Cycle Savvy

Grade settings: Maximum grade: 100
Disable external file upload, paste and drop external content: Yes
Based on: Cycle Savvy
Run: Yes **Evaluate**: Yes
Automatic grade: Yes

"Cycle Savvy" was a well-known cycle shop located in the heart of the city. The shop had a vast range of bicycles, catering to the needs of all types of cycling enthusiasts. They had five types of cycles - road, mountain, hybrid, touring, and electric cycles. They wanted to calculate the bill amount including the tax. An extra amount of Rs 1000 will be charged for any accessories if they are required. The manager contacts a software developer for assistance with their process. You are the software developer, and you are creating a Java program based on the requirements.

Component Specification: CycleInfo

Type (Class)	Attributes	Methods
CycleInfo	String cycleId String cycleName String cycleType String accessories double price	Necessary getters, setters, and five-argument constructor are provided as part of the code skeleton.

Functional Requirement 1: Extract the details of the cycle and create an object of CycleInfo class.

Type (Class)	Methods	Responsibilities
UserInterface	public static CycleInfo **extractDetails**(String cycleDetails)	This method accepts **cycleDetails** separated by a colon as an argument and should extract the properties of the **CycleInfo** from the argument by parsing

		the **cycleDetails**. Set these values to the **CycleInfo** object and return this object.

Functional Requirement 2: Calculate the bill amount to be paid by the customer, including the tax percentage.

Type (Class)	**Methods**	**Responsibilities**
CycleInfo	public double **calculateBillAmount**()	This method is used to calculate the bill amount to be paid by the customer, including the tax percentage and the accessories (if required). If the **cycleType** is **"Road Cycle"**, the tax is 2% of the price. If the **cycleType** is **"Mountain Cycle"**, the tax is 5% of the price. If the **cycleType** is **"Hybrid Cycle"**, the tax is 6% of the price. If the **cycleType** is **"Touring Cycle"**, the tax is 4% of the price. If the **cycleType** is **"Electric Cycle"**, the tax is 3% of the price. ***Condition****:* • ***CycleType*** *is case-sensitive.* • *If the* ***cycleType*** *does not match any of the above types, return -1.*

		• ***Accessories** are case-insensitive.* • *If the **accessories** is other than "**yes**" or "**no**", return -1.* • *If the **price** is less than or equal to 0, return -1.*

Formula to calculate the bill amount to be paid:
Bill amount= price+(price* tax% (based on the cycleType)/100) + 1000(if accessories required)

Example: 1

Let the cycleType be "**Hybrid Cycle**", so the tax% is **6**, the price is **13000** and if accessories is "**yes**"

Bill amount =13000+((13000*6)/100)=(13000+780)=13780+1000=14780.0

Example: 2

Let the cycleType be "**Hybrid Cycle**", so the tax% is **6**, the price is **13000** and if accessories is "**no**"

Bill amount= = 13000+((13000*6)/100)=(13000+780)=13780.0

The main method in the UserInterface class is excluded from the evaluation. You are free to write your own code in the main method to invoke the business methods to check its correctness.

Note:

- In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
- Ensure to follow the object-oriented specifications provided in the question.
- Ensure to provide the names for classes, attributes, and methods as specified in the question.
- Adhere to the code template, if provided.

Sample Input/Output 1:

Enter the Cycle details

CYCAS345:Hi-Fast Gangster x89:Road Cycle:yes:24000

Cycle Details

Cycle ID: CYCAS345

Cycle Name: Hi-Fast Gangster x89

Cycle Type: Road Cycle

Wanted Accessories: yes

Cycle Price: 24000.0

Bill amount to be paid: 25480.0

Sample Input/Output 2:

Enter the Cycle details

CYCMO567:FireFox Kreed 27.5D:Mountain Cycle:no:17900

Cycle Details

Cycle ID: CYCMO567

Cycle Name: FireFox Kreed 27.5D

Cycle Type: Mountain Cycle

Wanted Accessories: no

Cycle Price: 17900.0

Bill amount to be paid: 18795.0

Sample Input/Output 3:

Enter the Cycle details

CYCFL908:LifeLong Falcon678:Folding Cycle:yes:18890

Invalid Cycle details

[**Explanation:** As the **cycleType** is not one of those mentioned, the input is considered as invalid.]

Sample Input/Output 4:

Enter the Cycle details

CYCHU908:LifeLong Falcon678:Hybrid Cycle:yes:-18890

Invalid Cycle details

[**Explanation:** As the **price** is negative, the input is considered as invalid.]

