Inheritance Assignment

Program 1

```cpp
// File: Program1.cpp
// Date: 05-22-2024
// Name: Muhammad Tayyab Imran
// Registration No: 2023-BS-AI-2023
// Imagine a publishing company that markets both book and audiocasse e versions
of its works. Create a class publica on that stores the  tle (a string) and price
(type float) of a publica on. 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 crea ng instances of
them, asking the user to fill in data with getdata(), and then displaying the
data with putdata().

#include<iostream>
using namespace std;

// Base class for publications
class Publication
{
protected:
    string title; // Title of the publication
    float price; // Price of the publication

public:
    // Function to input data for the publication
    void getData()
    {
        cout<<"Enter Title: ";
        cin>>title;
        cout<<"Enter Price: ";
        cin>>price;
    }
```

```
    // Function to display data of the publication
    void putData()
    {
        cout<<"Title: "<<title<<endl;
        cout<<"Price: "<<price<<endl;
    }
};

// Derived class for books inheriting from Publication
class Book : public Publication
{
private:
    int pageCount; // Number of pages in the book

public:
    // Function to input data for the book
    void getData()
    {
        Publication::getData(); // Calling base class function to input common
data
        cout<<"Enter Page Count: ";
        cin>>pageCount;
    }

    // Function to display data of the book
    void putData()
    {
        Publication::putData(); // Calling base class function to display common
data
        cout<<"Page Count of Book: "<<pageCount<<endl;
    }
};

// Derived class for tapes inheriting from Publication
class Tape : public Publication
{
private:
    int playingTime; // Playing time of the tape in minutes

public:
```

```
    // Function to input data for the tape
    void getData()
    {
        Publication::getData(); // Calling base class function to input common
data
        cout<<"Enter Playing Time (in minutes): ";
        cin>>playingTime;
    }

    // Function to display data of the tape
    void putData()
    {
        Publication::putData(); // Calling base class function to display common
data
        cout<<"Playing Time of Tape: "<<playingTime<<" Min"<<endl;
    }
};

// Main function
int main()
{
    Book obj1; // Object for Book class
    cout<<"Enter Book Details"<<endl;
    obj1.getData(); // Inputting book details
    cout<<"\n\nBook Details"<<endl;
    obj1.putData(); // Displaying book details

    Tape obj2; // Object for Tape class
    cout<<"\n\nEnter Tape Details"<<endl;
    obj2.getData(); // Inputting tape details
    cout<<"\n\nTape Details"<<endl;
    obj2.putData(); // Displaying tape details

    return 0;
}
```

Output

```
Enter Book Details
Enter Title: OOP
Enter Price: 1000
Enter Page Count: 300


Book Details
Title: OOP
Price: 1000
Page Count of Book: 300


Enter Tape Details
Enter Title: OOP
Enter Price: 600
Enter Playing Time (in minutes): 45


Tape Details
Title: OOP
Price: 600
Playing Time of Tape: 45 Min
```

Program 2

```cpp
// File: Program2.cpp
// Date: 05-22-2024
// Name: Muhammad Tayyab Imran
// Registration No: 2023-BS-AI-2023
// 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.

#include<iostream>
using namespace std;
```

```
// Base class for sales
class Sales
{
protected:
    float array[3]; // Array to store sales data for 3 months

public:
    // Function to input sales data for 3 months
    void getData()
    {
        int n=1;
        for(int i=0; i<3; i++)
        {
            cout<<"Enter Sales of Month "<<n<<": ";
            cin>>array[i];
            n++;
        }
    }

    // Function to display sales data for 3 months
    void putData()
    {
        int n=1;
        for(int i=0; i<3; i++)
        {
            cout<<"Sales of Month "<<n<<": $"<<array[i]<<endl;
            n++;
        }
    }
};

// Base class for publications
class Publication
{
protected:
    string title; // Title of the publication
    float price; // Price of the publication

public:
    // Function to input publication data
    void getData()
    {
        cout<<"Enter Title: ";
        cin>>title;
        cout<<"Enter Price: ";
```

```
        cin>>price;
    }

    // Function to display publication data
    void putData()
    {
        cout<<"Title: "<<title<<endl;
        cout<<"Price: "<<price<<endl;
    }
};

// Derived class for books inheriting from Publication and Sales
class Book : public Publication, public Sales
{
private:
    int pageCount; // Number of pages in the book

public:
    // Function to input data for the book including sales data
    void getData()
    {
        Publication::getData(); // Input publication data
        cout<<"Enter Page Count: ";
        cin>>pageCount;
        Sales::getData(); // Input sales data
    }

    // Function to display data of the book including sales data
    void putData()
    {
        Publication::putData(); // Display publication data
        cout<<"Page Count of Book: "<<pageCount<<endl;
        Sales::putData(); // Display sales data
    }
};

// Derived class for tapes inheriting from Publication and Sales
class Tape : public Publication, public Sales
{
private:
    int playingTime; // Playing time of the tape in minutes

public:
    // Function to input data for the tape including sales data
    void getData()
```

```
    {
        Publication::getData(); // Input publication data
        cout<<"Enter Playing Time: ";
        cin>>playingTime;
        Sales::getData(); // Input sales data
    }

    // Function to display data of the tape including sales data
    void putData()
    {
        Publication::putData(); // Display publication data
        cout<<"Playing Time of Tape: "<<playingTime<<" Min"<<endl;
        Sales::putData(); // Display sales data
    }
};

// Main function
int main()
{
    Book obj1; // Object for Book class
    cout<<"Enter Book Details"<<endl;
    obj1.getData(); // Inputting book details
    cout<<"\n\nBook Details"<<endl;
    obj1.putData(); // Displaying book details


    Tape obj2; // Object for Tape class
    cout<<"\n\nEnter Tape Details"<<endl;
    obj2.getData(); // Inputting tape details
    cout<<"\n\nTape Details"<<endl;
    obj2.putData(); // Displaying tape details

    return 0;
}
```

