Міністерство освіти і науки України

Відокремлений структурний підрозділ «Тернопільський фаховий коледж Тернопільського національного технічного університету імені Івана Пулюя»

Циклова комісія комп’ютерних наук

ЗВІТ

про виконання лабораторної роботи №6

з дисципліни: “Об’єктно-орієнтоване програмування”

на тему: “Вивчення типів зв’язків між об’єктами”

Виконала:
студентка групи КН-321
Байдецька. В. П.

Прийняв:
Слободян Р.О

Тернопіль 2025

Мета: Навчитись реалізовувати різні типи зв’язків між об’єктами.

Кардинальність визначає кількісні відносини між класами. Композиція описує жорсткий зв'язок, де один об'єкт є невід'ємною частиною іншого. Наприклад, у автомобіля є лише один карбюратор, а карбюратор належить не більше ніж одному автомобілю, хоча карбюратори можуть існувати як окремі частини, відокремлені від конкретного автомобіля. Агрегація описує слабкий зв'язок між об'єктами. Наприклад, став може мати нуль або більше качок, а качка перебуває не більше ніж на одному ставку одночасно.

Хід роботи

1. Розробіть клас Student (в окремих файлах student.h і student.cpp) із

атрибутами:

– прізвище;

– ім’я;

– по батькові;

– номер залікової книжки

– державник/платник (тип bool).

Визначте для даного:

– класу конструктор по замовчуванню, який буде запитувати у користувача

дані для заповнення атрибутів об’єкта;

– параметризований конструктор;

– операцію виводу у потік std::cout.

У головній функції виконайте перевірку функціонування методів класу

створивши три об’єкти різними способами і вивівши їх на екран за допомогою

оператора виводу у потік (<<).

2. Розробіть клас Grupa, який міститиме як атрибут:

– назву групи (тип char * або std::string);

– спеціальність;

– список студентів групи. студенти описуються за допомогою класу

Student, який визначений у попередньому пункті завдання.

Визначте для даного класу всі можливі конструктори, деструктор, операції

виводу в потік.

Тип відношення між класами Grupa і Student – агрегація із кардинальністю

0..01 – 1..*.

3. Розробіть клас Facultet, який міститиме наступні атрибути:

– назву факультету (тип char * або std::string);

– список груп. Групи описуються за допомогою класу Grupa, який

визначений у попередньому пункті завдання.

Визначте для даного класу всі можливі конструктори, деструктор, операції

виводу в потік.

Тип відношення між класами Facultet і Grupa – композиція із кардинальністю

1 – 1..*.

Код:

student.h:

```
#include <iostream>
#include <string>
#include <vector>
using namespace std;

//student class
	class Student{
			string firstName;
			string lastName;
			string paternalName;
			bool scholarship;
		public:
			Student(); //+
			Student(string firstName, string lastName, string paternalName,
bool scholarship);
			//визначення виводу
			friend ostream& operator<<(ostream&, Student& student){
```

```
                  cout<<" Last Name: "<<student.lastName<<"\n First name:
"<<student.firstName<<"\n Paternal name"<<student.paternalName<<endl;
                  if (student.scholarship == true)
                        cout<<"студент вчиться на бюджеті ";
                  else
                        cout<<"студент вчиться на контракті";
                  return cout;
            };
      };

            Student::Student(){
                  int temp;
                  cout<<"Введіть прізвище: ";
                  cin>>lastName;
                  cout<<"Введіть ім'я: ";
                  cin>>firstName;
                  cout<<"Введіть побатькові: ";
                  cin>>paternalName;
                  cout<<"Чи навчається цей студент на бюджеті 0 - ні, 1 - так ";
                  cin>>temp;

                        switch(temp){
                              case 1:
                              {
                                    scholarship = true;
                                    break;
                              }
                              case 0:
                              {
                                    scholarship = false;
                                    break;
                              }
                              default:
                              {
                                    cout<<"неправильне значення. Автоматично
призначається на платне";
                                    scholarship = false;
                                    break;
                              }

                        }
            }

            Student::Student(string firstName, string lastName, string
paternalName, bool scholarship){
                  this->firstName = firstName;
                  this->lastName = lastName;
                  this->paternalName = paternalName;
                  this->scholarship = scholarship;
            }

//group class
      class Group{
                  string groupName;
                  int major;
                  vector<Student> students;
            public:
```

