

ASSIGNMENT

B. LIKHITHA

Assignment 1

Title: Write Data to CSV File

Problem Statement:

Write student details (id, name, marks) into a CSV file named students.csv.

Conditions:

- Use FileWriter
- Data format:
- id,name,marks
- 101,Ravi,78
- 102,Anita,85

Expected Outcome:

CSV file should be created and data should be stored correctly.

The screenshot shows a Java code editor and a terminal window. The code editor displays the following Java code:

```
1 package com;
2
3 import java.io.*;
4
5 class WriteCSV {
6     public static void main(String[] args) {
7         try {
8             FileWriter fw = new FileWriter("students.csv");
9
10            fw.write("id,name,marks\n");
11            fw.write("101,Ravi,78\n");
12            fw.write("102,Anita,85\n");
13
14            fw.close();
15            System.out.println("CSV file created successfully.");
16        } catch (Exception e) {
17            System.out.println(e);
18        }
19    }
20 }
21
22
23
```

The terminal window below shows the output of the program:

```
Problems Javadoc Declaration Console <terminated> WriteCSV [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026)
CSV file created successfully.
```

Assignment 2

Title: Read CSV File and Display Data

Problem Statement:

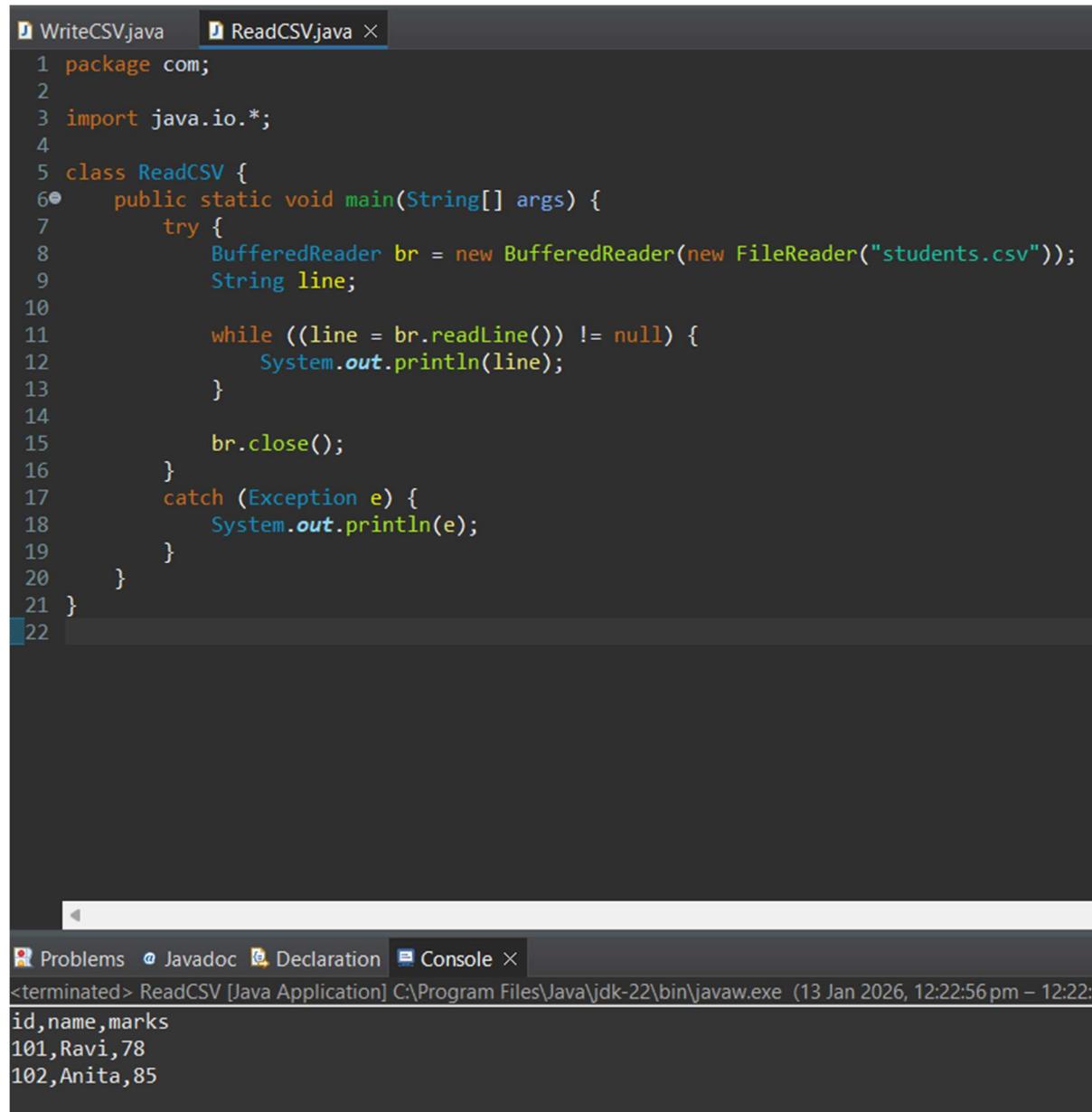
Read all records from students.csv and display them on the console.

