설계문제 #2 소스

/\*------------------------------------------------------

파일이름: personType.h

-------------------------------------------------------\*/

#ifndef \_\_PERSON\_TYPE\_H\_\_

#define \_\_PERSON\_TYPE\_H\_\_

#include <string>

using namespace std;

class personType

{

public:

void print() const;

//Function to output the first name and last name in

//the form firstName lastName

void setName(string first, string last);

//Function to set firstName and lastName according to the

//parameters.

//Postcondition: firstName = first; lastName = last

personType& setFirstName(string first);

//Function to set the first name.

//Postcondition: firstName = first

// After setting the first name, a reference to the

// object, that is, the address of the object, is

// returned.

personType& setLastName(string last);

//Function to set the last name.

//Postcondition: lastName = last

// After setting the last name, a reference to the object,

// that is, the address of the object, is returned.

string getFirstName() const;

//Function to return the first name.

//Postcondition: The value of firstName is returned.

string getLastName() const;

//Function to return the last name.

//Postcondition: The value of lastName is returned.

personType(string first = "", string last = "");

//Constructor

//Sets firstName and lastName according to the parameters.

//Postcondition: firstName = first; lastName = last

private:

string firstName; //variable to store the first name

string lastName; //variable to store the last name

};

#endif

/\*------------------------------------------------------

파일이름: personType.cpp

-------------------------------------------------------\*/

#include "personType.h"

#include <iostream>

void personType::print() const

{

cout << firstName << " " << lastName;

}

void personType::setName(string first, string last)

{

firstName = first;

lastName = last;

}

string personType::getFirstName() const

{

return firstName;

}

string personType::getLastName() const

{

return lastName;

}

//Constructor with parameters

personType::personType(string first, string last)

{

firstName = first;

lastName = last;

}

personType& personType::setLastName(string last)

{

lastName = last;

return \*this;

}

personType& personType::setFirstName(string first)

{

firstName = first;

return \*this;

}

/\*------------------------------------------------------

파일이름: studentType.h

-------------------------------------------------------\*/

#ifndef \_\_STUDENT\_TYPE\_H\_\_

#define \_\_STUDENT\_TYPE\_H\_\_

#include <vector>

#include <string>

#include "personType.h"

#include "courseType.h"

using namespace std;

class studentType : public personType

{

public:

void setInfo(string fname, string lName, int ID, bool isTPaid, vector<courseType> courses);

//Function to set the student's information

//The private data members are set according

//to the parameters.

void print(ostream& out, double tuitionRate);

//Function to print the student's grade report

//The output is stored in a file specified by the

//parameter out.

studentType();

//Default constructor

//Postcondition: Data members are initialized to

//the default values.

int getHoursEnrolled();

//Function to return the credit hours a student

//is enrolled in.

//Postcondition: The number of credit hours in which a

// student is enrolled is calculated and returned.

double getGpa();

//Function to return the grade point average.

//Postcondition: The GPA is calculated and returned.

double billingAmount(double tuitionRate);

//Function to return the tuition fees

//Postcondition: The tuition fees due is calculated

// and returned.

private:

int sId; //variable to store the student ID

int numberOfCourses; //variable to store the number

//of courses

bool isTuitionPaid; //variable to indicate if the tuition

//is paid

vector<courseType> coursesEnrolled;//vector to store the courses

};

#endif

/\*------------------------------------------------------

파일이름: studentType.cpp

-------------------------------------------------------\*/

#include "studentType.h"

#include <algorithm>

#include <iomanip>

void studentType::setInfo(string fName, string lName, int ID, bool isTPaid, vector<courseType> courses)

{

setName(fName, lName);

sId = ID;

isTuitionPaid = isTPaid;

numberOfCourses = courses.size();

coursesEnrolled = courses;

sort(coursesEnrolled.begin(), coursesEnrolled.end());

}

studentType::studentType()

{

numberOfCourses = 0;

sId = 0;

isTuitionPaid = false;

