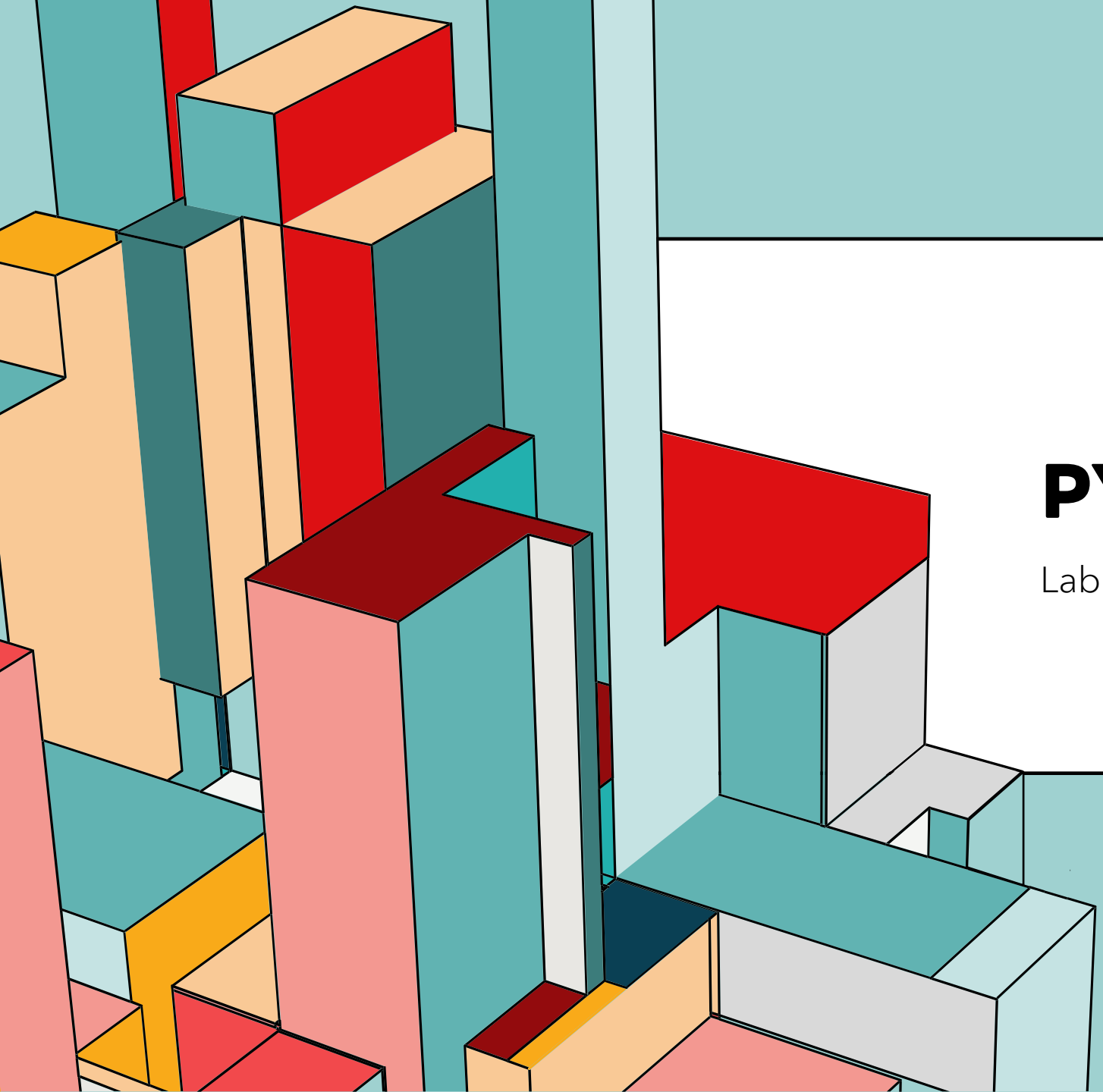
