



01076103, 01076104

Programming Fundamental

Programming Project

print, input, variable, math, f-string

คำสั่ง print



- คำสั่ง print ใช้ในการแสดงผล

```
main.py x
1 print("Hello Python world!")
2
```

Console Shell

Hello Python world!

- กรณีของการพิมพ์ข้อความ (String) เป็นข้อความ จะใช้เครื่องหมาย " หรือ ' ครอบเอาไว้ โดยเครื่องหมายเปิดและปิด ต้องเป็นเครื่องหมายเดียวกัน

```
main.py x
1 print("Hello Python world!")
2 print('Hello KMITL!')
3
```

Console Shell

Hello Python world!
Hello KMITL!




การเข้า Teams ของ Replit

- ให้เข้าที่ Link ต่อไปนี้เพื่อเข้า Teams
- <https://replit.com/teams/join/vxouahxgjxwgheovukeganjgnrkwdirz-CE-Python-65>
- ใช้ google และใช้ E-Mail สถาบัน




การเข้า Teams ของ Replit



- จะพบหน้าจอดังนี้ (ให้เลือก Start Project ดำเนินการและ Submit)



CE-Python-65
@CE-Python-65



Lesson 1 : Mathematic Operation

TITLE	DUE DATE	SUBMISSION
 Ex_1_1 ให้ใช้คำสั่ง print พิมพ์ข้อความ ต่อไปนี้ออกจอภาพ	—	Continue working
 Ex_1_2 จากโปรแกรมต่อไปนี้ ให้หาจุดที่ผิด และ แก้ไขให้ถูก	—	Start project

