

Please grab a paper for today's tutorial,  
we will be writing lots of code!



# Week 5 Tutorial

---

COMP10001 – Foundations of Computing

Semester 1, 2025

Clement Chau

- Iterations
- Lists and Dictionaries
- Introduction to Sets

# Question 1ai, what does this output?

```
(a) i = 2
while i < 8:
    print(f"The square of {i} is {i * i}")
    i = i + 2
```

**A:** The square of 2 is 4  
The square of 4 is 16  
The square of 6 is 36

# Question 1bi, what does this output?

```
(b) for ingredient in ("corn", "pear", "chilli", "fish"):
    if ingredient.startswith('c'):
        print(ingredient, "is delicious!")
    else:
        print(ingredient, "is not!")
```

**A:** corn is delicious!  
pear is not!  
chilli is delicious!  
fish is not!

# Question 1ci, what does this output?

```
(c) i = 0
colours = ("pink", "red", "blue", "gold", "red")
while i < len(colours):
    if colours[i] == "red":
        print("Found red at index", i)
    i += 1
```

**A:** Found red at index 1  
Found red at index 4

# Question 1d, what does this output?

```
(d) input_str = "4+3+2\n3+1+1\n0+4+5+7"
for line in input_str.split("\n"):
    num_sum = 0
    for num in line.split("+"):
        num_sum += int(num)
    print(num_sum)
```

**A:** 9  
5  
16

# Question 1aii, rewrite using for loop.

```
(a) i = 2
while i < 8:
    print(f"The square of {i} is {i * i}")
    i = i + 2
```

## Answer:

A:

```
for i in range(2, 8, 2):
    print(f"The square of {i} is {i*i}")
```

# Question 1bii, rewrite using while loop

```
(b) for ingredient in ("corn", "pear", "chilli", "fish"):
    if ingredient.startswith('c'):
        print(ingredient, "is delicious!")
    else:
        print(ingredient, "is not!")
```

## Answer:

A:

```
ingredients = ("corn", "pear", "chilli", "fish")
i = 0
while i < len(ingredients):
    ingredient = ingredients[i]
    if ingredient.startswith('c'):
        print(ingredient, "is delicious!")
    else:
        print(ingredient, "is not!")
    i += 1
```

# Question 1cii, rewrite using for loop

```
(c) i = 0
colours = ("pink", "red", "blue", "gold", "red")
while i < len(colours):
    if colours[i] == "red":
        print("Found red at index", i)
    i += 1
```

## Answer:

**A:**

```
colours = ("pink", "red", "blue", "gold", "red")
for i in range(len(colours)):
    if colours[i] == "red":
        print("Found red at index", i)
```



## Question 2

2. Consider the following `while` loop and two conversions to `for` loops. Are the two `for` loops equivalent? Why might you choose one over the other?

```
count = 0
items = ('eggs', 'spam', 'more eggs')
while count < len(items):
    print(f"need to buy {items[count]}")
    count += 1
```

```
items = ('eggs', 'spam', 'more eggs')
for count in range(len(items)):
    print(f"need to buy {items[count]}")
```

```
items = ('eggs', 'spam', 'more eggs')
for item in items:
    print(f"need to buy {item}")
```

# Revision: Dictionaries

```
subjects = {  
    "COMP10001": "Foundations of Computing",  
    "MAST10006": "Calculus 2",  
    "MAST10007": "Linear Algebra",  
    "SCIE10005": "Today's Science, Tomorrow's World",  
}
```

```
print(subjects["COMP10001"])
```



```
subjects["SCIE10005"] = "TSTW"
```



# Revision: Dictionaries

```
subjects = {  
    "COMP10001": "Foundations of Computing",  
    "MAST10006": "Calculus 2",  
    "MAST10007": "Linear Algebra",  
    "SCIE10000": "Today's Science, Tomorrow's World",  
}
```

```
print(subjects.keys())
```

➡ ["COMP10001", "MAST10006", "MAST10007", "SCIE10000"]

```
print(subjects.values())
```

➡ [  
 "Foundations of Computing",  
 "Calculus 2",  
 "Linear Algebra",  
 "Today's Science, Tomorrow's World"  
]

