Week 11

**1. Describe briefly the following terminologies/symbols.**

**a. Class:**

A class in C++ is the building block that leads to Object-Oriented programming. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A C++ class is like a blueprint for an object.

**b. Object:**

An **Object** is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.

**c. Member function:**

A member function of a class is a function that has its definition or its prototype within the class definition like any other variable. It operates on any object of the class of which it is a member, and has access to all the members of a class for that object.

**d. Data member:**

The variables which are declared in any class by using any fundamental data types (like int, char, float etc) or derived data type (like class, structure, pointer etc.) are known as Data Members.

**e. Access specifiers:**

Access specifier determines or sets the boundary for the availability of class members (data members and member functions) beyond that class.

**f. Class definition:**

Class are a blueprint or a set of instructions to build a specific type of object. It is a basic concept of Object-Oriented Programming which revolve around the real-life entities. Class determines how an object will behave and what the object will contain.

**g. Encapsulations:**

Encapsulation is defined as wrapping up of data and information under a single unit. In Object Oriented Programming, Encapsulation is defined as binding together the data and the functions that manipulates them.

**h. Inheritance :**

Inheritance allows us to define a class in terms of another class, which makes it easier to create and maintain an application. This also provides an opportunity to reuse the code functionality and fast implementation time.

**2. Modify Class GradeBook**

**a. Include another string data member that represents the course lecturer’s name.**

**b. Provide a set function to change the lecturer’s name and a get function to retrieve it.**

**c. Modify the constructor to specify course name and lecturer’s name parameters.**

**d. Modify function displayMessage to output the welcome message and course name, and the string "This course is presented by: " followed by the lecturer’s name**

**Code:**

#include<iostream>

#include <string>

using namespace std;

class GradeBook

{

public:

GradeBook( string, string );

void setCourseName( string );

string getCourseName();

void setInstructorName(string);

string getInstructorName();

void displayMessage();

private:

string courseName;

string instructorName;

};

GradeBook::GradeBook( string course, string instructor )

{

courseName=course;

instructorName=instructor;

}

void GradeBook::setCourseName( string course)

{

if ( course.length() <= 25 )

courseName = course;

if ( course.length() > 25 )

{

courseName = course.substr( 0, 25 );

cout << "Name \"" << course <<"\" exceeds maximum length (25).\n"<< "Limiting courseName to first 25 characters.\n" << endl;

}

}

void GradeBook::setInstructorName(string instructor )

{

if ( instructor.length() <= 25 )

instructorName = instructor;

if ( instructor.length() > 25 )

{

instructorName = instructor.substr( 0, 25 );

cout << "Name \"" << instructor <<"\" exceeds maximum length (25).\n"<< "Limiting instructorName to first 25 characters.\n" << endl;

}

}

string GradeBook::getCourseName()

{

return courseName;

}

string GradeBook::getInstructorName()

{

return instructorName;

}

void GradeBook::displayMessage()

{

cout << "Welcome to the grade book for\n" << getCourseName()<< "!" << endl;

cout << "This course is presented by: \n" << getInstructorName()<< "." << endl;

}

int main()

{

GradeBook gradeBook1( "CS101 Introduction to Computer Science" ,"Pam");

GradeBook gradeBook2( "CS102 C++ Data Structures", "Dwight" );

cout << "gradeBook1's initial course name is: "<< gradeBook1.getCourseName()<< "\ngradeBook2's initial course name is: "<< gradeBook2.getCourseName() << endl;

cout << "gradeBook1's initial instructor name is: "<< gradeBook1.getInstructorName()<< "\ngradeBook2's initial instructor name is: "<< gradeBook2.getInstructorName() << endl;

gradeBook1.setCourseName( "CS101 C++ Programming" );

cout << "\ngradeBook1's course name is: "<< gradeBook1.getCourseName()<< "\ngradeBook2's course name is: "<< gradeBook2.getCourseName() << endl;

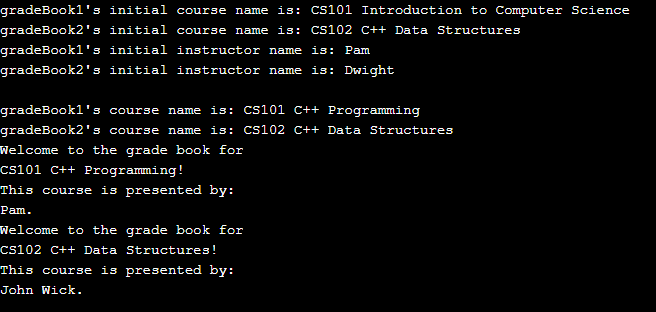
gradeBook2.setInstructorName("John Wick");

gradeBook1.displayMessage();

gradeBook2.displayMessage();

}

**Output:**



**3.**

**(a) Define a C++ base class named Rectangle containing length and width data members. From this class, derive a class named Box with another data member named depth. The member functions for the base class Rectangle should consist of a constructor and an area() function. The derived class Box should have a constructor, a volume() function and an override function named  area() that returns the surface area of the box.**

**(b) Include the classes written in part a, above in a working C++ program that creates an object  for each class and calls each member function for each class to test them. Verify the results manually.**

**Code:**

#include <iostream>

#include <cmath>

using namespace std;

const double PI = 2.0 \* asin(1.0);

// class declaration section

class Rectangle

{

protected:

double length;

double width;

public:

Rectangle(double = 4.0, double = 8.0); // constructor

double area();

};

// class implementation section for Point

Rectangle::Rectangle(double ll, double ww) // constructor

{

length = ll;

width = ww;

}

double Rectangle::area()

{

double area = length \* width;

return area;

}

// class declaration section where Box is derived from Rectangle

class Box : public Rectangle

{

protected:

double depth; // add one data member and

public: // two member functions

Box(double d = 5.0, double l = 3.0, double w = 4.0) : Rectangle(l,w), depth(d) {}

double volume();

double surf\_area();

};

// class implementation section

double Box::surf\_area() // calculates a volume

{

double surf\_area = (2\*length\*width) +(2 \*length \* depth) + (2\*width \* depth);

return surf\_area; // note the base function call

}

double Box::volume()

{

double volume = length \* width \* depth;

return volume;

}

int main()

{

Rectangle Rect\_1, Rect\_2; // create two Rectangle objects

Box Box\_1; // create one BOX object

cout << "The area of box 1 is " << Box\_1.area() << endl;

cout << "The volume of box 1 is " << Box\_1.volume() << endl;

cout << "The area of the rectangle is " << Rect\_2.area() << endl;

system("PAUSE");

return 0;

}

**Output:**

