Experiment:1.3

Student Name: Jitesh Kumar **UID: 20BCS2334**
Branch: CSE **Section/Group: 903/A**
Semester: 5th
Subject Name: PROJECT BASED LEARNING IN JAVA LAB
Subject Code: 20CSP-321

Aim :- Create a application to calculate interest for FDs, RDs based on certain conditions using inheritance.

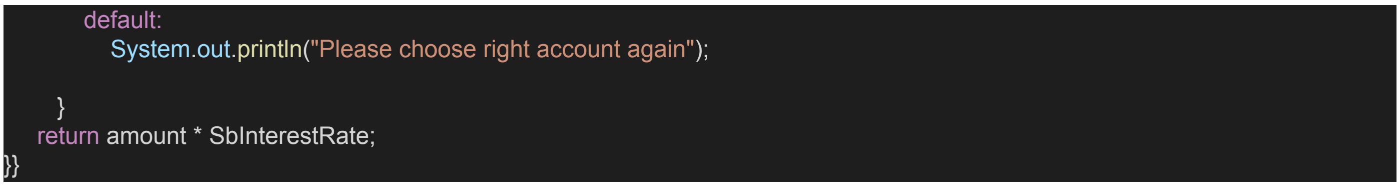
S/W Requirement:- Eclipse_IDE - (Java) , Netbean..

Source Code :-

```
import java.util.Scanner;

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");
    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 : Rs " + 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: Rs " + 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");

        }
```

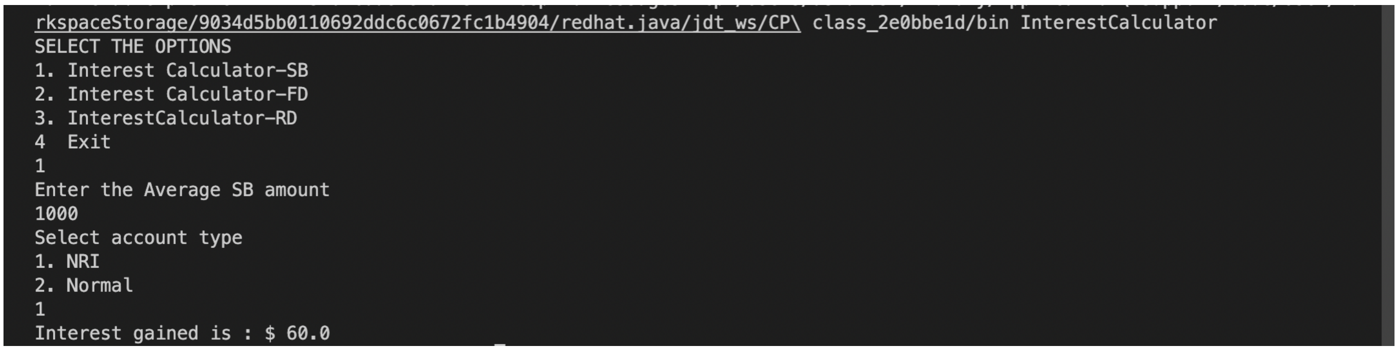
```
                break;
            case 3:
                try {
                    RDaccount rd = new RDaccount();
                    System.out.println("Enter the RD amount");
                    double Ramount = sc.nextDouble();
                    System.out.println("Interest gained is: Rs " + 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("DO YOU WANT TO CALCULATE AGAIN ????" + " "
                        + "RUN AGAIN THE PROGRAM");
            default:
                System.out.println("Wrong choice");

        }
        sc.close();
    }

}

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

 class FDaccount extends Account {

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

    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;
    }
}
 class InvalidAgeException extends Exception{}

 class InvalidAmountException extends Exception{}

 class InvalidDaysException extends Exception{}

 class InvalidMonthsException extends Exception{}

 class RDaccount extends Account {

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

```
  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;

  }

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


  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;
}}
```

Output-:



```
rkspaceStorage/9034d5bb0110692ddc6c0672fc1b4904/redhat.java/jdt_ws/CP\ class_2e0bbe1d/bin InterestCalculator
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. InterestCalculator-RD
4  Exit
1
Enter the Average SB amount
1000
Select account type
1. NRI
2. Normal
1
Interest gained is : $ 60.0
```

Learning outcomes (What I have learnt):

1. Learnt about abstract classes in java.

2. Learnt about inheritance.

3. Learnt about exception handling.