VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

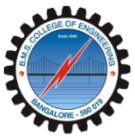
LAB REPORT
on

Object Oriented Java Programming

Submitted by

SAMARTH M SHETTY (1BM21CS184)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Oct 2022-Feb 2023

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**OBJECT ORIENTED JAVA PROGRAMMING**" carried out by **SAMARTH M SHETTY(1BM21CS184),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022-23. The Lab report has been approved as it satisfies the academic requirements in respect of Data structures Lab **- (22CS3PCDST)**work prescribed for the said degree.

Dr. Rajeshwari B S
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

`

Index Sheet

LAB PROGRAM 1:

Develop a Java program that prints all real solutions to the quadratic equation. ax2+bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2-4ac is negative, display a message stating that there are no real solutions.

```
import java.util.Scanner;
import java.lang.Math;

class QuadraticEquation{
	public static void main(String [] args){
	int a,b,c;
	int d=0;
	Scanner scanner = new Scanner(System.in);

	System.out.println("Enter the coefficients of the quadratic equation in the form a, b,c where ax2 + bx + c=0");
	a = scanner.nextInt();
	b = scanner.nextInt();
	c = scanner.nextInt();

	d = ((int)Math.pow(b,2)- 4*a*c);
	double result1 = (-b + Math.pow(d,0.5))/(2*a);
	double result2 = (-b - Math.pow(d,0.5))/(2*a);

	if(d==0){
		System.out.println("The roots are real and equal");
		System.out.println("The root is:" + " " + result1 );
```

```
        }
        else if(d>0){
                System.out.println("The roots are real and distinct");
                System.out.println("The roots are: ");
                System.out.println("root 1 " + result1);
                System.out.println("root 2" + result2);
        }
        else{
                double factor = (Math.sqrt(Math.abs(d))) / (2*a);
                System.out.println("The roots are imaginary");
                System.out.println("The roots are: ");
                System.out.println("root 1 " + -b/(2*a) + "+i " + factor);
                System.out.println("root 2 " + -b/(2*a) + "-i " + factor);
        }

}
}
```

```
import java.util.Scanner;
import java.lang.math;

class QuadraticEquation {
    public static void main (String[] args) {
    int a, b, c;
    int d = 0;
    Scanner scanner = new Scanner (System.in);

System.out.println ("Enter the coefficients of the quadratic
equation in the form a, b, c (where ax^2 + bx + c = 0 ),
a = scanner.nextInt();
b = scanner.nextInt();
c = scanner.nextInt();
d = ((int) Math.pow (b, 2)) - 4*a*c);
double result1 = (-b + Math.pow(d,0.5)) / (2*a);
double result2 = (-b - Math.pow(d, 0.5)) /(2*a);

if (d == 0) {
    System.out.println ("The roots are real & equal");
    System.out.println ("The root is :" + " " + result1);
}
else if (d>0) {
    System.out.println ("The roots are real & distinct");
    System.out.println ("The roots are:");
    System.out.println ("root 1" + result1);
    System.out.println ("root 2" + result2);
}
else {
    double factor = (Math.sqrt (Math.abs(d))) / (2*a);
```

System.out.println("Root2" + -b/(2*a) + "-i" + f
}
}
}

output:

Enter the Coefficient
1
2
2

Root are imaginary.
The Roots are
Root 1 -1+i1.0
Root 2 -1-i1.0

Enter the coefficient
1
2
0
Root are real & defined
Root 1 0.0
Root 2 -2.0

Enter the coefficient
1
2
1
Root are real & equal
The root is : -1.0.

Output:

```
C:\Users\bmsce\Desktop\Sam184>javac QuadraticEquation.java

C:\Users\bmsce\Desktop\Sam184>java QuadraticEquation
Enter the coefficients of the quadratic equation in the form a, b,c where ax2 + bx + c=0
1
2
2
The roots are imaginary
The roots are:
root 1 -1+i 1.0
root 2 -1-i 1.0

C:\Users\bmsce\Desktop\Sam184>java QuadraticEquation
Enter the coefficients of the quadratic equation in the form a, b,c where ax2 + bx + c=0
1
2
0
The roots are real and distinct
The roots are:
root 1 0.0
root 2-2.0

C:\Users\bmsce\Desktop\Sam184>java QuadraticEquation
Enter the coefficients of the quadratic equation in the form a, b,c where ax2 + bx + c=0
1
2
1
The roots are real and equal
The root is: -1.0
```

Lab Program 2:

Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

```
import java.util.Scanner;
class Studentx{
   Scanner s=new Scanner(System.in);
   private int usn;
   private String name;
   int n;
   int[] marks=new int[10];
   int[] credits=new int[10];
   void getMarks(){
      System.out.println("Enter no of subjects:");
      this.n=s.nextInt();
      System.out.println("Enter marks of subjects and credits consecutively:");
      for(int i=0;i<n;i++){
         this.marks[i]=s.nextInt();
         this.credits[i]=s.nextInt();
      }
   }
   int[] getGrade(){
      int[] grade=new int[10];
      for(int i=0;i<n;i++){
         if(marks[i]>=90){
            grade[i]=10;
         }else if(marks[i]>=80){
            grade[i]=9;
         }else if(marks[i]>=70){
            grade[i]=8;
         }else if(marks[i]>=60){
            grade[i]=7;
         }else if(marks[i]>=50){
            grade[i]=6;
         }else if(marks[i]>=40){
            grade[i]=5;
         }else if(marks[i]>=30){
            grade[i]=4;
         }else{
            grade[i]=0;
         }
      }
      return grade;
```

```
    }
    double calc(){
        int sum1=0,sum2=0;
        for(int i=0;i<n;i++){
            int[] grade;
            grade=getGrade();
            sum1+=grade[i]*this.credits[i];
            sum2+=this.credits[i];
        }
        return (double)sum1/sum2;
    }
    void display(){
        System.out.println("Marks\tCredits\n");
        for(int i=0;i<n;i++){
            System.out.println(this.marks[i]+"\t\t"+this.credits[i]+"\n");
        }
        System.out.println("SGPA="+calc());
    }
}
class mainx{
    public static void main(String args[]){
        Studentx s=new Studentx();
        s.getMarks();
        s.calc();
        s.display();
    }
}
```

```
Import java.util.Scanner;

class Student {
    Scanner S= new Scanner(System.in);
    private int usn;
    private String name;
    int n;
    int[] marks = new int[10];
    int[] credit = new int[10];
    void getMarks(){
        System.out.println("Enter no of subject");
        this.n = S.nextInt();
        System.out.println("Enter marks of subject and credit");
        for(int i=0; i<n; i++){
            this.marks[i] = S.nextInt();
            this.credit[i] = S.nextInt();
        }
    }
    int[] getGrade(){
        int[] grade = new int[10];
        for(int i=0; i<n; i++){
            if(marks[i]>=90){
                grade[i]=0;}
            else if(marks[i]>=80)
                grade[i]=9
            else if(marks[i]>=70)
                grade[i]=8;
            else if(marks[i]>=60)
                grade[i]=7;
```

```
        else if (marks[i] >= 80)
            grade[i] = 4;
        else
            grade[i] = 0;
    }
    return grade;
}

double calculate()
    int sum1 = 0, sum2 = 0;
    for (int i = 0; i < n; i++){
        int[] grade;
        grade = getGrade();
        sum1 += grade[i] * this.credits[i];
        sum2 += this.credits[i];
    }
    return (double) sum1 / sum2;
}
void display(){
    System.out.println("Marks \t Credits");
    for (int i = 0; i < n; i++){
    System.out.println(this.marks[i] + "\t\t" + this.c
    }
    System.out.println("SGPA :" + calculate());
}

class Main{
    public static void main(String args[]){
```

paperism
Date.

Output :

Enter NO of Subject

4

Enter mark of Subject and Credit consecutively.

80 3

90 4

85 5

80 2

Mark	Credit
80	3
90	4
85	5
80	2

SGPA = 9.3333 ...

27/1/23

Output:

```
Enter no of subjects:
4
Enter marks of subjects and credits consecutively:
80 3
90 4
85 3
80 2
Marks   Credits

80      3

90      4

85      3

80      2

SGPA=9.333333333333334

Process finished with exit code 0
```

Lab Program 3:

Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program to create n book objects.

```
import java.util.Scanner;

public class BookDetails{
	public static void main(String args[]){
		Scanner scanner = new Scanner(System.in);
		int noPage;
		double price;
		String title;
		String author;
		int noBooks;
		System.out.println("Enter the number of books");
		noBooks = scanner.nextInt();

		Book book[] = new Book[noBooks];

		for(int i = 0; i<noBooks; i++){
			System.out.println("Enter the book title");
			title = scanner.next();

			System.out.println("Enter the book author");
			author = scanner.next();

			System.out.println("Enter the book price");
			price = scanner.nextDouble();

			System.out.println("Enter the number of pages in the book");
			noPage = scanner.nextInt();

			book[i] = new Book();
			book[i].setTitle(title);
			book[i].setAuthor(author);
			book[i].setPrice(price);
			book[i].setPage(noPage);
		}
```

```
		for(int i = 0; i<noBooks; i++){
			System.out.println(book[i]);
		}
	}
}

class Book{
	String title;
	String author;
	double price;
	int noPage;

	Book(){
		title = "Default";
		author = "Default";
		price = 0.0;
		noPage = 0;
	}

	//setters
	public void setTitle(String title){
		this.title = title;
	}

	public void setAuthor(String author){
		this.author = author;
	}

	public void setPrice(double price){
		this.price = price;
	}

	public void setPage(int noPage){
		this.noPage = noPage;
	}

	//getters
	public String toString(){
		return title+"\t"+author+"\t"+price+"\t"+noPage;
	}

}
```

paperism

Date:

