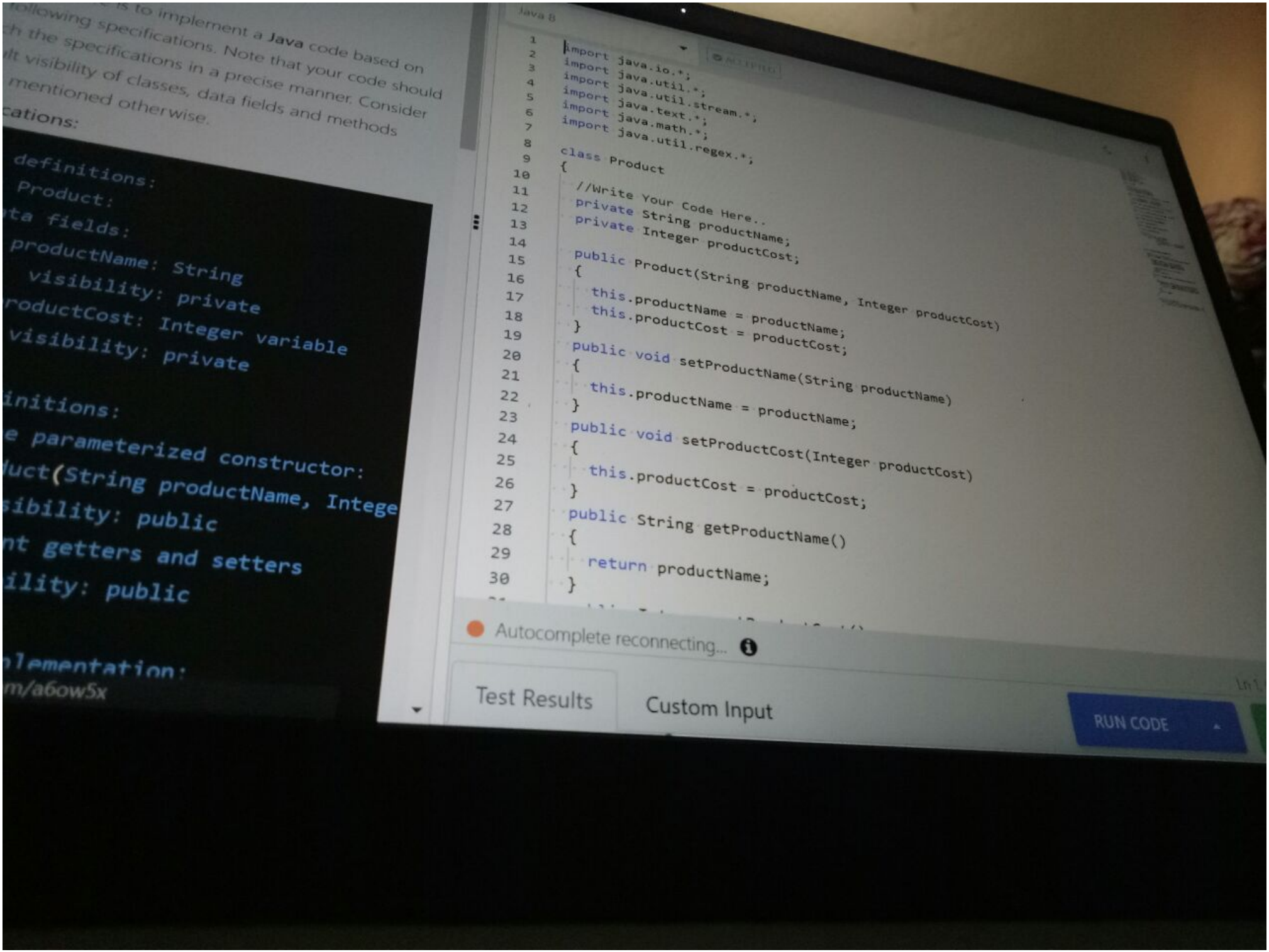
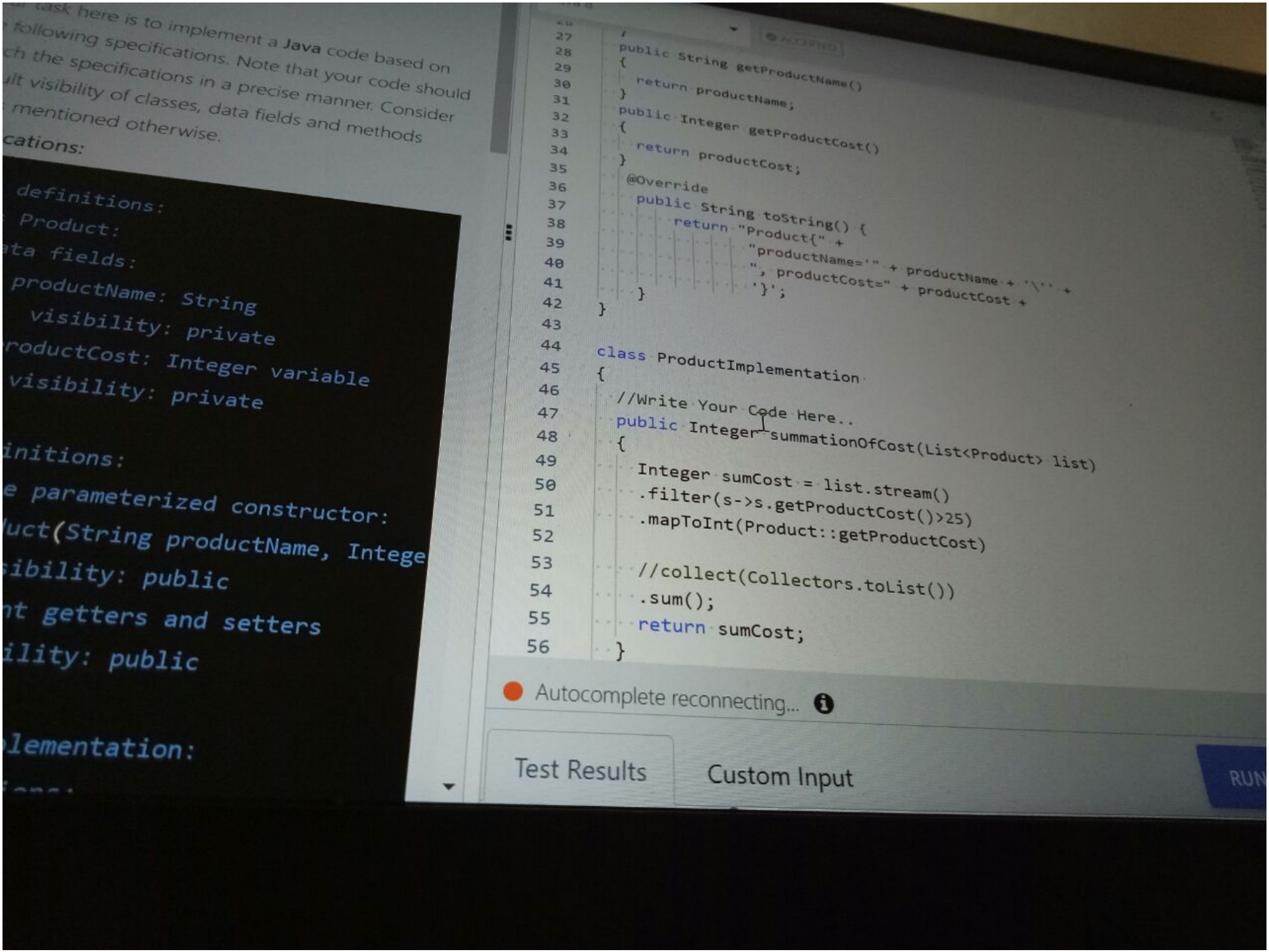
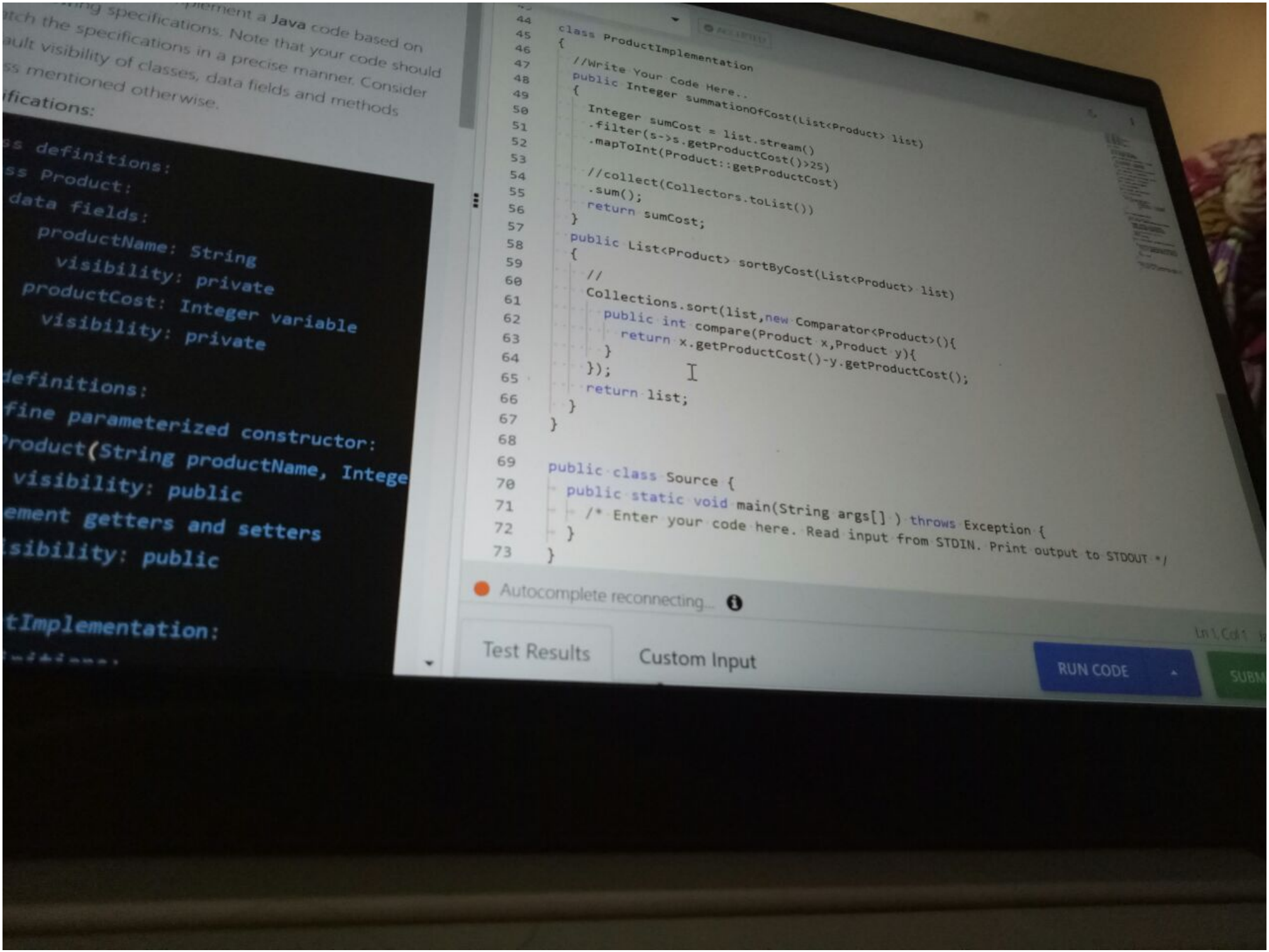

is to implement a Java code based on
following specifications. Note that your code should
ch the specifications in a precise manner. Consider
lt visibility of classes, data fields and methods
mentioned otherwise.
cations:
definitions:
Product:
ta fields:
productName: String
visibility: private
roductCost: Integer variable
visibility: private
initions:
e parameterized constructor:
uct(String productName, Intege
sibility: public
nt getters and setters
ility: public
lementation:
m/a6ow5x
Java 8
import java.io.*;
import java.util.*;
import java.util.stream.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
class Product
{
//Write Your Code Here..
private String productName;
private Integer productCost;
public Product(String productName, Integer productCost)
{
this.productName = productName;
this.productCost = productCost;
}
public void setProductName(String productName)
{
