**Assignment-05**

// File: assignment no 5 task 1.cpp

// Date: 18-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\* Imagine a publishing company that markets both book and audiocassete versions of its works. Create a

class publication that stores the title (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() func on 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(). \*/

#include <iostream>

#include <string>

using namespace std;

class Publication {

protected:

string title;

float price;

public:

void getData() {

cout << "Enter the title of book: ";

cin>>title;

cout << "Enter the price of book: ";

cin >> price;

}

void putData() const {

cout << "Title of book: " << title << endl;

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

}

};

class Book : public Publication {

private:

int pageCount;

public:

void getData() {

Publication::getData();

cout << "Enter the page count of book: ";

cin >> pageCount;

}

void putData() const {

Publication::putData();

cout << "Page count of book: " << pageCount << endl;

}

};

class Tape : public Publication {

private:

float playingTime;

public:

void getData() {

Publication::getData();

cout << "Enter the time of book for plying in minutes: ";

cin >> playingTime;

}

void putData() const {

Publication::putData();

cout << "Playing time of book: " << playingTime << " minutes" << endl;

}

};

int main() {

Book myBook;

Tape myTape;

cout << "Enter details for book:" << endl;

myBook.getData();

cout << "\nEnter details for tape:" << endl;

myTape.getData();

system("CLS");

cout << "\nBook details:" << endl;

myBook.putData();

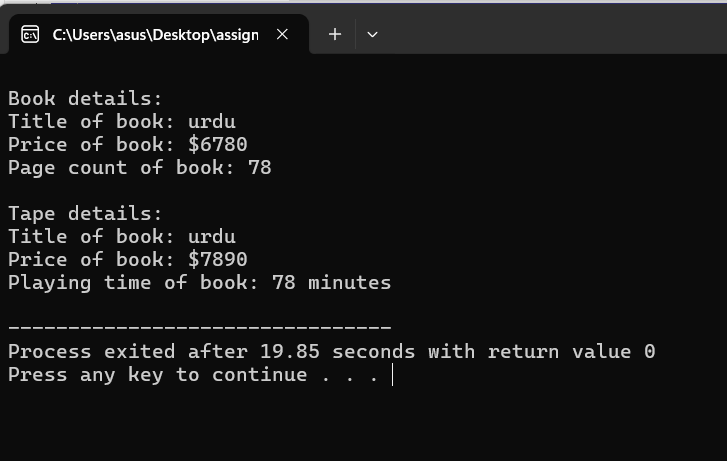
cout << "\nTape details:" << endl;

myTape.putData();

return 0;

}

Output:



// File: assignment no 5 task 2.cpp

// Date: 18-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\* Start with the publication, book, and tape classes of Quesion 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() func on

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() func on to create a book object and a tape object and exercise their input/output

capabilities. \*/

#include <iostream>

#include <string>

using namespace std;

class publication

{

protected:

string title;

float price;

public:

void getdata()

{

cout << "Enter the title of book: ";

cin>>title;

cout << "Enter the price of book: ";

cin >> price;

}

void putdata() const

{

cout << "Title of book: " << title << endl;

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

}

};

class sales

{

protected:

float salesData[3];

public:

void getdata()

{

cout << "Enter the book sales for the last three months: " << endl;

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

{

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

cin >> salesData[i];

}

}

void putdata() const

{

cout << "Enter the book sales for the last three months: " << endl;

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

{

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

}

}

};

class book : public publication, public sales

{

private:

int pageCount;

public:

void getdata()

{

publication::getdata();

cout << "Enter the page count of book: ";

cin >> pageCount;

sales::getdata();

}

void putdata() const

{

publication::putdata();

cout << "page count of book: " << pageCount << endl;

sales::putdata();

}

};

class tape : public publication, public sales

{

private:

float playingTime;

public:

void getdata()

{

publication::getdata();

cout << "Enter the time of book for plying in minutes: ";

cin >> playingTime;

sales::getdata();

}

void putdata() const

{

publication::putdata();

cout << "Playing time of book: " << playingTime << " minutes" << endl;

sales::putdata();

