

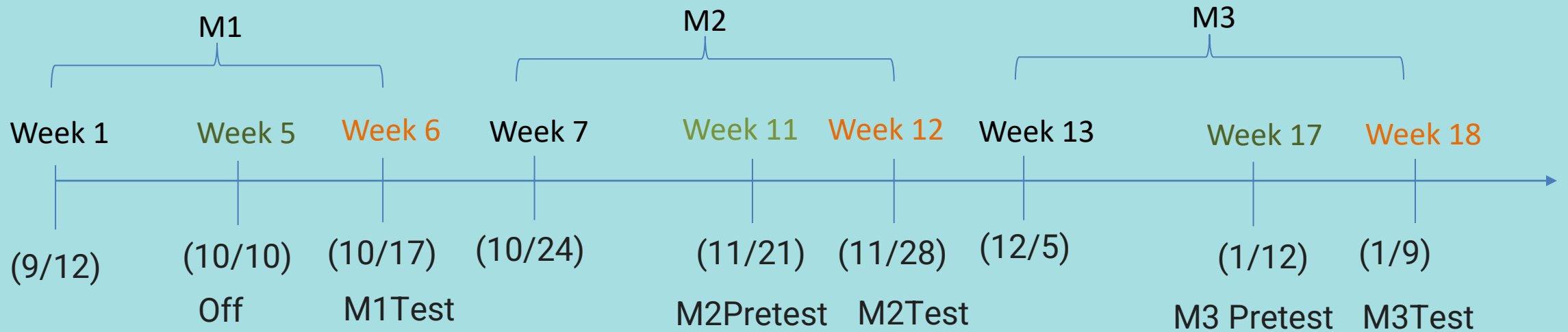
A row of stylized books in various colors (white, red, blue, yellow) with some having decorative patterns like stripes or dots.

Fundamental Programming Course

Week 4

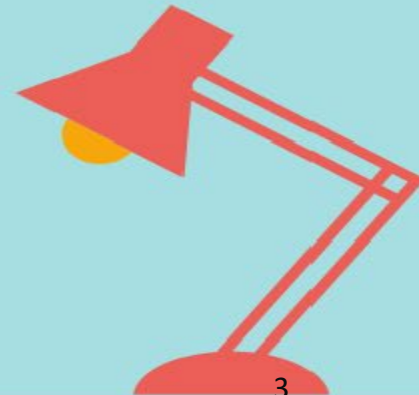
Huseh-Ting Chu@Asia University, 2021

Schedule



Syllabus

- W1-Python Introduction and Programming Tools
- W2-Variables and Operations
- W3-Loop and formatted output
- **W4-Condition and Containers**
- W5-String and built-in functions
- W6-M1 test
- W7-Dictionary Container
- W8-File I/O
- W9-Function
- W10-Advanced flow control
- W11-Advanced operations and generators
- W12-M2 test
- W13-Advanced functions
- W14-Class fundamentals (classes, objects, properties, constructors, methods)
- W15-Advanced Classes (Static methods, class Methods and class decorators)
- W16-Modules and Packages
- W17-Advanced programming(Argparse and Venv)
- W18-M3 test



Python cheat sheet

Beginner's Python Cheat Sheet

Variables and Strings

Variables are used to store values. A string is a series of characters, surrounded by single or double quotes.

Hello world

```
print("Hello world!")
```

Hello world with a variable

```
msg = "Hello world!"  
print(msg)
```

Concatenation (combining strings)

```
first_name = 'albert'  
last_name = 'einstein'  
full_name = first_name + ' ' + last_name  
print(full_name)
```

Lists

A list stores a series of items in a particular order. You access items using an index, or within a loop.

Make a list

```
bikes = ['trek', 'redline', 'giant']
```

Get the first item in a list

```
first_bike = bikes[0]
```

Get the last item in a list

```
last_bike = bikes[-1]
```

Looping through a list

```
for bike in bikes:  
    print(bike)
```

Adding items to a list

```
bikes = []  
bikes.append('trek')  
bikes.append('redline')  
bikes.append('giant')
```

Making numerical lists

```
squares = []  
for x in range(1, 11):  
    squares.append(x**2)
```

Lists (cont.)

List comprehensions

```
squares = [x**2 for x in range(1, 11)]
```

Slicing a list

```
finishers = ['sam', 'bob', 'ada', 'bea']  
first_two = finishers[:2]
```

Copying a list

```
copy_of_bikes = bikes[:]
```

Tuples

Tuples are similar to lists, but the items in a tuple can't be modified.