this.productName = productName;
}
public void setProductCost(Integer productCost)
{
this.productCost = productCost;
}
public String getProductName()
{
return productName;
}
Autocomplete reconnecting...
Test Results
Custom Input
RUN CODE


task here is to implement a Java code based on
following specifications. Note that your code should
ch the specifications in a precise manner. Consider
lt visibility of classes, data fields and methods
mentioned otherwise.
cations:
definitions:
Product:
ta fields:
productName: String
visibility: private
roductCost: Integer variable
visibility: private
initions:
e parameterized constructor:
luct(String productName, Intege
sibility: public
nt getters and setters
ility: public
lementation:
public String getProductName()
{
return productName;
}
public Integer getProductCost()
{
return productCost;
}
@Override
public String toString() {
return "Product{" +
"productName='" + productName + '\'' +
", productCost=" + productCost +
'}';
}
}
class ProductImplementation
{
//Write Your Code Here..
public Integer summationOfCost(List<Product> list)
{
Integer sumCost = list.stream()
.filter(s->s.getProductCost()>25)
.mapToInt(Product::getProductCost)
//collect(Collectors.toList())
.sum();
return sumCost;
}
Autocomplete reconnecting...
Test Results
Custom Input
RUN



```
class ProductImplementation
{
    //Write Your Code Here..
    public Integer summationOfCost(List<Product> list)
    {
        Integer sumCost = list.stream()
        .filter(s->s.getProductCost()>25)
        .mapToInt(Product::getProductCost)
        //collect(Collectors.toList())
        .sum();
        return sumCost;
    }
    public List<Product> sortByCost(List<Product> list)
    {
        //
        Collections.sort(list,new Comparator<Product>(){
            public int compare(Product x,Product y){
                return x.getProductCost()-y.getProductCost();
            }
        });
        return list;
    }
}

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

Autocomplete reconnecting...

Test Results

Custom Input

RUN CODE