QUES 1:

#include <iostream>

using namespace std;

class SavingsAccount {

string name;

int acc\_no;

float balance;

public:

void initialize(string n, int a, float b) {

name = n;

acc\_no = a;

balance = b;

}

void deposit(float amount) {

balance += amount;

cout << "Deposited successfully in Savings Account.\n";

}

void withdraw(float amount) {

if (amount > balance) {

cout << "Insufficient balance in Savings Account.\n";

} else {

balance -= amount;

cout << "Withdrawal successful from Savings Account.\n";

}

}

void computeInterest(float rate, int time) {

float interest = (balance \* rate \* time) / 100;

balance += interest;

cout << "Simple Interest of Rs. " << interest << " added to Savings Account.\n";

}

void showBalance() {

cout << "Savings Account Balance: Rs. " << balance << endl;

}

};

class CurrentAccount {

string name;

int acc\_no;

float balance;

float min\_balance;

float penalty;

public:

void initialize(string n, int a, float b, float min\_b, float pen) {

name = n;

acc\_no = a;

balance = b;

min\_balance = min\_b;

penalty = pen;

}

void deposit(float amount) {

balance += amount;

cout << "Deposited successfully in Current Account.\n";

}

void withdraw(float amount) {

if (amount > balance) {

cout << "Insufficient balance in Current Account.\n";

} else {

balance -= amount;

cout << "Withdrawal successful from Current Account.\n";

}

}

void checkBalance() {

if (balance < min\_balance) {

balance -= penalty;

cout << "Penalty of Rs. " << penalty << " imposed for low balance in Current Account.\n";

} else {

cout << "Minimum balance maintained in Current Account.\n";

}

}

void showBalance() {

cout << "Current Account Balance: Rs. " << balance << endl;

}

};

int main() {

SavingsAccount sav;

sav.initialize("Garima", 1111, 5000);

sav.deposit(1000);

sav.computeInterest(5.0, 2); // 5% simple interest for 2 years

sav.withdraw(1500);

sav.showBalance();

CurrentAccount cur;

cur.initialize("Rohit", 2222, 4000, 5000, 500); // min = 5000, penalty = 500

cur.deposit(2000);

cur.withdraw(2500);

cur.checkBalance();

cur.showBalance();

return 0;

}

QUES 2:

#include <iostream>

using namespace std;

class SavingsAccount {

string name;

int acc\_no;

float balance;

public:

SavingsAccount(string n, int a, float b) {

name = n;

acc\_no = a;

balance = b;

}

void deposit(float amount) {

balance += amount;

cout << "Deposited successfully in Savings Account.\n";

}

void withdraw(float amount) {

if (amount > balance) {

cout << "Insufficient balance in Savings Account.\n";

} else {

balance -= amount;

cout << "Withdrawal successful from Savings Account.\n";

}

}

void computeInterest(float rate, int time) {

float interest = (balance \* rate \* time) / 100;

balance += interest;

cout << "Simple Interest of Rs. " << interest << " added to Savings Account.\n";

}

void showBalance() {

cout << "Savings Account Balance: Rs. " << balance << endl;

}

};

class CurrentAccount {

string name;

int acc\_no;

float balance;

float min\_balance;

float penalty;

public:

CurrentAccount(string n, int a, float b, float min\_b, float pen) {

name = n;

acc\_no = a;

balance = b;

min\_balance = min\_b;

penalty = pen;

}

void deposit(float amount) {

balance += amount;

cout << "Deposited successfully in Current Account.\n";

}

void withdraw(float amount) {

if (amount > balance) {

cout << "Insufficient balance in Current Account.\n";

} else {

balance -= amount;

cout << "Withdrawal successful from Current Account.\n";

}

}

void checkBalance() {

if (balance < min\_balance) {

balance -= penalty;

cout << "Penalty of Rs. " << penalty << " imposed for low balance in Current Account.\n";

} else {

cout << "Minimum balance maintained in Current Account.\n";

}

}

void showBalance() {

cout << "Current Account Balance: Rs. " << balance << endl;

}

};

int main() {

SavingsAccount sav("Garima", 1111, 5000);

sav.deposit(1000);

sav.computeInterest(5.0, 2); // 5% simple interest for 2 years

sav.withdraw(1500);

sav.showBalance();