# Revision: Dictionaries

```
subjects = {  
    "COMP10001": "Foundations of Computing",  
    "MAST10006": "Calculus 2",  
    "MAST10007": "Linear Algebra",  
    "SCIE10000": "Today's Science, Tomorrow's World",  
}
```

```
print(subjects.items())
```

→

```
[  
    ("COMP10001", "Foundations of Computing"),  
    ("MAST10006", "Calculus 2"),  
    ("MAST10007", "Linear Algebra"),  
    ("SCIE10000", "Today's Science, Tomorrow's World")  
]
```

Question 4, given that `d = { "R": 0, "G": 255, "B": 0, "other": {"opacity": 0.6} }`, evaluate:

<code>"R" in d</code>	<code>d["R"]</code>	<code>d["R"] = 255</code>	<code>d["A"]</code>
<code>d["A"] = 50</code>	<code>d.pop("G")</code>	<code>d["other"]["blur"] = 0.1</code>	<code>d.items()</code>

Question 4, given that `d = { "R": 0, "G": 255, "B": 0, "other": {"opacity": 0.6} }`, evaluate:

<p><code>"R" in d</code></p> <p><b>True</b></p> <p>(this is used as a test if value belongs to a membership among keys)</p>	<p><code>d["R"]</code></p> <p><b>0</b></p>	<p><code>d["R"] = 255</code></p> <p><code>print(d)</code> now gives: <code>{'R': 255, 'G': 255, 'B': 0, 'other': {'opacity': 0.6}}</code></p>	<p><code>d["A"]</code></p> <p><b>KeyError: 'A'</b></p> <p>(requesting a nonexistent key gives an error)</p>
<p><code>d["A"] = 50</code></p> <p><code>print(d)</code> now gives: <code>{'R': 0, 'G': 255, 'B': 0, 'other': {'opacity': 0.6}, 'A': 50}</code></p> <p>(assigning to a nonexistent key adds that (key: value) pair to the dictionary)</p>	<p><code>d.pop("G")</code></p> <p>255 (removes key from dictionary, returning its value)</p> <p><code>print(d)</code> now gives: <code>{'R': 0, 'B': 0, 'other': {'opacity': 0.6}}</code></p>	<p><code>d["other"]["blur"] = 0.1</code></p> <p><code>print(d)</code> now gives: <code>{'R': 0, 'G': 255, 'B': 0, 'other': {'opacity': 0.6, 'blur': 0.1}}</code></p>	<p><code>d.items()</code></p> <p><code>dict_items([('R', 0), ('G', 255), ('B', 0), ('other', {'opacity': 0.6})])</code></p>

# Revision: Sets

$$A = \{1, 2, 3\}$$

$$B = \{1, 4, 9\}$$

Question 5, given that  $s1 = \{1, 2, 4\}$  and  $s2 = \{3, 4, 5\}$

$s1.add(7)$	$s1.add(2)$	$s2.remove(5)$
$s1 \& s2$ $s1.intersection(s2)$	$s1 \mid s2$ $s1.union(s2)$	$s1 - s2$ $s1.difference(s2)$



Question 5, given that  $s1 = \{1, 2, 4\}$  and  $s2 = \{3, 4, 5\}$

<p><code>s1.add(7)</code></p> <p>7 not in s1 set, so add 7 to s1: print(s1) now gives <b>{1, 2, 4, 7}</b></p>	<p><code>s1.add(2)</code></p> <p>s1 does not change since 2 is already in the set: print(s1) still gives <b>{1, 2, 4}</b></p>	<p><code>s2.remove(5)</code></p> <p>we remove 5 from s2: print(s2) now gives <b>{3, 4}</b> Note: doing this when 5 is not in s2 gives an error.</p>
<p><code>s1 &amp; s2</code> <code>s1.intersection(s2)</code></p> <p>same elements in s1 and s2 <b>{4}</b></p>	<p><code>s1   s2</code> <code>s1.union(s2)</code></p> <p>all unique elements in s1 and s2 <b>{1, 2, 3, 4, 5}</b></p>	<p><code>s1 - s2</code> <code>s1.difference(s2)</code></p> <p>elements in s1, but not in s2 <b>{1, 2}</b></p>

# Programming on Paper



# Problem 1 / 5

1. Write a function which takes a tuple of strings and returns a list containing only the strings which contain at least one exclamation mark or asterisk symbol. `words_with_symbols(('hi', 'there!', '*_*'))` should return `['there!', '*_*']`.

