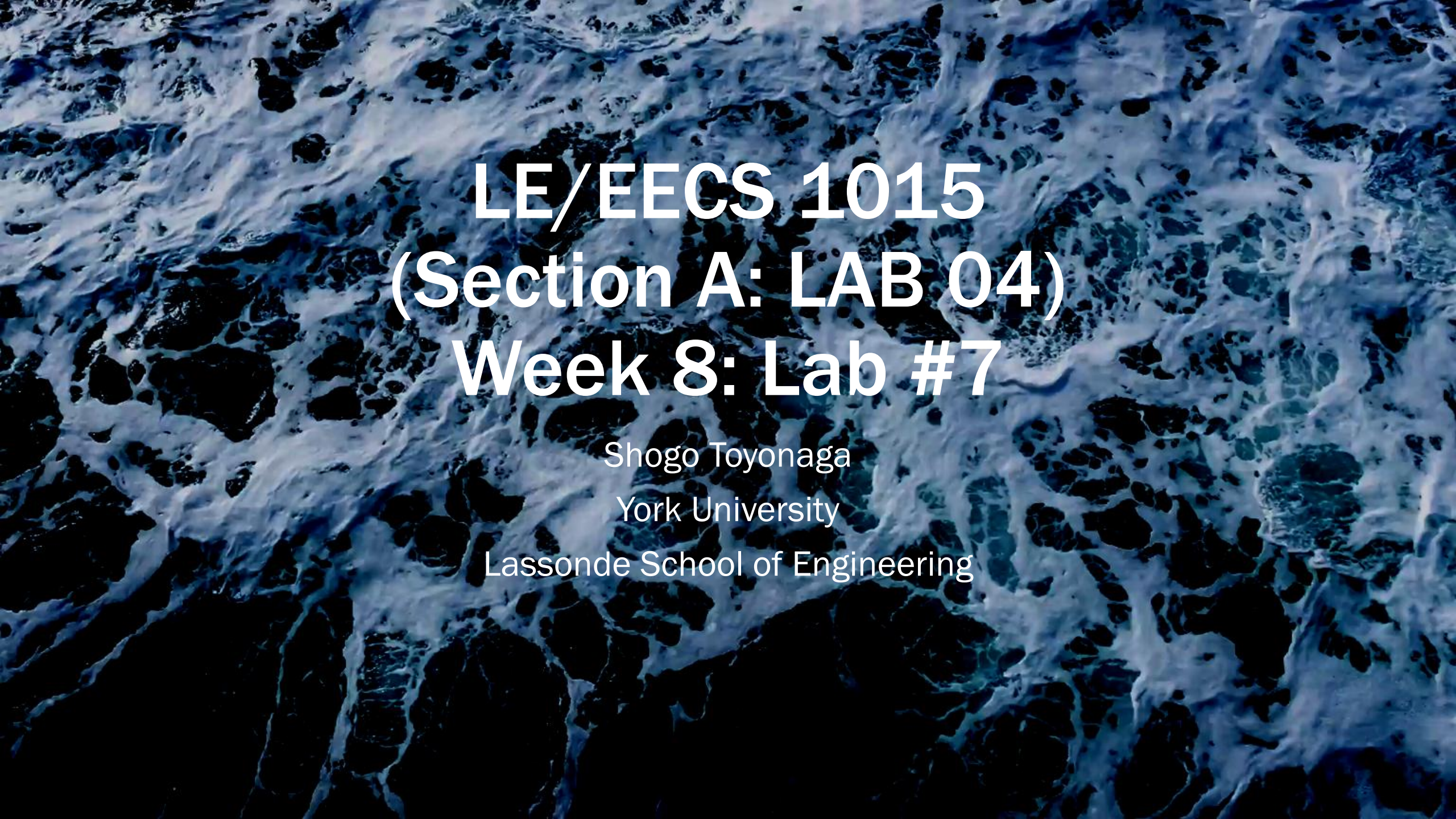

# LE/EECS 1015 (Section A: LAB 04) Week 8: Lab #7

Shogo Toyonaga

York University

Lassonde School of Engineering

# Goals of Lab 7

1. Practice with Python Collections
2. Debugging
3. Refactoring
4. Software Design

# Concept Review

1. Loops
    - For
    - While
    - Keywords: continue, break
2. Collections
    - Tuples
    - Lists
    - Sets
    - Dictionaries

# Loops (Review)

- A while loop will execute statements while its loop-condition remains true. It gives us some more flexibility.
- A for-loop will execute statements for a specified number of iterations or objects. It is very clear and well-defined.
- $\boldsymbol{break}$ and $\boldsymbol{continue}$ offer ways to forcibly exit a loop or an iteration.

# Loops (Review)

1. Every for-loop has an equivalent while-loop
2. Every while-loop does not have an equivalent for-loop

TLDR:

- Use the best tool for the job;
  - You should be able to justify why you used a certain paradigm in your algorithm

# Tuples

- Represents an ordered and immutable collection of data
- Generally instantiated with round brackets or the $\mathrm{tuple}()$ constructor
- Supports unpacking for quick assignment of multiple variables
- Tuple-Specific Methods:
  - $count(self, value, /)$: Returns the number of occurrences of value
  - $index(self, value, start, stop, /)$: Returns the first index of value and throws a ValueError if it not present
- Run $help(tuple)$ in the interactive terminal for documentation!

# Lists

- Represents an ordered and mutable collection of data
- Instantiated with square brackets or the $list()$ constructor
- Supports unpacking for quick assignment of multiple variables
- List-Specific Methods:
  - $append(self, object, /)$: Append object to the end of the list
  - $clear(self, /)$: Remove all items from the list
  - $copy(self, \ /)$: Returns a shallow copy of the list
  - $count(self, value, /)$: Returns the number of occurrences of value
  - $extend(self, iterable, /)$: Appends the iterable to the end of the list.
  - $index(self, value, start, stop, /)$: Returns the first index of the value. Raises a ValueError if the value is not present.
  - $insert(self, index, object, /)$: Inserts the object before index.
  - $pop(self, index = -1, /)$: Removes and returns the last item in the list (by default).
  - $remove(self, value, /)$: Removes the first occurrence of value.
  - $reverse(self, /)$: Reverses the list, in-place.
  - $sort(self, /, *, key, reverse)$: Sorts the list in-place and returns None.
- Run $list(tuple)$ in the interactive terminal for documentation!

# Sets

- Represents an unordered and mutable collection of data.
- Generally instantiated with curly brackets or the $set()$ constructor
- Supports unpacking for quick assignment of multiple variables
- Duplicates will be automatically removed; elements must be distinct.
- Set-Specific Methods:
  - $add(\dots)$: Add an element to a set if it is not present.
  - $clear(\dots)$: Remove all elements from the set.
  - $copy(\dots)$: Returns a shallow copy of a set.
  - $difference(\dots)$: Return the difference of two or more sets as a new set.
  - $difference\_update(\dots)$: Remove all elements of another set from this set.
  - $discard(\dots)$: Remove an element from a set if it is a member. It will not throw an exception when an element is missing from the set.
  - $intersection(\dots)$: Return the intersection of two sets as a new set.
  - $intersection\_update(\dots)$: Update a set with the intersection of itself and another.
  - $isdisjoint(\dots)$: Return True if two sets have a null intersection.
  - $issubset(self, other, /)$: Test whether every element in the set is in the other.
  - $issuperset(self, other, /)$: Test whether every element in other is in the set.
  - $pop(\dots)$: Remove and return an arbitrary set element. Raises KeyError if the set is empty.
  - $remove(\dots)$: Remove an element from a set; it must be a member.
  - $symmetric\_difference(\dots)$: Returns the symmetric difference of two sets as a new set.
  - $symmetric\_difference\_update(\dots)$: Updates a set with the symmetric difference of itself and another.
  - $union(\dots)$: Return the union of sets as a new set.
  - $update(\dots)$: Update a set with the union of itself and others.
- Run $help(set)$ in the interactive terminal for documentation!

# Dictionaries

- Represents an ordered and mutable collection of data which relies on unique key-value pairs.
- Generally instantiated with curly brackets with the key-value pairs  or the dict() constructor
- Supports unpacking for quick assignment of multiple variables
- Dictionary-Specific Methods:
  - $clear(\dots)$: Removes all items from the dictionary.
  - $copy(\dots)$: Creates a shallow copy of the dictionary.
  - $get(self, key, default, /)$: Return the value for the key if it exists.
  - $items(\dots)$: Returns a set-like object of the values.
  - $keys(\dots)$: Returns a set-like object of the dictionary keys.
  - $pop(\dots)$: Removes a specified key and returns the corresponding value.
  - $popitem(self, /)$: Removes and returns a $(key, value)$ pair as a tuple.
  - $setdefault(self, key, default, /)$: Insert a key with a value of default if the key is not in the dictionary.
  - $update(\dots)$: Updates the dictionary from an iterable.
  - $values(\dots)$: Returns an object providing all of the values.
- Run $help(dict)$ in the interactive terminal for documentation!

# Writing DocTest

1. The following names can be assigned to a variable or return type when writing the method signature:
    - tuple
    - list
    - set
    - dict
2. You may also specify the data type of an element in a collection when writing the method signature. However, this will require that you import modules.

For examples, please refer to: https://github.com/Shogz-Labs/EECS1015_F24_Assets/tree/main/TA%20Demos

# Lab 7 – Objectives

Task 1: Follow the Steps (Calculate Average) (/30)

Task 2: Debugging (Fruits) (/30)

Task 3: Coding (Catalogue) (/10)

Task 4: Implementation (Intersecting Chars) (/10)

Task 5: Implementation (Remove List Item) (/5)

Task 6: Implementation (Remove List Index) (/5)

Task 7: Implementation (Remove Tuple Index) (/10)

# Thank You!

Shogo Toyonaga

Lassonde School of Engineering