Problem No 1:

Imagine a publishing company that markets both book and audiocassetie versions of its works. Create a class publication that stores the tle (a string) and price (type float) of a publication. From this class derive two classes: book, which adds a page count (type int), and tape, which adds a playing me in minutes (type float). Each of these three classes should have a getdata() function to get its data from the user at the keyboard, and a putdata() func on to display its data. Write a main() program to test the book and tape classes by creating instances of them, asking the user to fill in data with getdata(), and then displaying the data with putdata().

Code :

#include <iostream>

#include <string>

using namespace std;

class Publication

{

protected:

string title;

float price;

public:

// Get publication details from user

void getDetails()

{

cout << "Enter title: ";

cin >> title;

cout << "Enter price: $";

cin >> price;

}

// Display publication details

void showDetails()

{

cout << "\nTitle: " << title << endl;

cout << "Price: $" << price << endl;

}

};

class Book : public Publication

{

private:

int pageCount;

public:

// Get book details from user

void getDetails()

{

cout << "Enter page count: ";

cin >> pageCount;

}

// Display book details

void showDetails()

{

cout << "Page Count: " << pageCount << endl;

}

};

class Tape : public Publication

{

private:

float pTime;

public:

// Get tape details from user

void getDetails()

{

cout << "Enter playing time (minutes): ";

cin >> pTime;

}

// Display tape details

void showDetails()

{

cout << "Playing Time: " << pTime << " minutes" << endl;

}

};

int main()

{

Book book;

Tape tape;

cout << "\*\*\*\*\* Get Input from User \*\*\*\*\*"

<< endl

<< "\nEnter Book Details\n";

book.getDetails();

cout << "\*\*\*\*\*\*\*\*\*"

<< endl

<< "\nEnter Tape Details\n";

tape.getDetails();

cout << "\*\*\*\*\* Show Output \*\*\*\*\*"

<< endl

<< "\nBook Details\n";

book.showDetails();

cout << "\*\*\*\*\*\*\*\*\*"

<< endl

<< "\nTape Details\n";

tape.showDetails();

return 0;

}

Output :

|  |  |
| --- | --- |
| \*\*\*\*\*\*\*\*\*  Enter Book Details  Enter title: Robert Lafore  Enter price: $10  Enter page count: 6790  \*\*\*\*\*\*\*\*\*  Enter Tape Details  Enter title: JavaScript  Enter price: $20  Enter playing time (minutes): 20 Minutes | \*\*\*\*\*\*\*\*\*  Book Details  Title: Robert Lafore  Price: $10  Page Count: 6790  \*\*\*\*\*\*\*\*\*  Tape Details  Title: JavaScript  Price: $20  Playing Time: 20 minutes |

Problem No 2:

Start with the publication, book, and tape classes of Question 1. Add a base class sales that holds an array of three floats so that it can record the dollar sales of a particular publication for the last three months. Include a getdata() function to get three sales amounts from the user, and a putdata() function to display the sales figures. Alter the book and tape classes so they are derived from both publication and sales. An object of class book or tape should input and output sales data along with its other data. Write a main() function to create a book object and a tape object and exercise their input/output capabilities.

Code :

#include <iostream>

#include <string>

using namespace std;

class Publication

{

protected:

string title;

float price;

public:

// Get publication details from user

void getDetails()

{

cout << "Enter title: ";

cin >> title;

cout << "Enter price: $";

cin >> price;

}

// Display publication details

void showDetails()

{

cout << "\nTitle: " << title << endl;

cout << "Price: $" << price << endl;

}

};

class Sales

{

protected:

float sales[3];

// Array to store sales of 3 months

public:

// Get sales data from user

void getSalesData()

{

cout << "Enter sales figures for the last 3 months:\n";

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

{

cout << "Month " << i + 1 << ": $";

cin >> sales[i];

}

}

// Display sales data

void showSalesData()