คำสั่ง print



- Exercise 1.1 ให้ใช้คำสั่ง print พิมพ์ข้อความ ต่อไปนี้ออกจอภาพ

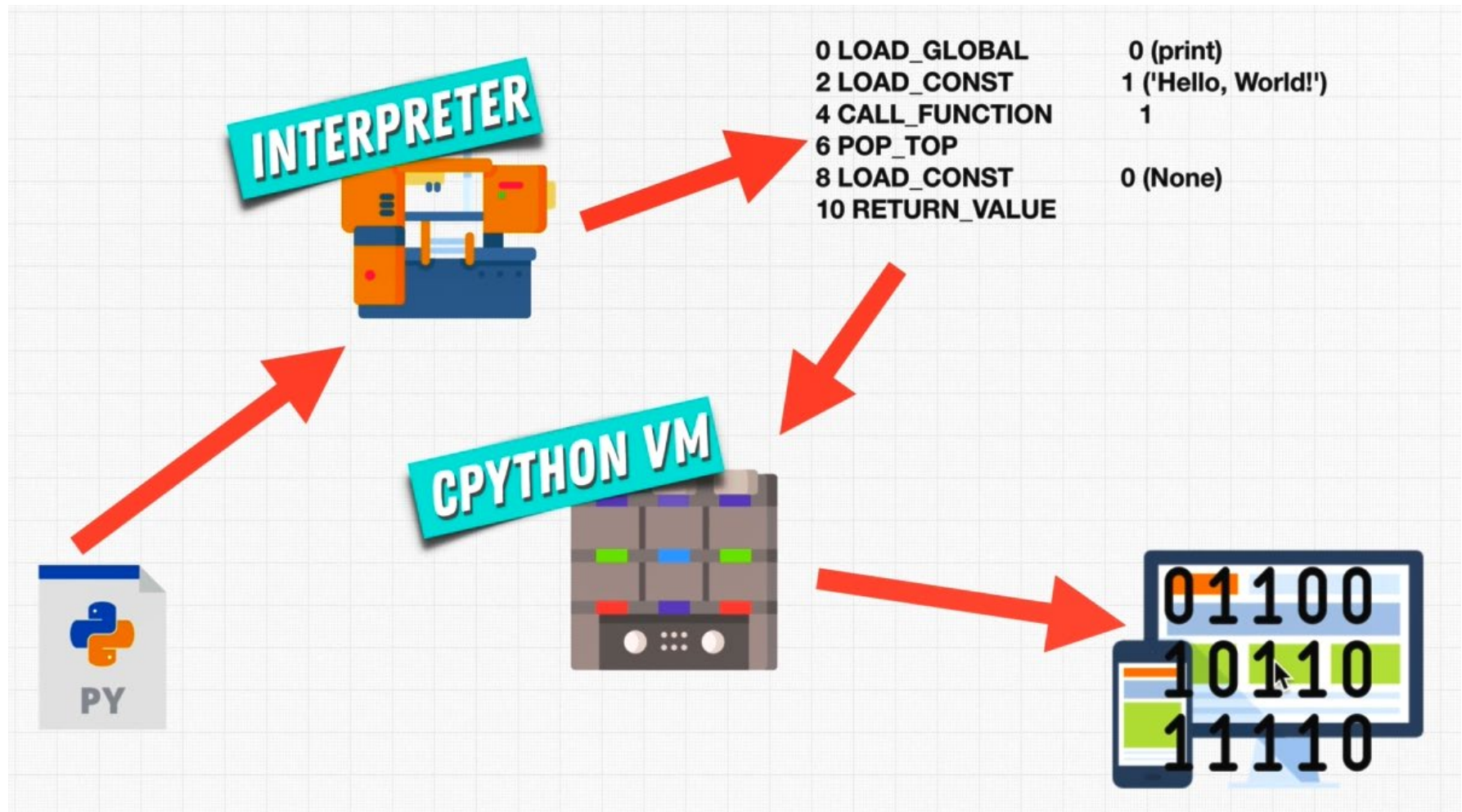
Python Print Function

The function is declared like this:

```
print('what to print')
```



การทำงานของ Interpreter





คำสั่ง print

- เราสามารถสั่งให้ขึ้นบรรทัดใหม่ใน string เดียวกัน โดยใช้อักขระพิเศษ \n (มีความหมาย = new line หรือ ขึ้นบรรทัดใหม่)

```
main.py x
1 print("Hello\nPython")
2
3
```

Console Shell

```
Hello
Python
>
```

- String สามารถเอามาต่อกันได้โดยเครื่องหมาย +

```
main.py x
1 print("Hello" + " " + "Thana")
2
3
```

Console Shell

```
Hello Thana
>
```

คำสั่ง print



- Exercise 1.2 จากโปรแกรมต่อไปนี้ ให้หาจุดที่ผิด และ แก้ไขให้ถูก

```
print(Day I - String Manipulation")
print("String Concatenation is done with the "+" sign.")
  print('e.g. print("Hello " + "world")')
print(("New lines can be created with a backslash and n.")
```




คำสั่ง print

- นอกจากการ + สำหรับ string แล้ว เรายังสามารถใช้เครื่องหมาย * ได้ด้วย

```
main.py x
1 print("KMITL "*5)
2
3
```

Console Shell

KMITL KMITL KMITL KMITL KMITL



Input

- ในโปรแกรมใดๆ ก็ตาม มักจะต้องมีการ Input ข้อมูล เพื่อนำมาประมวลผล
- รูปแบบการใช้งาน (Syntax)

```
input(prompt)
```

- โดย prompt คือ ข้อความที่แสดงว่าต้องการให้ Input ข้อมูลอะไร

```
main.py x
1 input("What's you name : ")
2 print("Hello " + input("What's you name : "))
3
```

```
Console Shell
What's you name : Thana
What's you name : Thana
Hello Thana
> 
```



