на английском языке, номер гнезда: " << i+1 << endl;

return i;

break;

}

}

cout << endl;

return 0;

}

//Отсортировать значения в словаре по первой букве

void SortNiduses()

{

cout << "Сортировка словаря по первой букве значения. Если желаете отсортировать по русским значениям, введите 1, если по английским, введите 2, и любое другое значение, если хотите выйти: ";

int choise;

cin >> choise;

if (choise == 1) {

sort(niduses.begin(), niduses.end(), SortCompareNidusRu);

}

if (choise == 2) {

sort(niduses.begin(), niduses.end(), SortCompareNidusEn);

}

}

vector <Nidus> NidusesToSend()

{

vector <Nidus> nidusesToSendList;

//вывести список пар и отправить диапазон пар

cout << "Вывести список пар для выбора перемещаемых пар? <Y/N> (Если нет - будут перемещены все пары): ";

if (YNflag() == '1')

{

printPars();

cout << "Введите начало диапазона пар для переноса: ";

int zn1;

zn1 = CheckNumNiduses(niduses);

cout << "Введите конец диапазона пар для переноса: ";

int zn2;

zn2 = CheckNumNiduses(niduses);

for (int i = 0; i < niduses.size(); i++)

{

if ((i > zn1) and (i < zn2))

{

nidusesToSendList.push\_back(niduses[i]);

niduses.erase(niduses.begin() + i);

vector<Nidus>(niduses).swap(niduses);

}

}

}

else {

for (int i = 0; i < niduses.size(); i++) {

nidusesToSendList.push\_back(niduses[i]);

niduses.erase(niduses.begin() + i);

vector<Nidus>(niduses).swap(niduses);

}

}

return nidusesToSendList;

}

int GetDictionaryInfo()

{

if (niduses.size() > 0)

return 1;

return 0;

}

//Перегрузка сложения

void operator+(Dictionary &S2)

{

for (int i = 0; i < S2.niduses.size(); i++)

{

niduses.push\_back(S2.niduses[i]);

}

//вопрос характеристик обьединенного словаря

//this->autor = S2.autor;

//this->name = S2.name;

//this->amount = S2.amount;

//this->date = S2.date;

}

};

//----------------------------Функции--------------------------

int CheckNumNiduses(vector <Nidus> Paras)

{

int flag = 1;

int num;

while (flag == 1)

{

cin >> num;

if ((num - 1 <= Paras.size()) && (num > 0))

{

flag = 0;

return num - 1;

}

else cout << "Ошибка! Такой пары не существует. Введите заново: " << endl;

}

return num - 1;

}

int CheckNum(vector <Dictionary> Dictionaries)

{

int flag = 1;

int num;

while (flag == 1)

{

cin >> num;

if ((num - 1 <= Dictionaries.size()) && (num > 0))

{

flag = 0;

return num - 1;

}

else cout << "Ошибка! Такого словаря не существует. Введите заново: " << endl;

}

return num - 1;

}

//Вывод списка словарей

void PrintDictionaries(vector <Dictionary> Dictionaries)

{

int i = 0;

cout << endl << "Список имеющихся словарей: " << endl;

if (Dictionaries.size() != 0)

{

for (i; i < Dictionaries.size(); i++)

{

cout << i+1 << ". ";

Dictionaries[i].printS();

cout << endl;

}

//cout << endl;

}

else cout << "На данный момент ни одного словаря в БД нет" << endl << endl;

}

//Создавние нового словаря

void CreateDictionary(vector <Dictionary>& Dictionaries)

{

cout << endl << "Добавление нового словаря в БД" << endl;

string name, autor;

int date, amount;

cout << "Введите фамилию автора: ";

cin >> autor;

cout << "Введите наименование словаря: ";

cin >> name;

cout << "Введите год выпуска словаря: ";

cin >> date;

cout << "Введите количество пар в словаре: ";

cin >> amount;

Dictionary s(autor, name, date, amount);

Dictionaries.push\_back(s);

cout << "Был создан словарь " << name << endl;

cout << endl;

