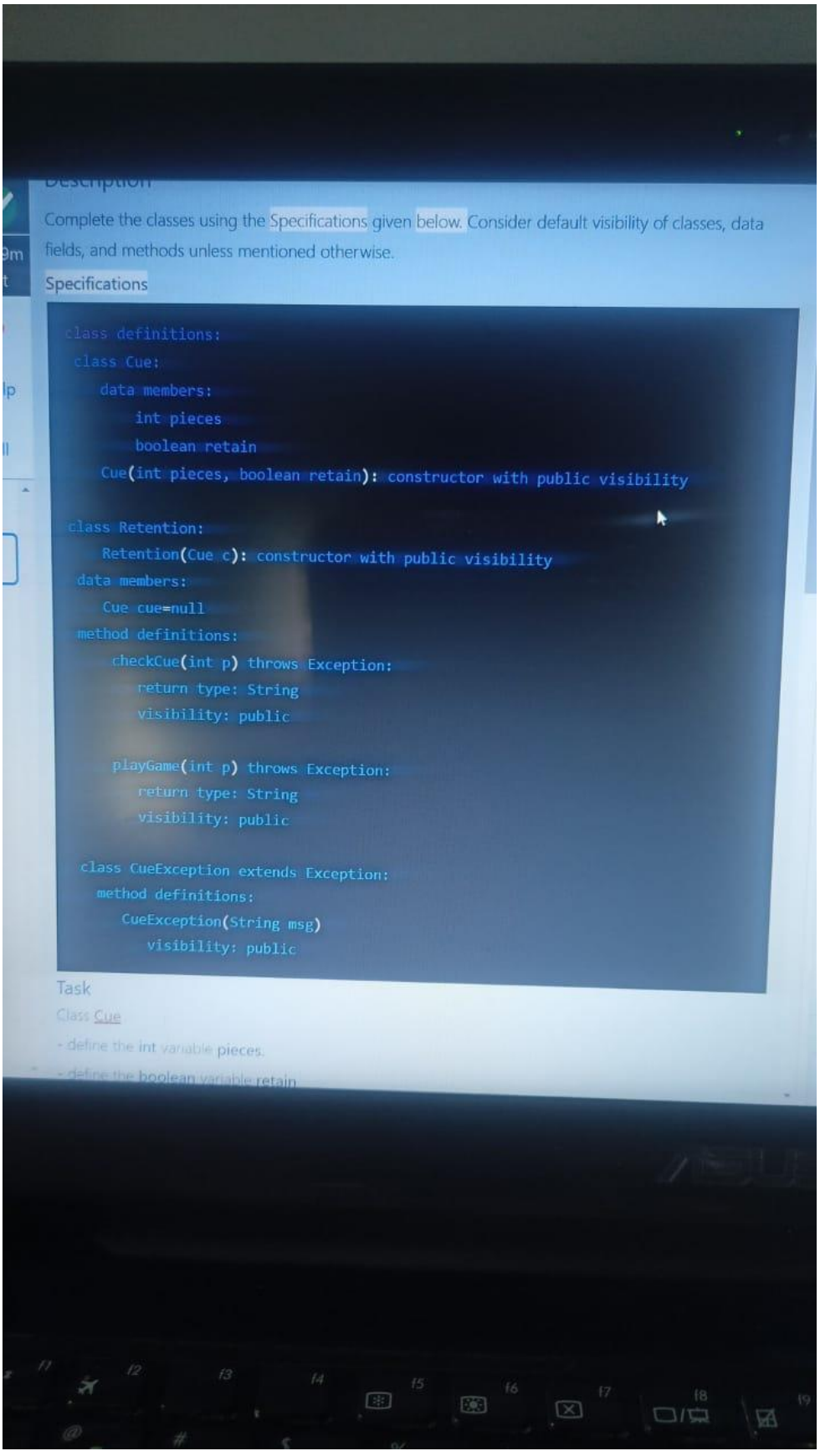


Description

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

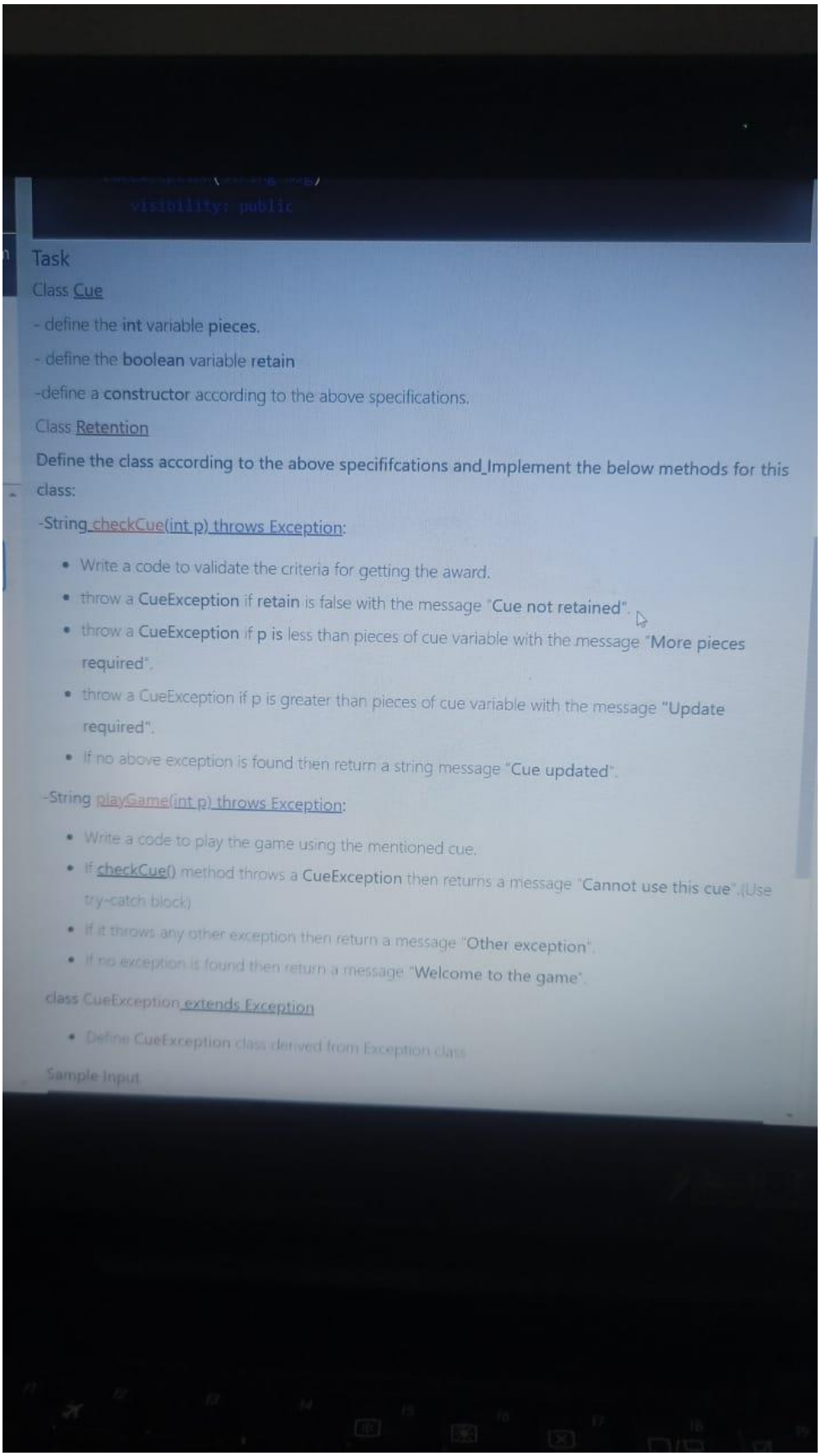
Specifications

```
class definitions:
 class Cue:
    data members:
        int pieces
        boolean retain
    Cue(int pieces, boolean retain): constructor with public visibility

class Retention:
    Retention(Cue c): constructor with public visibility
 data members:
    Cue cue=null
 method definitions:
     checkCue(int p) throws Exception:
        return type: String
        visibility: public

     playGame(int p) throws Exception:
        return type: String
        visibility: public

 class CueException extends Exception:
   method definitions:
      CueException(String msg)
         visibility: public
```

Task

Class Cue

- define the int variable pieces.
- define the boolean variable retain



visibility: public

Task

Class Cue

- define the **int** variable **pieces**.

- define the **boolean** variable **retain**

-define a **constructor** according to the above specifications.

Class Retention

Define the class according to the above specifiifcations and Implement the below methods for this class:

-String checkCue(int p) throws Exception:

- Write a code to validate the criteria for getting the award.
- throw a **CueException** if **retain** is false with the message "**Cue not retained**".
- throw a **CueException** if **p is** less than pieces of cue variable with the message "**More pieces required**".
- throw a CueException if p is greater than pieces of cue variable with the message "**Update required**".
- If no above exception is found then return a string message "**Cue updated**".

-String playGame(int p) throws Exception:

- Write a code to play the game using the mentioned cue.
- If checkCue() method throws a **CueException** then returns a message "**Cannot use this cue**".(Use try-catch block)
- If it throws any other exception then return a message "**Other exception**".
- If no exception is found then return a message "**Welcome to the game**".

class CueException extends Exception

- Define CueException class derived from Exception class

Sample Input

- Write a code to play the game using the mentioned cue.
- If checkCue() method throws a **CueException** then returns a message "**Cannot use this cue**".(Use try-catch block)
- If it throws any other exception then return a message "**Other exception**".
- If no exception is found then return a message "**Welcome to the game**".

class CueException extends Exception

- Define **CueException** class derived from Exception class

Sample Input

```
Cue c=new Cue(13,true);
Retention r= new Retention(c);
String ans = r.playGame(5);
```

Sample Output

```
cannot use this cue
```

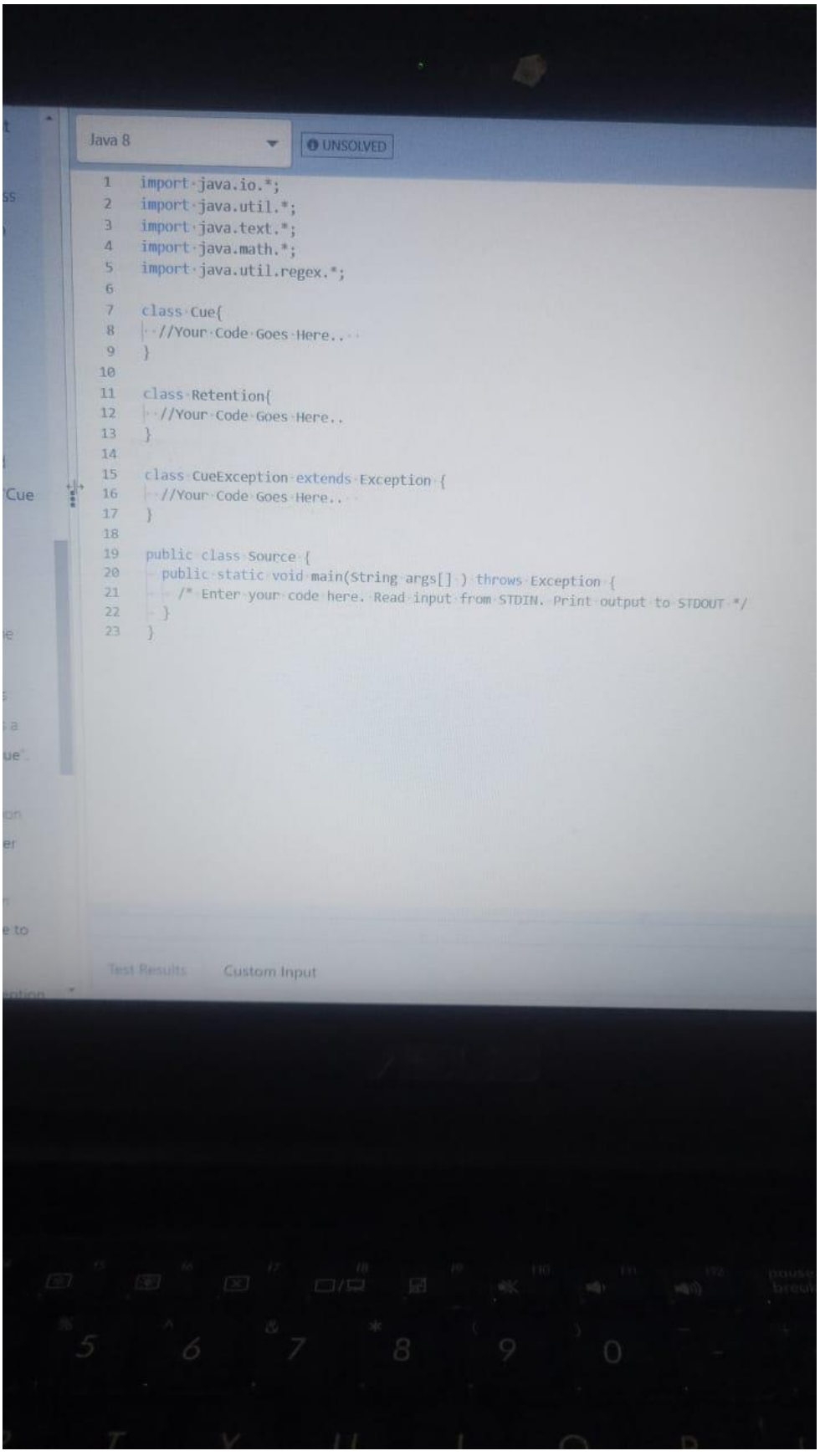
NOTE:

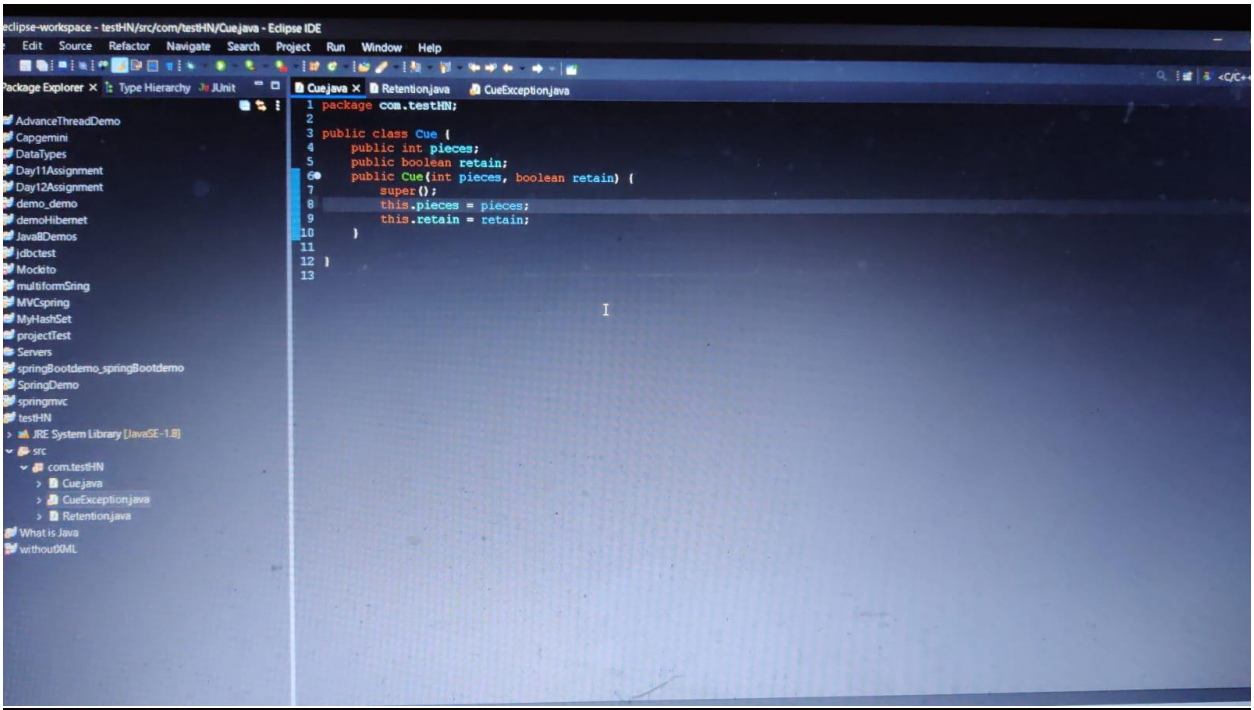
- You can make suitable function calls and use the RUN CODE button to check your main() method output.