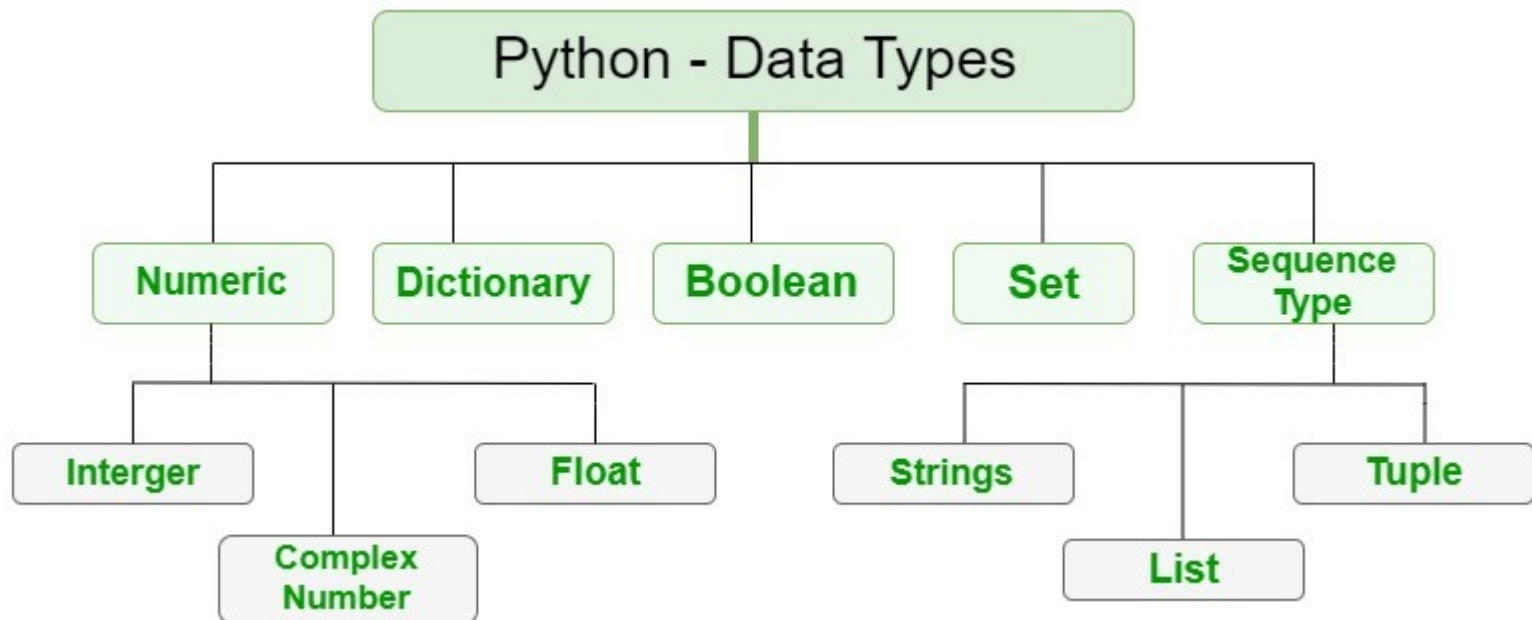
Input

- **Exercise 1.3** จากโปรแกรมใน slide ก่อนหน้า ให้ใช้ google หาวิธี โดยแทนที่จะพิมพ์ชื่อกลับมา ให้แสดงจำนวนตัวอักษรที่รับมาแทน
- การใช้ google ค้นหาข้อมูลเป็นทักษะที่สำคัญ “มาก” ของวิศวกรคอมพิวเตอร์
- วิธีทดสอบ ให้พิมพ์ Johnny ต้องได้ผลลัพธ์ = 5



ตัวแปร (Variable)

- บางครั้งเราจำเป็นต้องเก็บข้อมูลเอาไว้ชั่วคราวเพื่อประมวลผลต่อไป เราจะใช้สิ่งที่เรียกว่าตัวแปร
- ข้อมูลและตัวแปรในภาษา Python มีประเภทต่างๆ ดังนี้





ตัวแปร (Variable)

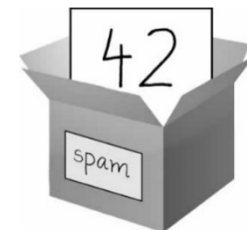
- เราสามารถเอาข้อมูลกำหนดค่าให้ตัวแปรได้

```
main.py x
1 name = "Peter"
2 print(name)
3 name = "John"
4 print(name)
```

Console Shell

Peter
John
> |

- การกำหนดค่าให้กับตัวแปร จะใช้เครื่องหมาย =
- จะเห็นว่าสามารถนำข้อมูลมากำหนดค่าให้ตัวแปร name ได้ ตามที่ต้องการ
- ให้มองว่าตัวแปรคล้ายกับกล่องเก็บของ ที่มีชื่อเรียก





ตัวแปร (Variable)

- จากโปรแกรมนี้จะเห็นว่าเราสามารถรับข้อมูลมาใส่ตัวแปร และดำเนินการกับข้อมูลนั้นได้

```
main.py x
1 s1 = "Hello"
2 s2 = "World"
3 s3 = s1 + s2
4 print(s3)
5 print("Hello" + "World")

Console Shell
HelloWorld
HelloWorld
>
```



ตัวแปร (Variable)

- นอกจากการรับข้อมูลเป็นตัวอักษร เรายังรับข้อมูลเป็นตัวเลขได้

```
main.py x
1 age = input('Enter your age: ')
2 print('You Entered - ' + age)
3
```

Console Shell

Enter your age: 28
You Entered - 28
>

- แต่สิ่งที่รับเข้ามา ยังไม่ใช่ตัวเลข เช่น หากเขียนโปรแกรมดังนี้ จะทำงานไม่ถูก

```
main.py x
1 in1 = input('Enter First number : ')
2 in2 = input('Enter Second number : ')
3 sum = in1 + in2
4 print(sum)
```

Console Shell

Enter First number : 12
Enter Second number : 34
1234
>

- เนื่องจากข้อมูลที่ได้รับมาเป็นตัวอักษร เมื่อนำมาบวกกัน คือ เอามาต่อกัน ไม่ได้เกิด
การคำนวณ



ตัวแปร (Variable)

- เราสามารถตรวจสอบชนิดของตัวแปรได้โดยใช้คำสั่ง `type`

```
main.py x
1 in1 = input('Enter First number : ')
2 in2 = input('Enter Second number : ')
3
4 print(type(in1))
5 int1 = int(in1)
6 print(type(int1))
```

Console Shell

```
Enter First number : 12
Enter Second number : 34
<class 'str'>
<class 'int'>
>
```

- และสามารถเปลี่ยนชนิดของข้อมูล (Type Casting) จาก `str` เป็น `int` โดยใช้คำสั่ง `int` และหากจะแปลงจาก `int` เป็น `str` จะใช้คำสั่ง `str`



ตัวแปร (Variable)

- ดังนั้นโปรแกรมก่อนหน้านี้ต้องเขียนแบบนี้ จึงจะทำงานได้ถูกต้อง

```
main.py x
1 in1 = int(input('Enter First number : '))
2 in2 = int(input('Enter Second number : '))
3 sum = in1 + in2
4 print(sum)
5
```

Console Shell

```
Enter First number : 12
Enter Second number : 23
35
> 
```

- จะเห็นว่าเมื่อรับข้อมูลโดยคำสั่ง input จากนั้นจะถูกคำสั่ง int แปลงให้เป็นข้อมูลประเภท int จึงสามารถนำมาบวกกันได้
- ประเภทของข้อมูลเป็นสิ่งที่ต้องระวังในการเขียนโปรแกรมภาษา python



ตัวแปร (Variable)

- ภาษา python เป็นภาษาที่ชนิดข้อมูลของตัวแปรไม่แน่นอน สามารถเปลี่ยนแปลงได้ บางครั้งเรียกว่า duck typing หรือ dynamic typing ซึ่งตรงข้ามกับ Strong Typing หรือ Static Typing

```
main.py x
1 str = "hello"
2 print(str)
3 print(type(str))
4 str = 20
5 print(str)
6 print(type(str))
7
```

```
hello
<class 'str'>
20
<class 'int'>
> 
```



ตัวแปร (Variable)

- **Exercise 1.4** จากโปรแกรมต่อไปนี้ ให้แก้ไขโปรแกรมโดยให้รับชื่อใส่ตัวแปรแล้วค่อยนำมาใช้งาน

main.py ×

```
1 print("Hello" + input("What's you name : "))  
2
```



ตัวแปร (Variable)

- Exercise 1.5 จาก
โปรแกรมต่อไปนี้ ให้เขียน
โปรแกรมที่แสดงค่าของ a
และ b สลับกัน เช่น ตอน
รับ a = 3, b = 5 ก็ให้แสดง
a = 5, b = 3
โดยเขียนในบรรทัดที่ 8-11

```
main.py x
1  #-----
2  # 🚫 Don't change the code below 👉
3  a = input("a: ")
4  b = input("b: ")
5  # 🚫 Don't change the code above 👈
6  #####
7  #Write your code below this line 👉
8
9
10
11
12 #Write your code above this line 👈
13 #####
14 # 🚫 Don't change the code below 👉
15 print("a: " + a)
16 print("b: " + b)
```



Variable Type

- แม้ชนิดของตัวแปรจะมีมากมาย แต่ในเบื้องต้น เราจะทำความรู้จักกับตัวแปรเพียง 4 ชนิดก่อน
 - Str ใช้เก็บข้อความหรือตัวอักษร (String)
 - Int ใช้เก็บตัวเลขที่เป็นจำนวนเต็ม (Integer)
 - Float ใช้เก็บตัวเลขแบบทศนิยม (Floating Point)
 - Bool ใช้เก็บ True หรือ False (Boolean)

```
main.py x
1 a = "hello"      #! String
2 print(a)
3 print(type(a))
4
5 a = 10           #! integer
6 print(a)
7 print(type(a))
8
9 a = 10.0         #! Float
10 print(a)
11 print(type(a))
12
13 a = True        #! boolean
14 print(a)
15 print(type(a))
```

```
hello
<class 'str'>
10
<class 'int'>
10.0
<class 'float'>
True
<class 'bool'>
>
```



Variable Type

- ให้ตอบว่า แต่ละบรรทัดต่อไปนี้เป็นชนิดข้อมูลชนิดใด

main.py x

```
1 print(123+456)
2
3 print(3.14)
4
5 print(1==1)
6 print(1==2)
7 print(True)
8
```



ตัวแปร (Variable)

- การใช้ตัวแปร ทำให้สะดวกต่อการใช้งาน สามารถนำไปใช้ได้หลายครั้ง
- การกระทำระหว่าง Int กับ Float จะได้ผลลัพธ์เป็น Float
- เราสามารถ assign ค่าให้กับตัวแปรหลายตัวในบรรทัดเดียวกันได้

```
main.py x
1 a = 1; b = 2.0
2 c = a + b
3 print(c)
4
5 a = 1 < 2
6 print(a)
7
8 a,b = 5,6
9 print(a + b, a * b)
10
11 a,b = 4,5
12 a,b = b,a
13 print(a,b)
14
15 a=b=c=1
16 print(a,b,c)
17
```

Console Shell

```
3.0
True
11 30
5 4
1 1 1
>
```



ตัวแปร (Variable)

- การตั้งชื่อตัวแปร จะมีหลักนิยมอยู่ 3 แบบ
 - Snake Case จะใช้คำภาษาอังกฤษเขียนเป็นตัวเล็กทั้งหมด แล้วคั่นด้วยเครื่องหมาย _ เช่น `sum_of_student_score` จะเห็นว่าใช้คำเขียนต่อกันคล้ายงู จึงเรียกว่า snake case
 - Camel Case จะใช้ตัวอักษรตัวใหญ่เฉพาะตัวอักษรแรกของคำ “ยกเว้น” คำแรก เช่น `sumOfStudentScore`
 - Pascal Case จะใช้ตัวอักษรตัวใหญ่เฉพาะตัวอักษรแรกของคำ โดยไม่มียกเว้น เช่น `SumOfStudentScore`
 - สำหรับวิชานี้ให้ใช้ Snake Case เพราะอ่านง่ายที่สุด
- ตัวอักษรตัวใหญ่เล็ก ถือว่าเป็นคนละตัวกัน (Case Sensitive)
- ตัวแรกต้องเป็นตัวอักษร หรือ _ เท่านั้น ตัวถัดมาเป็นตัวอักษร ตัวเลข และ _



ตัวแปร (Variable)

- ชื่อตัวแปรต้องไม่เป็น reserve word

None	True	False			
and	or	not			
if	else	elif			
for	while	break	continue	pass	
def	lambda	global	nonlocal	return	yield
del	in	is	assert	class	
try	except	finally	raise		
import	from	with	as		



ตัวแปร (Variable)

- Exercise 1.6 จงแก้ไขโปรแกรมต่อไปนี้ให้ถูกต้อง

main.py ×

```
1 num_char = len(input("Enter your name :"))  
2 print("Your name has " + num_char + " charecters.")  
3
```



