아래의 [요구사항]에 맞게 프로그램을 완성하시오. 단, [시작코드]는 변경하지 말 것. 작성한 프로그램은 시작코드를 포함한 하나의 <학번\_문제>.cpp 파일 형태로 업로드 하시오. 파일명에 한글이나 특수문자는 사용하지 마시오. [20점]

ex) 2021000111\_bonus\_q1.cpp 파일 업로드

[요구사항]

- Point class는 2차원 좌표계의 한 점 (x, y)를 멤버 변수로 가지며, 생성자 및 operator==를 가짐(모든 멤버 변수/함수는 public으로 선언, 생성자를 포함한 모든 멤버 함수는 class 외부에서 정의할 것, 이때 .h와 .cpp를 따로 만들어서 나누지 말고 main파일 안에 모두 구현할 것)

- get\_command()는 아래와 같은 메뉴를 화면에 출력하고, 키보드로부터 command를 입력 받음(do while을 사용, 잘못된 입력일 경우 메뉴를 다시 출력)

- main()의 for\_each()는 <algorithm> 라이브러리의 함수이며, myPoints의 모든 element를 화면에 출력.

- count\_val()은 container의 iterator를 입력으로 받음. count\_val() 내부에서는 Point 정보 (x, y)를 키보드로부터 입력 받고, 동일한 myPoints의 element중 동일한 좌표값을 가지는 element의 개수를 화면에 출력.

[시작코드]

int main() {

Point<int> myPoints[] = { {1, 2}, {-1, 5}, {3, 7}, {2, 4}, {1, 10}, {1, 2} };

string command;

while (1) {

get\_command(command);

if (command == "1") for\_each(/\*채울 것\*/);

else if (command == "2") count\_val<int>(/\*채울 것\*/);

else if (command == "0") break;

}

cout << "Exit program.." << endl;

return 0;

}

#include <iostream>

#include <algorithm>

#include <vector>

using namespace std;

template <typename T>

class Point {

public:

T x, y;

Point(T \_x, T \_y);

bool operator==(const Point& seek);

void display();

};

template <typename T>

Point<T>::Point(T \_x, T \_y) : x(\_x), y(\_y) {}

template <typename T>

bool Point<T>::operator==(const Point& seek) {

return this->x == seek.x && this->y == seek.y;

}

template <typename T>

void Point<T>::display() {

cout << "x: " << x << "\ty:" << y << endl;

}

void get\_command(string& command) {

do {

cout << "Enter the command : ";

cin >> command;

if (command == "1" || command == "2" || command == "0")

break;

} while (command != "1" && command != "2" && command != "0");

}

template <typename T>

void count\_val(Point<int>\* iter\_begin, Point<int>\* iter\_end) {

int cnt = 0;

int x, y;

cout << "x : ";

cin >> x;

cout << "y : ";

cin >> y;

for (auto& cursor = iter\_begin; cursor != iter\_end; cursor++) {

if ((\* cursor).x == x && (\* cursor).y == y)

cnt++;

}

cout << "The number of values :" << cnt << endl;

}

int main() {

Point<int> myPoints[] = { {1, 2}, {-1, 5}, {3, 7}, {2, 4}, {1, 10}, {1, 2} };

string command;

while (1) {

get\_command(command);

if (command == "1") for\_each(begin(myPoints), end(myPoints), [](Point<int> myPoints) { myPoints.display(); });

else if (command == "2") count\_val<int>(begin(myPoints), end(myPoints));

else if (command == "0") break;

}

cout << "Exit program.." << endl;

return 0;

}

아래 main 함수가 동작하도록 하는 3개의 클래스(Student, Course, KHUIS) 및 오버로딩된 연사자(<<) 함수를 작성하시오. 작성한 프로그램은 시작코드를 포함한 하나의 <학번\_문제>.cpp 파일 형태로 업로드 하시오. 파일명에 한글이나 특수문자는 사용하지 마시오. [20점]

ex) 2021000111\_bonus\_q2.cpp 파일 업로드

[요구사항]

- main함수와 같은 코드를 실행했을 때 그림1과 동일한 결과가 출력되도록 한다.

