Assignment 1

SOURCE CODE:

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

class Account {
protected:
    string customerName;
    int accountNumber;
    string accountType;
    float balance;

public:
    void initialize(string name, int accNo, string type, float bal) {
        customerName = name;
        accountNumber = accNo;
        accountType = type;
        balance = bal;
    }

    void deposit(float amount) {
        balance = balance + amount;
        cout << "Amount deposited: " << amount << endl;
    }

    void displayBalance() {
        cout << "Current balance: " << balance << endl;
    }

    float getBalance() {
        return balance;
    }

    void updateBalance(float newBalance) {
        balance = newBalance;
    }
};

// Savings Account
class Sav_acct : public Account {
public:
    void computeInterest(float rate, int time) {
        float interest = balance;
        for (int i = 0; i < time; i++) {
            interest = interest + (interest * rate / 100);
        }
        interest = interest - balance;
        cout << "Interest computed: " << interest << endl;
        deposit(interest);
    }

    void withdraw(float amount) {
        if (amount > balance) {
            cout << "Insufficient balance." << endl;
        } else {
            balance = balance - amount;
            cout << "Withdrawal successful. Withdrawn: " << amount << endl;
        }
    }
};

// Current Account
class Cur_acct : public Account {
public:
    void withdraw(float amount) {
        if (amount > balance) {
            cout << "Insufficient balance." << endl;
        } else {
            balance = balance - amount;
            cout << "Withdrawal successful. Withdrawn: " << amount << endl;
            checkMinBalance();
        }
    }

    void checkMinBalance() {
        float minBalance = 500.0;
        float serviceCharge = 50.0;
        if (balance < minBalance) {
            cout << "Balance below minimum. Service charge imposed: " <<
serviceCharge << endl;
            balance = balance - serviceCharge;
        }
    }
};

// Main Function
int main() {
    int choice;
    cout << "Choose Account Type:\n1. Savings Account\n2. Current
Account\nEnter your choice: ";
    cin >> choice;

    if (choice == 1) {
        Sav_acct sav;
        string name;
        int accNo;
        float initBal, depAmount, withdrawAmount, rate;
        int time;

        cout << "\nEnter customer name: ";
        cin >> name;
        cout << "Enter account number: ";
        cin >> accNo;
        cout << "Enter initial balance: ";
        cin >> initBal;
        sav.initialize(name, accNo, "Saving", initBal);

        cout << "Enter amount to deposit: ";
        cin >> depAmount;
        sav.deposit(depAmount);

        cout << "Enter interest rate (%): ";
        cin >> rate;
        cout << "Enter time (in years): ";
        cin >> time;
        sav.computeInterest(rate, time);

        cout << "Enter amount to withdraw: ";
        cin >> withdrawAmount;
        sav.withdraw(withdrawAmount);

        sav.displayBalance();
    }

    else if (choice == 2) {
        Cur_acct cur;
        string name;
        int accNo;
        float initBal, depAmount, withdrawAmount;
```

```
        cout << "\nEnter customer name: ";
        cin >> name;
        cout << "Enter account number: ";
        cin >> accNo;
        cout << "Enter initial balance: ";
        cin >> initBal;

        cur.initialize(name, accNo, "Current", initBal);

        cout << "Enter amount to deposit: ";
        cin >> depAmount;
        cur.deposit(depAmount);

        cout << "Enter amount to withdraw: ";
        cin >> withdrawAmount;
        cur.withdraw(withdrawAmount);

        cur.displayBalance();
    }

    else {
        cout << "Invalid choice!";
    }

    return 0;
}
```

OUTPUT

```
Choose Account Type:
1. Savings Account
2. Current Account
Enter your choice: 1

Enter customer name: Ram
Enter account number: 101
Enter initial balance: 1000
Enter amount to deposit: 500
Amount deposited: 500
Enter interest rate (%): 5
Enter time (in years): 2
Interest computed: 157.625
Amount deposited: 157.625
Enter amount to withdraw: 300
Withdrawal successful. Withdrawn: 300
Current balance: 1357.62
```

Assignment 2

SOURCE CODE:

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

class Account {
protected:
    string customerName;
    int accountNumber;
    string accountType;
    float balance;

public:
    // Constructor for Account class
    Account(string name, int accNo, string type, float bal) {
        customerName = name;
        accountNumber = accNo;
        accountType = type;
        balance = bal;
    }

    void deposit(float amount) {
        balance = balance + amount;
        cout << "Amount deposited: " << amount << endl;
    }

    void displayBalance() {
        cout << "Current balance: " << balance << endl;
    }

    float getBalance() {
        return balance;
    }

    void updateBalance(float newBalance) {
        balance = newBalance;
    }
};

// Savings Account with constructor
class Sav_acct : public Account {
public:
    // Constructor for Savings Account
    Sav_acct(string name, int accNo, float bal) : Account(name, accNo, "Saving",
bal) {}

    void computeInterest(float rate, int time) {
        float interest = balance;
        for (int i = 0; i < time; i++) {
            interest = interest + (interest * rate / 100);
        }
        interest = interest - balance;
        cout << "Interest computed: " << interest << endl;
        deposit(interest);
    }

    void withdraw(float amount) {
        if (amount > balance) {
            cout << "Insufficient balance." << endl;
        } else {
            balance = balance - amount;
            cout << "Withdrawal successful. Withdrawn: " << amount << endl;
        }
    }
};

// Current Account with constructor
class Cur_acct : public Account {
public:
    // Constructor for Current Account
    Cur_acct(string name, int accNo, float bal) : Account(name, accNo,
"Current", bal) {}

    void withdraw(float amount) {
        if (amount > balance) {
            cout << "Insufficient balance." << endl;
        } else {
            balance = balance - amount;
            cout << "Withdrawal successful. Withdrawn: " << amount << endl;
            checkMinBalance();
        }
    }

    void checkMinBalance() {
        float minBalance = 500.0;
        float serviceCharge = 50.0;
        if (balance < minBalance) {
            cout << "Balance below minimum. Service charge imposed: " <<
serviceCharge << endl;
            balance = balance - serviceCharge;
        }
    }
};

// Main Function
int main() {
    int choice;
    cout << "Choose Account Type:\n1. Savings Account\n2. Current
Account\nEnter your choice: ";
    cin >> choice;

    if (choice == 1) {
        string name;
        int accNo;
        float initBal, depAmount, withdrawAmount, rate;
```

```
int time;

cout << "\nEnter customer name: ";
cin >> name;
cout << "Enter account number: ";
cin >> accNo;
cout << "Enter initial balance: ";
cin >> initBal;

Sav_acct sav(name, accNo, initBal);

cout << "Enter amount to deposit: ";
cin >> depAmount;
sav.deposit(depAmount);

cout << "Enter interest rate (%): ";
cin >> rate;
cout << "Enter time (in years): ";
cin >> time;
sav.computeInterest(rate, time);

cout << "Enter amount to withdraw: ";
cin >> withdrawAmount;
sav.withdraw(withdrawAmount);

sav.displayBalance();
```

```
}

else if (choice == 2) {
    string name;
    int accNo;
    float initBal, depAmount, withdrawAmount;

    cout << "\nEnter customer name: ";
    cin >> name;
    cout << "Enter account number: ";
    cin >> accNo;
    cout << "Enter initial balance: ";
    cin >> initBal;

    Cur_acct cur(name, accNo, initBal);

    cout << "Enter amount to deposit: ";
    cin >> depAmount;
    cur.deposit(depAmount);

    cout << "Enter amount to withdraw: ";
    cin >> withdrawAmount;
    cur.withdraw(withdrawAmount);

    cur.displayBalance();
}
```

```
    else {
        cout << "Invalid choice!";
    }

    return 0;
}
```

