iChat Report

Student Name : ANKESH .

Exam Name : JAVA EXCEPTION HANDLING - WRITE

Exam Type : WRITEN **Exam Date :** 2024-08-20

Total Score : 20 **Achieved Score :** 7

Question : Imagine you are designing a simple banking system. Write a Java program that simulates a bank account. Implement the following: Task 1: Create a custom exception, InsufficientFundsException, that is thrown when an attempt is made to withdraw more money than the account balance. Task 2: Implement a method void performTransaction(double amount) that allows both deposits and withdrawals. If a withdrawal exceeds the account balance, throw the InsufficientFundsException and ensure the transaction is rolled back, leaving the account balance unchanged. Task 3: Demonstrate the usage of your program with multiple transactions, including scenarios where exceptions are thrown and caught.?	**Score : 5/10**

Answer :

```
public class InsufficientFundsException extends RuntimeException{
    public InsufficientFundsException() {
        super();
    }
    public InsufficientFundsException(String message) {
        super(message);
    }
    @Override
    public String getMessage() {
        return super.getMessage();
    }
    @Override
    public StackTraceElement[] getStackTrace() {
        return super.getStackTrace();
    }
}
public class Bank {
    double balance;
    public Bank(double balance) {
        this.balance = balance;
    }
    private void deposit(double amount){
        this.balance += amount;
    }
    private double withdraw(double amount)throws InputMismatchException {
        if(this.balance < amount){
            throw new InsufficientFundsException("Insufficient Funds");
        }
        this.balance -= amount;
        return this.balance;
    }
    public void performTransaction(double amount, int tnxType)throws InputMismatchException
{
        if (tnxType == 1){
            this.deposit(amount);
        }
        if(tnxType == 2){
            this.withdraw(amount);
        }
    }
}
public class Main {
```

Suggestion :

The code provided does not fully address the specified tasks. The custom exception InsufficientFundsException is correctly implemented, but the performTransaction method does not handle the exception properly and does not ensure the transaction is rolled back if an exception occurs. Additionally, the main method does not demonstrate catching the exception. Update the performTransaction method to properly handle the exception by rolling back the transaction if an InsufficientFundsException is thrown. Also, modify the main method to include try-catch blocks to catch and handle the InsufficientFundsException..

Question : Write a Java program that reads data from a text file containing integers. Implement exception handling to deal with the following scenarios: Scenario 1: If the file is not found, catch the exception and print a user-friendly error message. Scenario 2: If there is a non-integer value in the file, catch the exception and skip that particular value, continuing with the next one.?

Score : 2/10

Answer :

```
public class Main {
    public static void main(String[] args) {
        File obj = new File("input.txt");
        Scanner sc;
        try{
            sc = new Scanner(obj);
            while(sc.hasNext()){
                try{
                    System.out.println(sc.nextInt());
                }catch (InputMismatchException inp){
                    System.out.println(inp);
                    break;
                }
            }
        }catch (FileNotFoundException fnfe){
            try{
                obj.createNewFile();
            }catch (IOException io){
                System.out.println(io);
            }
        }
    }
}.
```

Suggestion :

The code provided does not fully address the specified task. It lacks handling non-integer values in the file as required. Additionally, creating a new file when the input file is not found is not necessary based on the task requirements. Consider revising the code to handle non-integer values by catching the InputMismatchException and skipping the problematic value. Also, remove the unnecessary file creation part. Ensure to continue reading the next value after catching the InputMismatchException..