**What is a set and why would you, as a programmer need to use one. Provide an example.**

A set is an unordered collection of items with no duplicates. Sets can contain numbers, strings, tuples, and Boolean values but cannot contain lists or other sets. A set would be useful when a membership test is needed on a collection of unique strings or integers for a program. An example would be creating a set of the names of discovered planets. If a new planet is discovered, a person could enter the new name for this planet to determine if there’s already a planet with that name.

**Describe how the split method is used to access the fields of a CSV-formatted file. Provide an example.**

The split method access the fields of a CSV file by splitting a line into multiple pieces, using a comma as the point at which to split the string. These “pieces” can be assigned to variables, or saved in a list. Standard list operations can then be performed on this list to extract more information. This could be used—as in my example below—to take the garbled mess that results from a waypoint export from my GPS and display it in a more user-friendly manner.

**Describe one of the ways (from Chapter 5) that you might use a dictionary in the processing of data.**

Dictionaries are pairs of items, stored as a key:value pair. This makes them handy for finding data when the key is known. Example 3 in the book on page 224-225 demonstrates a good use of a dictionary. In this example, an if/elif statement is replaced with a one line, easy to read dictionary to determine the admission fee for various age groups. If the user were to enter ‘senior’ as the input, the program would have to check if senior == child, then minor, then adult, and finally arrive at senior to return $8. Using a dictionary, the entry ‘senior’ just returns the $8 when invoked.