Task 7: Exception Handling

1. To create custom exception : TrackingNumberNotFoundException which throw this exception when user try to withdraw amount or transfer amount to another account

Creation of Custom Exception Steps:

1. Define the exception class
2. Throw it in a method
3. Handle it using try-catch-finally

TrackingNumberNotFoundException

1. Define the exception class

```
package Exception;

public class TrackingNumberNotFoundException extends Exception{

   public TrackingNumberNotFoundException(String message)
   {
      super(message);
   }
}
```

2. Throw it in a method

```
package Exception;

import Entities.Courier;

public class Withdraw_Transfer {
   public static void Withdraw(double balance, double amount,
Courier courier)throws TrackingNumberNotFoundException {

      String trackingNumber= courier.getTrackingNumber();
      if(amount>balance)
      {
```

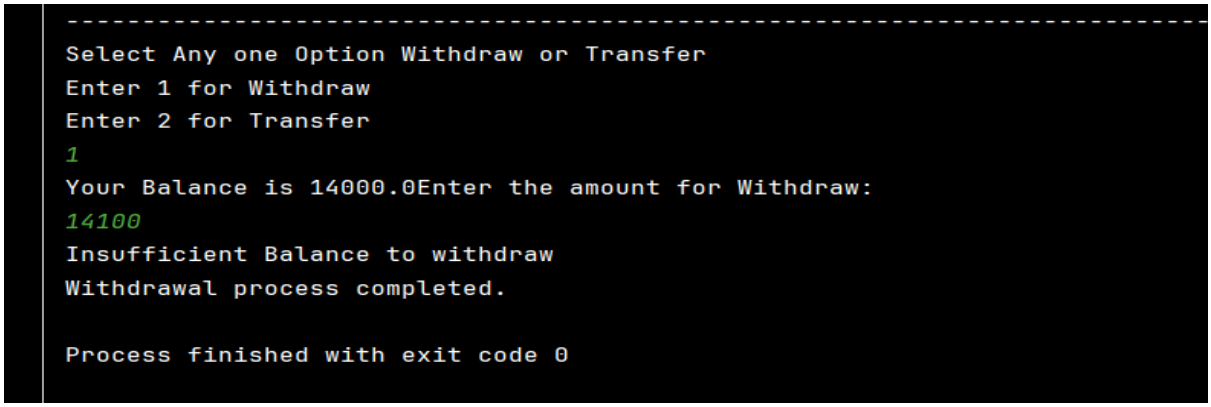
```
            throw new TrackingNumberNotFoundException("Insufficient
Balance to withdraw");
        }
        else
        {
            balance -= amount;
            System.out.println("Withdrawal successful! New balance: " +
balance);
        }
    }

    public static void transfer(double balance,double amount)throws
TrackingNumberNotFoundException{
        if (amount > balance)
        {
            throw new TrackingNumberNotFoundException("Insufficient
balance for transfer!");
        }
        else
        {
            balance -= amount;
            System.out.println("Transfer successful! New balance: " +
balance);
        }
    }
}
```

