UNIVERSITY INSTITUTE OF ENGINEERING

Department of Computer Science & Engineering

Subject Name:

Subject Code: 20CSP 321

Submitted to:	**Submitted by:**
Er. Kirat Kaur	Name: Rajiv Paul
	UID:20BCS1812
	Section:20BCS_WM-702
	Group: A

INDEX

Ex. No	List of Experiments	Date	Conduct (MM: 12)	Viva (MM: 10)	Record (MM: 8)	Total (MM: 30)	Remarks/ Signature
1.1							
1.2							
1.3	Calculate interest based on the type of the account and the status of the account holder. The rates of interest changes according to the amount (greater than or less than 1 crore), age of account holder (General or Senior citizen) and number of days if the type of account is FD or RD.						
2.1							
2.2							
2.3							
2.4							
3.1							
3.2							
3.3							

Experiment 3

Student Name: Rajiv Paul **UID:**20BCS1812
Branch:CSE **Section/Group:**702A
Semester: 5th **Date of Performance:** 6/9/2022
Subject Name:PBLJ Lab **Subject Code:** 20CSP 321

1. **Aim:**

To calculate interest based on account type and status of account holder.

2. Requirements:

Software:

IntelliJ IDEA, JDK,MacOs,Netbeans

Hardware:

Macbook(Laptop)

Ram:4GB(Minimum)

Processor: M1

3. Code:

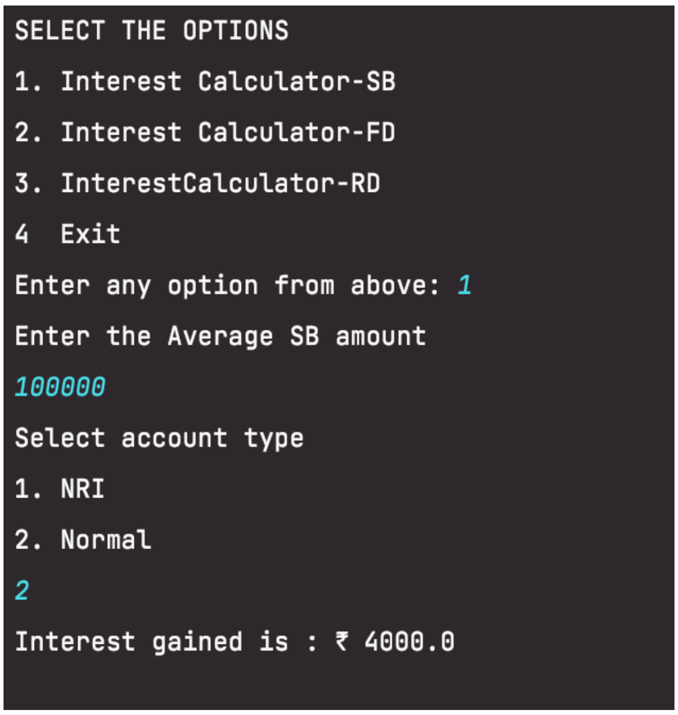
```
package com.PBLJ.Project3;

import java.util.*;

public class InterestCalculator {

   public static void main(String[] args) {

      Scanner sc = new Scanner(System.in);
      System.out.println("SELECT THE OPTIONS " + "\n1." + " Interest
Calculator-SB" + " \n2." + " Interest Calculator-FD"
            + "\n3." + " InterestCalculator-RD" + "\n4 " + " Exit");
      System.out.print("Enter any option from above: ");
      int choice = sc.nextInt();
      switch (choice) {
         case 1:
            SBaccount sb = new SBaccount();
            try {
               System.out.println("Enter the Average SB amount ");
               double amount = sc.nextDouble();
```

```
                    System.out.println("Interest gained is : ₹ " +
sb.calculateInterest(amount));

                } catch (InvalidAmountException e) {
                    System.out.println("Exception : Invalid amount");
                }
                break;

            case 2:
                try {
                    FDaccount fd = new FDaccount();
                    System.out.println("Enter the FD Amount");
                    double fAmount = sc.nextDouble();
                    System.out.println("Interest gained is: ₹ " +
fd.calculateInterest(fAmount));
                } catch (InvalidAgeException e) {
                    System.out.println("Invalid Age Entered");
                } catch (InvalidAmountException e) {
                    System.out.println("Invalid Amount Entered");

                } catch (InvalidDaysException e) {
                    System.out.println("Invalid Days Entered");

                }

                break;
            case 3:
                try {
                    RDaccount rd = new RDaccount();
                    System.out.println("Enter the RD amount");
                    double Ramount = sc.nextDouble();
                    System.out.println("Interest gained is: ₹ " +
rd.calculateInterest(Ramount));
                } catch (InvalidAgeException e) {
                    System.out.println("Invalid Age Entered");
                } catch (InvalidAmountException e) {
                    System.out.println("Invalid Amount Entered");

                } catch (InvalidMonthsException e) {
                    System.out.println("Invalid Days Entered");
                }

                break;

            case 4:
                System.out.println("Thank you for using my calculator !");
            default:
                System.out.println("Wrong choice");
```