CurrentAccount cur("Rohit", 2222, 4000, 5000, 500); // min = 5000, penalty = 500

cur.deposit(2000);

cur.withdraw(2500);

cur.checkBalance();

cur.showBalance();

return 0;

}

QUES 3:

#include <iostream>  
using namespace std;  
  
class Staff {  
protected:  
    int code;  
    string name;  
public:  
    void getData() {  
        cout << "Enter code: ";  
        cin >> code;  
        cout << "Enter name: ";  
        cin >> name;  
    }  
  
    void displayData() {  
        cout << "Code: " << code << endl;  
        cout << "Name: " << name << endl;  
    }  
};  
  
class Teacher : public Staff {  
protected:  
    string subject, publication;  
public:  
    void input() {  
        getData();  
        cout << "Enter subject: ";  
        cin >> subject;  
        cout << "Enter publication: ";  
        cin >> publication;  
    }  
  
    void display() {  
        displayData();  
        cout << "Subject: " << subject << endl;  
        cout << "Publication: " << publication << endl;  
    }  
};  
  
class Officer : public Staff {  
protected:  
    char grade;  
public:  
    void input() {  
        getData();  
        cout << "Enter grade: ";  
        cin >> grade;  
    }  
  
    void display() {  
        displayData();  
        cout << "Grade: " << grade << endl;  
    }  
};  
  
class Typist : public Staff {  
protected:  
    float speed;  
public:  
    void input() {  
        getData();  
        cout << "Enter typing speed (wpm): ";  
        cin >> speed;  
    }  
  
    void display() {  
        displayData();  
        cout << "Typing Speed: " << speed << " wpm" << endl;  
    }  
};  
  
class Regular : public Typist {  
public:  
    void input() {  
        Typist::input();  
    }  
  
    void display() {  
        Typist::display();  
    }  
};  
  
class Casual : public Typist {  
protected:  
    float dailyWages;  
public:  
    void input() {  
        Typist::input();  
        cout << "Enter daily wages: ";  
        cin >> dailyWages;  
    }  
  
    void display() {  
        Typist::display();  
        cout << "Daily Wages: " << dailyWages << endl;  
    }  
};  
  
int main() {  
    Teacher t;  
    Officer o;  
    Regular r;  
    Casual c;  
  
    cout << "\nEnter Teacher details:\n";  
    t.input();  
  
    cout << "\nEnter Officer details:\n";  
    o.input();  
  
    cout << "\nEnter Regular Typist details:\n";  
    r.input();  
  
    cout << "\nEnter Casual Typist details:\n";  
    c.input();  
  
    cout << "\n\n--- Displaying Information ---\n";  
  
    cout << "\nTeacher:\n";  
    t.display();  
  
    cout << "\nOfficer:\n";  
    o.display();  
  
    cout << "\nRegular Typist:\n";  
    r.display();  
  
    cout << "\nCasual Typist:\n";  
    c.display();  
  
    return 0;  
}

QUES 4:

#include <iostream>

#include <string>

using namespace std;

class Person {

protected:

string name;

int age;

public:

void getPersonInfo() {

cout << "Enter your name: ";

cin >> name;

cout << "Enter your age: ";

cin >> age;

}

void showPersonInfo() {

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

cout << "Age: " << age << endl;

}

};

class Account : public Person {

protected:

string accountNumber;

float balance;

public:

void getAccountDetails() {

cout << "Enter account number: ";

cin >> accountNumber;

cout << "Enter account balance: ";

cin >> balance;

}

void showAccountDetails() {

cout << "Account Number: " << accountNumber << endl;

cout << "Balance: " << balance << endl;

}

};

class Admin : public Person {

protected:

string adminCode;

public:

void getAdminDetails() {

cout << "Enter admin code: ";

cin >> adminCode;

}

void showAdminDetails() {

cout << "Admin Code: " << adminCode << endl;

}

};

class Master : public Account, public Admin {

public:

void getAllDetails() {

cout << "\n--- Enter Details ---\n";

getPersonInfo();

getAccountDetails();

getAdminDetails();

}

void showAllDetails() {

cout << "\n--- Showing All Details ---\n";

showPersonInfo();

showAccountDetails();

showAdminDetails();

}

};

int main() {

Master obj;

obj.getAllDetails();

obj.showAllDetails();

return 0;

