IS 430 – Foundations of Information Processing
Instructor: Kevin Trainor
Assignment: Severance - Chapter 14 Coding Assignment
Course Component: Coding Assignments
Grading Rubric

Base Point Allocation

Base Points (23 available points)

Requirements
Assignment submitted on-time or within the allowable late period.

Percent Credit	Description
100	Meets all expectations.
0	Not submitted or submitted too late.

Submission

Timeliness (16 available points)

Requirements
Must be submitted by date and time indicated in the weekly schedule.

Percent Credit	Description
100	On Time
0	Late
0	Not submitted or submitted too late

File Submitted (10 available points)

Requirements
Submit only 1 file.
File type must be .ZIP.
File name must conform to all requirements stated in assignment instructions.
Contents of .ZIP file must be a properly named directory that represents a PyCharm project.
Directory contents must be properly named PyCharm project files.
The PyCharm project directory must include a data sub-directory to hold data files to be read or written by the exercise programs.
The data sub-directory must include any data starter files provided for the assignment. The grader cannot be expected to load starter files in order to test your code.

Percent Credit	Description
100	Meets all expectations.
50	Meets nearly all expectations.
0	Does not meet expectations.
0	Not submitted or submitted too late.

Exercise 1 (Required)

Completeness (5 available content points)

Requirements
Must produce the expected quantity of results.
Must produce the exact values expected.
Results must be formatted as expected.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Technique (6 available content points)

Requirements
Production code and test code must be included.
Test code must include all appropriate test cases.
Test cases must pass when run.
The Python module must be properly named.
The Python module must begin with a proper Docstring.
The top-level code in the Python program must be located in a procedure named "main".
If present, lower-level code in the Python program must be located in procedures that have properly formed descriptive names.
If present, any variables used in the Python program must have properly formed descriptive names.
The only code in the Python module that is not contained inside of a procedure must be the code that calls the procedure "main".
The code that calls that procedure "main" should appear last in the Python module.
The Python module must not contain syntax errors.
The Python module must be run-able.
The Python module must make use of all techniques that have been demonstrated in the video tutorial for this assignment.
The Python module must conform to the PEP 8 Style Guide for Python Code.
The code should not contain any "magic numbers". Instead, it should define descriptive CONSTANTS to hold standard values.
The code should use f-strings for string interpolation and number formatting.
When processing items from Python lists and tuples, the code should unpack the values into variables with meaningful variable names to avoid using less meaningful indexed expressions.
When tested, the program should be able to handle missing input. This includes empty files, blank lines in files, or the user not entering any characters at the console. The program should behave reasonably and end gracefully.
All functions that are not main() should have descriptive, action-oriented names.
All functions should be of reasonable size.
All functions should have high cohesion, and low coupling.
Use of the break statement is allowed but not encouraged.
Use of the continue statement is forbidden.
Regular expression patterns should be expressed as Python raw strings.
Your finished code must be refactored to meet all good program design practices covered in this course.
Your refactored code must be retested to demonstrate that refactoring has not altered program functionality.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Exercise 2 (Required)

Completeness (5 available content points)

Requirements
Must produce the expected quantity of results.
Must produce the exact values expected.
Results must be formatted as expected.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Technique (6 available content points)

Requirements
The Python module must be properly named.
The Python module must begin with a proper Docstring.
The top-level code in the Python program must be located in a procedure named "main".
If present, lower-level code in the Python program must be located in procedures that have properly formed descriptive names.
If present, any variables used in the Python program must have properly formed descriptive names.
The only code in the Python module that is not contained inside of a procedure must be the code that calls the procedure "main".
The code that calls that procedure "main" should appear last in the Python module.
The Python module must not contain syntax errors.
The Python module must be run-able.
The Python module must make use of all techniques that have been demonstrated in the video tutorial for this assignment.
The Python module must conform to the PEP 8 Style Guide for Python Code.
The code should not contain any "magic numbers". Instead, it should define descriptive CONSTANTS to hold standard values.
The code should use f-strings for string interpolation and number formatting.
When processing items from Python lists and tuples, the code should unpack the values into variables with meaningful variable names to avoid using less meaningful indexed expressions.
When tested, the program should be able to handle missing input. This includes empty files, blank lines in files, or the user not entering any characters at the console. The program should behave reasonably and end gracefully.
All functions that are not main() should have descriptive, action-oriented names.
All functions should be of reasonable size.
All functions should have high cohesion, and low coupling.
Use of the break statement is allowed but not encouraged.
Use of the continue statement is forbidden.
Regular expression patterns should be expressed as Python raw strings.
Your finished code must be refactored to meet all good program design practices covered in this course.
Your refactored code must be retested to demonstrate that refactoring has not altered program functionality.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Exercise 3 (Required)

Completeness (6 available content points)

Requirements
Must produce the expected quantity of results.
Must produce the exact values expected.
Results must be formatted as expected.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Technique (6 available content points)

Requirements
Production code and test code must be included.
Test code must include all appropriate test cases.
Test cases must pass when run.
The Python module must be properly named.
The Python module must begin with a proper Docstring.
The top-level code in the Python program must be located in a procedure named "main".
If present, lower-level code in the Python program must be located in procedures that have properly formed descriptive names.
If present, any variables used in the Python program must have properly formed descriptive names.
The only code in the Python module that is not contained inside of a procedure must be the code that calls the procedure "main".
The code that calls that procedure "main" should appear last in the Python module.
The Python module must not contain syntax errors.
The Python module must be run-able.
The Python module must make use of all techniques that have been demonstrated in the video tutorial for this assignment.
The Python module must conform to the PEP 8 Style Guide for Python Code.
The code should not contain any "magic numbers". Instead, it should define descriptive CONSTANTS to hold standard values.
The code should use f-strings for string interpolation and number formatting.
When processing items from Python lists and tuples, the code should unpack the values into variables with meaningful variable names to avoid using less meaningful indexed expressions.
When tested, the program should be able to handle missing input. This includes empty files, blank lines in files, or the user not entering any characters at the console. The program should behave reasonably and end gracefully.
All functions that are not main() should have descriptive, action-oriented names.
All functions should be of reasonable size.
All functions should have high cohesion, and low coupling.
Use of the break statement is allowed but not encouraged.
Use of the continue statement is forbidden.
Regular expression patterns should be expressed as Python raw strings.
Your finished code must be refactored to meet all good program design practices covered in this course.
Your refactored code must be retested to demonstrate that refactoring has not altered program functionality.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Exercise 4 (Required)

Completeness (6 available content points)

Requirements
Must produce the expected quantity of results.
Must produce the exact values expected.
Results must be formatted as expected.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Technique (6 available content points)

Requirements
The Python module must be properly named.
The Python module must begin with a proper Docstring.
The top-level code in the Python program must be located in a procedure named "main".
If present, lower-level code in the Python program must be located in procedures that have properly formed descriptive names.
If present, any variables used in the Python program must have properly formed descriptive names.
The only code in the Python module that is not contained inside of a procedure must be the code that calls the procedure "main".
The code that calls that procedure "main" should appear last in the Python module.
The Python module must not contain syntax errors.
The Python module must be run-able.
The Python module must make use of all techniques that have been demonstrated in the video tutorial for this assignment.
The Python module must conform to the PEP 8 Style Guide for Python Code.
The code should not contain any "magic numbers". Instead, it should define descriptive CONSTANTS to hold standard values.
The code should use f-strings for string interpolation and number formatting.
When processing items from Python lists and tuples, the code should unpack the values into variables with meaningful variable names to avoid using less meaningful indexed expressions.
When tested, the program should be able to handle missing input. This includes empty files, blank lines in files, or the user not entering any characters at the console. The program should behave reasonably and end gracefully.
All functions that are not main() should have descriptive, action-oriented names.
All functions should be of reasonable size.
All functions should have high cohesion, and low coupling.
Use of the break statement is allowed but not encouraged.
Use of the continue statement is forbidden.
Regular expression patterns should be expressed as Python raw strings.
Your finished code must be refactored to meet all good program design practices covered in this course.
Your refactored code must be retested to demonstrate that refactoring has not altered program functionality.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Exercise 5 (Optional)

Completeness (2 available content points)

Requirements
Must produce the expected quantity of results.
Must produce the exact values expected.
Results must be formatted as expected.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Technique (3 available content points)

Requirements
Production code and test code must be included.
Test code must include all appropriate test cases.
Test cases must pass when run.
The Python module must be properly named.
The Python module must begin with a proper Docstring.
The top-level code in the Python program must be located in a procedure named "main".
If present, lower-level code in the Python program must be located in procedures that have properly formed descriptive names.
If present, any variables used in the Python program must have properly formed descriptive names.
The only code in the Python module that is not contained inside of a procedure must be the code that calls the procedure "main".
The code that calls that procedure "main" should appear last in the Python module.
The Python module must not contain syntax errors.
The Python module must be run-able.
The Python module must make use of all techniques that have been demonstrated in the video tutorial for this assignment.
The Python module must conform to the PEP 8 Style Guide for Python Code.
The code should not contain any "magic numbers". Instead, it should define descriptive CONSTANTS to hold standard values.
The code should use f-strings for string interpolation and number formatting.
When processing items from Python lists and tuples, the code should unpack the values into variables with meaningful variable names to avoid using less meaningful indexed expressions.
When tested, the program should be able to handle missing input. This includes empty files, blank lines in files, or the user not entering any characters at the console. The program should behave reasonably and end gracefully.
All functions that are not main() should have descriptive, action-oriented names.
All functions should be of reasonable size.
All functions should have high cohesion, and low coupling.
Use of the break statement is allowed but not encouraged.
Use of the continue statement is forbidden.
Regular expression patterns should be expressed as Python raw strings.
Your finished code must be refactored to meet all good program design practices covered in this course.
Your refactored code must be retested to demonstrate that refactoring has not altered program functionality.

Percent Credit	Description
100	Meets all expectations.
90	Meets nearly all expectations.
75	Meets most expectations.
50	Meets some expectations.
25	Meets few expectations.
10	Meets nearly no expectations.
0	Meets no expectations.
0	Not submitted or submitted too late.

Total Available Points = 100

Please Note: This grading rubric allows for adjustments to be made to your content point score. The total number of content points available to be awarded on this assignment is 51. An adjustment of up to 36 content points may be made for submissions that have a low content point score and yet meet the following criteria: Assignment must be submitted on time. Work submitted must show good faith effort on all REQUIRED EXERCISES. It is possible to qualify for the points adjustment without having submitted work on the CHALLENGE EXERCISE.