Experiment: 3

Student Name: Jashanpreet Singh
UID: 22BCS11724
Branch: CSE
Section/Group: 22IOT_635 - B
Semester: 6th
Date of Performance:
Subject Name: JAVA
Subject Code: 21CSH-314

1. **Aim:** Create an application to calculate interest for FDs, RDs based on certain conditions using inheritance.

2. **Objective:** Write a program to create an application to make an Account holders list and calculate interest for FDs, RDs based on certain conditions using inheritance.

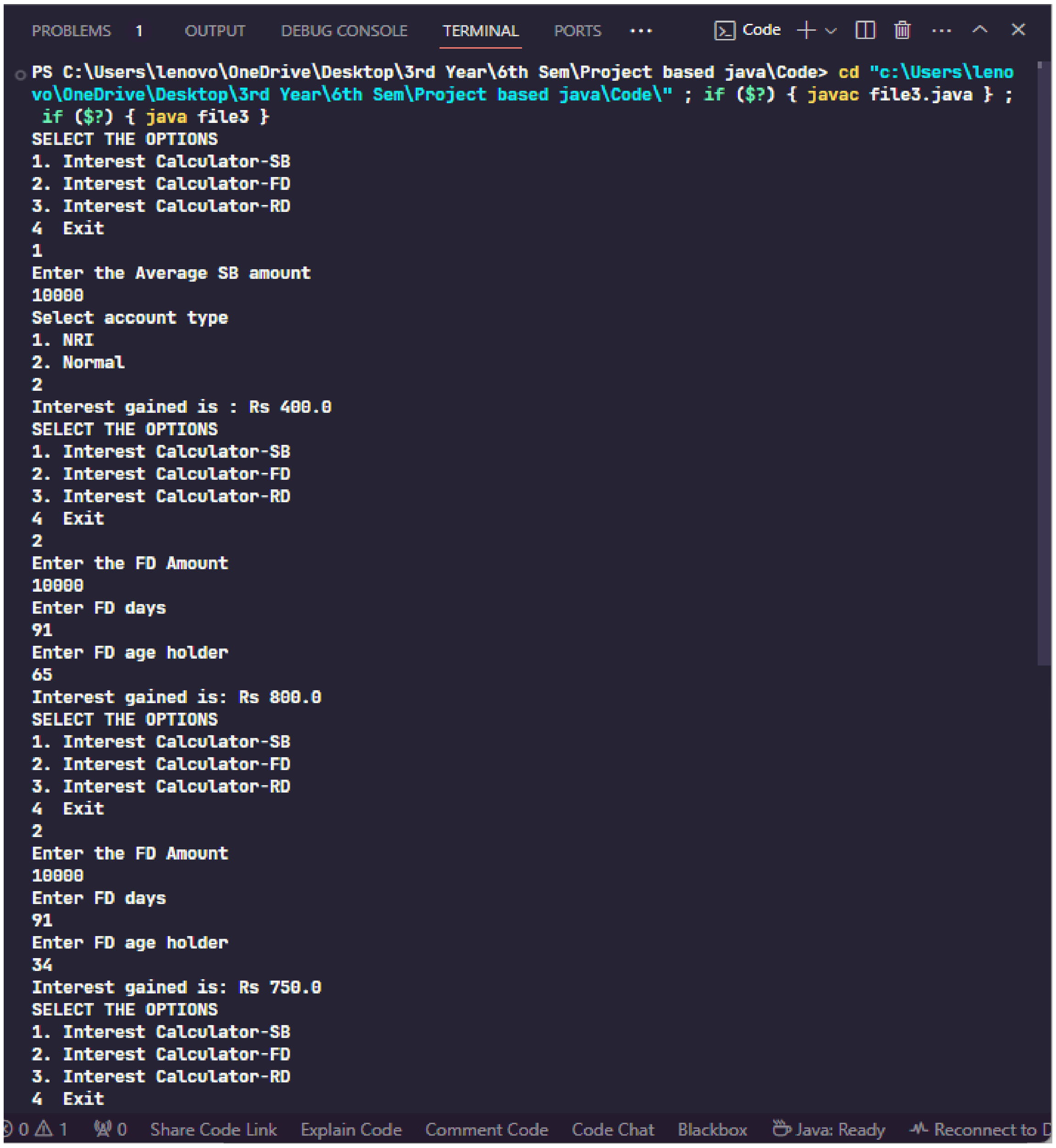
3. **Code:**

```
import java.util.Scanner;

class InvalidAgeException extends Exception {
}

class InvalidAmountException extends Exception {
}

class InvalidDaysException extends Exception {
}

class InvalidMonthsException extends Exception {
}

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 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 = 0.0750;
            SCitizen = 0.080;
        } else if (noOfMonths >= 7 && noOfMonths <= 9) {
            General = 0.0775;
            SCitizen = 0.0825;
        } else if (noOfMonths >= 10 && noOfMonths <= 12) {
            General = 0.0800;
            SCitizen = 0.0850;
        } else if (noOfMonths >= 13 && noOfMonths <= 15) {
            General = 0.0825;
            SCitizen = 0.0875;
        } else if (noOfMonths >= 16 && noOfMonths <= 18) {
            General = 0.0850;
            SCitizen = 0.0900;
        } else if (noOfMonths >= 22) {
            General = 0.0875;
            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 = 0.06;
                break;
            case 2:
                sbInterestRate = 0.04;
                break;
            default:
                System.out.println("Please choose right account again");
                break;
        }
        return amount * sbInterestRate;
    }
}
```

```
public class file3 {
    public static void main(String[] args) {
        boolean val = true;
        Scanner sc = new Scanner(System.in);
        while (val) {
            System.out.println("SELECT THE OPTIONS " + "\n1." + " Interest Calculator-SB" + " \n2."
+ " Interest Calculator-FD" + "\n3." + " Interest Calculator-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 entered.");
                    }
                    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 Months Entered");
                    }
                    break;
                case 4:
                    val = false;
                    System.out.println("Exiting the program.");
                    break;
                default:
                    System.out.println("Wrong choice");
                    break;
            }
        }
        sc.close();
    }
}
```

4. OUTPUT:



```
PS C:\Users\lenovo\OneDrive\Desktop\3rd Year\6th Sem\Project based java\Code> cd "c:\Users\leno
vo\OneDrive\Desktop\3rd Year\6th Sem\Project based java\Code\" ; if ($?) { javac file3.java } ;
 if ($?) { java file3 }
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. Interest Calculator-RD
4  Exit
1
Enter the Average SB amount
10000
Select account type
1. NRI
2. Normal
2
Interest gained is : Rs 400.0
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. Interest Calculator-RD
4  Exit
2
Enter the FD Amount
10000
Enter FD days
91
Enter FD age holder
65
Interest gained is: Rs 800.0
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. Interest Calculator-RD
4  Exit
2
Enter the FD Amount
10000
Enter FD days
91
Enter FD age holder
34
Interest gained is: Rs 750.0
SELECT THE OPTIONS
1. Interest Calculator-SB
2. Interest Calculator-FD
3. Interest Calculator-RD
4  Exit
```

0 ⚠ 1   0   Share Code Link   Explain Code   Comment Code   Code Chat   Blackbox   Java: Ready   Reconnect to D

5. Learning Outcomes:

- Recognizing the use of classes and objects to structure and organize code.
- Gaining proficiency in handling exceptions to manage unexpected situations.
- Implementing custom exception classes to enhance code robustness.