NAME: M.HASHIR AFZAAL

2023-BS-AI-062

OBJECT ORIENTED PROGRAMMING

ASSIGNMENT#3

// File: P1

// Date: 22-MAY-2023

// Name: M.HASHIR AFZAAL

// Registration No: 062

/*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>
#include <string>
using namespace std;

class publication {
  protected:
    string title;
    float price;
  public:
    void get() {
      cout << "Enter title of a publication: ";
      cin >> title;
```

```
        cout << "Enter price of the publication: ";
        cin >> price;
    }
    void put() {
        cout << "Title: " << title << endl;
        cout << "Price: " << price << endl;
    }
};

class book : virtual protected publication {
  private:
    int pagecount;
  public:
    void getd() {
        publication::get();
        cout << "Enter page count: ";
        cin >> pagecount;
    }
    void putd() {
        publication::put();
        cout << "Page count: " << pagecount << endl;
    }
};

class tape : virtual protected publication {
  private:
    float playtime;
  public:
    void getdata() {
```

```
        publication::get();
        cout << "Enter playing time in minutes: ";
        cin >> playtime;
    }
    void putdata() {
        publication::put();
        cout << "Playing time: " << playtime << " minutes" << endl;
    }
};

int main() {
    book b;
    tape t;

    cout << "Enter data for book\n";
    b.getd();

    cout << "Enter data for tape\n";
    t.getdata();

    cout << "\nDisplaying data of book\n";
    b.putd();

    cout << "\nDisplaying data of tape\n";
    t.putdata();

    return 0;
}
```

```
Enter data for book
Enter title of a publication: HASHIR
Enter price of the publication: 9877
Enter page count: 76
Enter data for tape
Enter title of a publication: HASHIR
Enter price of the publication: 7655
Enter playing time in minutes: 43

Displaying data of book
Title: HASHIR
Price: 9877
Page count: 76

Displaying data of tape
Title: HASHIR
Price: 7655
Playing time: 43 minutes
```

// File: P2

// Date: 22-MAY-2023

// Name: M.HASHIR AFZAAL

// Registration No: 062

/*Start with the publica on, book, and tape classes of Ques on 1. Add a base class sales that holds an array of three floats so that it can record the dollar sales of a par cular publica on for the last three months. Include a getdata() func on 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 publica on 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 capabili es. */

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

class publication {
   protected:
      string title;
      float price;
   public:
```

```
    void get() {
        cout << "Enter title of a publication: ";
        cin >> title;
        cout << "Enter price of the publication: ";
        cin >> price;
    }
    void put() {
        cout << "Title: " << title << endl;
        cout << "Price: " << price << endl;
    }
};
class sales {
  protected:
    float sales_data[3];
  public:
    void getdata() {
        for (int i = 0; i < 3; ++i) {
            cout << "Enter sales for month :";
            cin >> sales_data[i];
        }
    }
    void putdata() {
        for (int i = 0; i < 3; ++i) {
            cout << "Sales for month " << sales_data[i] << endl;
        }
    }
};

class book : virtual protected publication,virtual protected sales {
```

```
    private:
        int pagecount;
    public:
        void getd() {
            publication::get();
            sales::getdata();
            cout << "Enter page count: ";
            cin >> pagecount;
        }
        void putd() {
            publication::put();
            sales::putdata();
            cout << "Page count: " << pagecount << endl;
        }
};

class tape : virtual protected publication,virtual protected sales {
    private:
        float playtime;
    public:
        void getdata() {
            publication::get();
            sales::getdata();
            cout << "Enter playing time in minutes: ";
            cin >> playtime;
        }
        void putdata() {
            publication::put();
            sales::putdata();
```

```
        cout << "Playing time: " << playtime << " minutes" << endl;
    }
};


int main() {
    book b;
    tape t;

    cout << "Enter data for book\n";
    b.getd();

    cout << "Enter data for tape\n";
    t.getdata();

    cout << "\nDisplaying data of book\n";
    b.putd();

    cout << "\nDisplaying data of tape\n";
    t.putdata();

    return 0;
}
```

// File: P3

// Date: 22-MAY-2023

// Name: M.HASHIR AFZAAL

// Registration No: 062

/*Assume that the publisher in Ques on 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 publica on. 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>
#include <string>
using namespace std;

class publication {
   protected:
      string title;
      float price;
   public:
```

```
    void get() {
        cout << "Enter title of a publication: ";
        cin >> title;
        cout << "Enter price of the publication: ";
        cin >> price;
    }
    void put() {
        cout << "Title: " << title << endl;
        cout << "Price: " << price << endl;
    }
};
class sales {
  protected:
    float sales_data[3];
  public:
    void getdata() {
        for (int i = 0; i < 3; ++i) {
            cout << "Enter sales for month :";
            cin >> sales_data[i];
        }
    }
    void putdata() {
        for (int i = 0; i < 3; ++i) {
            cout << "Sales for month " << sales_data[i] << endl;
        }
    }
};

class book : virtual protected publication,virtual protected sales {
```

```
    private:
        int pagecount;
    public:
        void getd() {
            publication::get();
            sales::getdata();
            cout << "Enter page count: ";
            cin >> pagecount;
        }
        void putd() {
            publication::put();
            sales::putdata();
            cout << "Page count: " << pagecount << endl;
        }
};

class tape : virtual protected publication,virtual protected sales {
    private:
        float playtime;
    public:
        void getdata() {
            publication::get();
            sales::getdata();
            cout << "Enter playing time in minutes: ";
            cin >> playtime;
        }
        void putdata() {
            publication::put();
            sales::putdata();
```

```
        cout << "Playing time: " << playtime << " minutes" << endl;
    }
};
class disk : virtual protected publication, virtual protected sales {
  private:
    enum Type { CD, DVD } disktype;
  public:
    void getdata() {
        publication::get();
        sales::getdata();
        char type;
        cout << "Enter disk type \n c for CD \n d for DVD: ";
        cin >> type;
        disktype = (type == 'c' || type == 'C') ? CD : DVD;
    }
    void putdata() {
        publication::put();
        sales::putdata();
        cout << "Disk type: " << (disktype == CD ? "CD" : "DVD") << endl;
    }
};

int main() {
  book b;
  tape t;
  disk d;

  cout << "Enter data for book\n";
  b.getd();
```

```
    cout << "Enter data for tape\n";
    t.getdata();

    cout << "Enter data for disk\n";
    d.getdata();

    cout << "Displaying data of book\n";
    b.putd();

    cout << "Displaying data of tape\n";
    t.putdata();

    cout << "Displaying data of disk\n";
    d.putdata();

    return 0;
}
```

```
Enter sales for month :77
Enter sales for month :88
Enter disk type
 c for CD
 d for DVD: d
Displaying data of book
Title: HASHIR
Price: 6570
Sales for month 656
Sales for month 786
Sales for month 64
Page count: 632
Displaying data of tape
Title: HASHIR
Price: 6474
Sales for month 765
Sales for month 765
Sales for month 765
Playing time: 76 minutes
Displaying data of disk
Title: HASHIR
Price: 564
Sales for month 66
Sales for month 77
Sales for month 88
Disk type: DVD
```

```
// File: P4
// Date: 22-MAY-2023
// Name: M.HASHIR AFZAAL
// Registration No: 062
```

/*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, scien st, and laborer classes and from

compensa on. This way none of the original classes needs to be modified*/

```
#include <iostream>
#include <string>

using namespace std;

enum Period { HOURLY, WEEKLY, MONTHLY };

class Employee {
protected:
   string name;
   unsigned long number;
public:
   void getdata() {
      cout << "Enter name: ";
      cin >> name;
      cout << "Enter number: ";
      cin >> number;
   }
```

