CIS 103: Introduction to Programming
Lab 1: Getting Started with Python and GitHub

Md Ali

Due: 09/07/2024 @ 1159

Objective:

- Learn the basics of Python programming.
- Understand how to write, save, and run Python scripts.
- Familiarize yourself with basic Python syntax, including variables, data types, and output.
- Get started with version control using Git and GitHub.

Part 1: Getting Started with GitHub

In this part of the lab, you will set up Git and GitHub on your computer. Git is a version control system that lets you manage and keep track of your source code history. GitHub is a cloud-based hosting service that lets you manage Git repositories. If you need help, please let me know.

Step 1: Create a GitHub Account

If you don't already have a GitHub account, go to https://github.com/ and sign up for a free account.

Step 2: Install Git

Install Git on your computer by following the instructions at https://git-scm.com/book/en/v2/Getting-Started-Installing-Git.

Step 3: Set Up Git

Open a terminal or command prompt and configure your Git username and email by running the following commands:

```sh
git config --global user.name "Your Name"
git config --global user.email "youremail@example.com"
```

Step 4: Create a New Repository

1. Go to your GitHub account and create a new repository.
2. Name the repository 'CIS103-Lab1'.
3. Make sure to initialize the repository with a README file.

Step 5: Clone the Repository

In your terminal or command prompt, navigate to the directory where you want to store your project. Then, clone the repository to your local machine by running:

```sh
git clone https://github.com/yourusername/CIS103-Lab1.git
```

Step 6: Add Your Python Script

After completing your Python script in Part 2 of this lab, save it in the cloned repository folder on your computer.

Step 7: Commit and Push Your Changes

1. Use the following commands to stage, commit, and push your changes to GitHub:
```sh
git add lab1.py
git commit -m 'Initial commit of lab1.py'
git push origin main
```
2. Verify that your script is now visible in your GitHub repository online.

Part 2: Writing Your First Python Script

1. Create a New Python File:
- Open your IDE and create a new Python file named 'lab1.py'.

2. Write a Simple Python Script:
- Write a script that prints a greeting message to the user. The script should include a header comment block and comments explaining each part of the code.

```python
# ---------------------------------------------------------
# Lab 1: Getting Started with Python
# CIS 103: Introduction to Programming
# Instructor: [Your Name]
# Student Name: [Your Name]
# Date: [Today's Date]
# Description:
# This script prints a greeting message to the user.
# -------------------------------------------------------

# The following line of code prints a greeting message
print("Hello, welcome to CIS 103!")
```

3. Run Your Script:
- Run the script from within your IDE or using the terminal/command prompt by navigating to the script's directory and typing `python lab1.py` (or `python3 lab1.py` on some systems).
- Verify that the message is printed to the screen.

Part 3: Exploring Python Variables and Data Types

1. Modify Your Script:
- Extend your script to include variables and demonstrate basic data types (string, integer, and float). The script should ask the user for their name and age, then print a message using this information.

```python
# -------------------------------------------------------
# Lab 1: Getting Started with Python
# CIS 103: Introduction to Programming
# Instructor: [Your Name]
# Student Name: [Your Name]
# Date: [Today's Date]
# Description:
# This script prints a personalized greeting message
# and demonstrates the use of variables and basic data types.
# -------------------------------------------------------

# Get the user's name (string) and age (integer)
name = input("Enter your name: ")
age = int(input("Enter your age: "))

# Calculate the year the user was born
current_year = 2024
birth_year = current_year - age

# Print a personalized greeting message
print(f"Hello, {name}! You were born in {birth_year}.")
```

2. Run and Test:
- Run the script again and test it with different inputs to ensure it works correctly.

Part 4: Submission

1. Review Your Code:
- Make sure your code is well-commented and adheres to the Python syntax and style guidelines.
- Ensure that your header comment block is complete and accurate.

2. Submit Your Work:
- Save your Python file ('lab1-[your name].py').
- Push your code to GitHub following the steps in Part 1.
- Submit the link to your GitHub repository through Brightspace.

Grading Criteria:

- **Correctness: The script runs without errors and produces the correct output.**
- **Commenting: Code is well-commented, and the header comment block is complete.**
- **Clarity: Variable names are meaningful, and the code is easy to read and understand.**
- **Completion: All parts of the lab are completed and submitted on time.**