Module 2

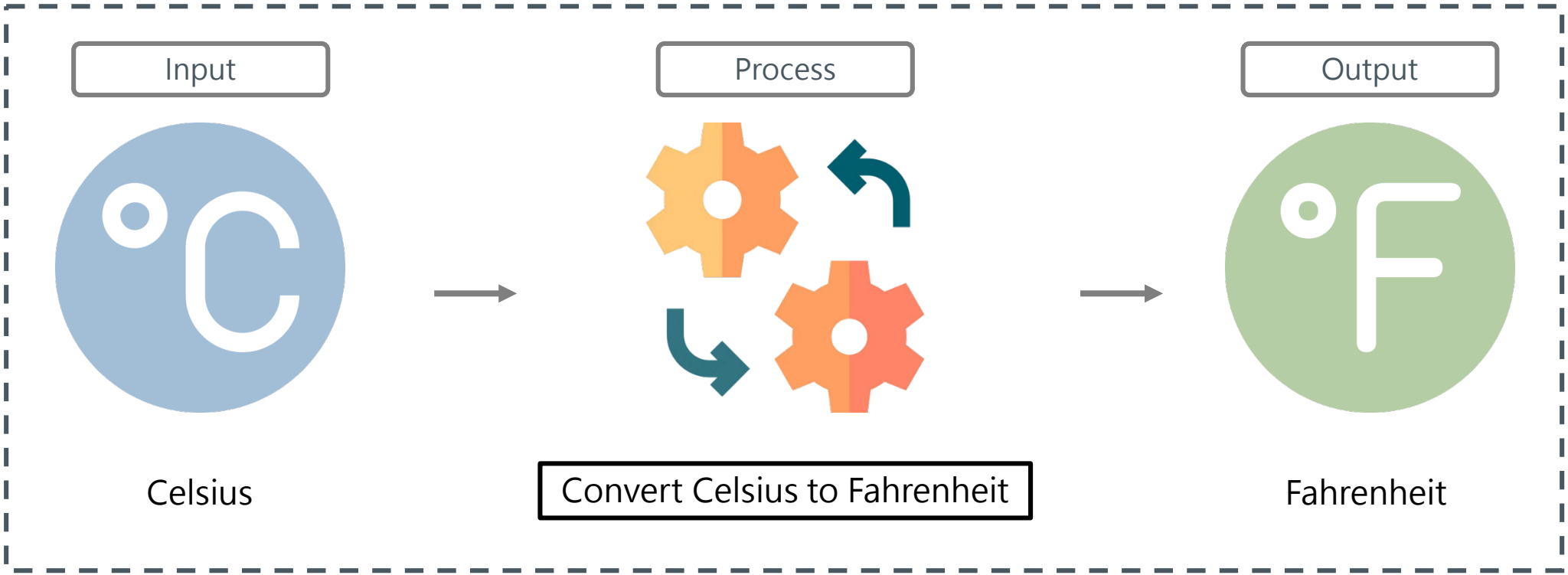
Elementary Programming: Input, Processing and Output

Instructor: Jonny C.H Yu

Input, Processing, and Output(IPO)

- Typically, computer performs three-step process
  - Receive input
    - Input: any data that the program receives while it is running.
  - Perform some process on the input
    - Example: mathematical calculation.
  - Produce output

IPO Example: Temperature Converter



- Prompt the user for input (Celsius)
- Process it to convert it to Fahrenheit using F = 9/5(C) + 32
- Output the result by displaying it on the screen.

IPO Example: Temperature Converter

- Prompt the user for input (Celsius)
- Process it to convert it to Fahrenheit using F = 9/5(C) + 32
- Output the result by displaying it on the screen.

```
[1]: #
     # An IPO Example: Convert Celsius to Fahrenheit
     #

[2]: celsius = float(input("What is the Celsius temperature?"))
     What is the Celsius temperature? 25.6

[3]: fahrenheit = (9 / 5) * celsius + 32

[4]: print("The temperature is ", fahrenheit, " degrees Fahrenheit.")
     The temperature is  78.08000000000001  degrees Fahrenheit.
```

Let us try it: download Codes_Module02.zip, unzip and run M2_TempConvert.ipynb

Displaying Output with the "print" Function

```
[4]: print("The temperature is ", fahrenheit, " degrees Fahrenheit.")
```

```
The temperature is  78.08000000000001  degrees Fahrenheit.
```

- Function: piece of prewritten code that performs an operation.
- `print` function: displays output on the screen.
- Argument: data given to a function.
  - Example: data that is printed to screen; often involve some **Strings**.