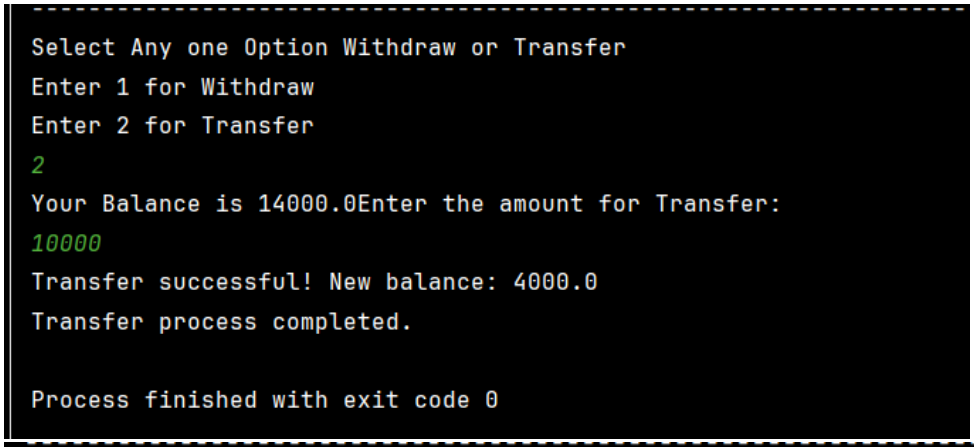
3. Handle it using try-catch-finally

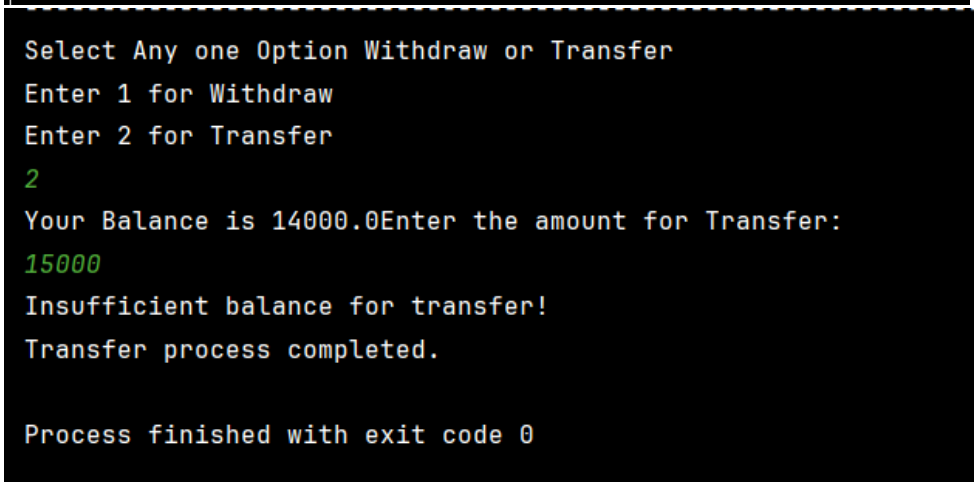
```
System.out.println("Select Any one Option Withdraw or
Transfer\nEnter 1 for Withdraw\nEnter 2 for Transfer");
    int num=sc.nextInt();
    double balance = 14000;


    //Withdrawing
    if(num==1) {
```

```
        System.out.println("Your Balance is " + balance + "Enter the
amount for Withdraw: ");
        double amount = sc.nextDouble();
        try {
            Withdraw_Transfer.Withdraw(balance, amount, courier);
        } catch (TrackingNumberNotFoundException e) {
            System.out.println(e.getMessage());
        }finally {
            System.out.println("Withdrawal process completed.");
        }
    }
    //Transfering
    else {
        System.out.println("Your Balance is " + balance + "Enter the
amount for Transfer: ");
        double amount1 = sc.nextDouble();
        try {
            Withdraw_Transfer.transfer(balance, amount1);

        } catch (TrackingNumberNotFoundException e) {
            System.out.println(e.getMessage());
        } finally {
            System.out.println("Transfer process completed.");
        }
    }
}
```

OUTPUT:







InvalidEmployeeIdException

2. To create custom exception: InvalidEmployeeIdException throw this exception when id entered for the employee not existing in the system

1. Define the exception class

package Exception**;**

public class InvalidEmployeeIdException extends Exception{

```
    public InvalidEmployeeIdException(String message)
    {
        super(message);
    }}
```

2. Throw it in a method

```
package Exception;

import java.util.Set;

public class IDCheck {

    public static void checkEmployeeid(int empid,Set<Integer>
employeeidset) throws InvalidEmployeeIdException
    {

        if(!employeeidset.contains(empid))
        {
            throw new InvalidEmployeeIdException("Entered Employee
Id is not an Existing One, Kindly check the ID");
        }
        else
        {
            System.out.println("Entered ID is an Existing one and Valid");
        }

    }
}
```

3. Handle it using try-catch-finally

```
System.out.println("Enter Employee ID to fetch Employee Details");
int empid=sc.nextInt();
try
```

```
{
 checkEmployeeid(empid,employeeidset);
}
catch (InvalidEmployeeIdException e) {
   System.out.println(e.getMessage());
}finally {
   System.out.println("Employee ID validation process completed.");
}
```

OUTPUT: