Assignment 3

Files, functions, and lists

Submit a single zip file called **assignment3.zip**.
This assignment has 50 marks.
See the marking scheme that is posted on the course webpage.

Problem 1 (Student Grades)

For this problem, you will create a grade analysis program in a file called **a3p1.py**. When this program runs, it should prompt the user for a specific filename (you can assume the user input is valid). The program must then read the specified file and produce a summary of the grades, including: the number of people who passed the class, the number of people who did not pass the class, the average of all the final grades within the class and the name/grade of the students with the highest/lowest grades in the class. The final grade can be calculated with the following weights: assignment=25%, midterm=25%, exam=50%. To pass the course, a student must have received a final grade of 50% or greater and a final exam grade of 50% or higher. Several example input files have been included on the assignment page, along with files summarizing the expected output.

All files used for this problem will have the following line-by-line structure:

```
Student1_first_name
Student1_last_name
Student1_student_number
Student1_assignment_grade
Student1_midterm_grade
Student1_exam_grade
Student2_first_name
Student2_last_name
Student2_student_number
Student2_assignment_grade
Student2_midterm_grade
Student2_exam_grade
…etc…
```

Problem 2 (Common Multiples)

Write a function called **ismultiple** which takes 2 integer arguments (*a* and *b*). This function must return True if *b* is a multiple of *a* (i.e., *a* divides into *b* evenly) and False otherwise. Write a second function called **commonmultiple** which takes 3 integer arguments (*a*, *b*, and *c*). This function must return True if *c* is a multiple of both *a* and *b*, and False otherwise. **Additionally, the commonmultiple function must use two calls to the ismultiple function to compute its return value.** Once you have implemented both functions, include testing code that asks the user to enter two numbers (*a* and *b*) and prints out all numbers between 1 and 100 (inclusive) that are common multiples of *a* and *b*. Save your functions and testing code in a file called **a3p2.py** and add it to your submission zip file.

Problem 3 (List Slicing Function)

Python provides slicing functionality for lists, but for this question, you will implement your own function capable of producing list slices (**note: you cannot use the slicing operator in your solution**). The function should be called **slice** and take the following three arguments in this specific order:

1. A list, **source**, which the slice will be created from. **This list cannot be modified by your function.**
2. A positive integer, **start**, representing the starting index of the slice you will create. If this value is not in the range [0, len(list)-1], your function should return an empty list.
3. A positive integer, **end**, representing the ending index of the slice you will create. If this value is not in the range [start, len(list)-1], your function should return an empty list.

If the parameter values are acceptable, your function will return a list that contains the items from **source** beginning at the index **start** and ending at the index **end** (inclusive). This is different from the Python slice operator, as the item at the index **end** is also included in the new list. Examples:

mylist = [“A”, “B”, “C”, “D”, “E”, “F”, “G”, “H”, “I”, “J”]
slice(mylist, 0, 9) should be [“A”, “B”, “C”, “D”, “E”, “F”, “G”, “H”, “I”, “J”]
slice(mylist, 3, 4) should be [“D”, “E”]
slice(mylist, 4, 3) should be [ ]
slice(mylist, 3, 8) should be [“D”, “E”, “F”, “G”, “H”, “I”]
slice(mylist, 4, 4) should be [“E”]

Save your code is a file called **a3p3.py** and add it to your submission zip.

Recap

Your zip file should contain your **a3p1.py**, **a3p2.py**, and **a3p3.py** files.
Submit your **assignment3.zip** file to cuLearn.
Make sure you download the zip after submitting and verify the file contents.