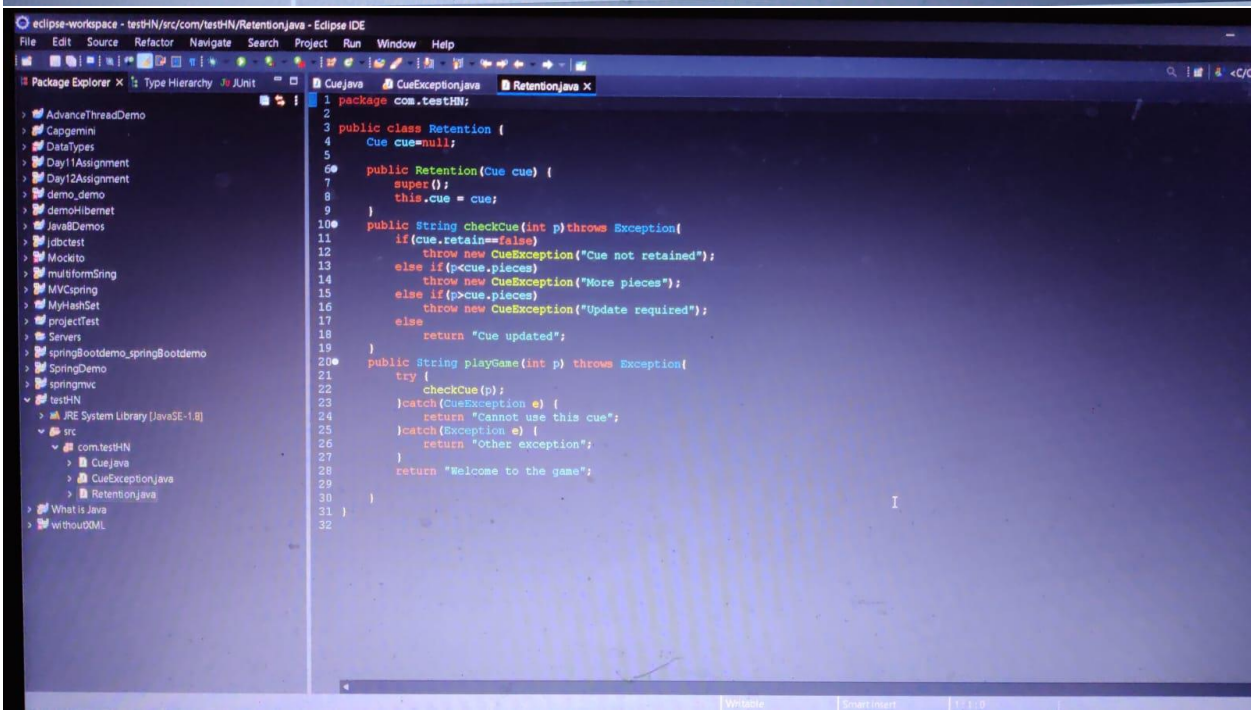
Execution time limit

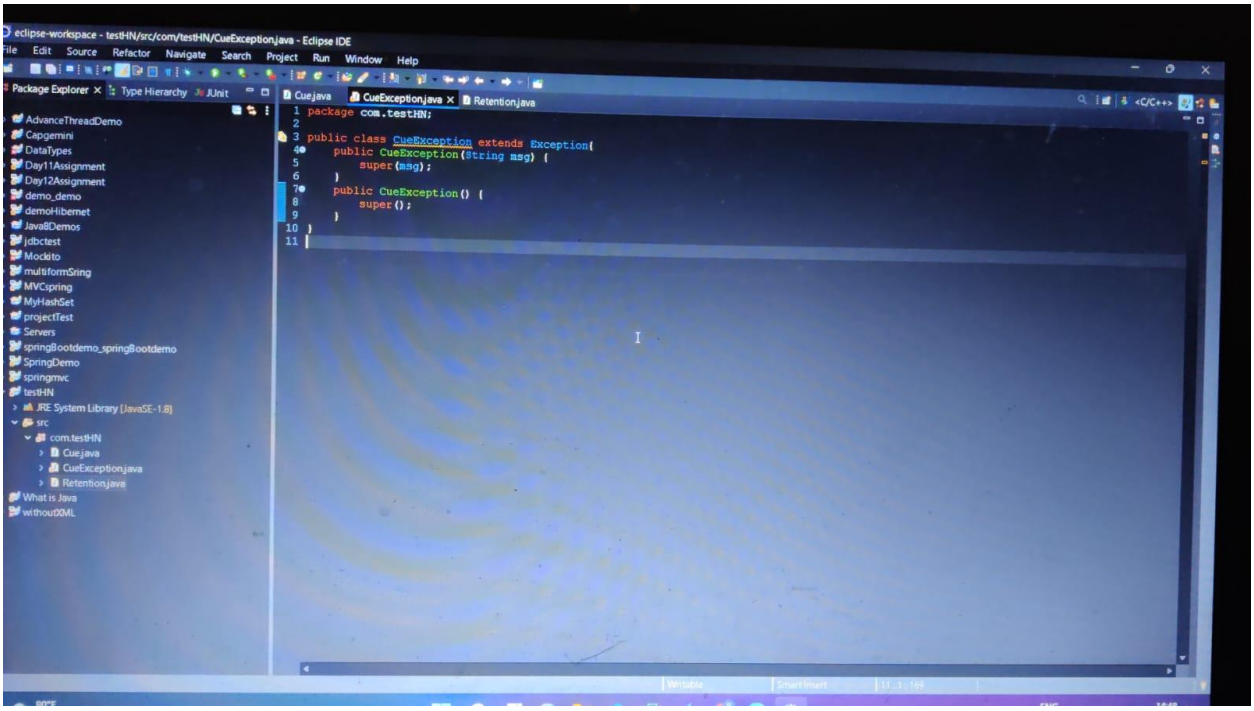
10 seconds

REPORT AN ISSUE

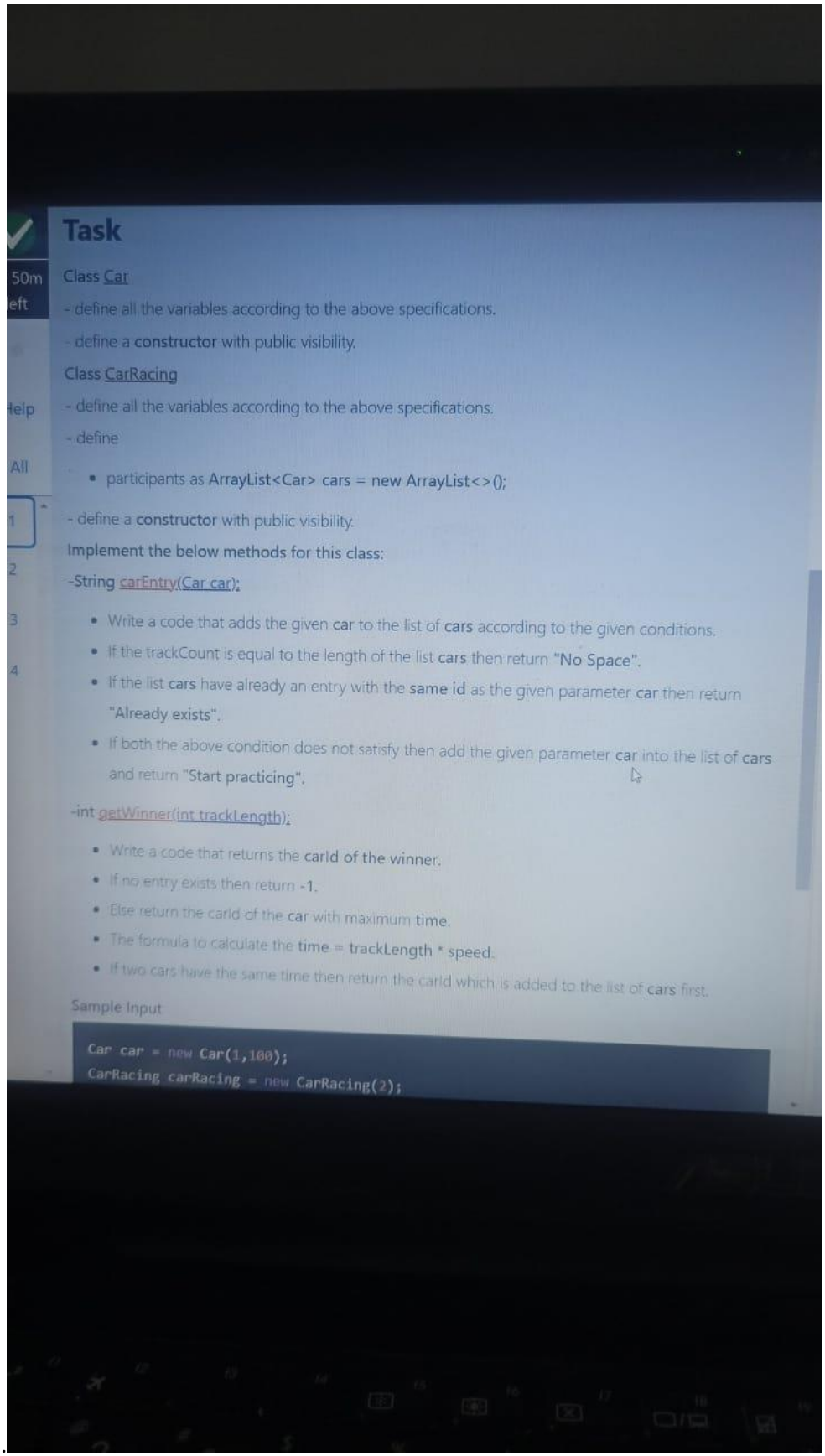

Java 8
UNSOLVED
1 import java.io.*;
2 import java.util.*;
3 import java.text.*;
4 import java.math.*;
5 import java.util.regex.*;
6
7 class Cue{
8 //Your Code Goes Here..
9 }
10
11 class Retention{
12 //Your Code Goes Here..
13 }
14
15 class CueException extends Exception {
16 //Your Code Goes Here..
17 }
18
19 public class Source {
20 public static void main(String args[]) throws Exception {
21 /* Enter your code here. Read input from STDIN. Print output to STDOUT */
22 }
23 }
Test Results
Custom Input


eclipse-workspace - testHN/src/com/testHN/Cue.java - Eclipse IDE
Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer Type Hierarchy JUnit
Cue.java RetentionJava CueException.java
AdvanceThreadDemo
Capgemini
DataTypes
Day11Assignment
Day12Assignment
demo_demo
demoHibernet
Java8Demos
jdbctest
Mockito
multiformSring
MVCspring
MyHashSet
projectTest
Servers
springBootdemo_springBootdemo
SpringDemo
springmvc
testHN
JRE System Library [JavaSE-1.8]
src
com.testHN
Cue.java
CueException.java
RetentionJava
What is Java
withoutXML
1 package com.testHN;
2
3 public class Cue {
4 public int pieces;
5 public boolean retain;
6 public Cue(int pieces, boolean retain) {
7 super();
8 this.pieces = pieces;
9 this.retain = retain;
10 }
11
12 }
13


eclipse-workspace - testHN/src/com/testHN/RetentionJava - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer Type Hierarchy JUnit
Cue.java CueException.java RetentionJava
AdvanceThreadDemo
Capgemini
DataTypes
Day11Assignment
Day12Assignment
demo_demo
demoHibernet
Java8Demos
jdbctest
Mockito
multiformSring
MVCspring
MyHashSet
projectTest
Servers
springBootdemo_springBootdemo
SpringDemo
springmvc
testHN
JRE System Library [JavaSE-1.8]
src
com.testHN
Cue.java
CueException.java
RetentionJava
What is Java
withoutXML
1 package com.testHN;
2
3 public class Retention {
4 Cue cue=null;
5
6 public Retention(Cue cue) {
7 super();
8 this.cue = cue;
9 }
10 public String checkCue(int p)throws Exception{
11 if(cue.retain==false)
12 throw new CueException("Cue not retained");
13 else if(p<cue.pieces)
14 throw new CueException("More pieces");
15 else if(p>cue.pieces)
16 throw new CueException("Update required");
17 else
18 return "Cue updated";
19 }
20 public String playGame(int p) throws Exception{
21 try {
22 checkCue(p);
23 }catch(CueException e) {
24 return "Cannot use this cue";
25 }catch(Exception e) {
26 return "Other exception";
27 }
28 return "Welcome to the game";
29
30 }
31 }
32
Writable
Smart Insert


eclipse-workspace - testHN/src/com/testHN/CueException.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Type Hierarchy
JUnit
AdvanceThreadDemo
Capgemini
DataTypes
Day11Assignment
Day12Assignment
demo_demo
demoHibernet
Java8Demos
jdbctest
Mockito
multiformSring
MVCspring
MyHashSet
projectTest
Servers
springBootdemo_springBootdemo
SpringDemo
springmvc
testHN
JRE System Library [JavaSE-1.8]
src
com.testHN
Cue.java
CueException.java
Retention.java
What is Java
without0ML
Cue.java
CueException.java
Retention.java
<C/C++>
1 package com.testHN;
2
3 public class CueException extends Exception{
4 public CueException(String msg) {
5 super(msg);
6 }
7 public CueException() {
8 super();
9 }
10 }
11
Writable
Smart Insert

2.



Task

Class Car

- define all the variables according to the above specifications.
- define a **constructor** with public visibility.

Class CarRacing

- define all the variables according to the above specifications.
- define

  - participants as `ArrayList<Car> cars = new ArrayList<>();`

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

Implement the below methods for this class:

-String carEntry(Car car);

- Write a code that adds the given **car** to the list of **cars** according to the given conditions.
- If the trackCount is equal to the length of the list **cars** then return **"No Space"**.
- If the list **cars** have already an entry with the **same id** as the given parameter **car** then return **"Already exists"**.
- If both the above condition does not satisfy then add the given parameter **car** into the list of **cars** and return **"Start practicing"**.

-int getWinner(int trackLength);

- Write a code that returns the **carId** of the winner.
- If no entry exists then return **-1**.
- Else return the carId of the **car** with maximum **time**.
- The formula to calculate the **time** = **trackLength** * **speed**.
- If two cars have the same time then return the carId which is added to the list of **cars** first.

Sample Input

```
Car car = new Car(1,100);
CarRacing carRacing = new CarRacing(2);
```

