

ASSIGNMENT-5.4

2303A51595

B-10

TASK-1

Prompt : Write a Java program to take user name, age, and email from input. Add comments on how to protect user data, like hashing email and not storing data openly.

Code :

```
import java.util.Scanner;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class UserInfoCollector {
    private static String hashEmail(String email) {
        try {
            MessageDigest digest = MessageDigest.getInstance("SHA-256");
            byte[] hash = digest.digest(email.getBytes());
            StringBuilder hexString = new StringBuilder();
            for (byte b : hash) {
                String hex = Integer.toHexString(0xff & b);
                if (hex.length() == 1) hexString.append('0');
                hexString.append(hex);
            }
            return hexString.toString();
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException("Error hashing email", e);
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Welcome! We will collect some basic information from you.");
        System.out.print("Do you consent to provide your information? (yes/no): ");
        String consent = sc.nextLine().trim().toLowerCase();

        if (!consent.equals("yes")) {
```

```

        System.out.println("Thank you! No data will be collected.");
        sc.close();
        return;
    }
    System.out.print("Enter your name: ");
    String name = sc.nextLine().trim();

    System.out.print("Enter your age: ");
    int age = Integer.parseInt(sc.nextLine().trim());

    System.out.print("Enter your email address: ");
    String email = sc.nextLine().trim();
    String hashedEmail = hashEmail(email);
    System.out.println("\nThank you for providing your information!");
    System.out.println("Name: " + name);
    System.out.println("Age: " + age);
    System.out.println("Hashed Email: " + hashedEmail);
    sc.close();
}
}

```

Output :

The screenshot shows an IDE with the following components:

- EXPLORER:** A list of files including .vscode, java, Bank.class, Bill.class, CentimeterConverter.class, duplicate.java, EnergyBillCalculator.class, fibo.java, FileLineCount.class, javac, NameFormatter.class, palindrome.java, PrimeNum.java, recursion.java, reversearray.java, sum.java, UserInfoCollector.class, and UserInfoCollector.java (selected).
- EDITOR:** The code for `UserInfoCollector.java` is displayed. It includes imports for `Scanner`, `MessageDigest`, and `NoSuchAlgorithmException`. The `hashEmail` method uses `MessageDigest` to hash the email address.
- TERMINAL:** The output of the program is shown, including the command `javac UserInfoCollector.java` and the runtime output:


```

C:\Users\deep\Downloads\OneDrive\Desktop\AI ASSISTED CODE>java UserInfoCollector
Welcome! We will collect some basic information from you.
Do you consent to provide your information? (yes/no): yes
Enter your name: arthi reddy
Enter your age: 19
Enter your email address: arthireddy@gmail.com

Thank you for providing your information!
Name: arthi reddy
Age: 19
      
```

Analysis :

In this task, we collect user details like name, age, and email. User data is private, so it should not be stored openly. Email can be protected by hashing. Collecting

only required data helps in keeping user information safe. Developers are responsible for protecting user privacy.

TASK-2

Prompt : Generate a Java program for sentiment analysis (positive, negative, neutral).

Code :

```
import java.util.*;
public class SentimentAnalysis {

    static List<String> positiveWords = Arrays.asList("good", "happy", "excellent",
"nice");
    static List<String> negativeWords = Arrays.asList("bad", "sad", "poor", "worst");

    static List<String> offensiveWords = Arrays.asList("hate", "stupid");

    public static String analyzeSentiment(String text) {
        text = text.toLowerCase();

        int score = 0;

        for (String word : offensiveWords) {
            text = text.replace(word, "");
        }

        for (String word : positiveWords) {
            if (text.contains(word)) {
                score++;
            }
        }

        for (String word : negativeWords) {
            if (text.contains(word)) {
                score--;
            }
        }
    }
}
```

```

        if (score > 0) {
            return "Positive";
        } else if (score < 0) {
            return "Negative";
        } else {
            return "Neutral";
        }
    }
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    System.out.println("Enter a sentence:");
    String input = sc.nextLine();

    String result = analyzeSentiment(input);
    System.out.println("Sentiment: " + result);

    sc.close();
}
}

