CSC 401 ASSIGNMENT ONE

Due Date: Tuesday, Jul. 28th by 11:58 PM (Chicago time)

The purpose of this assignment is to assess your understanding of

- The basic concepts of arithmetical evaluation, variables, string and list data types, built-in functions.
- The process of creating a Python program.
- How variables are stored in memory.
- How to change the flow of execution using IF control structure
- Testing Python code.
- Use computational thinking to determine the best way to solve each problem

SUBMISSION

- **Include your full name as a comment in the first line of your Python program**
- **Include the problem number as a comment in the second line of your Python program**
- Save each program to a separate file labeled as YourName_hw1_1.py, YourName_hw1_2.py and YourName_hw1_3.py, accordingly
- Upload each file to Submissions folder in D2L.

PROBLEMS

Note: you may not use Python statements, functions, data types, etc. that where not discussed in the reading assignment or the lecture notes/videos for this week. This is a class for students who have not programmed before and I expect everyone to code on the same level. If you have a better way of writing the code, then upload two versions: one that codes according to the specifications and the other that demonstrates advanced programming techniques.

I encourage you to use computational thinking to solve the problems. These are straight-forward solutions, but developing a good habit of analyzing the problem and describing the steps will serve you well as the problems get more complex.

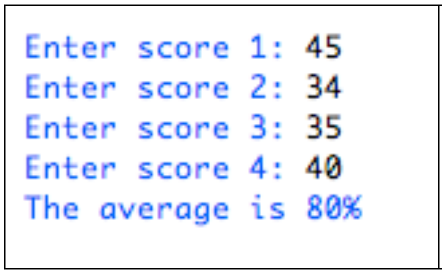
PROBLEM ONE (10 PTS.)
BUILT-IN FUNCTIONS, CALCULATIONS

Write a short Python program that allows the user to enter four homework scores. Each homework is graded on a 50-point basis. The professor will drop the lowest homework score to determine your average homework score for the class. Calculate the average (percentage) of the scores (less the lowest score) and display the results as shown below.
Assign each score and the result to a variable.

Hint: since you do not know what scores the user will enter, you need to use one of the built-in functions to determine the minimum score. Also, you may need to convert the result of the calculation to an int( ) data type or use the round() function on the result in the print statement.

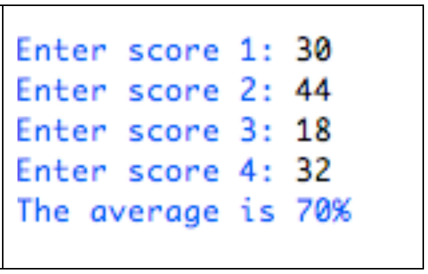
A list of built-in functions can be found at https://docs.python.org/3/library/functions.html

Sample 1:



```
Enter score 1: 45
Enter score 2: 34
Enter score 3: 35
Enter score 4: 40
The average is 80%
```

Sample2:



```
Enter score 1: 30
Enter score 2: 44
Enter score 3: 18
Enter score 4: 32
The average is 70%
```

At the end of the program code, include, as comments, two test cases that you used to test your code and the predicted results. It must be different from the sample data.
For example:
# scores are 40, 35, 50, 30
# predicted results is 83%

PROBLEM TWO (15 PTS.)
STRING, LIST, DECISION

Write a program that accepts a password from a user. The program checks if the password meets the following criteria:

- The password must have at least 9 characters.
- The password may contain only
  - alpha (a – z, A – Z) characters
  - numeric (0 – 9) characters
  - The password may NOT contain spaces or other special characters like $, ?, _, etc.

  Hint: The string function isalnum() returns TRUE if all the characters in the string are alpha or numeric and False if one or more characters in the string are not alpha or numeric. See https://www.w3schools.com/python/ref_string_isalnum.asp for additional information
  Note: isalnum () is the only method you may use. You may NOT use other methods like isdigit() or isalpha().

  - The second character in the password must be a numeric (0 – 9)
  - The last character in the password must be an alpha character (a – z, A – Z).

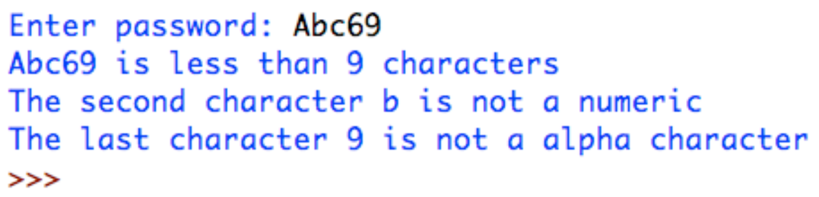
Your program must implement all of the above rules, if rules are violated, print the following messages for each violation. Do not write your own violation messages.