Conditions:

- Use FileReader and BufferedReader
- Split each line using comma (,)

Expected Outcome:

All rows should be printed.



```
1 package com;
2
3 import java.io.*;
4
5 class ReadCSV {
6     public static void main(String[] args) {
7         try {
8             BufferedReader br = new BufferedReader(new FileReader("students.csv"));
9             String line;
10
11             while ((line = br.readLine()) != null) {
12                 System.out.println(line);
13             }
14
15             br.close();
16         }
17         catch (Exception e) {
18             System.out.println(e);
19         }
20     }
21 }
22
```

Problems Javadoc Declaration Console <terminated> ReadCSV [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:22:56 pm – 12:22:59 id,name,marks
101,Ravi,78
102,Anita,85

Assignment 3

Title: Display Students with Marks Greater Than 60

Problem Statement:

Read students.csv and display only students whose marks are greater than 60.

Conditions:

- Convert marks to integer
- Use if condition

Expected Outcome:

Only eligible students should be printed.

The screenshot shows an IDE interface with two tabs open: 'WriteCSV.java' and 'MarksAbove60.java'. The 'MarksAbove60.java' tab is active, displaying the following Java code:

```
1 package com;
2 import java.io.*;
3
4 class MarksAbove60 {
5     public static void main(String[] args) {
6         try {
7             BufferedReader br = new BufferedReader(new FileReader("students.csv"));
8             String line;
9
10            br.readLine(); // skip header
11
12            while ((line = br.readLine()) != null) {
13                String[] data = line.split(",");
14                int marks = Integer.parseInt(data[2]);
15
16                if (marks > 60) {
17                    System.out.println(line);
18                }
19            }
20            br.close();
21        } catch (Exception e) {
22            System.out.println(e);
23        }
24    }
25 }
26
27 }
```

Below the code editor, the 'Console' tab is selected, showing the output of the program:

```
<terminated> MarksAbove60 [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:23:50 pm - 12:2
101,Ravi,78
102,Anita,85
```

Assignment 4

Title: Count Number of Records in CSV File

Problem Statement:

Count how many student records are present in students.csv.

Conditions:

- Ignore header line
- Use counter variable

Expected Outcome:

Total number of students displayed.

The screenshot shows an IDE interface with several tabs at the top: WriteCSV.java, ReadCSV.java, MarksAbove60.java, and CountRecords.java (which is the active tab). The code in CountRecords.java is as follows:

```
1 package com;
2 import java.io.*;
3
4 class CountRecords {
5     public static void main(String[] args) {
6         try {
7             BufferedReader br = new BufferedReader(new FileReader("students.csv"));
8             String line;
9             int count = 0;
10
11            br.readLine(); // skip header
12
13            while ((line = br.readLine()) != null) {
14                count++;
15            }
16
17            System.out.println("Total students: " + count);
18            br.close();
19        } catch (Exception e) {
20            System.out.println(e);
21        }
22    }
23 }
24
25 }
```

At the bottom, the Console tab shows the output: <terminated> CountRecords [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:24:54 pm – 12s) Total students: 2

Assignment 5**Title: Search Student by Name****Problem Statement:**

Search a student by name from students.csv.