Book Details

```
import java.util.Scanner;

public class BookDetails
    public static void main (String args[]){
        Scanner scanner = new Scanner (System.in);
        int noPage
        double price;
        String title, author;
        int noBooks;
        System.out.println ("Enter the number of books");
        noBooks = scanner.nextInt();

        Book book[] = new Book[noBooks];

    for (int i=0; i<noBooks;i++){
        System.out.println ("Enter the book title");
          title = scanner.next();

        System.out.println ("Enter the book author");
          author = scanner.next();

        System.out.println ("Enter the book price");
    price = scanner.nextDouble();

        System.out.println ("Enter the number of pages in the book");
          noPage = scanner.nextInt();

        book[i] = new Book();
        book[i].setTitle(Title);
        book[i].setAuthor(author);
```

```
for (int i = 0; i < noBooks; i++) {
        System.out.println(Book[i]);
    }
}
}

Class Book {
    String title;
    String author;
    double price;
    int noPages;

    Book() {
        title = "Default";
        author = "Default";
        price = 0.0;
        noPage = 0;

    public void setAuthor (String author) {
        this.author = author;
    }

    public void setTitle (String title) {
        this.title = title;
    }
    public void setPrice (String double price) {
        this.noPage = noPages;
    }
```

Date:

Input:

Enter the number of books

2

Enter the book title

abc

Enter the book outher

xyz

Enter the book price

125

Enter the number of pages in the book

125

Enter the book title

def

Enter the book outher

qwerty

Enter the book price

10.99

Enter the number of pages in the book

50

OUTPUT:

abc	xyz	125	125
def	qwerty	10.99	50.

2/12/22

Output:

```
C:\Users\bmsce\Desktop\Sam184>javac BookDetails.java

C:\Users\bmsce\Desktop\Sam184>java BookDetails
Enter the number of books
2
Enter the book title
abc
Enter the book author
xyz
Enter the book price
12.5
Enter the number of pages in the book
128
Enter the book title
def
Enter the book author
qwerty
Enter the book price
10.99
Enter the number of pages in the book
50
abc     xyz     12.5    128
def     qwerty  10.99   50
```

Lab Program 4:

Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea( ) that prints the area of the given shape.

```
class AbstractRunner{
	public static void main(String args []){
		Rectangle r = new Rectangle(5, 10);
		Triangle t = new Triangle(5,10);
		Circle c = new Circle(5);

		r.printArea();
		t.printArea();
		c.printArea();
	}
}

abstract class Shape{
	int a;
	int b;
	Shape(int a, int b){
		this.a = a;
		this.b = b;
	}
	public abstract void printArea();
}

class Rectangle extends Shape{
	Rectangle(int a, int b){
		super(a, b);
	}
	public void printArea(){
		System.out.println(a * b);
	}
}

class Triangle extends Shape{
	Triangle(int a, int b){
```

```
                super(a, b);
        }
        public void printArea(){
                System.out.println(0.5 * a * b);
        }
}

class Circle extends Shape{
        Circle(int a){
                super(a, a);
        }
        public void printArea(){
                System.out.println(3.14 * a * b);
        }
}
```

4.

```
abstract class Shape {
    int a;
    int b;
    Shape (int a, int b) { this.a = a; this.b = b; }
    public abstract void printArea();
}

class Rectangle extends Shape {
    Rectangle (int a, int b) {
        super (a, b);
    }
    void
    public void printArea() {
        System.out.println (a * b);
    }
}

class Triangle extends Shape {
    Triangle (int a, int b) {
        super (a, b);
    }
    void
    public void printArea() {
        System.out.println (0.5 * a * b);
    }
}

class Circle extends Shape {
    Circle (int a) {
        super (a, a);
    }
    void
    public void printArea() {
```

paperism
Date:

```
Class Main {
    public static void main (String args[]){

        Rectangle r = new Rectangle (5, 10);
        Triangle t = new Triangle (5, 10);
        Circle c = new Circle (8);

        r.printArea();
        t.printArea();
        c.printArea();

    }
}
```

Output

50

25.0

78.5

Bali
9/12/22

Output:

```
PS D:\C_data_structures> cd "d:\C_data_structures\" ; if ($?) { javac AbstractRunner.java }
50
25.0
78.5
PS D:\C_data_structures> cd "d:\C_data_structures\" ; if ($?) { javac AbstractRunner.java }
110
15.0
50.24
```

Lab Program 5:

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

a) Accept deposit from customer and update the balance.

b) Display the balance.

