

# Лабораторна робота №6

**Тема:** Композиція об'єктів в ООП

**Мета:** ознайомитись із способами та механізмами об'єктної композиції в ООП

ПОРЯДОК ВИКОНАННЯ РОБОТИ 1.

Ознайомитись з теоретичним матеріалом.

2. 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. 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.

3. Розробіть клас Grupa, який міститиме як атрибут назву групи (тип char \* або std::string), спеціальність і список студентів групи, студенти описуються за допомогою класу Student, який визначений у попередньому завданні. 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..\*.

4. Розробіть клас Facultet, який міститиме наступні атрибути: назву факультету (тип char \* або std::string) і список груп, групи описуються за допомогою класу Grupa, який визначений у попередньому завданні. 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 програми

```
//////////////////////////////////// Facultet.cpp////////////////////////////////////
#ifndef FACULTET_H
#define FACULTET_H
#include <iostream>
#include "student.h"
#include "grupa.h"

#include <string>

using namespace std;

class Facultet
{
private:
    string facultet;
public:
    Grupa *g;
    Grupa *g2;
    Grupa *g3;
```

```

    Facultet();
    Facultet(string nameFacultet);

    void buildGroups();

    friend ostream & operator <<(ostream & out, Facultet & fack);

};

#endif // FACULTET_H
//////////////////////////////////// Grupa.cpp////////////////////////////////////
#ifndef GRUPA_H
#define GRUPA_H
#include <iostream>
#include "student.h"

#include <string>

using namespace std;
class Grupa
{
    string Group;
    string specialty;
public:
    Student *stud1;
    Student *stud2;
    Student *stud3;
    Grupa();
    Grupa(string Group, string specialty, Student & student1, Student & student2, Student & student3);

    friend ostream & operator << (ostream & in, Grupa & grup);
    void print() ;
};
#endif // GRUPA_H

//////////////////////////////////// Student.cpp////////////////////////////////////
#ifndef STUDENT_H
#define STUDENT_H

#include <string>

using namespace std;

class Student
{
private:
    string lName;
    string fName;
    string sName;
    int numberZalikBook;
    bool state;
    bool payers;
public:
    Student();

```

```

    Student(string lName, string fName, string sName, int numberZalikBook, bool state, bool payers);
    friend ostream & operator<<(ostream & out, Student & obj);
};
#endif // STUDENT_H

```

```

//////////////////////////////////// Facultet.cpp////////////////////////////////////
#include "facultet.h"

```

```

Facultet::Facultet()
{

}

```

```

Facultet::Facultet(string nameFacultet)
{
    this->facultet = nameFacultet;
    buildGroups();
}

```

```

void Facultet::buildGroups()
{
    {

        cout << "EDIT GROUP " << endl ;
        string Group;
        cout << "Enter group =";
        cin >> Group;

        string Specialty;
        cout << "Enter specialty =";
        cin >> Specialty;
        cout << "\nENTER STUDENTS " << endl ;
        Student student1;

        Student student2;

        Student student3;
        Grupa _g(Group, Specialty, student1, student2, student3);
        g = &_g;

        cout << "\nEDIT GROUP " << endl ;
        cout << "Enter group =";
        cin >> Group;

        cout << "Enter specialty =";
        cin >> Specialty;
        cout << "\nENTER STUDENTS " << endl;
        Student student4;

        Student student5;

        Student student6;

        Grupa _g2(Group, Specialty, student4, student5, student6);
    }
}

```

```

g2 = &_g2;
cout << "\nEDIT GROUP " << endl ;

cout << "Enter group =";
cin >> Group;

cout << "Enter specialty =";
cin >> Specialty;
cout << "\nENTER STUDENTS " << endl ;

Student student7;

Student student8;

Student student9;

Grupa _g3(Group, Specialty, student7, student8, student9);
g3 = &_g3;

system("cls");
cout << "LIST OF STUDENTS OF THE "<< facultet<<" FACULTY" << endl ;
cout << endl<<" _____ "<<endl;
cout << *g;
cout << endl<<" _____ "<<endl;
cout << *g2;
cout << endl << " _____ " << endl;
cout << *g3;

}
}

ostream &operator <<(ostream &out, Facultet &fact)
{
    cout<<fact.g;
    cout << fact.g2;
    cout << fact.g3;
    return out;
}

//////////////////////////////////// Grupa.cpp////////////////////////////////////
#include "grupa.h"

Grupa::Grupa()
{

}

Grupa::Grupa(string Group, string specialty, Student &student1, Student &student2, Student &student3)
{
    this->Group = Group;
    this->specialty = specialty;
    this->stud1 = &student1;
    this->stud2 = &student2;
    this->stud3 = &student3;
}