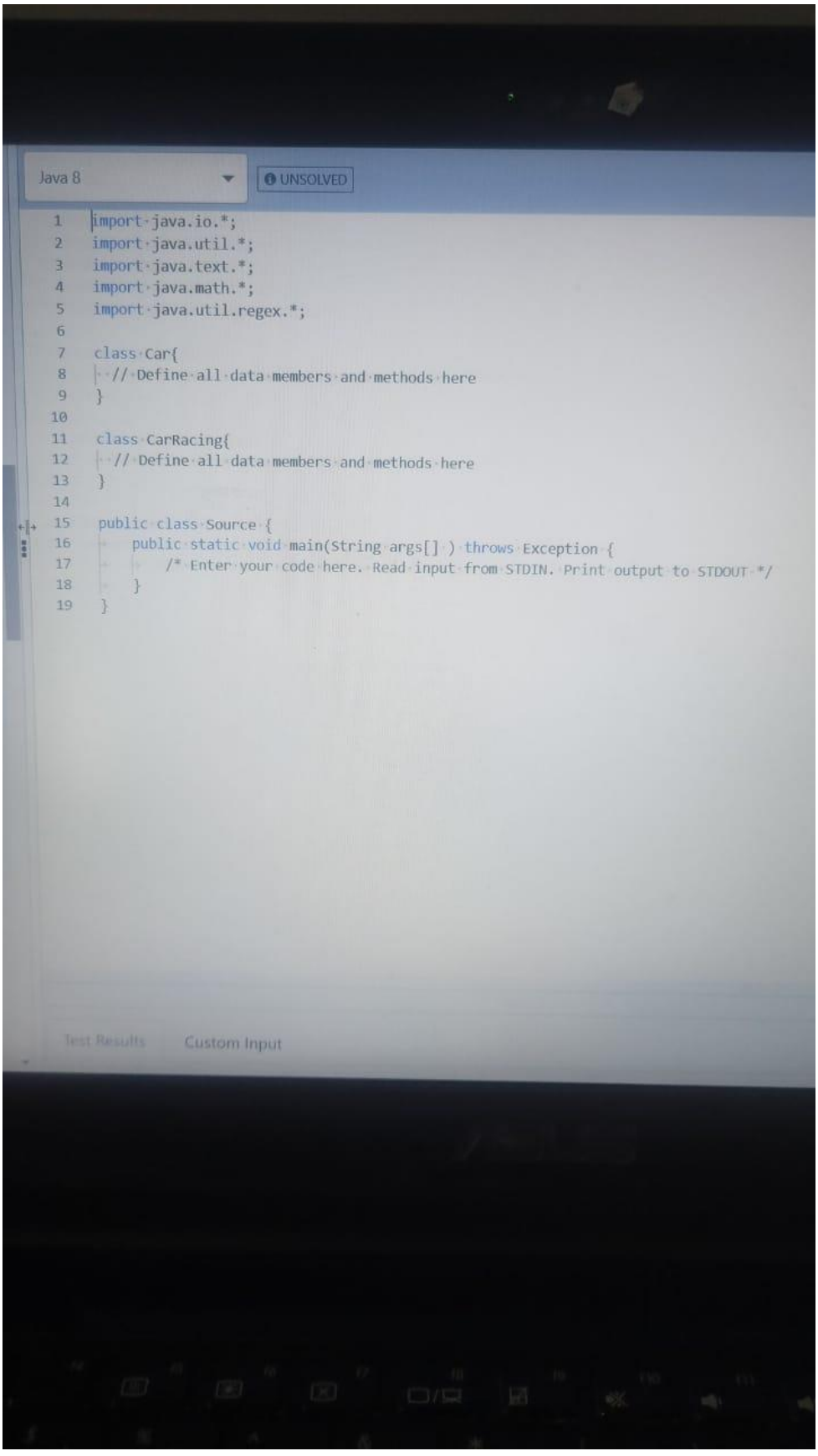
Description

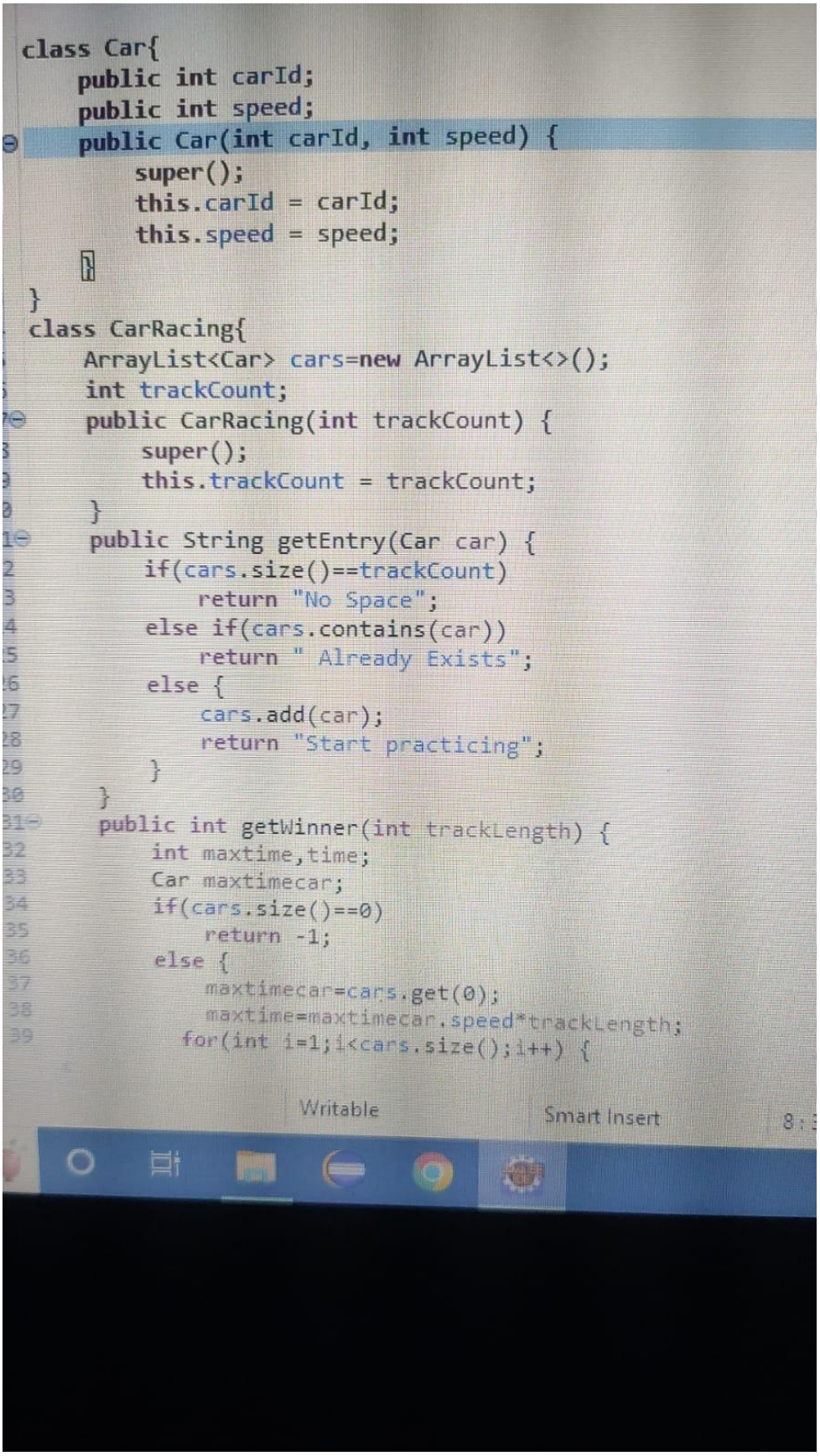
Complete the classes using the Specifications given below. Consider default visibility of classes, data fields, and methods unless mentioned otherwise.

Specifications

```
class definitions:
  class Car:
    data members:
      int carId
      int speed
       visibility : public

    Car(int carId, int speed): constructor with public visibility

  class CarRacing:
    data members:
      String ArrayList<Car> cars
      int trackCount
      visibility : public

    CarRacing(int trackCount): constructor with public visibility

    method definitions:
      carEntry(Car car):
        return type: String
        visibility: public

      getWinner(int trackLength):
        return type: int
        visibility: public
```

Task


Java 8
UNSOLVED
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
class Car{
// Define all data members and methods here
}
class CarRacing{
// Define all data members and methods here
}
public class Source {
public static void main(String args[]) throws Exception {
/* Enter your code here. Read input from STDIN. Print output to STDOUT */
}
}
Test Results
Custom Input



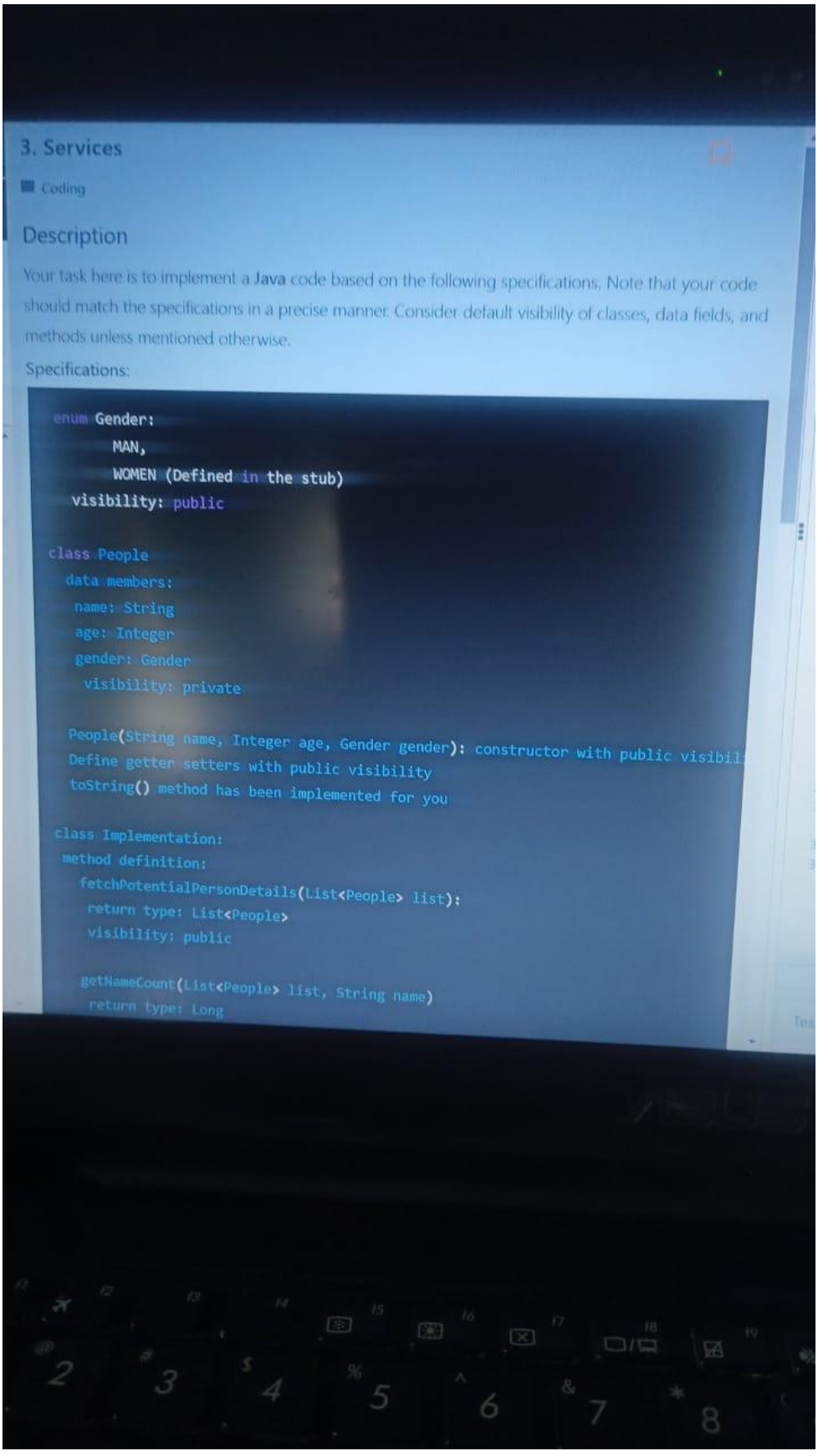
```
class Car{
    public int carId;
    public int speed;
    public Car(int carId, int speed) {
        super();
        this.carId = carId;
        this.speed = speed;
    }
}
class CarRacing{
    ArrayList<Car> cars=new ArrayList<>();
    int trackCount;
    public CarRacing(int trackCount) {
        super();
        this.trackCount = trackCount;
    }
    public String getEntry(Car car) {
        if(cars.size()==trackCount)
            return "No Space";
        else if(cars.contains(car))
            return " Already Exists";
        else {
            cars.add(car);
            return "Start practicing";
        }
    }
    public int getWinner(int trackLength) {
        int maxtime,time;
        Car maxtimecar;
        if(cars.size()==0)
            return -1;
        else {
            maxtimecar=cars.get(0);
            maxtime=maxtimecar.speed*trackLength;
          for(int i=1;i<cars.size();i++) {
```

Writable Smart Insert 8 :

Order.java | Medicine.java | Customer.java | Category

```
    public CarRacing(int trackCount) {
        super();
        this.trackCount = trackCount;
    }
    public String getEntry(Car car) {
        if(cars.size()==trackCount)
            return "No Space";
        else if(cars.contains(car))
            return " Already Exists";
        else {
            cars.add(car);
            return "Start practicing";
        }
    }
    public int getWinner(int trackLength) {
        int maxtime,time;
        Car maxtimecar;
        if(cars.size()==0)
            return -1;
        else {
            maxtimecar=cars.get(0);
            maxtime=maxtimecar.speed*trackLength;
          for(int i=1;i<cars.size();i++) {
            Car c=cars.get(i);
            time=c.speed*trackLength;
            if(time>maxtime)
                {
                    maxtimecar=c;
                    maxtime=time;
                }
          }
        }
        return maxtimecar.carId;
    }
}
```

3.


3. Services
Coding
Description
Your task here is to implement a Java code based on the following specifications. Note that your code should match the specifications in a precise manner. Consider default visibility of classes, data fields, and methods unless mentioned otherwise.
Specifications:
enum Gender:
MAN,
WOMEN (Defined in the stub)
visibility: public
class People
data members:
name: String
age: Integer
gender: Gender
visibility: private
People(String name, Integer age, Gender gender): constructor with public visibil
Define getter setters with public visibility
toString() method has been implemented for you
class Implementation:
method definition:
fetchPotentialPersonDetails(List<People> list):
return type: List<People>
visibility: public
getNameCount(List<People> list, String name)
return type: Long

potentially working people from the list (i.e., from the age of 18(inclusive) and considering that women retire at 55(exclusive), and a man at 60(exclusive))

- static Long getNameCount(List<People> list, String name): return the count of name from the list

Refer to sample output for more details

Sample Input

```
List<People> list = new ArrayList<>();
        list.add(new People("Scarlet", 30, Gender.WOMEN));
        list.add(new People("David Beckham", 25, Gender.MAN));

list //Input for the first method
list, Scarlet //Input for third method
```

Sample Output

```
[People{name='Scarlet', age=30, gender=WOMEN}, People{name='David Beckham', age=25,
------------------------------------------------------------
1
```

NOTE

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

Execution time limit

10 seconds

REPORT AN ISSUE

```
getNameCount(List<People> list, String name)
 return type: Long
 visibility: public
```

Task:

enum Gender: MAN,WOMEN (Present in the code stub)

class People:

- define **data members** according to the above specifications

-define a **constructor** and **getter setters** according to the above specifications

-toString() method has been implemented for you as a part of the code stub

class Implementation:

Implement the below method for this class using in Stream API:

- static List<People> fetchPotentialPersonDetails(List<People> list): Fetch the **details of potentially working people from the list** (i.e., from the age of 18(inclusive) and considering that women retire at 55(exclusive), and a man at 60(exclusive))
- static Long getNameCount(List<People> list, String name): return the **count of name** from the list

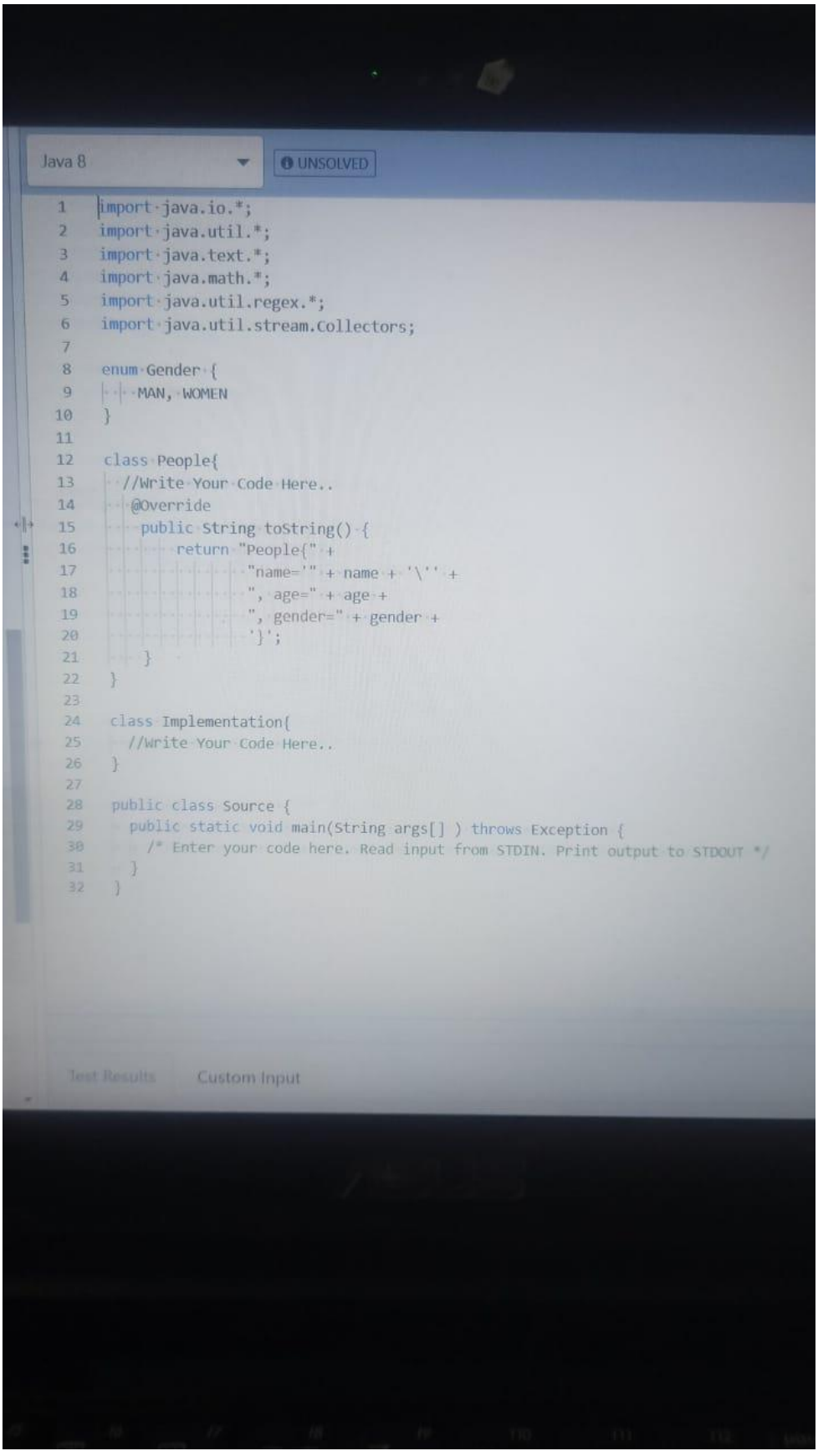
> *Refer to sample output for more details*

Sample Input

```
List<People> list = new ArrayList<>();
            list.add(new People("Scarlet", 30, Gender.WOMEN));
            list.add(new People("David Beckham", 25, Gender.MAN));
-----------------------------------------------
list //Input for the  first method
list, Scarlet //Input for third method
```

Sample Output

```
[People{name='Scarlet', age=30, gender=WOMEN}, People{name='David Beckham', age=25
```