Making a tuple

```
dimensions = (1920, 1080)
```

If statements

If statements are used to test for particular conditions and respond appropriately.

Conditional tests

equals	x == 42
not equal	x != 42
greater than	x > 42
or equal to	x >= 42
less than	x < 42
or equal to	x <= 42

Conditional test with lists

```
'trek' in bikes  
'surly' not in bikes
```

Assigning boolean values

```
game_active = True  
can_edit = False
```

A simple if test

```
if age >= 18:  
    print("You can vote!")
```

If-elif-else statements

```
if age < 4:  
    ticket_price = 0  
elif age < 18:  
    ticket_price = 10  
else:  
    ticket_price = 15
```

Dictionaries

Dictionaries store connections between pieces of information. Each item in a dictionary is a key-value pair.

A simple dictionary

```
alien = {'color': 'green', 'points': 5}
```

Accessing a value

```
print("The alien's color is " + alien['color'])
```

Adding a new key-value pair

```
alien['x_position'] = 0
```

Looping through all key-value pairs

```
fav_numbers = {'eric': 17, 'ever': 4}  
for name, number in fav_numbers.items():  
    print(name + ' loves ' + str(number))
```

Looping through all keys

```
fav_numbers = {'eric': 17, 'ever': 4}  
for name in fav_numbers.keys():  
    print(name + ' loves a number')
```

Looping through all the values

```
fav_numbers = {'eric': 17, 'ever': 4}  
for number in fav_numbers.values():  
    print(str(number) + ' is a favorite')
```

User input

Your programs can prompt the user for input. All input is stored as a string.

Prompting for a value

```
name = input("What's your name? ")  
print("Hello, " + name + "!")
```

Prompting for numerical input

```
age = input("How old are you? ")  
age = int(age)
```

```
pi = input("What's the value of pi? ")  
pi = float(pi)
```

Python Crash Course

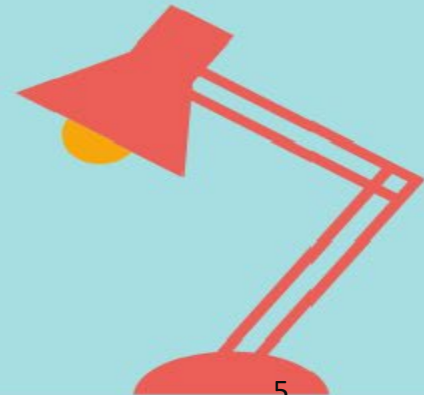
Covers Python 3 and Python 2

nostarchpress.com/pythoncrashcourse



This Week's Content – Loop and Formatting Output

- Essential - basic
 - IPO model: input–process–output (input-processing-output)
 - Input: use of map() function
 - Process: if conditional
 - Process: Comparison Operators, Boolean Values and Boolean Logic
 - Output: string format(), zfill(), center(), rjust() and ljust() functions
- Advanced-Advanced
 - Data containers: List, Tuple, Dict, Set
 - Built-in functions - max(), min(), sum()



Kissipo Learning

Kissipo = KISS principle + IPO model

KISS principle

"keep it simple, stupid" or "keep it stupid simple", is a design principle noted by the U.S. Navy in 1960.

https://en.wikipedia.org/wiki/KISS_principle

IPO model

The input–process–output (IPO) model is a widely used approach in systems analysis and software engineering for describing the structure of an information processing program or other process.

https://en.wikipedia.org/wiki/IPO_model



Kissipo Learning for Programming with Python(PWP)

Courseware: Notebook+ Github

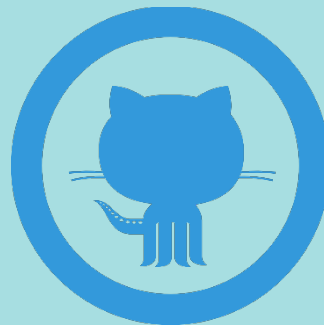
- (1) Teaching with Notebook (Google Colab).
- (2) Use Github to build lesson plans

Keep:

Variables and assignment
operator and expression
left-hand side and right-hand side
unpacking

S&S:

help(), type(), len(), size()



IPO-I: input

input()
int(), float(), str()
split(), map()

