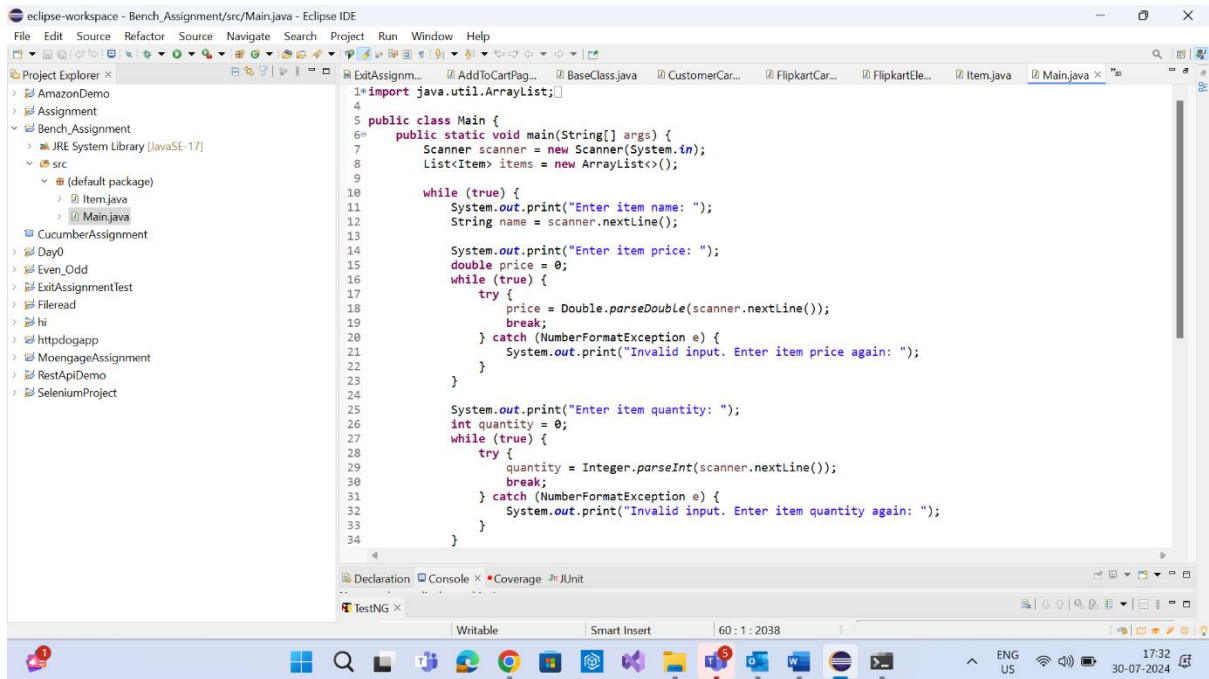


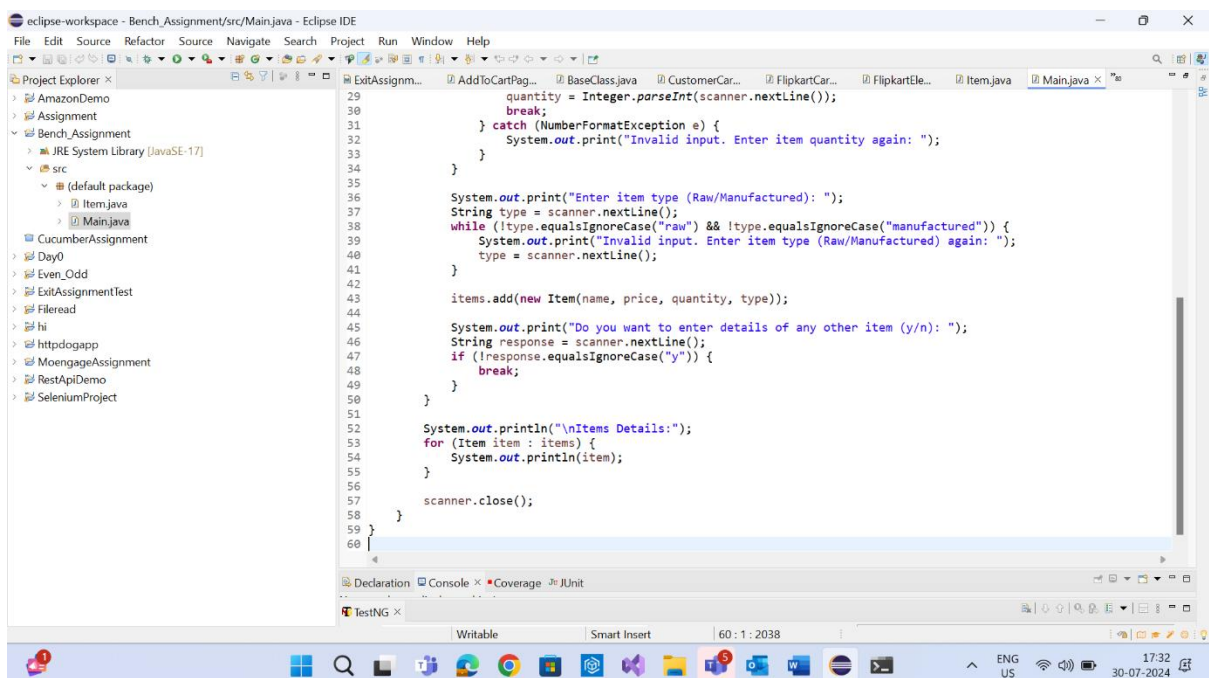
# Java Assignment

## Question 1.

### Main.java

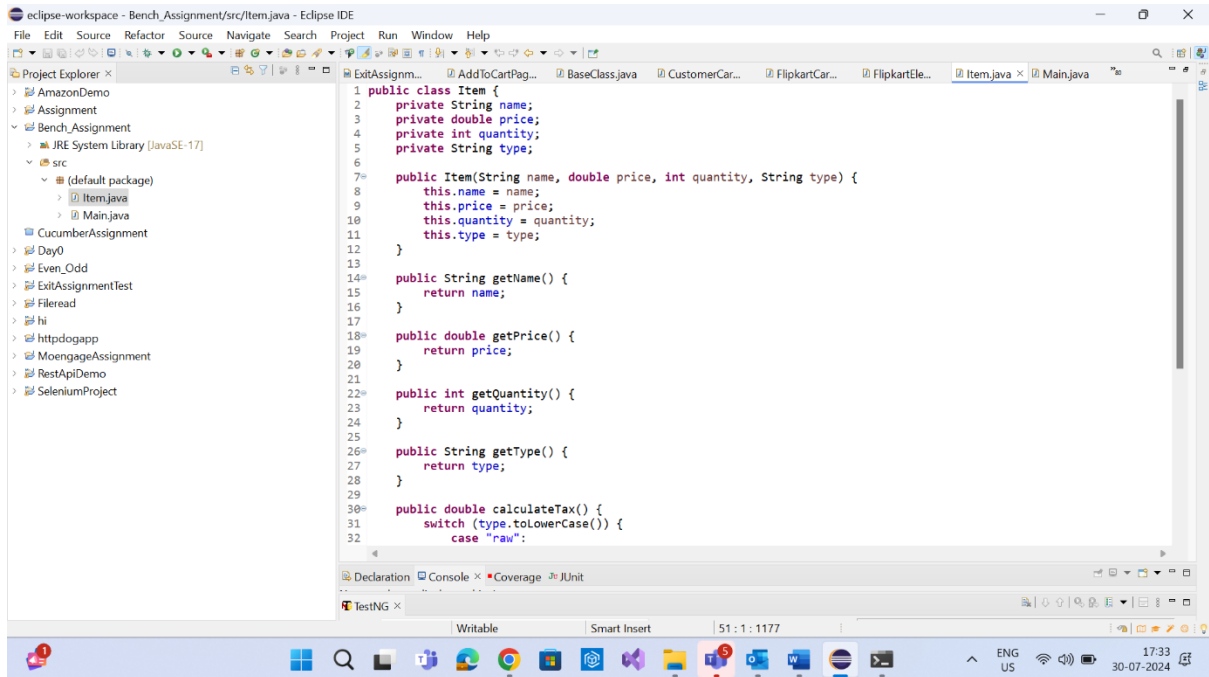


```
1 import java.util.ArrayList;
2
3 public class Main {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         List<Item> items = new ArrayList<>();
7
8         while (true) {
9             System.out.print("Enter item name: ");
10            String name = scanner.nextLine();
11
12            System.out.print("Enter item price: ");
13            double price = 0;
14            while (true) {
15                try {
16                    price = Double.parseDouble(scanner.nextLine());
17                    break;
18                } catch (NumberFormatException e) {
19                    System.out.print("Invalid input. Enter item price again: ");
20                }
21            }
22
23            System.out.print("Enter item quantity: ");
24            int quantity = 0;
25            while (true) {
26                try {
27                    quantity = Integer.parseInt(scanner.nextLine());
28                    break;
29                } catch (NumberFormatException e) {
30                    System.out.print("Invalid input. Enter item quantity again: ");
31                }
32            }
33
34            items.add(new Item(name, price, quantity));
35        }
36    }
37}
```



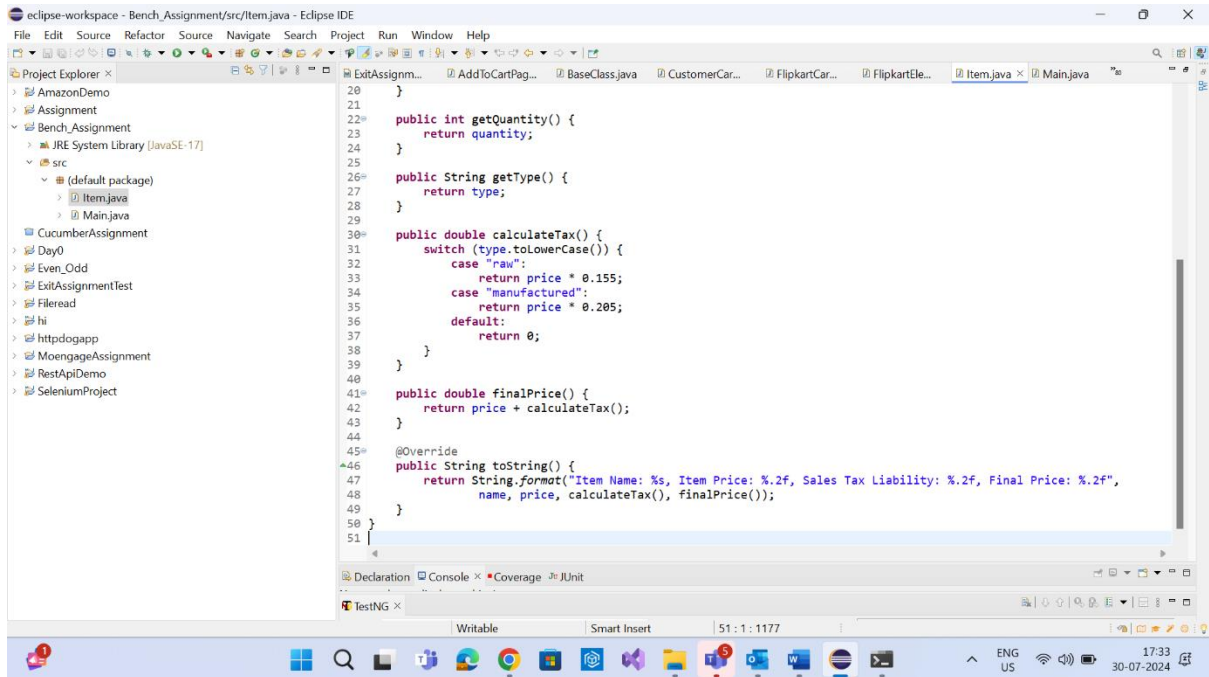
```
29
30 quantity = Integer.parseInt(scanner.nextLine());
31 break;
32 } catch (NumberFormatException e) {
33     System.out.print("Invalid input. Enter item quantity again: ");
34 }
35
36 System.out.print("Enter item type (Raw/Manufactured): ");
37 String type = scanner.nextLine();
38 while (!type.equalsIgnoreCase("raw") && !type.equalsIgnoreCase("manufactured")) {
39     System.out.print("Invalid input. Enter item type (Raw/Manufactured) again: ");
40     type = scanner.nextLine();
41 }
42
43 items.add(new Item(name, price, quantity, type));
44
45 System.out.print("Do you want to enter details of any other item (y/n): ");
46 String response = scanner.nextLine();
47 if (!response.equalsIgnoreCase("y")) {
48     break;
49 }
50
51 }
52
53 System.out.println("\nItems Details:");
54 for (Item item : items) {
55     System.out.println(item);
56 }
57
58 scanner.close();
59
60 }
```