}

QUES 5:

#include <iostream>  
#include <string>  
using namespace std;  
  
class Book {  
private:  
    string title;  
    string author;  
    float price;  
    string publisher;  
    int stock;  
  
public:  
    void setData(string t, string a, float p, string pub, int s) {  
        title = t;  
        author = a;  
        price = p;  
        publisher = pub;  
        stock = s;  
    }  
  
    void displayBookDetails() {  
        cout << "Title: " << title << endl;  
        cout << "Author: " << author << endl;  
        cout << "Publisher: " << publisher << endl;  
        cout << "Price: " << price << endl;  
        cout << "Stock: " << stock << endl;  
    }  
  
    bool matches(string t, string a) {  
        return (title == t && author == a);  
    }  
  
    bool isAvailable(int requestedCopies) {  
        return stock >= requestedCopies;  
    }  
  
    float getPrice() {  
        return price;  
    }  
  
    void reduceStock(int requestedCopies) {  
        stock -= requestedCopies;  
    }  
};  
  
int main() {  
    const int numBooks = 3;  
    Book books[numBooks];  
  
   
    books[0].setData("C++", "Stroustrup", 400.0, "Addison", 10);  
    books[1].setData("1984", "Orwell", 250.0, "Secker", 5);  
    books[2].setData("Mockingbird", "Lee", 300.0, "J.B.Lippincott", 7);  
  
    string searchTitle, searchAuthor;  
    cout << "Enter book title: ";  
    cin >> searchTitle;  
    cout << "Enter book author: ";  
    cin >> searchAuthor;  
  
    bool found = false;  
  
    for (int i = 0; i < numBooks; i++) {  
        if (books[i].matches(searchTitle, searchAuthor)) {  
            found = true;  
            cout << "\nBook found:\n";  
            books[i].displayBookDetails();  
  
            int copies;  
            cout << "\nEnter number of copies required: ";  
            cin >> copies;  
  
            if (books[i].isAvailable(copies)) {  
                float total = books[i].getPrice() \* copies;  
                cout << "Total cost: " << total << endl;  
                books[i].reduceStock(copies);  
            } else {  
                cout << "Required copies not in stock." << endl;  
            }  
            break;  
        }  
    }  
  
    if (!found) {  
        cout << "Book not found in inventory." << endl;  
    }  
  
    return 0;  
}

ques7

#include <iostream>  
using namespace std;  
  
class SimpleCircle {  
private:  
    int itsRadius;  
  
public:  
   
    SimpleCircle() {  
        itsRadius = 5;  
    }  
  
     
    SimpleCircle(int radius) {  
        itsRadius = radius;  
    }  
  
  
    SimpleCircle(const SimpleCircle& other) {  
        itsRadius = other.itsRadius;  
    }  
  
     
    ~SimpleCircle() {  
     
    }  
  
     
    int GetRadius() {  
        return itsRadius;  
    }  
  
  
    void SetRadius(int radius) {  
        itsRadius = radius;  
    }  
  
     
    void operator=(SimpleCircle other) {  
        itsRadius = other.itsRadius;  
    }  
  
     
    SimpleCircle operator++(int) {  
        SimpleCircle temp(itsRadius);    
        itsRadius = itsRadius + 1;        
        return temp;                
    }  
};  
  
int main() {  
    SimpleCircle circle1;          
    SimpleCircle circle2(9);    
  
    circle1 = circle1++;          
    circle2 = circle2++;            
  
    cout << "Circle1 Radius: " << circle1.GetRadius() << endl;  
    cout << "Circle2 Radius: " << circle2.GetRadius() << endl;  
  
    circle1 = circle2;            
  
    cout << "After assignment:" << endl;  
    cout << "Circle1 Radius: " << circle1.GetRadius() << endl;  
    cout << "Circle2 Radius: " << circle2.GetRadius() << endl;  
  
    return 0;  
}

QUES 6:

#include <iostream>

using namespace std;

class Employee {

private:

int age;

int yearsOfService;

double Salary;

public:

Employee(int a, int y, double s) {

age = a;

yearsOfService = y;

Salary = s;

}

void setAge(int a) {

age = a;

}

void setYearsOfService(int y) {

yearsOfService = y;

}

void setSalary(double s) {

Salary = s;

}

int getAge() {

return age;

