МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

### федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

(ГУАП)

КАФЕДРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СЕТЕЙ

Преподаватель

канд. техн. наук, доцент Л.Н. Бариков

Отчёт

по лабораторной работе №14

по дисциплине ОСНОВЫ ПРОГРАММИРОВАНИЯ

на тему: «Линейные списки»

Работу выполнил

студент гр. 4941 Н.С. Горбунов

Санкт-Петербург

2020

**Цель лабораторной работы**: изучение способов создания и принципов использования односвязных линейных списков; изучение стандартных средств языка C/C++ для работы с динамической памятью; совершенствование навыков модульного программирования на языке C/C++ при решении задач обработки линейных списков; изучение способов разработки многофайловых проектов.

**Задание на программирование**: используя технологию процедурного программирования разработать программу обработки односвязных линейных списков с числом элементов в списке не менее десяти в соответствии с индивидуальным заданием.

**Вариант 13**

Перевернуть исходный список *L*, то есть изменить ссылки в этом списке так, чтобы его элементы оказались расположенными в обратном порядке. Дополнительный список не использовать.

**Математическая модель**

Для начала необходимо реализовать односвязный список. Создан класс *List* – сам список, в нем класс *mem* – ячейка списка. Mem содержит в себе поле типа *elem* и типа ссылка на *mem*. Конструктор класса mem принимает параметры типа *elem* и типа ссылка на *mem*, выполняя заполнение полей нового объекта соответственно.

В рамках класса *List* реализованы:

* поле *head* – указатель на первый элемент списка.
* Конструктор класса, в котором инициализируется *head*
* Функция *endPasteNew* принимает параметр *newData* типа *elem* вставляет в конец списка новый элемент. Перебирается список, пока не будет найден элемент, с полем *link* равный NULL, создается новая ячейка, ее адрес записывается, в нее записывается *newData*.
* Функция *showList* выводит в консоль элементы списка через пробел.
* Функция *Remove* принимает индекс элемента списка и удаляет его.
* Функция *Transposition* транспонирует список. Перебирается список, нулевому элементу записывается в *link NULL*. А каждому последующему следующий. В конце *head* присваивается последний

**Текст программы**

**Заголовочный файл List.h**

//Текст файла List.h

#pragma once

typedef char elem;

class List

{

private:

class mem

{

public:

elem data;

mem\* link;

mem(mem\* newLink, elem newData)

{

data = newData;

link = newLink;

}

};

mem\* head;

public:

List();

void endPasteNew(elem newData);

void showList();

void Transposition();

void Delete();

};

**Основной файл List.cpp**

//Текст файла List.cpp

#include "List.h"

#include <iostream>

List::List()

{

this->head = NULL;

}

void List::endPasteNew(elem newData) //добавить в конец списка новый элемент

{

if (this->head != NULL) // если список не пустой

{

mem\* select = this->head; // переменная выбранного элемента списка

while (select->link != NULL) // пока не дойдем до последнего элемента

{

select = select->link; // идем по списку

}

select->link = new mem(NULL, NULL); // создаем новую ячейку и записываем ее адрес в последнюю ячейку

select->link->data = newData; // записываем данные в новую ячейку

}

else this->head = new mem(NULL, newData); // иначе вставляем значение в первую ячейку

}

void List::showList() //вывод списка в консоль

{

std::cout << "\n";

if (this->head != NULL) // если список не пустой

{

mem\* select = this->head; // переменная выбранного элемента списка

do

{

std::cout << select->data /\*<< " "\*/; //вывести значение ячейки

select = select->link; // перейти на следующую

} while (select != NULL); // пока есть элемент

}

else std::cout << "NULL"; // иначе вывести в консоль NULL

}

void List::Transposition()

{

if (this->head != NULL)

{

mem\* select = this->head;

mem\* head2 = this->head;

mem\* last;

while (head2->link != NULL)

{

select = head2;

head2 = head2->link;

}

head2->link = select;

while (this->head->link->link != this->head)

{

select = this->head;

do

{

last = select;

select = select->link;

} while (select->link->link != select);

select->link = last;

}

this->head->link = NULL;

this->head = head2;

}

}

void List::Delete() // функция освобождение динамической памяти. Объект списка не удаляется!

{

mem\* select = this->head->link;

mem\* next;

this->head = NULL;

do

{

next = select->link;

delete select;

select = next;

} while (select != NULL);

}

**Основной файл Program.cpp**

//Текст файла Program.cpp

#include <iostream>

#include "List.h"

#include <string>

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

cout << "Программа демонстрирует транспонирование односвязного списка\n Введите символы последовательно, точка явлется завершением ввода\n";

char word;

List list = List();

do

{

cin >> word;

if (word != '.')

{

list.endPasteNew(word);

}

} while (word != '.');

cout << "исходный список: ";

list.showList();

list.Transposition();

cout << "\nтранспонированный список: ";

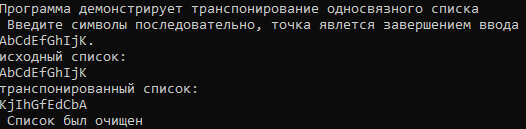
list.showList();

list.Delete();

cout << "\n Список был очищен";

}

**Примеры работы программы**

****

**Вывод**: используя технологию процедурного программирования разработал программу обработки односвязных линейных списков с числом элементов в списке не менее десяти в соответствии с индивидуальным заданием. Разработанная программа транспонирует односвязный список.