ตัวแปร (Variable)

- **Exercise 1.7** ให้เขียนโปรแกรมรับตัวเลข 2 หลัก และเอาแต่ละหลักบวกกัน
เช่น input เป็น 35 ให้แสดงผล 8 ($3+5$)
 - คำแนะนำ ใช้การหารเข้ามาช่วย การหารโดย python จะใช้เครื่องหมาย /



ตัวแปร (Variable)

- เรื่องของ string จะมีความพิเศษกว่าตัวแปรชนิดอื่น
- String สามารถแยกใช้งานเป็นตัวๆ ได้ เช่น "Hello"[0] จะหมายถึงตัว H เพียงตัวเดียว (การใช้ต้องใช้เครื่องหมาย [] กำกับ โดยเริ่มนับจาก 0 เป็นต้นไป หมายถึงตัวอักษรตัวแรกของ String)
- มีอักขระพิเศษ นอกจาก \n ที่หมายถึงการขึ้นบรรทัดใหม่แล้ว ยังมี \t ซึ่งหมายถึง Tab

```
main.py x
1 print("\tPython")
2 print("Hello\nPython")
3 print("Languages:\n\tPython\n\tC\n\tJavaScript")
4
5
```

Console Shell

```
Python
Hello
Python
Languages:
Python
C
JavaScript
```



ตัวแปร (Variable)

- เครื่องหมายที่ครอบ String จะมี 3 แบบ
 - เครื่องหมาย ‘ หรือ “ จะใช้กรณีที่ข้างใน string ไม่มีเครื่องหมายที่เหมือนกับตัวเครื่องหมายคำพูดเอง เช่น

'One of Python's strengths is its diverse community.'
 - เครื่องหมาย ‘’ หรือ “” สามารถใช้ครอบทุกข้อความได้หมด

```
print(''I'm "Python".''')  
print("""I'm "Python".""" )
```



การคำนวณทางคณิตศาสตร์

- ภาษา Programming จะมีส่วนคำนวณทางคณิตศาสตร์เสมอ สำหรับภาษา Python จะมีตัวดำเนินการ (Operator) ดังนี้
 - การบวก (+) เช่น $a + b$
 - การลบ (-) เช่น $a - b$
 - การคูณ (*) เช่น $a * b$
 - การหาร (/) เช่น b / a
 - การหารเพื่อเอาเฉพาะเศษ (Modulus %) เช่น $b \% a$
 - การหารแบบเอาผล (ไม่เอาเศษ (Floor Division)) //) เช่น $9 // 2 = 4$
 - การยกกำลัง (**) เช่น $a ** b$



การคำนวณทางคณิตศาสตร์

- ข้อที่ควรระวังอย่างหนึ่งของการคำนวณทางคณิตศาสตร์ คือ ชนิดข้อมูลของผลลัพธ์การคำนวณ ซึ่งอาจจะไม่ตรงกับข้อมูลที่น่ามากระทำกัน
- เช่น ข้อมูล int หารด้วย int จะได้ชนิดข้อมูลเป็น float
- ถือเป็นหลักง่ายๆ ก็ได้ว่า ถ้าข้อมูลชนิด float ทำ operation กับข้อมูลชนิดอื่น ผลลัพธ์จะได้เป็น float แต่ถ้า int กับ int หารกันจะได้เป็น float เสมอ

```
main.py x
1 print(6 / 3)
2 print(2.0 ** 3)
3 print(5 % 2)
4 print(5.0 % 2)
5 print(5 // 2)
```

Console Shell

```
2.0
8.0
1
1.0
2
>
```



การคำนวณทางคณิตศาสตร์

- กรณีที่มีหลายการกระทำ ก็จะต้องมีการกระทำก่อนหรือหลังกัน จะใช้หลักที่เขียนเป็นตัวย่อว่า PEMDAS
 - P = Parentheses หรือวงเล็บ () คือถ้ามีวงเล็บ ให้ทำในวงเล็บก่อน ถ้าวางเล็บซ้อนกันหลายชั้น จะทำชั้นในสุดก่อน
 - E = Exponent จากนั้นถ้ามีการยกกำลัง จะทำยกกำลังก่อน
 - M = Multiplication, D = Division จากนั้นถ้ามีการคูณหรือหาร ก็จะทำก่อน โดยคูณและหาร มีความสำคัญเท่ากัน เจออันไหนก่อนทำอันนั้นก่อน
 - A = Add, S = Subtraction จากนั้นจึงทำบวกและลบเป็นลำดับสุดท้าย โดยบวกและลบ มีความสำคัญเท่ากัน เจออันไหนก่อนทำอันนั้นก่อน
 - ในกรณีที่ลำดับความสำคัญเท่ากัน การทำงานจะทำจากซ้ายไปขวา