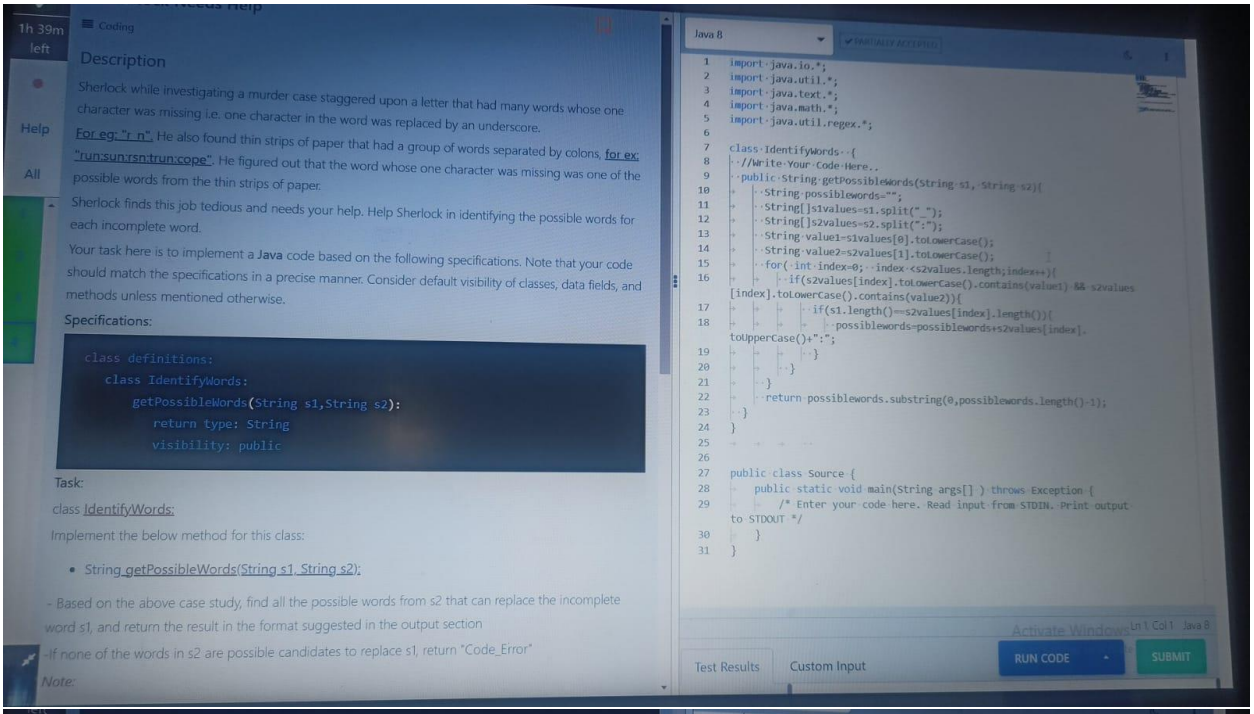
Java 8

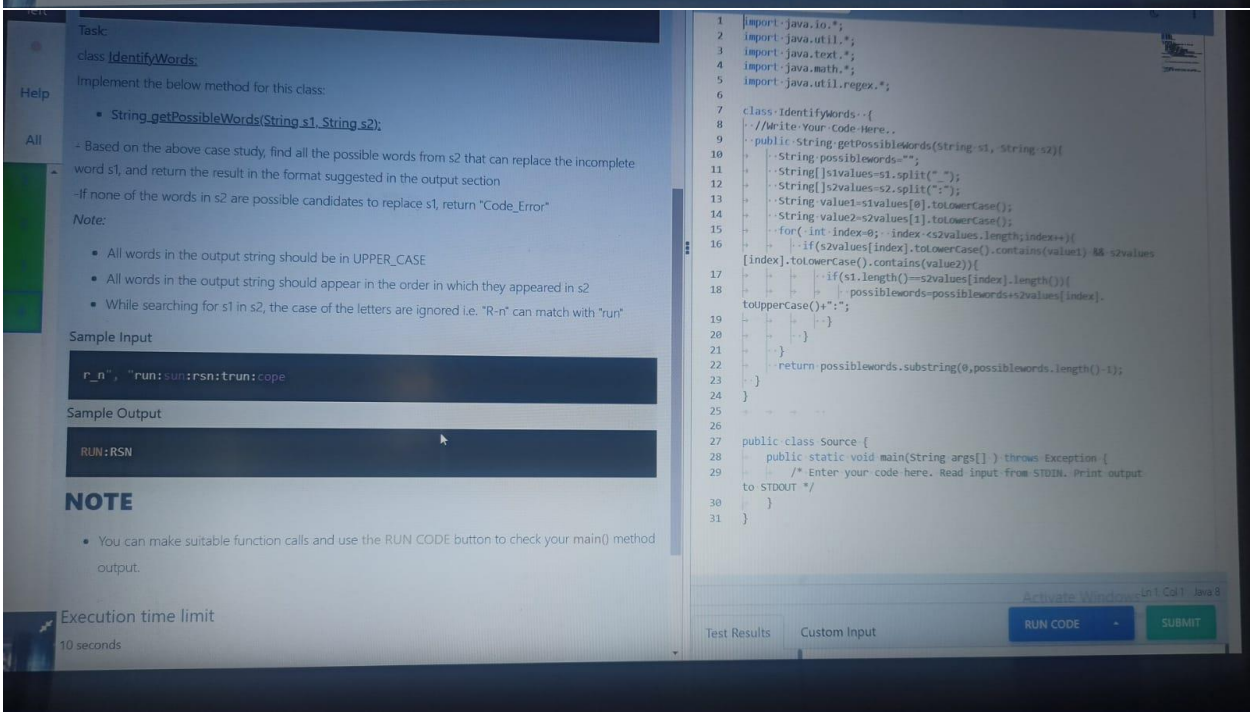
UNSOLVED

```java
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
import java.util.stream.Collectors;

enum Gender {
    MAN, WOMEN
}

class People{
    //Write Your Code Here..
    @Override
    public String toString() {
        return "People{" +
                "name='" + name + '\'' +
                ", age=" + age +
                ", gender=" + gender +
                '}';
    }
}

class Implementation{
    //Write Your Code Here..
}

public class Source {
    public static void main(String args[] ) throws Exception {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    }
}
```

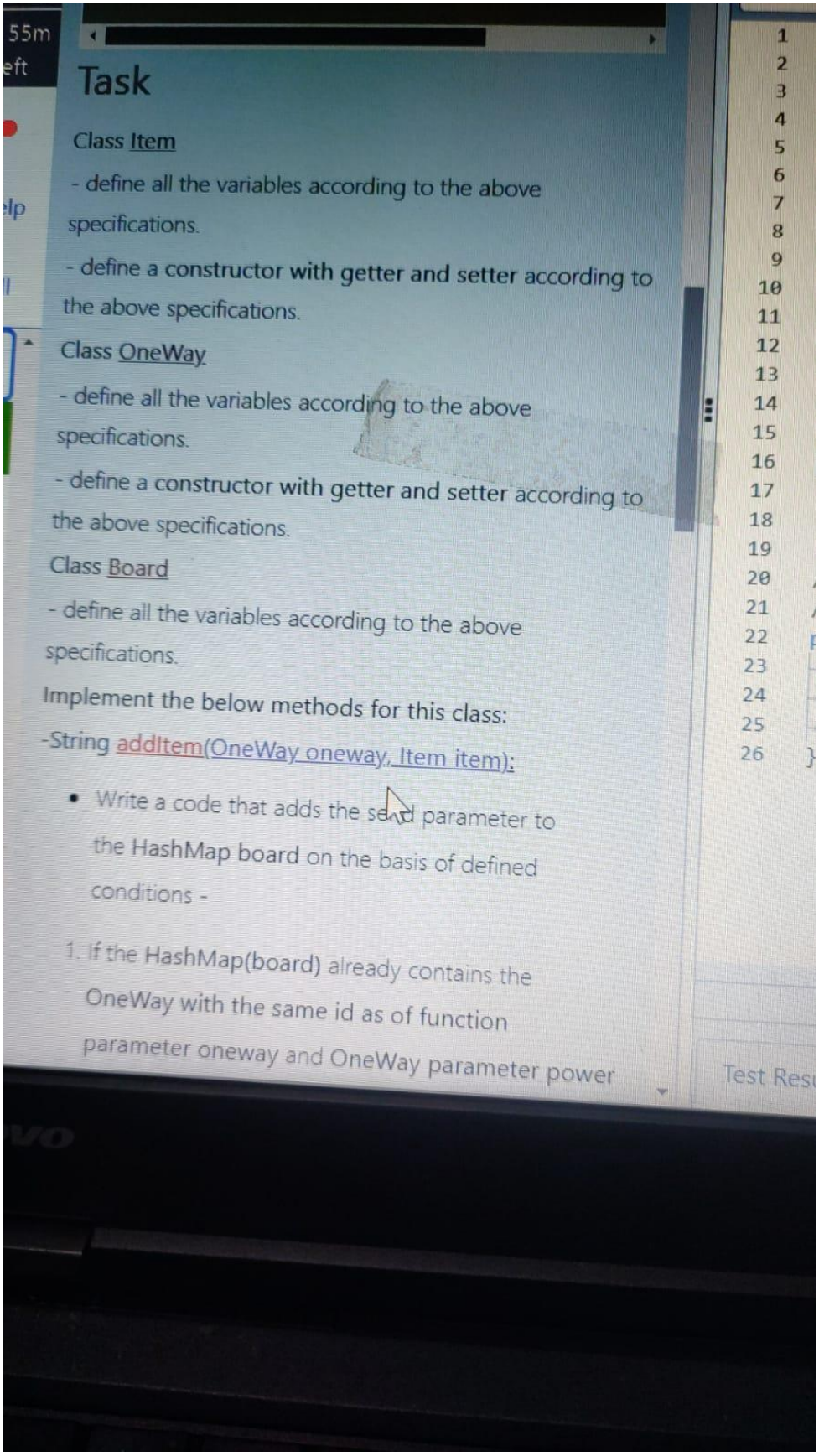
Test Results Custom Input

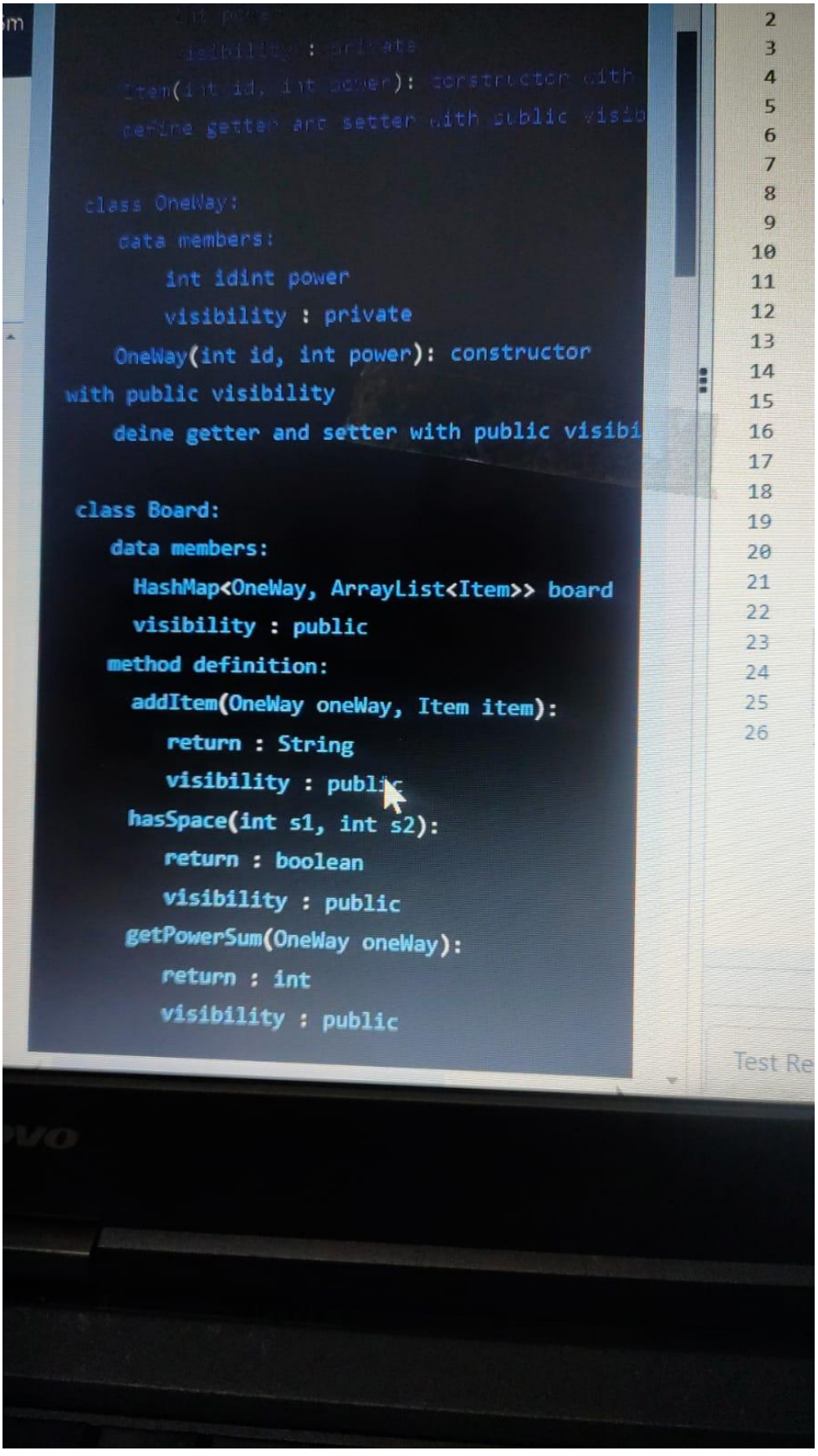
4.

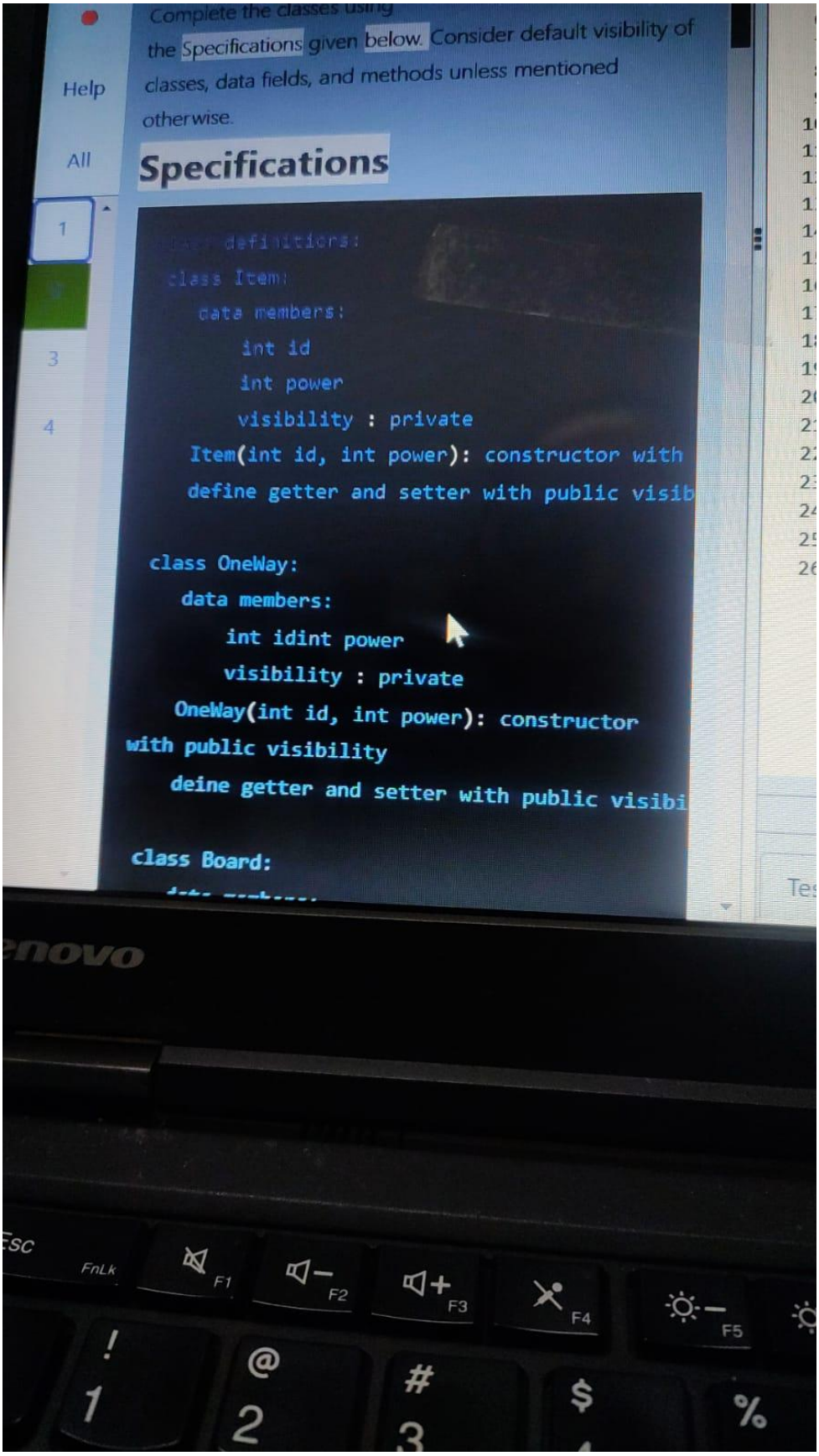

1h 39m left
Coding
Description
Sherlock while investigating a murder case staggered upon a letter that had many words whose one character was missing i.e. one character in the word was replaced by an underscore.
For eg: "r_n". He also found thin strips of paper that had a group of words separated by colons, for ex: "run:sun:rsn:trun:cope". He figured out that the word whose one character was missing was one of the possible words from the thin strips of paper.
Sherlock finds this job tedious and needs your help. Help Sherlock in identifying the possible words for each incomplete word.
Your task here is to implement a Java code based on the following specifications. Note that your code should match the specifications in a precise manner. Consider default visibility of classes, data fields, and methods unless mentioned otherwise.
Specifications:
class definitions:
class IdentifyWords:
getPossibleWords(String s1,String s2):
return type: String
visibility: public
Task:
class IdentifyWords:
Implement the below method for this class:
String getPossibleWords(String s1, String s2):
- Based on the above case study, find all the possible words from s2 that can replace the incomplete word s1, and return the result in the format suggested in the output section
-If none of the words in s2 are possible candidates to replace s1, return "Code_Error"
Note:
Java 8
RUN CODE
SUBMIT
Test Results
Custom Input

