Lab05	IT618 Enterprise Computing
	PM Jat, DA-IICT, Winter’2017

Problem #1: Consider having a poll for a student body position at your institute. You need to create two classes that simulate the polling. Let us name the classes as PollSimulator and PollDB.

1. You need to implement the class **`PollDB`** such that code below runs correctly as expected.

```
import java.util.Random;

public class PollSimulator {

    public static void main(String[] args) {

        String[] candNames = new String[5];
        candNames[0] = "Deep Goyal";
        candNames[1] = "Sneha Mohan";
        candNames[2] = "Praveen Shah";
        candNames[3] = "Akshay Singh";
        candNames[4] = "Somin Majumdar";

        PollDB poll_db = new PollDB( candNames );

        int nc = poll_db.getNumberOfCandidates();
        Random rand = new Random(55);
        for( int i = 1; i <=50; i++ ) {
            int x = 1 + rand.nextInt(nc);
            poll_db.voteTo( x );
        } //Assume that candidates are identified as 1 through 5

        //Report the election result
        for(int i = 1;
             i <= poll_db.getNumberOfCandidates(); i++ ) {
             System.out.println(poll_db.getCandidateName(i)
                        + ": " + poll_db.votesCount(i));
        }
        System.out.println("Winner: " + poll_db.getWinner());
    }
}
```

2. Modify **`PollSimulator`** class such that, it takes vote cast from the console (in a finite loop) rather than random generated casts like the one happening above.

3. Also, modify PollSimulator such that before producing the election result, it prompts for the password, and if password entered is correct, it produces the result. If password is wrong, it should prompt again (total three chance only). Also correct password should be encapsulated in PollDB class as field, not in code or PollSimulator.

continued … 2

Problem #2: Consider BankAcount class that has been discussed in one of the lectures. It is available on course website at http://courses.daiict.ac.in/mod/folder/view.php?id=9106. Suppose we need extend functionality of this class by modifying it. Needed enhancements are described below.

1. Suppose we want to record all transactions on a BankAccount. For that we decide to create a new class called Transaction. Let us say a transaction object holds four values- Date, Narration, Debit/Credit, and Amount. You need to implement this class with appropriate constructor and accessor/mutator methods. For date you can use **`java.util.Calendar`** class.

2. You also need to modify implementation of deposit and withdraw methods so that appropriate transaction object is added (in addition to updating the balance) in transaction history. You may also have to add appropriate field for the same in BankAccount object.

3. Let us add a method **payInterest** to the BankAccount class. Use interface given below for the method. Assume that this method is automatically called on midnight of last day of every month: **`double payInterest(double rate)`**. This method computes interest, adds it to the balance, and appends appropriate transaction object in the history.

   Interest is computed for minimum balance in the month. Computing minimum balance for the month could be complex, therefore you can simulate it; for instance, you can rely on a private methods called **`getMinimumBalalnceInMonth()`**. Rate of interest is supplied as parameter to the method. The methods also returns the interest that was paid to the account.

4. Let us also create a class called Bank. The bank class primarily is a collection of BankAccount object. It is expected that you use **`java.util.HashMap`** method to hold account objects. Let us say Bank object also holds following values-
   a. BankName
   b. Interest Rate: rate of interest that is paid on a saving account.
   c. InterestPaidLog: Is basically ordered list tuples <Date, AccountNo, Amount>

5. We add following methods to Bank object class

```
int openNewAccount(double initial_balance)
  //returns account number of newly opened account
BankAccount find(int acc_no)
BankAccount close(int acc_no)
void payInterest(int month, int year)
  //iterates through all accounts and pays interest to all accounts by sending payInterest to
  // respective account. The method also saves an entry in InterestPaidLog of the bank.
double getTotalInterestPaid (int year)
double getTotalDeposits(int year)
```

6. Create a simulator Tester application for the Bank class.