Core Java Assignments

Day #3: Packages and Exception Handling

Objective: At the end of the assignments, participants will be able to understand and create their own packages and also handle Exceptions

Concept: Packages

Estimated time: 1.5 Hours

1. Create a package org.itp.hyd .

   For e.g., org.itp.hyd

   Now create a Greeter class in this package having the following features:

```
Attributes:
        name   string   //indicates name of the person to be greeted
Member functions:
        Greeter(aName)        //constructor to initialize the name of the //person to
                              be greeted by this greeter.
        sayHello()            //returns a hello message with the name of the
                              //person initialized earlier.
        sayGoodBye() //bids goodbye to the person named earlier.
```

   Create another class in the same package called Advisor that has the following features:

```
Attributes:
        message        string[5]        //contains five advice messages
Member functions:
        Advisor()             //default constructor to initialize an array of //strings
                              with atleast five advice messages
        getAdvice()           //randomly selects an advice from the available //list
                              of messages and returns it to the caller of //this
                              method
```

   Outside the package, from your working directory, create a class GreeterTest that constructs Greeter objects for all command-line arguments and prints out the results of calling sayHello().

   The program should then display an advice and finally bid goodbye to each of the persons/entities in reverse order of the names entered at the command line.

For e.g.,

```
java   GreeterTest    Mars    Venus
```

then the program should print

```
Hello, Mars!
Hello, Venus!
Advice: Never say No
Goodbye Venus!
Goodbye Mars!
```

2. Create a package esg.itp.shape containing the following classes and interfaces.

An interface Polygon containing the members as given below:

area	float
perimeter	float

void calcArea( );	abstract method to calculate area of a particular polygon given its dimensions
void calcPeri( );	abstract method to calculate perimeter of a particular polygon given its dimensions
void display( );	method to display the area and perimeter of the given polygon

Create a class Square that implements Polygon and has the following member:

side	float

Square(float s);	constructor to initialize side of square

Create another class Rectangle that implements Polygon and has the following member:

length	float
breadth	float

Rectangle(int len, int bre);	constructor to initialize length and breadth of a rectangle

Outside the package, create a class that imports the above package an instantiates an object of the Square class and an object of the Rectangle class.
Call the above methods on each of the classes to calculate the area and perimeter given the side and the length/breadth of the Square class and the Rectangle class respectively.

Concept : Exception Handling

Estimated time: 2 Hours

3. Create a class called CalcAverage that has the following method:

    public double avgFirstN(int N)

This method receives an integer as a parameter and calculates the average of first N natural numbers. If N is not a natural number, throw an exception IllegalArgumentException with an appropriate message.

4.Create a class Number having the following features:

Attributes

int	first number	
int	second number	
double	result	stores the result of arithmetic operations performed on a and b

Member functions

Number(x, y)	constructor to initialize the values of a and b
add()	stores the sum of a and b in result
sub()	stores difference of a and b in result
mul()	stores product in result
div()	stores a divided by b in result

Test to see if b is 0 and throw an appropriate exception since division by zero is undefined.

Display a menu to the user to perform the above four arithmetic operations.

6.Create a class BankAccount having the members as given below:

accNo integer
custName string
accType string (indicates 'Savings' or 'Current')
balance float

Include the following methods in the BankAccount class:

void deposit(float amt);
void withdraw(float amt);
float getBalance();

deposit(float amt) method allows you to credit an amount into the current balance. If amount is negative, throw an exception **NegativeAmount** to block the operation from being performed.

withdraw(float amt) method allows you to debit an amount from the current balance. Please ensure a minimum balance of Rs. 1000/- in the account for savings account and Rs. 5000/- for current account, else throw an exception **InsufficientFunds** and block the withdrawal operation. Also throw an exception **NegativeAmount** to block the operation from being performed if the amt parameter passed to this function is negative.

getBalance() method returns the current balance. If the current balance is below the minimum required balance, then throw an exception **LowBalanceException** accordingly.

Have constructor to which you will pass, accno, cust_name, acctype and initial balance.
And check whether the balance is less than 1000 or not in case of savings account and less than 5000 in case of a current account. If so, then raise a **LowBalanceException**.
In either case if the balance is negative then raise the **NegativeAmount** exception accordingly.

7. Create a class with following specifications.

Class Emp

empId	int
empName	string
designation	string
basic	double

hra double readOnly

Methods

printDET()
calculateHRA()
printDET() methods will show details of the EMP.

calculateHRA() method will calculate HRA based on basic.

There will 3 designations supported by the application.

If designation is “Manager” - HRA will be 10% of BASIC
if designation is “Officer” - HRA will be 12% of BASIC
if designation is “CLERK” - HRA will be 5% of BASIC
Have constructor to which you will pass empId, name, designation, basic .
And checks whether the BASIC is less than 500 or not. If it is less than 500 raise a custom Exception as given below

Create LowSalException class with proper user message to handle BASIC less than 500.

8. Create a class USERTRAIL with following specifications.
val1, val2 type int

Methods

boolean show () will check if val1 and val2 are greater or less than Zero

Have constructor which will val1, val2 and check whether if it is less than 0 then raise a custom Exception (name: Illegal value exception.)