- **password** is less than 9 characters – (replace **password** with the user's input)
- **password** contains a non-alphanumeric character – (replace **password** with the user's input)
- The second character **x** is not a numeric – (replace '**x**' with the actual character).
- The last character **y** must be an alpha character – (replace '**y**' with the actual character).

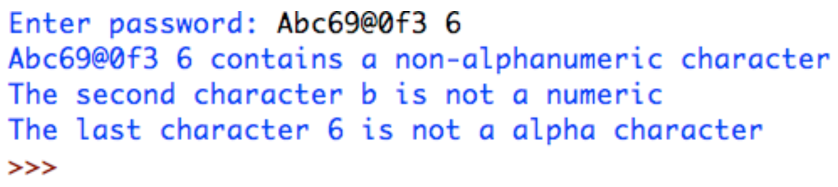
If all rules are satisfied, print 'Congratulations, password meets all the criteria' (replace password with the user's input.

Each rule requires its own IF statement. At the start of the program, assign True to a Boolean variable. Each time a password fails to meet a rule, assign False to the Boolean variable. To determine if the password met all the rules, check the value of the Boolean variable after all the rules have been verified.

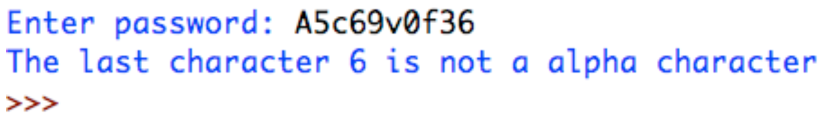
Your program should display the results as shown in the following sample test case:



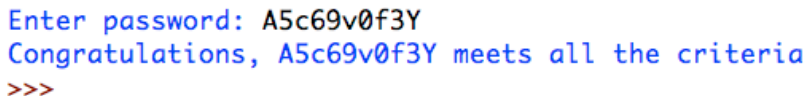
```
Enter password: Abc69
Abc69 is less than 9 characters
The second character b is not a numeric
The last character 9 is not a alpha character
>>>
```



```
Enter password: Abc69@0f3 6
Abc69@0f3 6 contains a non-alphanumeric character
The second character b is not a numeric
The last character 6 is not a alpha character
>>>
```



```
Enter password: A5c69v0f36
The last character 6 is not a alpha character
>>>
```



```
Enter password: A5c69v0f3Y
Congratulations, A5c69v0f3Y meets all the criteria
>>>
```

At the end of the program code, include as comments a test case, one for each rule, that you used to test your code and predicted results. It must be different from the sample test case.

PROBLEM THREE (15 PTS.)
DECISION, STRING, LIST

Write a Python program to build a simple "English language" calculator that does the following:

- Takes a three-character string from the user, where the first and last characters are digits (0-9).

- The second character is '+' or '–' or '*' or '/' which represent addition, subtraction, multiplication and division.
- You should assume that the data input by the user contain valid digits and operators.
- Output is the description of the operation in plain English, as well as the numeric result; for example, if the user inputs **5+3,** the output should be **five plus three is 8.**
- The operators should be translated into English as follows:

+	plus
-	minus
*	multiplied by
/	divided by

You should use **if**, **if else** and **nested if** statements only to code the conditions that determine which operation is requested.

You must take the following situation into consideration:
division operator: you cannot divide by 0, and you should test whether the second number is 0. If it is 0, then you should output a message saying 'Division by zero is not allowed'.
You are to assume that all digits and operators entered are valid. Also, all input is a 3-character string.

Your program should display the results as shown in the following sample test cases:

```
Enter calculation to be performed, e.g. 5+9: 0-6
zero minus six is -6

Enter calculation to be performed, e.g. 5+9: 8/0
Division by zero is not allowed

Enter calculation to be performed, e.g. 5+9: 7*3
seven multiplied by three is 21
```

At the end of the program code, include, as comments, the data you used to test your code and the results obtained. The above test cases are not complete. Your test cases should test all operations and combinations of values.

IF YOU HAVE ANY QUESTIONS REGARDING THIS ASSIGNMENT, PLEASE POST THEM TO THE ASSIGNMENT ONE DISCUSSION FORUM.

Assignment One Grading Rubric For Problem One

Learning outcomes:

- Identify and use a basic subset of Python data types (objects) and methods
- Use built-in functions print, input and eval
- Evaluate algebraic expressions
- Use variables to store data in memory
- Create test data

Problem	Proficient 10 – 9	Nearing Proficiency 8 – 7	Needs Improvement 6 – 0
One	Shows a comprehensive understanding of the basic subset of Python data types, operations, methods and built-in functions discussed in this week's lectures and reading assignments to problem-solve, program and test. • Uses built-in function to determine lowest score • Uses variables to store data in memory as requested • Uses variables to store data in memory as requested • All calculations evaluated correctly • All results are displayed as shown in the sample cases • Completes thorough testing with given data and some data that was not given in sample cases • No syntax errors • Program executes with no run-time errors • Included 2 test cases per problem • All problem specifications are coded.	Shows an adequate understanding of the basic subset of Python data types, operations, methods, and built-in functions discussed in this week's lectures and reading assignments to problem-solve, program and test. • Use one or more Python data types, methods or built-in functions not presented in this week's lectures and reading assignments. • Uses some variables to store data in memory as requested • Some calculations are incorrect. • Display results differ somewhat from that shown in the sample cases. • Only tested with given data or only partially tested. • 1 or 2 syntax errors • Included 1 test case per problem • Some deviations from specifications	Shows minimal or no understanding of the basic subset of Python data types, operations, methods and built-in functions discussed in this week's lectures and reading assignments to problem-solve, program and test. • Uses two or more Python data types, methods or built-in functions not presented in this week's lectures and reading assignments. • Does not use variables to store data in memory • Incorrect calculations. • Display results that use a format different from that shown in the sample cases. • Missing test cases, poorly tested. • More than 2 syntax errors • 1 or more run time errors. • Hardly followed the specifications.

Assignment One Grading Rubric For Problems Two and Three

Learning outcomes:

- Identify and use String and List data types
- Identify the shared operations and functions used by strings and lists
- Describe the difference between strings and lists
- Explain mutable and immutable object creation in memory
- Recognize the impact of decision structures in programming

- Recognize the advantage to using computational thinking for problem-solving

Problem number	Proficient 15 - 13	Nearing Proficiency 12 – 10	Needs Improvement 9 – 0
	Shows a comprehensive understanding of the basic list, string and bool data types, operators and methods and the impact of the decision control structure on problem-solving, programming and testing.	Shows an adequate understanding of the basic list, string and bool data types, operators and methods and the impact of the decision control structure on problem-solving, programming and testing.	Shows minimal or no understanding of the basic list, string and bool data types, operators and methods and the impact of the decision control structure on problem-solving, programming and testing.
Two	• Uses list, string data types to store data in memory • Uses decision structure to implement program design • Creates and uses Boolean variable (bool data type) in decision structure to determine the flow of control • Uses appropriate built-in function to determine alphanumeric characters. • All results are displayed as shown in the sample cases • Program executes with no run-time errors • Program is syntactically correct • Complete thorough testing with given data and some data that was not given in sample cases • All problem specifications are correctly coded	• Uses a Python data type, method or operator not presented in lectures and reading assignments for this week or previous weeks • Uses list, string data types to store some data in memory • Creates Boolean variable, but incorrectly implemented in decision structure • Some results differ somewhat from that shown in the same cases • Some problem specifications are coded incorrectly or missing • Program has one or more syntax errors • Program does not execute because of run-time errors • Only tested with given data or only partially tested. • Some deviations from specifications	• Uses two or more Python data types, methods or operators not presented in lectures and reading assignments for this week and previous weeks • Does not use list, string data types to store data in memory • Does not create Boolean variable • Incorrectly uses decision structure to implement program design • Displays results that use a format different from that shown in the sample cases • Minimal or no results are correct or are incomplete • Missing test data or poorly tested. • More than two syntax errors • Program does not execute because of run-time errors. • Hardly follows the specifications

Problem number	Proficient 15 - 13	Nearing Proficiency 12 – 10	Needs Improvement 9 – 0
Three	• Demonstrates comprehensive understanding of index relationship between two lists • Uses list of English words for all numbers in expression • Use English words for all operators • Correctly retrieves each character in the string for processing • Correctly translates all arithmetic operators to words • Correctly performs all calculations • Displays message for division by zero and does not perform calculation • Displays all results as shown in the sample cases • Completes thorough testing with given data and some data that was not given in sample cases • Program has no syntax errors • Program executes with no run-time errors • All problem specifications are coded	• Uses a Python data type, method or operator not presented in lectures and reading assignments for this week or previous weeks • Demonstrates adequate understanding of index relationship between two lists • Uses list of English words for some numbers in expression • Correctly retrieves some of the characters in the string for processing • Correctly translates some arithmetic operators to words • Correctly performs some calculations • Display message for division by zero, but does the calculation • Displays some results as shown in the same cases • Only tested with given data or only partially tested • Program has one or two syntax errors • Program has run-time error • Some deviations from specifications	• Uses two or more Python data types, methods or operations not presented for this week or previous weeks • Demonstrates minimal or no understanding of index relationship between two lists • Does not use list of English words for numbers • Does not correctly retrieve each characters in the string for processing • Does not correctly translate arithmetic operators to words • Does not display message for division by zero and performs the calculation • Does not correctly perform calculations • Displays results that use a format different from that shown in sample cases • Missing test data, poorly tested. • Program has more than two syntax errors • Program as more than one run time error. • Hardly followed the specifications.