OUTPUT:

```
Choose Account Type:
1. Savings Account
2. Current Account
Enter your choice: 1

Enter customer name: Jay
Enter account number: 101
Enter initial balance: 1000
Enter amount to deposit: 500
Amount deposited: 500
Enter interest rate (%): 5
Enter time (in years): 2
Interest computed: 157.625
Amount deposited: 157.625
Enter amount to withdraw: 300
Withdrawal successful. Withdrawn: 300
Current balance: 1357.62
```

Assignment 3

SOURCE CODE:

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

// Base class
class Staff {
protected:
    int employeeId;
    string name;

public:
    Staff(int id, string n) {
        employeeId = id;
        name = n;
    }

    void displayStaff() {
        cout << "Employee ID: " << employeeId << endl;
        cout << "Name: " << name << endl;
    }
};

// Derived class: Teacher
class Teacher : public Staff {
```

```
private:
    string subject;
    string bookPublished;

public:
    Teacher(int id, string n, string sub, string book)
        : Staff(id, n) {
        subject = sub;
        bookPublished = book;
    }

    void display() {
        displayStaff();
        cout << "Subject: " << subject << endl;
        cout << "Book Published: " << bookPublished << endl;
    }
};

// Derived class: Officer
class Officer : public Staff {
private:
    string grade;

public:
    Officer(int id, string n, string g)
        : Staff(id, n) {
```

```
        grade = g;
    }

    void display() {
        displayStaff();
        cout << "Grade: " << grade << endl;
    }
};

// Derived class: Typist
class Typist : public Staff {
protected:
    int speed;

public:
    Typist(int id, string n, int s)
        : Staff(id, n) {
        speed = s;
    }

    void displayTypist() {
        displayStaff();
        cout << "Typing Speed: " << speed << " wpm" << endl;
    }
};
```

```
// Derived class: Regular Typist
class Regular : public Typist {
public:
    Regular(int id, string n, int s)
        : Typist(id, n, s) {}

    void display() {
        cout << "--- Regular Typist Details ---" << endl;
        displayTypist();
    }
};

// Derived class: Casual Typist
class Casual : public Typist {
private:
    float perDaySalary;

public:
    Casual(int id, string n, int s, float salary)
        : Typist(id, n, s) {
        perDaySalary = salary;
    }

    void display() {
        cout << "--- Casual Typist Details ---" << endl;
        displayTypist();
        cout << "Per Day Salary: " << perDaySalary << endl;
    }
};

int main() {
    int choice;
    cout << "Choose Staff Type:\n1. Teacher\n2. Officer\n3. Regular Typist\n4.
Casual Typist\nEnter choice: ";
    cin >> choice;

    if (choice == 1) {
        int id;
        string name, subject, bookPublished;
        cout << "\nEnter Employee ID: ";
        cin >> id;
        cout << "Enter Name: ";
        cin >> name;
        cout << "Enter Subject: ";
        cin >> subject;
        cout << "Enter Book Published: ";
        cin >> bookPublished;

        Teacher t(id, name, subject, bookPublished);
        cout << "\n--- Teacher Details ---" << endl;
        t.display();
    }
```

```
else if (choice == 2) {
    int id;
    string name, grade;
    cout << "\nEnter Employee ID: ";
    cin >> id;
    cout << "Enter Name: ";
    cin >> name;
    cout << "Enter Grade: ";
    cin >> grade;

    Officer o(id, name, grade);
    cout << "\n--- Officer Details ---" << endl;
    o.display();
}

else if (choice == 3) {
    int id, speed;
    string name;
    cout << "\nEnter Employee ID: ";
    cin >> id;
    cout << "Enter Name: ";
    cin >> name;
    cout << "Enter Typing Speed: ";
    cin >> speed;