- 오버로딩된 연산자 및 KHUIS 클래스는 Student 클래스와 Course클래스의 friend 함수/클래스로서 Student 클래스와 Course 클래스의 private 맴버 변수들을 이용해서 화면에 정보를 출력한다.

- Main()함수에서 동적 할당된 pCourse는 main()함수 종료 전 할당 해제해 준다.

[클래스 변수 정보]

class Student //학생 클래스

private: //맴버 변수

string name; //학생 이름

string id; //학생 아이디

string code[100]; //들은 수업(course)의 code (최대 100개)

int numCourseTaken; //들은 수업의 총 개수

class Course //수업 클래스

private: //맴버 변수

string name; //수업의 이름

string code; //수업의 코드

class KHUIS //종정시 클래스: 학생정보와 학생이 듣고있는 수업 정보를 출력, Student class와 Course class에서 friend class로 정의됨

public:

//학생의 정보를 입력으로 받아 학생정보 출력

// 학생의 들은 수업(code)에 대한 수업 이름을 Course class에서 찾아서 출력

//numCourse(int)는 \_course 배열의 길이를 의미

void printInfo(Student& \_stu, Course\* \_course, int numCourse);

[Main 함수]

#include <iostream>

#include <string>

#include <iomanip>

using namespace std;

//클래스 구현

int main() {

int numTotalCourse = 3; //총 과목의 개수

Course \*pCourse = new Course[numTotalCourse];

pCourse[0].setName("객체지향프로그래밍");

pCourse[0].setCode("CSE100");

pCourse[1].setName("영상처리");

pCourse[1].setCode("CSE200");

pCourse[2].setName("머신러닝");

pCourse[2].setCode("CSE300");

cout << pCourse[0].getCode() << '\t' << pCourse->getName() << endl; // 그림1. (1)

cout << setw(50) << setfill('-') << "" << endl;

cout << pCourse[0] << endl; // 그림1. (2)

cout << pCourse[1] << endl;

cout << pCourse[2] << endl;

cout << setw(50) << setfill('-') << "" << endl;

Student stu1("홍길동", "20181004"), stu2;

stu1.addCourse("CSE100");

stu1.addCourse("CSE200");

stu2.setName("김영희");

stu2.setId("20182000");

stu2.addCourse("CSE100");

stu2.addCourse("CSE300");

cout << stu1.getId() << '\t' << stu1.getName() << '\t' << stu1.getCode(0) << endl; // 그림1. (3)

cout << setw(50) << setfill('-') << "" << endl;

cout << stu1 << endl; // 그림1. (4)

cout << stu2 << endl;

cout << setw(50) << setfill('-') << "" << endl;

KHUIS khuis;

khuis.printInfo(stu1, pCourse, numTotalCourse); // 그림1. (5)

khuis.printInfo(stu2, pCourse, numTotalCourse);

//pCourse 할당 해제 할 것

return 0;

}

#include <iostream>

#include <string>

#include <iomanip>

using namespace std;

class Student {

private:

string name;

string id;

string code[100]; // 들은 수업(course)의 code (최대 100개)

int numCourseTaken = 0; // 들은 수업의 총 갯수

public:

Student () {}

Student (string \_name, string \_id) : name(\_name), id(\_id) {}

void setName(string \_name) {

name = \_name;

}

void setId(string \_id) {

id = \_id;

}

string getName() {

return name;

}

string getId() {

return id;

}

string getCode(int x) {

return code[x];

}

void addCourse(string \_code) {

code[numCourseTaken] = \_code;

numCourseTaken++;

}

friend class KHUIS;

};

class Course {

private:

string name; // 수업 이름

string code; // 수업 코드

public:

void setName(string \_name) {

name = \_name;

}

void setCode(string \_code) {

code = \_code;

}

string getName() {

return name;

}

string getCode() {

return code;

}

friend class KHUIS;

};

ostream& operator<<(ostream& os, Student stu) {

cout << "성명 : " << stu.getName() << " 학번 : " << stu.getId() << endl;

return os;

}

ostream& operator<<(ostream& os, Course cor) {

cout << "과목명 : " << cor.getName() << ", 학수번호 : " << cor.getCode() << endl;

return os;

