CIND830 - Python Programming for Data Science

Assignment 1 (15% of the final grade)

Due on October 23, 2023, 11:59 PM

This is a Jupyter Notebook document that extends a simple formatting syntax for authoring HTML and PDF. Review this website for more details on using Jupyter Notebooks.

Consider using a Jupyter Notebook platform to complete this assignment. Ensure using **Python 3.9 release or higher**, then complete the assignment by inserting your Python code wherever seeing the string `#WRITE YOUR ANSWER HERE.`

You are expected to submit the notebook file (in IPYNB format) and the exported version (either in PDF or HTML) in the same Assignment link in D2L. Use these guidelines to submit **both** the IPYNB and the exported file (HTML or PDF). Failing to submit both files will be subject to mark deduction.

Please be advised that you cannot get more than 100% in this assignment, and the **BONUS** question (if there is any) will only be graded if all other questions have been submitted.

Coverage:

1. Data Types and Expressions
2. Repetition Statements
3. Selection Statements
4. Strings, Lists and Text Files

```
print("CIND830 Assignement 1\nBy Stephanie Boissonneault\nOctober, 3rd 2023")
!python --version
```

```
CIND830 Assignement 1
By Stephanie Boissonneault
October, 3rd 2023
Python 3.10.12
```

Question 1 [30 pts]:

Suppose you have been hired by a local bookstore named "BookNook". The owner, Ms. Walker, is eager to integrate Python for maintaining her bookstore's records. She has outlined a few tasks that can help her manage the store better.

Ms. Walker has a list of book titles and their respective prices. She'd like to see the total expense she has made in acquiring the books and the average cost of a book.

a- [10 pts]

Given;

```
book_titles = ["The Great Gatsby", "Moby Dick", "To Kill a Mockingbird", "1984", "Pride and Prejudice"]
book_prices = [10.99, 12.45, 8.50, 9.99, 7.80]
```

1. Calculate the total expense made by Ms. Walker in acquiring these books. (3p)
2. Find out the average cost of a book in the list.(4p)
3. Display the title of the most expensive book. (3p)

```
# 1.
book_titles = ["The Great Gatsby", "Moby Dick", "To Kill a Mockingbird", "1984", "Pride and Prejudice"]
book_prices = [10.99, 12.45, 8.50, 9.99, 7.80]
print(book_titles)
print(book_prices, "\n")

total_expense = sum(book_prices)
print("1. The total expense in acquiring these books is $", str(total_expense))

# 2
import statistics
mean_cost = statistics.fmean(book_prices)
print("2. The average cost of a book in the list is $", str(round(mean_cost, 2)))

# Could also be calculated using the following:
# avg_cost <- (sum(book_prices)/len(book_prices))
# print(round(avg_cost, 2))"""

# 3
print("3. The most expensive book is ", str(book_titles[1]), " listed at $", str(max(book_prices)))
```

```
['The Great Gatsby', 'Moby Dick', 'To Kill a Mockingbird', '1984', 'Pride and Prejudice']
[10.99, 12.45, 8.5, 9.99, 7.8]

1. The total expense in acquiring these books is $ 49.73
2. The average cost of a book in the list is $ 9.95
3. The most expensive book is  Moby Dick  listed at $ 12.45
```

sb- [10 pts]

The bookstore is planning a sale. For each book that costs more than $10, Ms. Walker wants to offer a 10% discount.

1. Create a new list that holds the discounted prices of the books that cost more than $10. (5p)
2. Print each book title from book_titles and its new price side by side if it has a discount. (5p)

```
# 1 Multiplying the book prices by 0.9 (or 90%) to apply the 10% discount:
print(book_prices)
discounted_prices =[(book_prices[0]*0.9), (book_prices[1]*0.9),
                    book_prices[2], book_prices[3], book_prices[4]]
print(discounted_prices, "\n")

# 2 Printing the discounted prices next to book titles with disocunts:
x = 0
while x < len(book_titles):
    if x <= 1:
        print(book_titles[x], "Original price $", book_prices[x], "Discounted price $", str(round(discount
        x += 1
    else:
        print(book_titles[x], "Original price $", book_prices[x])
        x += 1
```

```
[10.99, 12.45, 8.5, 9.99, 7.8]
[9.891, 11.205, 8.5, 9.99, 7.8]

The Great Gatsby Original price $ 10.99 Discounted price $ 9.89
Moby Dick Original price $ 12.45 Discounted price $ 11.21
To Kill a Mockingbird Original price $ 8.5
1984 Original price $ 9.99
Pride and Prejudice Original price $ 7.8
```

c- [10 pts]

Ms. Walker sometimes gets requests for recommendations. Depending on the reader's age, she recommends different genres.

Write a program that:

1. Asks for the age of the reader. (5p)
2. Based on the given age, recommends a book genre using the following criteria: (5p)

- If the age is below 12, recommend "Children's Fiction".
- If the age is between 12 and 18, recommend "Young Adult".
- If the age is between 19 and 30, recommend "Contemporary Fiction".
- If the age is between 31 and 50, recommend "Historical Fiction".
- If the age is 51 and above, recommend "Classics".

Ensure that the user inputs a valid age. If the user inputs a non-numeric value or an age less than 0, prompt them to enter the age again.

```
print('Type in your age and hit "enter" to receive a book genre recomendation:')
try:
    age = int(input("How old are you?"))
    if age < 12:
        print("I recommend Children's Fiction")
    elif age <= 18:
        print("I recommend Young Adult")
    elif age <= 30:
        print("I recommend Contemporary Fiction")
    elif age <= 50:
        print("I recommend Historical Fiction")
    else:
        print("I recommend Classics")

except:
    print("Hmm I didn't quite catch that. Please try again using a whole number.")
```

```
Type in your age and hit "enter" to receive a book genre recomendation:
Hmm I didn't quite catch that. Please try again using a whole number.
How old are you?
```

Question 2 [30 pts]:

Suppose you have recently joined a Data Science team as a Python Developer at a large retail corporation. You are tasked with analyzing sales data in order to provide valuable insights to the team.

a- Sales Trend Analysis [15 pts]

Given a sales data list representing the total sales of a particular product in each month over a year as follows:

```
monthly_sales = [250, 280, 290, 260, 275, 300, 310, 305, 290, 285, 280, 300]
```

1. Your first task is to write a python script that uses a for loop to calculate and print the sum of sales over the year.

```
monthly_sales = [250, 280, 290, 260, 275, 300, 310, 305, 290, 285, 280, 300]

```

```
total = 0
for number in monthly_sales:
    total += number
print("The sum of sales over the year is $", total)
```

```
The sum of sales over the year is $ 3425
```

2. Next, you are required to analyze the sales trend over the year. Write a python script using a for loop to print each monthly sales and indicate whether it is increased, decreased, or stable compared to the previous month.

```
print(monthly_sales)
count = 1    #To track the months

last_month = 250
for month in monthly_sales:
    sale = (month - last_month)
    if sale > 0:
        print("Sales in month", count, "increased by $", sale, "since last month.")
    elif sale < 0:
        print("Sales in month", count, "decreased by $", sale, "since last month.")
    else:
        print("Sales in month ", count, "are stable since last month.")
    last_month = month
    count += 1
```

```
[250, 280, 290, 260, 275, 300, 310, 305, 290, 285, 280, 300]
Sales in month  1 are stable since last month.
Sales in month 2 increased by $ 30 since last month.
Sales in month 3 increased by $ 10 since last month.
Sales in month 4 decreased by $ -30 since last month.
Sales in month 5 increased by $ 15 since last month.
Sales in month 6 increased by $ 25 since last month.
Sales in month 7 increased by $ 10 since last month.
Sales in month 8 decreased by $ -5 since last month.
Sales in month 9 decreased by $ -15 since last month.
Sales in month 10 decreased by $ -5 since last month.
Sales in month 11 decreased by $ -5 since last month.
Sales in month 12 increased by $ 20 since last month.
```

b - Sales Growth Calculation [15 pts]

Now, your team leader wants to know the sales growth of this product from month to month in percentage, as well as the average growth over the year.

Sales growth is calculated as: `((Sales of current month - Sales of last month) / Sales of last month) * 100`

1. Write a python script using a for loop to calculate and print the sales growth of each month compared to the previous month.

```
# Sales growth month to month:
prev_month = 250
total_growth = 0
count = 1

for month in monthly_sales:
    growth = ((month - prev_month)/prev_month)*100
    print ("The sales growth for month", count, "is", round(growth,2), "%")
    prev_month = month
    count += 1
    total_growth += growth

```

```
# Average growth over the year:
avg_growth = total_growth/(len(monthly_sales)-1)
print("The avergage sales growth is", round(avg_growth, 2), "%")
```

```
The sales growth for month 1 is 0.0 %
The sales growth for month 2 is 12.0 %
The sales growth for month 3 is 3.57 %
The sales growth for month 4 is -10.34 %
The sales growth for month 5 is 5.77 %
The sales growth for month 6 is 9.09 %
The sales growth for month 7 is 3.33 %
The sales growth for month 8 is -1.61 %
The sales growth for month 9 is -4.92 %
The sales growth for month 10 is -1.72 %
The sales growth for month 11 is -1.75 %
The sales growth for month 12 is 7.14 %
The avergage sales growth is 1.87 %
```

2. To better understand the overall performance, compute the average sales growth over the year. Use a while loop to calculate the sum of sales growths for all months first. Average monthly sales growth is then calculated as:

```
total growth over the year / (number of months - 1)
```

Print out the average sales growth.

```
# Storing the monthly sales growth from previous question into the variable monthly_growth:
monthly_sales = [250, 280, 290, 260, 275, 300, 310, 305, 290, 285, 280, 300]
prev_month = 250
total_growth = 0
monthly_growth = []
for month in monthly_sales:
    growth = ((month - prev_month)/prev_month)*100
    prev_month = month
    monthly_growth.append(round(growth,2))
print(monthly_growth, "\n")


# Using the while loop to calculate the sum of sales growth:
x = 0
while x < len(monthly_growth):
    total_growth += monthly_growth[x]
    x += 1
print("The sum of sales growths for all months it", total_growth, "%")

# Average growth over the year:
avg_growth = total_growth/(len(monthly_sales)-1)
print("The avergage sales growth is", round(avg_growth, 2), "%")
```

```
[0.0, 12.0, 3.57, -10.34, 5.77, 9.09, 3.33, -1.61, -4.92, -1.72, -1.75, 7.14]

The sum of sales growths for all months it 20.56 %
The avergage sales growth is 1.87 %
```

Question 3 [40 pts]:

a - String Manipulation [10 pts] You have a string that contains a customer review:

```
review = "This product is amazing! It has changed my life."
```

1. Consider the following code snippet to extract the word `amazing` from the string and store it in a variable:

   ```
   review = "This product is amazing! It has changed my life."
   start_index = review.find("amazing")
   end_index = start_index + len("amazing")
   extracted_word = review[start_index:end_index]
   print("Extracted word:", extracted_word)
   ```

   Provide a detailed explanation for each line of code.

2. Consider the following code snippet to replace the word `amazing` with `incredible` in the original string:

   ```
   modified_review = review.replace("amazing", "incredible")
   print("Modified review:", modified_review)
   ```

   Provide an alternative solution to replace the word `amazing` with `incredible`.

```
# 1. Provide a detailed explanation for each line of code.

# Storing the customer's review as a string into a variable called review.
review = "This product is amazing! It has changed my life."

# Using python's find() function to search the position of the object "review"
# for the first occurrence of the word "amazing" and storing it in a variable.
# Note that indexing in python starts at 0 so start_index will return 16.
start_index = review.find("amazing")

# Adding the start index (16) to the length of the characters in "amazing" (7).
# This will help find the position of the last character in "amazing".
# The return value is 23 and is stored in a variable.
end_index = start_index + len("amazing")

# Characters from the word "amazing" are extracted based on their position
# from the original string (16:23) and are stored in a variable.
extracted_word = review[start_index:end_index]

# The extracted word is printed.
print("Extracted word:", extracted_word)

```

```
Extracted word: amazing
```

```
# 2
review = "This product is amazing! It has changed my life."
start_index = review.find("amazing")
end_index = start_index + len("amazing")

print("The original review is:", review)
# print(start_index)
# print(end_index)
# print(len(review))

modified_review = review[0:16] + "incredible" + review[23:48]
print("The modified review is:", modified_review)
```

```
The original review is: This product is amazing! It has changed my life.
The modified review is: This product is incredible! It has changed my life.
```

b - List Operations [10 pts] You have a list of words extracted from multiple customer reviews:

```
words_list = ['great', 'product', 'value', 'quality']
```

1. Add the words "excellent" and "affordable" to the list.
2. Sort the list in alphabetical order.

```
# 1
words_list = ['great', 'product', 'value', 'quality']
new_words_list = words_list + ['excellent', 'affordable']
print(new_words_list)

# 2
new_words_list.sort()
print(new_words_list)
```

```
['great', 'product', 'value', 'quality', 'excellent', 'affordable']
['affordable', 'excellent', 'great', 'product', 'quality', 'value']
```

c - String and List Integration [10 pts] Using the words_list from Q3.b:

1. Convert the list of words into a single string where each word is separated by a comma and a space.
2. How would you convert this string back into a list?

```
print(new_words_list)
print(type(new_words_list))

# 1. Convert to a single string:
new_string= " ".join(new_words_list)
print(new_string)
print(type(new_string))

# 2. Convert back to a list:
new_list = list(new_words_list)
print(new_list)
print(type(new_list))

```

```
['affordable', 'excellent', 'great', 'product', 'quality', 'value']
<class 'list'>
affordable excellent great product quality value
<class 'str'>
['affordable', 'excellent', 'great', 'product', 'quality', 'value']
<class 'list'>
```

d - Working with Text Files [10 pts] You have a text file named `customer_reviews.txt` that contains multiple lines of customer reviews.

1. Create this file and populate it with the following lines of customer reviews:

```
I love this product! Highly recommended.
Not what I expected. Could be better.
Excellent quality and fast shipping!
```

2. Read the created file and store all lines in a list. Then, count the number of lines in the file.

```
# 1. Create the file and populate it:
textFile = open("customer_reviews.txt", "w")
textFile.write("I love this product! Highly recommended\n\
Not what I expected. Could be better.\n\
Excellent quality and fast shipping!")
textFile.close()

#2. Read the created file:
textFile = open("customer_reviews.txt", "r")
print(textFile.read())
textFile.close()
```

```
I love this product! Highly recommended
Not what I expected. Could be better.
Excellent quality and fast shipping!
```

```
#3. Store all contents in a list:
textFile = open("customer_reviews.txt", "r")
content = textFile.readlines()
textList= [line.strip() for line in content]
print(textList)

#4. Count the number of lines in the file:
total_lines = len(textList)
print("The number of lines in the file is", total_lines)

textFile.close()
```

```
['I love this product! Highly recommended', 'Not what I expected. Could be better.', 'Excellent quality and fast shippi
The number of lines in the file is 3
```

This is the end of assignment 1