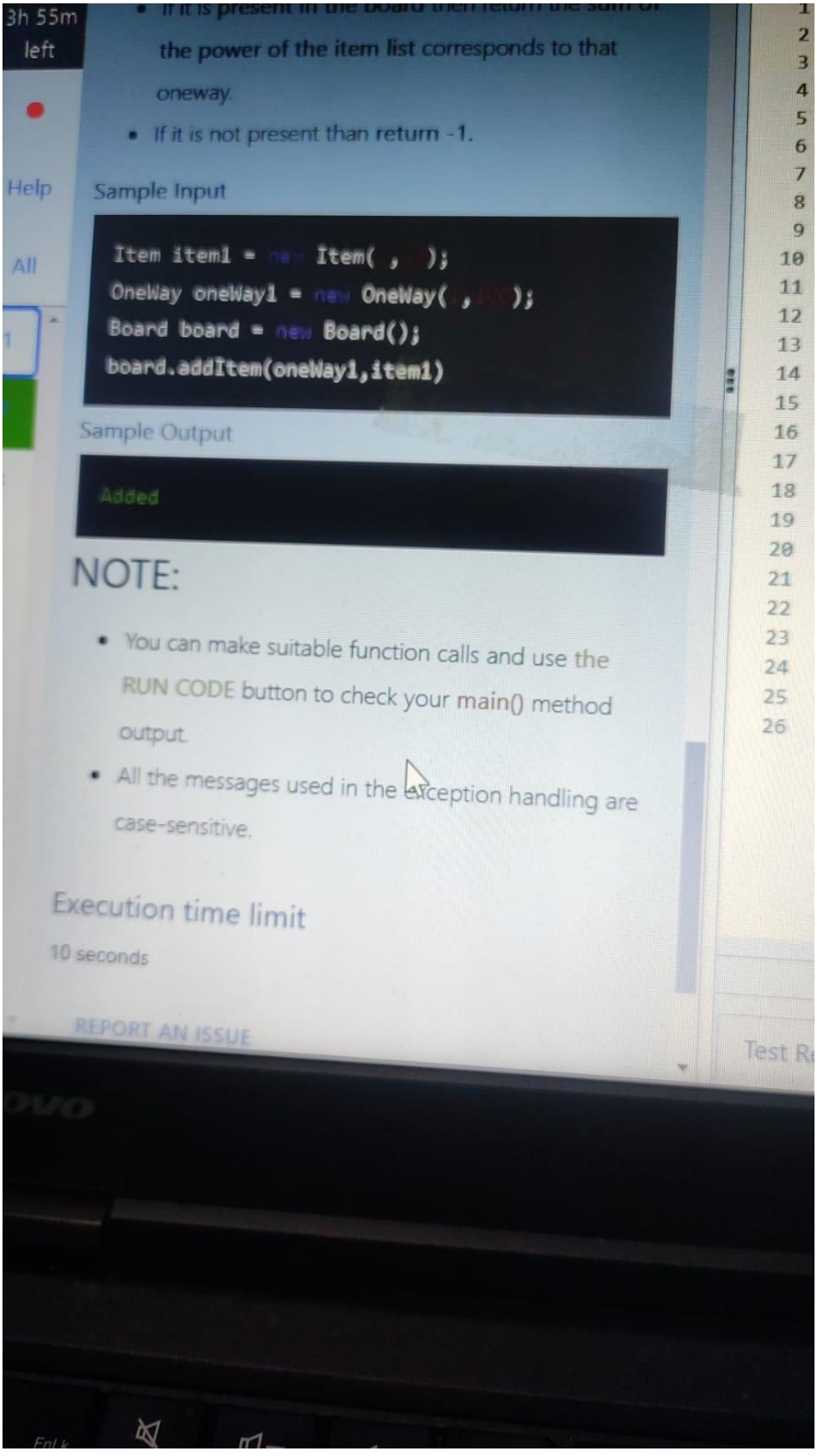

Task:
class IdentifyWords:
Implement the below method for this class:
String getPossibleWords(String s1, String s2):
- Based on the above case study, find all the possible words from s2 that can replace the incomplete word s1, and return the result in the format suggested in the output section
-If none of the words in s2 are possible candidates to replace s1, return "Code_Error"
Note:
All words in the output string should be in UPPER_CASE
All words in the output string should appear in the order in which they appeared in s2
While searching for s1 in s2, the case of the letters are ignored i.e. "R-n" can match with "run"
Sample Input
r_n", "run:sun:rsn:trun:cope
Sample Output
RUN:RSN
NOTE
You can make suitable function calls and use the RUN CODE button to check your main() method output.
Execution time limit
10 seconds
RUN CODE
SUBMIT
Test Results
Custom Input

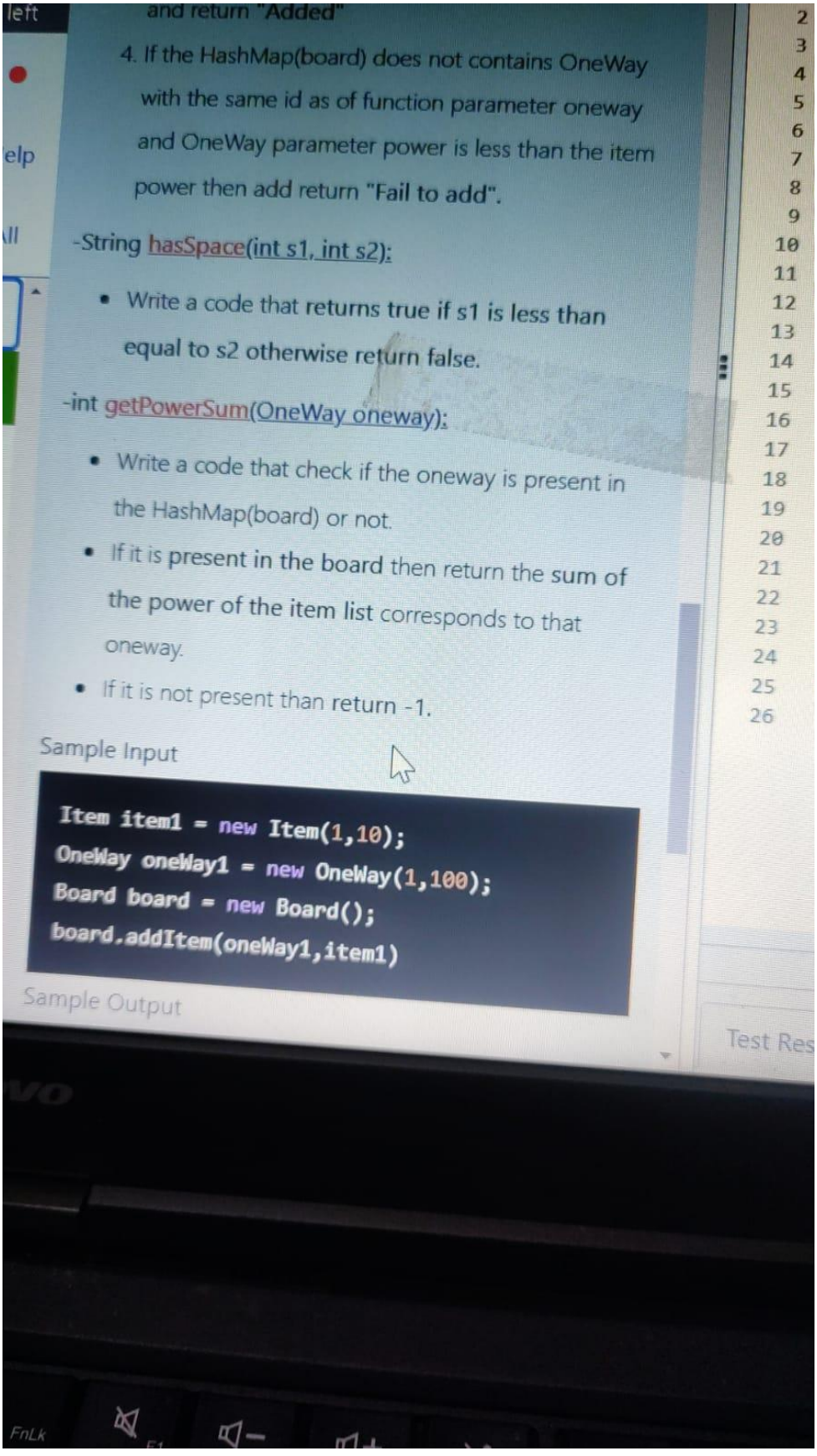
5.

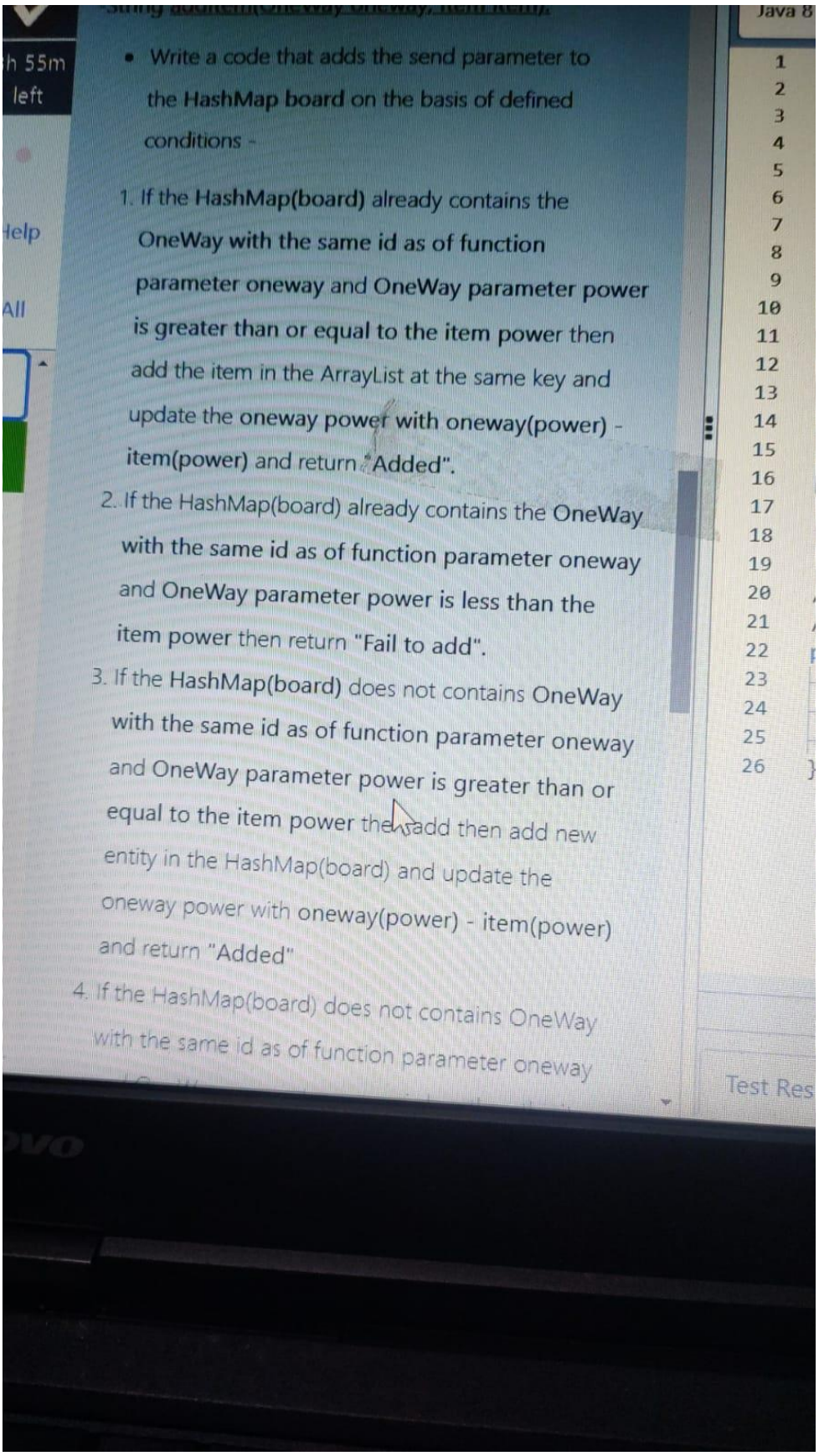

55m
eft
elp
Task
Class Item
- define all the variables according to the above specifications.
- define a constructor with getter and setter according to the above specifications.
Class OneWay
- define all the variables according to the above specifications.
- define a constructor with getter and setter according to the above specifications.
Class Board
- define all the variables according to the above specifications.
Implement the below methods for this class:
-String addItem(OneWay oneway, Item item):
Write a code that adds the second parameter to the HashMap board on the basis of defined conditions -
1. If the HashMap(board) already contains the OneWay with the same id as of function parameter oneway and OneWay parameter power
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 }
Test Resu

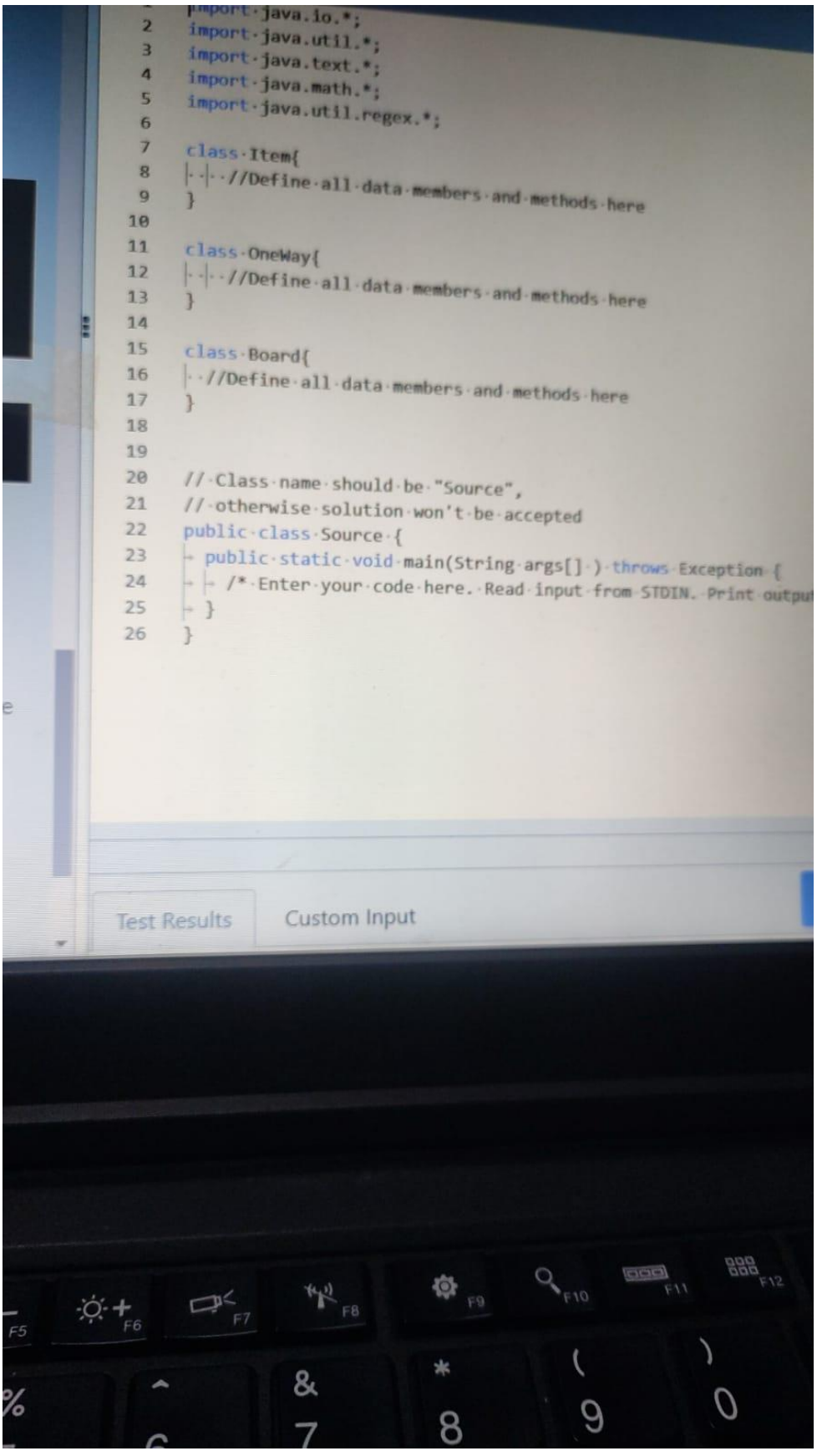

Item(int id, int power): constructor with
define getter and setter with public visib
class OneWay:
data members:
int idint power
visibility : private
OneWay(int id, int power): constructor
with public visibility
deine getter and setter with public visibi
class Board:
data members:
HashMap<OneWay, ArrayList<Item>> board
visibility : public
method definition:
addItem(OneWay oneWay, Item item):
return : String
visibility : public
hasSpace(int s1, int s2):
return : boolean
visibility : public
getPowerSum(OneWay oneWay):
return : int
visibility : public
Test Re