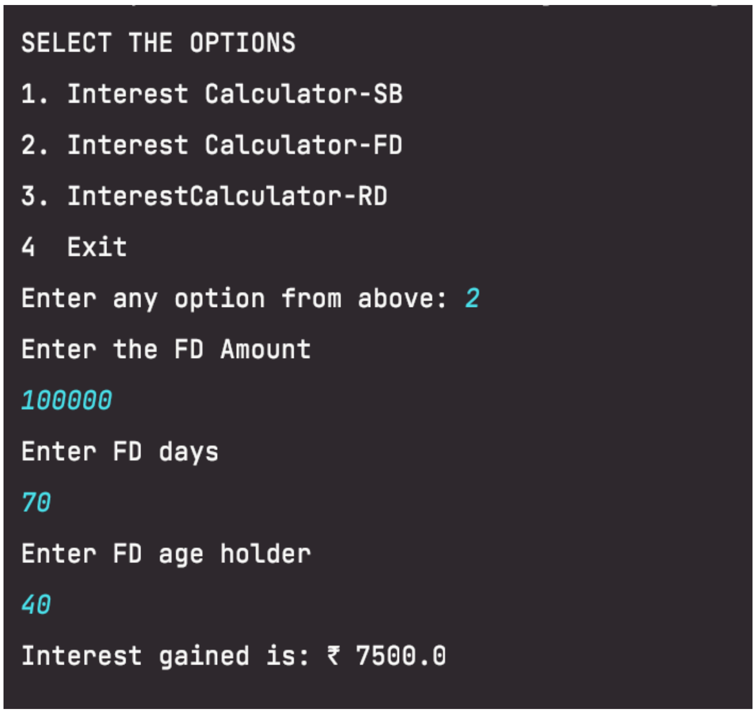
```
        }
    }

}


// Account class

public abstract class Account {
    double interestRate;
    double amount;
    abstract double calculateInterest(double amount)
            throws
InvalidMonthsException,InvalidAgeException,InvalidAmountException ,Invalid
DaysException;
}

//SBaccount class

public class SBaccount extends Account {
    double SBamount , SbInterestRate, interest;
    Scanner SBScanner = new Scanner(System.in);

    @Override
    double calculateInterest(double amount) throws InvalidAmountException{
        this.SBamount = amount;
        if(SBamount < 0 ){
            throw new InvalidAmountException();
        }
        System.out.println("Select account type \n1. NRI \n2. Normal ");
        int accountChoice = SBScanner.nextInt();
        switch (accountChoice) {
            case 1:
                SbInterestRate = .06;
                break;
            case 2:
                SbInterestRate = .04;
                break;
            default:
                System.out.println("Please choose right account again");

        }
        return amount * SbInterestRate;

    }

}

// FDaccount class
```

```
public class FDaccount extends Account {

    double FDinterestRate;
    double FDAmount;
    int noOfDays;
    int ageOfACHolder;
    double General, SCitizen;
    Scanner FDScanner = new Scanner(System.in);

    @Override
    double calculateInterest(double amount) throws
InvalidAgeException,InvalidAmountException ,InvalidDaysException {
        this.FDAmount = amount;

        System.out.println("Enter FD days");
        noOfDays = FDScanner.nextInt();
        System.out.println("Enter FD age holder ");
        ageOfACHolder = FDScanner.nextInt();
        if (amount < 0) {
            throw new InvalidAmountException();
        }
        if(noOfDays<0){
            throw new InvalidDaysException();
        }
        if(ageOfACHolder<0){
            throw new InvalidAgeException();
        }
        if (amount < 10000000) {
            if (noOfDays >= 7 && noOfDays <= 14) {
                General = 0.0450;
                SCitizen = 0.0500;
            } else if (noOfDays >= 15 && noOfDays <= 29) {
                General = 0.0470;
                SCitizen = 0.0525;
            } else if (noOfDays >= 30 && noOfDays <= 45) {
                General = 0.0550;
                SCitizen = 0.0600;
            } else if (noOfDays >= 45 && noOfDays <= 60) {
                General = 0.0700;
                SCitizen = 0.0750;
            } else if (noOfDays >= 61 && noOfDays <= 184) {
                General = 0.0750;
                SCitizen = 0.0800;
            } else if (noOfDays >= 185 && noOfDays <= 365) {
                General = 0.0800;
                SCitizen = 0.0850;
            }
            FDinterestRate = (ageOfACHolder < 50) ? General : SCitizen;
        } else {
            if (noOfDays >= 7 && noOfDays <= 14) {
                interestRate = 0.065;
            } else if (noOfDays >= 15 && noOfDays <= 29) {
                interestRate = 0.0675;
            } else if (noOfDays >= 30 && noOfDays <= 45) {
                interestRate = 0.00675;
            } else if (noOfDays >= 45 && noOfDays <= 60) {
                interestRate = 0.080;
            } else if (noOfDays >= 61 && noOfDays <= 184) {
                interestRate = 0.0850;
            } else if (noOfDays >= 185 && noOfDays <= 365) {
                interestRate = 0.10;
            }

        }

        return FDAmount * FDinterestRate;
    }
}

//RDaccount class

public class RDaccount extends Account {

    double RDInterestRate;
    double RDamount;
    int noOfMonths;
    double monthlyAmount;
    double General, SCitizen;
    Scanner RDScanner = new Scanner(System.in);

    @Override
    double calculateInterest(double Ramount) throws
InvalidMonthsException,InvalidAmountException ,InvalidAgeException {
        this.RDamount = Ramount;
        System.out.println("Enter RD months");
        noOfMonths = RDScanner.nextInt();
        System.out.println("Enter RD holder age");
        int age = RDScanner.nextInt();
        if (RDamount < 0) {
            throw new InvalidAmountException();
        }
        if(noOfMonths<0){
            throw new InvalidMonthsException();
        }
        if(age<0){
            throw new InvalidAgeException();
        }
        if (noOfMonths >= 0 && noOfMonths <= 6) {
            General = .0750;
            SCitizen = 0.080;
        } else if (noOfMonths >= 7 && noOfMonths <= 9) {
            General = .0775;
            SCitizen = 0.0825;
        } else if (noOfMonths >= 10 && noOfMonths <= 12) {
            General = .0800;
            SCitizen = 0.0850;
        } else if (noOfMonths >= 13 && noOfMonths <= 15) {
            General = .0825;
            SCitizen = 0.0875;
        } else if (noOfMonths >= 16 && noOfMonths <= 18) {
            General = .0850;
            SCitizen = 0.0900;
        } else if (noOfMonths >= 22) {
            General = .0875;
            SCitizen = 0.0925;
        }
        RDInterestRate = (age < 50) ? General : SCitizen;
        return RDamount * RDInterestRate;

    }

}

// InvalidAgeException class

public class InvalidAgeException extends Exception{

}

// InvalidAmountException class

public class InvalidAmountException extends Exception{

}

// InvalidDaysException class

public class InvalidDaysException extends Exception{

}

// InvalidMonthsException class

public class InvalidMonthsException extends Exception{

}
```