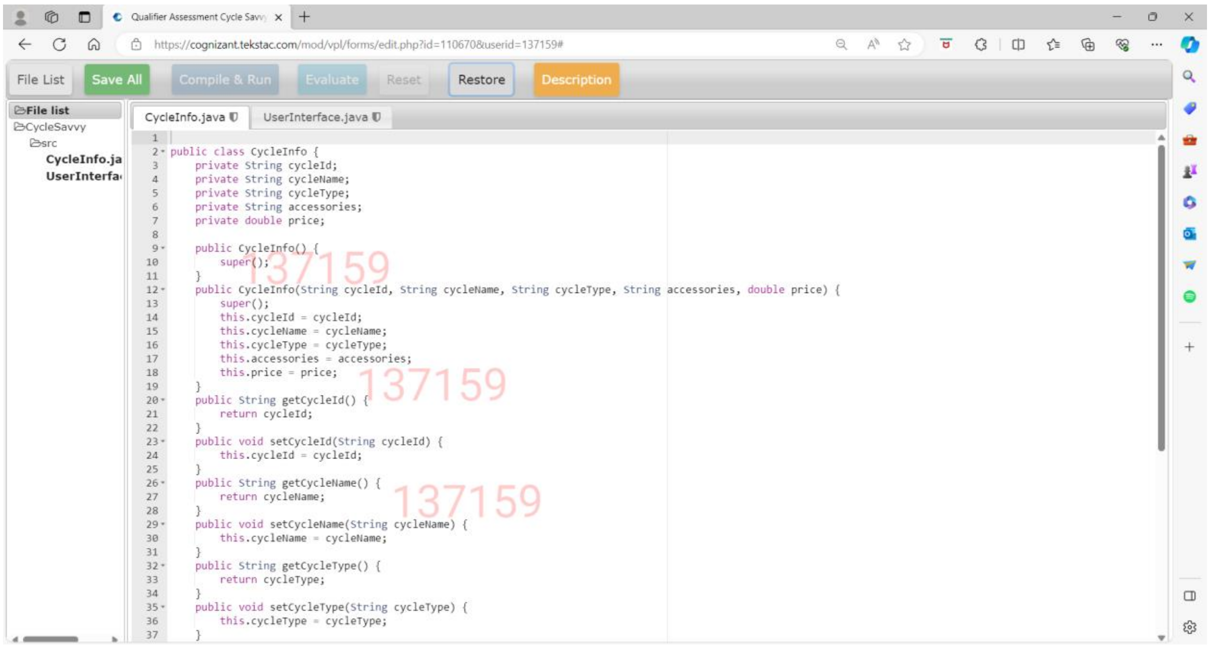
Sample Input/Output 5:

Enter the Cycle details

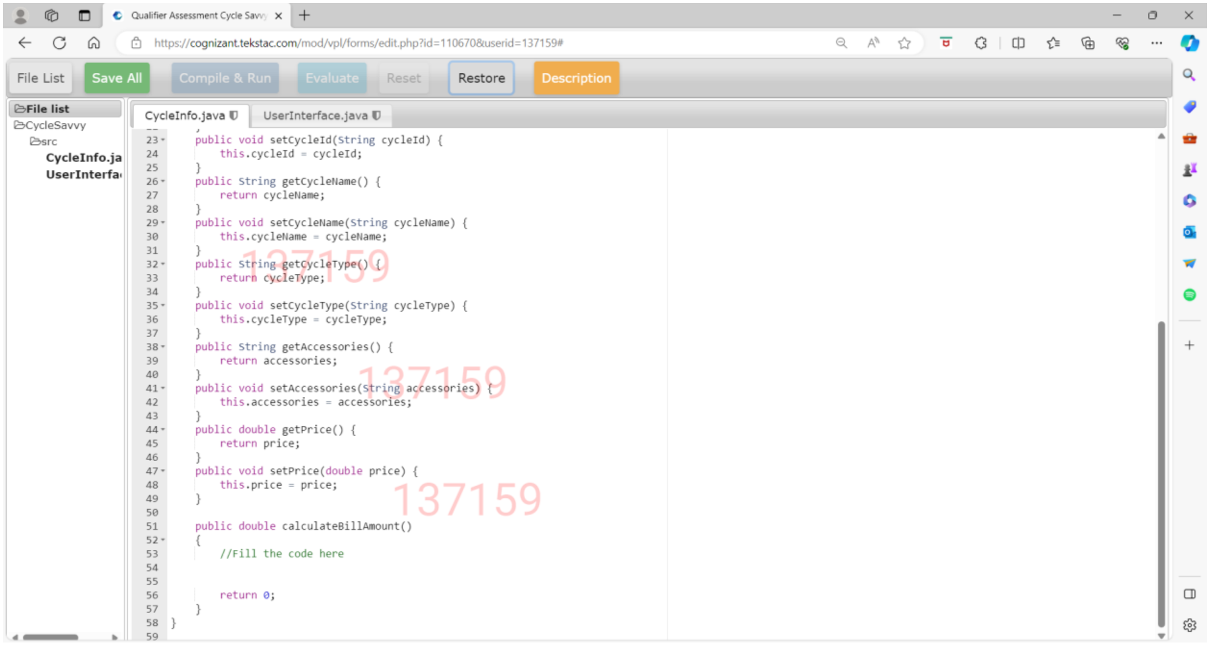
CYCEL908:LifeLong Falcon678:Electric Cycle:y:8890