    Regular r(id, name, speed);
```

```
        r.display();
    }

    else if (choice == 4) {
        int id, speed;
        float salary;
        string name;
        cout << "\nEnter Employee ID: ";
        cin >> id;
        cout << "Enter Name: ";
        cin >> name;
        cout << "Enter Typing Speed: ";
        cin >> speed;
        cout << "Enter Per Day Salary: ";
        cin >> salary;

        Casual c(id, name, speed, salary);
        c.display();
    }

    else {
        cout << "Invalid choice!" << endl;
    }

    return 0;
}
```

OUTPUT:

OUTPUT 1)

```
Choose Staff Type:

1. Teacher

2. Officer

3. Regular Typist

4. Casual Typist

Enter choice: 1


Enter Employee ID: 101

Enter Name: Ramesh

Enter Subject: Mathematics

Enter Book Published: Algebra for Beginners


Teacher Details

Employee ID: 101

Name: Ramesh

Subject: Mathematics

Book Published: Algebra for Beginners
```

OUTPUT 2)

```
Choose Staff Type:

1. Teacher

2. Officer

3. Regular Typist

4. Casual Typist

Enter choice: 2
Enter Employee ID: 202
Enter Name: Priya
Enter Grade: A

Officer Details
Employee ID: 202
Name: Priya
Grade: A
```

OUTPUT3)

```
Choose Staff Type:
1. Teacher
2. Officer
3. Regular Typist
4. Casual Typist
Enter choice: 3

Enter Employee ID: 303
Enter Name: Suresh
Enter Typing Speed: 55

Regular Typist Details
Employee ID: 303
Name: Suresh
Typing Speed: 55 wpm
```

OUTPUT4)

```
Choose Staff Type:
1. Teacher
2. Officer
3. Regular Typist
4. Casual Typist
Enter choice: 4

Enter Employee ID: 404
Enter Name: Meena
Enter Typing Speed: 45
Enter Per Day Salary: 450.75

Casual Typist Details
Employee ID: 404
Name: Meena
Typing Speed: 45 wpm
Per Day Salary: 450.75
```

Assignment 20

SOURCE CODE:

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

class AddAmount {
private:
    double amount;

public:
    AddAmount() {
        amount = 50.0;
    }

    AddAmount(double additionalAmount) {
        amount = 50.0 + additionalAmount;
    }

    void displayAmount() {
        cout << "The final amount in the Piggie Bank is: $" << amount << endl;
    }
};

int main() {
    AddAmount piggie1;
    piggie1.displayAmount();

    AddAmount piggie2(30.0);
    piggie2.displayAmount();

    return 0;
}
```

OUTPUT:

```
The final amount in the Piggie Bank is: $50
The final amount in the Piggie Bank is: $80
```

Assignment 4

SOURCE CODE:

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

class Person {
protected:
    string name;
    int code;

public:
    void getPersonInfo() {
        cout << "Enter name: ";
        cin >> name;
        cout << "Enter code: ";
        cin >> code;
    }

    void displayPersonInfo() {
        cout << "Name: " << name << endl;
        cout << "Code: " << code << endl;
    }
};
```

```
class Account : public Person {
protected:
    double pay;

public:
    void getAccountInfo() {
        cout << "Enter pay: ";
        cin >> pay;
    }

    void displayAccountInfo() {
        cout << "Pay: $" << pay << endl;
    }
};

class Admin : public Person {
protected:
    int experience;

public:
    void getAdminInfo() {
        cout << "Enter experience (in years): ";
        cin >> experience;
    }

    void displayAdminInfo() {
        cout << "Experience: " << experience << " years" << endl;
    }
};

class PERSON : public Account, public Admin {
public:
    void getPERSONInfo() {
        getPersonInfo();
        getAccountInfo();
        getAdminInfo();
    }

    void displayPERSONInfo() {
        displayPersonInfo();
        displayAccountInfo();
        displayAdminInfo();
    }
};

int main() {
    PERSON personObj;

    personObj.getPERSONInfo();
    cout << "\nPERSON Information: " << endl;
    personObj.displayPERSONInfo();
    return 0;
}
```

OUTPUT:

```
Enter name: Sham
Enter code: 101
Enter pay: 5000
Enter experience (in years): 5

PERSON Information:
Name: Sham
Code: 101
Pay: $5000
Experience: 5 years
```

Assignment 5

SOURCE CODE:

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

class Book {
public:
   string title;
   string author;
   string publisher;
   float price;
   int stock;

   // Constructor
   Book(string t, string a, string p, float pr, int s) {
      title = t;
      author = a;
      publisher = p;
      price = pr;
      stock = s;
   }

   void displayInfo() {
      cout << "Title: " << title << endl;
      cout << "Author: " << author << endl;
      cout << "Publisher: " << publisher << endl;
      cout << "Price: $" << price << endl;
      cout << "Stock: " << stock << " copies" << endl;
   }

   void sellBook(int copies) {
      if (copies <= stock) {
          cout << "Total cost: $" << price * copies << endl;
          stock -= copies;
      } else {
          cout << "Required copies not in stock!" << endl;
      }
   }
};

int main() {
   // Updated book names
   Book book1("Calculus", "Mark", "CodeHouse", 350.0, 8);
   Book book2("MachineLearning", "Sara", "AIWorld", 600.0, 4);

   string searchTitle, searchAuthor;
   int copies;

   cout << "Enter book title: ";
   cin >> searchTitle;
   cout << "Enter author name: ";
   cin >> searchAuthor;

   if (book1.title == searchTitle && book1.author == searchAuthor) {
      book1.displayInfo();
      cout << "Enter number of copies: ";
      cin >> copies;
      book1.sellBook(copies);
   }
   else if (book2.title == searchTitle && book2.author == searchAuthor) {
      book2.displayInfo();
      cout << "Enter number of copies: ";
      cin >> copies;
      book2.sellBook(copies);
   }
   else {
      cout << "Book not available." << endl;
   }

   return 0;
}
```