String

- String: sequence of characters that is used as data
  - Can be enclosed in single (') or double (") quote marks
  - Can be enclosed in triple quotes (''' or """)
    - Enclosed string can contain both single and double quotes and can have multiple lines

Examples

- May enclose a string in single quote or double quote.
- Python programmers prefer single quote.

```
[1]: print('Hello world')
```

```
Hello world
```

```
[2]: print("Hello world")
```

```
Hello world
```

- Statements execute in order

```
[1]: print('Jonny C.H. Yu')
     print('Assistant Professor of Engineering Science')
     print('National Cheng Kung University')
     print('Research Associate')
     print('Massachusetts Institute of Technology')
```

```
Jonny C.H. Yu
Assistant Professor of Engineering Science
National Cheng Kung University
Research Associate
Massachusetts Institute of Technology
```

Examples

- (Trick) Use double quote to contain single quote and single quote to contain double quote.

```
[4]: print("Don't know.")
     print("I'm here!")
```

```
Don't know.
I'm here!
```

```
[5]: print('Einstein said "Everything should be made as simple as possible, but no simpler."')
```

```
Einstein said "Everything should be made as simple as possible, but no simpler."
```

- (Trick) Triple quote (''' or """) can contain single quote and double quote
- (Trick) Triple quote can be used to surround multiline strings.

```
[6]: print("""I'm reading "Hamlet" tonight.""")
```

```
I'm reading "Hamlet" tonight.
```

```
[7]: print("""One
     Two
     Three""")
```

```
One
Two
Three
```

Let us try it and learn more on print function! (M2_PrintFunction.ipynb)

Comments

- Comments: notes of explanation within a program.
  - Ignored by Python interpreter.
    - Intended for a person reading the program's code.
  - Begin with a # character

[1]:
```
#
# An IPO Example: Convert Celsius to Fahrenheit
#
```

Variables

```
[2]: celsius = float(input("What is the Celsius temperature?"))

What is the Celsius temperature? 25.6

[3]: fahrenheit = (9 / 5) * celsius + 32
```

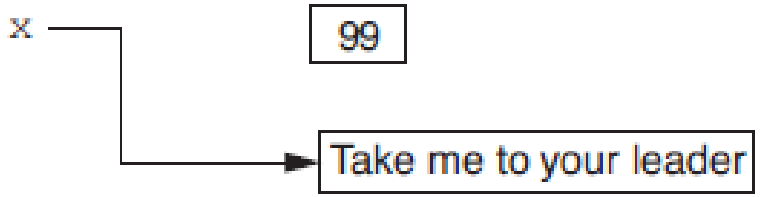
- Variable: to access and manipulate data stored in memory.
  - A variable references the value it represents.
- Assignment statement: to create a variable and make it reference data
  - General format is `variable = expression`
    - Example: `age = 29`
    - Assignment operator: the equal sign (=)

Variables Reassignment

- Variables can reference different values while program is running.
- A variable can refer to item of any type.
  - Variable that has been assigned to one type can be reassigned to another type.

Figure 2-7 The variable x references an integer

Figure 2-8 The variable x references a string



Examples

- Pass variable as an argument to `print` function

```
[1]: width = 10

[2]: print('width')
width

[3]: print(width)
10

[4]: print('The width of rectangle is')
print(width)
The width of rectangle is
10
```

- Display multiple items with `print` function

```
[5]: print('The width of rectangle is', width)
The width of rectangle is 10
```

  - Items are separated by commas when passed as arguments
  - Arguments displayed in the order they are passed to the function
  - Items are automatically separated by a space when displayed on screen

Examples

- Variable reassignment

[6]:
```
dollars = 2.75
print ('I have', dollars, 'in my account')
dollars = 99.95
print ('But now I have', dollars, 'in my account')
```

```
I have 2.75 in my account
But now I have 99.95 in my account
```

[7]:
```
x = 99
print(x)
x = 'Take me to your leader'
print(x)
```

```
99
Take me to your leader
```

Let us try it and learn more on variable and assignment. (M2_Variable.ipynb)

Reading Input from the Keyboard

- Most programs need to read input from the user.
- Built-in `input` function reads input from keyboard.
  - Returns the data as a string.
  - Format: *`variable = input(prompt)`*
    - `prompt` is typically a string instructing user to enter a value.

```
[1]: name = input('What is your name? ')

     What is your name?  Jonny

[2]: print('Hello', name)

     Hello Jonny
```

Let us try it! M2_InputShort.ipynb

Reading Numbers with the `input` Function

- `input` function always returns a string
- Built-in functions convert between data types
  - `int(`*`item`*`)` converts *`item`* to an `int`
  - `float(`*`item`*`)` converts *`item`* to a `float`
  - Nested function call: general format:
    `function1``(`*`function2`*`(`*`argument`*`))`
    - value returned by function2 is passed to function1

```
[2]: celsius = float(input("What is the Celsius temperature?"))
```

Examples

- Write a program to ask the user to enter his/her name, age and income and display these data.

```
[3]: name = input('What is your name? ')
     age = int(input('What is your age? '))
     income = float(input('What is your income? '))
```

```
What is your name?  Chris
What is your age?  20
What is your income?  35000.0
```

```
[4]: print('Here is the data you entered:')
     print('Name:', name)
     print('Age:', age)
     print('Income:', income)
```

```
Here is the data you entered:
Name: Chris
Age: 20
Income: 35000.0
```

Let us try it! (M2_InputLong.ipynb)What happens if you enter xyz for age?