```
    void putdata() const {
        cout << "Name: " << name << "\n";
        cout << "Number: " << number << "\n";
    }
};

class Employee2 : public Employee {
private:
    double compensation;
    Period period;
public:
    void getdata() {
        Employee::getdata();
        cout << "Enter compensation: ";
        cin >> compensation;
        int periodInput;
        cout << "Enter pay period (0 for Hourly, 1 for Weekly, 2 for Monthly): ";
        cin >> periodInput;
        period = static_cast<Period>(periodInput);
    }
    void putdata() const {
        Employee::putdata();
        cout << "Compensation: " << compensation << "\n";
        cout << "Pay period: ";
        switch (period) {
            case HOURLY: cout << "Hourly\n"; break;
            case WEEKLY: cout << "Weekly\n"; break;
            case MONTHLY: cout << "Monthly\n"; break;
        }
```

```
    }
};

class Manager2 : public Employee2 {
private:
    string title;
    double dues;
public:
    void getdata() {
        Employee2::getdata();
        cout << "Enter title: ";
        cin >> title;
        cout << "Enter dues: ";
        cin >> dues;
    }
    void putdata() const {
        Employee2::putdata();
        cout << "Title: " << title << "\n";
        cout << "Dues: " << dues << "\n";
    }
};

class Scientist2 : public Employee2 {
private:
    int publications;
public:
    void getdata() {
        Employee2::getdata();
        cout << "Enter number of publications: ";
```

```
        cin >> publications;
    }
    void putdata() const {
        Employee2::putdata();
        cout << "Publications: " << publications << "\n";
    }
};

class Laborer2 : public Employee2 {
    // No additional data members
};

int main() {
    Manager2 mgr;
    Scientist2 sci;
    Laborer2 lab;

    cout << "Enter manager data:\n";
    mgr.getdata();
    cout << "\nEnter scientist data:\n";
    sci.getdata();
    cout << "\nEnter laborer data:\n";
    lab.getdata();

    cout << "\nManager data:\n";
    mgr.putdata();
    cout << "\nScientist data:\n";
    sci.putdata();
    cout << "\nLaborer data:\n";
```

```
    lab.putdata();


    return 0;
}
```

```
Enter number of publications: 67

Enter laborer data:
Enter name: HAS
Enter number: 5737632713526
Enter compensation: Enter pay period (0 for Hourly, 1 for Weekly, 2 for Monthly):
Manager data:
Name: HASHIR
Number: 324445
Compensation: 67
Pay period: Monthly
Title: HASHIR
Dues: 454

Scientist data:
Name: HASH
Number: 666544321
Compensation: 6744
Pay period: Hourly
Publications: 67

Laborer data:
Name: HAS
Number: 4294967295
Compensation: 6.86682e-317
Pay period: Hourly
```

// File: P5

// Date: 22-MAY-2023

// Name: M.HASHIR AFZAAL

// Registration No: 062

/*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:

A ributes: color (type std::string).

Methods: A constructor to ini alize the color and a method printColor to display the color.

Circle Class:

A ributes: 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:

A ributes: 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>
#include <string>
#include <cmath>
using namespace std;

class shape {
   protected:
      string color;
   public:
      void getdata(){
         cout << "Enter color: ";
         cin >> color;
      }
      void putdata(){
         cout << "Color: " << color << endl;
      }
};

class circle : virtual protected shape {
   private:
      double radius;
      double area;
   public:
      circle() : radius(0), area(0) {}  // Default constructor
      void getdata(){
         shape::getdata();
```

```
        cout << "Enter radius of the circle: ";
        cin >> radius;
    }
    void setdata(){
        shape::putdata();
        area = M_PI * radius * radius;
        cout << "Area of the circle: " << area << endl;
    }
};

class rectangle : virtual protected shape {
  private:
    double length;
    double width;
    double area;
  public:
    rectangle() : length(0), width(0), area(0) {}  // Default constructor
    void getdata(){
        shape::getdata();
        cout << "Enter length of the rectangle: ";
        cin >> length;
        cout << "Enter width of the rectangle: ";
        cin >> width;
    }
    void setdata(){
        shape::putdata();
        area = length * width;
        cout << "Area of the rectangle: " << area << endl;
    }
```

```
};

int main() {
   char c;
   cout << "Press 'c' to calculate area of a circle || Press 'r' to calculate area of a rectangle: ";
   cin >> c;

   if (c == 'c') {
      circle circ;
      circ.getdata();
      circ.setdata();
   } else if (c == 'r') {
      rectangle rect;
      rect.getdata();
      rect.setdata();
   } else {
      cout << "Invalid input." << endl;
   }

   return 0;
}
```

```
Press 'c' to calculate area of a circle || Press 'r' to calculate area of a rectangle:  r
Enter color: orange
Enter length of the rectangle: 45
Enter width of the rectangle: 54
Color: orange
Area of the rectangle: 2430
```

```
// File: P6
// Date: 22-MAY-2023
// Name: M.HASHIR AFZAAL
// Registration No: 062
```