Invalid Cycle details

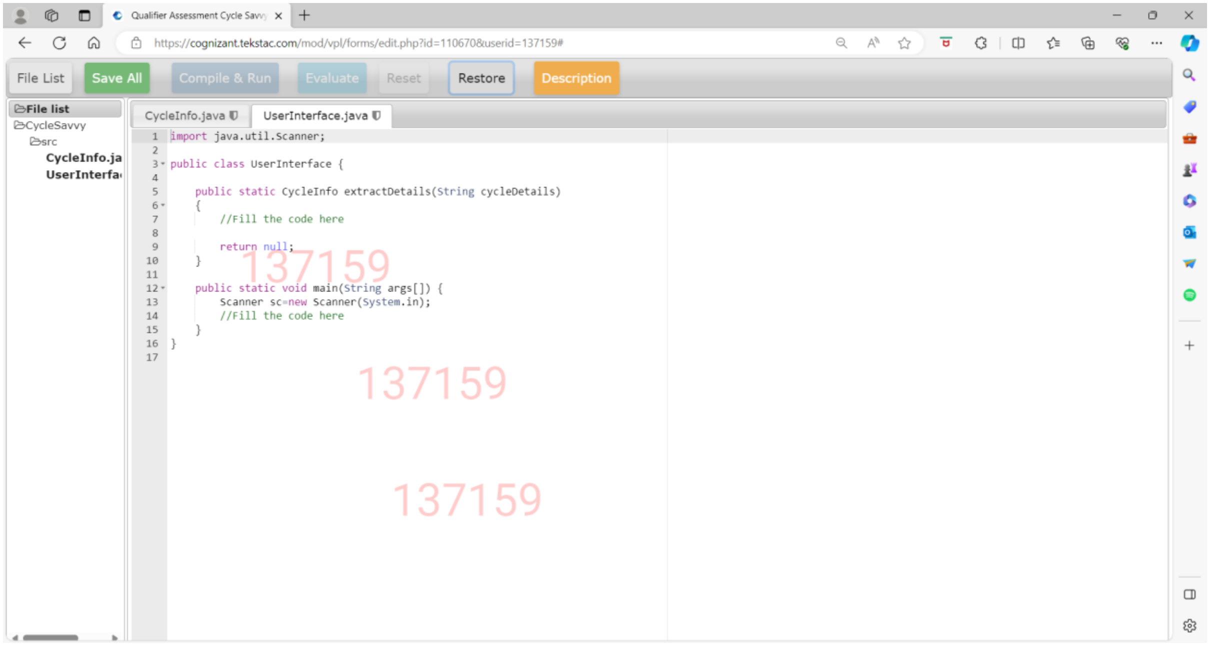
[**Explanation:** As the **accessories** is not one of those mentioned, the input is considered as invalid.]



```java

public class CycleInfo {
    private String cycleId;
    private String cycleName;
    private String cycleType;
    private String accessories;
    private double price;

    public CycleInfo() {
        super();
    }
    public CycleInfo(String cycleId, String cycleName, String cycleType, String accessories, double price) {
        super();
        this.cycleId = cycleId;
        this.cycleName = cycleName;
        this.cycleType = cycleType;
        this.accessories = accessories;
        this.price = price;
    }
    public String getCycleId() {
        return cycleId;
    }
    public void setCycleId(String cycleId) {
        this.cycleId = cycleId;
    }
    public String getCycleName() {
        return cycleName;
    }
    public void setCycleName(String cycleName) {
        this.cycleName = cycleName;
    }
    public String getCycleType() {
        return cycleType;
    }
    public void setCycleType(String cycleType) {
        this.cycleType = cycleType;
    }
```



```java
    public void setCycleId(String cycleId) {
        this.cycleId = cycleId;
    }
    public String getCycleName() {
        return cycleName;
    }
    public void setCycleName(String cycleName) {
        this.cycleName = cycleName;
    }
    public String getCycleType() {
        return cycleType;
    }
    public void setCycleType(String cycleType) {
        this.cycleType = cycleType;
    }
    public String getAccessories() {
        return accessories;
    }
    public void setAccessories(String accessories) {
        this.accessories = accessories;
    }
    public double getPrice() {
        return price;
    }
    public void setPrice(double price) {
        this.price = price;
    }

    public double calculateBillAmount()
    {
        //Fill the code here


        return 0;
    }
}
```



```java
import java.util.Scanner;

public class UserInterface {

    public static CycleInfo extractDetails(String cycleDetails)
    {
        //Fill the code here

        return null;
    }

    public static void main(String args[]) {
        Scanner sc=new Scanner(System.in);
        //Fill the code here
    }
}
```