IPO-P: Process

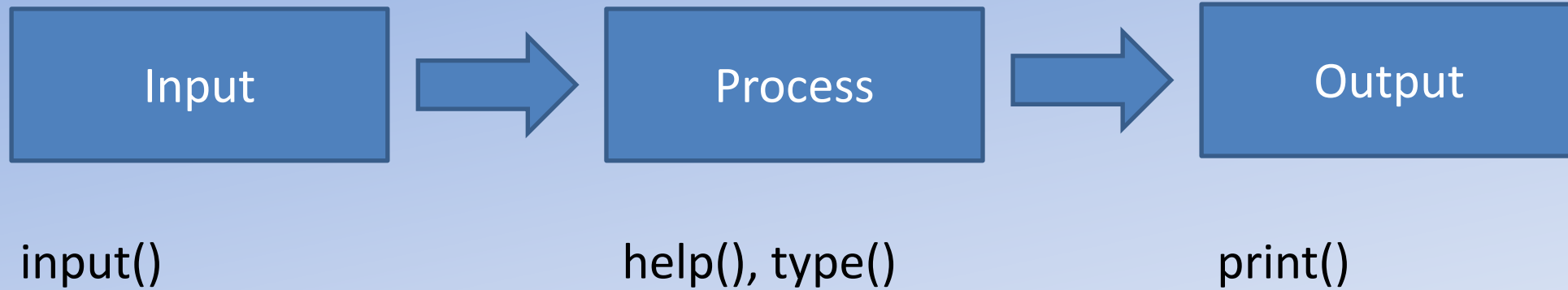
for-loop/while-loop
if, elif, else
range()

IPO-O: output

print()
open(), write()



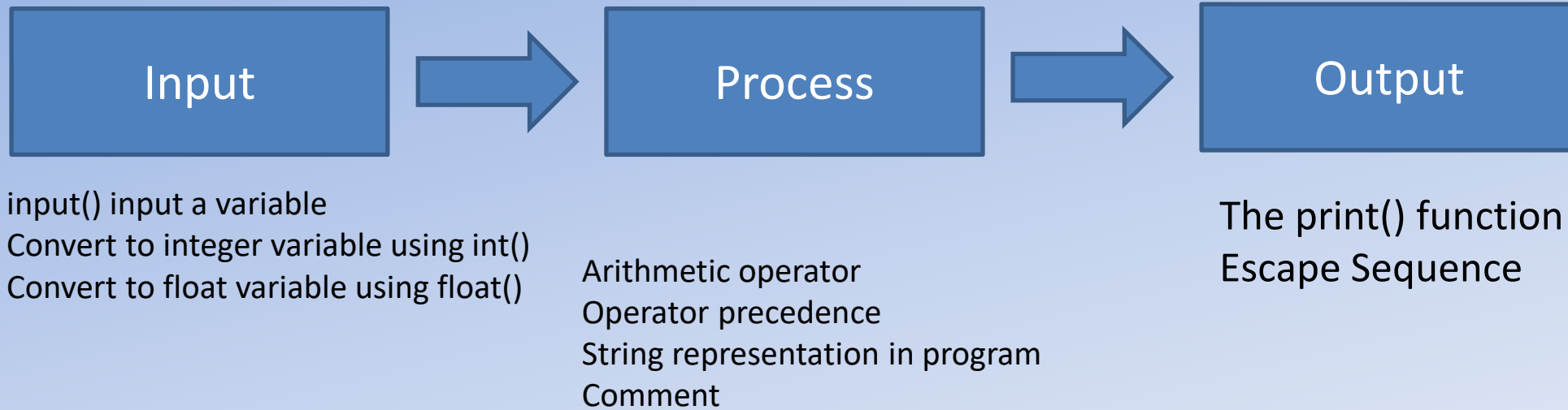
IPO Model(1)



The basic idea of this chapter is that students should know:
Input with `input()`, output with `print()`
`help()` can view the description of the function or category
`type()` can check the type of the variable

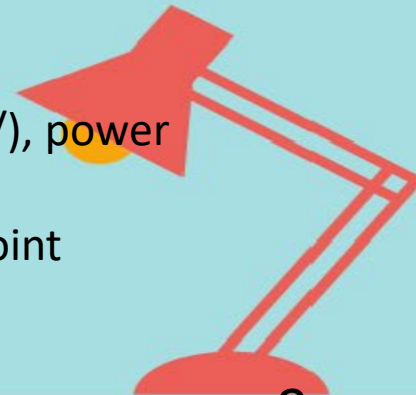


IPO Model (2)

