Python File Handling Assignments

Assignment 1: Log File Analyzer

Objective: Create a program that processes a log file, extracts specific information, and generates a summary report.

Requirements:

1. The program should read a log file containing entries in this format:

   ```
   [TIMESTAMP] [LOG_LEVEL] Message content here
   ```

   (Example: `[2024-05-10 14:32:15] [ERROR] Database connection failed`)

2. The program should:
   - Count occurrences of each log level (INFO, WARNING, ERROR, etc.)
   - Find the time period with the most ERROR messages
   - Extract and list all unique error messages
   - Generate a summary report in a new file
3. Use context managers (`with` statement) for all file operations
4. Implement proper exception handling
5. Process the file line by line rather than loading it all at once

Sample Input File:

```
[2024-05-10 08:15:22] [INFO] Application started
[2024-05-10 08:16:05] [INFO] User login: alice@example.com
[2024-05-10 08:20:14] [WARNING] High memory usage detected
[2024-05-10 08:22:30] [ERROR] Database connection failed
[2024-05-10 08:22:45] [INFO] Retry connection attempt
[2024-05-10 08:22:50] [INFO] Database connected successfully
[2024-05-10 09:30:12] [ERROR] Database connection failed
[2024-05-10 09:31:23] [ERROR] Database connection failed
```

Expected Output File:

```
LOG ANALYSIS SUMMARY
--------------------
Total log entries: 8
By level:
  INFO: 4
  WARNING: 1
  ERROR: 3

Most errors occurred between 09:30:12 and 09:31:23 (2 occurrences)

Unique error messages:
- Database connection failed
```

Assignment 2: CSV Data Processor

Objective: Create a program that reads data from a CSV file, performs data filtering and transformation, and writes results to new files.

Requirements:

1. The program should read a CSV file containing student information (name, ID, grades for multiple subjects)
2. It should:
   - Calculate each student's average grade
   - Filter students based on pass/fail criteria (average ≥ 60 is passing)
   - Write passing students to a "passed.csv" file
   - Write failing students to a "failed.csv" file
   - Create a summary report with statistics in a text file
3. Use the `csv` module for proper CSV handling
4. Properly handle missing or invalid data (e.g., non-numeric grades)
5. Log any data issues to an error file

Sample Input File (students.csv):

```
Name,ID,Math,Science,English,History
Alice Smith,1001,92,88,95,91
Bob Johnson,1002,45,52,57,61
Charlie Brown,1003,82,79,85,80
Diana Miller,1004,71,65,N/A,68
Evan Davis,1005,59,62,71,58
```

Expected Output Files:

passed.csv:

```
Name,ID,Average
Alice Smith,1001,91.5
Charlie Brown,1003,81.5
Diana Miller,1004,68.0
```

failed.csv:

```
Name,ID,Average
Bob Johnson,1002,53.75
Evan Davis,1005,62.5
```

summary.txt:

```
STUDENT GRADE SUMMARY
---------------------
Total students: 5
Passing: 3 (60%)
Failing: 2 (40%)
Highest average: Alice Smith (91.5)
Lowest average: Bob Johnson (53.75)
Class average: 71.45

Data issues:
- Missing grade: Diana Miller (ID: 1004), English
```