OUTPUT:

```
Enter book title: Calculus
Enter author name: Mark
Title: Calculus
Author: Mark
Publisher: CodeHouse
Price: $350
Stock: 8 copies
Enter number of copies: 3
Total cost: $1050
```

Assignment 6

SOURCE CODE:

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

class DB; // Forward declaration

class DM {
    int meters;
    int centimeters;

public:
    DM(int m = 0, int cm = 0) {
        meters = m;
        centimeters = cm;
    }

    void display() {
        cout << "Distance: " << meters << " meters " << centimeters << "
centimeters" << endl;
    }

    friend DM add(DM, DB);
};
```

```
class DB {
    int feet;
    int inches;

public:
    DB(int f = 0, int in = 0) {
        feet = f;
        inches = in;
    }

    void display() {
        cout << "Distance: " << feet << " feet " << inches << " inches" << endl;
    }

    friend DM add(DM, DB);
};

// 1 meter = 3.28084 feet
// 1 foot = 30.48 cm, 1 inch = 2.54 cm

DM add(DM d1, DB d2) {
    // Convert DB to centimeters
    float total_cm = d1.meters * 100 + d1.centimeters;
    total_cm += d2.feet * 30.48 + d2.inches * 2.54;
    // Convert total_cm to meters and cm
    int final_m = (int)(total_cm / 100);
    int final_cm = (int)(total_cm) % 100;

    return DM(final_m, final_cm);
}

int main() {
    DM d1(3, 40);        // 3 meters 40 cm
    DB d2(5, 8);         // 5 feet 8 inches

    DM result = add(d1, d2);

    cout << " Result in meters and centimeters " << endl;
    result.display();

    return 0;
}
```

OUTPUT:

```
Result in meters and centimeters
Distance: 5 meters 12 centimeters
```

}

Assignment 7

SOURCE CODE:

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

class SimpleCircle {
private:
   int itsRadius;

public:
   SimpleCircle() {
      itsRadius = 5;
   }

   ~SimpleCircle() {}

   int getRadius() const {
      return itsRadius;
   }

   SimpleCircle(int radius) {
      itsRadius = radius;
   }
```

```
    SimpleCircle operator++() {
        itsRadius++;
        return SimpleCircle(itsRadius);
    }

    SimpleCircle operator++(int) {
        SimpleCircle temp = *this;
        itsRadius++;
        return temp;
    }

    SimpleCircle(const SimpleCircle& other) {
        itsRadius = other.itsRadius;
    }

    SimpleCircle operator=(const SimpleCircle& other) {
        if (&other != this) {
            itsRadius = other.itsRadius;
        }
        return SimpleCircle(itsRadius);
    }
};

int main() {
    SimpleCircle circle1;
    SimpleCircle circle2(9);
    ++circle1;
    circle2++;

    cout << "circle1 radius after increment: " << circle1.getRadius() << endl;
    cout << "circle2 radius after increment: " << circle2.getRadius() << endl;

    circle1 = circle2;

    cout << "circle1 radius after assignment: " << circle1.getRadius() << endl;

    return 0;
}
```

OUTPUT:

```
circle1 radius after increment: 6
circle2 radius after increment: 10
circle1 radius after assignment: 10
```

Assignment 8

SOURCE CODE:

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

class Performance {
private:
    string time;
    int totalSeats;
    int bookedSeats;

public:
    Performance(string t, int seats) {
        time = t;
        totalSeats = seats;
        bookedSeats = 0;
    }

    void bookSeats(int seats) {
        if (seats <= (totalSeats - bookedSeats)) {
            bookedSeats = bookedSeats + seats;
            cout << seats << " seats booked for " << time << " show." << endl;
        } else {
            cout << "Not enough seats available for " << time << " show." << endl;
        }
    }

    void showRemainingSeats() {
        cout << "Remaining seats for " << time << " show: " << (totalSeats -
bookedSeats) << endl;
    }
};

int main() {
    Performance afternoon("1:00 PM", 100);
    Performance evening("5:00 PM", 100);
    Performance night("8:30 PM", 100);

    int choice, seats;

    while (true) {
        cout << "\n1. Book Seats\n2. View Remaining Seats\n3. Exit\nEnter your
choice: ";
        cin >> choice;

        if (choice == 3)
            break;

        cout << "Choose performance:\n1. 1:00 PM\n2. 5:00 PM\n3. 8:30
PM\nEnter: ";
```

```
        int show;
        cin >> show;

        Performance* selected;

        if (show == 1) selected = &afternoon;
        else if (show == 2) selected = &evening;
        else if (show == 3) selected = &night;
        else continue;

        if (choice == 1) {
            cout << "Enter number of seats to book: ";
            cin >> seats;
            selected->bookSeats(seats);
        } else if (choice == 2) {
            selected->showRemainingSeats();
        }
    }

    return 0;
}
```

OUTPUT:

```
1. Book Seats
2. View Remaining Seats
3. Exit
Enter your choice: 1
Choose performance:
1. 1:00 PM
2. 5:00 PM
3. 8:30 PM
Enter: 2
Enter number of seats to book: 3
3 seats booked for 5:00 PM show.

