Python - Functions

CS101

Motivation

- ถึงตอนนี้ การเขียนโค้ดของเรามีลักษณะเป็นลำดับของคำสั่งเรียงกันอยู่ในไฟล์เดียว
- เปรียบเทียบกับการเดินทาง เหมือนเราออกเดินทางจากจุดเริ่มต้นตรงดิ่งไปยังจุดหมายปลายทาง
- ในการเขียนโปรแกรม การเรียกใช้ฟังก์ชันจะเหมือนการแวะระหว่างทาง
  - เช่นเราอาจจะแวะเติมน้ำมัน แวะกินข้าว แวะเยี่ยมญาติ
  - และเมื่อเราทำภารกิจเหล่านั้นเสร็จเราก็จะกลับเข้าสู่ทางหลัก ณ จุดที่เราออกจากทางหลักไป

What is a function

- ฟังก์ชันคือลำดับของคำสั่งที่มีการตั้งชื่อไว้สำหรับเรียกใช้คำนวณ (a named sequence of statements)
- ฟังก์ชันมีการรับอินพุทเข้าไป และ(อาจ)ส่งผลลัพธ์กลับมา
- Example

```
>>> a = len("Python 3")
>>> print(a)
>>> 8
```

- We call a function len( ) which takes one argument and returns a result. The result is called the return value.

Why use functions? (1/2)

- Abstraction
  - ในการเรียกใช้ฟังก์ชันใด ๆเราต้องรู้เพียงว่า
    - ฟังก์ชันนั้นมีชื่อเรียกว่าอะไร
    - ฟังก์ชันนั้นมีหน้าที่อะไร
    - ฟังก์ชันนั้นต้องการ arguments อะไรบ้าง
    - ฟังก์ชันนั้นจะส่งผลลัพธ์อะไรกลับมา
  - เหมือนกับการขับรถยนต์ คุณไม่จำเป็นต้องรู้ว่าเครื่องยนต์ ระบบขับเคลื่อน หรือระบบช่วงล่างทำงานอย่างไร ถ้าสิ่งที่คุณต้องการคือใช้งานรถ
  - แน่นอน ว่าหากเราต้องการแต่งรถ ปรับปรุง เปลี่ยนแปลง หรือสร้างรถยี่ห้อของตัวเอง เราจำเป็นจะต้องรู้การทำงานของระบบต่าง ๆเหล่านั้น (เราจะเรียนการสร้างฟังก์ชันเองในบทต่อ ๆไป)

Why use functions? (2/2)

- Reusability
  - เรียกใช้การคำนวณเดิมๆหลายครั้งในโปรแกรมเดียว
  - เรียกใช้การคำนวณเดิมๆหลายครั้งในหลายๆโปรแกรม
- Easy maintenance, Readability

```
1 sentence = "Python 3"
2
3 if len(sentence) > 5:
4     print("Too long")
5 else
6     print("Okay")
7
```

```
1 sentence = "Python 3"
2 i = 1
3 while True:
4     try:
5         sentence[i]
6         i = i + 1
7     except IndexError:
8         break
9
10 if i > 5:
11     print("Too long")
12 else
13     print("Okay")
14
```

Functions in Python

- Built-in
  - กลุ่มฟังก์ชันที่มากับ Python สามารถเรียกใช้ได้เลย
  - See a list of the functions next page.
- User defined
  - ฟังก์ชันที่ผู้ใช้สร้างขึ้นเองสำหรับงานที่เฉพาะเจาะจงมากขึ้น
- ในบทนี้เราจะเรียนรู้การเรียกใช้ฟังก์ชันแบบ built-in ก่อน
  - Pay attention to!
    - Function specification: ฟังก์ชันนั้นต้องการ argument กี่ตัว และมีลำดับการส่ง argument อย่างไร
    - Return value: เมื่อฟังก์ชันทำงานเสร็จสิ้นจะส่งผลลัพธ์อะไรกลับมา

