**Working with Abstraction: Lab**

This document defines the lab for the ["Java Advanced" course @ Software University](https://softuni.bg/modules/59/java-advanced). Please submit your solutions (source code) of all below described problems in [Judge](https://judge.softuni.bg/Contests/1575/Working-with-Abstraction-Lab).

* **Rhombus of Stars**

Create a program that reads a **positive** **integer** **n** as input and prints on the console a **rhombus** with size **n**:

**Examples**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 1 | \* | 2 | \*  \* \*  \* | 3 | \*  \* \*  \* \* \*  \* \*  \* |

**Hint**

Create a **printRow()** method to easily reuse code.

* **Point in Rectangle**

Create a class **Point** and a class **Rectangle**. The **Point** should hold **coordinates X** and **Y** and the **Rectangle** should hold 2 **Points** – its **bottom** **left** and **top** **right** corners. In the **Rectangle** class, you should implement a **contains(Point point)** method that returns **true** or **false**, based on **whether** the **Point** given as an **attribute** is **inside** or **outside** of the **Rectangle** object. Points **on** **the** **side** of a Square are considered **inside**.

**Input**

* On the first line read the **coordinates** of the **bottom** **left** and **top** **right** corner of the **Rectangle** in the format: **"{bottomLeftX} {bottomLeftY} {topRightX} {topRightY}"**.
* On the second line, read an integer **N** and on the next **N** lines, read the **coordinates** of **points**.

**Output**

* For each point, print out the result of the **Contains()** method.

**Examples**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 0 0 3 3  5  0 0  0 1  4 4  5 3  1 2 | true  true  false  false  true |  | 2 -3 12 3  4  8 -1  11 3  1 1  2 4 | true  true  false  false |  | 5 8 12 15  6  0 0  5 8  12 15  8 15  7 15  8 12 | false  true  true  true  true  true |

* **Student System**

You are given a **working** **project** for a small **Student** **System**, but the code is very poorly organized. Break up the code **logically** into **smaller** **functional** **units** – **methods** and **classes** and don’t break the functionality.

The program supports the following commands:

* **"Create {studentName} {studentAge} {studentGrade}"** – creates a new student and adds them to the repository.
* **"Show {studentName}"** – prints on the console information about a student in the format:  
  **"{studentName} is {studentAge} years old. {commentary}."**, where the **commentary** is based on the student’s grade.
* **"Exit"** – closes the program.

**Do not** add any **extra validation** or **functionality** to the app!

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| Create Peter 20 5.50  Create Maria 18 4.50  Create George 25 3  Show Peter  Show Maria  Exit | Peter is 20 years old. Excellent student.  Maria is 18 years old. Average student. |
| Create Teo 19 2.00  Show Sam  Show Teo  Create Sam 20 3.00  Show Teo  Show Sam  Exit | Teo is 19 years old. Very nice person.  Teo is 19 years old. Very nice person.  Sam is 20 years old. Very nice person. |

* **Hotel Reservation**

Create a class **PriceCalculator** that calculates the total price of a holiday, given the **price** **per** **day**, **number** **of** **days**, the **season,** and a **discount** **type**.The **discount** **type** and **season** should be an **enum**.

Use the class in your **main()** method to read input and **print** on the console the **price** of the **whole** **holiday**.

The price per day will be multiplied depending on the season by:

* **1** during **Autumn**
* **2** during **Spring**
* **3** during **Winter**
* **4** during **Summer**

The discount is applied to the total price and is one of the following:

* **20**% for VIP clients - **VIP**
* **10**% for clients, visiting for a second time - **SecondVisit**
* **0**% if there is no discount - **None**

**Input**

On a **single** **line** you will receive all the **information** about the **reservation** in the format:  
**"{pricePerDay} {numberOfDays} {season} {discountType}"**, where:

* The price per day will be a valid decimal in the range **[0.01…1000.00]**.
* The number of days will be a valid integer in the range **[1…1000]**.
* The season will be one of **Spring**, **Summer**, **Autumn**, **Winter**.
* The discount will be one of **VIP**, **SecondVisit**, **None**.

**Output**

On a **single** **line**, print the **total** **price** of the **holiday**, rounded to **2** **digits** after the decimal separator.

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 50.25 5 Summer VIP | 804.00 |
| 40 10 Autumn SecondVisit | 360.00 |
| 120.20 2 Winter None | 721.20 |