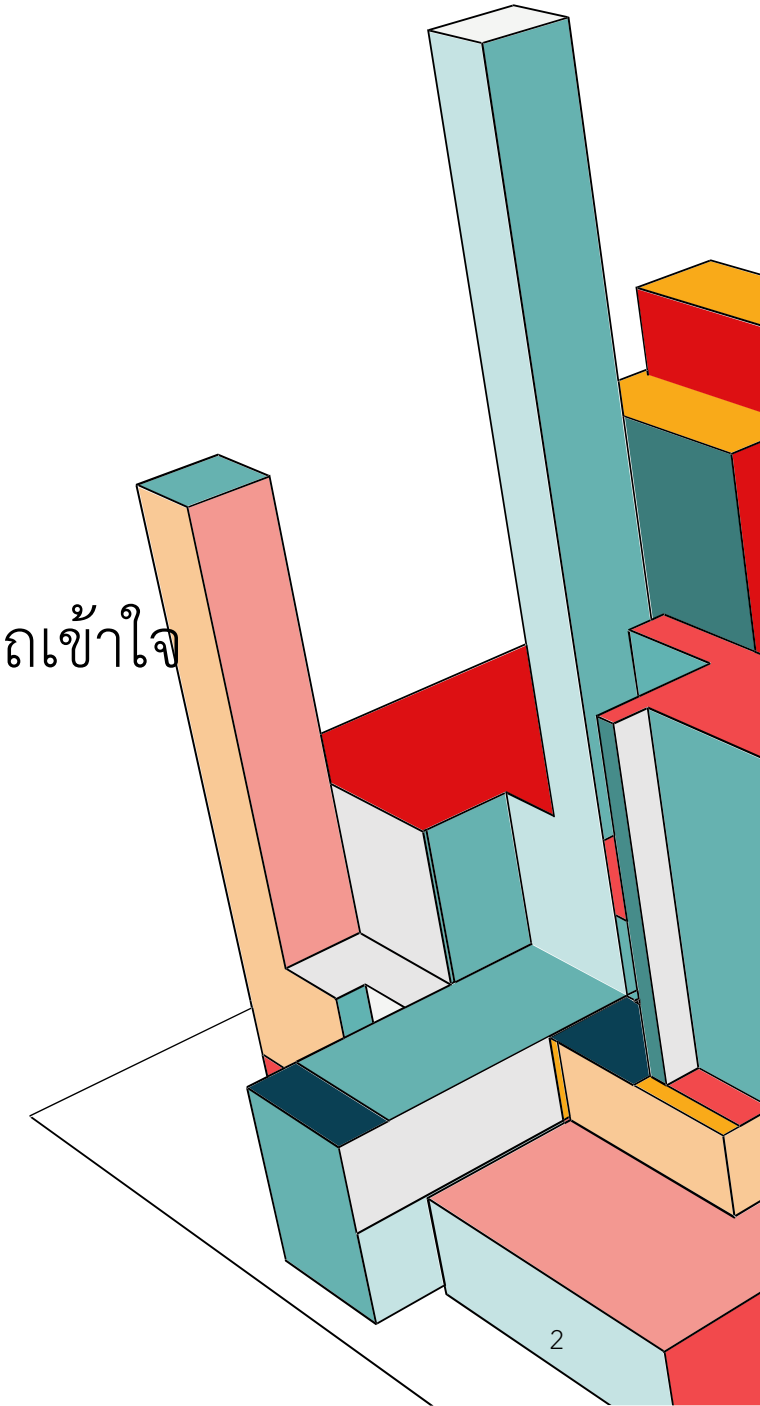