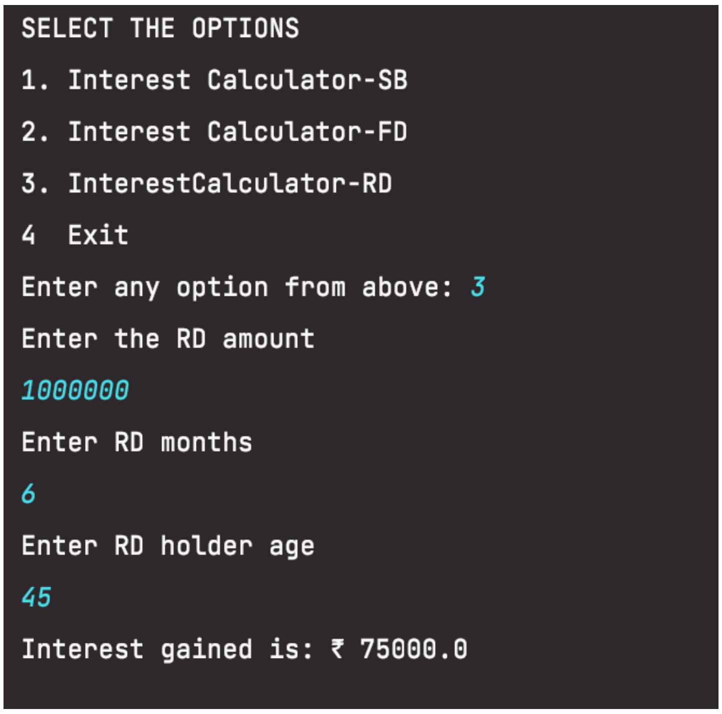
4. Output:



