
Основи програмування – 2

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені
Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 5 з дисципліни
«Основи програмування – 2.
Методології програмування»

«Наслідування та поліморфізм»

Варіант 29

Виконав студент ПІ-13 Романюк Діана Олексіївна
(шифр, прізвище, ім'я, по батькові)

Перевірів Вечерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Лабораторна робота 5

Мета – вивчити механізми створення класів та об'єктів.

Варіант 29

Постановка задачі:

29. Спроекувати клас “Особа”, який містить ПІБ, дата народження і методи визначення її віку та обчислення місячного заробітку. На основі цього класу створити класи-нащадки “Студент”, який містить додатково номер академічної групи студента, середній рейтинговий бал за результатами останньої сесії, статус отримання стипендії у новому семестрі (підвищена, звичайна, немає), та “Викладач”, який містить додатково назви дисциплін, які викладає даний викладач, планову кількість годин, які він повинен провести по кожній дисципліні, за місяць. Створити n студентів і m викладачів. Для студентів розрахувати розмір місячної стипендії (підвищена, якщо середній рейтинговий бал більше 95, і звичайна – якщо середній рейтинговий бал більше 85, але менше 95), для викладачів – місячну заробітну плату. Визначити вік викладача, що має найбільшу місячну заробітну плату.

Код програми на C++:

Header.h

```
#include <iostream>
#include <string>
#include <vector>
#include <Windows.h>

using namespace std;

class Person {
private:
    string name;
    string date_of_birth;
public:
    Person(string line);
    string getName();
    string count_age();
    virtual double monthly_income(double);
};

class Student : public Person {
private:
    int group_id;
    double average_grade;
    char status;
public:
    Student(string line1 = "none none none 01:01:1970", string line2 = "00 0.0");
    double monthly_income(double) override;
};

class Teacher : public Person {
private:
    string subject;
    int hours;
public:
    int getHours();
    double monthly_income(double) override;
};
```

Основи програмування – 2

```
Teacher(string line1 = "none none none 01:01:1970", string line2 = "none 0");  
};
```

```
std::vector<string> split(string, char sep = '');
```

Lab5.cpp

```
#include "Header.h"  
  
int main() {  
    vector<Student> base_of_students;  
    vector<Teacher> base_of_teachers;  
    int n, m;  
    string str1, str2;  
    cout << "Enter amount of students: "; cin >> n;  
    cin.ignore();  
    for (int i = 0; i < n; i++) {  
        cout << "[Surname Name Patronymic DD::MM:YYYY]\n";  
        getline(cin, str1);  
        cout << "[id grades]\n";  
        getline(cin, str2);  
        base_of_students.push_back(Student(str1, str2));  
    }  
    cout << "Enter amount of teachers: "; cin >> m;  
    cin.ignore();  
    for (int i = 0; i < m; i++) {  
        cout << "[Surname Name Patronymic DD::MM:YYYY]\n";  
        getline(cin, str1);  
        cout << "[subject hours]\n";  
        getline(cin, str2);  
        base_of_teachers.push_back(Teacher(str1, str2));  
    }  
    double grants; cout << "Enter the grants amount: ";  
    cin >> grants;  
    cout << "Students:" << endl;  
    for (auto& s : base_of_students) {  
        cout << s.getName() << ": " << s.monthly_income(grants) << " UAH" << endl;  
    }  
    double salary; cout << "Enter the teachers' salary per hour: ";  
    cin >> salary;  
    cout << "Teachers:" << endl;  
    Teacher richest;  
    if (base_of_teachers.size() > 0) {  
        richest = base_of_teachers[0];  
    }  
    for (int i = 0; i < m; i++) {  
        cout << base_of_teachers[i].getName() << ": " <<  
base_of_teachers[i].monthly_income(salary) << " UAH" << endl;  
        if (base_of_teachers[i].monthly_income(salary) > richest.monthly_income(salary)) richest =  
base_of_teachers[i];  
    }  
    cout << "The teacher with the biggest salary : " << richest.getName() << "\nHis age: " <<  
richest.count_age();  
    return 0;  
}
```

Header.cpp

```
#include "Header.h"  
  
Person::Person(string line) {  
    name = split(line)[0] + ' ' + split(line)[1] + ' ' + split(line)[2];  
    date_of_birth = split(line)[3];  
}
```

