Department of Statistics & Computer Science
University of Kelaniya
Academic Year – 2022/2023
COSC 12043 / BECS 12243 - Object Oriented Programming
Tutorial 09

---

1. Write a recursive method **factorial(n)** that computes the factorial of a given natural number 'n'. Given an n (>=1), the factorial of n is defined as the product of a positive integer and all the positive integers below it;

   i.e., n* (n-1)* (n-2)*...* 1.

   For example,

   factorial(1) → 1

   factorial(2) → 2 (2*1)

   factorial(3) → 6 (3*2*1)

   factorial(4) → 24 (4*3*2*1)

2. Explain step-by-step how the following method executes when called with **gcd(36, 60)**:

```
public static int gcd(int a, int b) {
    if (b == 0) return a;
    return gcd(b, a % b);
}
```

3. Write a recursive method to find the n^{th} **Fibonacci number**.

$$F(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ F(n-1) + F(n-2) & \text{if } n > 1 \end{cases}$$

   For example,

   F(0) → 0

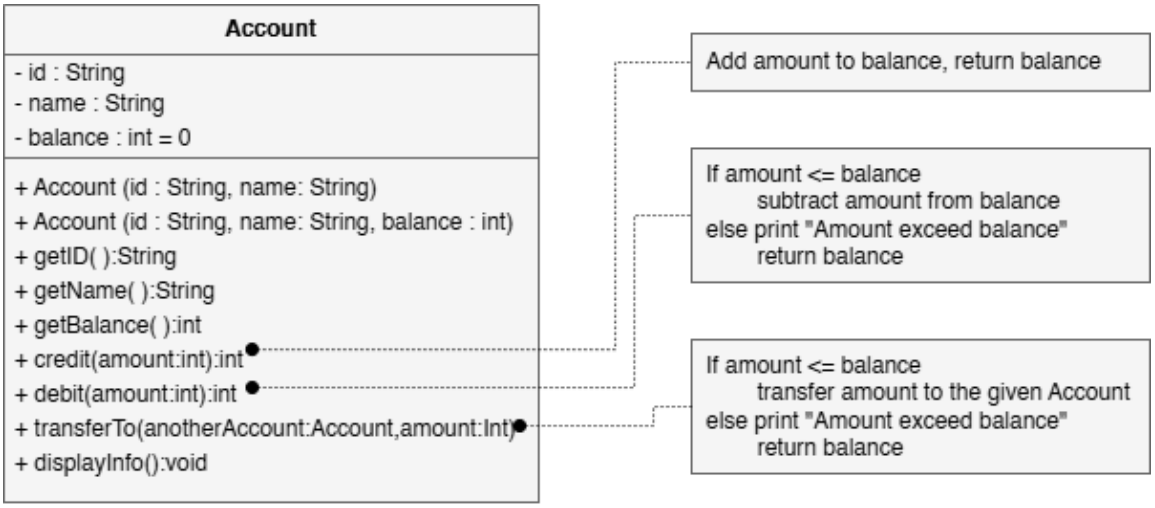
   F(1) → 1

   F(2) → 1 (1 + 0)

   F(3) → 2 (1 + 1)

   F(4) → 3 (2 + 1)

4. The following class diagram shows the **Account** class with methods such as credit(amount), debit(amount), and transferTo(anotherAccount, amount). Based on the information provided in the diagram, write a Java program to implement the **Account** class.