}

};

int main()

{

book b ;

tape t ;

cout << "Enter data for book:" << endl;

b.getdata();

cout << "\nEnter data for tape:" << endl;

t.getdata();

cout << "\nData for book:" << endl;

b.putdata();

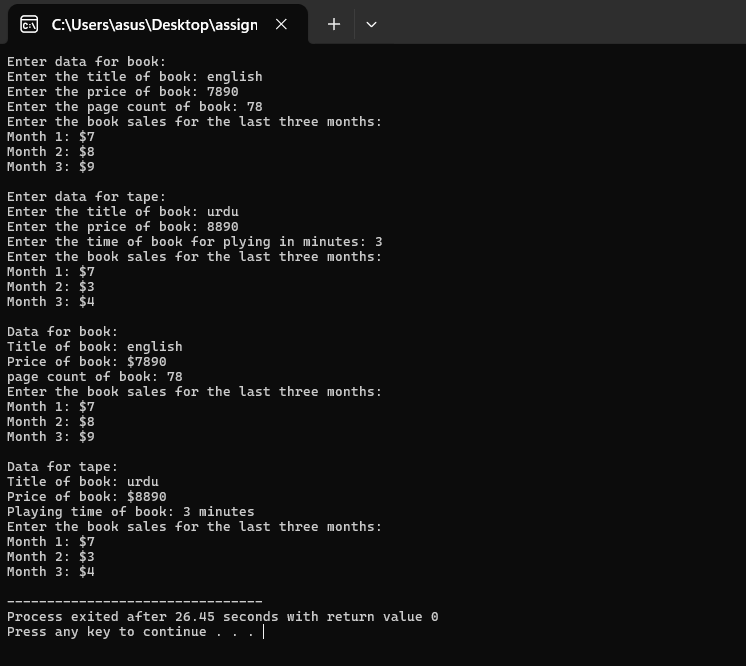
cout << "\nData for tape:" << endl;

t.putdata();

return 0;

}

Output:



// File: assignment no 5 task 3.cpp

// Date: 18-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\*Assume that the publisher in Question 1 and 2 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. \*/

#include <iostream>

#include <string>

using namespace std;

class Publication {

protected:

string title;

float price;

public:

void getData() {

cout << "Enter the title of book: ";

cin>>title;

cout << "Enter the price of book: ";

cin >> price;

}

void putData() const {

cout << "Title of book: " << title << endl;

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

}

};

class Sales {

protected:

float sale[3];

public:

void getData() {

cout << "\nEnter the book sales for the last three months:\n";

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

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

cin >> sale[i];

}

}

void putData() const {

cout << "enter the book Sales for the last three months:\n";

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

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

}

}

};

class Book : public Publication, public Sales {

private:

int pageCount;

public:

void getData() {

Publication::getData();

cout << "Enter the page count of book: ";

cin >> pageCount;

Sales::getData();

}

void putData() const {

Publication::putData();

cout << "Page count of book: " << pageCount << endl;

Sales::putData();

}

};

class Tape : public Publication, public Sales {

private:

float playingTime;

public:

void getData() {

Publication::getData();

cout << "Enter the time of book for playing in minutes: ";

cin >> playingTime;

Sales::getData();

}

void putData() const {

Publication::putData();

cout << "Playing time of book: " << playingTime << " minutes" << endl;

Sales::putData();

}

};

class Disk : public Publication, public Sales {

private:

enum DiskType {CD, DVD} diskType;

public:

void getData() {

Publication::getData();

char diskTypeInput;

cout << "Enter disk type of book (c for CD, d for DVD): ";

cin >> diskTypeInput;

if (diskTypeInput == 'c' || diskTypeInput == 'C') {

diskType = CD;}

else {

diskType = DVD;}

Sales::getData();

}

void putData() const {

Publication::putData();

cout << "Disk type of book: " << (diskType == CD ? "CD" : "DVD") << endl;

Sales::putData();

}

};

int main() {

Book b;

