A logo with text on it

Description automatically generated

ANL252 Python for Data Analytics

Tutor-Marked Assignment

July 2023 Presentation

Submission Date: 10 September 2023

Name: Loke Kum Wai

PI No: M2311263

# Question 1(a)

Why plagiarism in coding happens:

* Student related factors
  + Lack of interest, motivation,
  + Poor time management/procrastination
  + Looking for shortcuts/easy solutions
  + Sharing of work with other students
* Institutional related factors
  + Poorly constructed questions/assignments
* External factors
  + Availability of online resources,
  + Online tutorials,
  + Communities,
  + Open source codes,

How to avoid plagiarism:

Craft questions that

* Are difficult to plagiarise
  + Open-ended questions that require creative problem solving and critical thinking,
  + Which answers are not easy to find online
  + Students’ answers will more likely have to be unique
  + Copied answers will be more easily identifiable
* Test for deeper understanding
  + Explaining written code with clear comments
  + Students can be given code and asked to rewrite for certain changes and functionalities of their choice.
* Go beyond simply coding
  + Test for keeping code clean and easy for other parties to understand
  + Requesting for code to be written in certain standards (similar to how employees follow coding practices of companies)
* Test for understanding instead of originality.
  + Closed ended questions asks for specific answers but can test for understanding
  + Debugging, looking for errors, or modifying functions

# Question 1(b)

Python code as follows:

1. import random

2. import string

3.

4. def generate\_password(min\_length, numbers=True, special\_characters=True):

5. letters = string.ascii\_letters

6. digits = string.digits

7. special = string.punctuation

8.

9. characters = letters

10. if numbers:

11. characters += digits

12. if special\_characters:

13. characters += special

14. pwd = ""

15. has\_number = False

16. has\_special = False

17.

18. while len(pwd) < min\_length:

19. new\_char = random.choice(characters)

20. pwd += new\_char

21.

22. return pwd

23.

24. min\_length = int(input("Enter the minimum length: "))

25. has\_number = input("Do you want to have numbers(y/n)? ").lower() =="y"

26. has\_special = input("Do you want to have special characters (y/n)? ").lower() == "y"

27.

28. pwd = generate\_password(min\_length, has\_number, has\_special)

29. print(pwd)

Adapted from Tech With Tim, 2023.

The code above generates a random password based on the user’s preferences, which are password length, and if it includes numbers and special characters.

Lines 1 and 2 import the random and string libraries. “Random” has functions for generating random numbers or performing random operations. “String” has functions for generating different types of characters.

Lines 4 to 22 define a function to create the password. It requires 3 arguments, which are password length, and two other arguments asking if numbers and symbols should be included.

The function first defines 3 different types of variables, strings of letters, numbers, and special characters.

Next, the variable “letters” is stored in the variable “characters”, meaning that all passwords will include letters. If the arguments for numbers or special characters are “True”, then they will be added to “characters”.

Lines 16 to 20 generate the password using a while loop. While the length of “pwd” is less than the user’s desired length, it will add a new random character from “characters”.

Lines 24 to 26 ask the user for his password preferences. Lines 28 to 29 generate the password with the defined function and the user’s inputs, and prints it as the output.

(199 words)

# Question 1(c)

Rewritten code (code without number reference in appendix):

1. import random

2. import string

3.

4. letters\_str = string.ascii\_letters

5. digits\_str = string.digits

6. symbols\_str = string.punctuation

7.

8. while True:

9. try:

10. passwordLength = int(input("Enter Password Length: "))

11. except ValueError:

12. print("Please enter a number.")

13. else:

14. break

15.

16. characters = ""

17.

18. def get\_preferences(user\_preference):

19. while True:

20. x = input(f"Include {user\_preference}? (y/n)").lower()

21. if x == "y" or x == "n":

22. break

23. else:

24. print("Enter 'y' or 'n'.")

25. return x

26.

27. while True:

28. letters = "letters"

29. letters = get\_preferences(letters)

30.

31. digits = "digits"

32. digits = get\_preferences(digits)

33.

34. symbols = "symbols"

35. symbols = get\_preferences(symbols)

36.

37. if symbols == "n" and digits == "n" and letters == "n":

38. print("At least one needs to be included.")

39. else:

40. break

41.

42. if letters == "y":

43. characters += letters\_str

44.

45. if digits == "y":

46. characters += digits\_str

47.

48. if symbols == "y":

49. characters += symbols\_str

50.

51. while True:

52. password = ""

53.

54. for i in range(passwordLength):

55. password\_char = random.choice(characters)

56. password = password + password\_char

57.

58. print(f"Generated password: {password}")

59.

60. regenerate = input("Do you want to regenerate the password? (y/n)").lower()

61. if regenerate != "y":

62. break

Output:

Enter Password Length24

Include letters? (y/n)y

Include digits? (y/n)y

Include symbols? (y/n)y

Generated password: &<?NZYA"+faoUoCL~q'\LF6y

Do you want to regenerate the password? (y/n)n

First, I tried to understand the code and how it works. Then I started from scratch and referred occasionally while adding my additional features. This helps to reduce the amount of direct copying of the original work. However, in cases where the code is much simpler and solutions are closed-ended, it may be more easily detected as plagiarism by checkers.

Lines 1-6 is similar to the original code.

Lines 8 to 14, the code prevents a value error when the user does not provide an integer for the password length, which was missing from the original code.

Next, instead of having all passwords to include letters by default, it asks the user if it should be included. This allows for random numbers or symbols as passwords.

I also defined a “get\_preferences” function. Originally, the user is asked “(y/n)?” for inputs, but the code only checks for “y”. “get\_preferences” makes sure that the user only inputs “y” or “n”, or will continue prompting for it. It also makes the input lowercase so that “Y” or “N” is also acceptable.

“get\_preferences” is then used to ask the user if he wants letters, digits, and symbols in the password in lines 27 to 40. It also includes an error message if the user chooses none of the preferences (which makes it impossible to create a password). Using “get\_preferences” in the while loop from line 27 to 35 also makes it more efficient because it does not need to be repeated.

Lines 42 to 58 work similarly to the original code. Line 60 to 62, the user is asked if they want to regenerate the code.

Overall, the code is more robust where user inputs are not regarded as errors, and it does not break unnecessarily.

(292 words)

# Question 2

1. products = ['Laptop', 'Mouse', 'Webcam', 'Keyboard', 'Speaker']

2. print("We have a list of products here:")

3. for i in range(len(products)):

4. print(i + 1, products[i])

5. updated\_items = []

6. query = "y"

7. while query == "y":

8. item\_input = input(f"Hello! What do you want to buy? (Input 1-{len(products)} or Product Name)").capitalize()

9. if item\_input.isdigit():

10. item\_index = int(item\_input)

11. if 0 < item\_index < len(products) + 1:

12. item = products[int(item\_input) - 1]

13. else:

14. print("Wrong. Please choose a number from the list")

15. continue

16. elif item\_input in products:

17. item = item\_input

18. else:

19. print(f'Wrong product! Please try again.')

20. continue

21. while True:

22. try:

23. price\_of\_item = float(input(f"How much is {item} (SGD)? "))

24. except ValueError:

25. print("Please Enter a Number For Price ")

26. else:

27. break

28. updated\_items.append([item, price\_of\_item])

29. while True:

30. query = input("Would you like to continue? (y/n) ").lower()

31. if query == "y" or query == "n":

32. break

33. else:

34. print("Enter 'y' or 'n'")

35. print(f'This is the updated shopping list: ')

36. for j in range(len(updated\_items)):

37. print(updated\_items[j][0] + " $" + str(updated\_items[j][1]))

1. In the original code, whenever an input is required, the code will continue or stop if it is invalid. The proposed code prompts the user for correct answers and does not exit unnecessarily. This is done by various if-else statements, while loops and try function. In the first question, the proposed code now allows the user to enter the corresponding number, or type the name of the product, which is not case-sensitive. Previously, the user has to input the exact string in the list. These improve overall reliability and quality.