{

cout << "Sales figures:\n";

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

{

cout << "Month " << i + 1 << ": $" << sales[i] << endl;

}

}

};

class Book : public Publication, public Sales

{

public:

// Combined get details for book (publication + sales)

void getBookDetails()

{

getDetails(); // Call Publication::getDetails

getSalesData(); // Call Sales::getSalesData

}

// Combined show details for book (publication + sales)

void showBookDetails()

{

showDetails(); // Call Publication::showDetails

showSalesData(); // Call Sales::showSalesData

}

};

class Tape : public Publication, public Sales

{

public:

// Combined get details for tape (publication + sales)

void getTapeDetails()

{

getDetails(); // Call Publication::getDetails

getSalesData(); // Call Sales::getSalesData

}

// Combined show details for tape (publication + sales)

void showTapeDetails()

{

showDetails(); // Call Publication::showDetails

showSalesData(); // Call Sales::showSalesData

}

};

int main()

{

Book book;

Tape tape;

cout << "\*\*\*\*\*\*\*\*\*"

<< endl

<< "\nEnter Book Details\n";

book.getBookDetails();

cout << "\*\*\*\*\*\*\*\*\*"

<< endl

<< "\nEnter Tape Details\n";

tape.getTapeDetails();

cout << "\*\*\*\*\*\*\*\*\*"

<< endl

<< "\nBook Details\n";

book.showBookDetails();

cout << "\*\*\*\*\*\*\*\*\*"

<< endl

<< "\nTape Details\n";

tape.showTapeDetails();

return 0;

}

|  |  |
| --- | --- |
| \*\*\*\*\*\*\*\*\*  Enter Book Details  Enter title: DevCpp  Enter price: $20  Enter sales figures for the last 3 months:  Month 1: $1000  Month 2: $200  Month 3: $3000  \*\*\*\*\*\*\*\*\*  Enter Tape Details  Enter title: Robert Lafore  Enter price: $70  Enter sales figures for the last 3 months:  Month 1: $500  Month 2: $1270  Month 3: $3276  \*\*\*\*\*\*\*\*\* | Book Details  Title: DevCpp  Price: $20  Sales figures:  Month 1: $1000  Month 2: $200  Month 3: $3000  \*\*\*\*\*\*\*\*\*  Tape Details  Title: Robert Lafore  Price: $70  Sales figures:  Month 1: $500  Month 2: $1270  Month 3: $3276 |

Output :

Problem No 3:

Assume that the publisher in Question 1 and 3 decides to add a third way to distribute books: on computer disk, for those who like to do their reading on their laptop. Add a disk class that, like book and tape, is derived from publication. The disk class should incorporate the same member functions as the other classes. The data item unique to this class is the disk type: either CD or DVD. You can use an enum type to store this item. The user could select the appropriate type by typing c or d.

Code :

#include <iostream>

#include <string>

using namespace std;

class publication

{

protected:

string title;

float price;

public:

// Get details from user

void getDetails()

{

cout << "Enter title: ";

cin >> title;

cout << "Enter price (pretend): ";

cin >> price;

}

// Display details

void showDetails()

{

cout << "\nTitle: " << title << endl;

cout << "Price: $" << price << endl;

}

};

class Book : public publication

{

protected:

int count;

public:

// Get book details

void getDetails()

{

cout << "Enter Number of Pages of Book: ";

cin >> count;

}

// Show book details

void showDetails()

{

cout << "Number of Pages of Books are: " << count << endl;

}

};

class Tape : public publication

{

protected:

float min;

public:

// Get tape details

void getDetails()

{

cout << "Enter Duration of Tape: ";

cin >> min;

}

// Show tape details

void showDetails()

{

cout << "Duration of Tape is: " << min << endl;

}

};

class Disk : public publication

{

private:

string type; // CD or DVD

public:

// Get disk details

void getDetails()