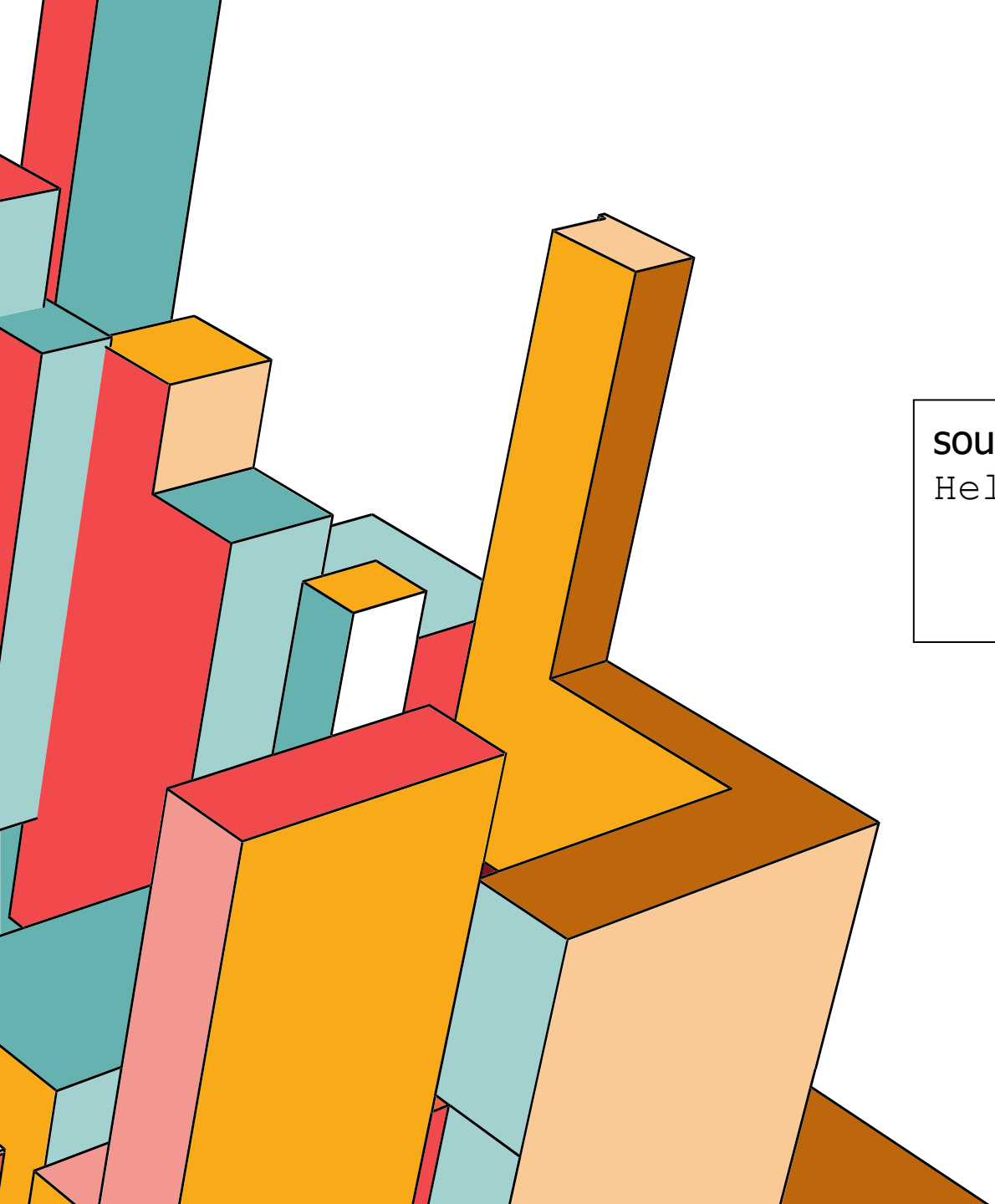
PYTHON PRIMER

Lab Documentation (Lab 1)