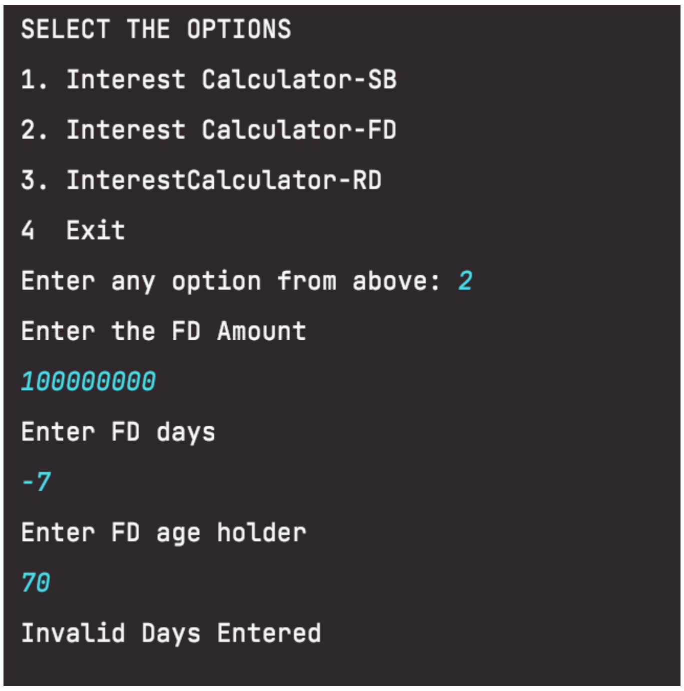
```
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. InterestCalculator-RD
4  Exit
Enter any option from above: 1
Enter the Average SB amount
100000
Select account type
1. NRI
2. Normal
2
Interest gained is : ₹ 4000.0
```



```
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. InterestCalculator-RD
4  Exit
Enter any option from above: 2
Enter the FD Amount
100000
Enter FD days
70
Enter FD age holder
40
Interest gained is: ₹ 7500.0
```



```
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. InterestCalculator-RD
4  Exit
Enter any option from above: 3
Enter the RD amount
1000000
Enter RD months
6
Enter RD holder age
45
Interest gained is: ₹ 75000.0
```



```
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. InterestCalculator-RD
4  Exit
Enter any option from above: 2
Enter the FD Amount
100000000
Enter FD days
-7
Enter FD age holder
70
Invalid Days Entered
```

Learning outcomes (What I have learnt):

1. Learnt about classes and its methods.

2. Learnt about exception handling.

3. Learnt about try catch block.

4. Leant abstract class in Java

5. Learnt about the difference between abstract and non abstract classes.