{

cout << "Enter disk type (CD or DVD): ";

cin >> type;

}

// Show disk details

void showDetails()

{

cout << "Disk Type: " << type << endl;

}

};

int main()

{

Book book;

Tape tape;

Disk disk;

cout << "\n\*\*\* Enter Book Details \*\*\*\n";

book.getDetails();

cout << "\n\*\*\* Enter Tape Details \*\*\*\n";

tape.getDetails();

cout << "\n\*\*\* Enter Disk Details \*\*\*\n";

disk.getDetails();

cout << "\n\*\*\* Book Details \*\*\*\n";

book.showDetails();

cout << "\n\*\*\* Tape Details \*\*\*\n";

tape.showDetails();

cout << "\n\*\*\* Disk Details \*\*\*\n";

disk.showDetails();

return 0;

}

Output :

|  |  |
| --- | --- |
| \*\*\* Enter Book Details \*\*\*  Enter Number of Pages of Book: 1700  \*\*\* Enter Tape Details \*\*\*  Enter Duration of Tape: 12  \*\*\* Enter Disk Details \*\*\*  Enter disk type (CD or DVD): DVD | \*\*\* Book Details \*\*\*  Number of Pages of Books are: 1700  \*\*\* Tape Details \*\*\*  Duration of Tape is: 12  \*\*\* Disk Details \*\*\*  Disk Type: DVD |

Problem No 4:

Derive a class called employee2 from the employee class in the EMPLOY program in this chapter. This new class should add a type double data item called compensation, and also an enum type called period to indicate whether the employee is paid hourly, weekly, or monthly. For simplicity you can change the manager, scientist, and laborer classes so they are derived from employee2 instead of employee. However, note that in many circumstances it might be more in the spirit of OOP to create a separate base class called compensation and three new classes manager2, scientist2, and laborer2, and use multiple inheritance to derive these three classes from the original manager, scientist, and laborer classes and from compensation. This way none of the original classes needs to be modified

Code :

#include <iostream>

#include <string>

using namespace std;

// Base class

class compensation

{

protected:

double compensation;

enum class Period

{

hourly,

weekly,

monthly

}

period;

public:

void getCompensationData()

{

cout << "Enter compensation: ";

cin >> compensation;

cout << "Select period:" << endl

<< "0. Hourly" << endl

<< "1. Weekly" << endl

<< "2. Monthly" << endl;

int periodChoice;

cin >> periodChoice;

switch (periodChoice)

{

case 0:

period = Period::hourly;

break;

case 1:

period = Period::weekly;

break;

case 2:

period = Period::monthly;

break;

default:

cout << "Invalid choice. Setting period to hourly by default." << endl;

period = Period::hourly;

}

}

void putCompensationData() const

{

cout << "Compensation: " << compensation << endl;

cout << "Period: ";

switch (period)

{

case Period::hourly:

cout << "Hourly";

break;

case Period::weekly:

cout << "Weekly";

break;

case Period::monthly:

cout << "Monthly";

break;

default:

cout << "Unknown";

}

cout << endl;

}

};

// Original employee class

class employee

{

protected:

string name;

long number;

public:

void getdata()

{

cout << "Enter name: ";

cin >> name;

cout << "Enter number: ";

cin >> number;

}

void putdata() const

{

cout << "Name: " << name << endl;

cout << "Number: " << number << endl;

}

};

class manager2 : public employee, public compensation

{

private:

string title;

int dues;

public:

void getdata()

{

employee::getdata();

cout << "Enter title: ";

cin >> title;

cout << "Enter dues: ";

cin >> dues;

}

void putdata() const

{

employee::putdata();

cout << "Title: " << title << "\n";

cout << "Dues: " << dues << "\n";

}

};

class scientist2 : public employee, public compensation

{

private:

int publications;

public:

void getdata()

{

employee::getdata();

cout << "Enter number of publications: ";

cin >> publications;

