CS128 – Spring 2022. Lab 12 – Lists and Strings.

NOTE: random.randint(n,m) generates a random x such that n <= x <= m.

Task1a – The starter code creates 4 lists. Update the code to generate a random sentence of the form "adjective noun verb adverb" by picking values in the same position from each of the 4 lists. For example, adorable puppy runs crazily (1st element from each list), old car steers crazily (2nd element from each list), or silly rabbit jumps foolishly (3rd element from each list). Run the cell enough times to see the 3 possible sentences generated at least once.

Task1b – Copy the code from task1a. Instead of 4 separate lists, create a single nested list named **words** that contains the 4 sublists/rows. The first row should contain the adjectives, the second row the nouns, the third row the verbs, and finally the adverbs. Generate a random sentence using indexing into the nested list, again producing a sentence of the form "adjective noun verb adverb". Run the cell to see the 3 possible sentences generated at least once.

Task2 – Prompt the user to enter a comma-separated sequence of words of the form "noun,verb,adjective,adverb". Use the split function to separate the substrings into an array based on the comma as delimeter. Produce a sentence of the form "The adjective noun verb adverb". If fewer than 4 words are entered, provide an error message as shown.

```
enter noun,verb,adjective,adverb:table,smiles,big,seriously
The big table smiles seriously
```

```
enter noun,verb,adjective,adverb:dog,barks
2 words entered, need 2 more to generate sentence.
```

```
enter noun,verb,adjective,adverb:cat,meows,pretty
3 words entered, need 1 more to generate sentence.
```

```
enter noun,verb,adjective,adverb:pizza
1 words entered, need 3 more to generate sentence.
```

Task3– Duplicate the functionality of task2, but you are not allowed to use the string split function. You will need to write code to look for the commas and extract the substrings between commas. HINT: use the count and find functions. Test your code with the same set of input strings as task2.

Task4 – The existing code creates two lists for pizza sizes and toppings. Use a **nested loop** to generate the 12 combinations of size and topping:

```
small mushroom pizza
small pepperoni pizza
small olives pizza
small sausage pizza
medium mushroom pizza
medium pepperoni pizza
medium olives pizza
medium sausage pizza
large mushroom pizza
large pepperoni pizza
large olives pizza
large sausage pizza
```

Task5 – The existing code creates a list of names. Use a nested loop to generate hyphenated names that combine values from the list. Do not hyphenate a name with itself (so no Jones-Jones). Do not create a second list.

```
Jones-Chen
Jones-Smith
Jones-Miller
Chen-Jones
Chen-Smith
Chen-Miller
Smith-Jones
Smith-Chen
Smith-Miller
Miller-Jones
Miller-Chen
Miller-Smith
```