Tape t;

Disk d;

cout << "Enter details for book:" << endl;

b.getData();

cout << "\nEnter details for tape:" << endl;

t.getData();

cout << "\nEnter details for disk:" << endl;

d.getData();

system("CLS");

cout << "\nBook details:" << endl;

b.putData();

cout << "\nTape details:" << endl;

t.putData();

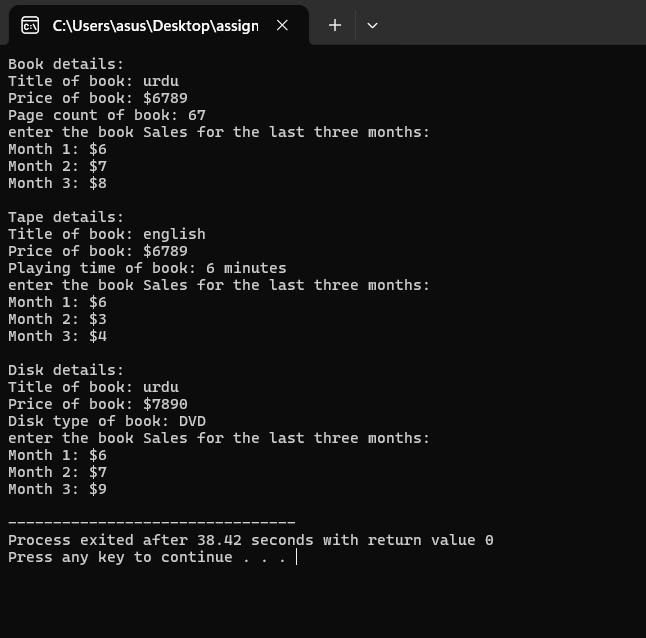
cout << "\nDisk details:" << endl;

d.putData();

return 0;

}

Output:



// File: assignment no 5 task 4 .cpp

// Date: 18-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\* This program derives a new class Employee2 from the base class Employee and introduces additional

data members and functionalities. Employee2 includes a double type data member for compensation

and an enumeration type to represent the pay period (hourly, weekly, or monthly). This program also

showcases how to modify derived classes Manager, Scientist, and Laborer to inherit from Employee2. \*/

#include <iostream>

#include <string>

using namespace std;

class Compensation {

protected:

double compensation;

public:

enum class Period { HOURLY, WEEKLY, MONTHLY };

Period payPeriod;

};

class Employee {

protected:

string name;

long id;

public:

virtual void getData() {

cout << "Enter name: ";

cin>>name;

cout << "Enter ID: ";

cin >> id;

}

virtual void putData() const {

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

cout << "ID: " << id << endl;

}

};

class Employee2 : public Employee, public Compensation {

public:

void getData() override {

Employee::getData();

cout << "Enter compensation: ";

cin >> compensation;

char periodInput;

cout << "Enter pay period (h for Hourly, w for Weekly, m for Monthly): ";

cin >> periodInput;

switch (tolower(periodInput)) {

case 'h':

payPeriod = Period::HOURLY;

break;

case 'w':

payPeriod = Period::WEEKLY;

break;

case 'm':

payPeriod = Period::MONTHLY;

break;

default:

cout << "Invalid pay period input! Defaulting to Hourly." << endl;

payPeriod = Period::HOURLY;

}

}

void putData() const override {

Employee::putData();

cout << "Compensation: $" << compensation << " per ";

switch (payPeriod) {

case Period::HOURLY:

cout << "hour" << endl;

break;

case Period::WEEKLY:

cout << "week" << endl;

break;

case Period::MONTHLY:

cout << "month" << endl;

break;

}

}

};

class Manager2 : public Employee2 {};

class Scientist2 : public Employee2 {};

class Laborer2 : public Employee2 {};

int main() {

Manager2 manager;

Scientist2 scientist;

Laborer2 laborer;

cout << "Enter details for manager:" << endl;

manager.getData();

cout << "\nEnter details for scientist:" << endl;

scientist.getData();

cout << "\nEnter details for laborer:" << endl;

laborer.getData();