}

void putdata() const

{

employee::putdata();

cout << "Publications: " << publications << "\n";

}

};

class laborer2 : public employee, public compensation {};

int main()

{

manager2 m;

scientist2 s;

laborer2 l;

cout << "Enter manager data:" << endl;

m.getdata();

m.getCompensationData();

cout << "\nManager data:" << endl;

m.putdata();

m.putCompensationData();

cout << "\nEnter scientist data:" << endl;

s.getdata();

s.getCompensationData();

cout << "\nScientist data:" << endl;

s.putdata();

s.putCompensationData();

cout << "\nEnter laborer data:" << endl;

l.getdata();

l.getCompensationData();

cout << "\nLaborer data:" << endl;

l.putdata();

l.putCompensationData();

return 0;

}

Output :

|  |  |
| --- | --- |
| Enter manager data:  Enter name: Humna  Enter number: 123  Enter title: Manager  Enter dues: 83  Enter compensation: 78  Select period:  0. Hourly  1. Weekly  2. Monthly  1  Manager data:  Name: Humna  Number: 123  Title: Manager  Dues: 83  Compensation: 78  Period: Weekly  Enter scientist data:  Enter name: Hamna  Enter number: 345  Enter number of publications: 786  Enter compensation: 98 | Select period:  0. Hourly  1. Weekly  2. Monthly  2  Scientist data:  Name: Hamna  Number: 345  Publications: 786  Compensation: 98  Period: Monthly  Enter laborer data:  Enter name: Humna  Enter number: 456  Enter compensation: 987  Select period:  0. Hourly  1. Weekly  2. Monthly  0  Laborer data:  Name: Humna  Number: 456  Compensation: 987  Period: Hourly |

Problem No 5:

Create a simple inheritance hierarchy for a Shape class, Circle class, and Rectangle class. The Shape class should be the base class, and Circle and Rectangle should be derived classes. Implement the following in C++: Shape Class: Attributes: color (type std::string). Methods: A constructor to initialize the color and a method printColor to display the color. Circle Class: Attributes: radius (type double). Methods: A constructor to initialize the color and radius, a method calculateArea to calculate the area of the circle (area = p \* radius \* radius), and a method printArea to display the area. Rectangle Class: Attributes: length and width (type double). Methods: A constructor to initialize the color, length, and width, a method calculateArea to calculate the area of the rectangle (area = length \* width), and a method printArea to display the area.

Code :

#include <iostream>

using namespace std;

// Base class

class Shape

{

protected:

string colour;

public:

Shape()

{

cout << "Enter Shape's Colour: ";

cin >> colour;

}

void printColour()

{

cout << "Colour: " << colour << endl;

}

};

// Derived class

class Circle : public Shape

{

private:

double radius;

double area;

public:

Circle()

{

cout << "Enter Radius of Circle: ";

cin >> radius;

}

void calculateArea()

{

area = 3.14 \* radius \* radius;

}

void printArea()

{

cout << "Area of Circle: " << area << endl;

}

};

// Derived class

class Rectangle : public Shape

{

private:

double length, width;

double area;

public:

Rectangle()

{

cout << "Enter Length of Rectangle: ";

cin >> length;

cout << "Enter Width of Rectangle: ";

cin >> width;

}

void calculateArea()

{

area = length \* width;

}

void printArea()

{

cout << "Area of Rectangle: " << area << endl;

}

};

int main()

{

cout << " >> Enter Circle Details << "

<< endl;

Circle obj1;

obj1.calculateArea();

cout << "\n >> Enter Rectangle Details << "

<< endl;

Rectangle obj2;

obj2.calculateArea();

cout << "\n << Rectangle Details >> "

<< endl;

obj2.printColour();

obj2.printArea();

cout << "\n << Circle Details >> "

<< endl;

obj1.printColour();

obj1.printArea();

return 0;

