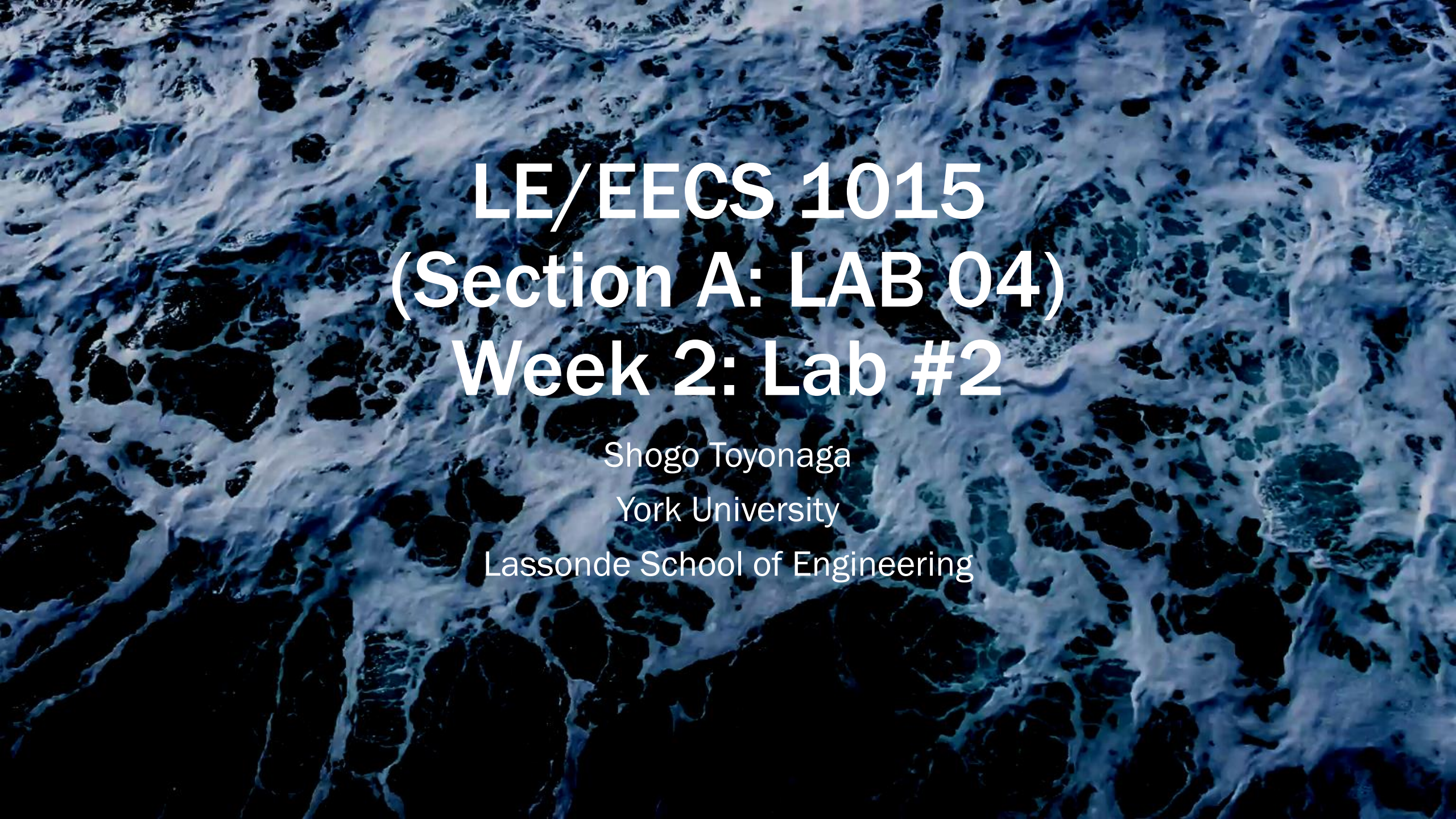

LE/EECS 1015 (Section A: LAB 04) Week 2: Lab #2

Shogo Toyonaga

York University

Lassonde School of Engineering

Goals of Lab 3

1. Practice computational thinking and planning.
2. Practicing the art of debugging
3. Writing scripts with strings (methods, slicing)
4. Emphasizing the importance of logic (LE/EECS 1019 & SC/MATH 1090)

Concept Review

1. Strings
    - Representation
        - ❖ Quotations
        - ❖ Raw-String (r)
        - ❖ Escape Characters
    - Methods
        - ❖ https://docs.python.org/3/library/stdtypes.html#text-sequence-type-str
    - Indexing & Slicing
        - ❖ $str\_var\_name[start:end:step]\ such\ that\ [start, end)$
    - Formatting
        - ❖ Concatenation (String + String)
        - ❖ Using %
        - ❖ Format Function (String_Variable.format(....))
        - ❖ Format-String (f)
2. Functions
3. Testing
    - Doctest
    - Unittest

Functions

- Reduces the amount of time you re-write the same code.
- Supports modular design
- Increases readability

Functions (Recipe)

1. Determine method properties (parameters, return type)
2. Consider the contract $(pre \rightarrow post)$:
   - Pre-Condition: The function expects certain conditions to be met…
     - $assert\ boolean\_condition\ "error\ message\ for\ AssertionError"$
   - Post-Condition: The function guarantees a certain output…
3. Write Test-Cases
4. Implement
5. Documentation
6. Debugging

Functions

Task: Write a method, $generate\_triangle(height: int)$, that returns a right-angle triangle made of asterisks if and only if $height \geq 0$.

Example: $generate\_triangle(5)$

```
*
**
***
****
*****
```

Functions

Task: Write a method, $generate\_triangle(height\colon int)$, that returns a right-angle triangle made of asterisks if and only if $height \geq 0$.

```python
def generate_triangle(height: int) -> str:
    assert height >= 0, "Negative height is not allowed."
    triangle = '\n'.join('*' * (i + 1) for i in range(height))
    return triangle
```

Testing (doctest, unit testing)

- There are many guidelines to writing tests which have been validated in the field of software engineering (LE/EECS 3311)
  - Big Bang (Post-Hoc Analysis)
  - Incremental Testing (Iterative)
  - White / Black Box Testing
  - Alpha / Beta Testing
- It is expected that you write test-cases that cover:
  - Expected Condition (Precondition Satisfied)
  - Edge Conditions
  - Exceptions (Precondition Unsatisfied)

Functions

Task: Write a method, $generate\_triangle(height\colon int)$, that returns a right-angle triangle made of asterisks if and only if $height \geq 0$.

```python
def generate_triangle(height: int) -> str:
    assert height >= 0, "Negative height is not allowed."
    triangle = '\n'.join('*' * (i + 1) for i in range(height))
    return triangle
```

Functions

Task: Write a method, $generate\_triangle(height\colon int)$, that returns a right-angle triangle made of asterisks if and only if $height \geq 0$.

```python
import doctest

def generate_triangle(height: int) -> str:
    r"""
    >>> generate_triangle(-1)
    Traceback (most recent call last):
    AssertionError: Negative height is not allowed.
    >>> generate_triangle(3)
    '*\n**\n***'
    >>> generate_triangle(5)
    '*\n**\n***\n****\n*****'
    """
    assert height >= 0, "Negative height is not allowed."
    triangle = '\n'.join('*' * (i + 1) for i in range(height))
    return triangle

if __name__ == '__main__':
    doctest.testmod(verbose=True)

```

```
3 tests in 2 items.
3 passed and 0 failed.
Test passed.
```

Functions

Task: Write a method, $generate\_triangle(height\colon int)$, that returns a right-angle triangle made of asterisks if and only if $height \geq 0$.

```python
def generate_triangle(height: int) -> str:
    assert height >= 0, "Negative height is not allowed."
    triangle = '\n'.join('*' * (i + 1) for i in range(height))
    return triangle

class TestTriangle(unittest.TestCase):
    def test_generate_negative_height_triangle(self):
        self.assertRaises(AssertionError, generate_triangle, -1)

    def test_generate_positive_height_triangle(self):
        result = generate_triangle(5)
        self.assertEqual(result, '*\n**\n***\n****\n*****')

if __name__ == '__main__':
    unittest.main()
```

```
..
----------------------------------------------------------------------
Ran 2 tests in 0.000s

OK
```

Lab 2 – Objectives

1. Follow the Steps (/30)
2. Debugging XOR (/30)
3. Implementation: Tickets (/10)
4. Implementation: Phone Number (/10)
5. Implementation: Full Name (/4)
6. Implementation: Last Name (/8)
7. Implementation: First Name (/8)

Thank You!

Shogo Toyonaga

Lassonde School of Engineering