Основи програмування – 2

```
string Person::getName() {
    return name;
}

double Person::monthly_income(double hours) {
    return (150 * hours);
}

string Person::count_age() {
    SYSTEMTIME current_date;
    GetLocalTime(&current_date);
    int birth_year = stoi(split(date_of_birth, ':')[2]);
    int birth_month = stoi(split(date_of_birth, ':')[1]);
    int birth_day = stoi(split(date_of_birth, ':')[0]);
    int d_day, d_month, d_year;
    d_year = current_date.wYear - birth_year;
    if (current_date.wMonth < birth_month) { d_month = current_date.wMonth + 12 - birth_month; }
    else { d_month = current_date.wMonth - birth_month; }
    if (current_date.wDay < birth_day) { d_day = current_date.wDay + 30 - birth_day; }
    else { d_day = current_date.wDay - birth_day; }
    if (current_date.wMonth < birth_month || current_date.wMonth == birth_month &&
        current_date.wDay < birth_day) {
        d_year--;
    }
    string age = "";
    age += to_string(d_year) + "year(s)" + to_string(d_month) + "month(s)" + to_string(d_day) +
"day(s)\n";
    return age;
}

Student::Student(string line1, string line2) : Person(line1) {
    group_id = stoi(split(line2)[0]);
    average_grade = stod(split(line2)[1]);
}

double Student::monthly_income(double grants)
{
    if (average_grade >= 95) {
        status = 'h';
        return grants * 1.4;
    }
    if (average_grade >= 85) {
        status = 'd';
        return grants;
    }
    status = 'n';
    return 0;
}

Teacher::Teacher(string line1, string line2) : Person(line1) {
    subject = split(line2)[0];
    hours = stoi(split(line2)[1]);
}

int Teacher::getHours() {
    return hours;
}

double Teacher::monthly_income(double payment)
{
    return (payment * this->getHours());
}

vector<string> split(string line, char sep) {
    vector<string> words;
    string temp_word = "";
    line += sep;
```

Основи програмування – 2

```
for (int i = 0; i < line.length(); i++) {
    if (line[i] == sep) {
        if (temp_word.length() > 0) {
            words.push_back(temp_word);
        }
        temp_word = "";
    }
    else {
        temp_word += line[i];
    }
}
return words;
}
```



Select Консоль отладки Microsoft Visual Studio

```
[Surname Name Patronymic DD::MM:YYYY]
Smone3 Smone3 Smone3 29:05:2000
[id grades]
15 74
Enter amount of teachers: >> 2
[Surname Name Patronymic DD::MM:YYYY]
T1 T1 T1 11:09:1986
[subject hours]
math 12
[Surname Name Patronymic DD::MM:YYYY]
T2 T2 T2 30:09:2001
[subject hours]
biology 8
Enter the grants amount: 2200
Students:
Smone Smone Sone: 2200 UAH
Smone2 Smone2 Smone2: 2200 UAH
Smone3 Smone3 Smone3: 0 UAH
Enter the teachers' salary per hour: 100
Teachers:
T1 T1 T1: 1200 UAH
T2 T2 T2: 800 UAH
The teacher with the biggest salary : T1 T1 T1
His age: 35year(s)9month(s)12day(s)
C:\Users\Asus\op_lab\second_sem\Lab5\C++\x64\Debug\Lab5.exe (процесс 9844) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

Код програми на Python

Lab5.py

```
from foo import *

base_of_students = list()
base_of_teachers = list()
n = int(input('Enter amount of students: '))
for i in range(n):
    line1 = input('[Surname Name Patronymic DD::MM:YYYY]\n')
    line2 = input('[id grades]\n')
    base_of_students.append(Student(line1, line2))
m = int(input('Enter amount of teachers: '))
for i in range(m):
    line1 = input('[Surname Name Patronymic DD::MM:YYYY]\n')
    line2 = input('[subject hours]\n')
    base_of_teachers.append(Teacher(line1, line2))
grants = float(input('Enter the grants amount: '))
print('STUDENTS:')
for i in range(n):
    print(f'student {base_of_students[i].get_name()} {base_of_students[i].monthly_income(grants)} UAH')
if m > 0:
    payment = float(input('Enter the payment per hour: '))
    richest = base_of_teachers[0]
```

Основи програмування – 2

```
for i in range(m):
    print(f'teacher {base_of_teachers[i].get_name()}
{base_of_teachers[i].monthly_income(payment)} UAH')
    if base_of_teachers[i].monthly_income(payment) > richest.monthly_income(payment):
        richest = base_of_teachers[i]
print(f'Richest teacher: {richest.get_name()}')
print(f'His age: {richest.get_age()}')
```

foo.py

```
from datetime import date
from abc import ABC, abstractmethod

class Person(ABC):
    def __init__(self, line: str):
        self.__name__ = line.split()[0] + ' ' + line.split()[1] + ' ' + line.split()[2]
        self.__date_of_birth = line.split()[3]

    def get_name(self):
        return self.__name__

    def get_age(self) -> str:
        today = date.today()
        b_day = int(self.__date_of_birth.split(':')[0])
        b_month = int(self.__date_of_birth.split(':')[1])
        b_year = int(self.__date_of_birth.split(':')[2])
        d_year = today.year - b_year
        if today.month < b_month:
            d_month = today.month + 12 - b_month
        else:
            d_month = today.month - b_month
        if today.day < b_day:
            d_day = today.day + 30 - b_day
        else:
            d_day = today.day - b_day
        if today.month < b_month or today.month == b_month and today.day < b_day:
            d_year -= 1
        age = f'{str(d_year)} year(s) {str(d_month)} month(s) {str(d_day)} day(s)'
        return age

