CS 251 InLab7

Java Basics

Instructions

The submissions will be **autograded**. Please read the problem statement carefully.

Your implementation will be run against multiple test cases and marks will be provided based on the fraction of test cases passed.

Commands to run Main.java: (Ensure that you are in the directory just outside Q$) for Q1 and Q2. For Q3, replace Main by PollutionCheck.

```
javac  Q$/*.java  &&  java Q$/Main
```

Submission directory must be named - <roll_no>_inlab7

If the Directory Tree doesn't match, you will get zero.

CS 251 InLab7


```
.
├── Q1
│   └── Matrix.java
├── Q2
│   └── Main.java
└── Q3
    ├── Car.java
    ├── PollutionCheck.java
    ├── Truck.java
    └── Vehicle.java

3 directories, 6 files
```

Q1 Matrix Algebra (20 point)

Pulkit, the Meth Gawd, requires a matrix calculator (in Java :P) for making through ICPC regionals this year. Ofcourse, he is too lazy and perhaps too high to implement

Implement the constructors and member functions of **class Matrix** as provided in the supplement file **Matrix.java**. **Make sure you make changes in the file only where it has been mentioned. Read the comments carefully.**

Constructors

1. Matrix (int n, int m): Create an n x m matrix with all entries equal to zero.

2. Matrix (int n, int m, int v): Create an n x m matrix with all entries equal to v.

Member Functions

1. int getrows(): Return number of rows.

2. int getcols(): Return number of columns.

3. int getelem(int i, int j): Return (i,j)th element of the matrix.

4. void setelem(int i, int j, int v): set (i,j)th element of the

obtained by matrix addition of A and B.

7. static Matrix matmul(Matrix A, Matrix B): return Matrix

(A x B) obtained by matrix multiplication of A and B .

To know about static methods, click here!

NOTE: Return a zero matrix of size 1x1 if addition or multiplication is not valid.
NOTE:

1. **Input arguments of getelem and setelem are 0-indexed.**
2. **In cases of possible errors/size mismatch the return values are mentioned in comments.**
3. Use **Main.java** to test your implementation. You don't have to submit it.
    a. Commands to run Main.java: (Ensure that you are in directory just outside Q1)
    javac Q1/*.java && java Q1/Main

Q2 Forbidden For (30 point)

Churning grass in the hot summer, Vaibhav the Cow wonders if the grass squares itself on chewing, how much satisfaction he would achieve. Since he can't run loops in

code.
Every time the keywords for, while is found in your code the auto grader will deduct 10 points.

NOTE: you can count number of occurrences of for/while using:

```
grep -c 'for\|while' Main.java
```

1. **0 points for wrong implementation.**
2. **30 - 10*k for correct implementation, where k is the output of above command.**
3. **Do not use keywords 'for' or 'while' in variable names or comments.**

Input format (STDIN) (You can import Scanner for taking input):
The first line contains a single integer **t** - the number of test cases. The description of test cases follows.
The first line of each test case contains a single integer **n** - the number of elements.
The second line of each test case contains **n** integers $a_1, a_2, \ldots\ldots a_n$. - elements of array.
Output format (STDOUT):
For each test case print the sum of squares of array elements.

Sample Case :

INPUT:
2
5
1 2 3 4 5
3
1 -1 0
OUTPUT:

Control (50 marks)

You are given the task of developing software for the Pollution Control Board of Mumbai city. While the environment department of IITB is yet to wake up, your main task would be to generate a pollution report for each registered vehicle in the Mumbai metropolitan area.

Each vehicle in the city is either classified as a **Car** or a **Truck**. Each vehicle is identified by a **unique Registration Number** of the form of "XXX123" (3 alphabets followed by 3 digits). Each vehicle has a **manufacturer name**, and the **name of the owner.** The 2 kinds of vehicles have different thresholds for each pollutant. A vehicle must not exceed the threshold for **any** of the 3 pollutants **CO2, CO,** and **HC**.

You are given a list of vehicles in **vehicles.txt** and a list of pollution reports in **pollution.txt**. Assume that each entry in the **pollution.txt** corresponds to a **valid** vehicle in vehicles.txt. You will be queried with a vehicle registration number, and you must return(print on stdout) whether the vehicle had passed the pollution control test. The queries are provided in **queries.txt**. The thresholds are given below.

For Cars
CO2 <= 15 CO <= 0.5 HC <= 750

---

a. **Attributes :**
   i. **String regNo:** vehicle registration number
   ii. **String manufacturer**: manufacturer name
   iii. **String owner:** name of the owner
   iv. **double CO2**: CO2 emission level
   v. **double CO**: CO emission level
   vi. **double HC**: HC emission level
   vii. **String pollutionStatus**: Can be “PASS”, “FAIL”, “PENDING”, ”NOT REGISTERED”
b. **Functions (public):**
   i. **void checkPollutionStatus()**: self explanatory. Hint: Modify the attribute **pollutionStatus** based on the CO2, CO and HC levels of this vehicle
   ii. **String toString():** returns a string of the form
   “Reg No: <XXX123>
   Manufacturer: <Manufacturer Name>
   Owner: <Owner Name>
   Pollution Status: <pollution status>”

You can add other functions (public/private) and constructors of your own as you feel necessary

2. Design a class **Car** that inherits **Vehicle**:
   a. No additional derived class attributes
   b. No additional derived class methods

Note: You should override some function of the base class to implement the correct functionality

contains your main method.

Input file format

vehicles.txt - each line of this file contains a string of the form
XXX123, Manufacturer Name, Owner Name, Type

pollution.txt - each line of this file contains a string of the form
XXX123, CO2, CO, HC

Queries.txt - Each line of this contains a single string of the form
XXX123

Assume that all registration IDs across all files are of the form XXX123. If a vehicle in the query file is not in the vehicle list, print "NOT REGISTERED" instead. If a vehicle in the query file is in the vehicle list but not in the pollution list, print "PENDING" instead.
Example vehicles.txt
WBA303, Honda, Darth Pivedi, Car
HRY712, Scania, Illogical Apple, Truck
MHA569, Hyundai, Chen Wang, Car

Example pollution.txt
WBA303, 10, 0.1, 1
HRY712, 26, 0.5, 278

Example queries.txt
WBA303
HRY712
MHA569

CS 251 InLab7

The files will be given as command line arguments. Your code will be run as

java Q3/PollutionCheck Q3/<vehicles-file> Q3/<pollution-file> Q3/<queries-file>