27.

```cpp
#include <iostream>
#include <string>
#include <vector>

class Student;
class CourseSection;

class Registration {
 public:
  Registration(Student* s, CourseSection* cs)
      : student{s}, course_seciton{cs} {}
  void set_grade(int grade) { this->grade = grade; }
  int get_grade() const { return this->grade; }
  Student* get_student() const { return this->student; }
  CourseSection* get_course_section() const { return this->course_seciton; }

 private:
  int grade{0};
  Student* student;
  CourseSection* course_seciton;
};

class Student {
 public:
  Student(std::string name) : name{name} {}
  void AddRegistration(Registration* r) {
    this->registration_list.push_back(r);
  }
  void PrintGradeOfCourse(CourseSection* cs) const {
    for (auto& r : this->registration_list) {
      if (r->get_course_section() == cs) {
        std::cout << r->get_grade() << "\n";
        break;
      }
    }
  }

 private:
  std::string name;
  std::vector<Registration*> registration_list{};
};

class CourseSection {
 public:
  CourseSection(std::string name) : name{name} {}
  /**
    在註冊階段用 Registration 連接 Student 與 CourseSection 的關係
    */
  void RequestToRegister(Student* student) {
    Registration* r = new Registration(student, this);
    student->AddRegistration(r);
    this->registration_list.push_back(r);
  }
  void SetGradeForStudent(Student* s, int grade) {
    for (auto& r : this->registration_list) {
      if (r->get_student() == s) {
```

```
        r->set_grade(grade);
        break;
      }
    }
  }
  ~CourseSection() {
    for (auto& r : this->registration_list) {
      delete r;
    }
  }

 private:
  std::string name;
  std::vector<Registration*> registration_list{};
};

int main() {
  CourseSection* oose = new CourseSection("OOSE");
  Student* alice = new Student("Alice");
  oose->RequestToRegister(alice);

  alice->PrintGradeOfCourse(oose); // 0
  oose->SetGradeForStudent(alice, 99);
  alice->PrintGradeOfCourse(oose); // 99

  delete alice;
  delete oose;
  return 0;
}
```