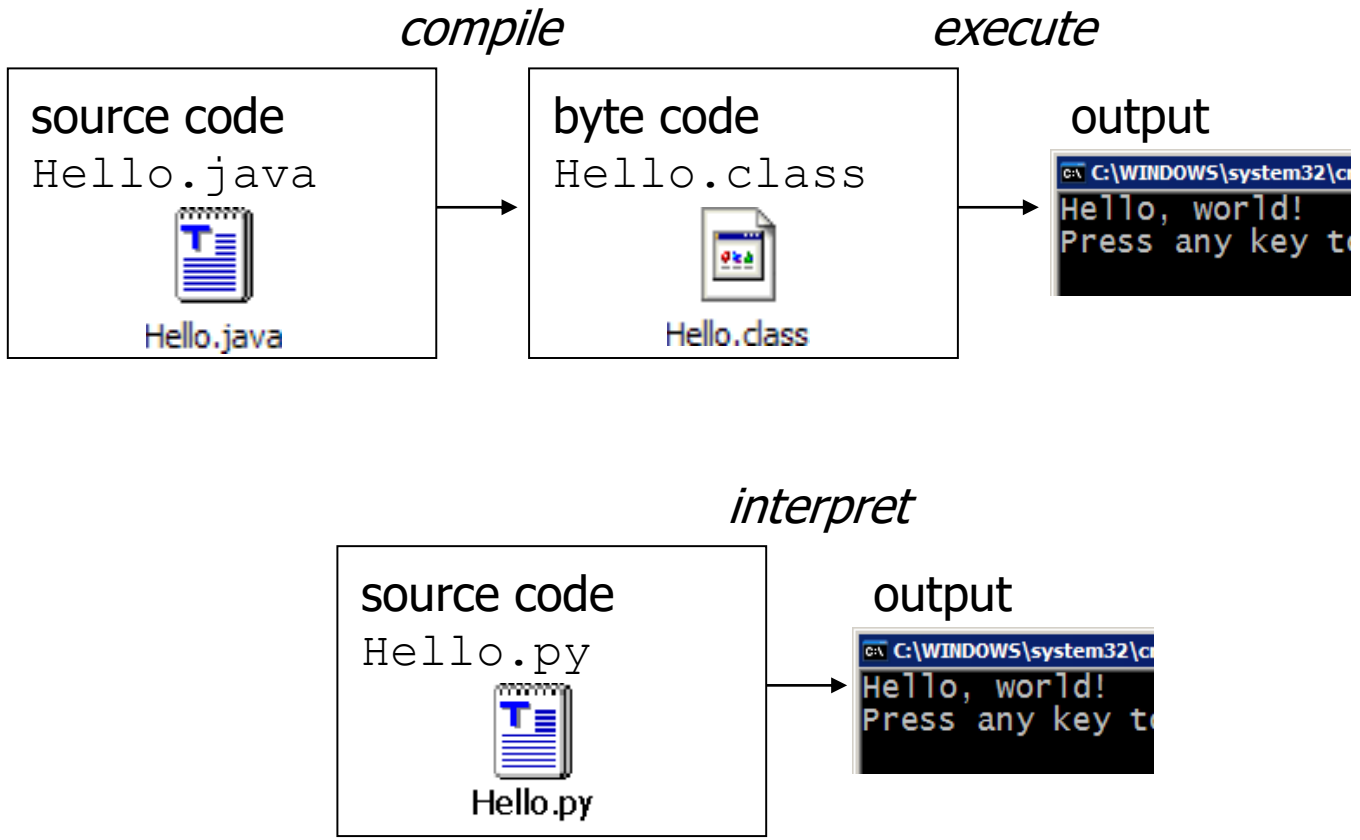
PYTHON

- พัฒนาโดย Guido van Rossum
- เป็นภาษาระดับสูง (High-Level Language)
- ใช้ Interpreter ในการแปลภาษาไพธอนให้คอมพิวเตอร์สามารถเข้าใจได้
- Python website: http://www.python.org/
- ปัจจุบัน Python version 3.11.1





COMPILER & INTERPRETER



VARIABLE IN PYTHON

- ตัวแปร ใช้เก็บค่าข้อมูลต่างๆ ซึ่งสามารถนำไปประมวลผลต่อได้
- การตั้งชื่อตัวแปร
  - ขึ้นต้นด้วยตัวอักษรภาษาอังกฤษ หรือ _
  - ประกอบด้วย ตัวอักษรภาษาอังกฤษ ตัวเลข หรือ _
  - ชื่อต้องไม่ซ้ำกับคำสงวน (Reserved words)
  - ตัวอักษรตัวพิมพ์เล็กและใหญ่ จะถูกมองเป็นคนละตัวกัน เช่น ตัวแปร result กับ Result จะถือเป็นตัวแปรคนละตัว
- ตัวอย่างชื่อตัวแปร เช่น name, result1, answer_2 เป็นต้น
- Python เวอร์ชั่น 3 สามารถกำหนดชื่อตัวแปรเป็นภาษาไทยได้

RESERVED WORDS

ห้ามนำมาใช้ตั้งชื่อตัวแปร

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

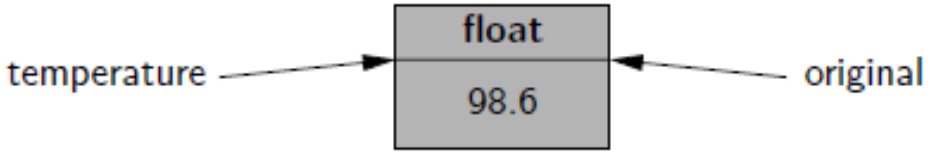
VARIABLE ASSIGNMENT

- เราสามารถสร้างตัวแปร เพื่อเก็บข้อมูลประเภทต่างๆ ได้
- ไม่จำเป็นต้องกำหนดชนิดข้อมูลให้กับตัวแปร
- ตัวอย่าง
  - name = “John”
    - ตัวแปร name เป็นตัวแปรชนิดสตริง และเก็บข้อมูลค่า “John”
  - age = 25
    - ตัวแปร age เป็นตัวแปรชนิดจำนวนเต็ม และเก็บข้อมูลค่า 25