การคำนวณทางคณิตศาสตร์

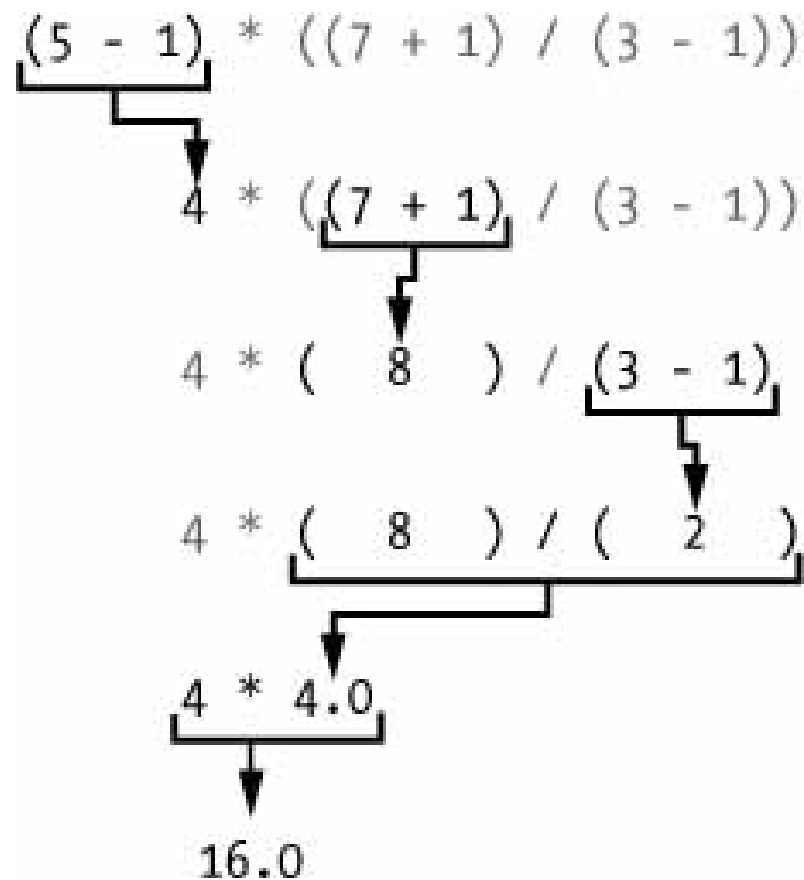
- Exercise 1.8 จากคำสั่งต่อไปนี้ ให้ใส่วงเล็บเพื่อให้ได้ผลลัพธ์เป็น 3.0

```
print(3 * 3 + 3 / 3 - 3)
```



การคำนวณทางคณิตศาสตร์

- ให้บอกลำดับการทำงานของ $(5 - 1) * ((7 + 1) / (3 - 1))$





การคำนวณทางคณิตศาสตร์

- ในการคำนวณทางคณิตศาสตร์ที่ให้ผลเป็นเลขทศนิยม (เช่นในบรรทัดแรก)
- ในกรณีที่เราต้องการเฉพาะข้อมูลบางส่วน สามารถทำได้หลายวิธี
 - ใช้ `int` เพื่อแปลงเป็นจำนวนเต็ม หรือใช้ `//`
 - ใช้คำสั่ง `round` เพื่อปัดเศษ โดยสามารถเลือกได้จะต้องการทศนิยมกี่ตำแหน่ง

```
main.py x Console Shell
1 print(8/3)
2 print(int(8/3))
3 print(8//3)
4 print(round(8/3))
5 print(round((8/3),2))
```

```
2.6666666666666665
2
2
3
2.67
>
```

การคำนวณทางคณิตศาสตร์



- Exercise 1.9 จงเขียนโปรแกรมหาBody Mass Index (BMI) (ไม่มีทศนิยม)

สูตรคำนวณ ดัชนีมวลกาย (BMI)

“น้ำหนักตัว (กิโลกรัม) ÷ ส่วนสูง (เมตร) ยกกำลัง 2”



$$\frac{\text{น้ำหนัก (กก.)}}{\text{ส่วนสูง (ม.}^2\text{)}}$$



การคำนวณทางคณิตศาสตร์

- จะมีเรื่องที่ชวนงงเล็กน้อยเกี่ยวกับเลขทศนิยม ดูตัวอย่างต่อไปนี้

```
~/14mathpy$ python
Python 3.8.12 (default, Aug 30 2021, 16:42:10)
[GCC 10.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> a = 0.2
>>> b = 0.1
>>> print(a+b)
0.30000000000000004
>>>
```

- จะเห็นว่า $0.2 + 0.1$ ไม่เท่ากับ 0.3
- การจะทำความเข้าใจเรื่องนี้ ต้องทราบก่อนว่าเลขที่อยู่หลังจุดทศนิยมมีการเก็บอย่างไร



การคำนวณทางคณิตศาสตร์

- ยกตัวอย่าง 0.125 จะมีค่าเท่ากับ $1/10 + 2/100 + 5/1000$
- แต่สำหรับในคอมพิวเตอร์จะมีการเก็บข้อมูลเป็นเลขฐาน 2 ดังนั้น $0.001_2 = 0/2 + 0/4 + 1/8$ หรือ $0+0+0.125$
- ดังนั้นการแทนค่าเลขทศนิยมฐาน 10 ด้วยฐาน 2 จึงยากมาก
- โดย 0.1_{10} ในเลขฐาน 2 จะเท่ากับ $1/16 (0.0625) + 1/32 (0.03125) + 0/64 + 0/128 + 1/256 (0.00390625) + \dots$ จนกว่าจะใกล้กับ 0.1 มากที่สุด โดยเลขที่ใกล้ 0.1 มากที่สุด คือ 0.100000001490116119384765625
- และเลขที่ใกล้ 0.2 มากที่สุด คือ 0.20000000298023223876953125
- ดังนั้นในภาษาโปรแกรม $0.1+0.2$ จึงไม่เท่ากับ 0.3 ถ้าเราทดลองเขียนว่า $.1 + .1 + .1 == .3$ ผลลัพธ์ จะได้เป็น False
- จึงควรระวังในการเปรียบเทียบทศนิยม