    @abstractmethod
    def monthly_income(self, hours: float) -> float:
        return 150 * hours

class Student(Person):
    def __init__(self, line1, line2):
        super().__init__(line1)
        self.group_id = int(line2.split()[0])
        self.__average_grade = float(line2.split()[1])
        self.__status = 'n'

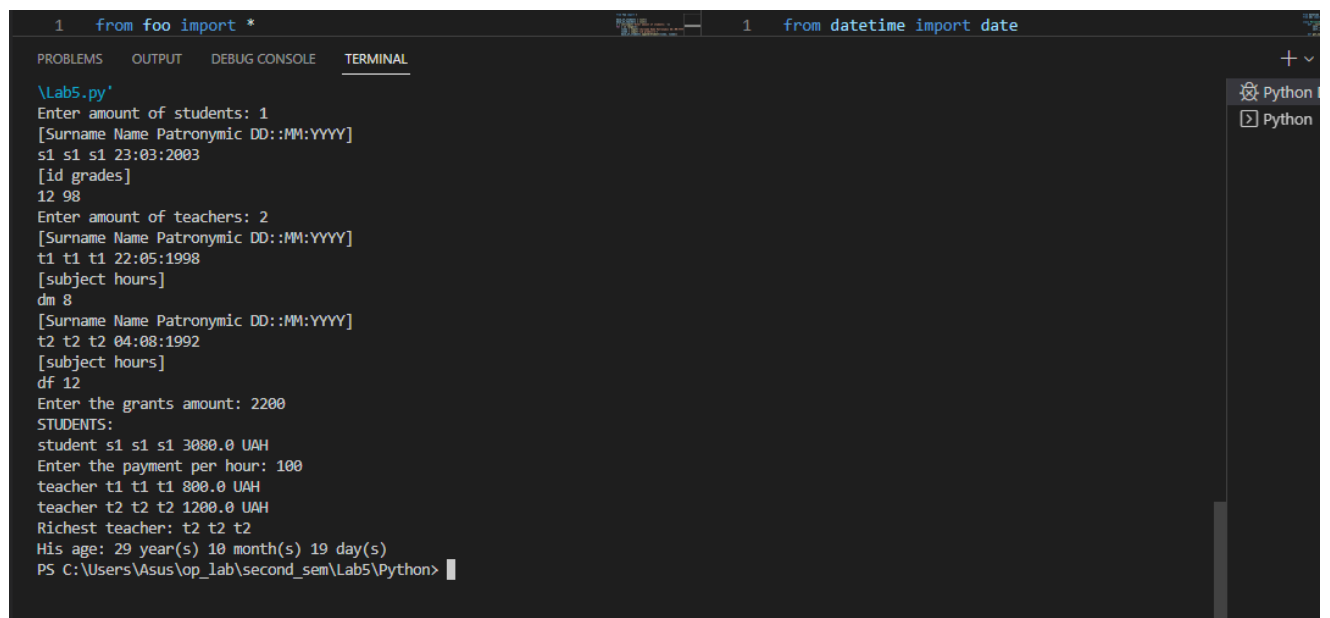
    def monthly_income(self, grants: float) -> float:
        if self.__average_grade >= 95:
            self.__status = 'h'
```

Основи програмування – 2

```
        return grants*1.4
    elif self.__average_grade >= 85:
        self.__status = 'd'
        return grants
    else:
        return 0

class Teacher(Person):
    def __init__(self, line1, line2):
        super().__init__(line1)
        self.__subject = line2.split()[0]
        self.hours = float(line2.split()[1])

    def monthly_income(self, salary: float) -> float:
        return salary*self.hours
```



```
1 from foo import *
1 from datetime import date

\Lab5.py
Enter amount of students: 1
[Surname Name Patronymic DD::MM:YYYY]
s1 s1 s1 23:03:2003
[id grades]
12 98
Enter amount of teachers: 2
[Surname Name Patronymic DD::MM:YYYY]
t1 t1 t1 22:05:1998
[subject hours]
dm 8
[Surname Name Patronymic DD::MM:YYYY]
t2 t2 t2 04:08:1992
[subject hours]
df 12
Enter the grants amount: 2200
STUDENTS:
student s1 s1 s1 3080.0 UAH
Enter the payment per hour: 100
teacher t1 t1 t1 800.0 UAH
teacher t2 t2 t2 1200.0 UAH
Richest teacher: t2 t2 t2
His age: 29 year(s) 10 month(s) 19 day(s)
PS C:\Users\Asus\op_lab\second_sem\Lab5\Python>
```

Висновки: під час лабораторної роботи ми вивчили механізми створення класів та об'єктів.