Lab 6 – The Sequence Data Structure

Your task is to write a program that *creates a list of elements **common to a list of lists***.

Short story:

1. Prompt user for *number of lists (5-10) and number of elements in each list (15-20)*
   Remember to *coerce the user input to an integer!*
   We'll skip the checking – we didn't cover exception handling yet

2. Create a data structure – *buncha_lists* - that has [*number of lists*] lists, each list has *number of elements in each list* of random integers between 0 and 20

3. Print out the generated data structure:
   *One list per line; **sort each list*** (makes it easier to see if program works!)

4. Write a function that looks for **common elements for two lists.**

5. Write a function that calls the function you wrote in step 4, *passing two lists in your buncha_lists structure* at a time
   The idea is that you get the common elements from *buncha_lists* [0] and [1] from the function in step 4, pass this list of common elements and *buncha_lists[2]* to the function in step 4 **and get a list of common elements to three lists**, pass the new common list and *buncha_lists[3]* to that step 4 function, getting a list common to **four lists**. Continue until you pass **all lists in** *buncha_lists* to the step 4 function.

The program with a few comments/blank lines for readability is **about 40 lines**

Here is some starter code (**skel.py**) on the next page and in your lab folder – shows the *function definitions* (since we didn't yet drill down to function usage yet!) and lots of hints.

```python
#!/usr/bin/env python3

from random import randint           # To generate random integers
# Find common elements for a pair of iterators
def find_common_unique_elements( iter1, iter2 ):
    list_with_common = []
    # Do something here to fill up the list
    return list_with_common
# This function calls the above for two lists
# We check if we have at least two lists
def find_common_elems_buncha_iters( buncha_iters ):
    # Need to know HOW MANY LISTS we have
    # Pass first two iterables; save common elements
    # How do we access the first two elements of the buncha_iters structure?
    common_elems = find_common_unique_elements( ??? )
    # Need an expression that ITERATES OVER REMAINING LISTS IN buncha_iters
    # In the block coded above, call find_common_unique_elements with arguments
    # common_elems and the NEXT LIST in the remaining lists in buncha_iters
        ??? = find_common_unique_elements( common_elems, ??? )
    # When done, this has ALL the common elements
    return common_elems
# Generate and return a random list of num_items items
# Each integer between 0 and 15
def gen_return_random_num_list( num_items ) :
    #return a LIST COMPREHENSION that creates a list of
    # num_items random integers between 0 and 20
    # The expression        randint(0, 20)      generates a random #
    return [ ????? ]
def main( ):
    # Prompt user for num_iterators and num_each_iterator
    # Remember to coerce user input to an integer!
    num_iterators = ???
    num_each_iterator = ???
    # Create buncha_iterator structure
    # Use a LIST COMPREHENSION to create a LIST OF LISTS (buncha_iters)
    # Each list (element) in buncha_iters is a LIST OF RANDOM INTEGERS
    #
    # You'll need to call gen_return_random_num_list, passing num_each_iterator
    # for num_iterators times
    buncha_iters = [ ????? ]
    # Print the buncha_iters generated
    # One list per line - iterate over buuncha_items, print an element
    # Sort the 'per line' list - makes it easier to read
    # OK - let's print out the common elements - this one's a gimme
    print(f'Elements common to all lists:
{find_common_elems_buncha_iters( buncha_iters )}')
main()
```

Sample Execution

```
Enter number of iterators (5-10) ==> 5
Enter number for each iterator (15-20) ==>20
[3, 3, 4, 7, 7, 9, 10, 10, 10, 10, 10, 10, 11, 12, 12, 13, 13, 14, 14, 19]
[1, 2, 2, 4, 4, 5, 6, 7, 7, 8, 10, 10, 12, 12, 12, 14, 17, 18, 18, 19]
[0, 0, 0, 2, 2, 4, 4, 8, 9, 11, 12, 12, 13, 14, 15, 15, 16, 18, 18, 20]
[0, 2, 4, 6, 7, 8, 9, 12, 13, 14, 14, 14, 15, 16, 16, 17, 18, 19, 19, 19]
[1, 2, 2, 5, 5, 5, 6, 6, 6, 9, 11, 12, 14, 15, 16, 17, 18, 19, 19, 20]
Elements common to all lists: [14, 12]
```