c) Compute and deposit interest

d) Permit withdrawal and update the balance

```
import java.lang.Math;
import java.util.Scanner;


class Bank {
        public static void main(String args []){
                Scanner scanner = new Scanner(System.in);
                String name, accNo;
        CurrentAccount cacc;
        SavingsAccount sacc;
                int choice;
```

```
System.out.println("Enter an option");
System.out.println("1 : Open a Savings Account");
System.out.println("2 : Open a Current Account");
choice = scanner.nextInt();

System.out.println("Enter your Name and Account NO");
name = scanner.next();
accNo = scanner.next();

switch(choice){

        case 1:  sacc = new SavingsAccount(name, accNo);
                 System.out.println("Enter your starting balance");
                 sacc.balance = scanner.nextDouble();

                 int opt = 0;

                 System.out.println("1 :  Deposit");
                 System.out.println("2 :  Withdraw");
                 System.out.println("3 : Display Balance");
                 System.out.println("4: Quit");
                 while(opt != 4){
                         System.out.println("Select an option");
                         opt = scanner.nextInt();
                         switch(opt){
```

```
                                        case 1: System.out.println("Enter the amount you want
to deposit");
                                        double damount = scanner.nextDouble();
                                        sacc.deposit(damount);
                                        break;
                                        case 2:  System.out.println("Enter the amount you want
to withdraw");
                                        double wamount = scanner.nextDouble();
                                        sacc.withdraw(wamount);
                                        break;
                                case 3:
                                        sacc.display();
                                        sacc.interestCalculator(1);
                                        break;
                                case 4:  break;
                }
                }
                break;
        case 2:  cacc= new CurrentAccount(name, accNo);
                System.out.println("Enter your starting balance");
                cacc.balance = scanner.nextDouble();
                opt = 0;

                System.out.println("1 :  Deposit");
                System.out.println("2 :  Withdraw");
                System.out.println("3 : Display Balance");
                System.out.println("4: Quit");
```

```
                while(opt != 4){
                        System.out.println("Select an option");
                        opt = scanner.nextInt();
                        switch(opt){
                                case 1: System.out.println("Enter the amount you want to deposit");
                                double damount = scanner.nextDouble();
                                cacc.deposit(damount);
                                break;
                        case 2:  System.out.println("Enter the amount you want to withdraw");
                                double wamount = scanner.nextDouble();
                                cacc.withdraw(wamount);
                                break;
                        case 3:
                                cacc.display();
                                break;
                        case 4:  break;
                        }
                        }
                break;

                }
        }
}

 abstract class Account {
        String customerName;
```

```
        String accountNumber;
    int accountType;
    double balance = 0;

    Account(String customerName, String accountNumber, int accountType) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
    }

    abstract public void withdraw(double amount);
    public void deposit(double amount) {
        balance += amount;
        System.out.println("Your Balance is : " + balance);
    }
    public void display() {
            System.out.println("Balance is:" + balance);
        }
}
class SavingsAccount extends Account {
        final double rateOfInterest = 0.06;
        final int term = 4;
        SavingsAccount(String customerName, String accountNumber) {
                super(customerName, accountNumber, 1);
        }
        public void withdraw(double amount) {
```

```
                if ((balance - amount) > 0.00)
                balance -= amount;
                else
                System.out.println("Insufficient Balance");
                this.display();
        }
        public void interestCalculator(int period) {
                double principal = this.balance, interestEarned;

                        double quarterlyInterest = rateOfInterest / term;
                        double quarterlyPeriod = period * term;

                        interestEarned = principal * Math.pow((1 + quarterlyInterest), quarterlyPeriod) -
principal;

                System.out.println("Interest earned for this balance (compounded quarterly) for one year
is :" + interestEarned);
        }
}
class CurrentAccount extends Account {
        final double penaltyPercent = 0.10;
        final double minimumBalance = 5000.00;
        CurrentAccount(String customerName, String accountNumber) {
                super(customerName, accountNumber, 2);
                        System.out.println("Cheque book has been issued");
        }
        public void withdraw(double amount) {
```

```
if ((balance - amount) > 0.00)
balance -= amount;
else
System.out.println("Insufficient Balance");
this.display();
if (balance < minimumBalance){
        System.out.println("A penalty of " + balance*penaltyPercent + "has been
imposed");
balance -= balance * penaltyPercent;
}       }       }
```

paperism
Date:

