

Автор: Татаренко А., КІТ-119а

Дата: 17 червня 2020

Лабораторна робота №6.

СПАДКУВАННЯ

Тема. Класи. Спадкування.

Мета – отримати знання про парадигму ООП – спадкування. Навчитися застосовувати отримані знання на практиці.

1 Завдання до роботи

Індивідуальне завдання 19.

Модернізувати попередню лабораторну роботу шляхом:

- додавання класу-спадкоємця, котрий буде поширювати функціонал «базового класу» відповідно до індивідуального завдання;
- додавання ще одного класу-списку, що буде керувати лише елементами класу-спадкоємця;
- у функціях базового класу та класу-спадкоємця обов'язкове використання ключових слів `final` та `override`.

2 Розробка алгоритму розв'язання задачі.

2.1 Опис змінних

```
Arr stud_array; class Student; class Arr;
```

Класи, методи, функції, конструктори

3 Код програми

audience.h

```
#pragma once
typedef short sint;

class Aud
{
private:
    sint aud_num;

public:
    sint get_aud_num() const;
    void set_aud_num(sint);

    Aud();
    Aud(sint);
    Aud(const Aud& other);
    ~Aud();
};
```

basic_class.h

```
#define _CRTDBG_MAP_ALLOC
#include <stdlib.h>
#include <crtDBG.h>

#include <iostream>
#include <fstream>
#include <sstream>
#include <regex>

#include "audience.h"
#include "faculty.h"

#define E1 3

using std::cout;
using std::cin;
using std::endl;
using std::stringstream;
using std::ofstream;
using std::ifstream;
using std::regex;

typedef bool (comp)(const int&, const int&);

class Student
{
private:
    int age;
    int number_stud;
    int middle_mark;
    string name;
    bool debt;
    int prog_d;
    Aud audience;
    Fac faculty;

public:
```

```

Student();

Student(int a, int n, int m, string na, bool d, int pd, sint ad, string fc);

Student(const Student &other);

int get_number_stud() const;

int get_age() const;

int get_middle_mark() const;

string get_name() const;

bool get_debt() const;

int get_prog_d() const;

sint get_aud();

string get_fac();

~Student()
{
    cout << "Вызвался деструктор!" << endl;
}

};

```

class_list.h:

```

#pragma once
#include "basic_class.h"

class Arr
{
private:
    Student* array_stud;

    int count = 1;

public:
    void create_array();

    void print_array(int) const;

    void delete_one();

    void add();

    string select() const;

    void fill_array();

    int count_plus();

    int count_minus();

    void print_one(Student) const;

    void delete_array();

```

```

    Student Construct(int);

    int prog_d_rand(bool);

    void find_debt();

    void in_f();

    void from_f(int);

    int get_count() const;

    int str_in_file(string) const;

    void set_count(int);

    void regex_task();

    void sort(comp);

    static bool sort_forward(const int&, const int&);

    static bool sort_back(const int&, const int&);

};

```

faculty.h

```

#pragma once

#include <string>
using std::string;

class Fac
{
private:
    string fname;
public:
    string get_fname() const;
    void set_fname(string);

    Fac();
    Fac(string);
    Fac(const Fac& other);
    ~Fac();
};

```

menu.h:

```

#pragma once
#include "class_list.h"

void menu(Arr);

```

surname_list.h

```

#pragma once
#include "surnames.h"

class Arr_Sur
{

```

```

private:
    Surname* array_sur;

    int count = 1;

public:
    void create_array();

    void print_array(int) const;

    void delete_one();

    void add();

    string select() const;

    void fill_array();

    int count_plus();

    int count_minus();

    void print_one(Surname) const;

    void delete_array();

    Surname Construct(int);

    int prog_d_rand(bool);

    void find_debt();

    void in_f();

    void from_f(int);

    int get_count() const;

    int str_in_file(string) const;

    void set_count(int);

    void regex_task();

    void sort(comp);

    static bool sort_forward(const int&, const int&);

    static bool sort_back(const int&, const int&);

};

```

surname.h

```

#pragma once
#include "basic_class.h"

class Surname : public Student
{
private:
    string sur_star;
    string sur_cur;

