CPP-A18: Polymorphism

📝 Problem 1: Smart Calculator

📌 **Problem Statement:**

Design a **Smart Calculator** that supports addition in multiple ways:

1. Adding two integers
2. Adding two floating-point numbers
3. Adding three integers
4. Concatenating two strings

🔷 **Hint:** Use **Function Overloading**.

🔻 **Example Input & Output:**

```
Input:
add(5, 10)
Output: 15

Input:
add(3.5, 2.5)
Output: 6.0

Input:
add("Hello", " World")
Output: Hello World
```

📝 Problem 2: Dynamic Array Operations

📌 **Problem Statement:**

Create a **Dynamic Array** class that allows:

1. **Adding an integer** to an array

2. **Adding two arrays** together using `+` operator
3. **Printing the array** using `<<` operator

🔷 **Hint:** Use **Operator Overloading** for `+` and `<<`.

🔻 **Example Input & Output:**

```
Input:
arr1 = {1, 2, 3};
arr2 = {4, 5, 6};
arr3 = arr1 + arr2;
cout << arr3;

Output:
{1, 2, 3, 4, 5, 6}
```

📝 Problem 3: Employee Salary System

📌 **Problem Statement:**

Design a class **Employee** that supports multiple ways of creating an employee object:

1. Default constructor
2. Parameterized constructor with `name` and `salary`
3. Copy constructor

🔷 **Hint:** Use **Constructor Overloading**.

🔻 **Example Input & Output:**

```
Input:
Employee e1;
Employee e2("John", 50000);
Employee e3 = e2;
e3.printDetails();

Output:
Name: John, Salary: 50000
```

📝 Problem 4: Complex Number Arithmetic

📌 **Problem Statement:**

Create a class **ComplexNumber** that supports:

1. Adding two complex numbers using `+` operator
2. Multiplying two complex numbers using operator
3. Printing complex numbers using `<<` operator

🔷 **Hint:** Use **Operator Overloading**.

🔻 **Example Input & Output:**

```
Input:
ComplexNumber c1(3, 4);
ComplexNumber c2(1, 2);
ComplexNumber sum = c1 + c2;
ComplexNumber product = c1 * c2;
cout << sum;
cout << product;

Output:
3 + 4i + 1 + 2i = 4 + 6i
(3 + 4i) * (1 + 2i) = -5 + 10i
```

📝 Problem 5: Banking System Transactions

📌 **Problem Statement:**

Implement a **Bank Account** system where:

1. `deposit(int amount)`: Deposits an integer amount.
2. `deposit(double amount)`: Deposits a floating-point amount.
3. `withdraw(int amount)`: Withdraws money.

🔷 **Hint:** Use **Function Overloading**.

🔻 **Example Input & Output:**

```
Input:
Account acc(1000);
```

```
acc.deposit(500);
acc.deposit(100.75);
acc.withdraw(300);
acc.printBalance();

Output:
Current Balance: 1300.75
```

📌 Problem 6: E-Commerce Discount System

An e-commerce platform wants to apply discounts dynamically based on different purchase categories.

- If a customer buys **electronics**, they get a **10% discount**.
- If a customer buys **clothing**, they get a **20% discount**.
- If a customer buys **groceries**, they get a **5% discount**.

📝 **Hint:** The discount calculation should be handled dynamically based on the type of purchase.

Input Example:

```
Enter category: Electronics
Enter price: 5000
```

Expected Output:

```
Final Price after discount: 4500
```

📌 Problem 7: Smart Home Automation System

A smart home system should handle different appliances that can be turned **on** or **off**.

- **Fan:** Can be switched on/off with speed control.
- **Light:** Can be switched on/off with brightness control.
- **TV:** Can be switched on/off with volume control.

📝 **Hint:** Each appliance should behave differently when turned on/off.

Input Example:

```
Turning on the Fan at speed 3
Turning on the Light at brightness level 5
Turning on the TV at volume 15
```

Expected Output:

```
Fan is running at speed 3
Light is glowing at brightness level 5
TV is playing at volume 15
```

📌 Problem 8: Online Food Delivery Bill Calculation

An online food delivery app calculates the **total bill** based on the type of order.

- If a customer orders **Pizza**, it costs **₹250**.
- If a customer orders **Burger**, it costs **₹150**.
- If a customer orders **Pasta**, it costs **₹200**.

📝 **Hint:** The system should handle different order types and calculate the final amount accordingly.

Input Example:

```
Enter food item: Pizza
Enter quantity: 2
```

Expected Output:

```
Total bill: ₹500
```

📌 Problem 9: Ride Fare Calculation in a Transport App

A transport app needs to calculate ride fares for different vehicles.

- **Bike Ride:** ₹5 per km.
- **Car Ride:** ₹10 per km.
- **Auto Ride:** ₹7 per km.

📝 **Hint:** The fare calculation should vary depending on the type of vehicle chosen.

Input Example:

```
Enter vehicle type: Car
Enter distance: 10 km
```

Expected Output:

```
Total fare: ₹100
```

📌 Problem 10: Banking System - Different Interest Rates

A banking system provides different interest rates based on the type of account.

- **Savings Account:** 4% annual interest.
- **Fixed Deposit Account:** 6% annual interest.
- **Recurring Deposit Account:** 5% annual interest.

📝 **Hint:** The interest calculation should be handled differently for each account type.

Input Example:

```
Account Type: Fixed Deposit
Deposit Amount: ₹10000
Duration: 2 years
```

Expected Output:

```
Total interest earned: ₹1200
```

Happy Coding!