Coursework Assignment Information

This is not group work and must be completed individually

This coursework assignment information document aims to guide you in the completion of the assignment and should be used as a reference in your in-class demonstration. Throughout this assignment you will be required to interact with two servers in order to retrieve your assignment archives. These archives are unique to each student taking part in the assignment, therefore it is important that you do not share your assignment archives or use incorrect identification details when interacting with the assignment servers. If you use identification details which are incorrect when interacting with the assignment servers, your results will be marked as incorrect.

To begin the assignment you will need two things:

1. The **PoP_Coursework_Assignment.zip** archive located on Brightspace, which contains:
    a. The assignment log form (**Assignment Log Form.docx**). You must complete all fields of the assignment log with information and data relevant to your unique assignment materials.
    b. The assignment information document (**Assignment Information.pdf**). You are currently reading this document.
    c. The client program (**client.py**). Once configured, this program will be used to interact with the assignment servers.
2. The assignment server **connection details** for **server 1** and **server 2** and the **initial password**. This information will be released to you within your scheduled Principles of Programming lectures. You must attend these lectures – it is possible that these details may change over time.

To complete the assignment you will need to submit the following deliverables:

- The **<studentID>_PoP_Assignment.zip** archive. This must be uploaded to Brightspace and contain all deliverables to be assessed. This will contain:
    - The **<studentID>_Assignment_Server1** folder. This will contain:
        - The unique files retrieved from server 1, and the Python source code you have written to complete the task.
    - The **<studentID>_Assignment_Server2** folder. This will contain:
        - The unique files retrieved from server 2, and the Python source code you have written to complete the task.
    - The completed assignment log form (**Assignment Log Form.docx**).
    - An in-class demonstration. You will be required to demonstrate your software in-class during one of your scheduled lab sessions. You will be asked about any assumptions you have made regarding the directory structure and the data you have operated on, and the steps you have taken to develop the software. You will also be required to present the source-code and answer any specific questions.

The deliverables must be in the following structure:

- <studentID>_PoP_Assignment.zip
  - <studentID>_Assignment_Server1
    - received_files
    - source_code
  - <studentID>_Assignment_Server2
    - received_files
    - source_code
  - main.py (entry point of your program)
  - Assignment Log Form.docx (including the steps/flow-chart you took to complete the assignment)

The structure above must be adhered to. **This will be assessed**.

SERVER 1

You will need to retrieve your unique assignment materials from **server 1**. To do so, configure the **client.py** program with:

- Your studentID
- The server host
- The server port

You will not be able to communicate with this server from your scheduled Principles of Programming lab room. You will need to find an alternative approach to connecting – it must be on campus.

Execute the script to receive your unique assignment materials, in the form of a password protected compressed archive (**received_files.zip**) containing:

- A compressed archive named **documents.zip**.
- A password protected compressed archive named **secret.zip**.

You will be required to enter the **initial password** to successfully extract this archive and gain access to the files within.

The password required to access **secret.zip** is hidden somewhere within **documents.zip**. To find this password, you will need to extract **documents.zip** and write a Python program to search for the password. Conveniently, the person who has hidden this password cannot resist the temptation of leaving their signature. The signature is the text string "Millenium2000". You can use this information to write a program (name it **FindPassword.py**) and locate the file containing the password. However, the file contents within **documents.zip** have been encoded as hexadecimal, so they will need to be decoded first. Once you have resolved the password, use it to extract **secret.zip**.

secret.zip contains two files: **data.txt** and **message.txt**. You must apply a method of data visualisation to produce a histogram of the data within **data.txt**. Your histogram must be presented such that it is clear which data value is the **mode** value, this means each bar must represent a single value rather than a range.

The first 8 characters of the text within **message.txt** form the password that is required to gain access to the password protected compressed archive, which is to be retrieved from **server 2**.

SERVER 2

You will need to retrieve your unique assignment materials from **server 2**. To do so, configure the **client.py** program with:

- Your studentID
- The server host
- The server port

You will not be able to communicate with this server from your scheduled Principles of Programming lab room. You will need to find an alternative approach to connecting – it must be on campus.

Execute the script to receive your unique assignment materials, in the form of a password protected compressed archive (**received_files.zip**) containing:

- A text file named **decode_me.txt** containing an encoded message.
- A password protected compressed archive named **decoder.zip**.

You will be required to enter the first 8 characters of the text within **message.txt** retrieved from **– server 1** in order to gain access to extract **received_files.zip**. To gain access to the decoder program (**decoder.py**), you will need to extract **decoder.zip**. The password to **decoder.zip** is the **mode** value within **data.txt** from **server 1**

You must then write a program which uses the appropriate function in **decoder.py** to decode the contents of **decode_me**.txt.