}

void studentType::print(ostream& outp, double tuitionRate)

{

outp << "Student Name: " << personType::getFirstName()

<< " " << personType::getLastName() << endl; //Step 1

outp << "Student ID: " << sId << endl; //Step 2

outp << "Number of courses enrolled: "

<< numberOfCourses << endl << endl; //Step 3

outp << left;

outp << "Course No" << setw(15) << " Course Name"

<< setw(8) << "Credits"

<< setw(6) << "Grade" << endl; //Step 4

outp.unsetf(ios::left);

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

coursesEnrolled[i].print(outp, isTuitionPaid); //Step 5

outp << endl;

outp << "Total number of credit hours: "

<< getHoursEnrolled() << endl; //Step 6

outp << fixed << showpoint << setprecision(2); //Step 7

if (isTuitionPaid) //Step 8

outp << "Midsemester GPA: " << getGpa() << endl;

else

{

outp << "\*\*\* Grades are being held for not paying "

<< "the tuition. \*\*\*" << endl;

outp << "Amount Due: $" << billingAmount(tuitionRate)

<< endl;

}

outp << "-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*"

<< "-\*-\*-\*-\*-" << endl << endl;

}

int studentType::getHoursEnrolled()

{

int totalCredits = 0;

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

totalCredits += coursesEnrolled[i].getCredits();

return totalCredits;

}

double studentType::billingAmount(double tuitionRate)

{

return tuitionRate \* getHoursEnrolled();

}

double studentType::getGpa()

{

double sum = 0.0;

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

{

switch (coursesEnrolled[i].getGrade())

{

case 'A':

sum += coursesEnrolled[i].getCredits() \* 4;

break;

case 'B':

sum += coursesEnrolled[i].getCredits() \* 3;

break;

case 'C':

sum += coursesEnrolled[i].getCredits() \* 2;

break;

case 'D':

sum += coursesEnrolled[i].getCredits() \* 1;

break;

case 'F':

break;

default:

cout << "Invalid Course Grade" << endl;

}

}

if (getHoursEnrolled() != 0)

return sum / getHoursEnrolled();

else

return 0;

}

/\*------------------------------------------------------

파일이름: courseType.h

-------------------------------------------------------\*/

#ifndef \_\_COURSE\_TYPE\_H\_\_

#define \_\_COURSE\_TYPE\_H\_\_

#include <iostream>

#include <string>

using namespace std;

class courseType

{

public:

void setCourseInfo(string cName, string cNo, char grade, int credits);

//Function to set the course information

//The course information is set according to the

//incoming parameters.

//Postcondition: courseName = cName; courseNo = cNo;

// courseGrade = grade; courseCredits = credits;

void print(ostream& outp, bool isGrade);

//Function to print the course information

//If the bool parameter isGrade is true, the grade is

//shown, otherwise three stars are printed.

int getCredits();

//Function to return the credit hours

//The value of the private data member courseCredits

//is returned.

void getCourseNumber(string& cNo);

//Function to return the course number

//Postcondition: cNo = courseNo;

char getGrade();

//Function to return the grade for the course

//The value of the private data member courseGrade

//is returned.

bool operator==(const courseType&) const;

bool operator!=(const courseType&) const;

bool operator<=(const courseType&) const;

bool operator<(const courseType&) const;

bool operator>=(const courseType&) const;

bool operator>(const courseType&) const;

courseType(string cName = "", string cNo = "", char grade = '\*', int credits = 0);

//Constructor

//The object is initialized according to the parameters.