# Item.java



The screenshot shows the Eclipse IDE with the 'Item.java' file open. The code defines a class 'Item' with private attributes 'name', 'price', 'quantity', and 'type'. It includes a constructor and four getter methods. The 'calculateTax()' method uses a switch statement with a 'raw' case.

```
1 public class Item {
2     private String name;
3     private double price;
4     private int quantity;
5     private String type;
6
7     public Item(String name, double price, int quantity, String type) {
8         this.name = name;
9         this.price = price;
10        this.quantity = quantity;
11        this.type = type;
12    }
13
14    public String getName() {
15        return name;
16    }
17
18    public double getPrice() {
19        return price;
20    }
21
22    public int getQuantity() {
23        return quantity;
24    }
25
26    public String getType() {
27        return type;
28    }
29
30    public double calculateTax() {
31        switch (type.toLowerCase()) {
32            case "raw":
```



The screenshot shows the Eclipse IDE with the 'Item.java' file open, displaying the completed code. The 'calculateTax()' method is now fully implemented with cases for 'raw' (0.155), 'manufactured' (0.205), and a default case (0). A 'finalPrice()' method is added, and the 'toString()' method is implemented using 'String.format()'.

```
20    }
21
22    public int getQuantity() {
23        return quantity;
24    }
25
26    public String getType() {
27        return type;
28    }
29
30    public double calculateTax() {
31        switch (type.toLowerCase()) {
32            case "raw":
33                return price * 0.155;
34            case "manufactured":
35                return price * 0.205;
36            default:
37                return 0;
38        }
39    }
40
41    public double finalPrice() {
42        return price + calculateTax();
43    }
44
45    @Override
46    public String toString() {
47        return String.format("Item Name: %s, Item Price: %.2f, Sales Tax Liability: %.2f, Final Price: %.2f",
48            name, price, calculateTax(), finalPrice());
49    }
50 }
51
```

## How to run ?

To run the code – Open a terminal and navigate to the directory where **Item.java** and **Main.java** are saved.

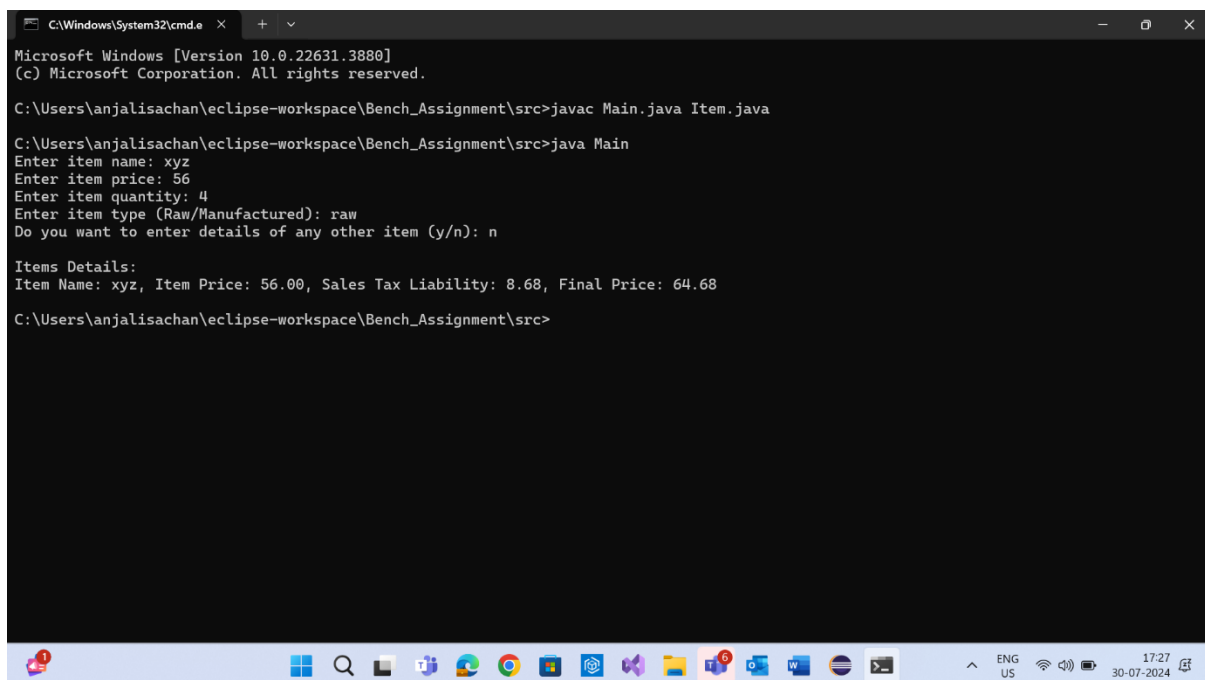
**Compile the program using the command:**

```
javac Main.java Item.java
```

**Run the program using the command:**

```
java Main
```

## Output :



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22631.3880]
(c) Microsoft Corporation. All rights reserved.