```

```

public:
    string get_sur_star() const;
    string get_sur_cur() const;

    void set_sur_star(string);
    void set_sur_cur(string);

    Surname();
    Surname(int, int, int, string, bool, int, sint, string, string, string);
    Surname(const Surname& other);
    ~Surname();
};

```

test.h:

```

#pragma once
#include "class_list.h"

bool test_count_plus(Arr);
bool test_count_minus(Arr);

```

audience.cpp

```

#include "audience.h"

sint Aud::get_aud_numb() const
{
    return aud_numb;
}

void Aud::set_aud_numb(sint a_n)
{
    aud_numb = a_n;
}

Aud::Aud() : aud_numb(0)
{
}

Aud::Aud(sint aud_numb) : aud_numb(aud_numb)
{
}

Aud::Aud(const Aud& other) : aud_numb(other.aud_numb)
{
}

Aud::~Aud()
{
}

```

basic_class.cpp:

```

#include "basic_class.h"

Student::Student() : age(0), number_stud(0), middle_mark(0), name("Name"), debt(0),
prog_d(0), audience(0), faculty("Non")
{
    cout << "Вызван стандартный конструктор!" << endl;
}

```

```

Student::Student(int a, int n, int m, string na, bool d, int pd, sint an, string fc) :
age(a), number_stud(n), middle_mark(m), name(na), debt(d), prog_d(pd), audience(an),
faculty(fc)
{
    cout << "Вызван конструктор с параметрами!" << endl;
}

Student::Student(const Student &other) : age(other.age), number_stud(other.number_stud),
middle_mark(other.middle_mark), name(other.name), debt(other.debt), prog_d(other.prog_d),
audience(other.audience), faculty(other.faculty)
{
    cout << "Вызван конструктор копирования!" << endl;
}

int Student::get_number_stud() const
{
    return number_stud;
}

int Student::get_age() const
{
    return age;
}

int Student::get_middle_mark() const
{
    return middle_mark;
}

string Student::get_name() const
{
    return name;
}

bool Student::get_debt() const
{
    return debt;
}

int Student::get_prog_d() const
{
    return prog_d;
}

sint Student::get_aud()
{
    return audience.get_aud_numb();
}

string Student::get_fac()
{
    return faculty.get_fname();
}

```

surname_list.cpp:

```

#include "surname_list.h"

```

```

void Arr_Sur::create_array()

```

```

{
    int c = get_count();

    array_sur = new Surname[c];
}

void Arr_Sur::fill_array()
{
    int c = get_count();

    for (size_t i = 0; i < c; i++)
    {
        array_sur[i] = Construct(i);
    }
}

Surname Arr_Sur::Construct(int i)
{
    bool d = 0;

    if (i == 0)
    {
        d = rand() % 2;
        Surname st0(i + 17, i + 1, i + 10 / 2, "Dani", d, prog_d_rand(d), 15,
"KIT", "Pavlov", "Jordon");
        return st0;
    }
    else if (i == 1)
    {
        d = rand() % 2;
        Surname st1(i + 17, i + 1, i + 10 / 2, "Peter", d, prog_d_rand(d), 15,
"KIT", "Pavlov", "Jordon");
        return st1;
    }
    else if (i == 2)
    {
        d = rand() % 2;
        Surname st2(i + 17, i + 1, i + 10 / 2, "Donald Tramp", d, prog_d_rand(d),
36, "PIT", "Hashin", "Ivanov");
        return st2;
    }
    else if (i == 3)
    {
        d = rand() % 2;
        Surname st2(i + 17, i + 1, i + 10 / 2, "Vladimir", d, prog_d_rand(d), 87,
"PIT", "Vityaz", "Ivanov");
        return st2;
    }

    d = rand() % 2;
    Surname st(i + 17, i + 1, i + 10 / 2, "Edgar", d, prog_d_rand(d), 17, "PIT",
"Dropov", "Ivanov");
    return st;
}

void Arr_Sur::print_array(int c) const
{
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
    ////////////
    for (size_t i = 0; i < c; i++)
    {