}

int getYearsOfService() {

return yearsOfService;

}

double getSalary() {

return Salary;

}

void reportSalaryInThousands() {

int salaryInThousands = (int)(Salary + 500) / 1000; // Manually round to the nearest thousand

cout << "Salary in thousands: " << salaryInThousands << "K" << endl;

}

};

int main() {

// Creating two Employee objects with initialized values

Employee employee1(25, 3, 55000); // Employee 1: age 25, 3 years of service, salary 55000

Employee employee2(40, 15, 80000); // Employee 2: age 40, 15 years of service, salary 80000

cout << "Employee 1 - Age: " << employee1.getAge() << ", Years of Service: " << employee1.getYearsOfService() << ", Salary: " << employee1.getSalary() << endl;

employee1.reportSalaryInThousands();

cout << "Employee 2 - Age: " << employee2.getAge() << ", Years of Service: " << employee2.getYearsOfService() << ", Salary: " << employee2.getSalary() << endl;

employee2.reportSalaryInThousands();

return 0;

}

QUES 8:

#include <iostream>

using namespace std;

class Cinema {

private:

int seats\_1pm;

int seats\_5pm;

int seats\_830pm;

const int total\_seats = 100;

public:

Cinema() {

seats\_1pm = total\_seats;

seats\_5pm = total\_seats;

seats\_830pm = total\_seats;

}

void book\_seat(int performance, int num\_seats) {

if (num\_seats <= 0) {

cout << "Invalid number of seats!" << endl;

return;

}

switch (performance) {

case 1:

if (seats\_1pm >= num\_seats) {

seats\_1pm -= num\_seats;

cout << "Successfully booked " << num\_seats << " seats for 1pm show!" << endl;

} else {

cout << "Not enough seats available for the 1pm show!" << endl;

}

break;

case 2:

if (seats\_5pm >= num\_seats) {

seats\_5pm -= num\_seats;

cout << "Successfully booked " << num\_seats << " seats for 5pm show!" << endl;

} else {

cout << "Not enough seats available for the 5pm show!" << endl;

}

break;

case 3:

if (seats\_830pm >= num\_seats) {

seats\_830pm -= num\_seats;

cout << "Successfully booked " << num\_seats << " seats for 8:30pm show!" << endl;

} else {

cout << "Not enough seats available for the 8:30pm show!" << endl;

}

break;

default:

cout << "Invalid performance time!" << endl;

break;

}

}

void display\_remaining\_seats(int performance) {

switch (performance) {

case 1:

cout << "Remaining seats for the 1pm show: " << seats\_1pm << endl;

break;

case 2:

cout << "Remaining seats for the 5pm show: " << seats\_5pm << endl;

break;

case 3:

cout << "Remaining seats for the 8:30pm show: " << seats\_830pm << endl;

break;

default:

cout << "Invalid performance time!" << endl;

break;

}

}

};

int main() {

Cinema cinema;

int choice;

int performance, num\_seats;

do {

cout << "\nCinema Booking System\n";

cout << "1. Book Seats\n";

cout << "2. View Remaining Seats\n";

cout << "3. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1:

cout << "Enter performance (1 for 1pm, 2 for 5pm, 3 for 8:30pm): ";

cin >> performance;

cout << "Enter number of seats to book: ";

cin >> num\_seats;

cinema.book\_seat(performance, num\_seats);

break;

case 2:

cout << "Enter performance to check remaining seats (1 for 1pm, 2 for 5pm, 3 for 8:30pm): ";

cin >> performance;

cinema.display\_remaining\_seats(performance);

break;

case 3:

cout << "Exiting the system. Thank you!" << endl;

break;

default:

cout << "Invalid choice! Please try again." << endl;

break;

}

} while (choice != 3);

return 0;

}

QUES 9:

