# Week008 – Machine Problem 3

**Objective/s:**

At the end of this activity, you should be able to:

* define classes
* implement polymorphism and inheritance
* instantiate an object array
* apply Exception to all input validation
* create a banking application for multiple users

**What to Prepare for the Activity:**

* NetBeans IDE 8.2
* JDK8 (Java Development Kit 8)

**Procedure:**

* Create a NetBeans project for this activity. The project name should be as follows:

Project Name: MP3\_<lastname\_firstname>

Example: **MP3\_Blanco\_Maria**

* The class names to be created are the following:
  + Client (the main class that contains the main method and the implementation of the main menu)
  + SavingsAccount (the subclass where attributes and methods are defined)
  + BankAccount (the superclass where attributes and methods are defined)
* All class names must be suffixed with your last name.
* Client<your\_last\_name> Ex. **ClientBlanco**
* SavingsAccount<your\_last\_name> Ex. **SavingsAccountBlanco**
* BankAccount<your\_last\_name> Ex. **BankAccountBlanco**
* Compress the NetBeans project into .rar or .zip format and then **upload to the link provided in the LMS.**
* **Only NetBeans project compressed in .rar or .zip format will be accepted. All other formats will be graded with 0.**

Write a Java program with the following specifications:

1. Define the following attributes and methods of the BankAccount class:

* Attributes: accountName, address, birthday, contactNumber
* Constructor
* Setter and Getter methods
* getClientDetails method

1. Define the following attributes and methods of the SavingsAccount class:

* Attribute: accountNo, balance, interestRate
* Constructor
* Setter and Getter methods
* balanceInquiry method
* deposit method
* withdraw method
* validateAcctNumber method
* closeAccount method

1. Note that the SavingsAccount class will be inherited from the BankAccount class.
2. Define the following methods in the Client class:

* main method
* displayMainMenu method
* other applicable methods

1. Note that the object to be instantiated in the main method is an object array.

For example: SavingsAccount[ ] sa = new SavingsAccount[100];

1. The program should have the main menu as shown below and will always loop back to the main menu after each transaction.

**Bank Main Menu**

[1] New Account

[2] Balance Inquiry

[3] Deposit

[4] Withdraw

[5] Client Profile

[6] Close Account

[7] Exit

1. The **New Account** option should implement the following:
   * Input client details: name, address, birthday, and contact number
   * Input the initial deposit of not less than PhP 5,000
   * Generate a four-digit account number randomly
2. The **Balance Inquiry** option should implement the following:
   * Input the account number and validate
   * If the account number is valid, display the client name and current balance
3. The **Deposit** option should implement the following:
   * Input the account number and validate
   * If the account number is valid, input the amount of deposit of not less than Php 100
   * Compute and update the current balance as follows:

Balance+=amount of deposit

Interest=5% of balance

Balance+=interest

1. The **Withdraw** option should implement the following:
   * Input the account number and validate
   * If the account number is valid, input the amount to be withdrawn, not less than Php 100, not greater than the balance, and an amount of Php 5,000 should be maintained (maintaining balance)
   * Compute and update the balance as follows:

Balance-=amount withdrawn

1. The **Client Profile** option should implement the following:

* Input the account number and validate
* If account number is valid, display the client details including the current balance.

1. The **Close Account** option should implement the following:

* Input the account number and validate
* If account number is valid, confirm if the user closes the account then set the balance to zero
* Otherwise, go back to the main menu

1. The **Exit** option terminates the program.
2. The following **Main Menu** validations should be implemented:

* Validate the transaction code by the user
* The user should not be allowed to perform transactions 2 to 5, unless a new account has been created
* Apply all input validations.