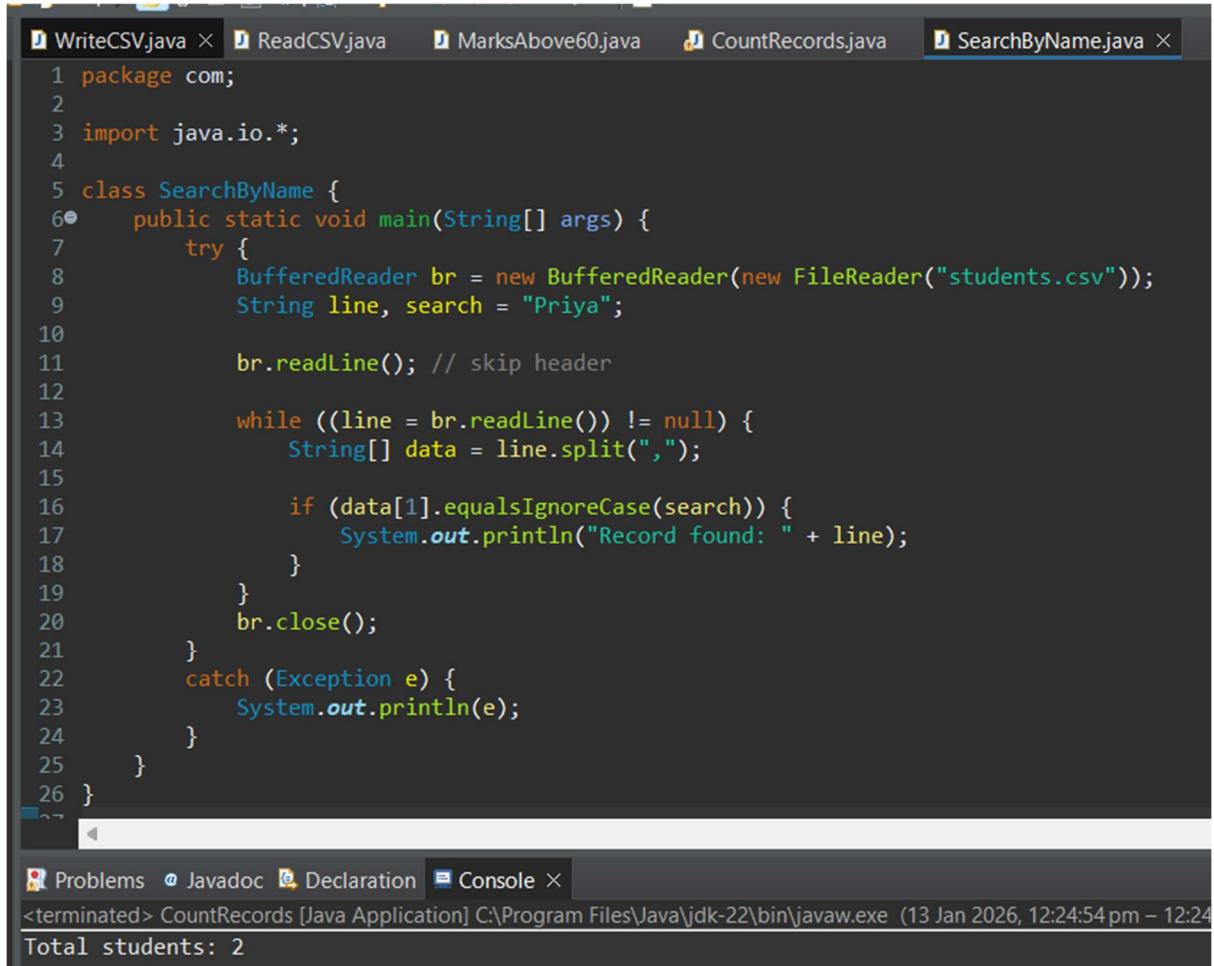
Conditions:

- Use equalsIgnoreCase()

- Display full record if name matches

Expected Outcome:

Matching student details printed.