1. Book Seats
2. View Remaining Seats
3. Exit
Enter your choice: 2
Choose performance:
1. 1:00 PM
2. 5:00 PM
3. 8:30 PM
Enter: 2
Remaining seats for 5:00 PM show: 97
```

Assignment 9

SOURCE CODE:

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

class Book {
private:
    int classMark;
    string status;

    static int totalBooks;
    static int booksOnLoan;
    static int booksReserved;
    static int booksMissing;

public:
    Book(int mark) {
        classMark = mark;
        status = "on_shelf";
        totalBooks++;
    }

    void loan() {
        if (status == "on_shelf") {
            status = "on_loan";
            booksOnLoan++;
        }
    }

    void missing() {
        if (status != "missing") {
            if (status == "on_loan") booksOnLoan--;
            if (status == "reserved") booksReserved--;
            status = "missing";
            booksMissing++;
        }
    }

    void reserved() {
        if (status == "on_loan") {
            status = "reserved";
            booksReserved++;
        }
    }

    void returned() {
        if (status == "on_loan") booksOnLoan--;
        if (status == "reserved") booksReserved--;
        if (status == "missing") booksMissing--;
        status = "on_shelf";
    }
```

```
    string state() {
        return status;
    }

    static void summary() {
        cout << "\nLibrary Summary:" << endl;
        cout << "Books in library = " << totalBooks << endl;
        cout << "Books on loan = " << booksOnLoan << endl;
        cout << "Books reserved = " << booksReserved << endl;
        cout << "Books missing = " << booksMissing << endl;
        cout << "Books on shelves = " << totalBooks - booksOnLoan -
booksReserved - booksMissing << endl;
    }
};

int Book::totalBooks = 0;
int Book::booksOnLoan = 0;
int Book::booksReserved = 0;
int Book::booksMissing = 0;

int main() {

    Book book1(201);
    Book book2(202);
    Book book3(203);
    Book book4(204);
    Book book5(205);

    book1.loan();      // book1 is on loan
    book2.loan();      // book2 is on loan
    book2.returned();  // book2 returned
    book3.missing();   // book3 marked missing
    book4.loan();      // book4 on loan
    book5.loan();      // book5 on loan
    book5.reserved();  // book5 reserved

    Book::summary();

    return 0;
}
```

OUTPUT:

```
Library Summary:
Books in library = 5
Books on loan = 2
Books reserved = 1
Books missing = 1
Books on shelves = 1
```

Assignment 10

SOURCE CODE:

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

class Employee {
private:
    int age;
    int yearsOfService;
    int salary;

public:
    Employee() {
        age = 0;
        yearsOfService = 0;
        salary = 0;
    }

    Employee(int a, int y, int s) {
        age = a;
        yearsOfService = y;
        salary = s;
    }
    void setAge(int a) {
        age = a;
    }

    void setYearsOfService(int y) {
        yearsOfService = y;
    }

    void setSalary(int s) {
        salary = s;
    }

    int salaryInThousands() {
        return (salary + 500) / 1000;
    }

    void display() {
        cout << "Age: " << age << endl;
        cout << "Years of Service: " << yearsOfService << endl;
        cout << "Salary: $" << salary << endl;
        cout << "Salary in thousands: $" << salaryInThousands() << "000" << endl;

    }
};

int main() {
    Employee e1(25, 3, 45000);
    Employee e2(40, 10, 98000);

    e1.display();
    e2.display();

    return 0;
}
```

OUTPUT:

```
Age: 25
Years of Service: 3
Salary: $45000
Salary in thousands: $45000

Age: 40
Years of Service: 10
Salary: $98000
Salary in thousands: $98000
```

Assignment 12

SOURCE CODE

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

class fruit{
    public:
    int n;
    void count(int x,int y){

        n=x+ y;
        cout<<"Total Fruits:"<<n<<endl;
    }

};
class Apple : public fruit{
    public:
    void count1(int x){
        cout<<"Total apples in basket:"<<x<<endl;
    }

};
class mango : public fruit{
    public:
```

```
    void count2(int y){
        cout<<"Total mangoes in basket:"<<y<<endl;
    }

};

int main()
{ int x,y;

       cout<<"enter apple:"<<endl;
       cin>>x;



       cout<<"enter mango:"<<endl;
       cin>>y;
       Apple a;
       a.count1(x);
        mango b;
        b.count2(y);
        fruit f;
        f.count(x,y);



    return 0;
}
```

OUTPUT

```
enter apple:
4
enter mango:
6
Total apples in basket:4
Total mangoes in basket:6
Total Fruits:10
```

Assignment 12

SOURCE CODE

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

class Fruit {
protected:
    int totalFruits;

public:
    Fruit() {
        totalFruits = 0;
    }

    void updateTotal(int count) {
        totalFruits += count;
    }

    void displayTotal() {
        cout << "Total Fruits in Basket: " << totalFruits << endl;
    }
};

class Apples : public Fruit {
private:
    int appleCount;
public:
    void setApples(int count) {
        appleCount = count;
        updateTotal(appleCount);
    }

    void displayApples() {
        cout << "Number of Apples: " << appleCount << endl;
    }
};

class Mangoes : public Fruit {
private:
    int mangoCount;

public:
    void setMangoes(int count) {
        mangoCount = count;
        updateTotal(mangoCount);
    }

    void displayMangoes() {
        cout << "Number of Mangoes: " << mangoCount << endl;
    }
};
```

```
int main() {
    Apples apple;
    Mangoes mango;

    int a, m;
    cout << "Enter number of apples: ";
    cin >> a;
    apple.setApples(a);

    cout << "Enter number of mangoes: ";
    cin >> m;
    mango.setMangoes(m);

    cout << "\n--- Fruit Basket Summary ---\n";
    apple.displayApples();
    mango.displayMangoes();