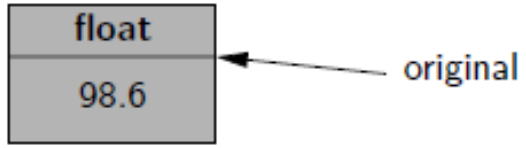
ASSIGNMENT STATEMENT

- temperature = 98.6
  (variable) *(object)*

- ตัวแปรแต่ละตัวจะมีการเชื่อมโยงถึงตำแหน่งในหน่วยความจำ (memory) ของอ๊อบเจ็คข้อมูลที่อ้างอิงถึง
- ไพธอนเป็นภาษาที่สนับสนุนให้สามารถเปลี่ยนแปลงชนิดข้อมูลของตัวแปรได้ (dynamic typing)
- original = temperature



- temperature = temperature + 5.0



DATA TYPES IN PYTHON

Text Type:	str
Numeric Types:	int, float, complex
Sequence Types:	list, tuple, range
Mapping Type:	dict
Set Types:	set, frozenset
Boolean Type:	bool
Binary Types:	bytes, bytearray, memoryview
None Type:	NoneType

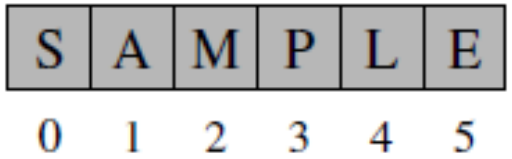
ที่มา : https://www.w3schools.com/python/python_datatypes.asp

DATA TYPES IN PYTHON

- จำนวนเต็ม (Integer)
  - ได้ทั้งจำนวนเต็มบวกและจำนวนเต็มลบ
  - เลขฐานต่างๆ
    - เลขฐานสอง จะเขียนขึ้นต้นด้วย 0b เช่น 0b1001010, 0b1010 เป็นต้น
    - เลขฐานแปด จะเขียนขึ้นต้นด้วย 0o เช่น 0o367, 0o1034 เป็นต้น
    - เลขฐานสิบ เช่น 124, 567800 เป็นต้น
    - เลขฐานสิบหก จะเขียนขึ้นต้นด้วย 0x เช่น 0x934ab, 0xab2c เป็นต้น

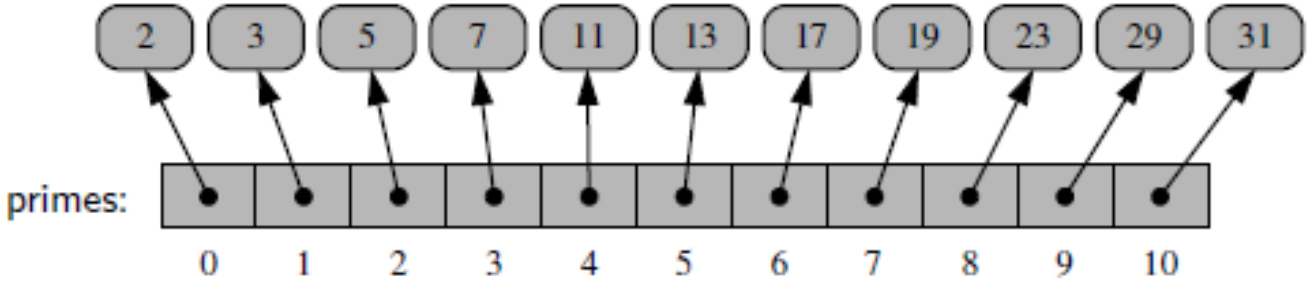
DATA TYPES IN PYTHON

- จำนวนทศนิยม (Float)
  - 10.25, 456.78
  - 4.25e4 ($4.25x10^{4}$), 3.75e-2 ($3.75x10^{-2}$)
- จำนวนเชิงซ้อน (Complex)
  - 3+4j, 4-5j
- ข้อมูลเชิงตรรกะ (Boolean)
  - True, False
- สายอักขระ หรือ สตริง (String)
  - “Python”, ‘Language’, “SAMPLE”
  - “Don’t worry”, ‘Don\’t worry’