}

//Удаление словаря

void DeleteDictionary(vector <Dictionary>& Dictionaries)

{

cout << endl << "Удаление словаря из БД" << endl;

cout << "Вывести список словарей? <Y/N>?: ";

if (YNflag() == '1')

{

PrintDictionaries(Dictionaries);

}

char flag = '1';

int num;

while (flag == '1')

{

cout << "Введите номер словаря, который хотите удалить из БД или введите 0 чтобы перейти в меню: ";

cin >> num;

if ((num - 1 <= Dictionaries.size()) && (num > 0))

{

if (Dictionaries[num - 1].GetDictionaryInfo() >= 1)

{

cout << "В словаре имеются пары. Всё равно удалить? <Y/N>?: ";

if (YNflag() == '1')

{

Dictionaries.erase(Dictionaries.begin() + num - 1);

vector<Dictionary>(Dictionaries).swap(Dictionaries);

flag = '0';

cout << "Выход в меню..." << endl;

}

flag = '0';

cout << "Выход в меню..." << endl;

}

else

{

Dictionaries.erase(Dictionaries.begin() + num - 1);

vector<Dictionary>(Dictionaries).swap(Dictionaries);

flag = '0';

//cout << "Выход в меню..." << endl;

}

}

else

{

if (num == 0)

{

flag = '0';

cout << "Выход в меню..." << endl;

}

else cout << "Ошибка! Такого словаря в БД не существует" << endl;

}

}

}

//Работа с отдельно взятым словарем

void ChangeDictionary(vector <Dictionary> & Dictionaries)

{

cout << endl << "Обработка данных одного словаря" << endl;

cout << "Вывести список словарей <Y/N>?: ";

if (YNflag() == '1')

{

PrintDictionaries(Dictionaries);

}

//Номер гнезда

int num;

cout << "Введите номер словаря: ";

num = CheckNum(Dictionaries);

int flag = 1;

int flagMenu = 1;

int flagMenuIn = -1;

while (flag != 0)

{

cout << "Операции с словарями: " << endl;

cout << "1. Просмотреть список пар в словаре" << endl;

cout << "2. Добавить новую пару" << endl;

cout << "3. Убрать пару из словаря" << endl;

cout << "4. Найти значение в словаре" << endl;

cout << "5. Отсортировать значения в словаре по первой букве" << endl;

cout << "6. Выйти в главное меню" << endl;

cout << "Выберите пункт меню (1|2|3|4|5|6): ";

cin >> flagMenuIn;

switch (flagMenuIn)

{

//Просмотреть список пар в словаре

case 1:

Dictionaries[num].printPars();

break;

//Добавить новую пару

case 2:

Dictionaries[num].AddPars();

break;

//Убрать пару из словаря

case 3:

Dictionaries[num].DeletePars();

break;

//Найти значение в словаре

case 4:

Dictionaries[num].SearchWord();

break;

//Отсортировать значения в словаре по первой букве

case 5:

Dictionaries[num].SortNiduses();

break;

//Выход в главное меню

case 6:

flag = 0;

cout << endl;

break;

default:

cout << "Ошибка ввода! Такого пункта не существует!" << endl;

break;

}

}

}

void MergeDictionaries(vector <Dictionary> & Dictionaries)

{

cout << endl << "Объединение двух словарей" << endl;

cout << "Вывести список словарей? <Y/N>: ";

if (YNflag() == '1')

{

PrintDictionaries(Dictionaries);

}

//Номер первого словаря

int num1;

cout << "Введите номер первого словаря: ";

num1 = CheckNum(Dictionaries);

//Номер второго словаря

int num2;

cout << "Введите номер второго словаря: ";

num2 = CheckNum(Dictionaries);

Dictionaries[num1] + Dictionaries[num2];//перегрузкааааааа

Dictionaries.erase(Dictionaries.begin() + num2);

vector<Dictionary>(Dictionaries).swap(Dictionaries);

cout << "Обьединение словарей успешно завершено!" << endl << endl;

