// ========================

// Attached: HW\_8a, HW\_8b, HW\_8c

// ========================

// Program: HW\_8a

// ========================

// Programmer: Milo Fisher

// Class: CS 1B

// ========================

#include<iostream>

using namespace std;

class Animal

{

protected:

string name;

double weight;

public:

Animal(string name, double weight){this->name = name; this->weight = weight;}

~Animal(){}

};

class Bird : public Animal

{

private:

string featherColor;

public:

Bird(string name, double weight, string featherColor):Animal(name,weight){this->name = name; this->weight = weight; this->featherColor = featherColor;}

~Bird(){}

void showAnimal(){cout << "Birds are vertebrates in the Aves class.\nThis " << name << " has " << featherColor << " feathers and weighs " << weight << " pounds.\n\n";}

};

class Mammal : public Animal

{

private:

string hairColor;

public:

Mammal(string name, double weight, string hairColor):Animal(name,weight){this->name = name; this->weight = weight; this->hairColor = hairColor;}

~Mammal(){}

void showAnimal(){cout << "Mammals are vertebrates in the Mammalia class.\nThis " << hairColor << " " << name << " weighs " << weight << " pounds.\n\n";}

};

int main()

{

Bird bird("bird", 8.5, "gray");

Mammal mammal("cow",8.5,"black and white");

bird.showAnimal();

mammal.showAnimal();

return 0;

}

// ============== OUTPUT ================

/\*

Birds are vertebrates in the Aves class.

This bird has gray feathers and weighs 8.5 pounds.

Mammals are vertebrates in the Mammalia class.

This black and white cow weighs 8.5 pounds.

\*/

// ======================================

// ========================

// Attached: HW\_8a, HW\_8b, HW\_8c

// ========================

// Program: HW\_8b

// ========================

// Programmer: Milo Fisher

// Class: CS 1B

// ========================

#include<iostream>

using namespace std;

class Date

{

private:

int month;

int day;

int year;

public:

Date(){month = 0; day = 0; year = 0;}

Date(int month, int day, int year){this->month = month; this->day = day; this->year = year;}

~Date(){}

void setDate(int month, int day, int year){this->month = month; this->day = day; this->year = year;}

void displaydate(){cout << month << "/" << day << "/" << year;}

};

class FamousPeople : public Date

{

private:

string name;

string quote;

Date birthdate;

public:

FamousPeople(){name = ""; quote = ""; birthdate = Date();}

~FamousPeople(){}

void setName(string name){this->name = name;}

void setQuote(string quote){this->quote = quote;}

void setDate(int month, int day, int year){birthdate.Date::setDate(month,day,year);}

void displayPerson(){cout << "Name: " << name << "\nBirthdate: "; birthdate.Date::displaydate(); cout << "\nQuotation:\n\"" << quote << "\"\n\n";}

};

void screenClear();

int main()

{

FamousPeople people[3];

string input;

int m;

int d;

int y;

for(int i = 0; i < 3; i++)

{

if(i == 0)

cout << "Enter the first famous person: ";

else if(i == 1)

cout << "Enter the second famous person: ";

else if(i == 2)

cout << "Enter the third famous person: ";

getline(cin,input);

people[i].setName(input);

cout << "\nEnter the quotation:\n";

getline(cin,input);

people[i].setQuote(input);

cout << "\nEnter the birthdate:\nMonth: ";

cin >> m;

cout << "Day: ";

cin >> d;

cout << "Year: ";

cin >> y;

people[i].setDate(m,d,y);

cin.ignore();

screenClear();

}

cout << "Here are the famous people:\n\n";

people[0].displayPerson();

people[1].displayPerson();

people[2].displayPerson();

return 0;

}

void screenClear()

{

for(int i = 0; i < 10; i++)

{

cout << "\n\n\n\n\n\n\n\n\n\n";

}

}

// ============== OUTPUT ================

/\*

Name: DaVinci

Birthdate: 4/15/1452

Quotation:

"Look at mah painting bro"

Name: Einstein

Birthdate: 3/14/1879

Quotation:

"E = MC^2"

Name: Hitler

Birthdate: 4/20/1889

Quotation:

"Blitzkrieg Bop"

\*/

// ======================================

// ========================

// Attached: HW\_8a, HW\_8b, HW\_8c

// ========================

// Program: HW\_8c

// ========================

// Programmer: Milo Fisher

// Class: CS 1B

// ========================

#include <iostream>

#include <iomanip>

class GradedActivity

{

protected:

double score;

public:

GradedActivity(){score = 0;}

GradedActivity(double testScore){score = testScore;}

~GradedActivity(){}

void setScore(double testScore){score = testScore;}

double getScore(){return score;}

char getLetterGrade(){if(score >= 90){return 'A';} else if(score >= 80){return 'B';} else if(score >= 70){return 'C';} else if(score >= 60){return 'D';} else{return 'F';}}

};

class PassFailActivity: public GradedActivity

{

protected:

double minPassingScore;

public:

PassFailActivity(){minPassingScore = 0;}

PassFailActivity(double mps){minPassingScore = mps;}

~PassFailActivity(){}

void setMinPassingScore(double mps){minPassingScore = mps;}

double getMinPassingScore(){return minPassingScore;}

char getLetterGrade(){return GradedActivity::getLetterGrade();}

};

class PassFailExam: public PassFailActivity

{

private:

int numQuestions;

double pointsEach;

int numMissed;

public:

PassFailExam(){numQuestions = 0; pointsEach = 0; numMissed = 0;}

PassFailExam(int questions, int missed, double mps):PassFailActivity(mps){set(questions,missed); minPassingScore = mps;}

~PassFailExam(){}

void set(int questions, int missed){double numericScore; numQuestions = questions; numMissed = missed; pointsEach = 100.0/numQuestions; numericScore = 100.0 - (missed\*pointsEach); setScore(numericScore);}

double getNumQuestions(){return numQuestions;}

double getPointsEach(){return pointsEach;}

double getNumMissed(){return numMissed;}

};

using namespace std;

int main()

{

int questions; // Number of questions

int missed; // Number of questions missed

double minPassing; // The minimum passing score

// Define a GradedActivity object.

GradedActivity exam\_1(88.0);

// This demonstrates that the getLetterGrade() method is called by

// the GradedActivity object.

cout << "The student's grade on Exam #1 is " << exam\_1.getLetterGrade() << endl << endl << "---------------------------\n\n";

// Get the number of questions on the exam.

cout << "How many questions are on the exam? ";

cin >> questions;

// Get the number of questions the student missed.

cout << "How many questions did the student miss? ";

cin >> missed;

// Get the minimum passing score.

cout << "Enter the minimum passing score for this test: ";

cin >> minPassing;

// Instantiate a PassFailExam object.

PassFailExam exam\_2(questions, missed, minPassing);

// Display the test results.

cout << fixed << setprecision(1);

cout << "\nEach question counts " << exam\_2.getPointsEach() << " points.\n";

cout << "The minimum passing score is " << exam\_2.getMinPassingScore() << endl;

cout << "The student's exam score is " << exam\_2.getScore() << endl;

// This demonstrates that the getLetterGrade() method is redefined and is

// called by the PassFailExam object.

cout << "The student's grade on Exam #2 is " << exam\_2.getLetterGrade() << endl;

return 0;

}

// ============== OUTPUT ================

/\*

How many questions are on the exam? 30

How many questions did the student miss? 3

Enter the minimum passing score for this test: 21

Each question counts 3.3 points.

The minimum passing score is 21.0

The student's exam score is 90.0

The student's grade on Exam #2 is A

\*/

// ======================================

/\*

There were some differences between the output and the instructions for some of the functions,

so mine may not match up 100%.

\*/