Performing Calculations

- Math expression: performs calculation and gives a value
  - Math operator: tool for performing calculation
  - Operands: values surrounding operator
    - Variables can be used as operands
- Two types of division:
  - / operator performs floating point division
  - // operator performs integer division
    - Positive results truncated, negative rounded away from zero

Table 2-3 Python math operators

Symbol	Operation
+	Addition
–	Subtraction
*	Multiplication
/	Division
//	Integer division
%	Remainder
**	Exponent

```
[1]: 5 / 2

[1]: 2.5

[2]: 5 // 2

[2]: 2

[3]: -5 // 2

[3]: -3
```

Operator Precedence and Grouping with Parentheses

- Python operator precedence:
  1. Operations enclosed in parentheses
     - Forces operations to be performed before others
  2. Exponentiation (**)
  3. Multiplication (*), division (/ and //), and remainder (%)
  4. Addition (+) and subtraction (-)

- Higher precedence performed first
  - Same precedence operators execute from left to right

The Exponent Operator and the Remainder Operator

- Exponent operator (`**`): Raises a number to a power
  - `x ** y` = x^y

- Remainder operator (`%`): Performs division and returns the remainder.
  - a.k.a. modulus operator
  - e.g., `4%2=0, 5%2=1`
  - Typically used to convert times and distances, and to detect odd or even numbers.

Examples

- Get a number of μm from the user and convert it to cm, mm and μm.

```
[1]: total_microns = float(input('Enter the total number of microns: '))
```

```
Enter the total number of microns:  37659
```

```
[2]: #Get the number of cm
     cm = total_microns // 10000
```

```
[3]: #Get the number of remaining mm
     mm = (total_microns // 1000) % 10
```

```
[4]: #Get the number of remaining microns
     um = total_microns % 1000
```

```
[5]: print('Here is the total', total_microns, 'um in cm, mm, and um:')
     print('Centimeters: ', cm)
     print('Millimeters: ', mm)
     print('Microns: ', um)
```

```
Here is the total 37659.0 um in cm, mm, and um:
Centimeters:  3.0
Millimeters:  7.0
Microns:  659.0
```

Let us try it! M2_MicronConvert.ipynb

Recap and More: Breaking Long Statements into Multiple Lines

- Multiline continuation character (\):

Allows to break a long statement into multiple lines

```
result = var1 * 2 + var2 * 3 + \
         var3 * 4 + var4 * 5
```

- **A statement enclosed in parentheses can be broken without \.**

```
print("Monday's sales are", monday,
      "and Tuesday's sales are", tuesday,
      "and Wednesday's sales are",
  Wednesday)
total = (value1 + value2 +
         value3 + value4 +
         value5 + value6)
```

More About Data Output

- `print` function displays line of output
  - Newline character at end of printed data
  - Special argument `end='delimiter'` causes `print` to place *`delimiter`* at end of data instead of newline character

```
[7]: print('One')
     print('Two')
     print('Three')
```

```
One
Two
Three
```

```
[8]: print('One', end = ' ')
     print('Two', end = ' ')
     print('Three')
```

```
One Two Three
```

Recap and More:
More About Data Output (cont'd.)

- Special characters appearing in string literal
  - Preceded by backslash (\)
    - Examples: newline (\n), horizontal tab (\t)
  - Treated as commands embedded in string

Table 2-8 Some of Python's escape characters

Escape Character	Effect
\n	Causes output to be advanced to the next line.
\t	Causes output to skip over to the next horizontal tab position.
\'	Causes a single quote mark to be printed.
\"	Causes a double quote mark to be printed.
\\	Causes a backslash character to be printed.

```
[12]: print("Mon\tTue\tWed")
      Mon     Tue     Wed

[13]: print('The path is C:\\temp\\data')
      The path is C:\temp\data
```

Formatting Numbers

- Can format display of numbers on screen using built-in `format` function
  - Two arguments:
    - Numeric value to be formatted
    - Format specifier
  - Returns string containing formatted number

```
[14]: print(format(12345.6789, '.2f'))
```

```
12345.68
```

```
[15]: print(format(12345.6789, '.2e'))
```

```
1.23e+04
```