```

Output :

The screenshot shows a VS Code editor with a Java file named `SentimentAnalysis.java`. The code defines a `SentimentAnalysis` class with three static lists of words: `positiveWords`, `negativeWords`, and `offensiveWords`. The `analyzeSentiment` method takes a string `text` and returns a sentiment based on the word lists. The `main` method prompts the user to enter a sentence and prints the sentiment.

```

1  import java.util.*;
2
3  public class SentimentAnalysis {
4
5      static List<String> positiveWords = Arrays.asList("good", "happy", "excellent", "nice");
6      static List<String> negativeWords = Arrays.asList("bad", "sad", "poor", "worst");
7
8      static List<String> offensiveWords = Arrays.asList("hate", "stupid");
9
10
11
12
13      public static String analyzeSentiment(String text) {
14          text = text.toLowerCase();
15
16          int score = 0;
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

The terminal output shows the program being run, with user input 'Arthi reddy' and 'Arthi reddy', and the output 'Sentiment: Neutral'.

```

Enter your name: Arthi reddy
Enter your age: 20
Enter your email address: arthireddy@gmail.com

Thank you for providing your information!
Name: Arthi reddy
Age: 20
Hashed Email: 13ce34bd5863a0b6d9ee91a5078920f9c36f3c8a58d207c359312584f275b5ad

C:\Users\deept\Downloads\OneDrive\Desktop\AI ASSISTED CODE>javac SentimentAnalysis.java
C:\Users\deept\Downloads\OneDrive\Desktop\AI ASSISTED CODE>java SentimentAnalysis
Enter a sentence:
Hello arthi
Sentiment: Neutral

```

Analysis :

This program finds whether the given text is positive, negative, or neutral. Bias can occur because of offensive words or unbalanced data. To reduce this, offensive words are removed and both positive and negative words are treated equally.

TASK-3

Prompt : Write a Java program that recommends products based on user purchase history.

Code :

```
import java.util.*;

public class ProductRecommendation {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        List<String> products = Arrays.asList(
            "Mobile", "Laptop", "Headphones", "Smart Watch", "Tablet"
        );
        List<String> userHistory = new ArrayList<>();

        System.out.print("Enter number of products you purchased: ");

        if (!sc.hasNextInt()) {
            System.out.println("Invalid input! Please enter a number.");
            sc.close();
            return;
        }

        int n = sc.nextInt();
        sc.nextLine();

        System.out.println("Enter purchased product names:");
        for (int i = 0; i < n; i++) {
            userHistory.add(sc.nextLine());
        }
    }
}
```

```
}
```

```
System.out.println("\nRecommended Products:");
```

```
boolean found = false;
```

```
for (String product : products) {
```

```
    if (!found) {
```

```
        System.out.println("No new recommendations available.");
```

```
    }
```

```
System.out.println("\nNote: Recommendations are based on your past purchases.");
```

```
System.out.print("Are these recommendations helpful? (yes/no): ");
```

```
String feedback = sc.nextLine();
```

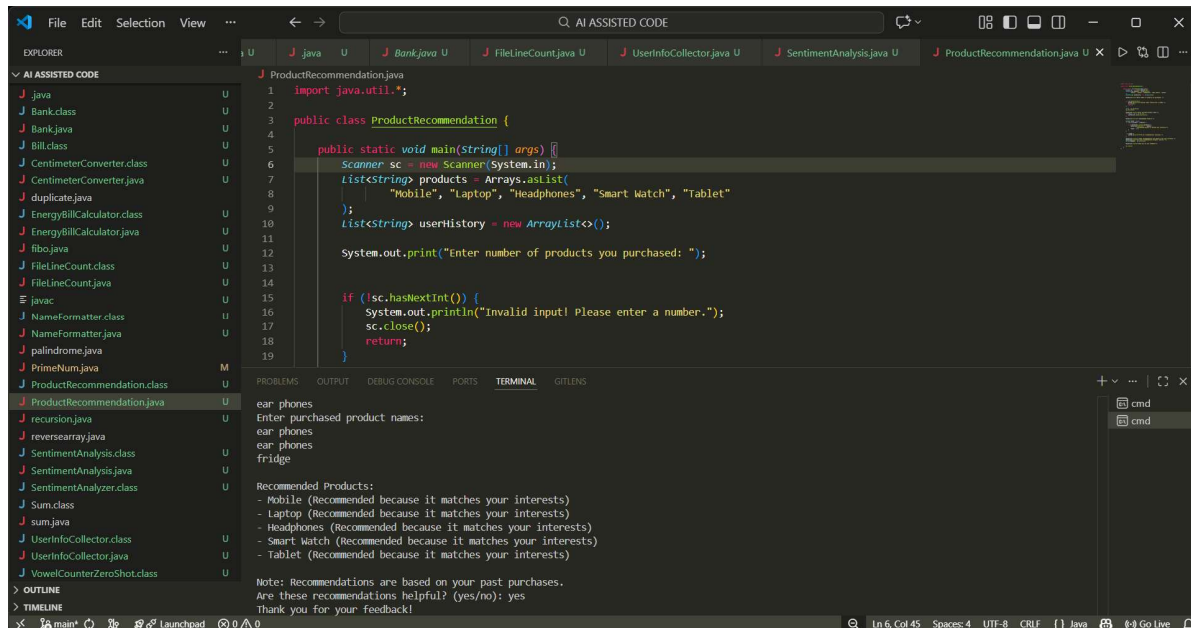
```
System.out.println("Thank you for your feedback!");
```

```
sc.close();
```

```
}
```

```
}
```

Output :



The screenshot shows an IDE with a project named "AI ASSISTED CODE". The Explorer panel on the left lists various Java files, including "ProductRecommendation.java". The main editor displays the code for "ProductRecommendation.java", which includes imports, a class definition, and a main method. The main method uses a Scanner to read input, checks for product names in a list, and prints recommendations based on user history. The Output panel at the bottom shows the execution results, including the input "ear phones", the list of recommended products (Mobile, Laptop, Headphones, Smart Watch, Tablet), and the feedback "yes".

```
1 import java.util.*;
2
3 public class ProductRecommendation {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         List<String> products = Arrays.asList(
8             "Mobile", "Laptop", "Headphones", "Smart Watch", "Tablet"
9         );
10        List<String> userHistory = new ArrayList<>();
11
12        System.out.print("Enter number of products you purchased: ");
13
14        if (!sc.hasNextInt()) {
15            System.out.println("Invalid input! Please enter a number.");
16            sc.close();
17            return;
18        }
19    }
20 }
```

Output:

```
ear phones
Enter purchased product names:
ear phones
ear phones
fridge

Recommended Products:
- Mobile (Recommended because it matches your interests)
- Laptop (Recommended because it matches your interests)
- Headphones (Recommended because it matches your interests)
- Smart Watch (Recommended because it matches your interests)
- Tablet (Recommended because it matches your interests)

Note: Recommendations are based on your past purchases.
Are these recommendations helpful? (yes/no): yes
Thank you for your feedback!
```

Analysis :

The program recommends products based on the user's previous purchases. It avoids repeating already bought products and clearly explains why recommendations are shown, making it fair and transparent.

TASK-4

Prompt : Write a Java program with logging functionality for an application. Make sure the logs do not store sensitive data like passwords or email IDs and add comments explaining safe logging.

Code :

```
import java.util.Scanner;
import java.util.logging.Logger;

public class SecureLogging {

    private static final Logger logger =
        Logger.getLogger(SecureLogging.class.getName());

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter username: ");
        String username = sc.nextLine();
        System.out.print("Enter password: ");
        String password = sc.nextLine();
        logger.info("Login attempt for user: " + username);
        if (username.equals("admin") &&
password.equals("admin123")) {
            logger.info("Login successful for user: " + username);
        } else {
```

```

        logger.warning("Login failed for user: " + username);
    }

    sc.close();
}
}

```

Output :

The screenshot shows an IDE with the following components:

- EXPLORER:** A list of files including `SecureLogging.java`, `Bank.java`, `FileLineCount.java`, `UserinfoCollector.java`, `SentimentAnalysis.java`, `ProductRecommendation.java`, `SecureLogging.java`, `Sum.java`, `palindrome.java`, `PrimeNum.java`, `ProductRecommendation.class`, `recursion.java`, `reversearray.java`, `SecureLogging.class`, `SentimentAnalysis.class`, `SentimentAnalyzer.class`, `Sum.class`, and `sum.java`.
- AI ASSISTED CODE:** The code for `SecureLogging.java` is displayed, showing imports for `Scanner` and `Logger`, and the `main` method logic.
- PROBLEMS:** A list of errors and warnings, including `SecureLogging.java` and `SecureLogging.class`.
- OUTPUT:** The output of the program is shown in the terminal, including the prompt `Enter username:` and the response `INFO: Login attempt for user: admin`.
- DEBUG CONSOLE:** The output of the program is shown in the terminal, including the prompt `Enter username:` and the response `INFO: Login attempt for user: admin`.
- TERMINAL:** The output of the program is shown in the terminal, including the prompt `Enter username:` and the response `INFO: Login attempt for user: admin`.

Analysis :

This program demonstrates ethical logging by recording only necessary application events and avoiding sensitive data such as passwords. It helps in debugging while ensuring user privacy and data security

TASK-5

Prompt : Write a Java program that takes user input and predicts sentiment like a simple machine learning model.

Code :

```
import java.util.*;

public class SimpleMLModel {

    static List<String> positiveWords = Arrays.asList("good", "nice", "happy",
"excellent", "love");
    static List<String> negativeWords = Arrays.asList("bad", "sad", "worst", "hate",
"poor");

    public static String predict(String text) {
        text = text.toLowerCase();
        int score = 0;

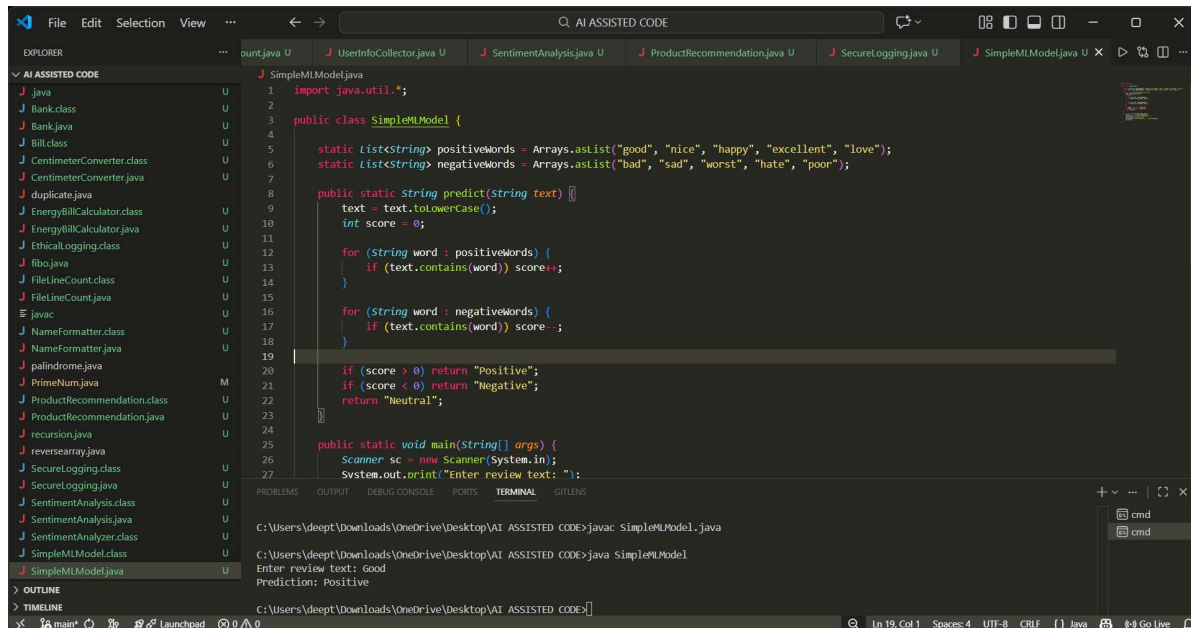
        for (String word : positiveWords) {
            if (text.contains(word)) score++;
        }