## Answer:

A:

```
def words_with_symbols(words):  
    with_symbols = []  
    for word in words:  
        for letter in word:  
            if letter in ('!', '*'):  
                with_symbols.append(word)  
                break  
    return with_symbols
```

## Problem 2 / 5

2. Write a function `sort_by_score(player_scores)` that takes a dictionary containing a player name as the key and their score as the value, and returns a list of `(score, player_name)` tuples sorted by highest to lowest score.

```
player_scores = {'Sonic': 299, 'Zelda': 421, 'Mario': 367, 'Pikachu': 152}
print(sort_by_score(player_scores))
```

Should output:

```
[(421, 'Zelda'), (367, 'Mario64'), (299, 'Sonic'), (152, 'Pikachu')]
```

## Answer:

A:

```
def sort_by_score(player_scores):
    score_players = []
    for player, score in player_scores.items():
        score_players.append((score, player))
    return sorted(score_players)[::-1]
```

# Problem 3 / 5

3. Write a function which takes a string as input and prints the frequency of each character in the string using a dictionary. `freq_counts('booboo')` should print:

```
b 2  
o 4
```

## Answer:

A:

```
def freq_count(words):  
    freqs = {}  
    for letter in words:  
        if letter in freqs:  
            freqs[letter] += 1  
        else:  
            freqs[letter] = 1  
    for key in freqs:  
        print(key, freqs[key])
```

# Problem 4 / 5

4. Write a function which takes a string, a character and an integer threshold and returns `True` if the character appears in the string with a frequency above the threshold, `False` if it appears at or below the threshold, and `None` if it doesn't appear at all. `above_thresh('I like the letter e', 'e', 3)` should return `True`.

## Answer:

A:

```
def above_thresh(text, char, threshold):  
    if char not in text:  
        return None  
    freq_dict = {}  
    for letter in text:  
        if letter in freq_dict:  
            freq_dict[letter] += 1  
        else:  
            freq_dict[letter] = 1  
    return freq_dict[char] > threshold
```

# Problem 5 / 5

5. **Challenge:** Write a function called `decode(key1, key2, ciphertext)` that takes two string keys and a string `ciphertext` to decode. To decode it:

- **Even indices of ciphertext:** If the character at this index of the `ciphertext` is in `key1` but not in `key2` then skip it, otherwise add it to the cleartext.
- **Odd indices of ciphertext:** If the character at this index of the `ciphertext` is in `key2` but not in `key1` then skip it, otherwise add it to the cleartext.

Your function should return the cleartext (decoded) string at the end.

```
key1 = "a01g4ds4?5atpv.qy52"  
key2 = "asb8gh.dvt7xyzlmz3"  
ciphertext = "0y5mpzpxpoquq 0s4zqhoh5l5hqv?eqh2xp8qx03p85hd3?m0x?zqz5b mim2bt!"  
print(decode(key1, key2, ciphertext))
```

The above code should print out:

```
you solved it!
```

# Problem 5 / 5

## Answer:

A:

```
def decode(key1, key2, ciphertext):  
    # set difference between key1 and key2 and vice versa  
    diff_even = set(key1) - set(key2)  
    diff_odd = set(key2) - set(key1)  
  
    # alternate between checking if the char is in diff_even or diff_odd  
    cleartext = ""  
    for i in range(len(ciphertext)):  
        char = ciphertext[i]  
        if i % 2 == 0 and char not in diff_even:  
            cleartext += char  
        elif i % 2 == 1 and char not in diff_odd:  
            cleartext += char  
  
    return cleartext
```



# Independent Work

- **Project 1** will be released this week! (possibly today...)
  - If you're struggling, please try to find assistance early than later.
- **Do worksheet 9** on Ed (**due next Monday at 6pm**)
  - Remember that Ed worksheets contributes to 10% of your total score!
- **Raise your hand** if you have any questions!

Scan here for annotated slides