}

|  |  |
| --- | --- |
| >> Enter Circle Details <<  What is Colour of the shape? : Blue  What is Radius of Circle? : 9  >> Enter Rectangle Details <<  What is Colour of the shape? : Pink  What is Length of Rectangle? : 7  What is Width of Rectangle? : 5 | << Rectangle Details >>  Colour: Pink  Area of Rectangle: 35  << Circle Details >>  Colour: Blue  Area of Circle: 254.34 |

Output :

Problem No 6:

Design a class hierarchy for an Employee management system. The base class should be Employee with derived classes SalariedEmployee and CommissionEmployee. Each class should have appropriate data members and member functions to handle the specific attributes and behaviors of each type of employee.

Employee: Should have data members for name, employee ID, and department. It should also have member functions to get and set these values.

Salaried Employee: Inherits from Employee and adds a data member for annual Salary. It should have member functions to get and set the salary, and to calculate the monthly pay.

Commission Employee: Inherits from Employee and adds data members for sales and commission Rate. It should have member functions to get and set these values, and to calculate the total pay based on sales and commission rate.

Code :

#include<iostream>

using namespace std;

// Base class for Employee

class Employee

{

protected:

string name;

int id;

string department;

public:

void getInfo()

{

cout<<"Enter Employee Name: ";

cin>>name;

cout<<"Enter Employee Id: ";

cin>>id;

cout<<"Enter Employee Department: ";

cin>>department;

}

void setInfo()

{

cout<<"Employee Name: "<<name<<endl;

cout<<"Employee Id: "<<id<<endl;

cout<<"Employee Department: "<<department<<endl;

}

};

// Derived class

class SalariedEmployee : public Employee

{

private:

double annualSalary;

double monthlySalary;

public:

void getInfo()

{

Employee::getInfo();

cout<<"Enter Annual Salary: ";

cin>>annualSalary;

}

void calculate()

{

monthlySalary = annualSalary / 12;

}

void setInfo()

{

Employee::setInfo();

cout<<"Annual Salary: "<<annualSalary<<endl;

cout<<"Monthly Salary: "<<monthlySalary<<endl;

}

};

// Derived class

class CommissionEmployee : public Employee

{

private:

double sales;

double commissionRate;

double totalSalary;

public:

void getInfo()

{

Employee::getInfo();

cout<<"Enter Sales: ";

cin>>sales;

cout<<"Enter Commission Rate: ";

cin>>commissionRate;

}

void calculate()

{

totalSalary = sales \* commissionRate;

}

void setInfo()

{

Employee::setInfo();

cout<<"Total Salary: "<<totalSalary<<endl;

}

};

int main()

{

cout<<"Enter Salaried Employee Info"<<endl;

SalariedEmployee obj1;

obj1.getInfo();

obj1.calculate();

cout<<"\nSalaried Employee Info"<<endl;

obj1.setInfo();

cout<<"\nEnter Commission Employee Info"<<endl;

CommissionEmployee obj2;

obj2.getInfo();

obj2.calculate();

cout<<"\nCommission Employee Info"<<endl;

obj2.setInfo();

return 0;

}

Output :

|  |  |
| --- | --- |
| Enter Salaried Employee Info  Enter Employee Name: Mishal  Enter Employee Id: 017  Enter Employee Department: Artificial  Enter Annual Salary: 2000000  Salaried Employee Info  Employee Name: Mishal  Employee Id: 17  Employee Department: Artificial  Annual Salary: 2e+006  Monthly Salary: 166667 | Enter Commission Employee Info  Enter Employee Name: Yumna  Enter Employee Id: 020  Enter Employee Department: CS  Enter Sales: 9876500  Enter Commission Rate: 686400  Commission Employee Info  Employee Name: Yumna  Employee Id: 20  Employee Department: CS  Total Salary: 6.77923e+012 |

⭐⭐⭐⭐⭐ The End ⭐⭐⭐⭐⭐