Complete the classes using
the Specifications given below. Consider default visibility of
classes, data fields, and methods unless mentioned
otherwise.
Help
All
Specifications
1
3
4
definitions:
class Item:
data members:
int id
int power
visibility : private
Item(int id, int power): constructor with
define getter and setter with public visib
class OneWay:
data members:
int idint power
visibility : private
OneWay(int id, int power): constructor
with public visibility
deine getter and setter with public visibi
class Board:
lenovo
Esc
FnLk
F1
F2
F3
F4
F5
!
1
@
2
#
3
$
%


3h 55m left
Help
All
the power of the item list corresponds to that oneway.
If it is not present than return -1.
Sample Input
Item item1 = new Item(,);
OneWay oneWay1 = new OneWay(,);
Board board = new Board();
board.addItem(oneWay1,item1)
Sample Output
Added
NOTE:
You can make suitable function calls and use the RUN CODE button to check your main() method output.
All the messages used in the exception handling are case-sensitive.
Execution time limit
10 seconds
REPORT AN ISSUE


and return "Added"
4. If the HashMap(board) does not contains OneWay with the same id as of function parameter oneway and OneWay parameter power is less than the item power then add return "Fail to add".
-String hasSpace(int s1, int s2):
Write a code that returns true if s1 is less than equal to s2 otherwise return false.
-int getPowerSum(OneWay oneway):
Write a code that check if the oneway is present in the HashMap(board) or not.
If it is present in the board then return the sum of the power of the item list corresponds to that oneway.
If it is not present than return -1.
Sample Input
Item item1 = new Item(1,10);
OneWay oneWay1 = new OneWay(1,100);
Board board = new Board();
board.addItem(oneWay1,item1)
Sample Output
Test Res
FnLk


h 55m
left
Help
All
Java 8
• Write a code that adds the send parameter to the HashMap board on the basis of defined conditions -
1. If the HashMap(board) already contains the OneWay with the same id as of function parameter oneway and OneWay parameter power is greater than or equal to the item power then add the item in the ArrayList at the same key and update the oneway power with oneway(power) - item(power) and return "Added".
2. If the HashMap(board) already contains the OneWay with the same id as of function parameter oneway and OneWay parameter power is less than the item power then return "Fail to add".
3. If the HashMap(board) does not contains OneWay with the same id as of function parameter oneway and OneWay parameter power is greater than or equal to the item power then add then add new entity in the HashMap(board) and update the oneway power with oneway(power) - item(power) and return "Added"
4. If the HashMap(board) does not contains OneWay with the same id as of function parameter oneway
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 }
Test Res



```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

class Item{
    //Define all data members and methods here
}

class OneWay{
    //Define all data members and methods here
}

class Board{
    //Define all data members and methods here
}


// Class name should be "Source",
// otherwise solution won't be accepted
public class Source {
    public static void main(String args[] ) throws Exception {
        /* Enter your code here. Read input from STDIN. Print output
    }
}
```

Test Results

Custom Input

Bulb Question

```
import java.util.*;
import java.util.stream.Collectors;

class Item {
	private int id;
	private int power;

	public Item(int id, int power) {
		this.id = id;
		this.power = power;
	}

	public int getId() {
		return id;
	}

	public void setId(int id) {
		this.id = id;
	}

	public int getPower() {
		return power;
	}

	public void setPower(int power) {
		this.power = power;
	}
```

```
}

class OneWay {
	private int id;
	private int power;

	public OneWay(int id, int power) {
		super();
		this.id = id;
		this.power = power;
	}

	public int getId() {
		return id;
	}

	public void setId(int id) {
		this.id = id;
	}

	public int getPower() {
		return power;
	}

	public void setPower(int power) {
		this.power = power;
	}

}
```

```
class Board {
	public HashMap<OneWay, ArrayList<Item>> board = new HashMap<OneWay,
ArrayList<Item>>();
	public ArrayList<Item> it = new ArrayList<Item>();

	public String addItem(OneWay oneWay, Item item) {
		if (board.containsKey(oneWay)) {
			if (oneWay.getId() == item.getId() && oneWay.getPower() >= item.getPower()) {
				return "Added";
			} else if (oneWay.getId() == item.getId() && oneWay.getPower() <
item.getPower()) {
				return "Fail to add";
			} else if (oneWay.getId() != item.getId() && oneWay.getPower() >=
item.getPower()) {
				oneWay.setPower(oneWay.getPower() - item.getPower());
				it.add(item);
				board.put(oneWay, it);
				return "Added";
			} else
				return "Fail to Add";
		} else
			return "Fail to add";
	}

	public Boolean hasSpace(int s1, int s2) {
		if (s1 <= s2)
			return true;
		else
			return false;
```

```
	}

	public int getPowerSum(OneWay oneWay) {
		int sum=0;
		if (board.containsKey(oneWay))
		{
			for(Item i : board.get(oneWay))
			{
				sum+=i.getPower();
			}
			return sum;
		}
		else
			return -1;
	}
}
class Source
{
	public static void main(String args[])
	{
		Item item1 = new Item(1, 10);
		OneWay oneWay1 = new OneWay(1, 100);
		Board board = new Board();
		board.addItem(oneWay1, item1);
	}
}
```

6.

2. Speedometer

Coding

Description

The RTO has designed a video game for children to bring in awareness regarding road safety.

Help them by writing a Java program to check the functionality of the game software. If the input speed is between **10** and **200** then return the message **"Valid Speed"** else program should alert with the message **"Error in Speedometer"** by throwing a user defined exception **SpeedometerException**.

```
class definitions:
 class Speed:
    data field:
       speed: String

 class SpeedImplementation:
        method definitions:
            setSpeed(Speed s, int speedValue):
                visibility: public
                return type: String

 class SpeedometerException extends Exception:
        method definitions:
            SpeedometerException(String s): Constructor
                visibility: public
```

TASKS:

class Speed

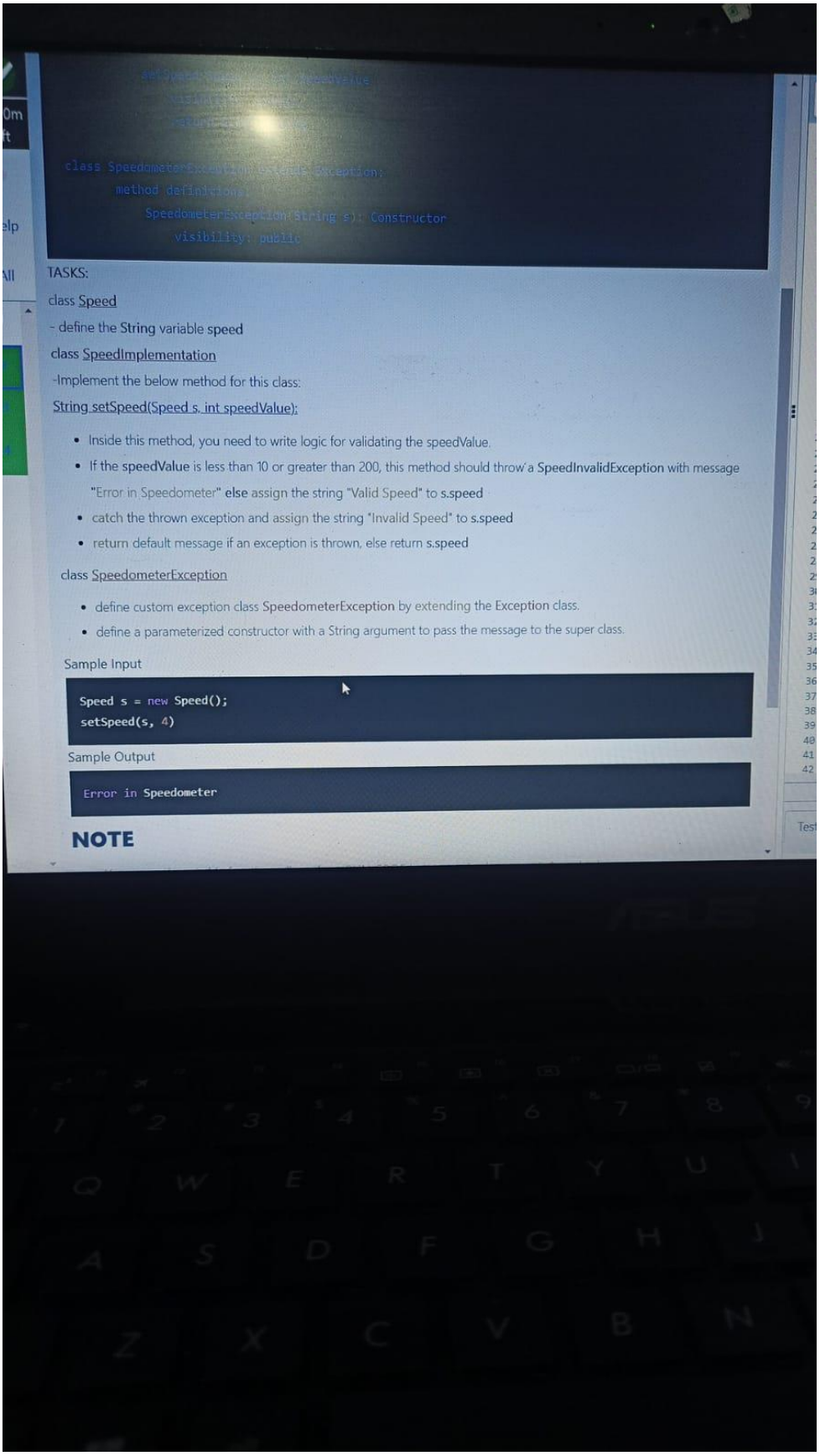
- define the String variable speed

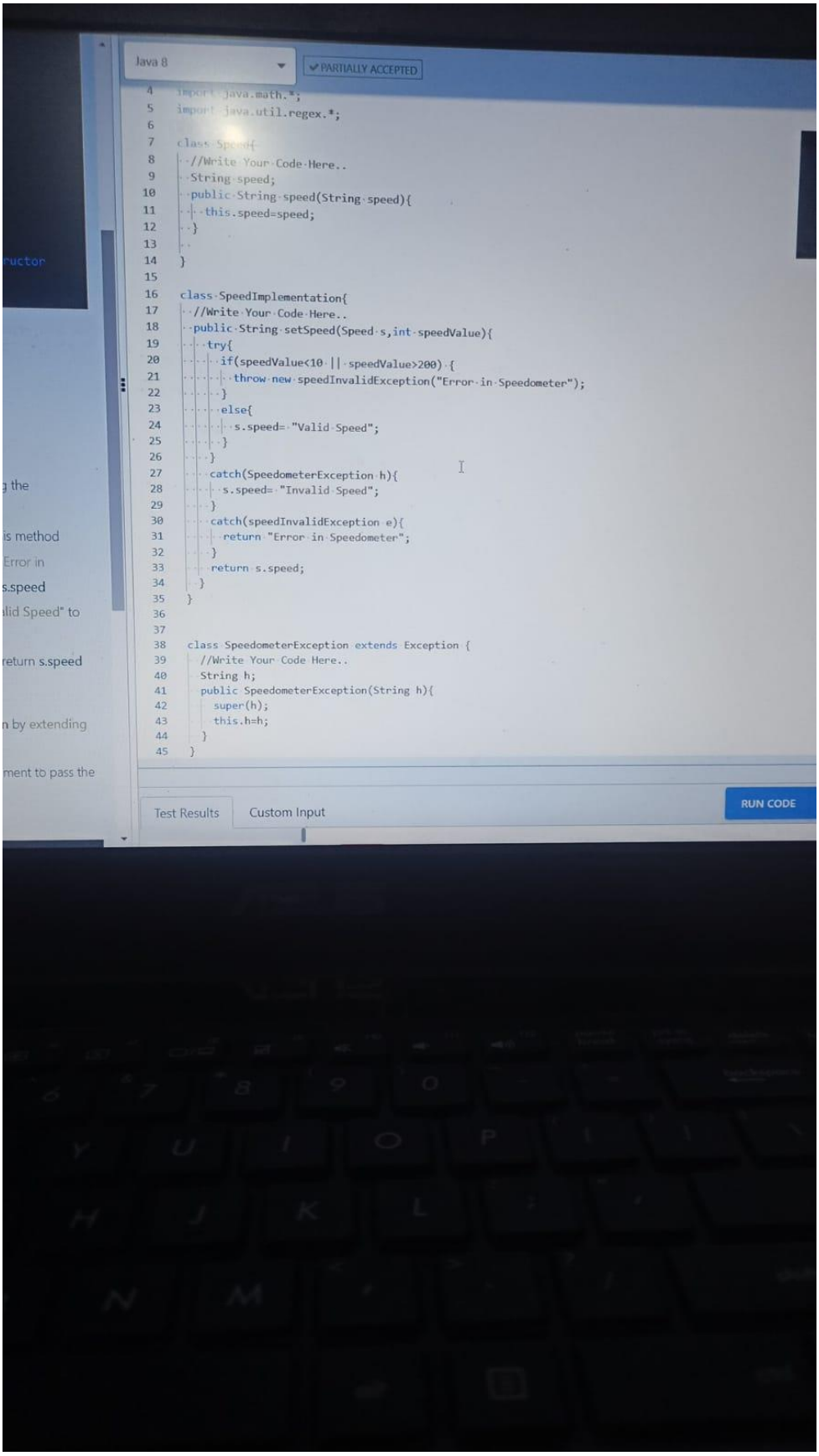
class SpeedImplementation

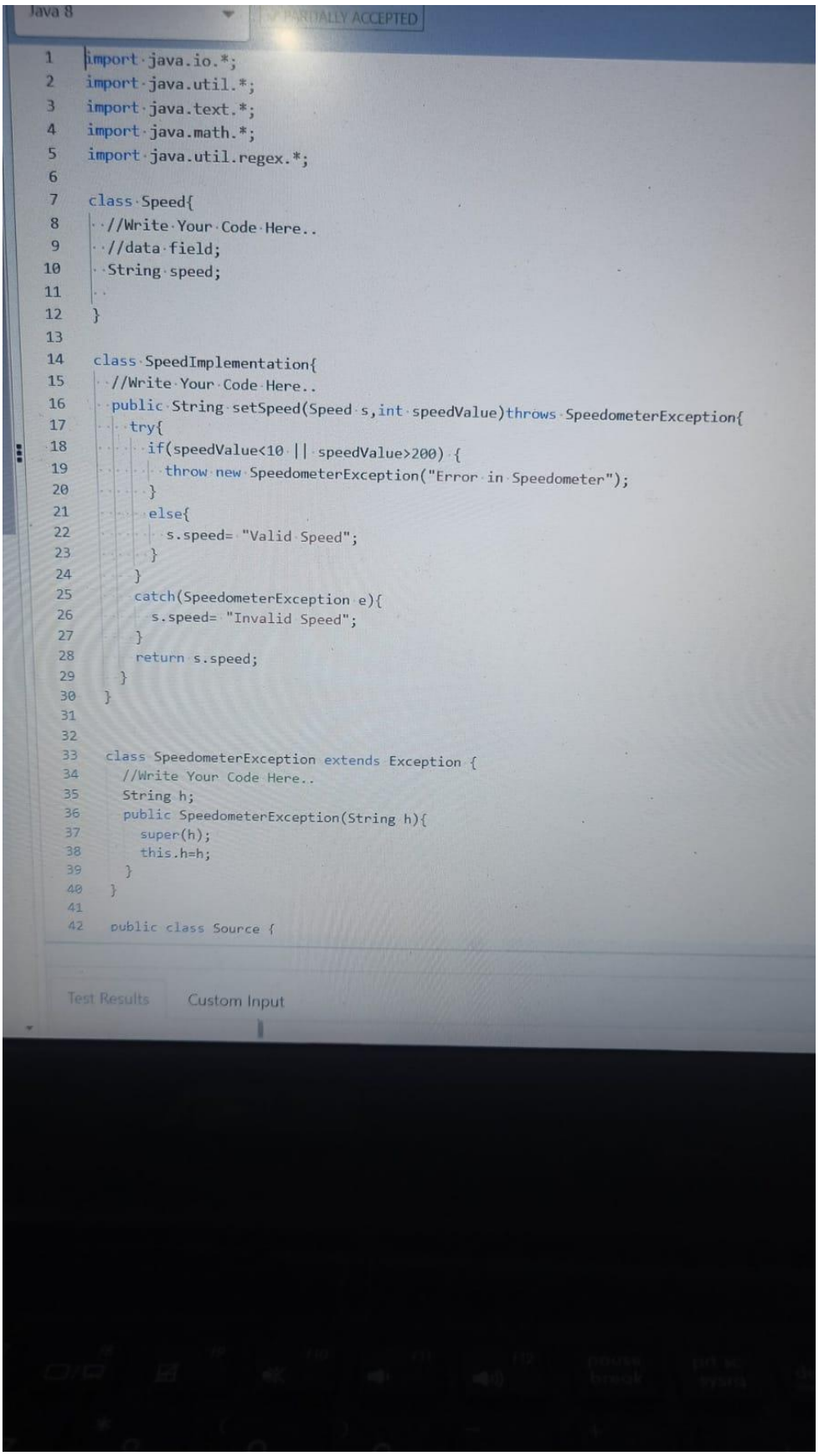
-Implement the below method for this class:

String setSpeed(Speed s, int speedValue):