The screenshot shows an IDE interface with multiple tabs at the top: WriteCSV.java, ReadCSV.java, MarksAbove60.java, CountRecords.java, and SearchByName.java. The SearchByName.java tab is active. The code implements a search for a student named "Priya" in a CSV file named "students.csv". It reads the file, skips the header, and then iterates through each line. If the second field (name) equals "Priya", it prints the entire line. The code uses BufferedReader and String.split() methods. The console output at the bottom shows the program has terminated and printed "Total students: 2".

```

1 package com;
2
3 import java.io.*;
4
5 class SearchByName {
6     public static void main(String[] args) {
7         try {
8             BufferedReader br = new BufferedReader(new FileReader("students.csv"));
9             String line, search = "Priya";
10            br.readLine(); // skip header
11            while ((line = br.readLine()) != null) {
12                String[] data = line.split(",");
13                if (data[1].equalsIgnoreCase(search)) {
14                    System.out.println("Record found: " + line);
15                }
16            }
17            br.close();
18        } catch (Exception e) {
19            System.out.println(e);
20        }
21    }
22 }
23
24
25
26 }
```

Problems Javadoc Declaration Console

<terminated> CountRecords [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:24:54 pm – 12:24 Total students: 2

Assignment 6

Title: Display Students Who Failed

Problem Statement:

Display students whose marks are less than 40.

Conditions:

- Use if (marks < 40)
- Read from CSV file

Expected Outcome:

Only failed students printed.

The screenshot shows a Java IDE interface with the following details:

- Top Bar:** Shows tabs for WriteCSV.java, ReadCSV.java, MarksAbove60.java, CountRecords.java, and SearchByName.java.
- Code Editor:** Displays the code for FailedStudents.java. The code reads a CSV file named "students.csv" and prints out lines where the third column (marks) is greater than 40. The code uses BufferedReader and System.out.println.
- Console Tab:** Shows the output of the program. The output window title is "Console". The output text is:
<terminated> FailedStudents [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:29:06 pm –)
101,Ravi,78
102,Anita,85

Assignment 7

Title: Calculate Average Marks

Problem Statement:

Read marks from students.csv and calculate average marks.

Conditions:

- Sum all marks
- Divide by number of students

Expected Outcome:

Average marks displayed.

The screenshot shows an IDE interface with a code editor and a terminal window.

Code Editor:

```
1 package com;
2
3 import java.io.*;
4
5 class AverageMarks {
6     public static void main(String[] args) {
7         try {
8             BufferedReader br = new BufferedReader(new FileReader("students.csv"));
9             String line;
10            int sum = 0, count = 0;
11
12            br.readLine(); // header
13
14            while ((line = br.readLine()) != null) {
15                String[] data = line.split(",");
16                int marks = Integer.parseInt(data[2]);
17
18                sum += marks;
19                count++;
20            }
21
22            double avg = sum / (double) count;
23            System.out.println("Average Marks: " + avg);
24
25            br.close();
26        }
27        catch (Exception e) {
28            System.out.println(e);
29        }
30    }
31 }
32 }
```

Terminal Window:

```
<terminated> AverageMarks [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:30:28 pm – 12:30:28)
Average Marks: 81.5
```

Assignment 8

Title: Copy Passed Students to New CSV File

Problem Statement:

Read students.csv and write students with marks ≥ 50 into passed_students.csv.

Conditions:

- Use FileWriter
- Maintain CSV format

Expected Outcome:

New CSV file created with passed students only.

