## лабораторная работа №1

«**ИССЛЕДОВАНИЕ ОСОБЕННОСТЕЙ ОБЪЕКТНОГО ПОДХОДА ПРИ ПРОГРАММИРОВАНИИ СЛОЖНЫХ СТРУКТУР ДАННЫХ С ДИНАМИЧЕСКИМИ ПОЛЯМИ»**

1. Цель работы

Исследование основных средств определения класса, создания объектов класса, приобретение навыков разработки и отладки программ, использующих динамическую память. Исследование особенностей использования конструкторов копирования.

Задача. Вариант 4

Необходимо для заданного по варианту динамического типа данных описать класс, содержащий указатель на динамический тип как поле данных. Для этого класса описать конструкторы (не менее трех, в том числе и конструктор копирования), деструктор, функцию печати данных. Создать экземпляр полученного класса и проиллюстрировать его корректную работу: распечатать данные, изменить данные и распечатать вновь. Создать второй экземпляр класса как копию первого и проиллюстрировать корректную работу конструктора копирования: распечатать и изменить данные объекта-копии, распечатать данные обоих объектов, сравнить результат. Предусмотреть обработку ошибок при манипуляции с данными.

Динамическая структура — циклическая очередь. В ней хранятся данные о возрасте: ФИО (строка) и возраст (число). Предусмотреть функции добавления элементов в очередь и удаления из нее, а также функцию поиска по фамилии.

**Ход работы:**

# **Текст программы С++:**

#include <iostream>

#include <windows.h>

using namespace std;

struct spisok

{

string FIO;

int age;

};

struct line

{

spisok inf;

line \*next;

};

class List

{

private:

line \*l, \*r;

public:

List();

List(line \*tempL);

void write();

List(const List \*ob);

void add();

void del();

void find();

~List();

};

void menu();

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int c;

List \*Original = 0, \*Copy = 0;

line \*tempL = new line;

menu();

do

{

cin>>c;

switch(c)

{

case 1:

if (Original == 0)

{

cout<<" Enter(1) to add elem with parameters"<<endl;

cout<<"Enter(2) to add elem without parameters"<<endl;

cin>>c;

if (c == 1)

{

cout<<" Enter FIO and age"<<endl;

cin>>tempL->inf.FIO>>tempL->inf.age;

Original = new List(tempL);

menu();

break;

}

if(c == 2)

{

Original = new List();

menu();

break;

}

}

else

{

cout<<" Error"<<endl;

menu();

break;

}

case 2:

if (Copy != 0)

{

cout<<" Original(1) or copy(2)"<<endl;

cin>>c;

if (c == 1)

{

if (Original != 0) Original->add();

else cout<<" Error"<<endl;

}

else

{

if (Copy != 0) Copy->add();

else cout<<" Error"<<endl;

}

menu();

break;

}

else

{

if (Original != 0) Original->add();

else cout<<" Error"<<endl;

menu();

break;

}

case 3:

if (Copy != 0)

{

cout<<" Original(1) or copy(2)"<<endl;

cin>>c;

if (c == 1)

{

if (Original != 0) Original->write();

else cout<<" Error"<<endl;

}

else

{

if (Copy != 0) Copy->write();

else cout<<" Error"<<endl;

}

menu();

break;

}

else

{

if (Original != 0) Original->write();

else cout<<" Error"<<endl;

menu();

break;

}

case 4:

if (Copy != 0)

{

cout<<" Original(1) or copy(2)"<<endl;

cin>>c;

if (c == 1)

{

if (Original != 0) Original->del();

else cout<<" Error"<<endl;

}

else

{

if (Copy != 0) Copy->del();

else cout<<" Error"<<endl;

}

menu();

break;

}

else

{

if (Original != 0) Original->del();

else cout<<" Error"<<endl;

menu();

break;

}

case 5:

if (Copy == 0)

{

Copy = new List(Original);

menu();

break;

}

else

{

cout<<" Error"<<endl;

menu();

break;

}

case 6:

if (Copy != 0)

{

cout<<" Original(1) or copy(2)"<<endl;

cin>>c;

if (c == 1)

{

if (Original != 0) Original->find();

else cout<<" Error"<<endl;

}

else

{

if (Copy != 0) Copy->find();

else cout<<" Error"<<endl;

}

menu();

break;

}

else

{

if (Original != 0) Original->find();

else cout<<" Error"<<endl;

menu();

break;

}

default:

cout<<" Error"<<endl;

menu();

break;

}

}

while (c != 7);

}

void menu()

{

cout<<endl;

cout<<" 1- Creation "<<endl;

cout<<" 2- Add "<<endl;

cout<<" 3- Cout"<<endl;

cout<<" 4- Delete"<<endl;

cout<<" 5- Copy"<<endl;

cout<<" 6- Search FIO"<<endl;

cout<<" 7- Exit"<<endl;

