**Data 8 Spring 2020**

**Discussion: Iteration and Conditionals (Lab 05)**

Welcome to Lab 5! This week we will be discussing conditional statements and iteration, which are powerful computational tools that we will use throughout the course. Conditional statements allow data scientists to make more complex decisions with their code, while for loops allow us to repeat the same action many times.

**Question 1.** What does the following function do? *Hint: try to figure out what the function would do on different inputs.*

def mystery\_function(n1, n2):

‘‘‘The function computes the absolute difference between two numbers.’’’

if n2 - n1 > 0:

return n2 - n1

elif n2 - n1 < 0:

return n1 - n2

else:

return 0

**Question 2.** The instructor of a lower division statistics class has assigned you a task: make a function that takes in a student’s score on a scale from 0 to 100 and assigns a letter grade based on the following grade boundaries.

|  |  |
| --- | --- |
| **Score** | **Letter Grade** |
| 0 - 69 | F |
| 70 - 79 | C |
| 80 - 89 | B |
| 90 - 100+ | A |

Complete the functioncompute\_letter\_grades. It takes in a student’s score and returns the letter grade they should receive.

def compute\_letter\_grades(score):

‘’’

compute\_letter\_grades(10)

>>> “F”

compute\_letter\_grades(99)

>>> “A”

‘’’

if \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

elif \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

elif \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

else:

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

if score <= 69:

return “F”

elif score <= 79:

return “C”

elif score <= 89:

return “B”

else:

return “A”

# We could also do it this way!

if score > 89:

return “A”

elif score > 79:

return “B”

elif score < 69:

return “C”

else:

return “F”

**Question 3.** Skeleton code for the function count\_evens is below. The function takes in an array of numbers and returns the count of even numbers in the array.

a. Use a combination of iteration and conditionals to complete the function below.

Hint: the % operator returns the remainder if you divide by a certain number! Example: 11 % 5 = 1

def count\_evens(n\_array):

num\_evens = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

for \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

if \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

def count\_evens(n\_array):

num\_evens = 0

for num in n\_array:

if num % 2 == 0:

num\_evens = num\_evens + 1

return num\_evens

b. Use array operations to complete the function below.

def count\_evens(n\_array):

remainder\_array = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

def count\_evens(n\_array):

remainder\_array = n\_array % 2

return np.count\_nonzero(remainder\_array == 0)

**Question 4.** Complete the function separate\_numbers, which takes in an array of numbers and a boolean value. It should return the number of even values in the array if the argument return\_even is True, or the number of odd values in the array if return\_even is False.

Hint: Use the count\_evens function you defined above!

def separate\_numbers(n\_array, return\_even):

num\_evens = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

if \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

else:

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

def separate\_numbers(n\_array, return\_evens):

num\_evens = count\_evens(n\_array)

if return\_evens == True:

return num\_evens

else:

return len(n\_array) - num\_evens