**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчЁт

**по лабораторной работе № 3**

**по дисциплине «Программирование»**

|  |  |  |  |
| --- | --- | --- | --- |
| Студент гр. 4354 |  |  | И.В. Чучалин |
| Преподаватель |  |  | В.А. Калмычков |

# Оглавление

[Оглавление 2](#_Toc192508697)

[Задание 3](#_Toc192508698)

[Неясности, предложения их решения и уточнения 3](#_Toc192508699)

[Контрольные примеры 3](#_Toc192508700)

[Организация интерфейса пользователя 3](#_Toc192508701)

[Организация входного файла 3](#_Toc192508702)

[Организация выходного файла 3](#_Toc192508703)

[Организация консоли 3](#_Toc192508704)

[Реализация ввода/вывода 3](#_Toc192508705)

[Внутреннее представление данных 4](#_Toc192508706)

[Описание внутренних функций 4](#_Toc192508707)

[Определение функций 4](#_Toc192508708)

[Назначение функций 5](#_Toc192508709)

[Описание алгоритма 5](#_Toc192508710)

[Общее описание 5](#_Toc192508711)

[Блок-схема 5](#_Toc192508712)

[Текст программы 8](#_Toc192508713)

[Результат работы программы 11](#_Toc192508714)

[Вывод 14](#_Toc192508715)

# Задание

Задан текст, представленный списком. Необходимо удалить все элементы со значениями, отличающимися от заданного значения.

# Контрольные примеры

Заданное значение: okay wow

1. Заданное значение есть в тексте.

Входные данные: Полученный результат:

okay wow dont read

dont read this text

this text or actually you can try

or actually you can try you did it

okay wow

you did it wait wait wait

okay wow okay stop

wait wait wait

okay stop

2. Заданного значения нет в тексте.

Входные данные: Полученный результат:

Hello world Hello world

Sunny day Sunny day

# Организация интерфейса пользователя

## Организация входного файла

О1. Текст.

## Организация выходного файла

1. Вывод заданного значения.
2. Вывод считанного текста.
3. Вывод количества строк.
4. Вывод информации о маркере.
5. Вывод обработанного текста.
6. Вывод количества элементов списка.

## Организация консоли

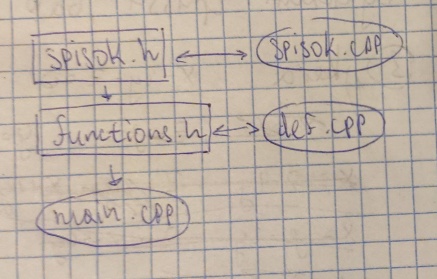
О1. Вывод ошибок.

# Реализация ввода/вывода

Использованы потоки ввода (cin>>) и вывода (cout<<) библиотеки «iostream», а также функции объекта cin: getline, clear, eof.

Использованы потоки чтения из файла (>>) и записи в файл (<<) библиотеки «fstream», а также функции объекта типа fstream: open, is\_open, fail, seekg, eof, close. Для настройки ввода были использованы функции setiosflags, resetiosflags библиотеки «iomanip».

# Модульное представление



# Внутреннее представление данных

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Переменная | Назначение |
| SPISOK | class | Node\* head | Головной элемент списка |
| Node\* tail | Хвостовой элемент списка |
| int size | Размер |
| Node | struct | char\* data | Дата |
| Node\* next | Следующий |

|  |  |  |
| --- | --- | --- |
| Имя переменной | Тип | Назначение |
| inputFile, outputFile | const char\* | Имена файлов обработки |
| match, showInfo | bool | Вспомогательная переменная |
| i, len, initialSize, removedCount | int |
| toDelete, this, newNode | Node\* |
| temp |
| current | Текущий элемент |
| prev | Предыдущий элемент |
| buffer[] | char | Буфер |
| lineCount | int | Счет строк |
| markerFound, foundMarker | bool | Истинность маркера |
| spisok | SPISOK | Список |
| out, in | ofstream, ifstream | Выходной и входной файлы |
| marker | char | Маркер |
| valueToSave[] | Заданное значение |

# Описание внутренних функций

## Определение функций

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Принадлежность | Имя функции | Аргументы | | | | Возвращаемое значение |
| Входные | Выходные | Модифи-цируемые | Транзитные |
| - | readMarker |  |  |  |  |  |
| processFile |  |  |  |  |  |
| Class SPISOK | SPISOK |  |  |  |  |  |
| ~SPISOK | - |  |  |  |  |
| Push\_back |  |  |  |  |  |
| Clear | - |  |  |  |  |
| getSize | - |  |  |  |  |
| removeAllExcept |  |  |  |  |  |
| printToFile |  |  |  |  |  |
| Struct Node | Node |  |  |  |  |  |
| ~Node | - |  |  |  |  |
| - | main | - |  |  |  |  |

## Назначение функций

# Описание алгоритма

## Общее описание

## Блок-схема

# Текст программы

1. main.cpp

#include "functions.h"

#include <iostream>

using namespace std;

void main()

{

const char\* inputFile = "in.txt";

const char\* outputFile = "out.txt";

processFile(inputFile, outputFile);

cout << "processing completed" << endl;

}

2. spisok.h

#pragma once

#include <fstream>

using namespace std;

class SPISOK

{

public:

SPISOK();

~SPISOK();

void push\_back(const char\* data);

void clear();

int getSize() const;

void removeAllExcept(const char\* value);

void printToFile(ofstream& out, bool showInfo) const;

private:

struct Node

{

char\* data;