}

void SeparateDictionaries(vector <Dictionary>& Dictionaries)

{

cout << endl << "Создание нового словаря на основе старого(разделение)" << endl;

cout << "Вывести список словарей? <Y/N>: ";

if (YNflag() == '1')

{

PrintDictionaries(Dictionaries);

}

//Номер первого словаря

int num;

cout << "Введите номер разделяемого словаря: ";

num = CheckNum(Dictionaries);

string name, autor;

int date, amount;

cout << "Введите характеристики нового словаря: " << endl;

cout << "Введите фамилию автора: ";

cin >> autor;

cout << "Введите наименование словаря: ";

cin >> name;

cout << "Введите год выпуска словаря: ";

cin >> date;

cout << "Введите количество пар в словаре: ";

cin >> amount;

Dictionary s(autor, name, date, amount, Dictionaries[num].NidusesToSend());

Dictionaries.push\_back(s);

cout << "Был создан новый словарь: " << name << endl;

cout << endl;

}

//YES-NO flag

char YNflag()

{

char c = '0';

char r;

char flag = '1';

while (flag == '1')

{

cin >> c;

if (c == 'N')

{

flag = '0';

r = '0';

}

else {

if (c == 'Y')

{

flag = '0';

r = '1';

}

else cout << "Ошибка! Введите только заглавные Y(es) или N(o): ";

}

}

return r;

}

## Файл Source.cpp:

#include "Header.h"

using namespace std;

//Добавление словаря

void CreateDictionary(vector <Dictionary>& Dictionaries);

//Вывод списка словарей

void PrintDictionaries(vector <Dictionary> Dictionaries);

//Удаление словаря

void DeleteDictionary(vector <Dictionary>& Dictionaries);

//Объединение двух словарей

void MergeDictionaries(vector <Dictionary>& Dictionaries);

//Разделение словарей

void SeparateDictionaries(vector <Dictionary>& Dictionaries);

//Работа с отдельно взятым словарем

void ChangeDictionary(vector <Dictionary>& Dictionaries);

int main()

{

system("chcp 1251 >> null");

cout << "Система словарей и гнезд(пар значений)" << endl;

cout << "Разработал: Рябкин А.В" << endl;

//Вектор для хранения всех словарей

vector <Dictionary> Dictionaries;

char flagMain = '1';

//char flagMenu = '1';

int flagMenuIn = -1;

//Меню

while (flagMain != '0')

{

cout << "===Меню операций с словарями===" << endl;

cout << "1. Просмотреть список словарей" << endl;

cout << "2. Добавить новый словарь в БД" << endl;

cout << "3. Удалить словарь из БД" << endl;

cout << "4. Объединить два словаря" << endl;

cout << "5. Разделить словарь" << endl;

cout << "6. Работа со словарями" << endl;

cout << "7. Выйти из программы" << endl;

cout << "Выберите пункт меню (1|2|3|4|5|6|7): ";

cin >> flagMenuIn;

switch (flagMenuIn)

{

//Вывод списка словарей

case 1:

PrintDictionaries(Dictionaries);

flagMenuIn = -1;

break;

//Добавление нового словаря

case 2:

CreateDictionary(Dictionaries);

flagMenuIn = -1;

break;

//Удаление словаря

case 3:

DeleteDictionary(Dictionaries);

flagMenuIn = -1;

break;

//Объединение двух словарей

case 4:

MergeDictionaries(Dictionaries);

flagMenuIn = -1;

break;

//Разделение словарей

case 5:

SeparateDictionaries(Dictionaries);

flagMenuIn = -1;

break;

//Работа со словарем

case 6:

ChangeDictionary(Dictionaries);

flagMenuIn = -1;

break;

//Выход из программы

case 7:

flagMain = '0';

break;

//Проверка правильности выбора пункта меню

default:

cout << endl << "Ошибка ввода! Такого пункта не существует!" << endl << endl;

break;

}

}

cout << "Работа завершена! ";

system("PAUSE");

return 0;

}