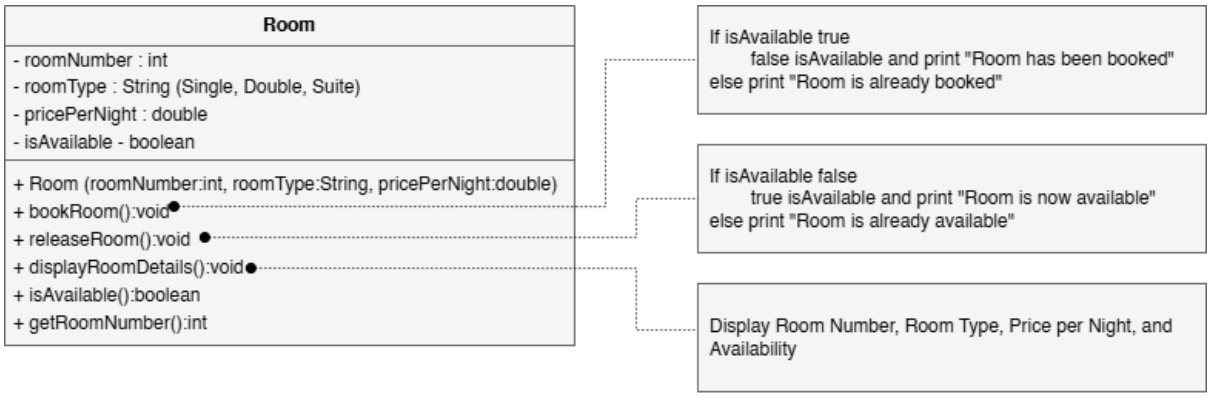


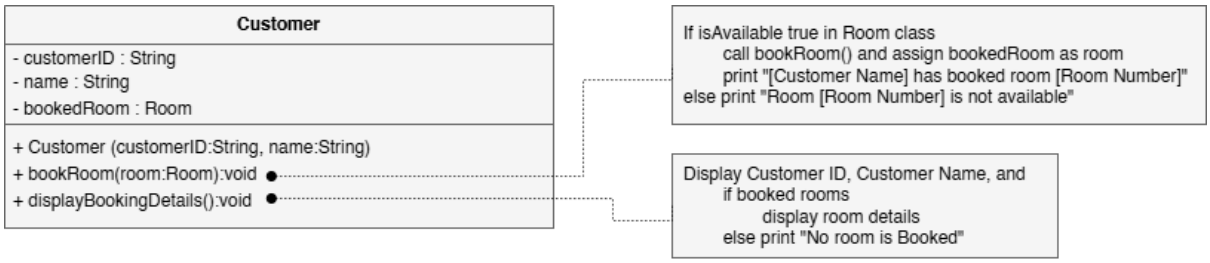
Below is a test driver to test the **Account** class. Run the test driver to get the output of the code.

```
public class TestAccount {
    public static void main(String[] args) {
        Account account1 = new Account("A013", "Alice", 1500);
        Account account2 = new Account("A014", "Bob", 500);

        System.out.println("Initial Balances:");
        account1.displayInfo();
        account2.displayInfo();

        System.out.println("Transferring 300 from Alice to Bob...");
        account1.transferTo(account2, 300);
        account1.displayInfo();
        account2.displayInfo();

        System.out.println("Transfer 2000 from Alice to Bob...");
        account1.transferTo(account2, 2000);
        account1.displayInfo();
        account2.displayInfo();
    }
}
```

5. Write a Java class called **GymMember** with two fields: memberName(String) and sessionsLeft(int). The class should have two constructors and five methods: getMemberName(), getSessionsLeft(), bookSession(), cancelSession(), and displayMemberInfo().
    a. The first constructor should initialize the memberName to null and sessionsLeft to 0.
    b. The second constructor should initialize the memberName and sessionsLeft based on the values passed as parameters.
    c. The bookSession() method should decrease sessionsLeft by 1. If no sessions are available, it should display a message: "No sessions left to book."
    d. The cancelSession() method should increase sessionsLeft by 1, simulating a canceled booking.
    e. The displayMemberInfo() method should display the member name and the number of sessions left. The message "No sessions available" should be displayed if no sessions are left.
    f. Write a client program called GymMemberClient that:
        i. Creates a GymMember object member1 with the name "John Doe" and sessionsLeft set to 10.
        ii. Simulates booking 3 sessions using bookSession() three times and displays the updated member info using displayMemberInfo().
        iii. Simulates canceling 1 session using cancelSession() and displays the updated member info again.
        iv. Creates another GymMember object member2 without passing any parameters and displays the member name and available sessions using displayMemberInfo().

6. Write Java programs called **Room,** and **Customer** based on the information provided in the following diagrams.





Write another Java program to do the following:

a. Create two rooms with the following details:

   Room 101: Type: Single, Price: 1500.0, Initially Available.

   Room 102: Type: Double, Price: 2500.0, Initially Available.

b. Create two customers with the following details:

   Customer 1: Name: Alice, ID: "C001", Initially no room booked.

   Customer 2: Name: Bob, ID: "C002", Initially no room booked.

c. Alice books Room 101. Display the booking details for both Alice and Bob.

d. Attempt to book Room 101 for Bob.

e. Alice releases Room 101. Display the updated booking details for both Alice and Bob after this action.