```

```
}
```

```
void Grupa::print()
```

```
{
```

```
    cout << *stud1;
```

```
    cout << *stud2;
```

```
    cout << *stud3;
```

```
}
```

```
ostream &operator <<(ostream &in, Grupa &grup)
```

```
{
```

```
    cout << "GROUP=";
```

```
    in << grup.Group;
```

```
    cout << endl;
```

```
    cout << "SPECIALTY=";
```

```
    in << grup.specialty;
```

```
    cout << endl;
```

```
    grup.print();
```

```
    return in;
```

```
}
```

```
//////////////////////////////////// Student.cpp //////////////////////////////////////
```

```
#include "student.h"
```

```
#include <iostream>
```

```
using namespace std;
```

```
Student::Student()
```

```
{
```

```
    cout << "Enter lastName ";
```

```
    cin >> lName;
```

```
    cout << endl;
```

```
    cout << "Enter firstName ";
```

```
    cin >> fName;
```

```
    cout << endl;
```

```
}
```

```
Student::Student(string _lName, string _fName, string _sName, int _numberZalikBook, bool _state, bool  
_payers)
```

```
{
```

```
    this->lName = _lName;
```

```
    this->fName = _fName;
```

```
    this->sName = _sName;
```

```
    this->numberZalikBook = _numberZalikBook;
```

```
    this->state = _state;
```

```
    this->payers = _payers;
```

```
}
```

```
ostream &operator <<(ostream &out, Student &obj)
```

```
{
```

```
    cout << endl;
```

```
    out << obj.lName;
```

```
    cout << " ";
```

```
    out << obj.fName;
```

```
    cout << endl;;
```

```
}
```

```

//////////////////////////////////// main.cpp //////////////////////////////////////
#include <iostream>
#include <string>

#include "facultet.h"

using namespace std;

int main()
{
    string faculty;
    cout << "Enter faculty= " ;

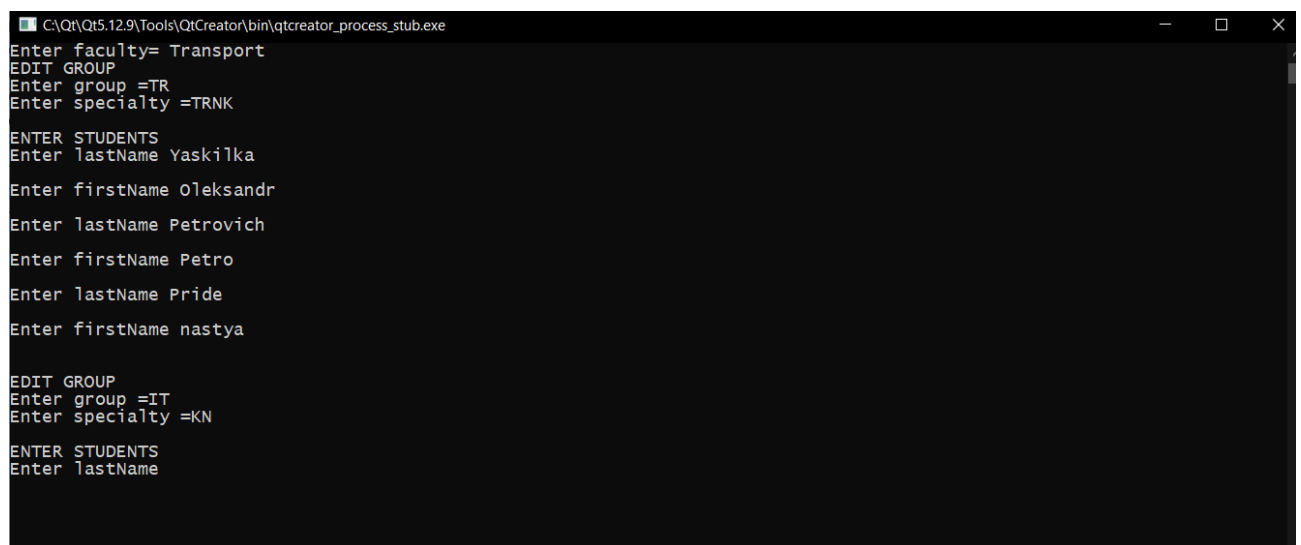
    cin >> faculty;
    Facultet myFacultet(faculty);
    cout << myFacultet;

    return 0;
}