//Postcondition: courseName = cName; courseNo = cNo;

// courseGrade = grade; courseCredits = credits;

private:

string courseName; //variable to store the course name

string courseNo; //variable to store the course number

char courseGrade; //variable to store the grade

int courseCredits; //variable to store the course credits

};

#endif

/\*------------------------------------------------------

파일이름: courseType.cpp

-------------------------------------------------------\*/

#include <iomanip>

#include "courseType.h"

void courseType::setCourseInfo(string cName, string cNo, char grade, int credits)

{

courseName = cName;

courseNo = cNo;

courseGrade = grade;

courseCredits = credits;

}

void courseType::print(ostream& outp, bool isGrade)

{

outp << left; //Step 1

outp << setw(8) << courseNo << " "; //Step 2

outp << setw(15) << courseName; //Step 3

outp.unsetf(ios::left); //Step 4

outp << setw(3) << courseCredits << " "; //Step 5

if (isGrade) //Step 6

outp << setw(4) << courseGrade << endl;

else

outp << setw(4) << "\*\*\*" << endl;

}

courseType::courseType(string cName, string cNo, char grade, int credits)

{

setCourseInfo(cName, cNo, grade, credits);

}

int courseType::getCredits()

{

return courseCredits;

}

char courseType::getGrade()

{

return courseGrade;

}

void courseType::getCourseNumber(string& cNo)

{

cNo = courseNo;

}

bool courseType::operator==(const courseType& right) const

{

return (courseNo == right.courseNo);

}

bool courseType::operator!=(const courseType& right) const

{

return (courseNo != right.courseNo);

}

bool courseType::operator<=(const courseType& right) const

{

return (courseNo <= right.courseNo);

}

bool courseType::operator<(const courseType& right) const

{

return (courseNo < right.courseNo);

}

bool courseType::operator>=(const courseType& right) const

{

return (courseNo >= right.courseNo);

}

bool courseType::operator>(const courseType& right) const

{

return (courseNo > right.courseNo);

}

/\*------------------------------------------------------

파일이름: main.cpp

작성일: 2017. 02. 27

-------------------------------------------------------\*/

#include <iostream>

#include <fstream>

#include <string>

#include <algorithm>

#include <vector>

#include <iterator>

#include "studentType.h"

using namespace std;

void getStudentData(ifstream& infile, vector<studentType> &studentList);

void printGradeReports(ofstream& outfile, vector<studentType> studentList, double tuitionRate);

int main()

{

vector<studentType> studentList;

double tuitionRate;

ifstream infile;

ofstream outfile;

infile.open("stData.txt");

if (!infile)

{

cout << "Input file does not exist. "

<< "Program terminates." << endl;

return 1;

}

outfile.open("stDataOut.txt");

infile >> tuitionRate; //get the tuition rate

getStudentData(infile, studentList);

printGradeReports(outfile, studentList, tuitionRate);

return 0;

}

void getStudentData(ifstream& infile, vector<studentType> &studentList)

{

//Local variable

string fName; //variable to store the first name

string lName; //variable to store the last name

int ID; //variable to store the student ID

int noOfCourses; //variable to store the number of courses

char isPaid; //variable to store Y/N, that is,

//is tuition paid

bool isTuitionPaid; //variable to store true/false

string cName; //variable to store the course name

string cNo; //variable to store the course number

int credits; //variable to store the course credit hours

char grade; //variable to store the course grade

vector<courseType> courses; //vector of objects to store course

//information

courseType cTemp;

studentType sTemp;

infile >> fName; //Step 1

while (infile)

{

infile >> lName >> ID >> isPaid; //Step 1

if (isPaid == 'Y') //Step 2

isTuitionPaid = true;

else

isTuitionPaid = false;

infile >> noOfCourses; //Step 3

courses.clear();

for (int i = 0; i < noOfCourses; i++) //Step 4

{

infile >> cName >> cNo >> credits >> grade; //Step 4.a

cTemp.setCourseInfo(cName, cNo, grade, credits); //Step 4.b

courses.push\_back(cTemp); //Step 4.c

}

sTemp.setInfo(fName, lName, ID, isTuitionPaid, courses); //Step 5

studentList.push\_back(sTemp); //Step 6

infile >> fName; //Step 1

}//end while

}

void printGradeReports(ofstream& outfile, vector<studentType> studentList, double tuitionRate)

{

for (int count = 0; count < studentList.size(); count++)

studentList[count].print(outfile, tuitionRate);

}