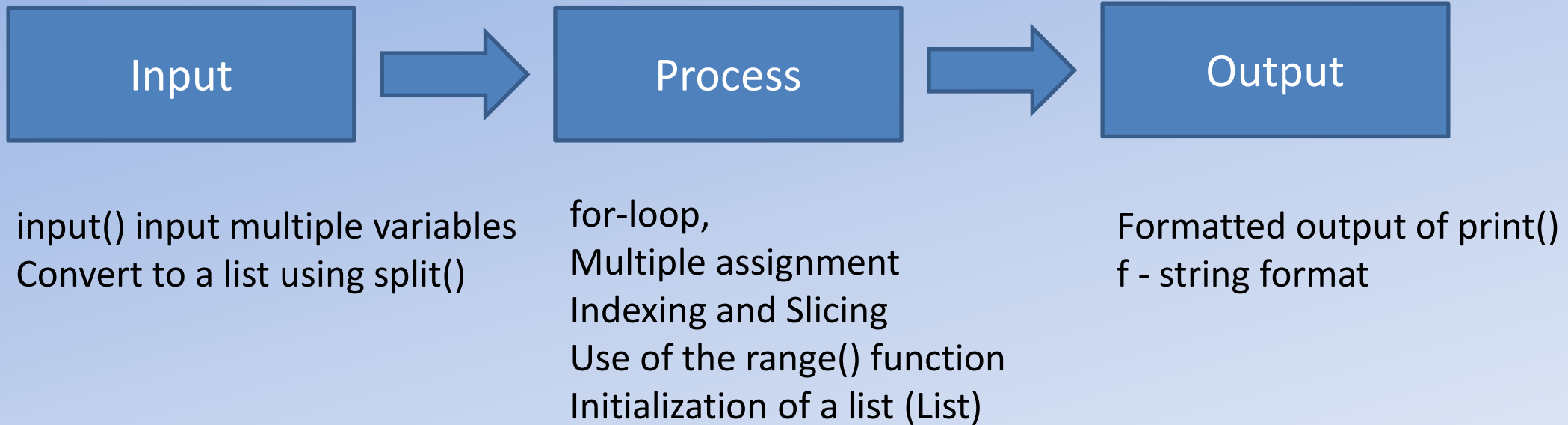


The basic idea of this chapter is that students should know:

- How to use `input()` to input numbers of different types
- Output `print()` has two parameters `sep` and `end` to control the output
- Arithmetic operations in Python include: addition, subtraction, multiplication and division (+ - * /), power (**), quotient (//) and remainder (%).
- The result of addition, subtraction, multiplication and division (+ - * /), power (**) is a floating point number. The quotient (//) and remainder (%) calculations result in integers.



IPO Model (3)

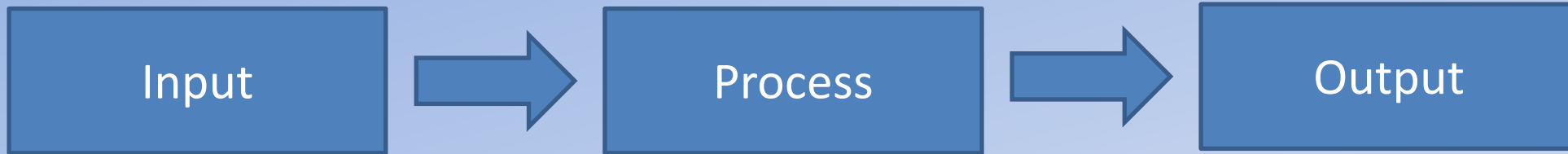


The basic idea of this chapter is that students should know:

- How to enter multiple variables with `input()`
- formatted output with f-format string
- Use of loops.
- What is Indexing and Slicing



IPO Model (4)



Use map() to input multiple variables

if conditional
comparison operator
Boolean Values and Boolean Logic
while loop

The basic idea of this chapter is that students should know:

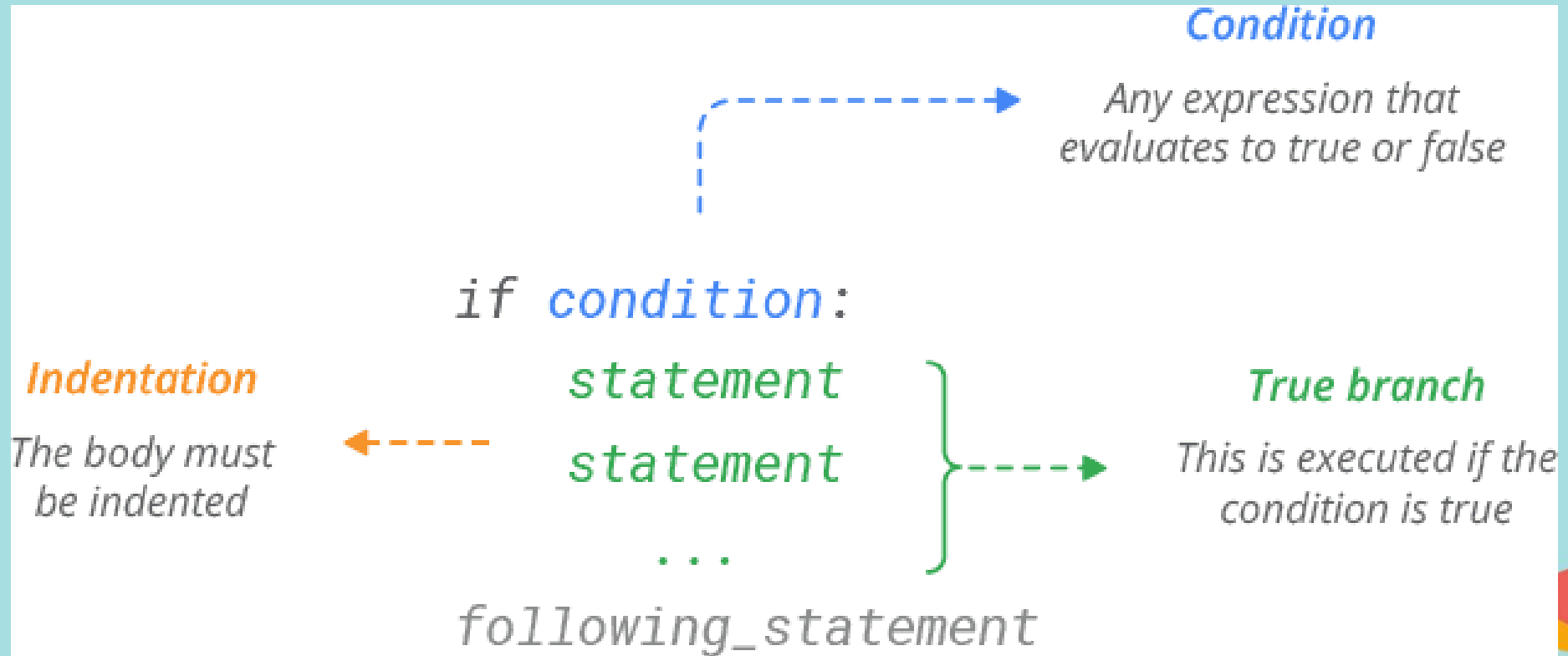
Use of conditional expressions.

Advanced String Formatting

Container: List, Tuple, Dict, Set



if condition



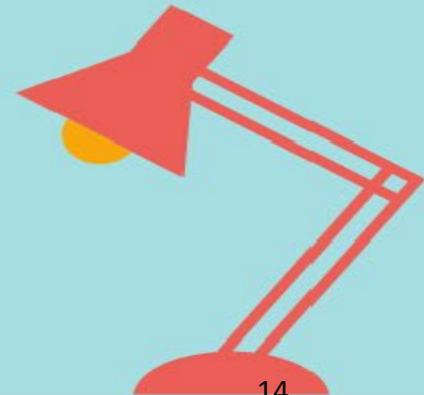
Topic 1-if condition statement

- Step 1: Boolean variables
- Step 2: if-condition
- Step 3: if-else-condition
- Step 4: if-elif-else-condition



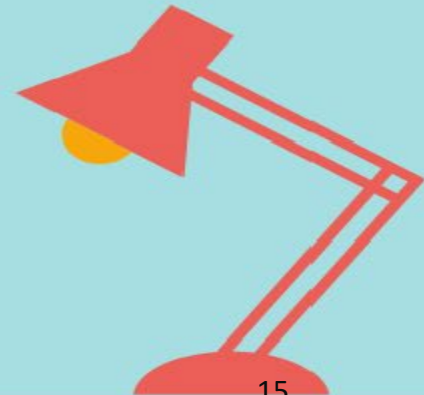
Topic 2-Operators

- Step 1: Comparison Operators: $>$, $>=$, $<$, $<=$, $==$
- Step 2: Boolean Operators: and, or, not