การคำนวณทางคณิตศาสตร์

- ในกรณีที่เราเขียนคำสั่งที่มีลักษณะเป็น $a = a + b$ เราสามารถเขียนแบบย่อได้
 - 1) $a = a + b$ สามารถเขียนเป็น $a += b$
 - 2) $a = a - b$ สามารถเขียนเป็น $a -= b$
 - 3) $a = a * b$ สามารถเขียนเป็น $a *= b$
 - 4) $a = a / b$ สามารถเขียนเป็น $a /= b$



การคำนวณทางคณิตศาสตร์

- Exercise 1.10 คำนวณปริมาตรของ 4 เหลี่ยมลูกบาศก์

main.py ×

```
1 width = float(input("Enter input width = "))
2 length = float(input("Enter input length = "))
3 height = float(input("Enter input height = "))
4
5 cube = 0      # ให้แก้ไขบรรทัดนี้
6
7 print("Volume of Cube : ",cube)
8
```




การคำนวณทางคณิตศาสตร์

- Exercise 1.11 จงเขียนโปรแกรมคำนวณภาษีมูลค่าเพิ่ม

main.py ×

```
1 Income = float(input("Enter your income (Baht): "))
2 VAT = float(input("Enter your VAT (%): "))
3 Tax =
4 print("Your Tax is ", Tax, "Baht")
```



การคำนวณทางคณิตศาสตร์

- **Exercise 1.12** จงเขียนโปรแกรมแปลงค่าจากองศาเซลเซียสไปเป็นองศาฟาเรนไฮต์
 - สูตรในการเปลี่ยนค่าจากองศาเซลเซียสไปเป็นองศาฟาเรนไฮต์มีดังนี้
 - $F = (9/5) * C + 32$
 - รับข้อมูลองศาเซลเซียสเป็นจำนวนจริง
 - แสดงผลตัวเลขจำนวนจริง เป็นองศาฟาเรนไฮต์
 - ตัวอย่าง 39.85 เซลเซียส = 103.73 ฟาเรนไฮต์



การคำนวณทางคณิตศาสตร์

- ฟังก์ชันทางคณิตศาสตร์อื่นๆ ที่น่าสนใจ
 - `abs(x)` ใช้ในการหา absolute value ของ `x`
 - `sqrt(x)` ใช้ในการหา square root ของ `x`
 - `floor(x)` ใช้ในการหาตัวเลข `int` ที่มากที่สุดที่น้อยกว่า `x`
 - `ceil(x)` ใช้ในการหาตัวเลข `int` ที่น้อยที่สุดที่มากกว่า `x`
- ฟังก์ชันทางคณิตศาสตร์อื่นๆ ดูได้ที่

<https://www.programiz.com/python-programming/modules/math>



การจัดรูปแบบการแสดงผล

- ในบางครั้งเราต้องการแสดงผลในรูปแบบที่เราต้องการ เราสามารถจัดรูปแบบการแสดงผลโดยใช้ `.format` เช่น

```
main.py ×
1 txt = "For only {price:.2f} baths!"
2 print(txt.format(price = 49))
```

Console Shell

```
For only 49.00 baths!
> 
```

- รูปแบบการใช้งานเป็นดังนี้

`string.format(value1, value2...)`

- ส่วนที่ต้องการให้จัดรูปแบบจะอยู่ข้างใน `{ }` และส่วนข้อมูลที่ต้องการแสดงผลและจัดรูปแบบจะอยู่ในวงเล็บของ `.format()`



การจัดรูปแบบการแสดงผล

- การระบุการจับคู่ระหว่างรูปแบบ กับ ข้อมูล สามารถทำได้หลายรูปแบบดังนี้

```
main.py x
1 txt1 = "My name is {fname}, I'm {age}"\
2     .format(fname = "John", age = 36)
3 txt2 = "My name is {0}, I'm {1}"\
4     .format("John",36)
5 txt3 = "My name is {}, I'm {}"\
6     .format("John",36)
7 print(txt1)
8 print(txt2)
9 print(txt3)
```

Console Shell

```
My name is John, I'm 36
My name is John, I'm 36
My name is John, I'm 36
>
```



การจัดรูปแบบการแสดงผล

- ข้างในเครื่องหมาย { } เราสามารถใส่สัญลักษณ์จัดรูปแบบได้
 - :< จัดชิดซ้าย
 - :> จัดชิดขวา
 - :^ จัดตรงกลาง

```
main.py x
1 txt = "We have {:<8} chickens."
2 print(txt.format(49))
3 txt = "We have {:>8} chickens."
4 print(txt.format(49))
5 txt = "We have {:^8} chickens."
6 print(txt.format(49))
```