5. Bank.

```
Import java.lang.Both;  Import java.util.Scanner;
abstract class Account{
    String CustomerName;
    String AccountNumber;
    int Account type;
    double balance=0;
    Account (String customerName, String AccountNumber, int type){
      this.CustomerName = customerName;
      this.AccountNumber = AccountNumber;
      this.AccountType = Type;
    }

abstract public void withdraw (double amount);

    public void dipoid (double amount){
        balance += amount;
        System.out.println ("Your balance:" + balance);
    }

    public void display(){
        System.out.println ("Your balance:" + balance);
    }
}

class SoungsAccount extends Account{
    final double rateOfInterest = 0.06;
    final int monthNumber = 12;

    public void withdraw (double amount){
```

```
public void interestCalculator (int period) {
        double principal = this.balance, interestEarned;
        double quarterlyInterest = rateOfInterest / term;
        double quarterlyPeriod = period * term;

        interestEarned = principal * ~~Math.pow((1+quarterlyInterest)~~
          Math.pow ((1+quarterlyInterest), quarterlyPeriod) - principle;

          System.out.println("Interest earned for this balance
(compounded quarterly) for one year is: " + interestEarned);
    }
}

Class CurrentAccount extends Account {
    final double penaltyPercent = 0.10
    final double minimumBalance = 5000.00;

    CurrentAccount (String customerName, String accountNumber) {
        super (customerName, accountNumber, 2.);
    }

    public void withdraw (double amount) {
        if ((balance - amount) > 0.00)
        balance -= amount;
        else
        System.out.println ("Insufficient Balance");
        this.display();
        if (balance < minimumBalance)
          balance -= balance * penaltyPercent;
    }
```

```
class Bank {
    public static void main (String args[]) {
        Scanner sconner = new Scanner (System.in);
        String name, accNO;
        CurrentAccount cacc;
        SavingsAccount sacc;
        int choice;

        System.out.println ("Enter an option");
        System.out.println ("1: Open a Savings account");
        System.out.println ("2) Open a Current account");
        choice = sconner.nextInt();

        System.out.println ("Enter your Name and Account NO");
        name = sconner.next();
        accNO = sconner.next();

        switch (choice) {
        case 1: sacc = new SavingsAccount (name, accNO);
            System.out.println ("Enter your starting balance");
            sacc.balance = sconner.nextDouble();

            int opt = 0;
            System.out.println ("1: Deposit");
            System.out.println ("2: Withdraw");
            System.out.println ("3: Display balance");
```

```
switch (opt) {
  case 1: System.out.println("Enter the amount you want to
          deposit");
          double damount = sc.nextDouble();
          acc.deposit(damount);
          break;
  case 2: System.out.println("Enter the amount you want
          to withdraw");
          double wamount = sc.nextDouble();
          acc.withdraw(wamount);
          break;
  case 3: acc.display();
          acc.interestCalculator();
          break;
  case 4: break;
  }
}
break;
case 2: acc = new CurrentAccount(name, accNo);
        System.out.println("Enter your starting balance");
        acc.balance = sc.nextDouble();
        opt = 0;

System.out.println("1: Deposit");
System.out.println("2: Withdraw");
System.out.println("3: Display Balance");
System.out.println("4: Quit");
```

```
double damount = Scanner.nextDouble();
acc.deposit(damount);
break;
case 2: System.out.println("Enter the amount you
        want to withdraw");
double wamount = scanner.nextDouble();
acc.withdraw(wamount);
break;
case 3:  acc.display();
         break;
case 4:  break;
}
}
break;
}
}
```

output

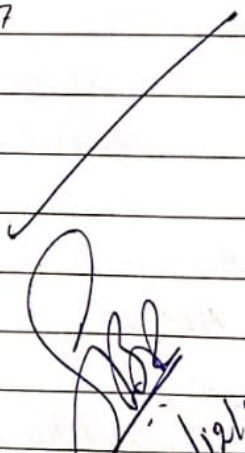
```
Enter an option
1: Open a Savings Account
2: Open a Current Account
1
Enter your name and Account No
Samarth
BANK00001245
Enter your starting balance
12000
```

paperism
Date:

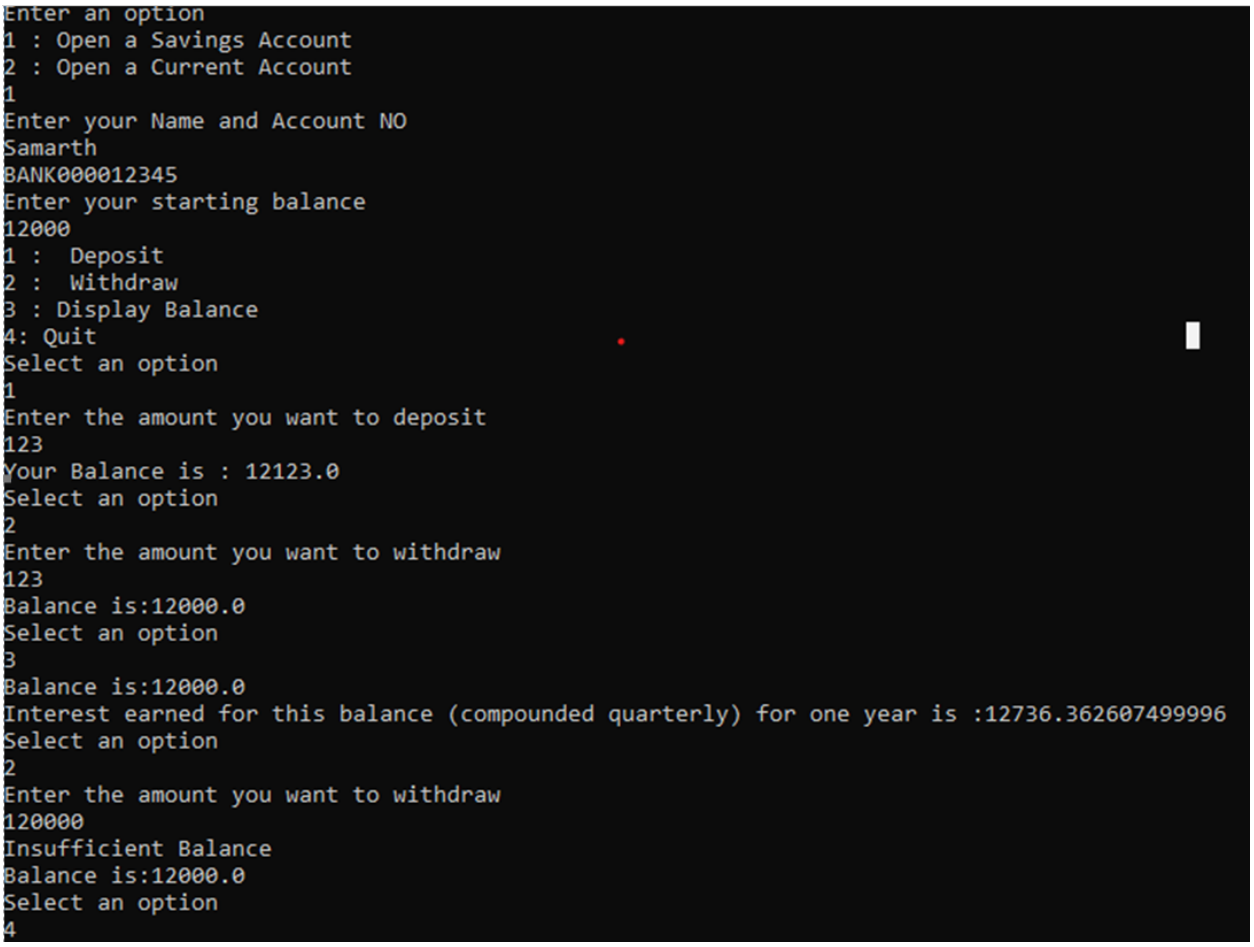
1
Enter amount you want to Deposit
123
Your Balance is : 12123.0
Select an option
2
Enter the amount you want to withdraw
123
Balance is 12 000.0
Select an option
3
Balance is 12000.0
Interest earned for this balance (compounded quarterly) for one year is : 736.36
Select an option
2
Enter the amount you want to withdraw
12 000.
Insufficient Balance.
Balance is 12000.00
Select an option
4.

~~Enter an option~~
Enter an option
1: Open a Savings Account
2: Open a Current Account
2
Enter your name & Account No.

1: Deposit
2: Withdraw
3: Display Balance
4: Quit
Select an option
1
Enter the amount you want to deposit
123
Your Balance is: 12123.0
Select an option
2
Enter the amount you want to withdraw
11000
Balance is: 1123.0
Select an option
1
Enter the amount you want to deposit
1
Your Balance is: 1011.7
Select an option
3
Balance is: 1011.7
Select an option
4.



Output:


Enter an option
1 : Open a Savings Account
2 : Open a Current Account
1
Enter your Name and Account NO
Samarth
BANK000012345
Enter your starting balance
12000
1 : Deposit
2 : Withdraw
3 : Display Balance
4: Quit
Select an option
1
Enter the amount you want to deposit
123
Your Balance is : 12123.0
Select an option
2
Enter the amount you want to withdraw
123
Balance is:12000.0
Select an option
3
Balance is:12000.0
Interest earned for this balance (compounded quarterly) for one year is :12736.362607499996
Select an option
2
Enter the amount you want to withdraw
120000
Insufficient Balance
Balance is:12000.0
Select an option
4

```
Enter an option
1 : Open a Savings Account
2 : Open a Current Account
2
Enter your Name and Account NO
Rahul
BANK0000004321
Enter your starting balance
12000
1 :  Deposit
2 :  Withdraw
3 : Display Balance
4: Quit
Select an option
1
Enter the amount you want to deposit
123
Your Balance is : 12123.0
Select an option
2
Enter the amount you want to withdraw
11000
Balance is:1123.0
Select an option
1
Enter the amount you want to deposit
1
Your Balance is : 1011.7
Select an option
3
Balance is:1011.7
Select an option
4
```

