

Coursework 1 - Practical Skills Assessment (40% of the module mark)

Scheduled : Week 8, Friday 21st March during timetabled labs - duration 90 minutes.

Feedback : Within 20 working days of the scheduled assessment date.

[Moderation Note] This specification document will be released in advance of the assessment taking place. Access to the partial solution code, needed to undertake the assessment, will be withheld until the time of the assessment.

This is an **individual** assignment.

When submitting your assignment you are agreeing to the following statement:

I declare that this is all my own work and does not contain unreferenced material copied from any other source. I have read the University's policy on plagiarism and understand the definition of plagiarism. If it is shown that material has been plagiarised, or I have otherwise attempted to obtain an unfair advantage for myself or others, I understand that I may face sanctions in accordance with the policies and procedures of the University. A mark of zero may be awarded and the reason for that mark will be recorded on my file.

The University policy on **plagiarism** is available at

<https://www.ulster.ac.uk/student/exams/cheating-and-plagiarism>

Assessment criteria

- Evidence of program comprehension (25%)
- Appropriate use of relevant libraries methods (25%)
- Code structure, readability, and file organisation (25%)
- Program execution correctness and completeness (25%)

Credit will be given for solutions written using Python constructs and techniques from the module notes and materials. Exceptionally, any other source must be fully referenced and the reason for its use explained as a comment in the code.

COM397 – Automation with Python

Overview

This assessment builds upon the tasks you have been exposed to in practical labs. It requires you to write several Python scripts using VS Code. Specifically, you must code automation processes involving the identification, retrieval and merging of PDF files, hosted remotely. To support your solution, you will be provided with access to a partially coded solution comprising approx. 200 lines of comments and code.

You have **90 minutes** to complete this assessment, which will be conducted within a timetabled lab setting. Within this timeframe, you will be expected to complete the following steps.

1. Access, unzip and open the partial code solution within a VS Code environment
 2. Install necessary libraries to your virtual environment from PIP (Package Installer for Python)
 3. Comprehend the specification and partial code solution provided
 4. Complete the tasks outlined in this specification
 5. Complete, separately, a task log sheet
 6. Submit your completed code, task log sheet and associated files to Blackboard learn.
- Note: if you do not complete one or more tasks within the available time, you should still proceed with a full submission of code, logs and files

This assessment is open book. You are permitted to refer to your own notes, lecture materials hosted on Blackboard Learn, and previous programs you've written in practical labs. You are also permitted to use VS Code. The use of search engines or other online resources beyond those outlined above is prohibited.

Program Requirements

Download the **zipped template code** located on Blackboard Learn, under the CW1 Assessment Folder. Unzip (Right-Click -> Extract All) the folder to reveal **three .py files** and **three sub-folders** (task1, task2 and task3). These resources provide a partial implementation and comments for each required task (described below). Each file includes several **TODO** instructions that you must implement to complete the tasks - each preceded with three #, for example:

```
### TODO-TASK1-X
```

```
### Write a descriptive comment for this function
```

What follows is a description and outline of the requirements for each task. Each task is independent so you may choose to move onto the next task if you experience challenges with any one task.

Task 1 (cw1_task1.py)

The objective of Task 1 is to programmatically retrieve a list of file names from a web page (webpage url *provided in cw1_task1.py template file*). Figure 1 presents a screenshot of the web page. Each file name is hyperlinked within the source HTML to a file location on the webserver.



Figure 1. Screenshot of the webpage *index.html*

`cw1_task1.py` should be fully implemented to retrieve the hypertext reference (`href`) for each file name, which are then stored to a list and written out to a local text file called `task1_pdf_link.txt`, as shown in Figure 2.

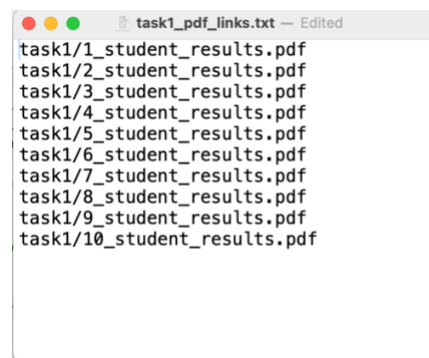


Figure 2. Screenshot of the text file holding the retrieved file names.

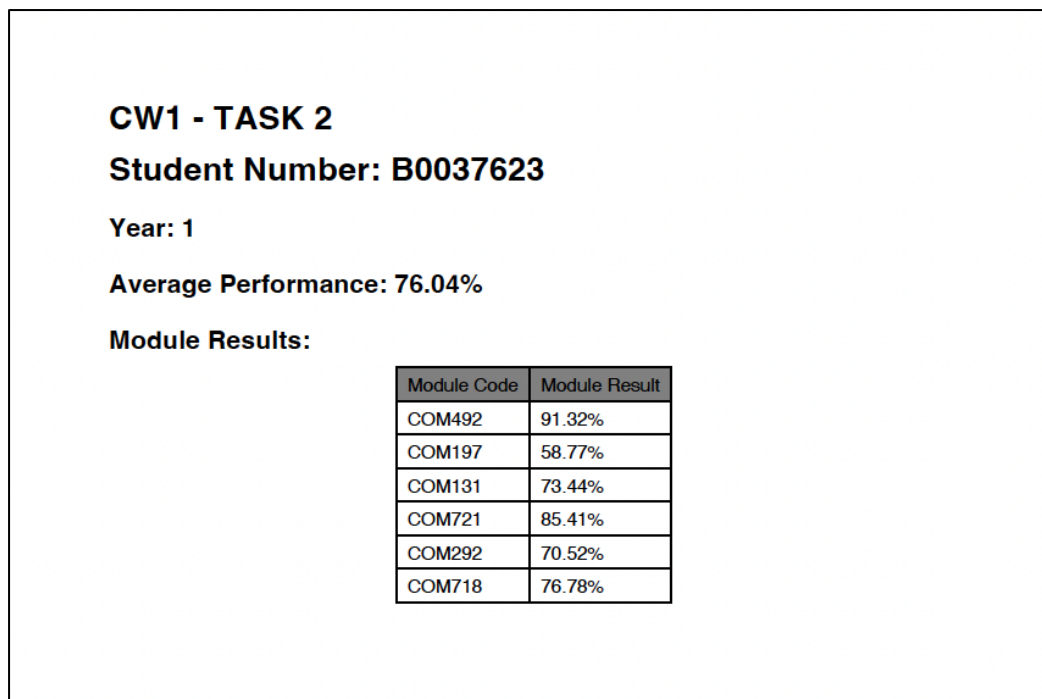
You should observe that `cw1_task1.py` does not initially execute any functional code. To address this, you may need to complete and extend the template file by implementing the instructions set out below each **TODO** statement.

