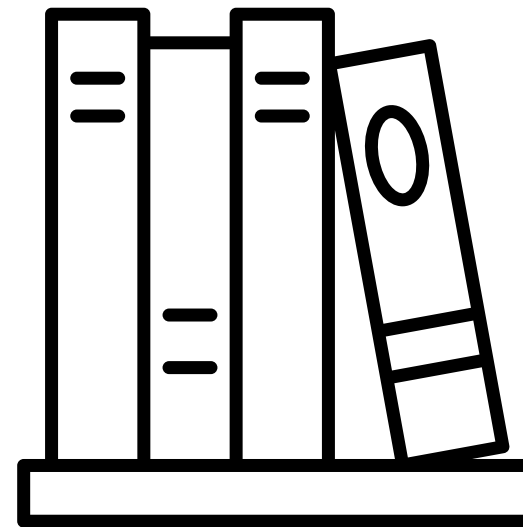COMP10001 - Sem 2 2024 - Week 8
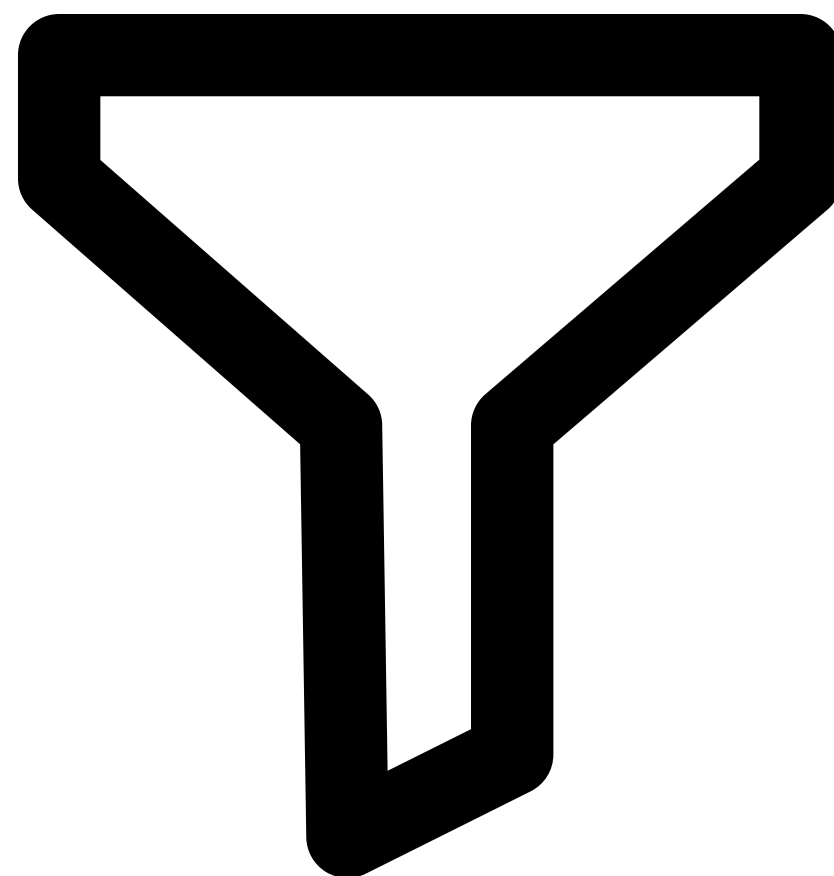
Foundations of Computing

Daksh Agrawal

Libraries

Default Dictionaries

```
text = "This is a test 123 456 789"

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

```
from collections import defaultdict

text = "This is a test 123 456 789"

freq = defaultdict(int)
for word in text.split():
    freq[word] += 1
```

```python
def square(x):
    return x * x


square_lambda = lambda x: x * x
```

```python
nums = [1, 2, 3, 4, 5]
square_lambda = lambda x: x * x
squares = map(square_lambda, nums)
squares = map(lambda x: x * x, nums)
```

```python
nums = [1, 2, 3, 4, 5]



1 usage  new *
def square(x):
    return x * x


squares = []
for i in nums:
    squares.append(square(i))
```

```python
nums = [1, 2, 3, 4, 5]

squares = map(lambda x: x ** 2, nums)
```

```python
nums = [1, 2, 3, 4, 5]



1 usage  new *
def is_even(num):
    return num % 2 == 0



even_nums = []

for num in nums:
    if is_even(num):
        even_nums.append(num)
```

```python
nums = [1, 2, 3, 4, 5]

even_nums = filter(lambda x: x % 2 == 0, nums)
```

Errors

$$5 \times \times 6$$

5 ++6

a++;

6/0

$$x^2 = (y + z)^2$$
$$x^2 = y^2 + z^2$$

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.

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

var = math.sqrt(2)
var = sqrt(2)
var = square_root(2)

Rewrite the following with a default dictionary

```python
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]
```

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

0 : 0, 3, 6, 9
1 : 1, 4, 7,
2 : 2, 5, 8

```python
food_list = ["sushi", "pizza", "hot pot", "fish and chips", "burgers"]
```

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

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

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

[10, 10, 14, 28, 14]

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

Handwritten annotations:
- Line 4: `text[0]` crossed out, replaced with `""`
- Line 6: `is` crossed out, leaving `not in`
- Line 7: `char + answer` crossed out, replaced with `answer + char`
- Line 8: `print` crossed out, replaced with `return`

Handwritten annotation above line 1: big_ratio

```python
def big_ratio(nums, n):
    """ Calculates and returns the ratio of numbers
    in non-empty list `nums`which are larger than `n`"""
    n = 0
    greater_n = 0
    for number in nums:
        if number > n:
            greater_n += 1
        total += 1
    return greater_n / total


nums = [4, 5, 6]
    low = 4
print(f"{100 * big_ratio(nums, low)}% of numbers are greater than {low}")
```

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 nth 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 10th one) until 89, then the 11th cool number is 135, and so on. Therefore, your function `get_cool_number(10)` should return 89.

Data Preprocessing

```
reviews = [
    ("5", "Absolutely love the energy of this song! It's a classic."),
❌  ("4", "Great melody and lyrics, but gets a bit repetitive after a while."),
    ("3", "Good song, but not my favorite. The chorus is catchy though."),
    ("5", "An all-time favorite! Can't stop humming along."),
    ("2", "Didn't really connect with it, but the beat is okay."),
    ("4", "Solid track with a nice rhythm. Perfect for road trips."),
    ("3", "It's decent, but I feel like it could have been more powerful."),
❌  ("5", "This song never gets old! The lyrics are so meaningful."),
    ("4", "Love the vibe! It's uplifting and fun to sing along to."),
    ("3", "It's a good song, but not something I would listen to on repeat."),
]
```