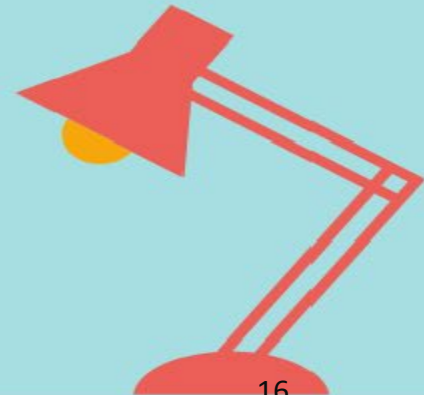
Topic 3-for-loop-if-condition

- Step 7: Find all factors of a number
- Step 8: Determine if a number is a prime number, with no factors other than 1 and itself
- Step 9: break-for-loop
- Step 10: continue-for-loop



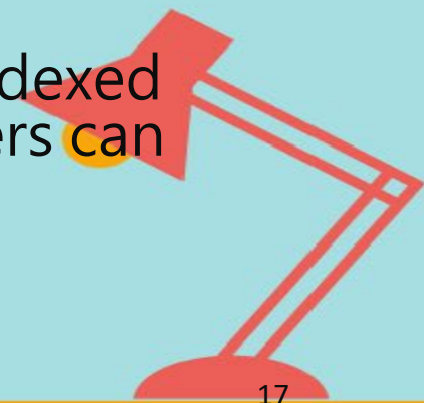
Topic 4-While Loop

- Step 1: Use a number to control the number of executions
- Step 2: Use a loop to control whether the program ends



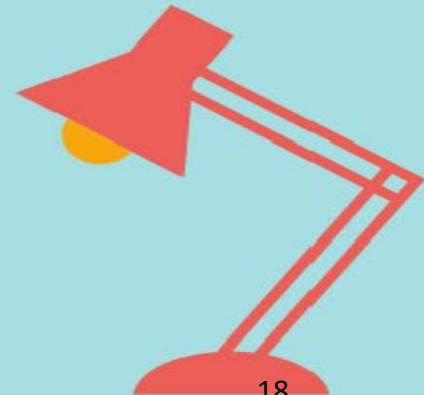
Container

- There are three types of parentheses in the Python language,
 - Parentheses () represent tuple data types,
 - Square brackets [] represent the data type of the list,
 - The curly brackets {} represent the dictionary (dict) or set (set) data type.
-
- Serial put a bunch of variables into [] to form a row of data
 - A tuple consists of several values separated by commas
 - A set is an unordered set of elements without repetitions
 - A dictionary (dict) puts a bunch of key and value correspondences into {} to form a collection of indexable data
 - Sequences are indexed by a range of numbers, and dictionaries are indexed by keys, which can be of any immutable type; both strings and numbers can be used as key values.



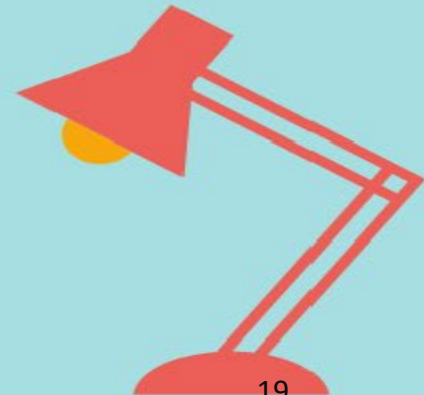
Topic 5: Container

- Step 1: Initialization of data container variables
- Step 2: Index of data container variables
- Step 3: Set comprehensions
- Step 4: Dictionary comprehensions



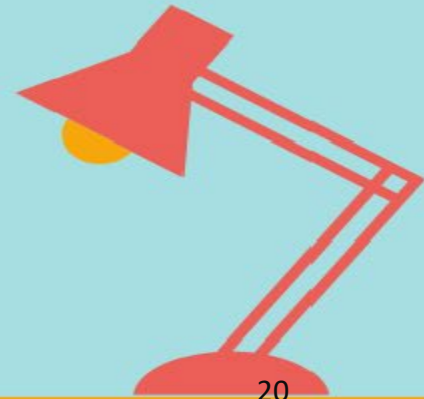
Topic 6: Loops using containers

- Step 1: in data container variable
- Step 2: Use items()
- Step 3: Use enumerate()
- Step 4: Use zip()



Topic 7:map()

- Step 1: Calculate the variable length of the data container
- Step 2: Convert the input string to an integer





Thanks!

Q&A