    // Total fruits from a fresh object (to ensure total is accurate)
    Fruit total;
    total.updateTotal(a + m);
    total.displayTotal();

    return 0;
}
```

OUTPUT

```
Enter number of apples: 5
Enter number of mangoes: 3

--- Fruit Basket Summary ---
Number of Apples: 5
Number of Mangoes: 3
Total Fruits in Basket: 8
```

Assignment 13

SOURCE CODE

```
#include <iostream>

#include <string>

using namespace std;



class Marks {

protected:

   static int nextRollNumber; // Static variable for automatic roll number
generation

   int rollNumber;

   string name;


public:

   Marks() {

      rollNumber = nextRollNumber++;

   }


   void inputName() {

      cout << "Enter student's name: ";

      cin >> name;

   }


   void displayDetails() {
        cout << "Roll No: " << rollNumber << ", Name: " << name;
    }
};

int Marks::nextRollNumber = 1; // Initialize static roll number



class Physics : virtual public Marks {
protected:
    float physicsMarks;

public:
    void inputPhysicsMarks() {
        cout << "Enter marks in Physics: ";
        cin >> physicsMarks;
    }

    float getPhysicsMarks() {
        return physicsMarks;
    }
};



class Chemistry : virtual public Marks {
protected:
    float chemistryMarks;
```

```
public:
    void inputChemistryMarks() {
        cout << "Enter marks in Chemistry: ";
        cin >> chemistryMarks;
    }

    float getChemistryMarks() {
        return chemistryMarks;
    }
};


class Mathematics : virtual public Marks {
protected:
    float mathMarks;

public:
    void inputMathMarks() {
        cout << "Enter marks in Mathematics: ";
        cin >> mathMarks;
    }

    float getMathMarks() {
        return mathMarks;
    }
```

```
};


class Student : public Physics, public Chemistry, public Mathematics {
public:
    void inputData() {
        inputName();
        inputPhysicsMarks();
        inputChemistryMarks();
        inputMathMarks();
    }

    float totalMarks() {
        return physicsMarks + chemistryMarks + mathMarks;
    }

    void displayData() {
        displayDetails();
        cout << ", Physics: " << physicsMarks
             << ", Chemistry: " << chemistryMarks
             << ", Mathematics: " << mathMarks
             << ", Total: " << totalMarks() << endl;
    }
};

int main() {
    int n;
    cout << "Enter number of students: ";
    cin >> n;

    Student students[100];
    float totalClassMarks = 0;

    for (int i = 0; i < n; i++) {
        cout << "\nEnter details for student " << (i + 1) << ":\n";
        students[i].inputData();
        totalClassMarks += students[i].totalMarks();
    }

    cout << "\n--- Student Marks ---\n";
    for (int i = 0; i < n; i++) {
        students[i].displayData();
    }

    float averageMarks = totalClassMarks / n;
    cout << "\nClass Average Marks: " << averageMarks << endl;

    return 0;
}
```

OUTPUT

```
Enter number of students: 4

Enter details for student 1:
Enter student's name: sanvi
Enter marks in Physics: 89
Enter marks in Chemistry: 79
Enter marks in Mathematics: 90

Enter details for student 2:
Enter student's name: Priya
Enter marks in Physics: 56
Enter marks in Chemistry: 78
Enter marks in Mathematics: 68

Enter details for student 3:
Enter student's name: Omkar
Enter marks in Physics: 100
Enter marks in Chemistry: 100
Enter marks in Mathematics: 100

Enter details for student 4:
Enter student's name: Anna
Enter marks in Physics: 79
Enter marks in Chemistry: 80
Enter marks in Mathematics: 70
```

```
--- Student Marks ---

Roll No: 1, Name: sanvi, Physics: 89, Chemistry: 79, Mathematics: 90, Total:
258

Roll No: 2, Name: Priya, Physics: 56, Chemistry: 78, Mathematics: 68, Total: 202

Roll No: 3, Name: Omkar, Physics: 100, Chemistry: 100, Mathematics: 100,
Total: 300