Output

```
Enter Book Details
Enter Title: OOP
Enter Price: 1000
Enter Page Count: 300
Enter Sales of Month 1: 15
Enter Sales of Month 2: 24
Enter Sales of Month 3: 35


Book Details
Title: OOP
Price: 1000
Page Count of Book: 300
Sales of Month 1: $15
Sales of Month 2: $24
Sales of Month 3: $35


Enter Tape Details
Enter Title: OOP
Enter Price: 600
Enter Playing Time: 45
Enter Sales of Month 1: 14
Enter Sales of Month 2: 2
Enter Sales of Month 3: 15


Tape Details
Title: OOP
Price: 600
Playing Time of Tape: 45 Min
Sales of Month 1: $14
Sales of Month 2: $2
Sales of Month 3: $15
```

Program 3

```cpp
// File: Program3.cpp
// Date: 05-22-2024
// Name: Muhammad Tayyab Imran
// Registration No: 2023-BS-AI-2023
// 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 func..ons 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>
using namespace std;

// Base class for publications
class Publication
{
protected:
    string title; // Title of the publication
    float price; // Price of the publication

public:
    // Function to input publication data
    void getData()
    {
        cout<<"Enter Title: ";
        cin>>title;
        cout<<"Enter Price: ";
        cin>>price;
    }

    // Function to display publication data
    void putData()
    {
        cout<<"Title: "<<title<<endl;
        cout<<"Price: "<<price<<endl;
    }
};
```

```
// Base class for sales
class Sales
{
protected:
    float array[3]; // Array to store sales data for 3 months

public:
    // Function to input sales data for 3 months
    void getData()
    {
        int n=1;
        for(int i=0; i<3; i++)
        {
            cout<<"Enter Sales of Month "<<n<<": ";
            cin>>array[i];
            n++;
        }
    }

    // Function to display sales data for 3 months
    void putData()
    {
        int n=1;
        for(int i=0; i<3; i++)
        {
            cout<<"Sales of Month "<<n<<": $"<<array[i]<<endl;
            n++;
        }
    }
};

// Derived class for books inheriting from Publication and Sales
class Book : public Publication, public Sales
{
private:
    int pageCount; // Number of pages in the book

public:
    // Function to input data for the book including sales data
    void getData()
    {
        Publication::getData(); // Input publication data
        cout<<"Enter Page Count: ";
        cin>>pageCount;
        Sales::getData(); // Input sales data
```

```
    }

    // Function to display data of the book including sales data
    void putData()
    {
        Publication::putData(); // Display publication data
        cout<<"Page Count of Book: "<<pageCount<<endl;
        Sales::putData(); // Display sales data
    }
};

// Derived class for tapes inheriting from Publication and Sales
class Tape : public Publication, public Sales
{
private:
    int playingTime; // Playing time of the tape in minutes

public:
    // Function to input data for the tape including sales data
    void getData()
    {
        Publication::getData(); // Input publication data
        cout<<"Enter Playing Time: ";
        cin>>playingTime;
        Sales::getData(); // Input sales data
    }

    // Function to display data of the tape including sales data
    void putData()
    {
        Publication::putData(); // Display publication data
        cout<<"Playing Time of Tape: "<<playingTime<<" Min"<<endl;
        Sales::putData(); // Display sales data
    }
};

// Derived class for disks inheriting from Publication and Sales
class Disk : public Publication, public Sales
{
private:
    enum DiskType{CD, DVD}; // Enumeration for types of disks
    DiskType diskType; // Type of the disk

public:
    // Function to input data for the disk including sales data
```

```
    void getData()
    {
        Publication::getData(); // Input publication data
        char type;
        cout << "Enter disk type (c for CD, d for DVD): ";
        cin >> type;
        if (type == 'c' || type == 'C')
        {
            diskType = CD;
        }
        else if (type == 'd' || type == 'D')
        {
            diskType = DVD;
        }
        Sales::getData(); // Input sales data
    }

    // Function to display data of the disk including sales data
    void putData()
    {
        Publication::putData(); // Display publication data
        cout << "Disk Type: "<<(diskType == CD ? "CD" : "DVD")<< endl;
        Sales::putData(); // Display sales data
    }
};


// Main function
int main()
{
    Book obj1; // Object for Book class
    cout<<"Enter Book Details"<<endl;
    obj1.getData(); // Inputting book details
    cout<<"\n\nBook Details"<<endl;
    obj1.putData(); // Displaying book details

    Tape obj2; // Object for Tape class
    cout<<"\n\nEnter Tape Details"<<endl;
    obj2.getData(); // Inputting tape details
    cout<<"\n\nTape Details"<<endl;
    obj2.putData(); // Displaying tape details

    Disk obj3; // Object for Disk class
    cout<<"\n\nEnter Disk Details"<<endl;
    obj3.getData(); // Inputting disk details
```

```
    cout<<"\n\nDisk Details"<<endl;
    obj3.putData(); // Displaying disk details

    return 0;
}
```

