**Object-Oriented Programming Lab#6, Spring 2019**

**Today’s Topics**

* Class and Object
* Initialization of fields.
* Constructor
* String concatenation
* Array of reference type

**Problems/Assignments**

**Problem#1**

Create a Banking System, where a user can create new account, deposit money, withdraw money and check the balance

**What you need to do:**

1. Create a **BankAccount** class which has **3 instance variables**; name, id and balance. Create a constructor with 3 arguments and initialize the attributes. The class also has the **following 4 methods**;
   1. void deposit(double depAmount)

* inside the method the balance need to be increased by the “depAmount” amount.
  1. void withdraw(double withAmount)
* the balance is decreased by “withAmount” amount. We have to make sure the balanced do not become negative.
  1. double getBalance()
* the method returns the balance.
  1. void display()

– this method displays the attributes in the format “Name:[name]; Id:[id]; Balance:[balance]”.

1. Now create a class name “**Bank**” which will mimic a real Bank that holds a list of BankAccount. You can use an Array or ArrayList to hold the list of BankAccount. So, the class will have only one attribute ***BankAccount[] accounts***. Add the following methods to the class.
   1. void addAccount(String name, String id, double balance)

* this method will add a new BankAccount object to the list. Use the parameters to create the BankAccount object.
  1. void deposit(String id, double amt)
* Inside the method call findAccount to find the account with matching id and then call deposit method of that object.
  1. void withdraw(String id, double amt)
* Inside the method call findAccount to find the account with matching id and then call withdraw method of that object.
  1. void display()
* Loop through the list of the BankAccount and call display of BankAccount class.
  1. BankAccount findAccount(String id)
* This method will loop through the list of the BankAccount and find the account that has matching id as the parameter. If the Account is available return the object otherwise return null.

1. Create an **application class** (that has the main method) named “**BankApp**” which will have the **main** method.
   * In the main method, display the following menu to user and take necessary action.
     + Input ‘1’ to add a new Account.
     + Input ‘2’ to deposit to an existing account
     + Input ‘3’ to withdraw from an account.
     + Input ‘4’ to display the list of the accounts.
     + Input ‘0’ to exit the system.