Roll No: 4, Name: Anna, Physics: 79, Chemistry: 80, Mathematics: 70, Total:
229


Class Average Marks: 247.25
```

Assignment 14

SOURCE CODE

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



class Vehicle {
public:
   int price;
   int milage;
};

class Car : public Vehicle {
public:
   int ownership_cost;
   int warranty;
   int seating_capacity;
   char fuel; // d for diesel, p for petrol
};



class Bike : public Vehicle {
public:
   int no_cylinder;
   int gears;
```

```
    string cooling;     // air, liquid, or oil
    string wheelType;   // alloys or spokes
    int fuel_tank_size; // in inches
};


class Audi : public Car {
public:
    string model_type;

    void setData() {
        cout << "Enter Audi Model: ";
        cin >> model_type;
        cout << "Enter ownership cost(in Lakh): ";
        cin >> ownership_cost;
        cout << "Enter warranty (years): ";
        cin >> warranty;
        cout << "Enter seating capacity: ";
        cin >> seating_capacity;
        cout << "Enter type of fuel (d for diesel, p for petrol): ";
        cin >> fuel;
        cout << "Enter mileage: ";
        cin >> milage;
        cout << "Enter price: ";
        cin >> price;
    }
    void display() {
        cout << "\n--- Audi Car Details ---" << endl;
        cout << "Model: " << model_type << endl;
        cout << "Ownership Cost: " << ownership_cost << endl;
        cout << "Warranty: " << warranty << " years" << endl;
        cout << "Seating Capacity: " << seating_capacity << endl;
        cout << "Fuel Type: " << (fuel == 'd' ? "Diesel" : "Petrol") << endl;
        cout << "Mileage: " << milage << " km/l" << endl;
        cout << "Price: ₹" << price << endl;
        cout<<"\n";
    }
};


class Ford : public Car {
public:
    string model_type;

    void setData() {
        cout << "Enter Ford Model: ";
        cin >> model_type;
        cout << "Enter ownership cost: ";
        cin >> ownership_cost;
        cout << "Enter warranty (years): ";
        cin >> warranty;
        cout << "Enter seating capacity: ";
        cin >> seating_capacity;
        cout << "Enter type of fuel (d for diesel, p for petrol): ";
        cin >> fuel;
        cout << "Enter mileage: ";
        cin >> milage;
        cout << "Enter price: ";
        cin >> price;
    }

    void display() {
        cout << "\n--- Ford Car Details ---" << endl;
        cout << "Model: " << model_type << endl;
        cout << "Ownership Cost: " << ownership_cost << endl;
        cout << "Warranty: " << warranty << " years" << endl;
        cout << "Seating Capacity: " << seating_capacity << endl;
        cout << "Fuel Type: " << (fuel == 'd' ? "Diesel" : "Petrol") << endl;
        cout << "Mileage: " << milage << " km/l" << endl;
        cout << "Price: ₹" << price << endl;
    }
};


class Bajaj : public Bike {
public:
    string make_type;
```

```
void setData() {
    cout << "Enter Bajaj Make Type: ";
    cin >> make_type;
    cout << "Enter number of cylinders: ";
    cin >> no_cylinder;
    cout << "Enter number of gears: ";
    cin >> gears;
    cout << "Enter cooling type (air/liquid/oil): ";
    cin >> cooling;
    cout << "Enter wheel type (alloys/spokes): ";
    cin >> wheelType;
    cout << "Enter fuel tank size (in inches): ";
    cin >> fuel_tank_size;
    cout << "Enter mileage: ";
    cin >> milage;
    cout << "Enter price: ";
    cin >> price;
}

void display() {
    cout << "\n--- Bajaj Bike Details ---" << endl;
    cout << "Make Type: " << make_type << endl;
    cout << "Cylinders: " << no_cylinder << endl;
    cout << "Gears: " << gears << endl;
    cout << "Cooling Type: " << cooling << endl;
```

```
        cout << "Wheel Type: " << wheelType << endl;
        cout << "Fuel Tank Size: " << fuel_tank_size << " inches" << endl;
        cout << "Mileage: " << milage << " km/l" << endl;
        cout << "Price: ₹" << price << endl;
    }
};


class TVS : public Bike {
public:
    string make_type;

    void setData() {
        cout << "Enter TVS Make Type: ";
        cin >> make_type;
        cout << "Enter number of cylinders: ";
        cin >> no_cylinder;
        cout << "Enter number of gears: ";
        cin >> gears;
        cout << "Enter cooling type (air/liquid/oil): ";
        cin >> cooling;
        cout << "Enter wheel type (alloys/spokes): ";
        cin >> wheelType;
        cout << "Enter fuel tank size (in inches): ";
        cin >> fuel_tank_size;
        cout << "Enter mileage: ";
        cin >> milage;
        cout << "Enter price: ";
        cin >> price;
    }

    void display() {
        cout << "\n--- TVS Bike Details ---" << endl;
        cout << "Make Type: " << make_type << endl;
        cout << "Cylinders: " << no_cylinder << endl;
        cout << "Gears: " << gears << endl;
        cout << "Cooling Type: " << cooling << endl;
        cout << "Wheel Type: " << wheelType << endl;
        cout << "Fuel Tank Size: " << fuel_tank_size << " inches" << endl;
        cout << "Mileage: " << milage << " km/l" << endl;
        cout << "Price: ₹" << price << endl;
    }
};


int main() {
    Audi a;
    a.setData();
    a.display();

    Ford f;
    f.setData();
    f.display();

    Bajaj b;
    b.setData();
    b.display();

    TVS t;
    t.setData();
    t.display();

    return 0;
}
```

OUTPUT

```
Enter Audi Model: Q5
Enter ownership cost(in Lakh): 46
Enter warranty (years): 3
Enter seating capacity: 5
Enter type of fuel (d for diesel, p for petrol): d
Enter mileage: 25
Enter price: 50

--- Audi Car Details ---
Model: Q5
Ownership Cost: 46
```

```
Warranty: 3 years
Seating Capacity: 5
Fuel Type: Diesel
Mileage: 25 km/l
Price: ₹50

Enter Ford Model: Figo
Enter ownership cost: 10
Enter warranty (years): 3
Enter seating capacity: 4
Enter type of fuel (d for diesel, p for petrol): d
Enter mileage: 14
Enter price: 15

--- Ford Car Details ---
Model: Figo
Ownership Cost: 10
Warranty: 3 years
Seating Capacity: 4
Fuel Type: Diesel
Mileage: 14 km/l
Price: ₹15

Enter Bajaj Make Type: Pulsar
Enter number of cylinders: 4
Enter number of gears: 3
Enter cooling type (air/liquid/oil): air
Enter wheel type (alloys/spokes): alloys
Enter fuel tank size (in inches): 4
Enter mileage: 13
Enter price: 16

--- Bajaj Bike Details ---
Make Type: Pulsar
Cylinders: 4
Gears: 3
Cooling Type: air
Wheel Type: alloys
Fuel Tank Size: 4 inches
Mileage: 13 km/l
Price: ₹16
Enter TVS Make Type: XYZ
Enter number of cylinders:
4
Enter number of gears: 4
Enter cooling type (air/liquid/oil): liquid
Enter wheel type (alloys/spokes): spokes
Enter fuel tank size (in inches): 32
Enter mileage: 14
Enter price: 15

--- TVS Bike Details ---
```

Make Type: XYZ

Cylinders: 4

Gears: 4

Cooling Type: liquid

Wheel Type: spokes

Fuel Tank Size: 32 inches

Mileage: 14 km/l

Price: ₹15

Assignment 15

SOURCE CODE

```
#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;
}
```

OUTPUT

```
This is a shape
Polygon is a shape
Rectangle is a polygon
Triangle is a polygon
Square is a rectangle
```