- Inside this method, you need to write logic for validating the,speedValue.
- If the **speedValue** is less than 10 or greater than 200, this method should throw a **SpeedInvalidException** with message "Error in Speedometer" **else** assign the string "Valid Speed" to s.speed
- catch the thrown exception and assign the string "Invalid Speed" to s.speed


class SpeedometerException extends Exception;
method definitions:
SpeedometerException(String s): Constructor
visibility: public
TASKS:
class Speed
- define the String variable speed
class SpeedImplementation
-Implement the below method for this class:
String setSpeed(Speed s, int speedValue):
Inside this method, you need to write logic for validating the speedValue.
If the speedValue is less than 10 or greater than 200, this method should throw a SpeedInvalidException with message "Error in Speedometer" else assign the string "Valid Speed" to s.speed
catch the thrown exception and assign the string "Invalid Speed" to s.speed
return default message if an exception is thrown, else return s.speed
class SpeedometerException
define custom exception class SpeedometerException by extending the Exception class.
define a parameterized constructor with a String argument to pass the message to the super class.
Sample Input
Speed s = new Speed();
setSpeed(s, 4)
Sample Output
Error in Speedometer
NOTE


Java 8
PARTIALLY ACCEPTED
import java.math.*;
import java.util.regex.*;
class Speed{
//Write Your Code Here..
String speed;
public String speed(String speed){
this.speed=speed;
}
}
class SpeedImplementation{
//Write Your Code Here..
public String setSpeed(Speed s,int speedValue){
try{
if(speedValue<10 || speedValue>200) {
throw new speedInvalidException("Error in Speedometer");
}
else{
s.speed= "Valid Speed";
}
}
catch(SpeedometerException h){
s.speed= "Invalid Speed";
}
catch(speedInvalidException e){
return "Error in Speedometer";
}
return s.speed;
}
}
class SpeedometerException extends Exception {
//Write Your Code Here..
String h;
public SpeedometerException(String h){
super(h);
this.h=h;
}
}
Test Results
Custom Input
RUN CODE



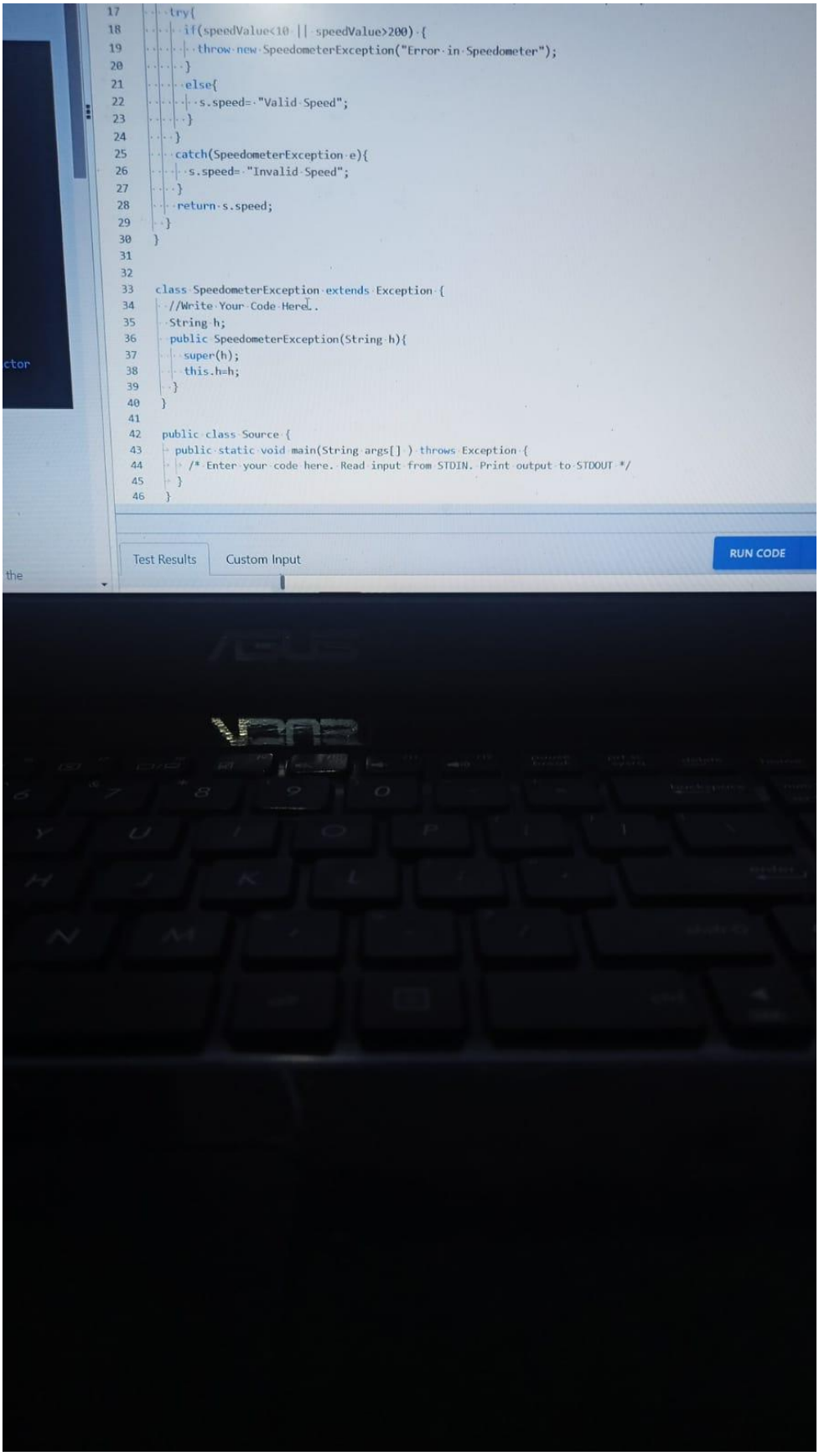
Java 8

PARTIALLY ACCEPTED

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

class Speed{
    //Write Your Code Here..
    //data field;
    String speed;

}

class SpeedImplementation{
    //Write Your Code Here..
    public String setSpeed(Speed s,int speedValue)throws SpeedometerException{
        try{
            if(speedValue<10 || speedValue>200) {
                throw new SpeedometerException("Error in Speedometer");
            }
            else{
                s.speed= "Valid Speed";
            }
        }
        catch(SpeedometerException e){
            s.speed= "Invalid Speed";
        }
        return s.speed;
    }
}


class SpeedometerException extends Exception {
    //Write Your Code Here..
    String h;
    public SpeedometerException(String h){
        super(h);
        this.h=h;
    }
}

public class Source {
```

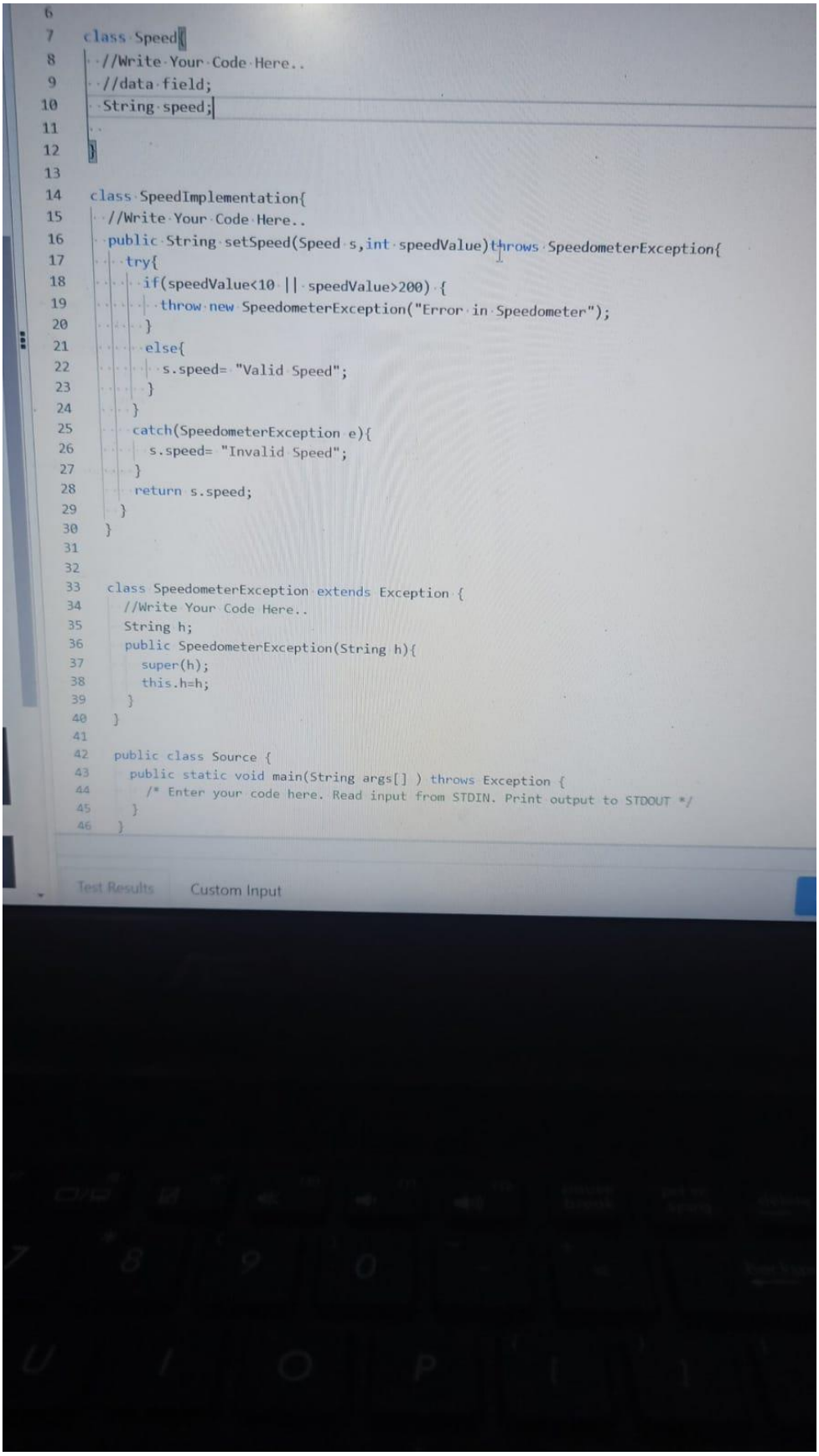
Test Results

Custom Input



```
        try{
            if(speedValue<10 || speedValue>200) {
                throw new SpeedometerException("Error in Speedometer");
            }
            else{
                s.speed= "Valid Speed";
            }
        }
        catch(SpeedometerException e){
            s.speed= "Invalid Speed";
        }
        return s.speed;
    }
}


class SpeedometerException extends Exception {
    //Write Your Code Here..
    String h;
    public SpeedometerException(String h){
        super(h);
        this.h=h;
    }
}

public class Source {
    public static void main(String args[] ) throws Exception {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    }
}
```

Test Results | Custom Input | RUN CODE



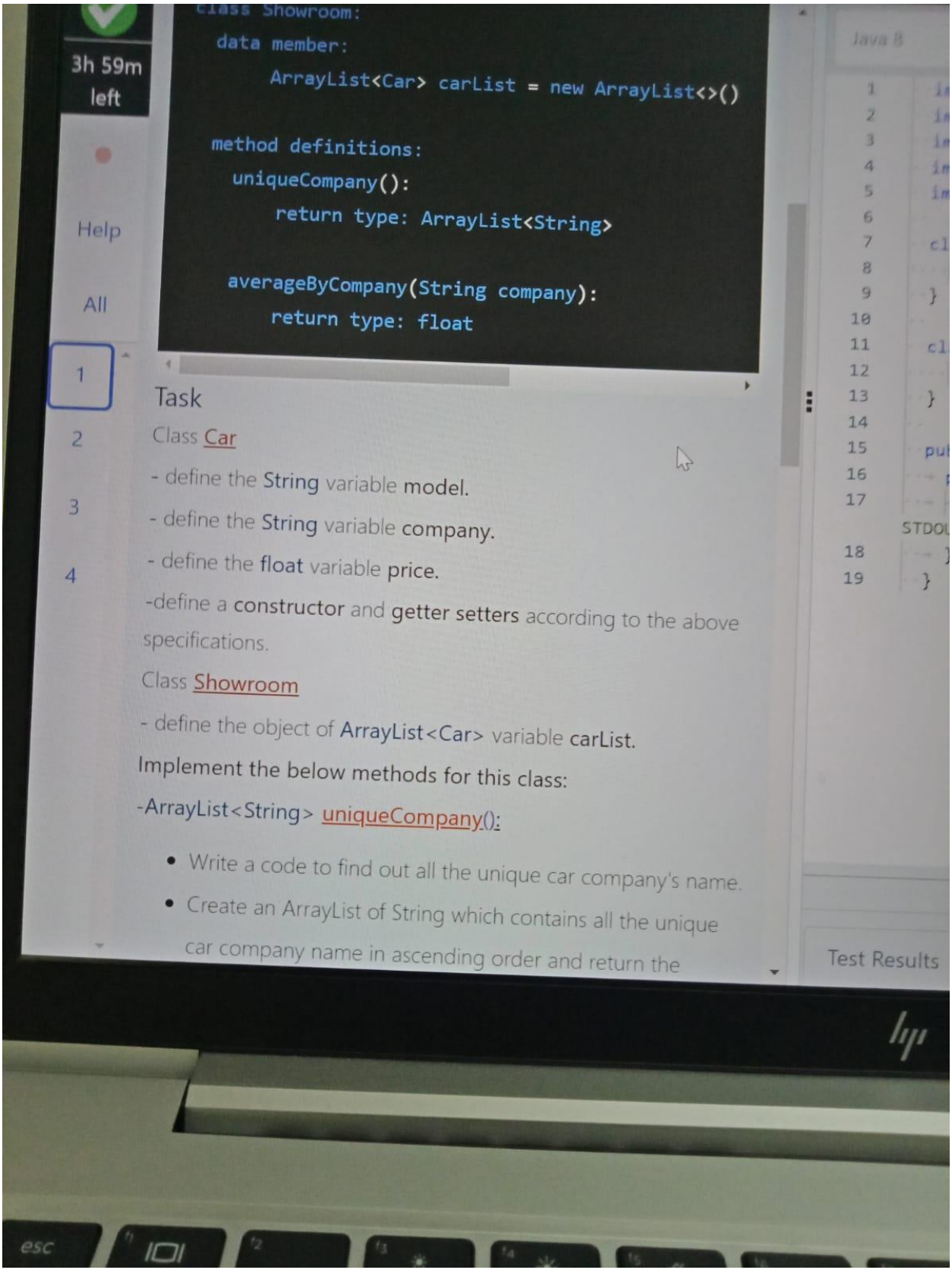
```
class Speed{
  //Write Your Code Here..
  //data field;
  String speed;
  ..
}

class SpeedImplementation{
  //Write Your Code Here..
  public String setSpeed(Speed s,int speedValue)throws SpeedometerException{
    try{
      if(speedValue<10 || speedValue>200) {
        throw new SpeedometerException("Error in Speedometer");
      }
      else{
        s.speed= "Valid Speed";
      }
    }
    catch(SpeedometerException e){
      s.speed= "Invalid Speed";
    }
    return s.speed;
  }
}


class SpeedometerException extends Exception {
  //Write Your Code Here..
  String h;
  public SpeedometerException(String h){
    super(h);
    this.h=h;
  }
}