```



```

        cout << "Номер: " << array_sur[i].get_number_stud() << "\tВозраст: " <<
array_sur[i].get_age() << "\tСредний балл: " << array_sur[i].get_middle_mark() << "\tИмя:
" << array_sur[i].get_name() << "\tАудитория: " << array_sur[i].get_aud() <<
"\tФакультет: " << array_sur[i].get_fac() << "\tФамилия старосты: " <<
array_sur[i].get_sur_star() << "\tФамилия куратора: " << array_sur[i].get_sur_cur() <<
endl;
    }
    ////////////
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
}

void Arr_Sur::delete_one()
{
    int input, i_arr = 0;
    cout << "Введите номер: ";
    cin >> input;

    for (size_t i = 0; i < E1; i++)
        if (input == array_sur[i].get_number_stud())
        {
            count_minus();
            int c = get_count();

            Surname* array_sur_new = new Surname[c];

            for (size_t i = 0; i < c + 1; i++)
            {
                if (input == array_sur[i].get_number_stud())
                    continue;
                array_sur_new[i_arr++] = array_sur[i];
            }

            delete[] array_sur;

            array_sur = new Surname[c];

            for (size_t i = 0; i < c; i++)
            {
                array_sur[i] = array_sur_new[i];
            }

            delete[] array_sur_new;

            print_array(c);

            return;
        }
    ////////////
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
    ////////////
    cout << "Этого студента не существует!" << endl;
    ////////////
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
    ////////////
}

void Arr_Sur::add()
{
    count_plus();

```

```

    int c = get_count();

    Surname* array_sur_new = new Surname[c];

    size_t i = 0;

    for (i; i < c - 1; i++)
    {
        array_sur_new[i] = array_sur[i];
    }

    array_sur_new[c - 1] = Construct(i);

    delete[] array_sur;

    array_sur = new Surname[c];

    for (i = 0; i < c; i++)
    {
        array_sur[i] = array_sur_new[i];
    }

    delete[] array_sur_new;

    print_array(c);
}

int Arr_Sur::count_plus()
{
    count = count + 1;

    return count;
}

int Arr_Sur::count_minus()
{
    count = count - 1;

    return count;
}

string Arr_Sur::select() const
{
    int input = 0;
    cout << "Введите номер: ";
    cin >> input;
    int c = get_count();

    for (size_t i = 0; i < c; i++)
    {
        if (input == array_sur[i].get_number_stud())
        {
            string info;
            stringstream ss;
            ss << "Номер: " << array_sur[i].get_number_stud() << "\tВозраст: "
<< array_sur[i].get_age() << "\t\tСредний балл: " << array_sur[i].get_middle_mark() <<
"\t\tИмя: " << array_sur[i].get_name() << "\tАудитория: " << array_sur[i].get_aud() <<
"\tФакультет: " << array_sur[i].get_fac() << "\tФамилия старосты: " <<
array_sur[i].get_sur_star() << "\tФамилия куратора: " << array_sur[i].get_sur_cur() <<
endl;

            info = ss.str();

            return info;
        }
    }
}

```

```

//////////
for (size_t i = 0; i < 50; i++)
    cout << "-";
cout << endl;
//////////
cout << "Этого студента не существует!" << endl;
//////////
for (size_t i = 0; i < 50; i++)
    cout << "-";
cout << endl;
//////////
}

void Arr_Sur::print_one(Surname stud) const
{
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
    //////////
    cout << "Номер: " << stud.get_number_stud() << "\tВозраст: " << stud.get_age() <<
"\t\tСредний балл: " << stud.get_middle_mark() << "\t\tИмя: " << stud.get_name() <<
"\t\tАудитория: " << stud.get_aud() << "\t\tФакультет: " << stud.get_fac() << "\tФамилия
старосты: " << stud.get_sur_star() << "\tФамилия куратора: " << stud.get_sur_cur() <<
endl;
    //////////
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
}

void Arr_Sur::delete_array()
{
    delete[] array_sur;
}

int Arr_Sur::prog_d_rand(bool d)
{
    int pd = 0;

    if (d == 1)
        pd = rand() % 50;

    return pd;
}

void Arr_Sur::find_debt()
{
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
    //////////
    for (size_t i = 0; i < count; i++)
        if (array_sur[i].get_debt() == 1)
        {
            cout << "Номер: " << array_sur[i].get_number_stud() << "\tВозраст: "
<< array_sur[i].get_age() << "\t\tСредний балл: " << array_sur[i].get_middle_mark() <<
"\t\tИмя: " << array_sur[i].get_name() << "\t\tДолг: Есть" << "\t\tДолг по
программированию: " << array_sur[i].get_prog_d() << "%" << endl;
        }
    //////////
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
}

```

```

void Arr_Sur::in_f()
{
    ofstream file;
    file.open("of2.txt", ofstream::app);

    if (!file.is_open())
    {
        cout << "Файл не открыт!" << endl;
        return;
    }

    string info;
    stringstream ss;
    int c = get_count();

    for (size_t i = 0; i < c; i++)
    {
        ss << "Номер: " << array_sur[i].get_number_stud() << "\tВозраст: " <<
array_sur[i].get_age() << "\tСредний балл: " << array_sur[i].get_middle_mark() << "\tИмя:
" << array_sur[i].get_name() << "\tАудитория: " << array_sur[i].get_aud() <<
"\tФакультет: " << array_sur[i].get_fac() << "\tФамилия старосты: " <<
array_sur[i].get_sur_star() << "\tФамилия куратора: " << array_sur[i].get_sur_cur() <<
endl;
    }

    for (size_t i = 0; i < 80; i++)
        ss << "-";
    ss << endl;

    info = ss.str();
    file << info;

    file.close();

    ////////////
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
    ////////////
    cout << "Файл записан успешно!" << endl;
    ////////////
    for (size_t i = 0; i < 50; i++)
        cout << "-";
    cout << endl;
    ////////////
}

void Arr_Sur::from_f(int size)
{
    string s;
    regex varEn("([\\d]* [\\d]* [\\d]* [A-Z]+[\\w,.;:-]* [0|1] [\\d]* [\\d]* [A-Z]*
[A-Z]+[\\w,.;:-]* [A-Z]+[\\w,.;:-]*)");

    ifstream file;
    file.open("if2.txt");

    if (!file.is_open())
    {
        cout << "Файл не открыт!" << endl;
        return;
    }

    delete[] array_sur;

```

```

array_sur = new Surname[size];

for (size_t i = 0; i < size; i++)
{
    getline(file, s);
    if (regex_match(s, varEn))
    {
        std::istringstream iss(s);
        int age;
        int number;
        int middle_mark;
        string name;
        bool debt;
        int prog_d;
        sint audience;
        string faculty;
        string sur_star;
        string sur_cur;

        iss >> age;
        iss >> number;
        iss >> middle_mark;
        iss >> name;
        iss >> debt;
        iss >> prog_d;
        iss >> audience;
        iss >> faculty;
        iss >> sur_star;
        iss >> sur_cur;

        Surname file_el(age, number, middle_mark, name, debt, prog_d,
        audience, faculty, sur_star, sur_cur);
        array_sur[i] = file_el;
    }
}

file.close();

cout << "Чтение с файла успешно!" << endl << endl;

set_count(size);

print_array(size);
}

int Arr_Sur::get_count() const
{
    return count;
}

int Arr_Sur::str_in_file(string fileName) const
{
    int size = 0;
    int c = get_count();
    string line;
    ifstream file(fileName);

    if (!file.is_open() || c == 0)
    {
        cout << "There is no such file" << endl << endl;
        return 0;
    }
}

```

```

        while (getline(file, line))
        {
            size++;
        }
        file.close();

        return size;
    }

    void Arr_Sur::set_count(int c)
    {
        count = c;
    }

    void Arr_Sur::regex_task()
    {
        for (size_t i = 0; i < 50; i++)
            cout << "-";
        cout << endl;
        ////////////////
        regex regular("(^[A-ZÀ-ß]+[\\wÀ-ßà-ÿ,.;:-]* [\\wÀ-ßà-ÿ,.;:-]+)");
        int listSize = get_count();

        for (size_t i = 0; i < listSize; i++)
        {
            if (regex_match(array_sur[i].get_name(), regular))
                print_one(array_sur[i]);

        }

        cout << endl;
        ////////////////
        for (size_t i = 0; i < 50; i++)
            cout << "-";
        cout << endl;
    }

    bool Arr_Sur::sort_forward(const int& a, const int& b)
    {
        return a > b;
    }

    bool Arr_Sur::sort_back(const int& a, const int& b)
    {
        return a < b;
    }

    void Arr_Sur::sort(comp condition)
    {
        Surname temp;
        int size = get_count();
        bool pr;

        do {
            pr = 0;
            for (size_t i = 0; i < size - 1; i++)
            {
                if (condition(array_sur[i].get_number_stud(), array_sur[i +
1].get_number_stud()))
                {
                    temp = array_sur[i];
                    array_sur[i] = array_sur[i + 1];
                    array_sur[i + 1] = temp;
                    pr = 1;
                }
            }
        } while (pr);
    }