2. Previously, the product and updated shopping list were printed directly as a list. The proposed code prints them as separate lines. Instead of asking, “How much is it”, the proposed code inserts the name of the product into the question based on the previous input. At the end, if the user wishes to continue, the updated shopping list is printed for the user. These make it more readable for the user.

3. If new products are added to the list, it will be updated automatically when asking what to buy. This is done by finding the length of the list in “products”. This improves the code maintainability.

(199 words)

# References

Tech With Tim, 2023, Mini Python Project Tutorial - Password Generator, <https://www.youtube.com/watch?v=XCIBOl3FTKo>

# Appendix

Codes without line numbering as follows:

## Question 1(b):

Code from Tech With Tim, 2023, Mini Python Project Tutorial - Password Generator, <https://www.youtube.com/watch?v=XCIBOl3FTKo>, before adaptation.

import random

import string

def generate\_password(min\_length, numbers=True, special\_characters=True):

letters = string.ascii\_letters

digits = string.digits

special = string.punctuation

characters = letters

if numbers:

characters += digits

if special\_characters:

characters += special

pwd = ""

meets\_criteria = False

has\_number = False

has\_special = False

while not meets\_criteria or len(pwd) < min\_length:

new\_char = random.choice(characters)

pwd += new\_char

if new\_char in digits:

has\_number = True

elif new\_char in special:

has\_special = True

meets\_criteria = True

if numbers:

meets\_criteria = has\_number

if special\_characters:

meets\_criteria = meets\_criteria and has\_special

return pwd

min\_length = int(input("Enter the minimum length: "))

has\_number = input("Do you want to have numbers(y/n)? ").lower() =="y"

has\_special = input("Do you want to have special characters (y/n)? ").lower() == "y"

pwd = generate\_password(min\_length, has\_number, has\_special)

print(pwd)

## Question 1(c):

import random

import string

letters\_str = string.ascii\_letters

digits\_str = string.digits

symbols\_str = string.punctuation

while True:

try:

passwordLength = int(input("Enter Password Length"))

except ValueError:

print("Please enter a number")

else:

break

characters = ""

def get\_preferences(user\_preference):

while True:

x = input(f"Include {user\_preference}? (y/n)").lower()

if x == "y" or x == "n":

break

else:

print("Enter 'y'or 'n'")

return x

while True:

letters = "letters"

letters = get\_preferences(letters)

digits = "digits"

digits = get\_preferences(digits)

symbols = "symbols"

symbols = get\_preferences(symbols)

if symbols == "n" and digits =="n" and letters == "n":

print("At least one needs to be included")

else:

break

if letters == "y":

characters += letters\_str

if digits == "y":

characters += digits\_str

if symbols == "y":

characters += symbols\_str

while True:

password = ""

for i in range(passwordLength):

password\_char = random.choice(characters)

password = password + password\_char

print(f"Generated password: {password}")

regenerate = input("Do you want to regenerate the password? (y/n)").lower()

if regenerate != "y":

break

## Question 2:

products = ['Laptop', 'Mouse', 'Webcam', 'Keyboard', 'Speaker']

print("We have a list of products here:")

for i in range(len(products)):

print(i+ 1, products[i])

updated\_items = []

query = "y"

while query == "y" :

item\_input = input(f"Hello! What do you want to buy? (Input 1-{len(products)} or Product Name)").capitalize()

if item\_input.isdigit():

item\_index = int(item\_input)

if 0 < item\_index < len(products) + 1 :

item = products[int(item\_input) - 1]

else:

print("Wrong. Please choose a number from the list")

continue

elif item\_input in products:

item = item\_input

else:

print(f'Wrong product! Please try again.')

continue

while True:

try:

price\_of\_item = float(input(f"How much is {item} (SGD)? "))

except ValueError:

print("Please Enter a Number For Price ")

else:

break

updated\_items.append([item, price\_of\_item])

while True:

query = input("Would you like to continue? (y/n) ").lower()

if query == "y" or query == "n":

break

else:

print("Enter 'y'or 'n'")

print(f'This is the updated shopping list: ')

for j in range(len(updated\_items)):

print(updated\_items[j][0] + " $" + str(updated\_items[j][1]))