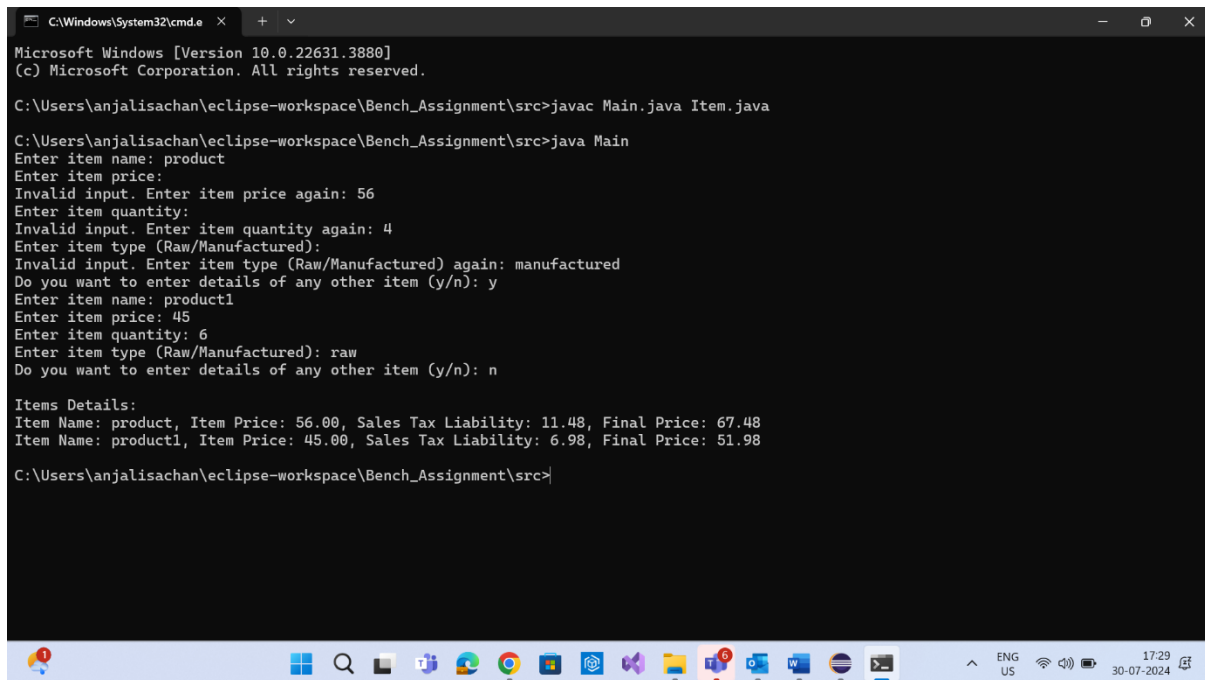
C:\Users\anjalisachan\workspace\Bench_Assignment\src>javac Main.java Item.java

C:\Users\anjalisachan\workspace\Bench_Assignment\src>java Main
Enter item name: xyz
Enter item price: 56
Enter item quantity: 4
Enter item type (Raw/Manufactured): raw
Do you want to enter details of any other item (y/n): n

Items Details:
Item Name: xyz, Item Price: 56.00, Sales Tax Liability: 8.68, Final Price: 64.68

C:\Users\anjalisachan\workspace\Bench_Assignment\src>
```

## Handling Exception:



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22631.3880]
(c) Microsoft Corporation. All rights reserved.

C:\Users\anjanisachan\workspace\Bench_Assignment\src>javac Main.java Item.java

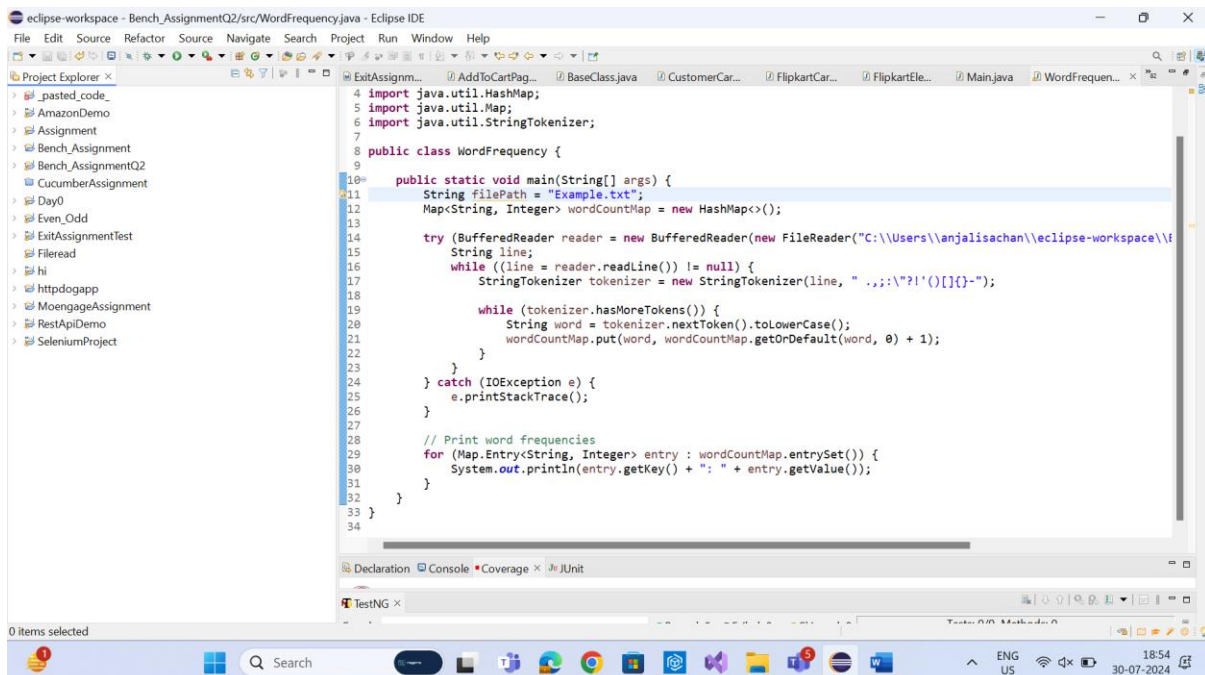
C:\Users\anjanisachan\workspace\Bench_Assignment\src>java Main
Enter item name: product
Enter item price:
Invalid input. Enter item price again: 56
Enter item quantity:
Invalid input. Enter item quantity again: 4
Enter item type (Raw/Manufactured):
Invalid input. Enter item type (Raw/Manufactured) again: manufactured
Do you want to enter details of any other item (y/n): y
Enter item name: product1
Enter item price: 45
Enter item quantity: 6
Enter item type (Raw/Manufactured): raw
Do you want to enter details of any other item (y/n): n

Items Details:
Item Name: product, Item Price: 56.00, Sales Tax Liability: 11.48, Final Price: 67.48
Item Name: product1, Item Price: 45.00, Sales Tax Liability: 6.98, Final Price: 51.98

C:\Users\anjanisachan\workspace\Bench_Assignment\src>
```

## Question 2.

### WordFrequency.java



```
eclipse-workspace - Bench_AssignmentQ2/src/WordFrequency.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help

Project Explorer
  _pasted_code_
  AmazonDemo
  Assignment
  Bench_Assignment
  Bench_AssignmentQ2
    CucumberAssignment
  Day0
  Even_Odd
  ExitAssignmentTest
  Fileread
  Hi
  httpdogapp
  MoengageAssignment
  RestApiDemo
  SeleniumProject

Main.java WordFrequency.java
4 import java.util.HashMap;
5 import java.util.Map;
6 import java.util.StringTokenizer;
7
8 public class WordFrequency {
9
10     public static void main(String[] args) {
11         String filePath = "Example.txt";
12         Map<String, Integer> wordCountMap = new HashMap<>();
13
14         try {
15             BufferedReader reader = new BufferedReader(new FileReader("C:\\Users\\anjanisachan\\workspace\\Bench_AssignmentQ2\\src\\Example.txt"));
16             String line;
17             while ((line = reader.readLine()) != null) {
18                 StringTokenizer tokenizer = new StringTokenizer(line, ".,:;'\\"");
19                 while (tokenizer.hasMoreTokens()) {
20                     String word = tokenizer.nextToken().toLowerCase();
21                     wordCountMap.put(word, wordCountMap.getOrDefault(word, 0) + 1);
22                 }
23             }
24         } catch (IOException e) {
25             e.printStackTrace();
26         }
27
28         // Print word frequencies
29         for (Map.Entry<String, Integer> entry : wordCountMap.entrySet()) {
30             System.out.println(entry.getKey() + " : " + entry.getValue());
31         }
32     }
33 }
34

Declaration Console Coverage x JUnit
TestNG x
0 items selected
```

