Question 4) Explain how you apply the self-documenting principle and encapsulation principle in the above code for Questions 1-3.

I implemented the encapsulation principle in questions 1-3 by moving much of the application logic to a dedicated class and method which would perform those actions. I used the self documenting principle by making sure that the classes and methods created are able to convey the idea about what kind of task is being executed and what kind of data is being returned.

Question 5) Explain the following:

1. What's the benefit of applying the self-documenting principle and encapsulation principle in the above code for Questions 1-3?

   By encapsulating the logic and actions into these separate classes, the code becomes more modular, useable, and pluggable and if I was required to implement the same functionality in another project, I need not recreate the logic again. By adding the self-documenting principle, I will be able to convey the intent and the processes within the code without having to elaborate on the code itself; making it easier for others to use this code. Therefore the encapsulation and self documenting principles also allows me to not worry about how the logic is implemented but know what it does and that it works.

2. What's the drawback if you do not apply those principles?

   By not applying these principles, the logic contained in your application will be less modular, you will end up writing and repeating the same or similar code again, and other programmers may not be able to infer what the class or method does.