Python built-in functions

		Built-in Functions		
abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

Two types of built-in function

- General function
  - สามารถกระทำกับตัวแปรหลายๆแบบได้ ยกตัวอย่างเช่น len()
  - len() สามารถหาความยาวของ string, หรือ list ได้
  - มีการเรียกใช้แบบ function ทางคณิตศาสตร์ len(argument)
- Object-type specific function
  - มีความจำเพาะกับตัวแปรนั้นๆ เช่น ฟังก์ชันทำเป็น uppercase จะใช้กับ string เท่านั้น int หรือ float ทำไม่ได้
  - มีการเรียกใช้ต่างออกไป โดยการใช้เครื่องหมาย dot ตามชื่อตัวแปร
  - เช่น “mystring”.upper() จะได้ผลลัพธ์เป็น MYSTRING

Function specification

- กำหนด input requirements ของฟังก์ชัน
  - The requirements are:
    - จำนวน argument ที่ต้องส่งให้ฟังก์ชัน
    - ลำดับของ argument ที่ต้องส่งให้ฟังก์ชัน

```
>>> len(“Python 3”)
8
```

len() ต้องการ 1 argument. จากนั้นมันจะคำนวณความยาวของ argument นี้ แล้วส่งผลลัพธ์กลับมา

Getting some help

- เราสามารถเปิดดูคำแนะนำการใช้งานฟังก์ชันเบื้องต้นโดยเรียกใช้ฟังก์ชัน help()

```
>>>
>>> help(len)
```

```
Help on built-in function len in module buil
tins:

len(...)
    len(object)

    Return the number of items of a sequence
 or collection.
~
~
```

ฟังก์ชันบางตัวอาจไม่ return ค่า

- ฟังก์ชันที่มีการส่งผลลัพธ์กลับ (สามารถ assign ค่าผลลัพธ์ให้กับตัวแปรได้) เรียกว่า fruitful functions
- ฟังก์ชันที่ไม่มีการส่งผลลัพธ์กลับ เรียกว่า void functions

```
>>> a = print('b')
b
>>> print(a)
None

>>> a = log(2)
>>> print(a)
>>> 0.6931471805599453
```

Module - Definition

- โมดูล คือไฟล์ .py ที่เก็บคำสั่งและนิยามเพื่อการเรียกใช้
- โดยปกติ โมดูล เป็นเสมือนกล่องที่เก็บ
  - นิยามค่าคงที่ต่างๆ Constants
  - ฟังก์ชันต่างๆ
- Example
  - A ‘math’ module contains a set of mathematical functions.
    - exp(), log(), ….
  - It also defines some useful constants in mathematics.
    - π

Using a module [1/2]

- เราสามารถเรียกใช้ฟังก์ชันใน Module โดยใช้

  from module_name **import** function_name

```
>>> from math import log
>>> log(2)
```

- หากเราอยาก Import ทุกๆฟังก์ชันในโมดูลนั้นให้ใช้เครื่องหมาย *

```
>>> from math import *
```

Using a module [2/2]

- หากต้องการ Import และเปลี่ยนชื่อ function จากโมดูลเป็นชื่อที่เราต้องการ สามารถใช้ keyword **as**

```
>>> from math import log as mylog
>>> mylog(2)
```

Making use of the return values

- Storing
  - เรียกฟังก์ชันแล้วเก็บผลลัพธ์ที่ได้ไว้ในตัวแปรก่อน

Example

```
>>> e1 = exp(1)
>>> e2 = exp(2)
>>> e3 = e1 + e2
```

- On-the-fly
  - ไม่เก็บไว้ในตัวแปร แต่นำมาใช้เลยทันที

Example

```
>>>
>>> e3 = exp(1) + exp(2)
>>>
```

Functions composition

- เราสามารถเรียกฟังก์ชันซ้อนกันได้ด้วย
  - Computing
  - $x = \sqrt{\log(10)}$

```
>>> a = log(10)
>>> b = sqrt(a)
>>> print(b)
1.5174271293851465
```

```
>>> b = sqrt(log(10))
>>> print(b)
1.5174271293851465
```

Useful Functions

- Input
  - การใช้ input() function
- Output
  - การใช้ string modulo

Output using print

- ฟังก์ชัน print() ต้องการ argument 3 ตัว คือ
  - ลิสต์ของสิ่งที่ต้องการจะแสดงผล
  - เครื่องหมายที่เอาไว้ขั้นระหว่างการแสดงผล (default คือไม่ขั้นอะไร)
  - เครื่องหมายที่เอาไว้ปิดท้ายการแสดงผล (default คือการขึ้นบรรทัดใหม่)
- Function specification