Assignment16

SOURCE CODE

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

class RBI {
public:
   float minInterestRate;
   float minBalance;
   float maxWithdraw;

   void setRules() {
      minInterestRate = 4.0;    // 4% interest
      minBalance = 1000;        // Rs. 1000
      maxWithdraw = 25000;      // Rs. 25000
   }

   void showRules() {
      cout << "RBI Rules:\n";
      cout << "Minimum Interest Rate: " << minInterestRate << "%\n";
      cout << "Minimum Balance: Rs. " << minBalance << endl;
      cout << "Max Withdrawal Limit: Rs. " << maxWithdraw << endl;
   }
};
```

```
class Bank {
public:
    string name;
    float interestRate;
    RBI r;

    void setBank(string n, float rate) {
        name = n;
        r.setRules();  // get RBI rules

        if (rate < r.minInterestRate) {
            interestRate = r.minInterestRate;
        } else {
            interestRate = rate;
        }
    }

    void showDetails() {
        cout << "\nBank Name: " << name << endl;
        cout << "Interest Rate: " << interestRate << "%\n";
        cout << "Minimum Balance: Rs. " << r.minBalance << endl;
        cout << "Max Withdrawal: Rs. " << r.maxWithdraw << endl;
    }
};
```

```
int main() {
    Bank b1;
    b1.setBank("SBI", 4.5);
    b1.showDetails();

    Bank b2;
    b2.setBank("MyBank", 3.0); // below RBI rate
    b2.showDetails();

    return 0;
}
```

OUTPUT

```
Bank Name: SBI
Interest Rate: 4.5%
Minimum Balance: Rs. 1000
Max Withdrawal: Rs. 25000

Bank Name: MyBank
Interest Rate: 4%
Minimum Balance: Rs. 1000
Max Withdrawal: Rs. 25000
```

Assignment 17

SOURCE CODE

```
#include <iostream>
using namespace std;
class RBI {
public:
   float minInterestRate;
   float minBalance;
   float maxWithdraw;

   RBI() {
      minInterestRate = 4.0;
      minBalance = 1000;
      maxWithdraw = 25000;
   }

   void showRules() {
      cout << "RBI Rules:\n";
      cout << "Minimum Interest Rate: " << minInterestRate << "%\n";
      cout << "Minimum Balance: Rs. " << minBalance << endl;
      cout << "Maximum Withdrawal Limit: Rs. " << maxWithdraw << endl;
   }
};
class Customer {
public:
   string name;
```

```
    int age;

    void setDetails(string n, int a) {
        name = n;
        age = a;
    }

    void showDetails() {
        cout << "Customer Name: " << name << endl;
        cout << "Customer Age: " << age << endl;
    }
};
class Account {
public:
    float balance;

    void setBalance(float b) {
        balance = b;
    }

    void showBalance() {
        cout << "Account Balance: Rs. " << balance << endl;
    }
};
class SBI : public RBI {
public:
    float interestRate;

    SBI() {
        interestRate = 4.5;
    }

    void showBankDetails() {
        cout << "\nBank: SBI\n";
        cout << "Interest Rate: " << interestRate << "%\n";
    }
};
class ICICI : public RBI {
public:
    float interestRate;

    ICICI() {
        interestRate = 5.0;
    }

    void showBankDetails() {
        cout << "\nBank: ICICI\n";
        cout << "Interest Rate: " << interestRate << "%\n";
    }
};
class PNB : public RBI {
public:
    float interestRate;

    PNB() {
        interestRate = 4.0;
    }

    void showBankDetails() {
        cout << "\nBank: PNB\n";
        cout << "Interest Rate: " << interestRate << "%\n";
    }
};
int main() {
    Customer c;
    c.setDetails("Riya", 20);
    c.showDetails();

    Account a;
    a.setBalance(3000);
    a.showBalance();
    SBI sbi;
    sbi.showBankDetails();
    sbi.showRules();

    ICICI icici;
    icici.showBankDetails();
    PNB pnb;
    pnb.showBankDetails();

    return 0;
}
```

OUTPUT

```
Customer Name: Riya
Customer Age: 20
Account Balance: Rs. 3000

Bank: SBI
Interest Rate: 4.5%
RBI Rules:
Minimum Interest Rate: 4%
Minimum Balance: Rs. 1000
Maximum Withdrawal Limit: Rs. 25000

Bank: ICICI
Interest Rate: 5%

Bank: PNB
Interest Rate: 4%
```

Assignment 18

SOURCE CODE

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

class Student {
public:
    string name;
    Student(string n = "Unknown") {
        name = n;
    }
    void showName() {
        cout << "Student Name: " << name << endl;
    }
};
int main() {
    Student s1("Mina");
    Student s2;
    s1.showName();
    s2.showName();
    return 0;
}
```

OUTPUT

```
Student Name: Mina
Student Name: Unknown
```

Assignment 19

SOURCE CODE

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

class Rectangle {
public:
   float length, breadth;
   Rectangle() {
      length = 0;
      breadth = 0;
   }
   Rectangle(float l, float b) {
      length = l;
      breadth = b;
   }

   Rectangle(float side) {
      length = side;
      breadth = side;
   }
   void calculateArea() {
      cout << "Area: " << length * breadth << endl;
   }
};

int main() {

   Rectangle r1;
   r1.calculateArea();

   Rectangle r2(4, 5);
   r2.calculateArea();

   Rectangle r3(6);
   r3.calculateArea();

   return 0;
}
```

OUTPUT

```
Area: 0
Area: 20
Area: 36
```