Lab Program 6:

Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

```
import java.util.Scanner;

public class InterfacedScoreCalc {

        public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

    Student student;

        int opt = 0;

        System.out.println("Choose an option\n1.UnderGraduate\n2.Graduate");

        opt = scanner.nextInt();

        switch(opt){

                case 1: System.out.println("Enter your name");

                                student = new UndergraduateStudent(scanner.next());

                                System.out.println("Enter your marks");

                                for(int i = 0; i<4; i++){

                                        student.setTestScore(i, scanner.nextInt());

                                }
```

```
                    System.out.println("Student Name: " + student.getStudentName());
            System.out.println("Result: " + student.getTestResult());
                    break;
            case 2: System.out.println("Enter your name");
                    student = new GraduateStudent(scanner.next());
                    System.out.println("Enter your marks");
                    for(int i = 0; i<4; i++){
                        student.setTestScore(i, scanner.nextInt());
                    }
                    System.out.println("Student Name: " + student.getStudentName());
            System.out.println("Result: " + student.getTestResult());
                    break;
        }
        } }
 interface StudentFunctions{
        void generateResult();
}
abstract class Student implements StudentFunctions{
        String studentName;
        int[] testScores = new int[4];
        String testResult;
        Student(String studentName) {
            this.studentName = studentName;
```

```
testResult = "Default";
 }
        abstract public void generateResult();

        void setTestScore(int testNumber, int testScore) {
testScores[testNumber] = testScore;
 }
        String getStudentName() {
return this.studentName;
        }
        void setStudentName(String studentName) {
this.studentName = studentName;
 }
        int[] getTestScores() {
return this.testScores;
        }
        String getTestResult() {
this.generateResult();
return testResult;
        }
        void setTestResult(String testResult) {
this.testResult = testResult;
        }
```

```
}
class UndergraduateStudent extends Student {
    UndergraduateStudent(String studentName) {
        super(studentName);          }


    @Override
    public void generateResult() {
        int a[] = this.getTestScores();
        int sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += a[i];
        }
        int avg = sum / 4;
        if (avg >= 60)
            this.setTestResult("Pass");
        else
            this.setTestResult("Fail");
             }
}
 class GraduateStudent extends Student {
    GraduateStudent(String studentName) {
        super(studentName);
             }
```

```
    @Override
    public void generateResult() {
        int a[] = this.getTestScores();
        int sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += a[i];
        }
        int avg = sum / 4;
        if (avg >= 70)
            this.setTestResult("Pass");
        else
            this.setTestResult("Fail");
             }
}
```

paperism

Date.

6 Interface

```
import java.util.Scanner;

public class InterfaceScoreCalc {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Student student;
        int opt = 0;
        System.out.println("Choose an option \n1. UnderGraduate
          \n2. Graduate");
        opt = scanner.nextInt();
        switch (opt) {
            case 1: System.out.println("Enter your name");
              student = new UnderGraduateStudent(scanner.next());
            System.out.println("Enter your marks");
            for (int i = 0; i < 4; i++) {
                    student.setTestScore(i, scanner.nextInt());
            }
            System.out.println("Student Name:" + student.getStudentName());
            System.out.println("Result:" + student.getTestResult());
            break;
        case 2: System.out.println("Enter your name");
            student = new GraduateStudent(scanner.next());
            System.out.println("Enter your marks");
            for (int i = 0; i < 4; i++) {
```

paperism
Date:

```
class UndergraduateStudent extends Student {
    UndergraduateStudent(String studentName) {
        super(studentName); }

    @Override
    public void generateResult() {
        int a[] = this.getTestScores();
        int sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += a[i]; }
        int avg = sum/4;
        if (avg >= 60)
            this.setTestResult("Pass");
        else
            this.setTestResult("Fail");
    }
}

class GraduateStudent extends Student {
    GraduateStudent(String studentName) {
        super(studentName); }

    @Override
    public void generateResult() {
        int a[] = this.getTestScores();
        int sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += a[i]; }
```

Output:

Choose an option

1. Undergraduate

2. Graduate

1

Enter your name.