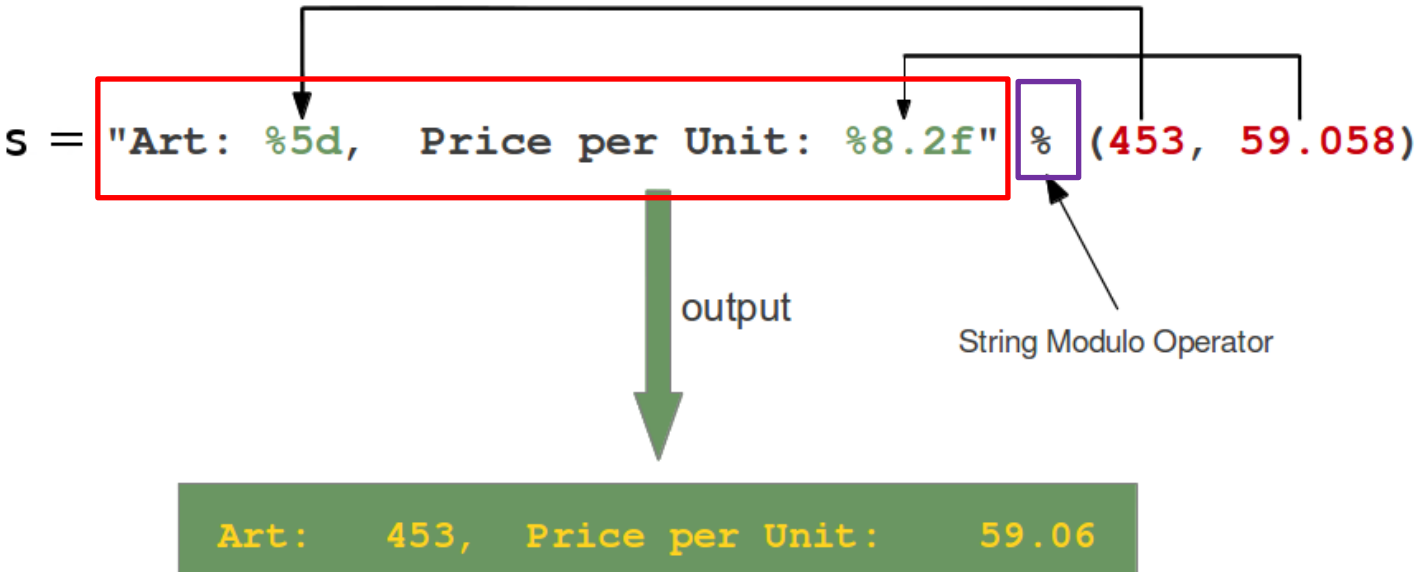
```
print(*objects, sep=' ', end='\n')
```

- Example

```
>>> print(“Python 3”)
Python 3
>>> print(1,4,5, sep=‘-’, end=‘\t’)
1-4-5   >>>
```

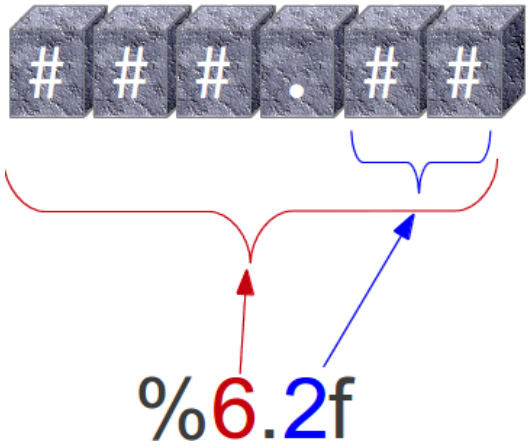
String modulo

- หากสิ่งที่เราต้องการแสดงผลเป็น string เราสามารถใช้เครื่องหมาย modulo (%) มาสร้าง string ที่ซับซ้อนขึ้นได้
- โดยการ แทรก placeholder ไว้ใน string ตามด้วย เครื่องหมาย modulo และตามด้วย ค่าที่จะนำมาเติมใน placeholder
- ค่าที่จะนำมาเติมจะอยู่ภายใต้วงเล็บ และมี คอมมาขั้นไว้



