**Task**

You have been tasked to show evidence of your learning <<some sequence of learning>> as an annotated portfolio.

**You must produce evidence of the following:**

* Dot point list of deliverables

**Evidence**

## What is a portfolio

You are probably familiar with the concept of a portfolio. A portfolio is a collection of your work that represents evidence of your learning over a sequence of work. At its most broad, your evidence will consist of you showing what we learned in class and examples of how you implemented this learning in various tasks/activities.

## What do is meant by an “annotated” portfolio?

So, what do we mean by an annotated portfolio? If a portfolio is a collection of your tasks, your annotation is you highlighting to us what your learning is, how advanced it is, how you’ve researched to go beyond the classroom activities, or how you’ve applied your knowledge to solve a novel or new way.

## Example

This example is not intended to be a completed annotation. Instead, the intention is to stimulate your brain with your thoughts about what you could submit.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Analysis, Synthesis & Evaluation** |  | | **SUBTOTAL** | | **A\_\_/20**  **T\_\_/ 18** |
| Variables and printing | You have produced evidence of your learning on how variables and printing works. This evidence must include both examples and annotations of your work.  A starting point in your evidence is submitting examples of your work given in weekly learning briefs.  To expand your potential, you may submit evidence of your completion of more complex tasks with these techniques and annotation of a deep understanding of the concepts in this section. | 4 | \_\_/4 | \_\_/4 | Ax1  Tx2 | A \_\_ / 4  T \_\_ / 8 |

|  |
| --- |
| Printing with variables Early in week 1, we were taught to print in python.  To print in python, we can use the print function. The print function takes string variables.  print("ada lovelace")  The print function can take multiple strings. In this example, I show how this can be incorporated.  print("ada", "lovelace")  One side effect of this system is that people tend to use it to concoct strings together. However, it doesn’t join the string together; it’s a side effect of how the print function works.  This side effect might be considered an anti-pattern because it’s hard to see why it is happening.  In this example, we use the + symbol to join the given\_name and family\_name into a full name. This way of joining two strings together is called concatenation.  given\_name = 'ada' family\_name = 'lovelace' print(given\_name, family\_name) full\_name = given\_name + " " + family\_name print(full\_name)  Concatenation is fine, but it is a bit wonky. Examples with more variables or text between variable insertion can overwhelm the user.  A common alternative to string concatenation is f-strings or formatted strings. In this example, we inject variables straight into the string.  given\_name = 'ada' family\_name = 'lovelace' full\_name = f"{given\_name} {family\_name}" print(full\_name)  f-strings are useful for putting together bespoke strings based on variables you may or may not know you will need before execution. In the example below, I construct an f-string to open up a file and dynamically insert the filename and the file type.  def example(file\_name: str, file\_type: str) -> str:  with open(f'{file\_name}.{file\_type}') as fd:  return fd.readlines()[0].strip() |