Philip

Enter your marks

70

69

71

55

Student Name: Philip

Result: Pass

Choose an option

1. undergraduate

2. Graduate

1

Enter your name

Jerry

Enter your marks

70

69

71

55

Student Name: Jerry

Result: Fail

Output:

```
Choose an option
1.UnderGraduate
2.Graduate
1
Enter your name
Philip
Enter your marks
70
69
71
55
Student Name: Philip
Result: Pass

C:\Users\bmsce\Desktop\sam 184>java InterfacedScoreCalc
Choose an option
1.UnderGraduate
2.Graduate
2
Enter your name
Jerry
Enter your marks
70
69
71
55
Student Name: Jerry
Result: Fail
```

Lab Program 7:

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that cases both father and son's age and throws an exception if son's age is >=father's age.

```
import java.util.Scanner;

class WrongAgeException extends IllegalArgumentException{
        WrongAgeException(){};
        WrongAgeException(String some){
                super(some);
        }
}

class Father{
        int ageFather;
        Father(){}
        Father(int ageFather) throws WrongAgeException{
                if(ageFather<=0){
                        throw new WrongAgeException("Age should be greater than zero!");
                }
                this.ageFather = ageFather;
        }
}

class Son extends Father{
        int ageSon;
        Son(){}
        Son(int ageFather, int ageSon) throws WrongAgeException{
                super(ageFather);
                if(ageFather <= ageSon){
                        throw new WrongAgeException("Age of son can't be greater than Father's
age!!!");
                }
                this.ageSon = ageSon;
        }

}

class ExceptionInheritTree{
        public static void main(String args[]){
```

```
		Scanner sc = new Scanner(System.in);
		int ageFather, ageSon;
		System.out.println("enter age of father and son");
		ageFather = sc.nextInt();
		ageSon = sc.nextInt();
		try{
			Son son = new Son(ageFather, ageSon);
			System.out.println("Father's age: " + son.ageFather + "\n" + "Sons's age: " +
son.ageSon);
		}
		catch(WrongAgeException wae){
			System.out.println(wae);
			System.out.println("Exception caught in main!#!");
		}

	}

}
```

paperism

Date:

Exception in Inheritance Tree

```
import java.util.Scanner;
import java.lang.*;

class Father {
    int ageFather;
    Father() { }
    Father (int ageFather) throws IllegalArgumentException {
    if (ageFather <= 0) {
        throw new IllegalArgumentException ("Age should be
    greater than zero!"); }
        this.ageFather = ageFather;
    }
}

class Son extends Father {
    int ageSon;
    Son() { }
    Son (int ageFather, int ageSon) throws IllegalArgumentException {
    if   super (ageFather);
        if (ageFather <= ageSon) {
            throw new IllegalArgumentException ("Age of
    son can't be greater than Father's age !!!"); }
        this.ageSon = ageSon;
    }
}

class ExceptionInheritTree {
    public static void main (String args[]) {
        Scanner sc = new Scanner (System.in);
        int ageFather, ageSon;
        System.out.println ("Enter age of father and son");
        ageFather = sc.nextInt();
```

output:

```
C:\Users\bmsce\Desktop\sam 184>java ExceptionInheritTree
enter age of father and son
50 25
Father's age: 50
Sons's age: 25

C:\Users\bmsce\Desktop\sam 184>java ExceptionInheritTree
enter age of father and son
0 12
java.lang.IllegalArgumentException: Age should be greater than zero!
Exception caught in main!#!

C:\Users\bmsce\Desktop\sam 184>java ExceptionInheritTree
enter age of father and son
30 35
java.lang.IllegalArgumentException: Age of son can't be greater than Father's age!!!
Exception caught in main!#!

C:\Users\bmsce\Desktop\sam 184>
```

Lab Program 8:

Write a program which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and another displaying "CSE" once every two seconds.

```
class MyThread extends Thread {
	long time;
	private volatile boolean running = true;

	MyThread() {
		System.out.println("Default");
	}

	MyThread(String name, long time) {
		super(name);
		this.time = time;
	}

	public void pause() {
		running = false;
	}

	public void run() {
		try {
			while (running) {
				System.out.println(this.getName());
				Thread.sleep(time * 1000);
			}
		} catch (InterruptedException ie) {
			System.out.println("Exception caught in method");
		}

	}
}

class ThreadRunner {
	public static void main(String[] args) {
		MyThread mt1 = new MyThread("BMS", 10);
		MyThread mt2 = new MyThread("CSE", 2);
		mt1.start();
		mt2.start();
		try {
			Thread.sleep(20 * 1000);
			mt1.pause();
			mt2.pause();
```

```
} catch (InterruptedException ie) {
        System.out.println("Exception caught in main");          }}}
```

paperism
Date:

Threads

8.

```
class MyThread extends Thread {
    long time;
    private volatile boolean running = true;
    MyThread() {
        System.out.println("Default");
    }
    MyThread(String name, long time) {
        super(name);
        this.time = time;
    }
    public void pause() {
        running = false;
    }
    public void run() {
        try {
        while (running) {
            System.out.println(this.getName());
            Thread.sleep(time * 1000);
        } catch (InterruptedException ie) {
            System.out.println("Exception caught in method");
        }
    }
}

class ThreadRunner {
    public static void main(String[] args) {
    MyThread mt1 = new MyThread("BMS", 10);
    MyThread mt2 = new MyThread("CSE", 2);
```

Output: