CSC 120 Lab 05

Instructions

* Do not copy paste code directly. You can refer to online sources and cite your references.
* For each problem, paste a screenshot of the code that you have written with a link to the Colab or original code files for reference.

# 1. Learn to write Pseudocode (20 points)

Your task in this question is to understand what pseudocode is and how to write it. Pseudocode is not real code that you can execute but code that you write on paper to develop an algorithm. Read the article below on how to write pseudocode. Write the pseudocode for finding the maximum number in a list. You should not use any inbuilt function like max or min in your pseudocode but come up with your own way to find the max.

Pseudocode writing guidelines [**https://student.cs.uwaterloo.ca/~cs231/resources/pseudocode.pdf**](https://student.cs.uwaterloo.ca/~cs231/resources/pseudocode.pdf)

***Input: a list List with items in random order***

***Output: the maximum number in the given list***

***List = [0, 50, 10, 66, 5, 89, 7]***

***print (‘list = ‘, list)***

***list.sort()***

***print (list[6])***

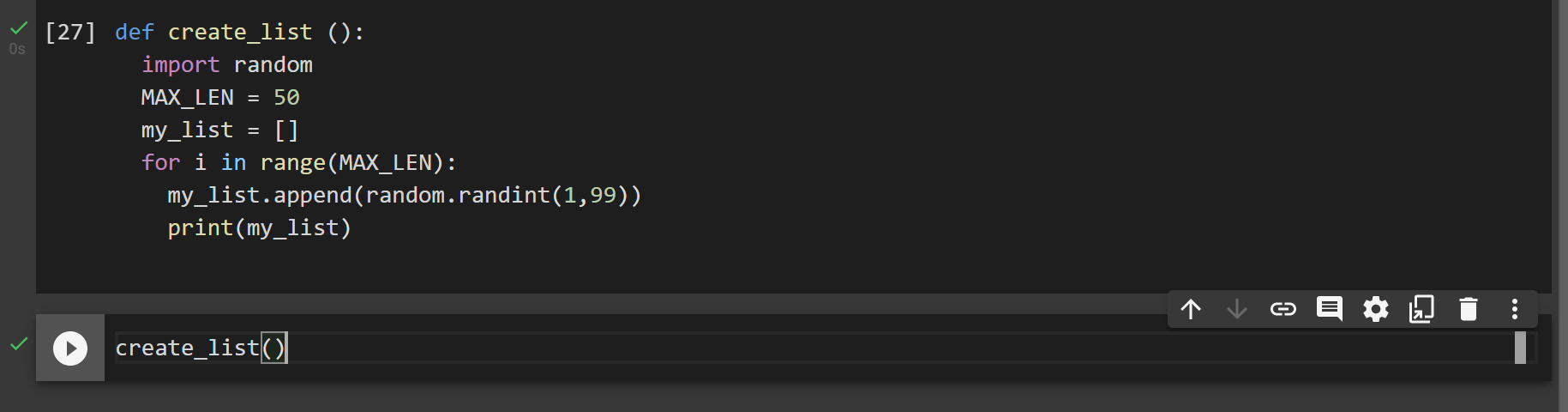
# 2. Code Refactoring exercise (10 points)

Convert the following Python code into a function create\_list(). The goal of refactoring code is to rewrite the code with modifications to make it efficient or readable.The function create\_list needs to return my\_list.

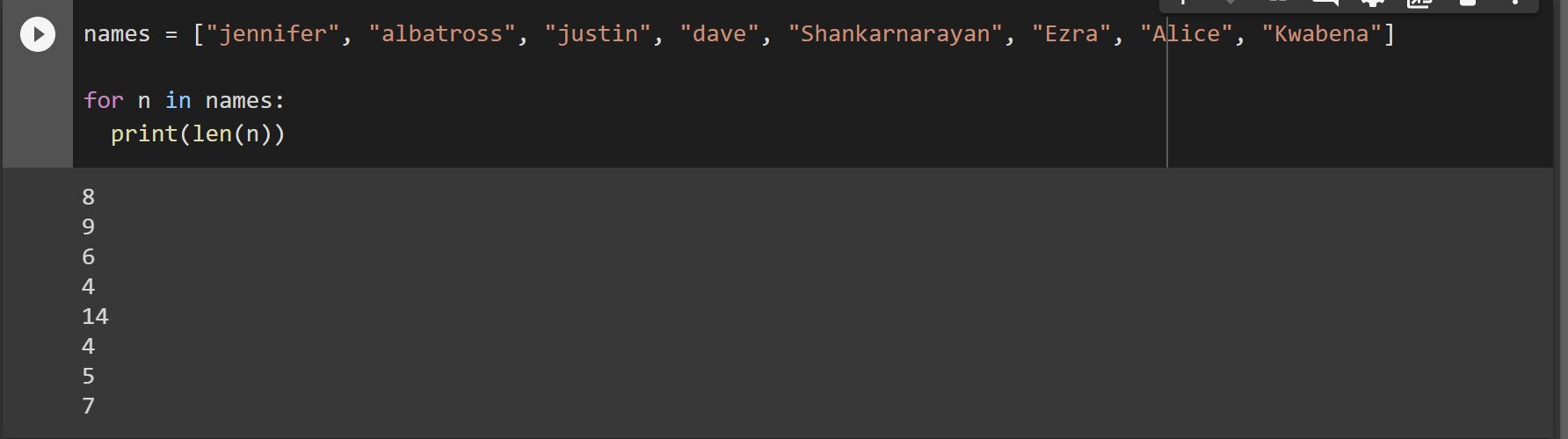
(Watch the lectures for how to create a list or how to write a function) import random MAX\_LEN = 50

my\_list = [] for i in range(MAX\_LEN):

my\_list.append(random.randint(1,99))



**3. Find the length of all the strings in the below list. Use a for loop or a while loop. (10 points)** names = [“Jennifer”, “Albatross”, “Justin”, “Dave”, “Shankarnarayan”, “Ezra”, “Alice”, “Kwabena”]



# 4. Write a program to find the count of all the names that are equal to or shorter than itself in the above list. Use two nested for loops to get all the points. (20 points)

names = [“Jennifer”, “Albatross”, “Justin”, “Dave”, “Shankarnarayan”, “Ezra”, “Alice”, “Kwabena”]

\*\*\*\*\* Sample Output for illustrative purposes only \*\*\*\*\*

Names shorter in length than Jennifer are 5

[“Justin”, “Dave”, “Ezra”, “Alice”, “Kwabena”]

**5. Modify the above program to find the count of all the names that are equal to or shorter than itself in the above list. Use two nested while loops to get all the points. (20 points)**

# 6. Implement a Leaderboard using a List (20 points)

Imagine you have a list ***top\_scores list*** that can store only 5 numbers. Implement an algorithm in Python that simulates a high score leaderboard in a game. The high score leaderboard tracks the top 5 scores that have been posted so far. The behaviour of your program should mimic a leaderboard. Make suitable

assumptions if necessary but write them in your solution, Use the following starter code import random # empty list top\_scores= []

# loop that generates 50 random numbers for i in range(50):

# generate a random number rand\_num = random.randint(0,100)

# # add your code here

* Your code should add the random number generated to the list top\_scores only if the random number is a new high score. A new high score is when the new number is among the top 5 scores observed so far.

Think in terms of a high score board in your favourite game.

* Think in terms of what will happen when your list has less than 5 elements.
* For example, if the list currently has [10,12,5,9,11] and the new number generated is 42, then 42 will get added and the lowest number 5 will get replaced. so the list will be [10,12,42,9,11]. If the next number generated is 3, nothing will happen as 3 is lesser than all the numbers in the list.
* **This is a hard question and you will really need to try to work this out using pencil and paper before trying to implement the algorithm in Python.** But if you get it, you have made a lot of progress!

**\*\*\*\* Extra Credit (30 points) \*\*\*\***

1. What is bubblesort? Write a program that implements the bubblesort algorithm. **(10 points)**

**Bubblesort: process of repeatedly comparing 2 adjecent names and interchanging them if they are not in the correct order relative to each other.**

1. Provide 2 examples each of where you encounter using stacks and queues in the computer

applications that you use. (**20 points)** Here is one example, the undo functionality in text editors can be implemented using a stack data structure. The last changed text snippet can be stored on the top of the stack. More details <https://www.youtube.com/watch?v=TkzZ1typgkI>

String: Forward and backward feature in web browser. String reversal.

Queues: buffer.

**Instructions: Upload the file with the screenshot on Blackboard with your firstname\_lastname.docx**