Week 8 Tutorial

COMP10001 – Foundations of Computing

Semester 1, 2025

Clement Chau

- Libraries
- Advanced Functions
  - Lambda, Map, Filter
- Types of errors
  - Syntax, Runtime, Logic Errors

Revision: Libraries

- A library is a **collection of code** designed to be reused in different programs
- Libraries tend to contain basic functionality for a particular specialized purpose
  - e.g. Maths, Generating Plots (Diagram) in Python
- Common ones we use in this subject:
  - **math** (for functions like **sqrt()**, **log10()**, **factorial()**, and constants like **pi**, **e**)
  - **collections** (for functions like **defaultdict()**)

sem1-2025 > week-8 > sqrt.py

```
1   print(sqrt(16)) # DOES NOT WORK
2
3   import math
4   print(math.sqrt(16)) # WORKS NOW
5
6   from math import sqrt
7   print(sqrt(16)) # WORKS NOW
8
9   from math import sqrt as square_root
10  print(square_root(16)) # WORKS NOW
11
```

Exercise 1 / 4

1. Assign the value of the square root of 2 to `var` using three different methods, with each method using one of the following ways of import.

```
import math
from math import sqrt
from math import sqrt as square_root
```

A:

(a)
```
import math
var = math.sqrt(2)
```

Here we import the entire `math` library, with any methods or constants it contains.

(b)
```
from math import sqrt
var = sqrt(2)
```

Here we only import the `sqrt` method from the `math` library.

(c)
```
from math import sqrt as square_root
var = square_root(2)
```

Here import the `sqrt` method from the `math` library and give it a "nickname" called `square_root`.

Answer:

Revision: defaultdict

defaultdict(str) = ''

↓

list = []

sem1-2025 > week-8 > dict.py > ...

```python
text = "Hello, World!"

freq = {}
for char in text:
    if char in freq:
        freq[char] += 1
    else:
        freq[char] = 1

```

Without defaultdict

sem1-2025 > week-8 > defaultdict.py > ...

```python
from collections import defaultdict

text = "Hello, World!"

freq = defaultdict(int)
for char in text:
    freq[char] += 1

```

{"H": 0}

With defaultdict

Exercise 2 / 4

2. Rewrite the following with a default dictionary

```
my_dict = {}
for i in range(10):
    if i % 3 in my_dict:
        my_dict[i % 3].append(i)
    else:
        my_dict[i % 3] = [i]
```

A:

Answer:

```
from collections import defaultdict

my_dict = defaultdict(list)
for i in range(10):
    my_dict[i % 3].append(i)
```

Revision: Lambda functions

sem1-2025 > week-8 > lambda.py > ...

```python
def last_char(s):
    return s[-1]

last_char = lambda s: s[-1]

```

Sort the list of animals according to their last character.

```python
animals = ['cat', 'dog', 'elephant', 'giraffe', 'hippo']
sorted_animals = sorted(animals, key=last_char)
sorted_animals = sorted(animals, key=lambda s: s[-1])
```

Revision: Map

```
sem1-2025 > week-8 > advanced-functions > map.py > ...
1   nums = [1, 2, 3, 4, 5]
2   # Write a program to return squared, squared numbers of the list nums.
```

```
4   # Without using map + lambda functions
5   squared = []
6   for num in nums:
7       squared.append(num ** 2)
```

len(squared) == len(nums)

function

num

```
9   # Using map + lambda functions
10  squared = list(map(lambda num: num ** 2, nums))
```

Revision: Filter

(filter obj at ...)

sem1-2025 > week-8 > advanced-functions > filter.py > ...

```python
nums = [1, 2, 3, 4, 5]
# Write a program to return a list of numbers greater than 2
```

```python
# Without using filter + lambda functions
greater_than_2 = []
for num in nums:
    if num > 2:
        greater_than_2.append(num)
```

function
T/F

list of items

```python
# Using filter + lambda functions
greater_than_2 = list(filter(lambda num: num > 2, nums))
```

Exercise 3 / 4

3. Given that `food_list = ["sushi", "pizza", "hot pot", "fish and chips", "burgers"]`, what are the outputs of the below snippets of code?