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

manager.putData();

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

scientist.putData();

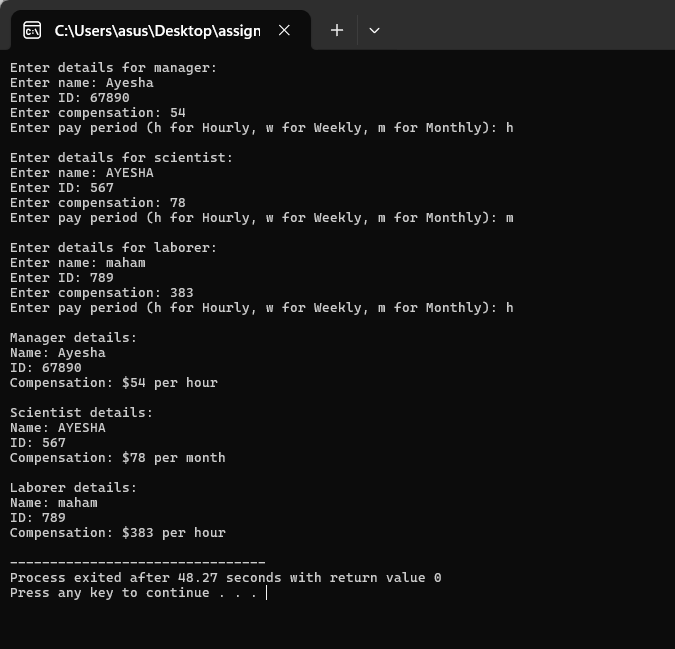
cout << "\nLaborer details:" << endl;

laborer.putData();

return 0;

}

Output:



// File: assignment no 5 task 5.cpp

// Date: 18-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\* This program demonstrates a simple inheritance hierarchy for shapes including a base Shape class

and derived Circle and Rectangle classes. The Shape class has a color attribute and a method to

print the color. The Circle class includes a radius attribute and methods to calculate and print

the area. The Rectangle class includes length and width attributes along with methods to calculate

and print the area. \*/

#include <iostream>

#include <string>

#include <cmath>

using namespace std;

const double PI = 3.14159;

// Base Class

class Shape {

protected:

string color;

public:

// Constructor to initialize color

Shape(const string& color) : color(color) {}

// Method to print color

void printColor() const {

cout << "Color: " << color << endl;

}

};

// Derived Circle Class

class Circle : public Shape {

private:

double radius;

public:

// Constructor to initialize color and radius

Circle(const string& color, double radius) : Shape(color), radius(radius) {}

// Method to calculate area

double calculateArea() const {

return PI \* radius \* radius;

}

// Method to print area

void printArea() const {

cout << "Area of Circle: " << calculateArea() << endl;

}

};

// Derived Rectangle Class

class Rectangle : public Shape {

private:

double length, width;

public:

// Constructor to initialize color, length, and width

Rectangle(const string& color, double length, double width)

: Shape(color), length(length), width(width) {}

// Method to calculate area

double calculateArea() const {

return length \* width;

}

// Method to print area

void printArea() const {

cout << "Area of Rectangle: " << calculateArea() << endl;

}

};

int main() {

// Create and display details for a Circle

Circle circle("Red", 10.0);

circle.printColor();

circle.printArea();

cout << endl;

// Create and display details for a Rectangle

Rectangle rectangle("Green", 5.0, 15.0);

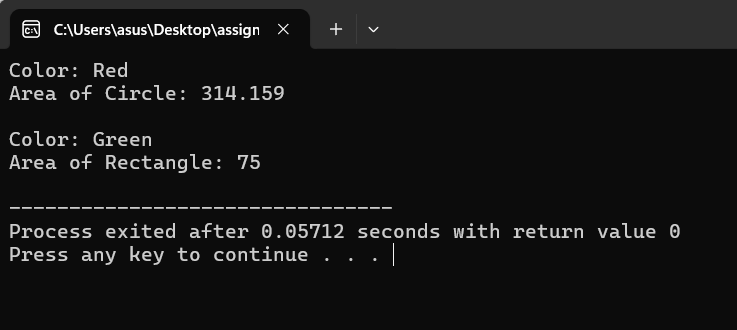
rectangle.printColor();

rectangle.printArea();

return 0;