}

class KHUIS { // 듣고 있는 수업 정보를 출력, Student / Course 랑 friend class

public:

void printInfo(Student& \_stu, Course\* \_course, int numCourse) {

cout << endl << "\* 학생 정보" << endl;

cout << "성명 : " << \_stu.getName() << " | 학번 : " << \_stu.getId() << endl;

cout << endl << "\* 수업 정보" << endl;

for (int i = 0; i < numCourse; i++) {

if (\_course[i].code == \_stu.getCode(i)) {

cout << \_course[i].name << endl;

}

for (int j = 0; j < numCourse; j++) {

if (i != j && \_course[i].code == \_stu.getCode(j)) {

cout << \_course[i].name << endl;

}

}

}

}

};

int main() {

int numTotalCourse = 3; //총 과목의 개수

Course\* pCourse = new Course[numTotalCourse];

pCourse[0].setName("객체지향프로그래밍");

pCourse[0].setCode("CSE100");

pCourse[1].setName("영상처리");

pCourse[1].setCode("CSE200");

pCourse[2].setName("머신러닝");

pCourse[2].setCode("CSE300");

cout << pCourse[0].getCode() << '\t' << pCourse->getName() << endl; // 그림1. (1)

cout << setw(50) << setfill('-') << "" << endl;

cout << pCourse[0] << endl; // 그림1. (2)

cout << pCourse[1] << endl;

cout << pCourse[2] << endl;

cout << setw(50) << setfill('-') << "" << endl;

Student stu1("홍길동", "20181004"), stu2;

stu1.addCourse("CSE100");

stu1.addCourse("CSE200");

stu2.setName("김영희");

stu2.setId("20182000");

stu2.addCourse("CSE100");

stu2.addCourse("CSE300");

cout << stu1.getId() << '\t' << stu1.getName() << '\t' << stu1.getCode(0) << endl; // 그림1. (3)

cout << setw(50) << setfill('-') << "" << endl;

cout << stu1 << endl; // 그림1. (4)

cout << stu2 << endl;

cout << setw(50) << setfill('-') << "" << endl;

KHUIS khuis;

khuis.printInfo(stu1, pCourse, numTotalCourse); // 그림1. (5)

khuis.printInfo(stu2, pCourse, numTotalCourse);

//pCourse 할당 해제 할 것

delete[] pCourse;

return 0;

}

#include <iostream>

using namespace std;

int main(){

int i, j;

int \*\*\*a = new int\*\* [2];

a[0] = new int\* [4];

a[1] = &a[0][2];

for(i=0; i<2; i++){

for(j=0; j<2; j++){

a[i][j] = new int;

\*(a[i][j]) = i + j;

cout << \*a[i][j] << " ";

}

cout << endl;

}

return 0;

}

보너스 문제) 위 프로그램의 수행과정 및 결과를 memory map을 그려서 자세히 설명하라. 포인터가 어디를 가리키는지를 명확히 화살표로 표현할 것. 타블렛 PC를 활용해서 그림을 그리고 업로드 혹은, 종이에 펜으로 그려서 사진 촬영 후 업로드 가능. 단, zoom을 이용한 비디오 스트리밍중인 스마트폰을 이용한 사진 촬영은 시험 종료 15분 전부터 가능. [보너스 20점]

문제) 위의 프로그램을 적당히 수정하여, 2x2x5 element를 갖는 3차원의 정수 array a[2][2][5]를 pointer를 사용하여 생성하고, 각 a[i][j][k]에 i+j+k가 들어가도록 하라. cout을 이용해서 화면에 아래의 출력이 되도록 하라. 하나의 완성된 프로그램이 되도록 할 것. 실행이 되어야 함. 메모리 누수 발생시 감점. 작성한 프로그램은 시작코드를 포함한 하나의 <학번\_문제>.cpp 파일 형태로 업로드 하시오. 파일명에 한글이나 특수문자는 사용하지 마시오. [14점]

#include <iostream>

using namespace std;

int main() {

int i, j, k;

int\*\*\* a = new int\*\* [2];

for (i = 0; i < 2; i++) {

a[i] = new int\* [2];

for (j = 0; j < 2; j++) {

a[i][j] = new int[5];

for (k = 0; k < 5; k++) {

a[i][j][k] = i + j + k;

cout << a[i][j][k];

}

cout << endl;

}

cout << endl;

}

for (int i = 0; i < 2; i++) {

for (int j = 0; j < 2; j++) {

delete[] a[i][j];

}

}

for (int i = 0; i < 2; i++) {

delete[] a[i];

}

delete[] a;

return 0;

}

[출력 화면]

0 1 2 3 4

1 2 3 4 5

1 2 3 4 5

2 3 4 5 6

Fill in the blank [ ] by selecting an answer from the options below.