Common placeholder types

Type	Meaning
%d	Signed integer decimal.
%f	Floating point decimal format.
%a.bf	Floating point decimal format with precision marker
%c	Single character



a คือความยาวทั้งหมดที่อยากจะใช้งาน

- หากตัวเลข ยาวกว่า a ตัว Python ก็ยังจะพิมพ์ทั้งหมด

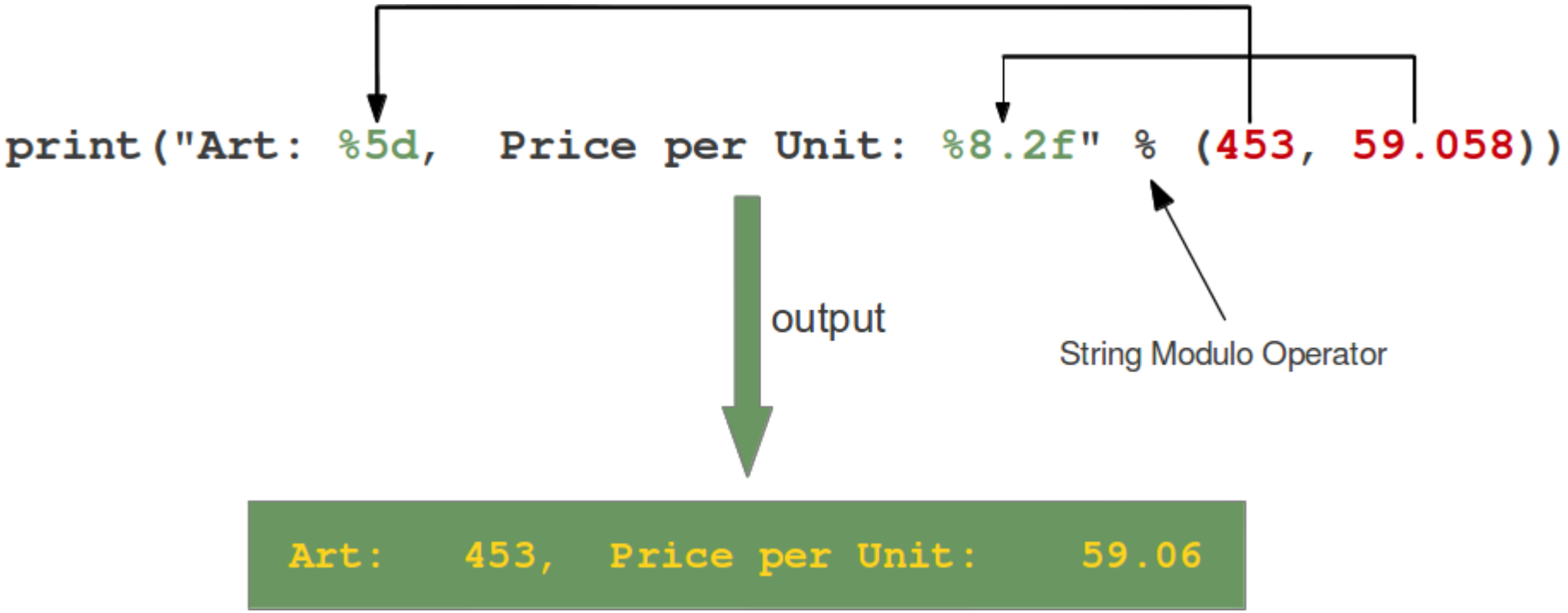
- หากตัวเลขยาวน้อยกว่า a ตัว Python จะพิมพ์ space

แต่จุดทศนิยม, b, จะต้องเป็นไปตามข้อกำหนดเท่านั้น คือ ตามความละเอียดที่กำหนด ในที่นี้คือสองจุด

