# Object Oriented Programming

## Assignment # 03

**Note:**

- First think about a problem statement and then write/draw your logic on paper.
- After designing the logic on paper, code the problem statement on any editor (VS Code, Gedit, etc).
- Copied tasks will be awarded **zero** marks without any investigation.
- Comments you code properly.
- Assignment After Due DateAssignment will not be Accepted.
- Plagiarism of any shape or form will not be tolerated. In case of plagiarism, the particular question will be marked zero and 50% marks from total obtained marks will be deducted.

## Bank Management System

**Note:** use Concepts like Composition or Aggregation, Friend Function, Friend Class.

You are asked to create a simplified bank management system using C++ classes. Your system should be able to manage multiple bank accounts, allow deposits and withdrawals, provide balance inquiries, and keep track of the total balance across all accounts. You should implement this system without using inheritance.

Here are the requirements for your system:

I. Create a **BankAccount** class with the following

- **private members** for the **BankAccount** class:
    1. **accountNumber** (an integer): A unique identifier for each account.
    2. **accountHolder** (a string): The name of the account holder.
    3. **balance** (a double): The current balance in the account.
    4. Implement a logic to keep track of the total accounts created. i.e., number of objects of **BankAccount**
- **member functions** for the **BankAccount** class:
    1. A constructor that initializes the account number to a unique value in sequence and sets the initial balance to 0 and account holders name to a default value of "N/A" or receive while the creation of an object.
    2. A **deposit** function that takes an amount and adds it to the account's balance.
    3. A **withdraw** function that takes an amount and subtracts it from the account's balance, if sufficient funds are available.

4. A **getBalance** function that returns the current balance.
5. A **getAccountNumber** function that returns the account number.
6. A **getTotalAccounts** function that returns the total number of bank accounts created.
7. Make A **Transfer Funds** function to transfer money from one account to another.

II. Implement a **BankManager** class that manages a collection of **BankAccount** objects. ·

private data members

1. a **pointer** of type **BankAccount**

· public member functions

1. Implement a function to **create** bank a account **dynamically** during runtime when required
2. Similarly, implement a function to **delete a bank account** whenever the bank manager required so.
3. Implement a function to **find a bank account** by account number and **return a pointer** to it.
4. Implement a function to **display all** the **account details** including account number, account holder name, and balance.

III. Create a menu drive CLI interface to interact with the application. *For simplicity, skip the password part.*

**NOTE:** Submit only the <rollno_yourname.cpp> file, 1 mark will be deducted otherwise. **For example 22p-0001_ali-imran.cpp.** 0 marks will be assigned to plagiarized assignments.