public class Source {
  public static void main(String args[] ) throws Exception {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
  }
}
```

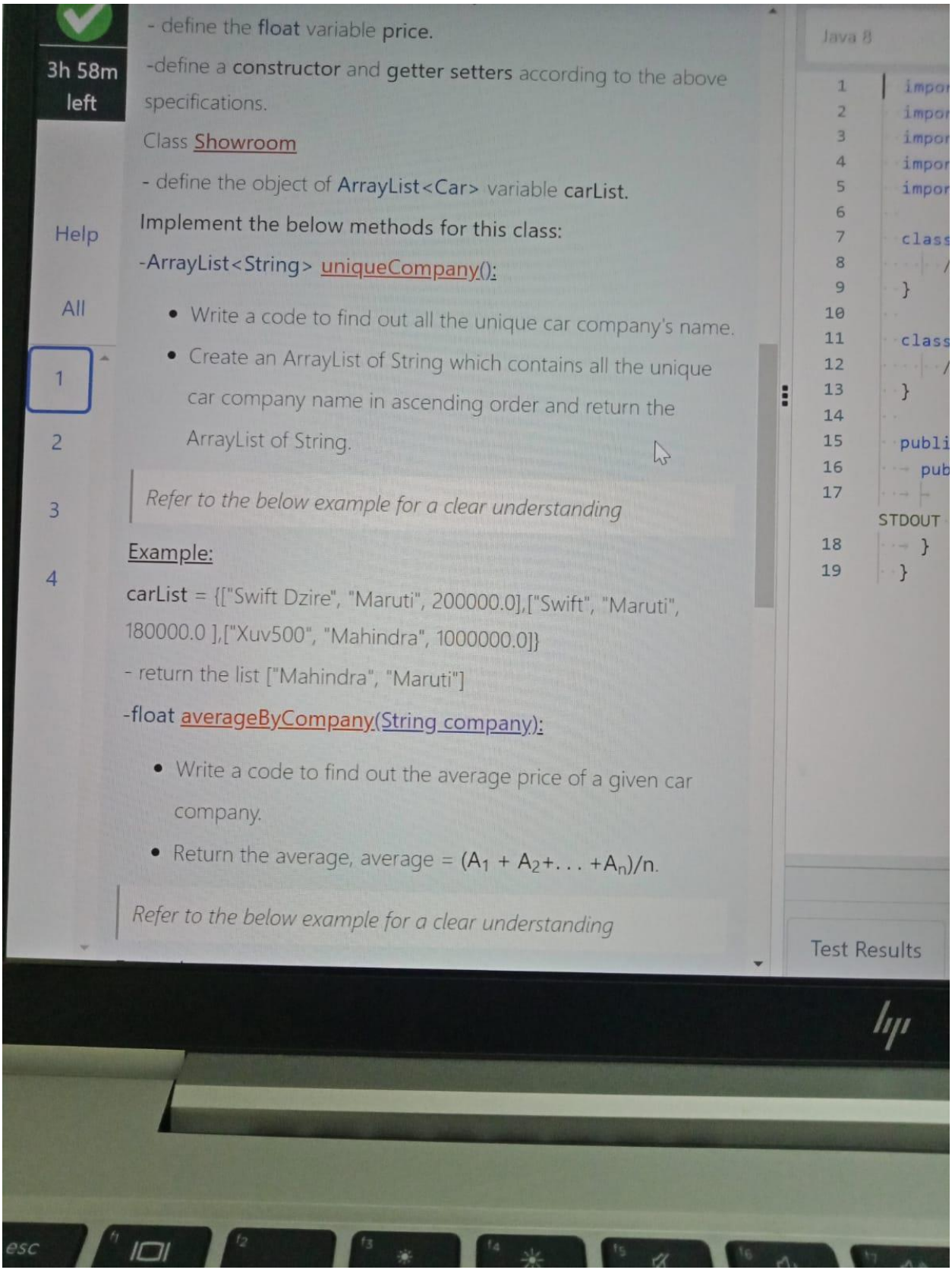
Test Results | Custom Input

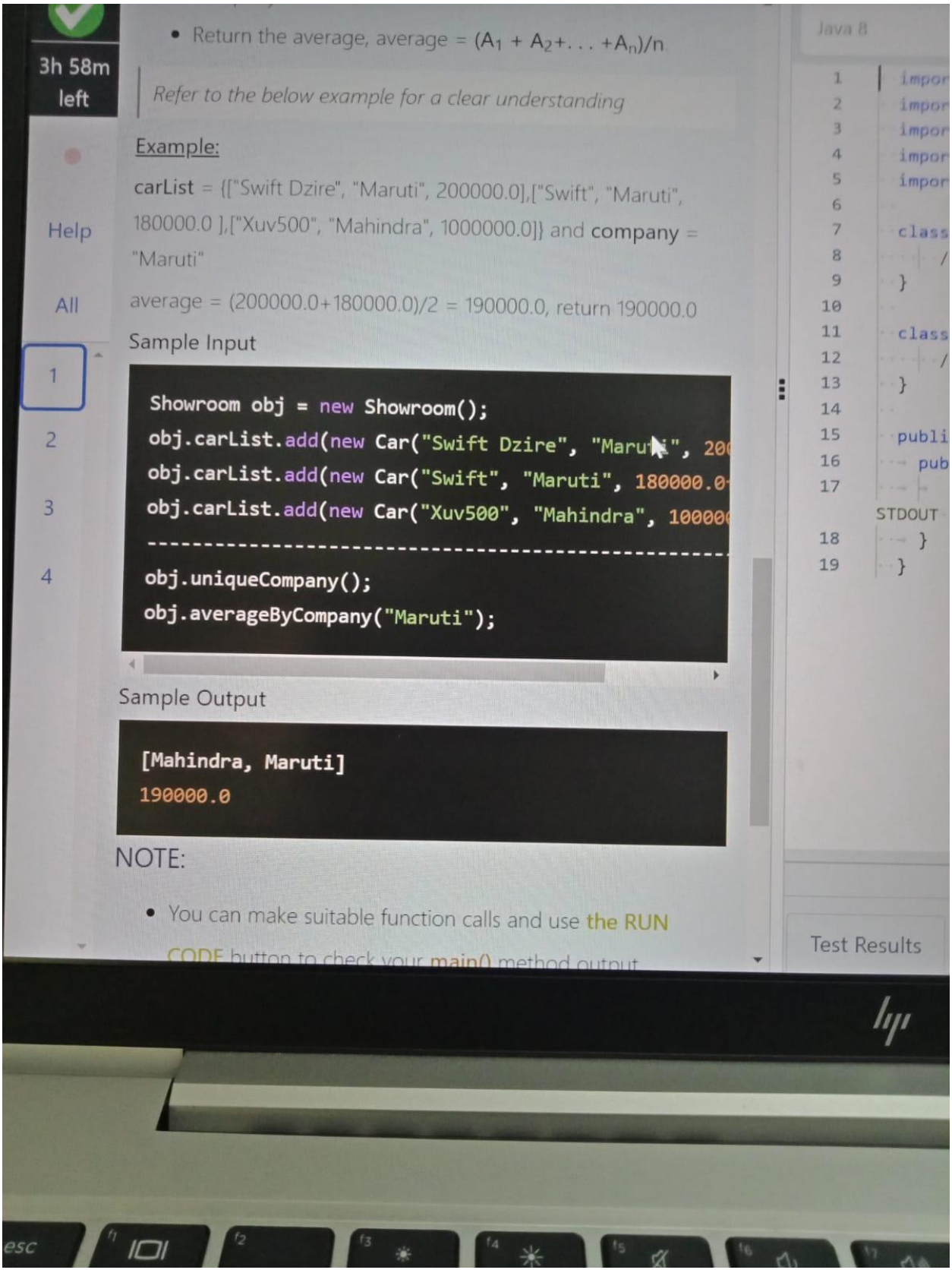
```
import java.util.*;

class Speed {
	String speed;
}
class SpeedInvalidException extends Exception {

	public SpeedInvalidException(String message) {
		super(message);
	}

}

class SpeedometerException extends Exception {

	public SpeedometerException(String message) {
		super(message);
	}

}


class SpeedImplementation {

	String speed;

	public String setSpeed(Speed s, int speedValue)
	{
```

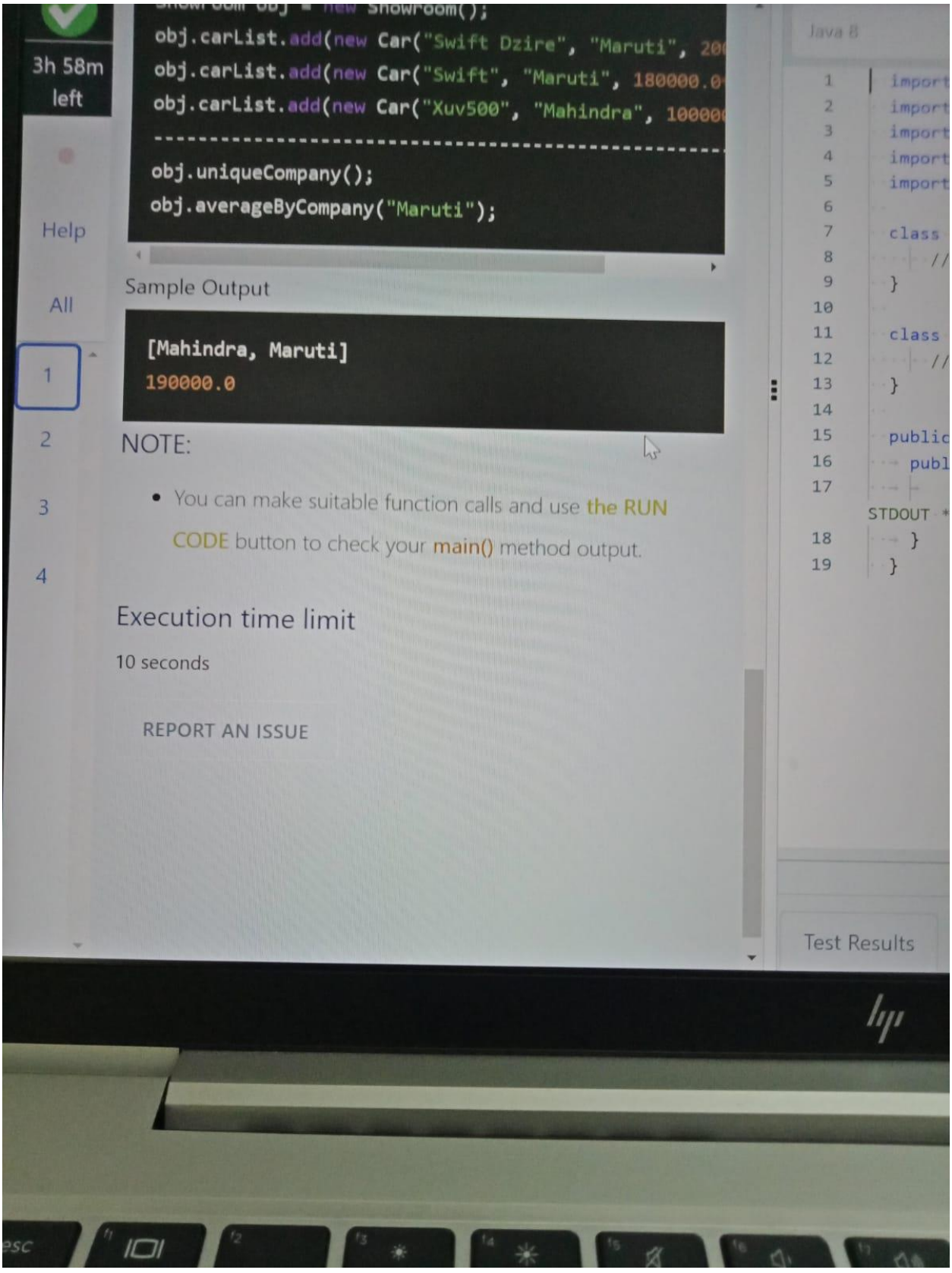
```
		try {
		if (speedValue < 10 || speedValue > 200)
		{
		throw new SpeedInvalidException("Error in Speedometer");
		}
		else {
			speed = "valid speed";
		}
		}
		catch (SpeedInvalidException e)
		{
				s.speed = "Invalid Speed";
				System.out.println(e);
				return s.speed;
		}
		return s.speed;
	}

	public static void main(String[] args) {
		Speed s = new Speed();
		SpeedImplementation obj = new SpeedImplementation();
		obj.setSpeed(s,4);
	}
}
```

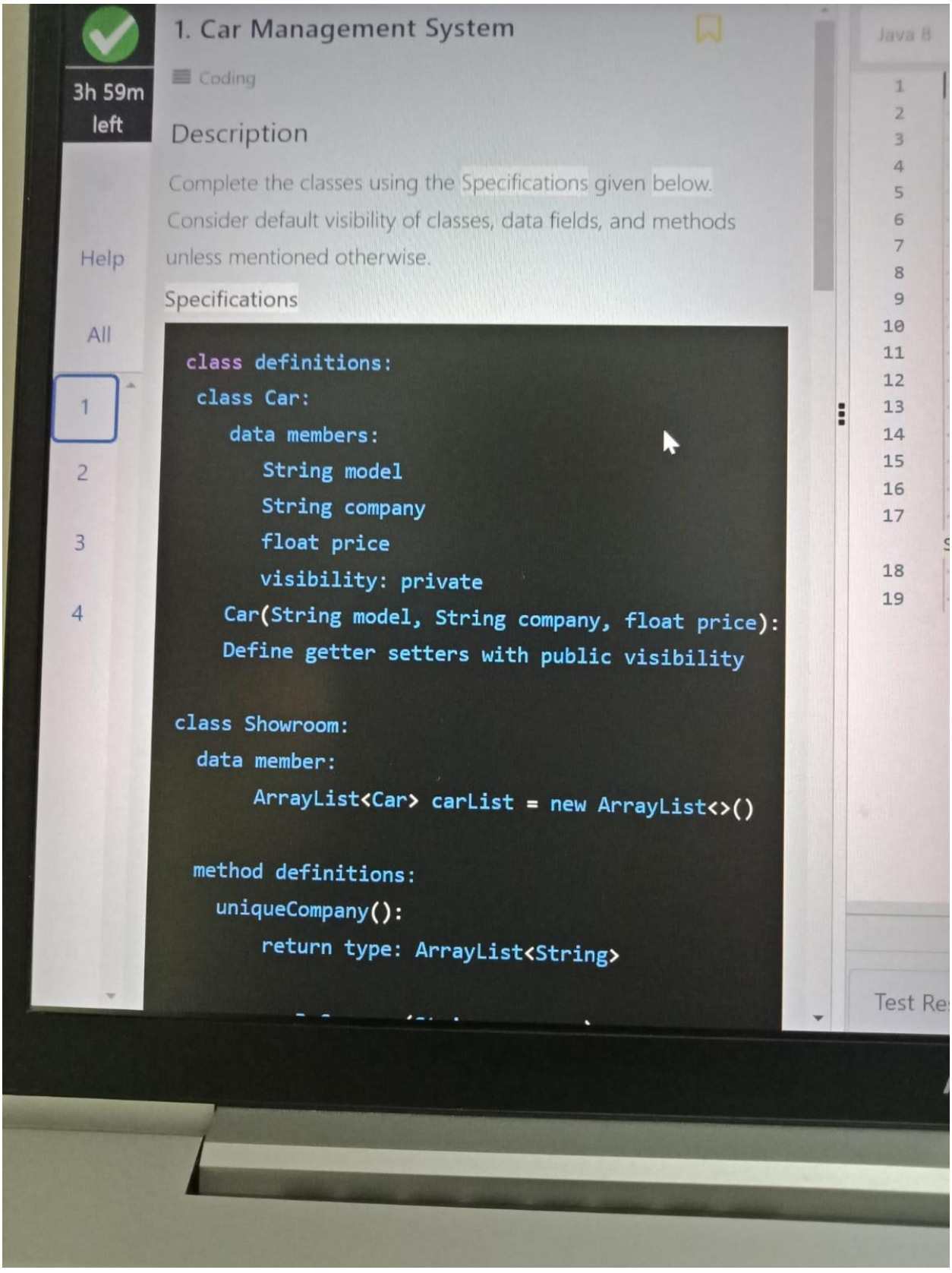
7.


3h 59m left
Help
All
1
2
3
4
class Showroom:
data member:
ArrayList<Car> carList = new ArrayList<>()
method definitions:
uniqueCompany():
return type: ArrayList<String>
averageByCompany(String company):
return type: float
Task
Class Car
- define the String variable model.
- define the String variable company.
- define the float variable price.
-define a constructor and getter setters according to the above specifications.
Class Showroom
- define the object of ArrayList<Car> variable carList.
Implement the below methods for this class:
-ArrayList<String> uniqueCompany();
Write a code to find out all the unique car company's name.
Create an ArrayList of String which contains all the unique car company name in ascending order and return the
Java 8
Test Results
esc


3h 58m left
Help
All
1
2
3
4
- define the float variable price.
-define a constructor and getter setters according to the above specifications.
Class Showroom
- define the object of ArrayList<Car> variable carList.
Implement the below methods for this class:
-ArrayList<String> uniqueCompany();
Write a code to find out all the unique car company's name.
Create an ArrayList of String which contains all the unique car company name in ascending order and return the ArrayList of String.
Refer to the below example for a clear understanding
Example:
carList = {["Swift Dzire", "Maruti", 200000.0],["Swift", "Maruti", 180000.0],["Xuv500", "Mahindra", 1000000.0]}
- return the list ["Mahindra", "Maruti"]
-float averageByCompany(String company);
Write a code to find out the average price of a given car company.
Return the average, average = (A1 + A2+. . . +An)/n.
Refer to the below example for a clear understanding
Java 8
STDOUT
Test Results
esc


3h 58m left
Help
All
1
2
3
4
Return the average, average = (A1 + A2+. . . +An)/n
Refer to the below example for a clear understanding
Example:
carList = {["Swift Dzire", "Maruti", 200000.0],["Swift", "Maruti", 180000.0],["Xuv500", "Mahindra", 1000000.0]} and company = "Maruti"
average = (200000.0+180000.0)/2 = 190000.0, return 190000.0
Sample Input
Showroom obj = new Showroom();
obj.carList.add(new Car("Swift Dzire", "Maruti", 200
obj.carList.add(new Car("Swift", "Maruti", 180000.0
obj.carList.add(new Car("Xuv500", "Mahindra", 100000
obj.uniqueCompany();
obj.averageByCompany("Maruti");
Sample Output
[Mahindra, Maruti]
190000.0
NOTE:
You can make suitable function calls and use the RUN CODE button to check your main() method output
Java 8
STDOUT
Test Results


3h 58m left
Help
All
1
2
3
4
obj.carList.add(new Car("Swift Dzire", "Maruti", 20
obj.carList.add(new Car("Swift", "Maruti", 180000.0
obj.carList.add(new Car("Xuv500", "Mahindra", 10000
obj.uniqueCompany();
obj.averageByCompany("Maruti");
Sample Output
[Mahindra, Maruti]
190000.0
NOTE:
You can make suitable function calls and use the RUN CODE button to check your main() method output.
Execution time limit
10 seconds
REPORT AN ISSUE
Java 8
STDOUT
Test Results


1. Car Management System
Coding
3h 59m left
Description
Complete the classes using the Specifications given below.
Consider default visibility of classes, data fields, and methods unless mentioned otherwise.
Help
All
Specifications
1
2
3
4
class definitions:
class Car:
data members:
String model
String company
float price
visibility: private
Car(String model, String company, float price):
Define getter setters with public visibility
class Showroom:
data member:
ArrayList<Car> carList = new ArrayList<>()
method definitions:
uniqueCompany():
return type: ArrayList<String>
Java 8
Test Re