#include <iostream>  
#include <vector>  
#include <string>  
using namespace std;  
  
class Book {  
private:  
    int classMark;  
    string status;  
  
public:  
    Book(int mark) {  
        classMark = mark;  
        status = "on\_shelf";  
    }  
  
    void loan() {  
        if (status == "on\_shelf") {  
            status = "on\_loan";  
        }  
    }  
  
    void reserve() {  
        if (status == "on\_loan") {  
            status = "reserved";  
        }  
    }  
  
    void missing() {  
        status = "missing";  
    }  
  
    void returned() {  
        status = "on\_shelf";  
    }  
  
    string getStatus() {  
        return status;  
    }  
};  
  
void summarizeLibrary(vector<Book> books) {  
    int total = books.size();  
    int onLoan = 0;  
    int reserved = 0;  
    int onShelf = 0;  
    int missing = 0;  
  
    for (int i = 0; i < total; i++) {  
        string status = books[i].getStatus();  
  
        if (status == "on\_loan") {  
            onLoan++;  
        } else if (status == "reserved") {  
            reserved++;  
        } else if (status == "on\_shelf") {  
            onShelf++;  
        } else if (status == "missing") {  
            missing++;  
        }  
    }  
  
    cout << "Books in library = " << total << endl;  
    cout << "Books on loan = " << onLoan << endl;  
    cout << "Books reserved = " << reserved << endl;  
    cout << "Books on shelves = " << onShelf << endl;  
    cout << "Books missing = " << missing << endl;  
}  
  
int main() {  
    vector<Book> library;  
  
    Book b1(101);  
    Book b2(102);  
    Book b3(103);  
    Book b4(104);  
    Book b5(105);  
  
    b1.loan();  
    b2.loan();  
    b2.reserve();  
    b3.missing();  
  
    library.push\_back(b1);  
    library.push\_back(b2);  
    library.push\_back(b3);  
    library.push\_back(b4);  
    library.push\_back(b5);  
  
    summarizeLibrary(library);  
  
    return 0;  
}

QUES10:

#include <iostream>

#include <cmath>

using namespace std;

class Employee {

private:

int age;

int yearsOfService;

double Salary;

public:

Employee(int a, int y, double s) {

age = a;

yearsOfService = y;

Salary = s;

}

void setAge(int a) {

age = a;

}

void setYearsOfService(int y) {

yearsOfService = y;

}

void setSalary(double s) {

Salary = s;

}

int getAge() {

return age;

}

int getYearsOfService() {

return yearsOfService;

}

double getSalary() {

return Salary;

}

void reportSalaryInThousands() {

int salaryInThousands = round(Salary / 1000);

cout << "Salary in thousands: " << salaryInThousands << "K" << endl;

}

};

int main() {

Employee employee1(25, 3, 55000);

Employee employee2(40, 15, 80000);

cout << "Employee 1 - Age: " << employee1.getAge() << ", Years of Service: " << employee1.getYearsOfService() << ", Salary: " << employee1.getSalary() << endl;

employee1.reportSalaryInThousands();

cout << "Employee 2 - Age: " << employee2.getAge() << ", Years of Service: " << employee2.getYearsOfService() << ", Salary: " << employee2.getSalary() << endl;

employee2.reportSalaryInThousands();

return 0;

}

Ques 11:

#include <iostream>

using namespace std;

class Mammals {

public:

void display() {

cout << "I am mammal" << endl;

}

};

class MarineAnimals {

public:

void display() {

cout << "I am a marine animal" << endl;

}

};

class BlueWhale : public Mammals, public MarineAnimals {

public:

void display() {

cout << "I belong to both the categories: Mammals as well as Marine Animals" << endl;

}

};

int main() {

Mammals mammal;

cout << "Mammal object: ";

mammal.display();

MarineAnimals marineAnimal;

cout << "MarineAnimal object: ";

marineAnimal.display();

BlueWhale blueWhale;

cout << "BlueWhale object: ";

blueWhale.display();

cout << "Accessing parent Mammals function from BlueWhale object: ";

blueWhale.Mammals::display();

cout << "Accessing parent MarineAnimals function from BlueWhale object: ";

blueWhale.MarineAnimals::display();

return 0;

}

QUES 12:

#include <iostream>

using namespace std;

class Fruit {

protected:

    int totalFruits;

public:

    Fruit() : totalFruits(0) {}

    void addToBasket(int count) {

        totalFruits += count;

    }

    int getTotalFruits() const {

        return totalFruits;

    }

};

class Apples : public Fruit {

private:

    int appleCount;

public:

    Apples(int count) : appleCount(count) {

        addToBasket(count);

    }

    int getAppleCount() const {

        return appleCount;

    }

};

class Mangoes : public Fruit {

private:

    int mangoCount;

public:

    Mangoes(int count) : mangoCount(count) {

        addToBasket(count);