The screenshot shows an IDE interface with a code editor and a terminal window. The code editor displays a Java program named `CopyPassedStudents`. The program reads a CSV file named `students.csv` and writes a new CSV file named `passed_students.csv`, containing only records where the marks field is 50 or greater. The code uses `BufferedReader` and `FileWriter` to handle the files, and `Integer.parseInt` to parse the marks field. The terminal window below shows the output of running the program, which prints "passed_students.csv created.".

```
1 package com;
2 import java.io.*;
3
4 class CopyPassedStudents {
5     public static void main(String[] args) {
6         try {
7             BufferedReader br = new BufferedReader(new FileReader("students.csv"));
8             FileWriter fw = new FileWriter("passed_students.csv");
9
10            fw.write("id,name,marks\n");
11            String line;
12
13            br.readLine(); // header
14
15            while ((line = br.readLine()) != null) {
16                String[] data = line.split(",");
17                int marks = Integer.parseInt(data[2]);
18
19                if (marks >= 50) {
20                    fw.write(line + "\n");
21                }
22            }
23
24            br.close();
25            fw.close();
26
27            System.out.println("passed_students.csv created.");
28        } catch (Exception e) {
29            System.out.println(e);
30        }
31    }
32 }
33 }
```

Problems Javadoc Declaration Console <terminated> CopyPassedStudents [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:31:33 pm – 1 passed_students.csv created.

Assignment 9

Title: Validate CSV Data While Reading

Problem Statement:

Skip records where marks field is empty or invalid.

Conditions:

- Use try-catch for NumberFormatException
- Ignore invalid rows

Expected Outcome:

Only valid records processed.

```
1 package com;
2
3 import java.io.*;
4
5 class ValidateCSV {
6     public static void main(String[] args) {
7         try {
8             BufferedReader br = new BufferedReader(new FileReader("students.csv"));
9             String line;
10            br.readLine(); // skip header
11
12            while ((line = br.readLine()) != null) {
13                String[] data = line.split(",");
14
15                try {
16                    int marks = Integer.parseInt(data[2]);
17                    System.out.println("Valid Record: " + line);
18                } catch (NumberFormatException e) {
19                    System.out.println("Invalid Record Skipped: " + line);
20                }
21            }
22            br.close();
23        } catch (Exception e) {
24            System.out.println(e);
25        }
26    }
27
28 }
29
30 }
31
32 }
```

Problems Javadoc Declaration Console X
<terminated> ValidateCSV [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:32:33 pm - 12:32:33
Valid Record: 101,Ravi,78
Valid Record: 102,Anita,85

Assignment 10

Title: Display Topper Details

Problem Statement:

Find and display the student with highest marks from students.csv.

Conditions:

- Compare marks
- Track maximum value

Expected Outcome:

Topper name and marks printed.

Sample CSV File (students.csv)

id,name,marks

101,Ravi,78

102,Anita,85

103,Sunil,35

104,Priya,92

105,Aman,45

The screenshot shows a Java IDE interface with the following details:

- Top Bar:** Shows tabs for WriteCSV.java, FailedStude..., AverageMark..., CopyPassedS..., ValidateCSV..., and TopperStude... (which is currently active).
- Code Editor:** Displays the code for TopperStudent.java. The code reads a CSV file "students.csv", finds the student with the highest marks, and prints their name and marks.
- Console Output:** Shows the terminal window with the command <terminated> TopperStudent [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:33:41 pm – 12:33:45 pm). The output printed is "Topper: Anita - Marks: 85".

```
1 package com;
2
3 import java.io.*;
4 class TopperStudent {
5     public static void main(String[] args) {
6         try {
7             BufferedReader br = new BufferedReader(new FileReader("students.csv"));
8             String line;
9
10            br.readLine(); // skip header
11
12            String topName = "";
13            int topMarks = -1;
14
15            while ((line = br.readLine()) != null) {
16                String[] data = line.split(",");
17                int marks = Integer.parseInt(data[2]);
18
19                if (marks > topMarks) {
20                    topMarks = marks;
21                    topName = data[1];
22                }
23            }
24
25            System.out.println("Topper: " + topName + " - Marks: " + topMarks);
26
27            br.close();
28        } catch (Exception e) {
29            System.out.println(e);
30        }
31    }
32 }
33 }
34 }
```

Problems Javadoc Declaration Console X
<terminated> TopperStudent [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (13 Jan 2026, 12:33:41 pm – 12:33:45 pm)
Topper: Anita - Marks: 85