Output

```
Enter Book Details
Enter Title: OOP
Enter Price: 1000
Enter Page Count: 300
Enter Sales of Month 1: 15
Enter Sales of Month 2: 25
Enter Sales of Month 3: 34


Book Details
Title: OOP
Price: 1000
Page Count of Book: 300
Sales of Month 1: $15
Sales of Month 2: $25
Sales of Month 3: $34


Enter Tape Details
Enter Title: OOP
Enter Price: 600
Enter Playing Time: 46
Enter Sales of Month 1: 12
Enter Sales of Month 2: 2
Enter Sales of Month 3: 14


Tape Details
Title: OOP
Price: 600
Playing Time of Tape: 46 Min
Sales of Month 1: $12
Sales of Month 2: $2
Sales of Month 3: $14


Enter Disk Details
Enter Title: OOP
Enter Price: 200
Enter disk type (c for CD, d for DVD): c
Enter Sales of Month 1: 14
Enter Sales of Month 2: 25
Enter Sales of Month 3: 36
```

Program 4

```cpp
// File: Program4.cpp
// Date: 05-22-2024
// Name: Muhammad Tayyab Imran
// Registration No: 2023-BS-AI-2023
// 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
compensa..on, 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, scien..st, 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 compensa..on and three
new classes manager2, scien..st2, and laborer2, and use mul..ple inheritance to
derive these three classes from the original manager, scientist, and laborer
classes and from compensayion. This way none of the original classes needs to be
modified
#include <iostream>
using namespace std;

// Base class for Compensation
class Compensation
{
protected:
    double compensation; // Compensation amount
    char period; // Period for compensation (h for hourly, w for weekly, m for
monthly)

public:
    // Constructor to input compensation and period
    Compensation()
    {
        cout<<"Enter Compensation: ";
        cin>>compensation;
        cout<<"Enter Period(h for Hourly, w for Weekly, m for Monthly): ";
        cin>>period;
    }

    // Function to display compensation details
    void display() const
    {
        cout << "Compensation: " << compensation << " per ";
        switch (period)
        {
```

```
            case 'h':
                cout<<"Hour";
                break;
            case 'w':
                cout<<"Week";
                break;
            case 'm':
                cout<<"Month";
                break;
        }
        cout << endl;
    }
};

// Base class for Employee
class Employee
{
protected:
    string name; // Name of the employee
    int id; // ID of the employee

public:
    // Constructor to input name and ID of the employee
    Employee()
    {
        cout<<"Enter Name: ";
        cin>>name;
        cout<<"Enter ID: ";
        cin>>id;
    }

    // Function to display employee details
    void display() const
    {
        cout << "Name: " << name << endl;
        cout << "ID: " << id << endl;
    }
};

// Derived class for Manager inheriting from Employee
class Manager : public Employee
{
public:
    Manager() {}
    void display() const
```

```
    {
        Employee::display();
    }
};

// Derived class for Scientist inheriting from Employee
class Scientist : public Employee
{
public:
    Scientist() {}
    void display() const
    {
        Employee::display();
    }
};

// Derived class for Laborer inheriting from Employee
class Laborer : public Employee
{
public:
    Laborer() {}
    void display() const
    {
        Employee::display();
    }
};

// Derived class for Manager2 inheriting from Manager and Compensation
class Manager2 : public Manager, public Compensation
{
public:
    Manager2() {}
    void display()
    {
        Manager::display();
        Compensation::display();
    }
};

// Derived class for Scientist2 inheriting from Scientist and Compensation
class Scientist2 : public Scientist, public Compensation
{
public:
    Scientist2() {}
    void display()
```

```
    {
        Scientist::display();
        Compensation::display();
    }
};

// Derived class for Laborer2 inheriting from Laborer and Compensation
class Laborer2 : public Laborer, public Compensation
{
public:
    Laborer2() {}
    void display()
    {
        Laborer::display();
        Compensation::display();
    }
};

// Main function
int main()
{
    cout<<"\nEnter Manager Info"<<endl;
    Manager2 manager; // Object for Manager2 class
    cout<<"\nEnter Scientist Info"<<endl;
    Scientist2 scientist; // Object for Scientist2 class
    cout<<"\nEnter Laborer Info"<<endl;
    Laborer2 laborer; // Object for Laborer2 class

    manager.display(); // Display manager info
    scientist.display(); // Display scientist info
    laborer.display(); // Display laborer info

    return 0;
}
```

Output

```
Enter Manager Info
Enter Name: Tayyab
Enter ID: 19
Enter Compensation: 500
Enter Period(h for Hourly, w for Weekly, m for Monthly): h

Enter Scientist Info
Enter Name: Haseeb
Enter ID: 33
Enter Compensation: 1000
Enter Period(h for Hourly, w for Weekly, m for Monthly): w

Enter Laborer Info
Enter Name: Zain
Enter ID: 52
Enter Compensation: 1500
Enter Period(h for Hourly, w for Weekly, m for Monthly): m

Name: Tayyab
ID: 19
Compensation: 500 per Hour

Name: Haseeb
ID: 33
Compensation: 1000 per Week

Name: Zain
ID: 52
Compensation: 1500 per Month
```

Program 5

```cpp
// File: Program5.cpp
// Date: 05-22-2024
// Name: Muhammad Tayyab Imran
// Registration No: 2023-BS-AI-2023
// 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 ini..alize the color and a method printColor to
display the color.
// Circle Class:
// Attributes: radius (type double).
// Methods: A constructor to ini..alize 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 ini..alize 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.

#include<iostream>
using namespace std;

// Base class for shapes
class Shape
{
protected:
    string colour; // Colour of the shape

public:
    // Constructor to input colour of the shape
    Shape()
    {
        cout<<"Enter Shape's Colour: ";
        cin>>colour;
    }

    // Function to print the colour of the shape
    void printColour()
    {
```

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

// Derived class for circles inheriting from Shape
class Circle : public Shape
{
private:
    double radius; // Radius of the circle
    double area; // Area of the circle

public:
    // Constructor to input radius of the circle
    Circle()
    {
        cout<<"Enter Radius of Circle: ";
        cin>>radius;
    }

    // Function to calculate the area of the circle
    void calculateArea()
    {
        area = 3.14 * radius * radius;
    }

    // Function to print the area of the circle
    void printArea()
    {
        cout<<"Area of Circle: "<<area<<endl;
    }
};

// Derived class for rectangles inheriting from Shape
class Rectangle : public Shape
{
private:
    double length, width; // Length and width of the rectangle
    double area; // Area of the rectangle

public:
    // Constructor to input length and width of the rectangle
    Rectangle()
    {
        cout<<"Enter Length of Rectangle: ";
        cin>>length;
```

```
        cout<<"Enter Width of Rectangle: ";
        cin>>width;
    }

    // Function to calculate the area of the rectangle
    void calculateArea()
    {
        area = length * width;
    }



    // Function to print the area of the rectangle
    void printArea()
    {
        cout<<"Area of Rectangle: "<<area<<endl;
    }
};

// Main function
int main()
{
    Circle obj1; // Object for Circle class
    obj1.calculateArea(); // Calculate area of the circle

    cout<<"\nCircle Details"<<endl;
    obj1.printColour(); // Print colour of the circle
    obj1.printArea(); // Print area of the circle

    Rectangle obj2; // Object for Rectangle class
    obj2.calculateArea(); // Calculate area of the rectangle

    cout<<"\nRectangle Details"<<endl;
    obj2.printColour(); // Print colour of the rectangle
    obj2.printArea(); // Print area of the rectangle

    return 0;
}
```

Output

```
Enter Circle Details
Enter Shape's Colour: Red
Enter Radius of Circle: 5

Circle Details
Colour: Red
Area of Circle: 78.5

Enter Rectangle Details
Enter Shape's Colour: Green
Enter Length of Rectangle: 4
Enter Width of Rectangle: 5

Rectangle Details
Colour: Green
Area of Rectangle: 20
```

Program 6

```
// File: Program6.cpp
// Date: 05-22-2024
// Name: Muhammad Tayyab Imran
// Registration No: 2023-BS-AI-2023
// 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 func□ons to handle the
specific a□ributes and behaviors of each type of employee.
// Employee: Should have data members for name, employee ID, and department. It
should also have member func□ons to get and set these values.
// Salaried Employee: Inherits from Employee and adds a data member for annual
Salary. It should have member func□ons 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 func□ons to get and set these values,
and to calculate the total pay based on sales and commission rate.

#include<iostream>
using namespace std;
```

```cpp
// Base class for Employee
class Employee
{
protected:
    string name; // Name of the employee
    int id; // Employee ID
    string department; // Department of the employee

public:
    // Function to input employee information
    void getInfo()
    {
        cout<<"Enter Employee Name: ";
        cin>>name;
        cout<<"Enter Employee Id: ";
        cin>>id;
        cout<<"Enter Employee Department: ";
        cin>>department;
    }

    // Function to display employee information
    void setInfo()
    {
        cout<<"Employee Name: "<<name<<endl;
        cout<<"Employee Id: "<<id<<endl;
        cout<<"Employee Department: "<<department<<endl;
    }
};



// Derived class for Salaried Employee inheriting from Employee
class SalariedEmployee : public Employee
{
private:
    double annualSalary; // Annual salary of the employee
    double monthlySalary; // Monthly salary of the employee

public:
    // Function to input salaried employee information
    void getInfo()
    {
        Employee::getInfo(); // Input base class information
        cout<<"Enter Annual Salary: ";
```

```
        cin>>annualSalary;
    }

    // Function to calculate monthly salary
    void calculate()
    {
        monthlySalary = annualSalary / 12;
    }

    // Function to display salaried employee information
    void setInfo()
    {
        Employee::setInfo(); // Display base class information
        cout<<"Annual Salary: "<<annualSalary<<endl;
        cout<<"Monthly Salary: "<<monthlySalary<<endl;
    }
};

// Derived class for Commission Employee inheriting from Employee
class CommissionEmployee : public Employee
{
private:
    double sales; // Sales made by the employee
    double commissionRate; // Commission rate of the employee
    double totalSalary; // Total salary of the employee

public:
    // Function to input commission employee information
    void getInfo()
    {
        Employee::getInfo(); // Input base class information
        cout<<"Enter Sales: ";
        cin>>sales;
        cout<<"Enter Commission Rate: ";
        cin>>commissionRate;
    }

    // Function to calculate total salary
    void calculate()
    {
        totalSalary = sales * commissionRate;
    }

    // Function to display commission employee information
    void setInfo()
```

```
    {
        Employee::setInfo(); // Display base class information
        cout<<"Total Salary: "<<totalSalary<<endl;
    }
};

// Main function
int main()
{
    SalariedEmployee obj1; // Object for SalariedEmployee class
    obj1.getInfo(); // Inputting salaried employee details
    obj1.calculate(); // Calculating monthly salary
    cout<<"\nSalaried Employee Info"<<endl;
    obj1.setInfo(); // Displaying salaried employee details

    CommissionEmployee obj2; // Object for CommissionEmployee class
    obj2.getInfo(); // Inputting commission employee details
    obj2.calculate(); // Calculating total salary
    cout<<"\nCommission Employee Info"<<endl;
    obj2.setInfo(); // Displaying commission employee details

    return 0;
}
```

Output

```
Enter Salaried Employee Info
Enter Employee Name: Tayyab
Enter Employee Id: 19
Enter Employee Department: Coding
Enter Annual Salary: 100000

Salaried Employee Info
Employee Name: Tayyab
Employee Id: 19
Employee Department: Coding
Annual Salary: 100000
Monthly Salary: 8333.33

Enter Commission Employee Info
Enter Employee Name: Haseeb
Enter Employee Id: 33
Enter Employee Department: Coding
Enter Sales: 15
Enter Commission Rate: 10000

Commission Employee Info
Employee Name: Haseeb
Employee Id: 33
Employee Department: Coding
Total Salary: 150000
```