}

List::List()

{

l = r = 0;

cout<<" Constructor"<<endl;

}

List::List(line \*tempL)

{

l = r = tempL;

l->next = r->next = 0;

cout<<" Constructor with parameters"<<endl;

}

void List::write()

{

if (l == 0) cout<<" Empty line"<<endl;

else

{

line \*t = l;

cout<<" List:"<<endl;

while (t != 0)

{

cout<<t->inf.FIO<<" ";

cout<<t->inf.age<<endl;

t = t->next;

}

}

}

List::List(const List \*Original)

{

line \*t = Original->l, \*t1, \*t2;

if (Original->l == 0)

{

l = r = 0;

cout<<" Copy constructor"<<endl;

}

else if (Original->l->next == 0)

{

t1 = new line;

t1->inf = t->inf;

t1->next = 0;

l = r = t1;

cout<<" Copy constructor"<<endl;

}

else

{

t1 = new line;

t1->inf = t->inf;

t1->next = 0;

l = r = t1;

for(t; t != 0; t = t->next)

{

t2 = new line;

t2->inf = t->inf;

t2->next = 0;

r = t2;

t1->next = t2;

t1 = t2;

}

cout<<" Copy constructor"<<endl;

}

}

void List::add()

{

if (l == 0)

{

line \*t = new line;

cout<<" Enter FIO and age"<<endl;

cin>>t->inf.FIO>>t->inf.age;

l = r = t;

l->next = r->next = 0;

}

else

{

line \*t = new line;

cout<<" Enter FIO and age"<<endl;

cin>>t->inf.FIO>>t->inf.age;

t->next = 0;

r->next = t;

r = t;

}

}

void List::del()

{

if (l == 0)

cout<<" Empty line"<<endl;

else if (l->next == 0)

{

delete l;

r = l = 0;

}

else

{

line \*t = l;

l = l->next;

delete t;

}

}

void List::find()

{

if (l == 0)

cout<<" Empty line"<<endl;

else

{

cout<<" Enter FIO"<<endl;

string s;

cin>>s;

line \*t = l;

cout<<" Ages:"<<endl;

for(t; t != 0; t = t->next)

{

if (s == t->inf.FIO)

{

cout<<t->inf.age<<endl;

}

}

}

}

List::~List()

{

if (l != 0)

{

line \*t1 = l, \*t2;

if (t1->next == 0) delete t1;

else

{

t2 = l->next;

while (t2->next != 0)

{

t1->next = 0;

delete t1;

t1 = t2;

t2 = t2->next;

}

t1->next = 0;

delete t1;

delete t2;

}

l = r = 0;

}

cout<<"Destructor"<<endl;

}

**Тестирование программы**

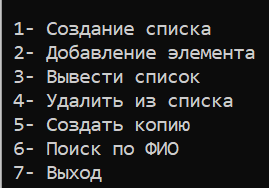


Рисунок 1 ‒ Консольное меню

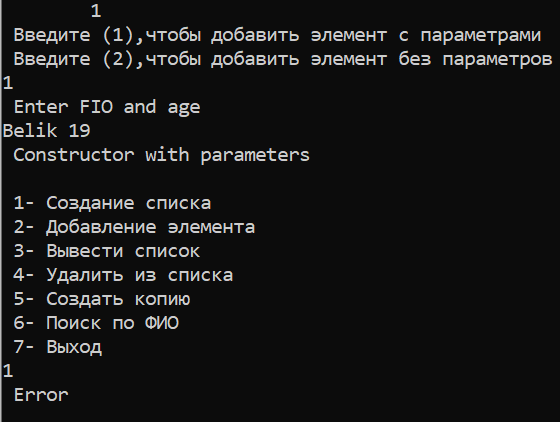


Рисунок 2 ‒ Добавление элемента с параметрами

# 

Рисунок 3 ‒ Добавление второго элемента, поиск по ФИО

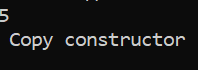


Рисунок 4 ‒ Создание копии

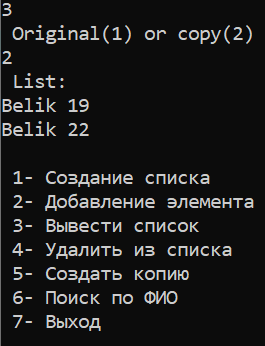


Рисунок 5 ‒Вывод списка копии

# C:\Users\Георгий\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Снимок экрана (49).png

Рисунок 6 ‒Поиск возрастов по ФИО иовув и Belik

Вывод: В ходе данной работы были исследованы основные средства определения класса, созданы объектов класса, приобретены навыки разработки и отладки программ, использующих динамическую память. Исследованы особенности использования конструкторов копирования.