```

## Результат

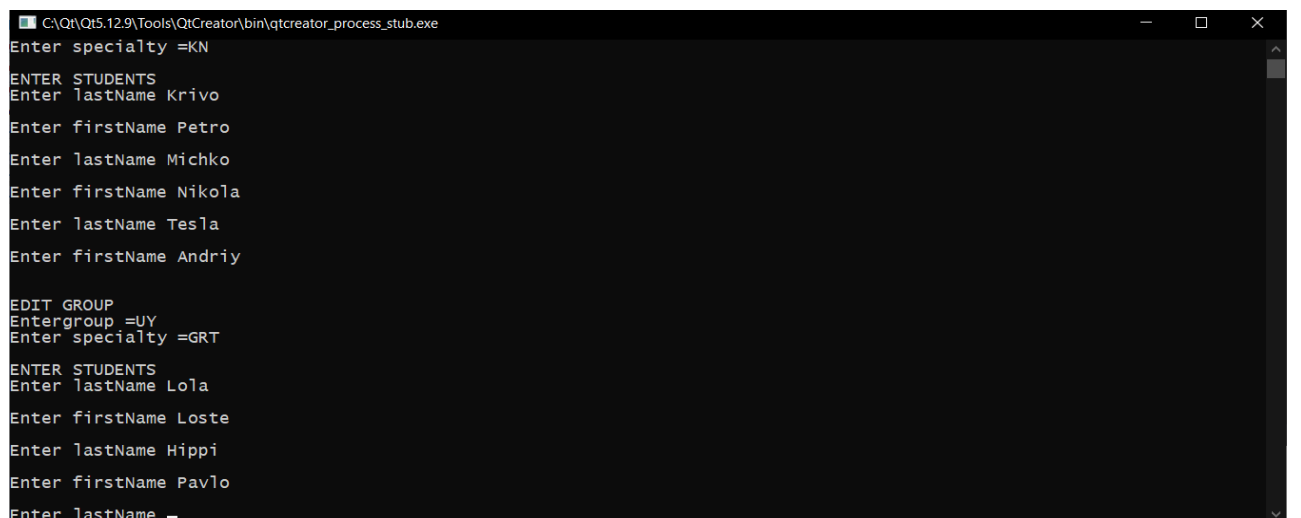
---



```

C:\Qt\Qt5.12.9\Tools\QtCreator\bin\qtcreator_process_stub.exe
Enter faculty= Transport
EDIT GROUP
Enter group =TR
Enter specialty =TRNK
ENTER STUDENTS
Enter lastName Yaskilka
Enter firstName Oleksandr
Enter lastName Petrovich
Enter firstName Petro
Enter lastName Pride
Enter firstName nastya
EDIT GROUP
Enter group =IT
Enter specialty =KN
ENTER STUDENTS
Enter lastName

```



```

Enter specialty =KN
ENTER STUDENTS
Enter lastName Krivo
Enter firstName Petro
Enter lastName Michko
Enter firstName Nikola
Enter lastName Tesla
Enter firstName Andriy
EDIT GROUP
Enter group =UY
Enter specialty =GRT
ENTER STUDENTS
Enter lastName Lola
Enter firstName Loste
Enter lastName Hippi
Enter firstName Pavlo
Enter lastName

```

```
C:\Qt\Qt5.12.9\Tools\QtCreator\bin\qtcreator_process_stub.exe

GROUP=TR
SPECIALTY=TRNK
Yaskilka Oleksandr
Petrovich Petro
Pride nastya

GROUP=IT
SPECIALTY=KN
Krivo Petro
Michko Nikola
Tesla Andriy

GROUP=UY
SPECIALTY=GRT
Lola Lose
Hippi Pavlo
Pank Zero
```

## Висновок:

Ознайомився із способами та механізмами об'єктної композиції в ООП