        for (String word : negativeWords) {
            if (text.contains(word)) score--;
        }

        if (score > 0) return "Positive";
        if (score < 0) return "Negative";
        return "Neutral";
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter review text: ");
        String input = sc.nextLine();
        System.out.println("Prediction: " + predict(input));
        sc.close();
    }
}
```

Output :



```
File Edit Selection View ... Q AI ASSISTED CODE
EXPLORER
  AI ASSISTED CODE
    java U
    Bank.class U
    Bank.java U
    Bill.class U
    CentimeterConverter.class U
    CentimeterConverter.java U
    duplicate.java U
    EnergyBillCalculator.class U
    EnergyBillCalculator.java U
    EthicalLogging.class U
    fibo.java U
    FileLineCount.class U
    FileLineCount.java U
    javac U
    NameFormatter.class U
    NameFormatter.java U
    palindrome.java U
    PrimeNum.java M
    ProductRecommendation.class U
    ProductRecommendation.java U
    recursion.java U
    reversearray.java U
    SecureLogging.class U
    SecureLogging.java U
    SentimentAnalysis.class U
    SentimentAnalysis.java U
    SentimentAnalyzer.class U
    SimpleMLModel.class U
    SimpleMLModel.java U
  J SimpleMLModel.java
    1 import java.util.*;
    2
    3 public class SimpleMLModel {
    4
    5     static List<String> positiveWords = Arrays.asList("good", "nice", "happy", "excellent", "love");
    6     static List<String> negativeWords = Arrays.asList("bad", "sad", "worst", "hate", "poor");
    7
    8     public static String predict(String text) {
    9         text = text.toLowerCase();
    10         int score = 0;
    11
    12         for (String word : positiveWords) {
    13             if (text.contains(word)) score++;
    14         }
    15
    16         for (String word : negativeWords) {
    17             if (text.contains(word)) score--;
    18         }
    19
    20         if (score > 0) return "Positive";
    21         if (score < 0) return "Negative";
    22         return "Neutral";
    23     }
    24
    25     public static void main(String[] args) {
    26         Scanner sc = new Scanner(System.in);
    27         System.out.println("Enter review text: ");
    28
    29         String text = sc.nextLine();
    30         String prediction = predict(text);
    31         System.out.println("Prediction: " + prediction);
    32     }
    33 }
  PROBLEMS OUTPUT DEBUG CONSOLE PORTS TERMINAL GIT LENS
  C:\Users\deept\Downloads\OneDrive\Desktop\AI ASSISTED CODE>javac SimpleMLModel.java
  C:\Users\deept\Downloads\OneDrive\Desktop\AI ASSISTED CODE>java SimpleMLModel
  Enter review text: Good
  Prediction: Positive
  > OUTLINE
  > TIMELINE
  C:\Users\deept\Downloads\OneDrive\Desktop\AI ASSISTED CODE>
  Ln 19, Col 1 Spaces: 4 UTF-8 CRLF () Java Go Live
```

Analysis :

This program takes user input and predicts sentiment using a simple model. Since the model is basic and may not be fully accurate, the results should be checked by humans and used responsibly.