Node\* next;

Node(const char\* data, Node\* next = nullptr);

~Node();

};

Node\* head;

Node\* tail;

int size;

};

3. spisok.cpp

#include "spisok.h"

#include <iostream>

SPISOK::SPISOK() : head(nullptr), tail(nullptr), size(0) {}

SPISOK::~SPISOK()

{

clear();

}

int SPISOK::getSize() const

{

return size;

}

void SPISOK::clear()

{

while (head)

{

Node\* temp = head;

head = head->next;

delete temp;

}

tail = nullptr;

size = 0;

}

void SPISOK::removeAllExcept(const char\* value)

{

if (!head || !value) return;

Node\* current = head;

Node\* prev = nullptr;

while (current)

{

bool match = true;

int i = 0;

while (value[i] != '\0' && current->data[i] != '\0')

{

if (value[i] != current->data[i])

{

match = false;

break;

}

i++;

}

if (match && value[i] == '\0' && current->data[i] == '\0')

{

prev = current;

current = current->next;

}

else

{

Node\* toDelete = current;

current = current->next;

if (toDelete == head)

{

head = current;

if (!head) tail = nullptr;

}

else

{

prev->next = current;

if (toDelete == tail) tail = prev;

}

delete toDelete;

size--;

}

}

}

SPISOK::Node::Node(const char\* data, Node\* next)

{

int len = 0;

while (data[len] != '\0') len++;

this->data = new char[len + 1];

for (int i = 0; i <= len; i++)

{

this->data[i] = data[i];

}

this->next = next;

}

SPISOK::Node::~Node()

{

delete[] data;

}

void SPISOK::push\_back(const char\* data)

{

if (!data) return;

Node\* newNode = new Node(data);

if (!head)

{

head = tail = newNode;

}

else

{

tail->next = newNode;

tail = newNode;

}

size++;

}

void SPISOK::printToFile(ofstream& out, bool showInfo) const {

if (!out) return;

if (showInfo)

{

out << "contents of the list:\n\n";

}

Node\* current = head;

while (current)

{

out << "->" << current->data << '\n';

current = current->next;

}

if (showInfo)

{

out << "\nnumber of elements in the list: " << size << '\n';

}

}

4. functions.h

#pragma once

using namespace std;

#include "spisok.h"

#include <fstream>

bool readMarker(ifstream& in, ofstream& out, SPISOK& spisok, const char marker);

void processFile(const char\* inputFile, const char\* outputFile);

5. def.cpp

#include "functions.h"

#include <iostream>

bool readMarker(ifstream& in, ofstream& out, SPISOK& spisok, const char marker)

{

char buffer[256];

int lineCount = 0;

bool markerFound = false;

while (in.getline(buffer, 255))

{

int i = 0;

bool foundMarker = false;

while (buffer[i] != '\0')

{

if (buffer[i] == marker)

{

foundMarker = true;

buffer[i] = '\0';

break;

}

i++;

}

if (foundMarker)

{

markerFound = true;

}

spisok.push\_back(buffer);

out << "-> " << buffer << '\n';

lineCount++;

if (markerFound) break;

}

out << "\nnumber of readed lines: " << lineCount << '\n';

return markerFound;

}

void processFile(const char\* inputFile, const char\* outputFile)

{

SPISOK spisok;

ofstream out(outputFile);

if (!out)

{

cerr << "out file is not open" << endl;

return;

}

ifstream in(inputFile);

if (!in)

{

out << "in file is not open" << endl;

out.close();

return;

}

char marker;

in >> marker;

in.ignore();

while (in.peek() == ' ') in.get();

char valueToSave[256];

in.getline(valueToSave, 255);

out << "target value: \"" << valueToSave << "\"\n\n";

out << "file data:\n\n";

if (!readMarker(in, out, spisok, marker))

{

out << "marker not found" << endl;

}

out << "\nresult of processing:\n\n";

int initialSize = spisok.getSize();

spisok.removeAllExcept(valueToSave);

int removedCount = initialSize - spisok.getSize();

spisok.printToFile(out, true);

out.close();

}

# Результат работы программы

1. Есть заданное значение.

Входной файл:

@

okay wow

dont read

this text

or actually you can try

okay wow

you did it

okay wow

wait wait wait

okay stop

Выходной файл:

target value: "okay wow"

file data:

-> dont read

-> this text

-> or actually you can try

-> okay wow

-> you did it

-> okay wow

->

-> wait wait wait

-> okay stop

number of readed lines: 9

marker not found

result of processing:

contents of the list:

->okay wow

->okay wow

number of elements in the list: 2

1. Нет заданного значения.

Входной файл:

@

okay wow

dont read

this text

or actually you can try

you did it

wait wa@it wait

okay stop

Выходной файл:

target value: "okay wow"

file data:

-> dont read

-> this text

-> or actually you can try

->

-> you did it

->

-> wait wa

number of readed lines: 7

result of processing:

contents of the list:

number of elements in the list: 0

1. Есть маркер.

Входной файл:

@

okay wow

dont read

this text

or actually you can try

okay wow

you did it

okay wow

wait wa@it wait

okay stop

Выхолной файл:

target value: "okay wow"

file data:

-> dont read

-> this text

-> or actually you can try

-> okay wow

-> you did it

-> okay wow

->

-> wait wa

number of readed lines: 8

result of processing:

contents of the list:

->okay wow

->okay wow

number of elements in the list: 2

# Вывод

Были изучены дополнительные возможности классов в языке программирования C++, а также была проведена работа со списками.