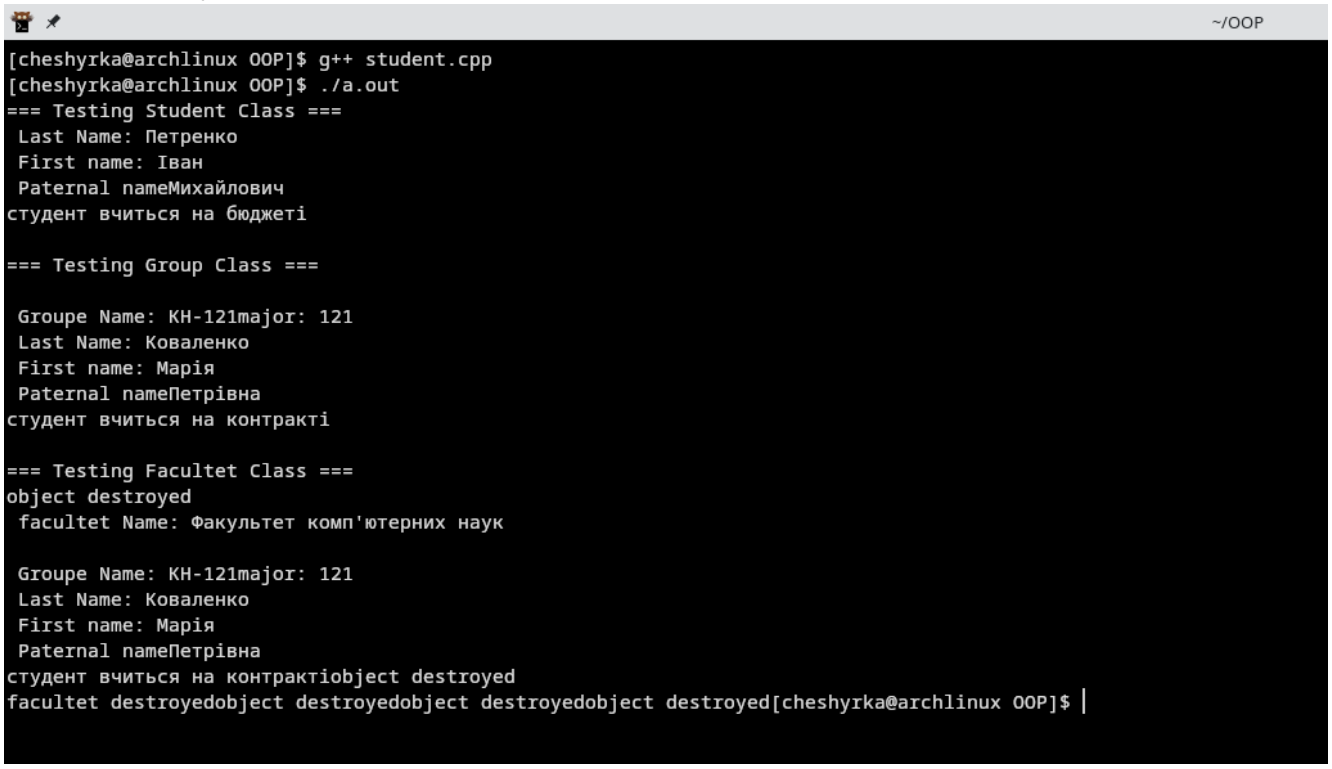
```
                // конструктор за замовчуванням
                Group();
                //параметризований конструктор
                Group(string groupName, int major, Student& student);
                //деструктор
                ~Group();
                //операція виводу в потік
                friend ostream& operator<<(ostream&, Group& group){
                cout<<"\n Groupe Name: "<<group.groupName<<"major:
"<<group.major<<endl;
                for (auto i: group.students)
                {
                      cout<<i;
                }
                return cout;
          }
         };

      Group::Group(){
            int temp;
            cout<<"Введіть назву групи: ";
            cin>>groupName;
            cout<<"Введіть номер спеціальності: ";
            cin>> major;
            cout<<"Введіть скільки студентів в групі: ";
            cin>>temp;
            int i = 0;
            do {
                  cout<<"Введіть студента ";
                  Student student;
                  students.push_back(student);
                  ++i;
            } while(i<temp);

      }

      Group::Group(string groupName, int major, Student& student){
            this->groupName = groupName;
            this->major = major;
            students.push_back(student);
      }

      Group::~Group(){
            cout<<"object destroyed";
      }

//facultet class
      class Facultet{
            string facultetName;
            vector<Group> groups;
            public:
                  Facultet();
                  Facultet(string facultetName,Group group);
                  friend ostream& operator<<(ostream&, Facultet& facultet){
                  cout<<"\n facultet Name: "<<facultet.facultetName<<endl;
                  for (auto i: facultet.groups)
                  {
```

```
                        cout<<i;
                  }
                  return cout;
            }
            ~Facultet();
      };

      Facultet::Facultet(){
            int temp;
            cout<<"Введіть назву факультету: ";
            cin>>facultetName;
            cout<<"Введіть скільки груп на факультеті в групі: ";
            cin>>temp;
            int i = 0;
            do {
                  cout<<"Введіть групу";
                  Group group;
                  groups.push_back(group);
                  ++i;
            } while(i<temp);
      }


      Facultet::Facultet(string facultetName,Group group){
            this->facultetName = facultetName;
            groups.push_back(group);

      }

      Facultet::~Facultet(){
            cout<<"facultet destroyed";
      }
```

student.cpp:

```
#include <iostream>
#include "student.h"
using namespace std;

int main(){
    cout<<"=== Testing Student Class ==="<<endl;

    Student student1("Іван", "Петренко", "Михайлович", true);
    cout<<student1<<endl;

    cout<<"\n=== Testing Group Class ==="<<endl;

    Student student2("Марія", "Коваленко", "Петрівна", false);
    Student student3("Олег", "Сидоренко", "Васильович", true);

    Group group1("КН-121", 121, student2);
    cout<<group1<<endl;

    Group group2("ПЗ-22", 122, student3);

    cout<<"\n=== Testing Facultet Class ==="<<endl;

    Facultet facultet1("Факультет комп'ютерних наук", group1);
```

```
    cout<<facultet1<<endl;


    return 0;
```




```
}
```

Висновок: в ході виконання лабораторної роботи я навчилась реалізовувати різні типи зв’язків між об’єктами.