    }

    int getMangoCount() const {

        return mangoCount;

    }

};

int main() {

    Apples apples(10);

    Mangoes mangoes(5);

    cout << "Number of apples: " << apples.getAppleCount() << endl;

    cout << "Number of mangoes: " << mangoes.getMangoCount() << endl;

    int total = apples.getAppleCount() + mangoes.getMangoCount();

    cout << "Total number of fruits: " << total << endl;

    return 0;

}

QUES 13:

#include <iostream>

using namespace std;

int rollCounter = 1;

class Marks {

protected:

int rollNumber;

string name;

public:

Marks() {

rollNumber = rollCounter++;

}

void inputName() {

cout << "Enter name: ";

cin >> name;

}

void displayBasicInfo() {

cout << "Roll No: " << rollNumber << ", Name: " << name;

}

};

class Physics : virtual public Marks {

protected:

float physicsMarks;

public:

void inputPhysics() {

cout << "Enter Physics marks: ";

cin >> physicsMarks;

}

};

class Chemistry : virtual public Marks {

protected:

float chemistryMarks;

public:

void inputChemistry() {

cout << "Enter Chemistry marks: ";

cin >> chemistryMarks;

}

};

class Mathematics : public Physics, public Chemistry {

protected:

float mathMarks;

float totalMarks;

public:

void inputAllMarks() {

inputName();

inputPhysics();

inputChemistry();

cout << "Enter Mathematics marks: ";

cin >> mathMarks;

totalMarks = physicsMarks + chemistryMarks + mathMarks;

}

void displayAllMarks() {

displayBasicInfo();

cout << ", Physics: " << physicsMarks

<< ", Chemistry: " << chemistryMarks

<< ", Math: " << mathMarks

<< ", Total: " << totalMarks << endl;

}

float getTotalMarks() {

return totalMarks;

}

};

int main() {

int num;

cout << "Enter number of students: ";

cin >> num;

Mathematics students[100]; // Max 100 students

float classTotal = 0;

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

cout << "\nEnter details for student " << (i + 1) << ":\n";

students[i].inputAllMarks();

classTotal += students[i].getTotalMarks();

}

cout << "\n--- Students Marks Summary ---\n";

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

students[i].displayAllMarks();

}

float average = classTotal / num;

cout << "\nAverage marks of the class: " << average << endl;

return 0;

}

QUES 14:

#include <iostream>

#include <string>

using namespace std;

class Vehicle {

protected:

    float mileage;

    float price;

public:

    void setVehicleData(float m, float p) {

        mileage = m;

        price = p;

    }

    void displayVehicleData() {

        cout << "Mileage: " << mileage << " km/l" << endl;

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

    }

};

class Car : public Vehicle {

protected:

    float ownershipCost;

    int warranty;

    int seatingCapacity;

    string fuelType;

public:

    void setCarData(float cost, int w, int seat, string fuel) {

        ownershipCost = cost;

        warranty = w;

        seatingCapacity = seat;

        fuelType = fuel;

    }

    void displayCarData() {

        cout << "Ownership Cost: " << ownershipCost << " INR" << endl;

        cout << "Warranty: " << warranty << " years" << endl;

        cout << "Seating Capacity: " << seatingCapacity << endl;

        cout << "Fuel Type: " << fuelType << endl;

    }

};

class Bike : public Vehicle {

protected:

    int cylinders;

    int gears;

    string coolingType;

    string wheelType;

    float fuelTankSize;

public:

    void setBikeData(int cyl, int gear, string cool, string wheel, float tank) {

        cylinders = cyl;

        gears = gear;

        coolingType = cool;

        wheelType = wheel;

        fuelTankSize = tank;

    }

    void displayBikeData() {

        cout << "Cylinders: " << cylinders << endl;

        cout << "Gears: " << gears << endl;

        cout << "Cooling Type: " << coolingType << endl;

        cout << "Wheel Type: " << wheelType << endl;

        cout << "Fuel Tank Size: " << fuelTankSize << " inches" << endl;

    }

};

class Audi : public Car {

private:

    string modelType;

public:

    void setAudiData(string model, float cost, int w, int seat, string fuel, float m, float p) {

        modelType = model;

        setCarData(cost, w, seat, fuel);

        setVehicleData(m, p);

    }

    void displayAudiData() {

        cout << "--- Audi Car ---" << endl;

        cout << "Model Type: " << modelType << endl;

        displayCarData();

        displayVehicleData();

    }

};

class Ford : public Car {

private:

    string modelType;

public:

    void setFordData(string model, float cost, int w, int seat, string fuel, float m, float p) {

        modelType = model;

        setCarData(cost, w, seat, fuel);

        setVehicleData(m, p);

    }

    void displayFordData() {

        cout << "--- Ford Car ---" << endl;

        cout << "Model Type: " << modelType << endl;

        displayCarData();

        displayVehicleData();

    }

};

class Bajaj : public Bike {

private:

    string makeType;

public:

    void setBajajData(string make, int cyl, int gear, string cool, string wheel, float tank, float m, float p) {

        makeType = make;

        setBikeData(cyl, gear, cool, wheel, tank);

        setVehicleData(m, p);

    }

    void displayBajajData() {

        cout << "--- Bajaj Bike ---" << endl;

        cout << "Make Type: " << makeType << endl;

        displayBikeData();

        displayVehicleData();

    }

};

class TVS : public Bike {

private:

    string makeType;

public:

    void setTVSData(string make, int cyl, int gear, string cool, string wheel, float tank, float m, float p) {

        makeType = make;

        setBikeData(cyl, gear, cool, wheel, tank);

        setVehicleData(m, p);

    }

    void displayTVSData() {

        cout << "--- TVS Bike ---" << endl;

        cout << "Make Type: " << makeType << endl;

        displayBikeData();

        displayVehicleData();

    }

};

int main() {

    Audi a;

    a.setAudiData("A4", 2000000, 4, 5, "Petrol", 18.5, 2500000);

    a.displayAudiData();

    cout << endl;

    Ford f;

    f.setFordData("Endeavour", 1800000, 3, 7, "Diesel", 15.0, 2300000);

    f.displayFordData();

    cout << endl;

    Bajaj b;

    b.setBajajData("Pulsar", 2, 5, "Air", "Alloys", 13.0, 50.0, 110000);

    b.displayBajajData();

    cout << endl;

    TVS t;

    t.setTVSData("Apache", 2, 5, "Oil", "Spokes", 12.0, 45.0, 95000);

    t.displayTVSData();

    return 0;

}

QUES 15:

#include <iostream>

using namespace std;

class Shape {

public:

void display() {

cout << "This is a shape" << endl;

}

};

class Polygon : public Shape {

public:

void display() {

cout << "Polygon is a shape" << endl;

}

};

class Rectangle : public Polygon {

public:

void display() {

cout << "Rectangle is a polygon" << endl;

}

};

class Triangle : public Polygon {

public:

void display() {

cout << "Triangle is a polygon" << endl;

}

};

class Square : public Rectangle {

public:

void display() {

cout << "Square is a rectangle" << endl;

}

};

int main() {

Shape shapeObj;

Polygon polygonObj;

Rectangle rectangleObj;

Triangle triangleObj;

Square squareObj;

shapeObj.display();

polygonObj.display();

rectangleObj.display();

triangleObj.display();

squareObj.display();

return 0;

}

QUES 16:

#include <iostream>

#include <string>

using namespace std;

class RBI {

protected:

    float minInterestRate;

    float minBalance;

    float maxWithdrawalLimit;

public:

    RBI() {

        minInterestRate = 4.0;

        minBalance = 1000.0;

        maxWithdrawalLimit = 25000.0;

    }

    void displayGuidelines() {

        cout << "--- RBI Guidelines ---" << endl;

        cout << "Minimum Interest Rate: " << minInterestRate << "%" << endl;

        cout << "Minimum Balance Allowed: " << minBalance << " INR" << endl;

        cout << "Maximum Withdrawal Limit: " << maxWithdrawalLimit << " INR" << endl;

    }

};

class SBI : public RBI {

private:

    float sbiInterestRate;

    float sbiMinBalance;

    float sbiMaxWithdrawal;

public:

    SBI() {

        sbiInterestRate = 4.5;

        sbiMinBalance = 1200;

        sbiMaxWithdrawal = 20000;

    }

    void displaySBIInfo() {

        cout << "\n--- SBI Bank Info ---" << endl;

        cout << "Interest Rate: " << sbiInterestRate << "%" << endl;

        cout << "Minimum Balance: " << sbiMinBalance << " INR" << endl;

        cout << "Maximum Withdrawal Limit: " << sbiMaxWithdrawal << " INR" << endl;

    }

};

class HDFC : public RBI {

private:

    float hdfcInterestRate;

    float hdfcMinBalance;

    float hdfcMaxWithdrawal;

public:

    HDFC() {

        hdfcInterestRate = 5.0;

        hdfcMinBalance = 1500;

        hdfcMaxWithdrawal = 22000;

    }

    void displayHDFCInfo() {

        cout << "\n--- HDFC Bank Info ---" << endl;

        cout << "Interest Rate: " << hdfcInterestRate << "%" << endl;

        cout << "Minimum Balance: " << hdfcMinBalance << " INR" << endl;

        cout << "Maximum Withdrawal Limit: " << hdfcMaxWithdrawal << " INR" << endl;

    }

};

int main() {

    RBI rbi;

    SBI sbi;

    HDFC hdfc;

    rbi.displayGuidelines();

    sbi.displaySBIInfo();

    hdfc.displayHDFCInfo();

    return 0;

}

QUES 17:

#include <iostream>

#include <string>

using namespace std;

class RBI {

public:

virtual void displayInterest() {

cout << "Interest rate: Base rate" << endl;

}

};

class SBI : public RBI {

public:

void displayInterest() override {

cout << "SBI Interest rate: 4%" << endl;

}

};

class ICICI : public RBI {

public:

void displayInterest() override {

cout << "ICICI Interest rate: 4.5%" << endl;

}

};

class PNB : public RBI {

public:

void displayInterest() override {

cout << "PNB Interest rate: 3.5%" << endl;

}

};

class Customer {

private:

string name;

int accountNumber;

RBI \*bank;

public:

Customer(string customerName, int accNumber, RBI \*bankAccount) {

name = customerName;

accountNumber = accNumber;

bank = bankAccount;

}

void displayCustomerDetails() {

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

cout << "Account Number: " << accountNumber << endl;

bank->displayInterest();

}

};

int main() {

SBI sbiBank;

ICICI iciciBank;

PNB pnbBank;

Customer customer1("Garima Rai", 1001, &sbiBank);

Customer customer2("Amit Sharma", 1002, &iciciBank);

Customer customer3("Neha Gupta", 1003, &pnbBank);

cout << "Customer 1 Details:" << endl;

customer1.displayCustomerDetails();

cout << endl;

cout << "Customer 2 Details:" << endl;

customer2.displayCustomerDetails();

cout << endl;

cout << "Customer 3 Details:" << endl;

customer3.displayCustomerDetails();

return 0;

}

QUES 18:

#include <iostream>

#include <string>

using namespace std;

class Student {

private:

    string name;

public:

    Student() {

        name = "Unknown";

    }

    Student(string studentName) {

        name = studentName;

    }

    void displayName() {

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

    }

};

int main() {

    Student s1;

    Student s2("Amit");

    s1.displayName();

    s2.displayName();

    return 0;

}

QUES 19:

#include <iostream>

using namespace std;

class Rectangle {

private:

float length;

float breadth;

public:

Rectangle() {

length = 0;

breadth = 0;

}

Rectangle(float l, float b) {

length = l;

breadth = b;

}

Rectangle(float side) {

length = side;

breadth = side;

}

float area() {

return length \* breadth;

}

};

int main() {

Rectangle rect1;

Rectangle rect2(5, 4);

Rectangle rect3(3);

cout << "Area of rectangle 1 (no parameters): " << rect1.area() << endl;

cout << "Area of rectangle 2 (length = 5, breadth = 4): " << rect2.area() << endl;

cout << "Area of rectangle 3 (side = 3): " << rect3.area() << endl;

return 0;

}

QUES 20:

#include <iostream>

using namespace std;

class AddAmount {

private:

    int amount;

public:

    AddAmount() {

        amount = 50;

    }

    AddAmount(int add) {

        amount = 50 + add;

    }

    void displayAmount() {

        cout << "Final amount in Piggie Bank: $" << amount << endl;

    }

};

int main() {

    AddAmount a1;

    AddAmount a2(30);

    a1.displayAmount();

    a2.displayAmount();

    return 0;

}