Examples

- Set field width and print numbers aligned in columns.

```
[15]: num1 = 127.899
      num2 = 3456.123
      num3 = 799.999
      num4 = 3.1
```

```
[16]: print(format(num1, '7.2f'))
      print(format(num2, '7.2f'))
      print(format(num3, '7.2f'))
      print(format(num4, '7.2f'))
```

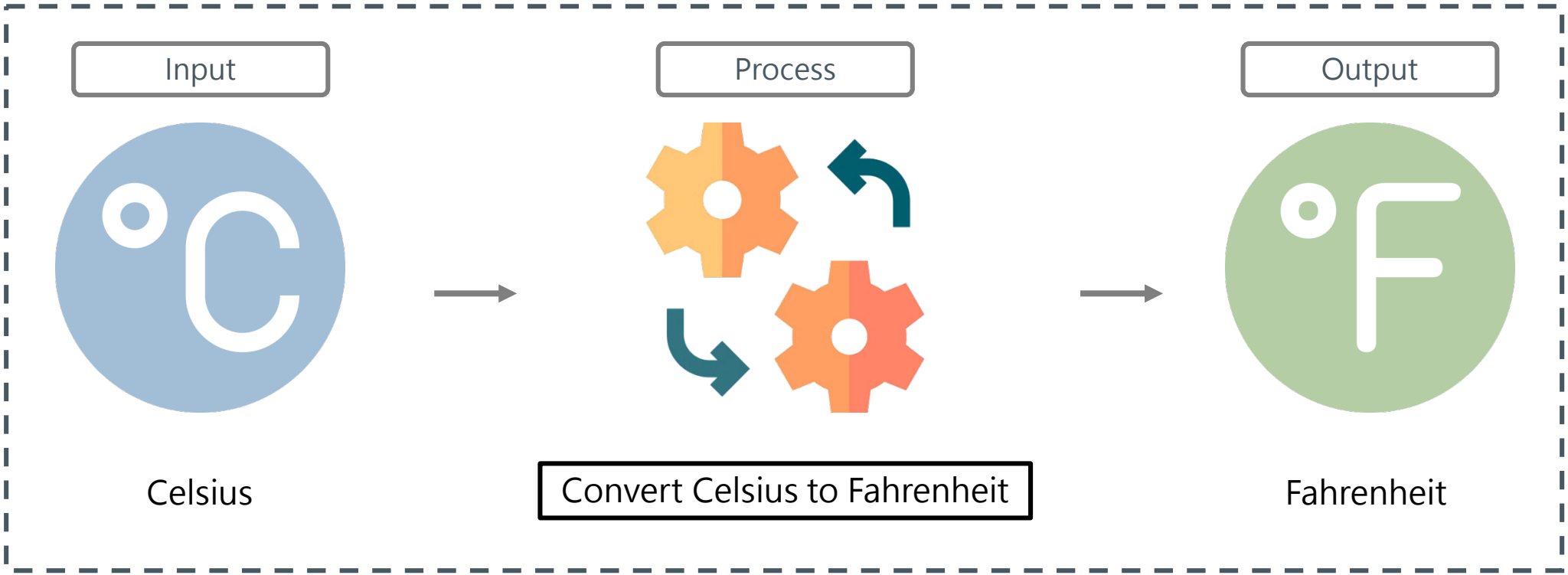
```
 127.90
3456.12
 800.00
   3.10
```

- Insert comma separators.

```
[17]: print(format(12345.6712, ',.2f'))
```

```
12,345.67
```

Revisit IPO Example: Temperature Converter



- Prompt the user for input (Celsius)
- Process it to convert it to Fahrenheit using F = 9/5(C) + 32
- Output the result by displaying it on the screen.

Revisit IPO Example: Temperature Converter

- Prompt the user for input (Celsius)
- Process it to convert it to Fahrenheit using F = 9/5(C) + 32
- Output the result by displaying it on the screen.

```
[1]: #
     # An IPO Example: Convert Celsius to Fahrenheit
     #

[2]: celsius = float(input("What is the Celsius temperature?"))

     What is the Celsius temperature? 25.6

[3]: fahrenheit = (9 / 5) * celsius + 32

[4]: print("The temperature is ", fahrenheit, " degrees Fahrenheit.")

     The temperature is  78.08000000000001  degrees Fahrenheit.

[5]: print("The temperature is ", format(fahrenheit, '.2f'), " degrees Fahrenheit.")

     The temperature is  78.08  degrees Fahrenheit.
```

Let us rewrite M2_TempConvert.ipynb

Summary

- This module covered:
  - Input, Processing, and Output
  - Displaying Output with `print` Function
  - Comments
  - Variables
  - Query variable's `type`
  - Reading Input from the Keyboard
  - Performing Calculations
  - More About Data Output
  - Formatting Numbers

To be continued…