

CSE 106 – Lab 1

- 1) Write an **adding** program that does the following (10 points):
 - a. Ask the user to enter two or more numbers separated by spaces
 - b. Print the sum of all the numbers to the console
 - c. Throw an error if they do not enter at least two numbers or contain a string
 - d. Note: the numbers can be integers or decimals
 - e. Example:
 - i. The user enters: 1 2 3 4
 - ii. The program prints: 10
- 2) Write a **punishment automation** program in Python that does the following (15 points):
 - a. Ask the user to enter a sentence and the times a sentence should be repeated
 - b. The program should write the sentence (with a line break) the number of times specified by the user to a file called "CompletedPunishment.txt"
 - c. Example:
 - i. The user enters this for the sentence: I will not sleep in class
 - ii. The user enters this for the number of repeats: 100
 - iii. The program should write "I will not sleep in class" 100 times to "CompletedPunishment.txt".
- 3) Write a **word count** program in Python that does the following (25 points):
 - a. Prompt the user to enter a word
 - b. Parse PythonSummary.txt and count the number of times the word occurs in the file
 - c. Tell the user how many times the word occurs
 - d. Note: It should find the word regardless of case (upper or lower) or punctuation
 - e. Example:
 - i. The user enters: python
 - ii. The program should print: The word python occurs 13 times
- 4) Write a **class schedule formatting** program that does the following (25 points):
 - a. Parses "classesInput.txt" for the following info (on the corresponding line):
 - Line 0: Number of courses (the following data should exist for each course)
 - Line 1: Course department
 - Line 2: Course number
 - Line 3: Course name
 - Line 4: Credits
 - Line 5: Lecture days
 - Line 6: Start time of the lecture
 - Line 7: End time of the lecture
 - Line 8: Average grade (percentage) for the course
 - b. Outputs a file with the data formatted as follows:

```
COURSE 1: <Course department><Course number>:<Course name>
Number of Credits: <Credits>
Days of Lectures: <Lecture days>
```

Lecture Time: <Start time> - <End time>

Stat: on average, students get <average grade> in this course

REPEAT for each additional class, up to <Number of courses>

c. **Example:**

Input:

2

CSE

030

Data Structures

4

Monday, Wednesday

4:30pm

5:45pm

85

CSE

165

Introduction to Object Oriented Programming

4

Tuesday, Thursday

9:00am

10:15am

87

Output:

COURSE 1: CSE030: Data Structures

Number of Credits: 4

Days of Lectures: Monday, Wednesday

Lecture Time: 4:30pm - 5:45pm

Stat: on average, students get 85% in this course

COURSE 2: CSE165: Introduction to Object Oriented Programming

Number of Credits: 4

Days of Lectures: Tuesday, Thursday

Lecture Time: 9:00am - 10:15am

Stat: on average, students get 87% in this course

- d. Note: to get full points, you must create a Python class that holds the above data and has a format function that returns, or outputs the formatted text

5) Create a **grades** program that does the following (25 points):

- Allows a user to create a student name and grade
- Allows a user to ask for a grade, given the full name of the student
- Allows a user to edit a grade
- Allows a user to delete a grade
- Reads/writes to grades.txt to store grade data persistently in JSON format
- Stores grades in memory data as a dictionary and updates grades.txt with any changes
- Loads grade data from grades.txt into dictionary on program start-up