*Hint: you may need to install and / or import **bs4** and **requests** libraries to support this task.*

Task 2 (cw1_task2.py)

Task 2 reads a new set of file references from the file **task2_pdf_link.txt**, which is located in sub-folder **task2**. The file references partial path information for several PDF files stored on a webserver (*base url provided in cw1_task2.py template file*)

Fully implement cw1_task2.py program so that it retrieves each file listed in the text from the web server and store these alongside the text file in the **task2 sub-folder**. Each file represents a dummy student record, similar to that which is presented in Figure 3.



CW1 - TASK 2

Student Number: B0037623

Year: 1

Average Performance: 76.04%

Module Results:

Module Code	Module Result
COM492	91.32%
COM197	58.77%
COM131	73.44%
COM721	85.41%
COM292	70.52%
COM718	76.78%

Figure 3. Preview of one of the PDF files hosted on the webserver

You should observe that cw1_task2.py does not initially execute. To address this, you need to complete and extend the template file by implementing the instructions set out below each **TODO** statement.

*Hint: you may need to verify the successful install and / or import of both **requests** and **pathlib** to support this task.*

Task 3 (cw1_task3.py)

The objective of Task 3 is to programmatically merge several files located in the task3 subfolder to a single PDF file. This combined PDF should be written to the task3_subfolder as **combined_output.pdf**.

You will observe that cw1_task3.py does not initially execute. To address this, you should complete the extend the template file by implementing the instructions set out below each **TODO** statement.

*Hint: you may need to install and / or import **pypdf** and **pathlib** libraries to support this task.*

Submission Procedure

When you have finished your assessment, you will need to upload it to Blackboard. To do this, you should use Windows Explorer to zip the main working folder (containing relevant sub-folders and files) used in your project. Rename the zip file to **COM397_CW1_B00XXXX.zip**, replacing XXXX with you unique Student ID. The zip folder should aim to include:

COMPLETED PROGRAM FILES

1. cw1_task1.py
2. cw1_task2.py
3. cw1_task3.py

SUBFOLDERS CONTAINING

4. Task1 -> task1_pdf_links.txt
5. Task2 -> task2_pdf_links PLUS all 15 downloaded PDF files
6. Task3 -> 5 sample PDF files PLUS combined_output.pdf

You **must retain your electronic submission receipt.**

If you have any difficulties with the upload to Blackboard then **email your zipped files** to mp.donnelly@ulster.ac.uk when complete.

COM397 APPLICATION DEVELOPMENT FOR IS

ASSESSMENT ONE (40%)

Criteria (100%)	0-29% low fail	30-39% fail	40-49% 3rd	50-59% 2.2	60-69% 2.1	70-79% 1st	80-100% High 1 st
Evidence of Program Comprehension (25%)	No understanding evidenced.	Erroneous or very limited code or comments to extend the template solution.	Some Task elements partially implemented or pseudocode with comments.	Several Task elements correct with some comments or pseudocode to supporting missing code.	Tasks mostly completed with appropriate comments or pseudocode for remaining functions. Basic implementation.	Strong evidence of understanding – tasks completed, and comments provided with at most minor modification to template.	Outstanding evidence of understanding - all tasks completed in line with task specification and templates. Descriptive and accurate comments.
Appropriate use of relevant libraries (25%)	None or minimal solution code provided.	Solution includes library keywords and other constructs but not organised to address tasks.	Solution includes some library keywords to perform at least part of one basic automation task.	Solution includes library keywords and arguments to correctly perform at least one automation task.	Solution includes library keywords, arguments, and iteration constructs to meet most of the specification.	Solution includes comprehensive set of library keywords, arguments, and full control constructs to meet specification.	Solution includes complete set of library keywords, arguments, full control constructs with optimised coding to fully meet specification.
Structure and readability of submission. (25%)	None or minimal solution code provided.	Disorganised or poorly edited elements of template code. Poor Layout.	Partial Task submission with inconsistently structured code. Disorganised file storage / submission.	All Task files submitted. Limited comments, but structured code.	All Task files submitted. Coded tasks largely structured with appropriate file storage.	All Task files submitted. Well-structured Task code / comments and file storage.	All Task files submitted. Well-structured code, enhanced detailed comments and accurate file storage.
Program execution correctness and completeness. (25%)	Incomplete submission.	Submitted files will not execute.	Submitted code runs but unstable following most function calls.	Submitted code runs and performs at least one basic function in more than more tasks without error.	Submitted code runs and meets most of the specification for at least two of the tasks.	Submitted code runs and meets the specification for all three tasks with at most minor issue / alteration to the task templates.	Submitted code runs with no modification to templated. Outputs fully meet the specification.