```

```

        }
    } while (pr == 1);

    print_array(size);
}

```

main.cpp:

```

/**
 * @mainpage
 * <b>Лабораторна робота № 6. <br/> СПАДКУВАННЯ </b>
 * <br/><b><i>Мета роботи:</i></b> отримати знання про базові регулярні вирази та
досвід роботи із застосування їх на практиці. <br/>
 * <b><i>Індивідуальне завдання 19:</i></b>
 * Поширити попередню лабораторну роботу. <br/>
 * <br/> <b> Arr p; class Student; class Arr; </b> Класи, методи, функції, конструктори,
потоки </b>
 * @author Tatarenko A.
 * @date 29-may-2020
 * @version 1.0
 */

#include "menu.h"

int main()
{
    setlocale(LC_ALL, "ru");

    Arr_Sur stud_array;

    auto input = 0;

    cout << "Введите 1 чтобы создать массив, 2 чтобы выйти: ";
    cin >> input;

    if (input == 1)
    {
        stud_array.create_array();
        cout << "Список созданный с помощью конструктора:" << endl;
        stud_array.print_array(stud_array.get_count());

        cout << "Рандомное заполнение списка:" << endl;
        stud_array.fill_array();
        stud_array.print_array(stud_array.get_count());

        menu(stud_array);

        stud_array.delete_array();
    }

    //////////////////////////////////////
    int l = _CrtDumpMemoryLeaks(); // Контроль витоку пам'яті
    if (l == 0)
        cout << "Утечки памяти не обнаружено!" << endl;
    else
        cout << "Обнаружена утечка памяти!" << endl;

    system("PAUSE");
    return 0;
}

```

faculty.cpp

```
#include "faculty.h"

string Fac::get_fname() const
{
    return fname;
}

void Fac::set_fname(string fn)
{
    fname = fn;
}

Fac::Fac() : fname("Non")
{
}

Fac::Fac(string fn) : fname(fn)
{
}

Fac::Fac(const Fac& other) : fname(other.fname)
{
}

Fac::~Fac()
{
}
```

surnames.cpp

```
#include "surnames.h"

string Surname::get_sur_star() const
{
    return sur_star;
}

string Surname::get_sur_cur() const
{
    return sur_cur;
}

void Surname::set_sur_star(string s_s)
{
    sur_star = s_s;
}

void Surname::set_sur_cur(string s_c)
{
    sur_cur = s_c;
}

Surname::Surname() : Student(0, 0, 0, "Name", 0, 0, 0, "Non"), sur_star("Surname"),
sur_cur("Surname")
{
}

Surname::Surname(int a, int n, int m, string na, bool d, int pd, sint an, string fc,
string s_s, string s_c) : Student(a, n, m, na, d, pd, an, fc), sur_star(s_s),
sur_cur(s_c)
```



```

{
}

Surname::Surname(const Surname& other) : Student(other), sur_star(other.sur_star),
sur_cur(other.sur_cur)
{
}

Surname::~Surname()
{
}

```

menu.cpp:

```

#include "menu.h"

void menu(Arr stud_array)
{
    int input = 0;

    cout << "1) Добавить объект в конец;" << endl << "2) Удалить объект;" << endl <<
    "3) Выбрать объект;" << endl << "4) Вывести все объекты;" << endl << "5) Поиск
    должников;" << endl << "6) Запись в файл;" << endl << "7) Читать с файла;" << endl << "8)
    REGEX TASK;" << endl << "9) Сортировать;" << endl << "10) Выйти;" << endl << "Ваше
    действие: ";
    cin >> input;

    switch (input)
    {
    case 1:
        stud_array.add();
        break;
    case 2:
        stud_array.delete_one();
        break;
    case 3:
        cout << stud_array.select();
        break;
    case 4:
        stud_array.print_array(stud_array.get_count());
        break;
    case 5:
        stud_array.find_debt();
        break;
    case 6:
        stud_array.in_f();
        break;
    case 7:
        stud_array.from_f(stud_array.str_in_file("if.txt"));
        break;
    case 8:
        stud_array.regex_task();
        break;
    case 9:
        cout << "Выберете тип сортировки:" << endl << "Сортировка по возрастанию;"
        << endl << "Сортировка по убыванию;" << endl << endl;
        int c;
        cin >> c;

        if (c == 1)
            stud_array.sort(stud_array.sort_forward);
    }
}