If we inherit a class into the derived class and provide a definition for one of the base class's function again inside the derived class, then that function is said to be [ ] , and this mechanism is called [ ].

[3점]

답변 선택 그룹

overloading, Function Overloading

overriden, Function Overridng @@ 정답

overriden, Function Overloading

overloaded, Function Overriding

이 문제에 플래그를 지정

문제 23점

Fill in the blank below in C++.

[ ] variables in a class: As the variables declared as static are initialized only once as they are allocated space in separate [ ] memory. So, the [ ] variables in a class are shared by the objects. There can not be multiple copies of the same [ ] variables for different objects. [3점]

답변 선택 그룹

public

local

private

static @@ 정답

global

아래의 소스코드를 기반으로 출력 조건을 만족하도록 복소수의 +, - 연산을 지원하는 프로그램을 작성하시오. 작성한 프로그램은 시작코드를 포함한 하나의 <학번\_문제>.cpp 파일 형태로 업로드 하시오. 파일명에 한글이나 특수문자는 사용하지 마시오. [20점]

#include <iostream>

using namespace std;

class Complex {

double re, im;

public:

Complex(double r, double i)

{

re = r; im = i;

}

Complex() { re = 0; im = 0; }

// + 연산자

// - 연산자

void print(){

cout << re << " + i" << im <<endl;

}

};

int main()

{

Complex a(.3, 8.4), b(4.5, 5.2), c(2.0, 7.7);

Complex sum, dif;

sum = a + b + 3.0; // 중요 : "3.0"과 정확히 동일하게 기입하여야 함

cout << "a + b + 3.0 = ";

sum.print();

dif = a - b - 2.0; // 중요 : "2.0"과 정확히 동일하게 기입하여야 함

cout << "a - b - 2.0 = ";

dif.print();

sum = a + b + 5 - c - 8; // 중요 : "5 - 8"과 정확히 동일하게 기입하여야 함

cout << "a + b + 5 - c - 8 = ";

sum.print();

return 0;

}

[설명]

+연산자, -연산자를 main() 함수의 코드와 같이 사용 가능하도록 작성 함

Main()함수의 중요로 지적된 사항의 숫자는 소수점 사항을 명확히 적용할 것(소수점 및 해당 사항이 다를 경우 감점)

#include <iostream>

using namespace std;

class Complex {

double re, im;

public:

Complex(double r, double i)

{

re = r; im = i;

}

Complex() { re = 0; im = 0; }

// + 연산자

Complex operator+(const Complex& co) {

Complex result(this->re + co.re, this->im + co.im);

return result;

}

// - 연산자

Complex operator-(const Complex& co) {

Complex result(this->re - co.re, this->im - co.im);

return result;

}

template <typename T>

Complex operator+(T a) {

Complex result(this->re + a, this->im);

return result;

}

template <typename T>

Complex operator-(T a) {

Complex result(this->re - a, this->im);

return result;

}

void print() {

cout << re << " + i" << im << endl;

}

};

int main()

{

Complex a(.3, 8.4), b(4.5, 5.2), c(2.0, 7.7);

Complex sum, dif;

sum = a + b + 3.0; // 중요 : "3.0"과 정확히 동일하게 기입하여야 함

cout << "a + b + 3.0 = ";

sum.print();

dif = a - b - 2.0; // 중요 : "2.0"과 정확히 동일하게 기입하여야 함

cout << "a - b - 2.0 = ";

dif.print();

sum = a + b + 5 - c - 8; // 중요 : "5 - 8"과 정확히 동일하게 기입하여야 함

cout << "a + b + 5 - c - 8 = ";

sum.print();

return 0;

}