## How to run ?

To run the code – Open a terminal and navigate to the directory where **WordFrequency.java** are saved.

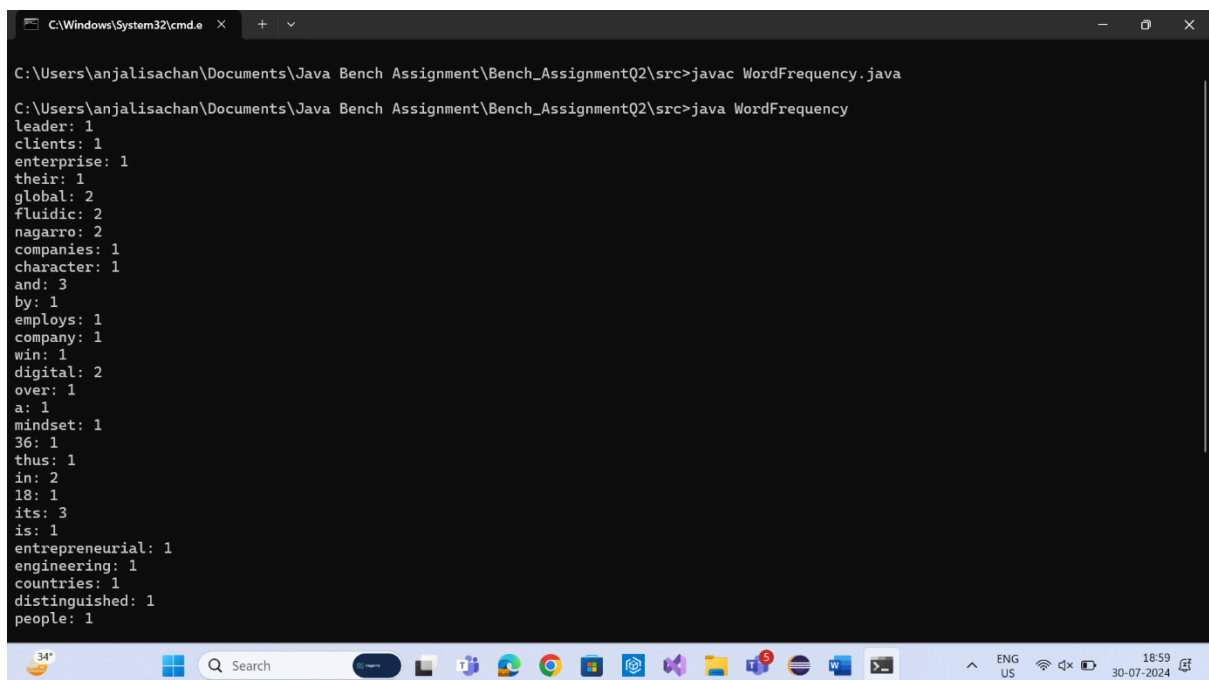
**Compile the program using the command:**

```
javac WordFrequency.java
```

**Run the program using the command:**

```
java WordFrequency.java
```

## Output:



```
C:\Windows\System32\cmd.exe x + v
C:\Users\anjalisachan\Documents\Java Bench Assignment\Bench_AssignmentQ2\src>javac WordFrequency.java
C:\Users\anjalisachan\Documents\Java Bench Assignment\Bench_AssignmentQ2\src>java WordFrequency
leader: 1
clients: 1
enterprise: 1
their: 1
global: 2
fluidic: 2
nagarro: 2
companies: 1
character: 1
and: 3
by: 1
employs: 1
company: 1
win: 1
digital: 2
over: 1
a: 1
mindset: 1
36: 1
thus: 1
in: 2
18: 1
its: 3
is: 1
entrepreneurial: 1
engineering: 1
countries: 1
distinguished: 1
people: 1
```