}

Output:



// File: assignment no 5 task 6.cpp

// Date: 18-05-2024

// Name: Ayesha Imran

// Registration No: 2023-BS-AI-057

/\* This program designs a class hierarchy for an Employee Management System. The base class is Employee,

with derived classes SalariedEmployee and CommissionEmployee. Each class contains appropriate data

members and functions to handle the specific attributes and behaviors of each type of employee. \*/

#include <iostream>

#include <string>

using namespace std;

class Employee {

private:

string name;

int id;

string department;

public:

// Constructor to initialize Employee

Employee(const string& name, int id, const string& department)

: name(name), id(id), department(department) {}

// Getter for name

string getName() const {

return name;

}

// Getter for ID

int getID() const {

return id;

}

// Getter for department

string getDepartment() const {

return department;

}

};

class SalariedEmployee : public Employee {

private:

double annualSalary;

public:

// Constructor to initialize SalariedEmployee

SalariedEmployee(const string& name, int id, const string& department, double annualSalary)

: Employee(name, id, department), annualSalary(annualSalary) {}

// Method to calculate the monthly salary

double calculateMonthlySalary() const {

return annualSalary / 12;

}

};

class CommissionEmployee : public Employee {

private:

double sales;

double commissionRate;

public:

// Constructor to initialize CommissionEmployee

CommissionEmployee(const string& name, int id, const string& department, double sales, double commissionRate)

: Employee(name, id, department), sales(sales), commissionRate(commissionRate) {}

// Method to calculate total pay based on sales and commission rate

double calculateTotalPay() const {

return sales \* commissionRate;

}

};

int main() {

SalariedEmployee salariedEmployee("AYESHA", 1, "AI", 1000000);

cout << "Employee Name: " << salariedEmployee.getName() << endl;

cout << "Employee ID: " << salariedEmployee.getID() << endl;

cout << "Department: " << salariedEmployee.getDepartment() << endl;

cout << "Monthly Salary: $" << salariedEmployee.calculateMonthlySalary() << endl;

cout << endl;

CommissionEmployee commissionEmployee("Maham", 2, "SE", 20000, 1.5);

cout << "Employee Name: " << commissionEmployee.getName() << endl;

cout << "Employee ID: " << commissionEmployee.getID() << endl;

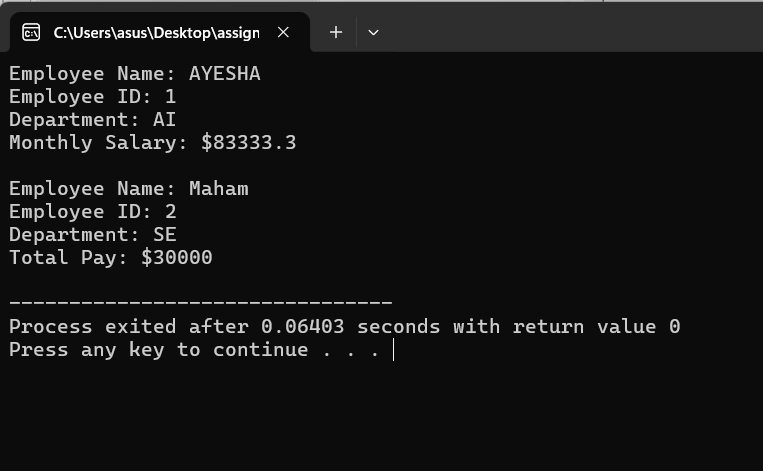
cout << "Department: " << commissionEmployee.getDepartment() << endl;

cout << "Total Pay: $" << commissionEmployee.calculateTotalPay() << endl;

return 0;

}

Output:

****