Console Shell

```
We have 49 chickens.
We have      49 chickens.
We have      49 chickens.
> 
```

- รูปแบบอื่น https://www.w3schools.com/python/ref_string_format.asp



การจัดรูปแบบการแสดงผล โดยใช้ f-string

- การจัดรูปแบบอีกแบบหนึ่งเรียกว่า f-string เป็นการรวมเอาส่วนการจัดรูปแบบและส่วนของข้อมูลเอาไว้ด้วยกัน

```
main.py x
1 first_name = "ada"
2 last_name = "lovelace"
3 full_name = f"{first_name} {last_name}"
4 message = f"Hello, {full_name}!"
5 print(message)
6 print(f"Hello, {full_name}!")
7
```

```
Console Shell
Hello, ada lovelace!
Hello, ada lovelace!
>
```



การจัดรูปแบบการแสดงผล โดยใช้ f-string

- Exercise Ex 1.13 สมมติว่าเราจะอยู่จนถึงอายุ 90 ปี
- คำนวณหา ว่าเราจะมีเวลาเหลืออีกเท่าไร ให้ใช้ f-string ประกอบด้วย

Input:

Enter age : 56

Output:

You have 12410 days, 1768 weeks, and 408 months left.



การจัดรูปแบบการแสดงผล โดยไม่ใช่ f-string

- ในการพิมพ์เราสามารถจัดรูปแบบการพิมพ์ได้ เพื่อความสวยงาม

main.py x

```
1 print("{}{}".format("th", "Thailand"))
2 print("{:5}|{:15}|".format("th", "Thailand")) # align left
3 print("{:<5}|{:<15}|".format("th", "Thailand")) # align left
4 print("{:>5}|{:>15}|".format("th", "Thailand")) # align right
5 print("{:*>5}|{:>15}|".format("th", "Thailand")) # align right
6 print("{:^5}|{:^15}|".format("th", "Thailand")) # align center
7
```



Console

Shell

```
thThailand
th  |Thailand  |
th  |Thailand  |
    th|        Thailand|
***th|-----Thailand|
    th |        Thailand  |
> []
```

— :15 = พื้นที่แสดง 15 ช่อง

— < = ซิดซ้าย

— > = ซิดขวา

— ^ = ตรงกลาง



การจัดรูปแบบการแสดงผล โดยใช้ f-string

- ในการพิมพ์เราสามารถจัดรูปแบบการพิมพ์ได้ เพื่อความสวยงาม

```
main.py x
1 print(f'{"th"}{"Thailand"}')
2 print(f'{"th":5}|{"Thailand":15}|')
3 print(f'{"th":<5}|{"Thailand":<15}|')
4 print(f'{"th":>5}|{"Thailand":>15}|')
5 print(f'{"th":*>5}|{"Thailand":->15}|')
6 print(f'{"th":^5}|{"Thailand":^15}|')
7
```

Console Shell

```
thThailand
th |Thailand |
th |Thailand |
   th|      Thailand|
***th|-----Thailand|
   th |      Thailand |
>
```



การจัดรูปแบบการแสดงผล โดยใช้ print

- คำสั่ง print เองก็มีส่วนที่ช่วยจัดรูปแบบได้บ้าง ดังนี้

```
main.py x
1 print("Hello", "how are you?", sep="---")
2
3 a = 5
4 print("a = ", a, sep='00000', end='\n\n\n')
5 print("a = ", a, sep='0', end='\n')
6
7 str1 = "Hello"
8 str2 = "python"
9 print(str1, end='')
10 print(str2)
```

Console Shell

```
Hello---how are you?
a = 000005

a = 05
Hellopython
> |
```

- sep คือ ใช้อย่างอื่นแทนช่องว่างระหว่างการแสดงผล
- end คือ ปิดท้ายบรรทัดด้วยอะไร ถ้าไม่บอก คือ การขึ้นบรรทัดใหม่



การรับ Input ครึ่งละหลายค่า

- เราสามารถรับ Input ครึ่งละหลายค่าได้

```
main.py x
1 a,b,c = input("abc : ").split()
2 # อ่าน string ต้องป้อน 3 ค่าคั่นด้วย space
3 print(a,b,c)
4
```

Console Shell

```
abc : 10 20 30
10 20 30
>
```

- การทำงานของคำสั่งข้างต้น คือ Input จะรับมาเป็น string ยาวๆ จากนั้นคำสั่ง `split()` จะทำหน้าที่แยกข้อมูลเป็นส่วนๆ (ข้อมูลต้องคั่นด้วยช่องว่าง) และจึงกำหนดค่าให้กับ `a`, `b`, `c` ตามลำดับ



การรับ Input ครึ่งละหลายค่า

- Exercise 1.14 รับข้อมูล 3 ตัว a, b กับ c คั่นด้วยช่องว่าง
a และ b เป็นตัวอักษรตัวเดียว ส่วน c เป็นจำนวนเต็ม
ให้แสดงตัวอักษรใน a ต่อกับตัวอักษรใน b แล้วตามด้วย c
แล้วตามด้วยตัวอักษร a+b จำนวน c ชุด
เช่น a b 5 -> ab5ababababab



For your attention