Python Coursework Guide

Author: Richard Krasso
Course: WEB 335 Introduction to NoSQL
Date: May 28, 2024

Table of Contents

Hands-On 1.1: Building Your First Python Program

In this hands-on project, you will be installing Python and building a simple Python program that prints a quote of your choosing.

Instructions:

1. Install Python on your computer. If you need help with this, refer to this week's reading and video material.

2. Build a simple Python program that prints a quote of your choosing. Be sure to give credit to the originating author of the quote you selected.

Hints:

- Remember to include file header and code comments. For example,
- Your program should use the print function to output the quote and author.
- You can use any quote you want, just make sure you give credit to the original author.

Code Examples:
Here are some code examples that will help you with the assignment.

1. Example of file header and code comments

```
"""
Author: Your Name
Date: Current Date
File Name: example.py
Description: This file demonstrates the use of both file header comments and normal level comments
"""

# Importing required module (normal level comment explaining the following code block)
import os

# Getting the current working directory
current_dir = os.getcwd()  # normal level comment explaining this line of code

print(current_dir)  # Printing the current working directory
```

Hands-On 2.1: Functions

In this hands-on project, you will be building a simple Python program that simulates a lemonade stand. You will create functions to calculate the cost of making lemonade and the profit from selling it.

Instructions:

1. Create a new Python program and name it <yourLastName>_lemonadeStand.py and add it to a folder named week_3 in your GitHub repository.

2. Create a function named calculate_cost with two parameters: lemons_cost and sugar_cost. In the body of the function, return the total cost of making the lemonade.

3. Create a function named calculate_profit with three parameters: lemons_cost, sugar_cost, and selling_price. In the body of the function, return the profit from selling the lemonade.

4. Create variables to test each function. Use a variable to build a string for the results. Use the format: (cost of lemons) + (cost of sugar) = (total cost).

5. Call each function passing in the variables you created in step 4 and print the results to the console using an output variable and string concatenation.

Hints:

- Remember to include file header and code comments.
- Your program should use the print function to output the cost and profit calculations.

Code Examples:
Here is a code example that will help you with the assignment.

```
"""
Author: Your Name
Date: Current Date
File Name: example.py
Description: This file demonstrates the use of both file header comments and normal level comments
"""

# Defining a function (normal level comment explaining the following code block)
def calculate_cost(lemons_cost, sugar_cost):  # normal level comment explaining this line
```

```
    total_cost = lemons_cost + sugar_cost  # Calculating the total cost

    return total_cost  # Returning the total cost

print(calculate_cost(5, 3))  # Printing the total cost
```

Hands-On 3.1: Conditionals, Lists, and Loops

In this hands-on project, you will be building a Python program that manages a weekly schedule for a lemonade stand. You will create lists to represent different tasks and use loops and conditionals to process these lists.

Instructions:

1. Create a new Python program and name it <yourLastName>_lemonadeStandSchedule.py and add it to a folder named week_5 in your GitHub repository.

2. Create a list of at least 5 tasks related to running a lemonade stand.

3. Use a for loop to iterate over the list of tasks and print them to the console window.

4. Create a list of days (Sunday through Saturday).

5. Use a for loop to iterate over the list of days and add an if...else statement to display what the task is for each day. For Saturday and Sunday display a message indicating it is a day off and you should rest. For all other days, display a message indicating the day of the week and the corresponding task from the tasks list.

Hints:

- Remember to include file header and code comments.
- Your program should use the print function to output the tasks and days of the week.

Code Examples:
Here is a code example that will help you with the assignment.

```
"""
Author: Your Name
Date: Current Date
File Name: example.py
Description: This file demonstrates the use of both file header comments and normal level comments
"""
```

```
# Defining a list (normal level comment explaining the following code block)
tasks = ["Buy lemons", "Make lemonade", "Sell lemonade", "Count earnings", "Clean
up"]  # normal level comment explaining this line of code

# Using a for loop to iterate over the list
for task in tasks:  # normal level comment explaining this line of code
   print(task)  # Printing the task
```

Hands-On 4.2: Python with MongoDB, Part I

In this hands-on project, you will be building a Python program that connects to your MongoDB database and performs various operations.

Instructions:

1. Create a new Python file and name it <yourLastName>_usersp1.py and add it to a folder named week_6 in your GitHub repository.

2. Install the pymongo package using pip. If you need help with this, refer to Pythons official documentation for installing pip packages.

3. Using the provided code examples, build a Python program that connects to your web335DB database.

4. Write the Python code to display all documents in the user's collection.

5. Write the Python code to display a document where the employeeId is 1011.

6. Write the Python code to display a document where the lastName is Mozart.

Hints:

- Remember to include file header and code comments.
- Your program should use the print function to output the documents.

Code Examples:

1. Connecting to MongoDB:

```
"""
Title: pymongo_conn.py
Author: Professor Krasso
Date: 27 June 2022
Description: Exercise 6.3
"""
```

```
# Import the MongoClient
from pymongo import MongoClient

# Build a connection string to connect to
client =
MongoClient("mongodb+srv://web335_user:s3cret@cluster0.lujih.mongodb.ne
t/web335DBretryWrites=true&w=majority")

print(client)
```

2. Example of finding all documents:

```
"""
Title: find_ex1.py
Author: Professor Krasso
Date: 27 June 2022
Description: Exercise 6.3
"""

# Import the MongoClient
from pymongo import MongoClient
import datetime

# Build a connection string to connect to
client =
MongoClient("mongodb+srv://web335_user:s3cret@cluster0.lujih.mongodb.ne
t/web335DBretryWrites=true&w=majority")

# Configure a variable to access the web335DB
db = client['web335DB']

# Call the find function to display all of the users in the collection; use
projections to only show the first and last names.
for user in db.users.find({}, {"firstName": 1, "lastName": 1}):
  print(user)
```

3. Example of finding one document:

```
"""
Title: find_one_ex2.py
Author: Professor Krasso
Date: 27 June 2022
Description: Exercise 6.3
```

```
"""

# Import the MongoClient
from pymongo import MongoClient
import datetime

# Build a connection string to connect to
client =
MongoClient("mongodb+srv://web335_user:s3cret@cluster0.lujih.mongodb.ne
t/web335DBretryWrites=true&w=majority")

# Configure a variable to access the web335DB
db = client['web335DB']

# Call the find_one function to display a user document by id
print(db.users.find_one({"employeeId": "1007"}))
```

Hands-On 5.2: Python with MongoDB, Part II

In this hands-on project, you will be building a Python program that connects to MongoDB database and performs various CRUD operations.

Instructions:

1. Create a new Python program and name it <yourLastName>_usersp2.py and add it to a folder named week_7 in your GitHub repository.

2. Using the provided code examples, build a Python program that connects to your web335DB database.

3. Write the Python code to create a new user document.

4. Write the Python code to prove the document was created.

5. Write the Python code to update the email address of the document you created in step 3.

6. Write the Python code to prove the document was updated.

7. Write the Python code to delete the document that was created in step 3.

8. Write the Python code to prove the document was deleted.

Hints:

- Remember to include file header and code comments.
- Your program should use the print function to output the documents.

Code Examples:

1. Example of creating a document:

```
"""
Title: insert_one_ex1.py
Author: Professor Krasso
Date: 27 June 2022
Description: Exercise 6.3
"""

# Import the MongoClient
from pymongo import MongoClient
import datetime

# Build a connection string to connect to
client =
MongoClient("mongodb+srv://web335_user:s3cret@cluster0.lujih.mongodb.net/web335DBretryWrites=true&w=majority")

# Configure a variable to access the web335DB
db = client['web335DB']

# Create a new user document and added it to the users collection
hayden = {
  "firstName": "Joseph",
  "lastName": "Haydn",
  "employeeId": "1013",
  "email": "jhaydn@me.com",
  "dateCreated": datetime.datetime.utcnow()
}

# Insert the document into the users collection
hayden_user_id = db.users.insert_one(hayden).inserted_id
print(hayden_user_id)

# Prove the insert worked by searching the collection for the document
print(db.users.find_one({"employeeId": "1013"}))
```

2. Example of updating a document:

```
"""
```

```
Title: update_one_ex.py
Author: Professor Krasso
Date: 27 June 2022
Description: Exercise 6.3
"""

# Import the MongoClient
from pymongo import MongoClient
import datetime

# Build a connection string to connect to
client =
MongoClient("mongodb+srv://web335_user:s3cret@cluster0.lujih.mongodb.ne
t/web335DBretryWrites=true&w=majority")

# Configure a variable to access the web335DB
db = client['web335DB']

# Create an update query to change the user's email address
db.users.update_one(
  {"employeeId": "1013"},
  {
    "$set": {
      "email": "joseph.haydn@me.com"
    }
  }
)

# Prove the update worked by searching the collection for the user by
employeeId
print(db.users.find_one({"employeeId": "1013"}))
```

3. Example of deleting a document:

```
"""
Title: delete_one_ex1.py
Author: Professor Krasso
Date: 27 June 2022
Description: Exercise 6.3
"""

# Import the MongoClient
from pymongo import MongoClient
import datetime
```

```
# Build a connection string to connect to
client =
MongoClient("mongodb+srv://web335_user:s3cret@cluster0.lujih.mongodb.net/web335DBretryWrites=true&w=majority")

# Configure a variable to access the web335DB
db = client['web335DB']

# Build a query to remove a user document
result = db.users.delete_one({
   "employeeId": "1013"
})

# Display the results of the query
print(result)

# Prove the delete worked by searching the collection for the deleted document
print(db.users.find_one({"employeeId": "1013"}))
```

References

Copilot. (n.d.). OpenAI. *Microsoft Copilot*. computer software. Retrieved December 19, 2023, from https://www.microsoft.com/en-us/microsoft-copilot.