Q2

Complete the classes using the Specifications given below. Consider default visibility of classes, data fields, and methods unless mentioned otherwise.

**Specifications**

class definitions:﻿

class Material:

int ﻿mass

int temperature

visibility : public

Material(int mass, int temperature) : Constructor **with** **public** visibility

﻿method definitions:

﻿flowOfHeat() throws **Exception**:

**return** **type**: **String**

visibility: **public**

﻿ checkMaterial() throws **Exception**:

**return** **type** : **String**

visibility : **public**

﻿

**class** MaterialException extends **Exception**:

﻿method definitions:

﻿ MaterialException(**String** msg)

visibility: **public** ﻿

**Task**

**Class** **Material**

**-define all the data members as per the given specifications.**

**-define the constructor with public visibility.**

**-Implement the below methods for this class:**

**-String** **checkMaterial() throws Exception:**

* Write a code to check whether the material is valid or not.
* **throw a MaterialException** if the mass is negative with a message "**Mass cannot be negative**".
* **throw a MaterialException** if the mass is zero with a message "**Mass cannot be zero**"
* if no exception is found return "**Valid material"**

**-String** **flowOfHeat() throws Exception:**

* If **checkMaterial()** method throws a **MaterialException** then returns a message "**Invalid Material**".(Use try-catch block)
* If it throws any other exception then return a message "**Other exception**".
* **If no exception is found then return "Both in equilibrium".**

**Sample Input**

Material m=**new** Material(12,32);

String s=m.flowOfHeat();

**Sample Output**

**Both in equilibrium**

**NOTE:**

* You can make suitable function calls and use **the RUN CODE** button to check your **main()** method output.

**ALLOWED TECHNOLOGIES**

* Java 8

**TAGS**

* Exceptions
* Exceptions and Exception Handling

//DOSELECT Problem Statement 2

//Material Class

**package** CAPG;

**public** **class** Material {

**int** mass;

**int** temperature;

**public** Material(**int** mass,**int** temperature) {

**this**.mass=mass;

**this**.temperature=temperature;

}

**public** String checkMaterial() **throws** Exception{

**if**(**this**.mass<0){

**throw** **new** MaterialException("Mass cannot be negative");

}

**else** **if**(**this**.mass==0){

**throw** **new** MaterialException("Mass cannot be zero");

}

**else**{

**return** "Valid material";

}

}

**public** String flowOfHeat() **throws** Exception{

**try**{

checkMaterial();

}

**catch**(MaterialException e1){

**return** "Invalid Material";

}

**catch**(Exception e2){

**return** "Other exception";

}

**return** "Both in equilibrium";

}

}

//DOSELECT Problem Statement 2

//MaterialException Class

**package** CAPG;

**public** **class** MaterialException **extends** Exception{

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**public** MaterialException(String msg){

**super**(msg);

}

}

//DOSELECT Problem Statement 2

//Main Class

**package** CAPG;

**public** **class** Main {

**public** **static** **void** main(String[] args) **throws** Exception {

Material m=**new** Material(12,32);

String s=m.flowOfHeat();

System.***out***.println(s);

}

}