/*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>
#include <string>
#include <cmath>

using namespace std;

class Employee {
protected:
   string name;
   int employeeId;
   string department;
};

class SalariedEmployee : virtual protected Employee {
private:
   double salary;
```

```
public:
   void get() {
      cout << "Enter Employee Name: ";
      cin >> name;
      cout << "Enter Employee ID: ";
      cin >> employeeId;
      cout << "Enter Department: ";
      cin >> department;
      cout << "Enter Annual Salary: ";
      cin >> salary;
   }

   void set() {
         cout << "----INFORMATION----";
      cout << "Employee Name: " << name << endl;
      cout << "Employee ID: " << employeeId << endl;
      cout << "Department: " << department << endl;
      double monthlySalary = salary / 12;
      cout << "Monthly Salary is: " << monthlySalary << endl;
   }
};

class CommissionEmp : virtual protected Employee {
private:
   double commissionRate;
   int sales;

public:
```

```
    void getdata() {
        cout << "Enter Employee Name: ";
        cin >> name;
        cout << "Enter Employee ID: ";
        cin >> employeeId;
        cout << "Enter Department: ";
        cin >> department;
        cout << "Enter Commission Rate on Each Sale: ";
        cin >> commissionRate;
        cout << "Enter Number of Sales: ";
        cin >> sales;
    }

    void setdata() {
            cout << "----INFORMATION----";
        cout << "Name is: " << name << endl;
        cout << "ID is: " << employeeId << endl;
        cout << "Department is: " << department << endl;
        cout << "Commission Rate is: " << commissionRate << endl;
        cout << "Number of Sales are: " << sales << endl;
        double commission = commissionRate * sales;
        cout << "Commission of an Employee is: " << commission << endl;
    }
};

int main() {
    char choice;

    cout << "Press 's' to calculate Salary of an Employee" << endl;
```

```
    cout << "Press 's' to calculate Salary of an Employee" << endl;
    cout << "Press 'c' to calculate Commission of an Employee" << endl;
    cin >> choice;

    switch (choice) {
        case 's': {
            SalariedEmployee ss;
            ss.get();
            ss.set();
            break;
        }
        case 'c': {
            CommissionEmp cc;
            cc.getdata();
            cc.setdata();
            break;
        }
        default:
            cout << "Invalid Choice!" << endl;
    }

    return 0;
}
```

```
Press 's' to calculate Salary of an Employee
Press 'c' to calculate Commission of an Employee
c
Enter Employee Name: hashir
Enter Employee ID: 654245
Enter Department: ai
Enter Commission Rate on Each Sale: 543
Enter Number of Sales: 67
----INFORMATION----Name is: hashir
ID is: 654245
Department is: ai
Commission Rate is: 543
Number of Sales are: 67
Commission of an Employee is: 36381
```