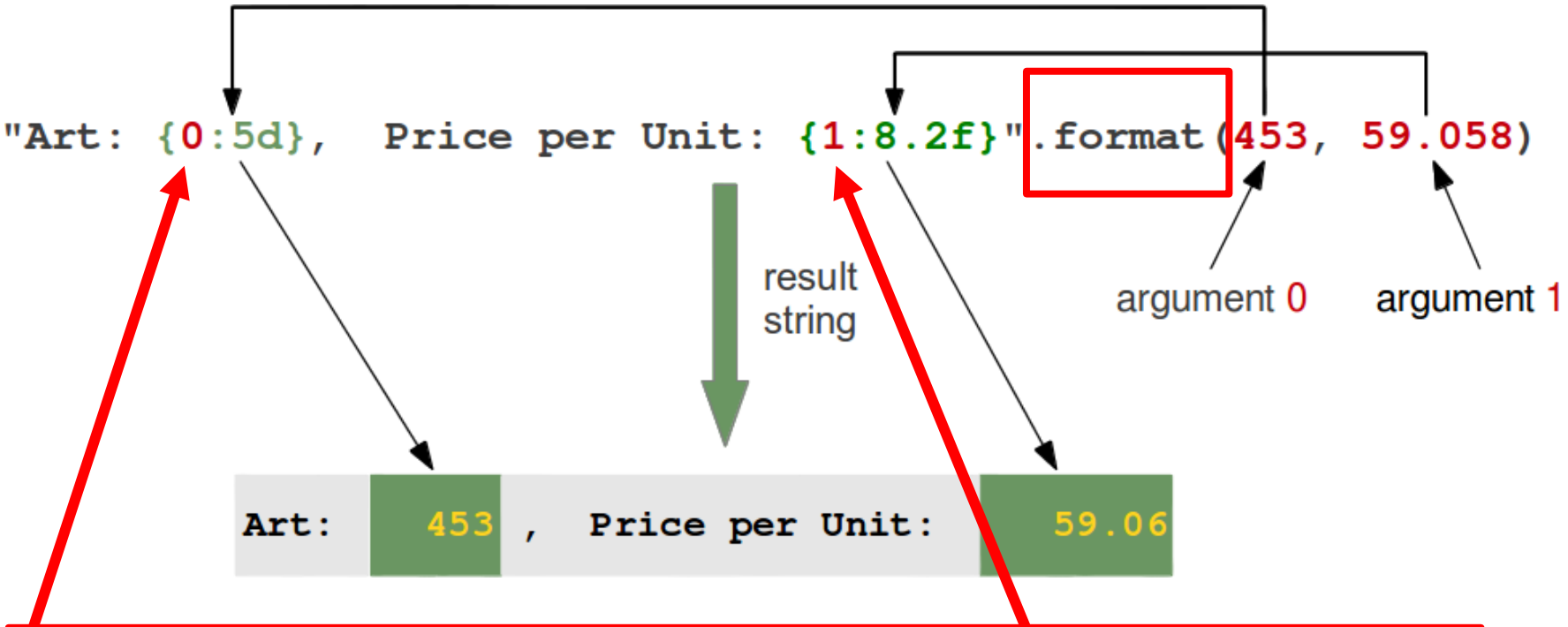
Examples

```
>>> print("Art: 453, Price: 59.058")
Art: 453, Price: 59.058
>>> s = "Art: 453, Price: 59.058"
>>> print(s)
Art: 453, Price: 59.058
>>> s = "Art: %d, Price: %f"%(453,59.058)
>>> print(s)
Art: 453, Price: 59.058000
>>> s = "Art: %7d, Price: %6.2f"%(453,59.058)
>>> print(s)
Art:     453, Price:  59.06
>>> p = 62.058
>>> s = "Art: %7d, Price: %6.2f"%(453,p)
>>> print(s)
Art:     453, Price:  62.06
>>>
```

String modulo on-the-fly



Using format() : a python way



เปลี่ยนจาก % เป็น {x: } เลข x แสดง ลำดับของ argument ที่จะเอาค่ามาแสดง

เช่น เดิมเป็น %5d เปลี่ยนเป็น {0:5d}

Examples of using format()

```
>>> "First argument: {0}, second one: {1}".format(47,11)
'First argument: 47, second one: 11'

>>> "Second argument: {1}, first one: {0}".format(47,11)
'Second argument: 11, first one: 47'

>>> "Second argument: {1:3d}, first one: {0:7.2f}".format(47.42,11)
'Second argument: 11, first one: 47.42'

>>> "First argument: {}, second one: {}".format(47,11)
'First argument: 47, second one: 11‘

>>> # arguments can be used more than once: ...
>>> "various precisions: {0:6.2f} or {0:6.3f}".format(1.4148)
'various precisions: 1.41 or 1.415'
```

Getting inputs using input()

- Can be done using input() function.
- You've already mastered it.
- Example

```
>>> ss = input("Enter your name: ")
Enter your name: Roger
>>> print(ss)
Roger
>>>
```

- Beware of converting String to int or float