



# CoGrammar

## List Comprehension & 2D Lists

**SKILLS  
FOR LIFE**

**SKILLS BOOTCAMPS**



Department  
for Education

# Lecture Objectives

1. Recall the fundamental characteristics of Lists.
2. Explain the concept of indexing in a 2D list.
3. Apply knowledge of 2D lists to traverse and manipulate elements.



# Poll

## Assessment



# CoGrammar

Recap on Lists

# List Comprehension

- ★ List comprehension is a condensed method for creating lists in Python. In comparison to conventional for-loops, it offers a more condensed syntax for creating lists.

List Comprehension:

# Basic Structure

```
new_list = [expression for item in iterable]
```

# Squaring numbers from 0 to 9

```
squares = [x**2 for x in range(10)]
```

# Result: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

### List Comprehension:

#### # Basic Structure

```
new_list = [expression for item in iterable]
```

#### # Squaring numbers from 0 to 9

```
squares = [x**2 for x in range(10)]
```

```
# Result: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

- ★ **Expression:** The expression to be evaluated and included in the new list.
- ★ **Item:** The variable representing an element in the iterable (e.g., a range, list, string).
- ★ **Iterable:** The source of data to iterate over.

# Benefits & Precautions

## ★ Benefits:

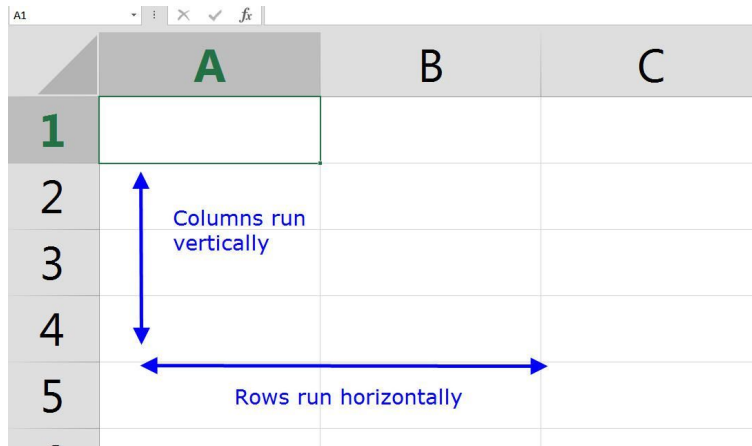
- **Conciseness:** Achieve the same result with less code.
- **Readability:** Express your intent more clearly and compactly.
- **Efficiency:** List comprehensions are often faster than equivalent for-loops.

## ★ Considerations:

- **Avoid Complexity:** While list comprehensions are powerful, avoid making them overly complex for the sake of readability.
- **Conditional Expressions:** You can use ternary expressions for conditional inclusion.

# 2D List

- ★ A List within a List.
- ★ Outer List (1 Dimension) + Inner List (1 Dimension) = 2D



The diagram shows an Excel spreadsheet with columns labeled A, B, and C, and rows labeled 1 through 5. A blue double-headed vertical arrow spans rows 1 to 5 in column A, with the text "Columns run vertically" next to it. A blue double-headed horizontal arrow spans columns A to C in row 5, with the text "Rows run horizontally" below it.

	A	B	C
1			
2			
3			
4			
5			



# Rows and Columns

- ★ Elements are essentially accessed using rows and column indices.

```
two_d = [  
    [1, 2, 3],  
    [4, 5, 6],  
    [7, 8, 9]  
]
```



# Question:

**What is a 2D list, and how does it differ from a 1D list?**



# Traversing

- ★ Nested Loops (iterate through rows and columns)
- ★ List comprehension

	0	1	2
0	(0,0)	(0,1)	(0,2)
1	(1,0)	(1,1)	(1,2)
2	(2,0)	(2,1)	(2,2)

Column Index

Row Index



# Poll

## Assessment



# Wrapping Up

---

## 2D Lists

2D lists in Python offer a powerful mechanism for organising and manipulating data in a structured manner.

## Rows and Columns

Rows represent individual lists within the main list, while columns denote elements within each of these lists.

## Traversal

Whether it's accessing specific elements, performing operations on the entire list, or searching for particular values, traversing techniques are central to unleashing the full potential of 2D lists.

# CoGrammar

Questions around 2D Lists



# CoGrammar

**Thank you for joining**

