KANPUR INSTITUTE OF TECHNOLOGY

(An Autonomous Institute of AKTU, Lucknow)

A- 1, UPSIDC Industrial Area, Rooma, Kanpur-208001 (U.P.) India

OBJECT ORIENTED SYSTEM DESIGN WITH C++ (BCS-054)

Department of Computer Science & Engineering

Session 2025-26

Branch: Computer Science

Year: 3rd

Semester:5th

Submitted by:

Name: Surya Pratap Singh

Roll No- 2301650100158

Section- B

Submitted to: Ms. Saba Nehal

Write a code to implement all the oops concept ?

Code-

OOPS Concepts Example: Restaurant Ordering System

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

// 1. Abstraction & Encapsulation

```
class MenuItem {
private:
    string name;
    double price;

public:
    MenuItem(string n, double p) : name(n), price(p) {}

    string getName() { return name; }
    double getPrice() { return price; }

    virtual string itemType() = 0; // Abstraction

    virtual void showInfo() {
        cout << left << setw(15) << itemType()
             << setw(25) << name
             << "₹" << price << endl;
    }
};
```

// 2. Inheritance

```
class Starter : public MenuItem {
public:
    Starter(string n, double p) : MenuItem(n, p) {}
    string itemType() override { return "Starter"; }
};

class MainCourse : public MenuItem {
public:
    MainCourse(string n, double p) : MenuItem(n, p) {}
    string itemType() override { return "Main Course"; }
};

class Dessert : public MenuItem {
public:
    Dessert(string n, double p) : MenuItem(n, p) {}
    string itemType() override { return "Dessert"; }
};
```

// 3. Polymorphism in Action

```
void showMenu(vector<MenuItem*> menu) {
    cout << "\n=== RESTAURANT MENU ===" << endl;
    cout << left << setw(15) << "Type" << setw(25) << "Item" << "Price" << endl;
    cout << "-----------------------------------------------" << endl;
    for(auto item : menu) {
        item->showInfo();
    }
}

void placeOrder(vector<MenuItem*> order) {
    double total = 0;
    cout << "\n=== ORDER DETAILS ===" << endl;
    cout << left << setw(25) << "Item" << "Price" << endl;
    cout << "----------------------------" << endl;
    for(auto item : order) {
        cout << left << setw(25) << item->getName() << "₹" << item->getPrice() << endl;
        total += item->getPrice();
    }
    cout << "----------------------------" << endl;
    cout << left << setw(25) << "Total Amount" << "₹" << total << endl;
}
```

// 4. Main Program

```
int main() {
    Starter s1("Spring Rolls", 150);
    Starter s2("Garlic Bread", 120);
    MainCourse m1("Pasta Alfredo", 350);
    MainCourse m2("Grilled Chicken", 450);
    Dessert d1("Chocolate Cake", 200);
    Dessert d2("Ice Cream", 120);

    vector<MenuItem*> menu = { &s1, &s2, &m1, &m2, &d1, &d2 };
    vector<MenuItem*> order = { &s1, &m2, &d1 };

    showMenu(menu);
    placeOrder(order);

    cout << "\n=== ENCAPSULATION DEMONSTRATION ===" << endl;
    cout << "Accessing menu item safely: " << s1.getName() << endl;

    return 0;
}
```

Output -

```
=== RESTAURANT MENU ===
Type           Item                      Price
-------------------------------------------------
Starter        Spring Rolls              ₹150
Starter        Garlic Bread              ₹120
Main Course    Pasta Alfredo             ₹350
Main Course    Grilled Chicken           ₹450
Dessert        Chocolate Cake            ₹200
Dessert        Ice Cream                 ₹120

=== ORDER DETAILS ===
Item                    Price
-----------------------------
Spring Rolls            ₹150
Grilled Chicken         ₹450
Chocolate Cake          ₹200
-----------------------------
Total Amount            ₹800

=== ENCAPSULATION DEMONSTRATION ===
Accessing menu item safely: Spring Rolls
```