```

```

        else if (c == 2)
            stud_array.sort(stud_array.sort_back);
        else
        {
            cout << "НЕВЕРНОЕ ДЕЙСТВИЕ!" << endl;
            ///////////////////////////////////
            for (size_t i = 0; i < 50; i++)
                cout << "-";
            cout << endl;
        }
        break;
    case 10:
        return;
    default:
        break;
}

menu(stud_array);
}

```

test.cpp:

```

#include "test.h"

int main()
{
    Arr stud_array;

    if (test_count_plus(stud_array))
        cout << "Function count_plus is right!" << endl;
    else
        cout << "Function count_plus is not right!" << endl;

    if (test_count_minus(stud_array))
        cout << "Function count_minus is right!" << endl;
    else
        cout << "Function count_minus is not right!" << endl;

    system("PAUSE");
}

bool test_count_plus(Arr p)
{
    int exp = E1 + 1;
    int cnt = p.count_plus();

    if (cnt == exp)
        return 1;
    else
        return 0;
}

bool test_count_minus(Arr p)
{
    int exp = E1 - 1;
    int cnt = p.count_minus();

    if (cnt == exp)
        return 1;
    else
        return 0;
}

```

4 Результати тестування

```
Введите 1 чтобы создать массив, 2 чтобы выйти: 1
Вызван конструктор с параметрами!
Вызван конструктор с параметрами!
Вызван конструктор с параметрами!
Список созданный с помощью конструктора:
-----
Номер: 0      Возраст: 0      Средний балл: 0 Имя: Name      Аудитория: 0      Факультет: Non      Фамилия старосты: Surname      Фамилия куратора: Surname
Номер: 0      Возраст: 0      Средний балл: 0 Имя: Name      Аудитория: 0      Факультет: Non      Фамилия старосты: Surname      Фамилия куратора: Surname
Номер: 0      Возраст: 0      Средний балл: 0 Имя: Name      Аудитория: 0      Факультет: Non      Фамилия старосты: Surname      Фамилия куратора: Surname
-----
Рандомное заполнение списка:
Вызван конструктор с параметрами!
Вызван конструктор копирования!
Вызвался деструктор!
Вызвался деструктор!
Вызван конструктор с параметрами!
Вызван конструктор копирования!
Вызвался деструктор!
Вызвался деструктор!
Вызван конструктор с параметрами!
Вызван конструктор копирования!
Вызвался деструктор!
Вызвался деструктор!
-----
Номер: 1      Возраст: 17      Средний балл: 5 Имя: Dani      Аудитория: 15      Факультет: KIT      Фамилия старосты: Pavlov      Фамилия куратора: Jordon
Номер: 2      Возраст: 18      Средний балл: 6 Имя: Peter      Аудитория: 15      Факультет: KIT      Фамилия старосты: Pavlov      Фамилия куратора: Jordon
Номер: 3      Возраст: 19      Средний балл: 7 Имя: Donald Tramp      Аудитория: 36      Факультет: PIT      Фамилия старосты: Hashin      Фамилия куратора: Ivanov
-----
1) Добавить объект в конец;
2) Удалить объект;
3) Выбрать объект;
4) Вывести все объекты;
5) Поиск должников;
6) Запись в файл;
7) Читать с файла;
8) REGEX TASK;
9) Сортировать;
10) Выйти;
Ваше действие:
```

5 Опис результатів

Програма виводить список студентів. Виконує видалення, додавання, вибір об'єкта, виводить інформацію про заборгованість. А також виконує запис і читання з файлу, та перевіряє читання через базові регулярні вирази.

Висновок:

Протягом цієї роботи було отримано базові знання про парадигму ООП – спадкування.