***ЛАБОРАТОРНА РОБОТА №6***

**Тема:** Композиція об’єктів в ООП  
**Мета:** Ознайомитись зі способами та механізмами

***ПОРЯДОК ВИКОНАННЯ РОБОТИ***

**Завдання 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. 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.

**Завдання 2.** 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.

**Завдання 3.** 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 і Student – композиція.

Код програми:

“student.h”

#include<string>

using namespace std;

#ifndef STUDENT\_H

#define STUDENT\_H

class Student

{

string familyName, name, fatherName;

int numberMarkbook;

bool studyType;

public:

Student();

Student(string familyName, string name, string fatherName, int numberMarkbook, bool studyType);

friend ostream& operator<<(ostream &, Student &);

};

#endif // STUDENT\_H

“student.cpp”

#include "student.h"

#include <iostream>

Student::Student()

{

cout << "Write student PIB: ";

cin >> familyName >> name >> fatherName;

cout << "Write markbook number: ";

cin >> numberMarkbook;

cout << "Choose study type (1-dergavne,2-platne): ";

int type;

cin >> type;

if(type == 1) studyType = true;

else studyType = false;

}

Student::Student(string familyName, string name, string fatherName, int numberMarkbook, bool studyType)

{

this->familyName = familyName;

this->name = name;

this->fatherName = fatherName;

this->numberMarkbook = numberMarkbook;

this->studyType = studyType;

}

ostream& operator<<(ostream &outSt, Student &student)

{

outSt << "Student: " << student.familyName << " " << student.name << " " << student.fatherName << endl;

outSt << "Markbook: " << student.numberMarkbook << endl;

outSt << "Studing type: ";

if(student.studyType) outSt << "statesman";

else outSt << "payer";

outSt << endl;

}

“grupa.h”

#include <string>

#include <student.h>

#ifndef GRUPA\_H

#define GRUPA\_H

using namespace std;

class Grupa

{

string name;

string specialty;

int studentsCount;

Student \*\*students;

void AddStudents(int count);

public:

Grupa();

Grupa(string name, string specialty, int studentsCount);

~Grupa();

friend ostream& operator<<(ostream&, Grupa&);

};

#endif // GRUPA\_H

“grupa.cpp”

#include "grupa.h"

#include <iostream>

using namespace std;

Grupa::Grupa()

{

cout << "Write group name: ";

cin >> name;

cout << "Write group specialty: ";

cin >> specialty;

cout << "Write studens count: ";

cin >> studentsCount;

AddStudents(studentsCount);

}

Grupa::Grupa(string name, string specialty, int studentsCount)

{

this->name = name;

this->specialty = specialty;

this->studentsCount = studentsCount;

AddStudents(studentsCount);

}

Grupa::~Grupa(){delete students;}

void Grupa::AddStudents(int count)

{

if(count < 1) studentsCount = 1;

students = new Student\*[studentsCount]();

for(int i = 0; i < studentsCount; i++)

students[i] = new Student();

}

ostream& operator<<(ostream& outSt, Grupa& group)

{

outSt << "Group: " << group.name << endl;

outSt << "Specialty: " << group.specialty << endl;

outSt << "Students in group(" << group.studentsCount << "): " << endl << endl;

for(int i = 0; i < group.studentsCount; i++)

outSt << \*group.students[i];

return outSt;

}

“facultet.h”

#include <string>

#include <grupa.h>

#ifndef FACULTET\_H

#define FACULTET\_H

using namespace std;

class Facultet

{

string name;

Grupa \*groups;

int groupsCount;

void AddGroups(int count);

public:

Facultet();

Facultet(string name, int groupsCount);

~Facultet();

friend ostream& operator<<(ostream& outSt, Facultet& facultet);

};

#endif // FACULTET\_H

“facultet.cpp”

#include "facultet.h"

#include <iostream>

Facultet::Facultet()

{

cout << "Write facultet name: ";

cin >> name;

cout << "Write groups count: ";

cin >> groupsCount;

AddGroups(groupsCount);

}

Facultet::Facultet(string name, int groupsCount)

{

this->name = name;

this->groupsCount = groupsCount;

AddGroups(groupsCount);

}

Facultet::~Facultet(){delete groups;}

void Facultet::AddGroups(int count)

{

if(count < 1) groupsCount = 1;

groups = new Grupa[groupsCount]();

}

ostream& operator<<(ostream& outSt, Facultet& facultet)

{

outSt << "Facultet: " << facultet.name << endl;

outSt << "Groups in facultet(" << facultet.groupsCount << "): " << endl << endl;

for(int i = 0; i < facultet.groupsCount; i++)

outSt << facultet.groups[i] << endl;

return outSt;

}

“main.cpp”

#include <QCoreApplication>

#include <iostream>

#include <student.h>

#include <grupa.h>

#include <facultet.h>

using namespace std;

int main(int argc, char \*argv[])

{

QCoreApplication a(argc, argv);

// ------------1------------

/\*Student st1;

Student st2("Qwert", "Asdfg", "Zxcv", 597394, true);

Student \*st3 = new Student();

cout << st1;

cout << st2;

cout << \*st3;\*/

//------------2-------------

/\*Grupa gr;

cout << gr;\*/

//-------------3------------

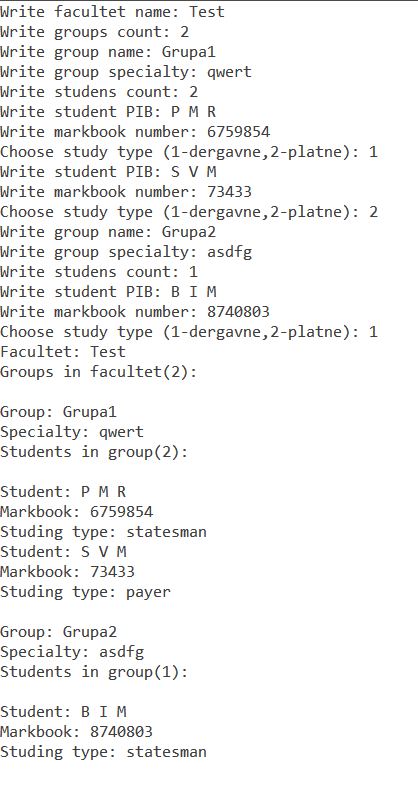
Facultet facultet;

cout << facultet;

return a.exec();

}

Вивід програми:



**Висновок:** Ознайомився зі способами та механізмами.