(a)
```
sorted(food_list, key=lambda x:x[-1])
```

(b)
```
list(filter(lambda x: len(x.split()) == 1, food_list))
```

(c)
```
def price(food):
    return len(food) * 2
list(map(price, food_list))
```

Answer:

(a) `sorted(food_list, key=lambda x:x[-1])`

A: `['pizza', 'sushi', 'fish and chips', 'burgers', 'hot pot']`

(b) `list(filter(lambda x: len(x.split()) == 1, food_list))`

A: `['sushi', 'pizza', 'burgers']`

(c)
```
def price(food):
    return len(food) * 2
list(map(price, food_list))
```

A: `[10, 10, 14, 28, 14]`

Revision: Types of errors

- Syntax Errors
  - if statements without the colon (:) at the end.
  - Unmatched brackets (e.g. an opening bracket without a closing bracket)
- Runtime Errors
  - IndexError
  - TypeError
  - KeyError
  - ZeroDivisionError
  - NameError
  - AttributeError
- Logic Errors
  - Simply, when your program is **not getting the expected output**.
  - **Hardest to debug**, because the compiler shows no error feedback!

Exercise 4 / 4, part (a)

4. Find three out of four errors in the following programs. For each error, specify the line number, the error type (syntax/runtime/logic) and provide the corrected line of code.

(a)
```
1  def disemvowel(text):
2      """ Returns string `text` with all vowels removed """
3      vowels = ('a', 'e', 'i', 'o', 'u')
4      answer = text[0]
5      for char in text:
6          if char.lower() is not in vowels:
7              answer = char + answer
8      print(answer)
```

Answer:

A: *Below is the line number of the error, the type of error and a replacement line of code to fix it.*

- *line 4; logic/run-time (if empty string);* `answer = ''`
- *line 6; syntax;* `if char.lower() not in vowels:`
- *line 7; logic;* `answer = answer + char` *or* `answer += char`
- *line 8; logic;* `return answer`

Exercise 4 / 4, part (b)

(b)
```
1  def big-ratio(nums, n):
2      """ Calculates and returns the ratio of numbers
3      in non-empty list `nums` which are larger than `n` """
4      n = 0
5      greater_n = 0
6      for number in nums:
7          if number > n:
8              greater_n += 1
9          total += 1
10     return greater_n / total
11
12 nums = [4, 5, 6]
13     low = 4
14 print(f"{100*big_ratio(nums, low)}% of numbers are greater than {low}")
```

Answer:

A:

- *line 1; syntax;* `def big_ratio(nums,n):`
- *line 4; logic/(run-time as well since it would cause error as* `total` *is undefinedl);* `total = 0`
- *line 9; logic; remove one level of indentation (outside* `if` *block)*
- *line 13; syntax; remove indentation*

Programming on Paper

1. In this task we will play with numbers again. For example, the number 89 has two digits - 8 at the first position and 9 at the second position. If we raise each digit of 89 to the power of its position and take a sum: $8^1 + 9^2$, the answer is 89 itself! We call the number with this property a *cool number*. Another example of a cool number would be 598 because $5^1 + 9^2 + 8^3 = 598$.

   Write a function `get_cool_number(n)` that takes a positive integer n and returns the n^{th} cool number, starting from 1. That is, the first nine cool numbers are 1 to 9, then 10 is not cool, 11 is not cool ... we don't see another cool number (the 10^{th} one) until 89, then the 11^{th} cool number is 135, and so on. Therefore, your function `get_cool_number(10)` should return 89.

Tip: Use a helper function if needed!

Answer:

A:

```
def get_cool_number(n):
    counter = 0
    current_number = 0
    while counter < n:
        current_number += 1
        if num_is_cool(current_number):
            counter += 1
    return current_number

def num_is_cool(num):
    digit_sum = 0
    for i in range(len(str(num))):
        digit = int(str(num)[i])
        digit_sum += digit ** (i + 1)
    return digit_sum == num
```

Independent Work

- **Next due dates:**
  - Your **Project 1** is **due this Friday, May 2nd, 6pm**.
    - For any questions, please go to the **First Year Centre 12pm-2pm every weekday** in Level 3, Melbourne Connect or ask in the **Ed Discussion** Forums!
    - We can only provide **very limited**, general guidance.
  - Ed Worksheets **12** and **13** is **due next Monday, May 5th, 6pm**.
- Your MST marks should be available now.
  - If you'd like to argue about your marks, please fill in the form in Canvas.

Scan here for annotated slides