Assignment – 6

1. *Problem 1 solution*

```
CLASS QuadraticEquation
BEGIN

    CREATE  a,b,c

    CONSTRUCTOR QuadraticEquation (a,b,c)
    BEGIN

        this.a←a
        this.b←b
        this.c←c
    END CONSTRUCTOR

    METHOD getA( )
    BEGIN

        RETURN this.a
    END METHOD

    METHOD getB( )
    BEGIN

        RETURN this.b
    END METHOD

    METHOD getC( )
    BEGIN
```

```
        RETURN this.c
    END METHOD

    METHOD getDiscriminant( )
    BEGIN

        disc←this.b*this.b-4*this.a*this.c;
        RETURN disc
    END METHOD

    METHOD getRoot1
    BEGIN

        R1←(-this.b+ (getDiscriminant( ) )^1/2)/ (2*this.a)
        RETURN R1
    END METHOD

    METHOD getRoot2
    BEGIN

        R2←(-this.b-(getDiscriminant( ))^1/2)/ (2*this.a)
        RETURN R2
    END METHOD
END CLASS


CLASS TestEquation
BEGIN

    METHOD MAIN( )
    BEGIN
```

```
        READ user_input for 3 coefficients of  a quadratic
                equation
        PRINT “Enter the first coefficient of the quadratic
                equation”
        a←user_input for first coefficient
        PRINT “Enter the first coefficient of the quadratic
                equation”
         b←user_input for second coefficient
        PRINT “Enter the first coefficient of the quadratic
                equation”
        c←user_input for third coefficient
        PRINTLINE( )

        CREATE D as QuadraticEquation
        D← NEW QuadraticEquation(a,b,c)

        PRINT “a= ”+D.getA( )
        PRINT “b= ”+D.getB( )
        PRINT “c= ”+D.getC( )

       IF (D.getDiscriminant( )>0) THEN

            PRINT “Root 1= ”+D.getRoot1( )
            PRINT “Root 2= ”+D.getRoot2( )
      ELSE

            PRINT “Root 1= undefined”
            PRINT “Root 2= undefined”
     ENDIF
END CLASS
```

2. *Problem 2 solution*

```
CLASS Counter
BEGIN

        CREATE counter
        counter←0

        METHOD increment( )
        BEGIN

            a←getValue( )
            a←a+1
            counter←a
        END METHOD

        METHOD getValue( )
        BEGIN

            RETURN counter
        END METHOD
END CLASS


CLASS coinToss
BEGIN

        METHOD MAIN( )
        BEGIN

            total←100
```

```
        CREATE Head and Tails as Counter
        Head←NEW Counter( )
        Tails←NEW Counter( )
       WHILE(total>0)

            IF (Math.random( )<0.5) THEN
                Head.increment( )
            ELSE
                Tails.increment( )
            ENDIF
            total←total-1
      END WHILE

      PRINT "Total number of heads: "+Head.getValue( )
      PRINT "Total number of tails: "+Tails.getValue( )
  END MAIN( )
END CLASS
```

3. *Problem 3 solution*

```
CLASS BankAccount
BEGIN

    CREATE id, balance, annualInterestRate
    Date dateCreated←NEW Date( )

    CONSTRUCTOR BankAccount( )
    BEGIN

        this.id←0
        this.balance←0.0
```

```
        this.annualInterestRate←0.0

END CONSTRUCTOR

CONSTRUCTOR BankAccount(id, balance)
BEGIN

    this.id←id
    this.balance←balance
END CONSTRUCTOR

METHOD setId (id)
BEGIN

    this.id←id
END METHOD

METHOD setBalance (balance)
BEGIN

    this.balance←balance
END METHOD

METHOD setAnnualInterestRate (annualInterestRate)
BEGIN

    this.annualInterestRate←annualInterestRate/100
END METHOD( )

METHOD getId( )
BEGIN
```

```
    RETURN this.id
END METHOD

METHOD getBalance( )
BEGIN

    RETURN this.balance
END METHOD

METHOD getAnnualInterestRate( )
BEGIN

    RETURN this.annualInterestRate
END METHOD

METHOD getDate( )
BEGIN

    RETURN this.dateCreated
END METHOD

METHOD getMonthlyInterestRate( )
BEGIN

    RETURN(this.annualInterestRate/12)*100
END METHOD

METHOD getMonthlyInterest( )
BEGIN

    RETURN (this.balance*getMonthlyInterestRate( ))
END METHOD
```

```
        METHOD withdraw (withdraw)
        BEGIN

              this.balance←balance-withdraw
              RETURN balance
        END METHOD

        METHOD deposit (deposit)
        BEGIN

             this.balance←balance+deposit
             RETURN balance
        END METHOD

        METHOD toString ( )
        BEGIN

              name← "Account id: "+this.id+ "\nAccount balance:
              "+this.balance+ "\nInterestRate:
              "+this.annualInterestRate*100+ " %"
              RETURN name
        END METHOD
END CLASS


CLASS TestBankAccount
BEGIN

        METHOD MAIN ( )
        BEGIN

             CREATE myObject as BankAccount
```

```
        myObject←NEW BankAccount(123456, 10000)
        myObject.setAnnualInterestRate←2.5
        myObject.withdraw←3500
        myObject.deposit←500
        myObject.getBalance( )

        PRINT myObject.toString( )
        PRINT (“Date created: ”+myObject.getDate( ))
        PRINT (“Earned Monthly Interest:
                ”+myObject.getMonthlyInterest( ))

    END MAIN( )
END CLASS
```