





//Shrabanti Basu

//May 1, 2016

//Program 11

//This program calculates the course percentage and course grade for a class.

#include <iostream>

#include <iomanip>

#include <vector>

using namespace std;

//function prototypes

void initVectors(vector<double> &, vector<double> &);

void printScores(vector<double>, vector<double>);

void average(vector<double>);

double scoresVectors(vector<double>);

void grades(double);

int main()

{

cout << "Shrabanti Basu\n";

cout << "May 1, 2016\n";

cout << "Program 11\n";

cout << "This program calculates the course average and course grade for a class.\n\n"

<< "Enter the scores for exercises, programming assignments, midterm quiz and program,\n"

<< "and final quiz and program. The program will tell you the average and weighted \n"

<< "scores as well as the overall percentage score and letter grade.\n";

cout << "The final programming test and the programming assignments each gets 30% weightage\n"

<< "and each of the rest gets 10% weightage.\n\n";

//declare constant variables to store maximum scores possible for each exercise and program

const double MAX\_POINT\_EX = 2.0;

const double MAX\_POINT\_PR = 10.0;

//vectors to hold scores for exercises and programming assignements

vector<double> exercises;

vector<double> programs;

double exerciseScores, programScores; //to keep weighted scores of exercise and programs

//weighted scores for midterm multiplechoice and program; final multiple choice and program

double midtermMC, midtermPr, finalMC, finalPr;

double total; //total of all weighted scores

int size; //to store size of a vector

char repeat; // user input for repeat

do

{

size = 0;

total = 0.0;

//call the initVectors function to populate exercises and programs vectors

initVectors(exercises, programs);

//call scoresVectors function passing exercises vectors as parameter

//find the weightage of the scores for exercises

//add the weighted score to the accumulator

//then do the same for the programs vector

size = exercises.size();

exerciseScores = (scoresVectors(exercises) / (size \* MAX\_POINT\_EX)) \* 10;

total += exerciseScores;

size = programs.size();

programScores = (scoresVectors(programs) / (size \* MAX\_POINT\_PR)) \* 30;

total += programScores;

//get scores for the other exams, weight and accumulate

cout << "Enter midterm score for multiple choice: ";

cin >> midtermMC;

midtermMC = (midtermMC / 50.0) \* 10;

total += midtermMC;

cout << "Enter midterm score for programming: ";

cin >> midtermPr;

midtermPr = (midtermPr / 10.0) \* 10;

total += midtermPr;

cout << "Enter final exam score for multiple choice: ";

cin >> finalMC;

finalMC = (finalMC / 50.0) \* 10;

total += finalMC;

cout << "Enter final exam score for programming: ";

cin >> finalPr;

finalPr = (finalPr / 10.0) \* 30;

total += finalPr;

//call the printScores function to print scores for exercises and programs

printScores(exercises, programs);

//call average functions for exercises and programs vector

cout << "\nThe average for the exercises are: ";

average(exercises);

cout << "The average for the programs are: ";

average(programs);

cout << endl;

//print weightes scores for exercises, programs, exams, etc

cout << "Weighted scores for the exercises: " << exerciseScores << endl;

cout << "Weighted Scores for programs: " << programScores << endl;

cout << "Weighted Scores for midterm multiple choice: " << midtermMC << endl;

cout << "Weighted Scores for midterm programming: " << midtermPr << endl;

cout << "Weighted Scores for final multiple choice: " << finalMC << endl;

cout << "Weighted Scores for midterm programming: " << finalPr << endl;

cout << "\nTotal of all weighted scores: " << total << endl;

//call the grades function to print letter grade

grades(total);

cout << "Do you want to repeat? Enter Y for yes: ";

cin >> repeat;

} while (repeat == 'Y' || repeat == 'y');

return 0;

}

//function definition

//the initVectors gets scores for exercises and programming assignments and populates the vectors.

//both vectors are passed by reference

void initVectors(vector<double> &exercises, vector<double> &programs)

{

exercises.clear();

programs.clear();

double points; //to store points

cout << "Enter your scores for the exercises or a negative number to quit: ";

cin >> points;

while (points >= 0)

{

exercises.push\_back(points);

cout << "Enter your scores for the exercises or a negative number to quit: ";

cin >> points;

}

cout << endl;

cout << "Enter your scores for the programs or a negative number to quit: ";

cin >> points;

while (points >= 0)

{

programs.push\_back(points);

cout << "Enter your scores for the programs or a negative number to quit: ";

cin >> points;

}

cout << endl;

}

//Definition of printScores function

//This program takes in as parameters the exercises and the programs vectors and prints the elements

void printScores(vector<double> exercises, vector<double> programs)

{

int size = 0.0; //to store size of a vector

size = exercises.size();

//print number of scores and the points for exercises

cout << "\nThe scores for " << size << " exercises are: \n";

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

cout << exercises[i] << " ";

cout << endl;

size = programs.size();

//print number of scores and the points for programs

cout << "\The scores for " << size << " programs are:\n";

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

cout << programs[i] << " ";

cout << endl;

}

//function definition of average

//This program takes a vector and prints the average of all the elements

void average(vector<double> vect)

{

int size = 0;

size = vect.size();

double total = 0.0;

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

{

total += vect[i];

}

cout << total / size << endl;

}

//function definition of scoresVectors

//This program takes a vector as parameter and returns the total of all elements

double scoresVectors(vector<double> vect)

{

double scores = 0.0;

int size = 0;

size = vect.size();

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

{

scores += vect[i];

}

return scores;

}

//grades function definition

//calculates letter grade based on total

void grades(double num)

{

char letterGrade;

if (num < 60)

letterGrade = 'F';

else if (num < 70)

letterGrade = 'D';

else if (num < 80)

letterGrade = 'C';

else if (num < 90)

letterGrade = 'B';

else

letterGrade = 'A';

cout << "Your letter grade is: " << letterGrade << endl;

}