DATA TYPES IN PYTHON

- ลิสต์ (List) – ใช้เก็บชุดข้อมูลที่เป็นลำดับ สามารถเก็บข้อมูลที่มีชนิดต่างกันได้ และเข้าถึงข้อมูลผ่าน index number
  - list1 = [‘red’, ‘green’, ‘blue’]
    - list1[0] -> ‘red’
  - list(‘hello’) -> [‘h’, ‘e’, ‘l’, ‘l’, ‘o’]
  - prime = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31]



DATA TYPES IN PYTHON

- เซ็ต (set) - ใช้เก็บชุดข้อมูลที่ไม่สำคัญและไม่มีลำดับ
- สมาชิกในเซ็ต สามารถเป็นชนิดข้อมูลที่ไม่สามารถเปลี่ยนแปลงได้เท่านั้น เช่น เลขจำนวนเต็ม (integer) เลขทศนิยม (float) และสตริง เป็นต้น
  - set() -> empty set
  - set1 = {‘red’, ‘green’, ‘blue’}
  - set(‘hello’) -> {‘h’, ‘e’, ‘l’, ‘o’}

DATA TYPES IN PYTHON

- ดิกชันนารี (dictionary) – ใช้เก็บชุดข้อมูลในรูปแบบของ Key และ Value โดยใช้ค่า Key เป็น index สำหรับเข้าถึงข้อมูล Value
- dict1 = { ‘ga’ : ‘Irish’ , ‘de’ : ‘German’ }
  - dict1[‘ga’] -> ‘Irish’
  - dict1[‘de’] -> ‘German’

การแสดงข้อมูลออกทางจอภาพ

- ฟังก์ชั่น print(ข้อความหรือออบเจ็คต่างๆ)
- ตัวอย่างการใช้คำสั่ง

```
print()
print(“Hello World”)
print(“Hello World”, “Hello Python”)
print(“My name is ” + ‘John’)
print(“I am”, 25, “years”, “old” )
print(“The result is”, 25+9/3)
```

การรับข้อมูลจากคีย์บอร์ด

- ฟังก์ชั่น input()
- สามารถรับค่าตัวเลขจำนวนเต็ม เลขทศนิยม หรือข้อความได้ โดยไพธอนจะแปลงข้อมูลนั้นให้เป็นสตริงทั้งหมด
- มีตัวแปรมารับค่าข้อมูล
- ตัวอย่างการใช้คำสั่ง

```
first_name = input(“Please enter your first name: ”)
last_name = input(“Please enter your last name: ”)
print(“Thank you,”, first_name, last_name)
```

การแปลงชนิดข้อมูล

- ทดลองรันโปรแกรมดังนี้

```
num = input(“Enter number : ”)
print(num+50)
```

  - num เป็นตัวแปรสตริง เอาไปบวกกับจำนวนเต็ม 50 ไม่ได้
  - ต้องทำการแปลงข้อมูลสตริงให้เป็นจำนวนเต็มก่อนนำไปบวกกับ 50
  - คำสั่ง int(ข้อมูลที่ต้องการแปลง) เป็นการแปลงข้อมูลให้เป็นจำนวนเต็ม

```
num_str = input(“Enter number : ”)
num = int(num_str)
print(num+50)
```

COMMENT

- ใช้เขียนคำอธิบายโปรแกรม โดยไม่นำมาใช้ประมวลผล
- สามารถเขียนได้ 2 แบบ
  - # : ใช้เขียนคำอธิบายเพียงบรรทัดเดียว
  - “““...””” หรือ ‘‘‘...’’’ : ใช้เขียนคำอธิบายได้หลายบรรทัด
- ตัวอย่าง
  - # This is a comment.
  - ‘‘‘ This is the first comment.

    This is the second comment.

    ’’’

EXPRESSION

- เป็นการดำเนินการทางคณิตศาสตร์ หรือทำการเปรียบเทียบหาค่าต่างๆ ประกอบด้วย
  - ตัวถูกดำเนินการ (Operand)
  - ตัวดำเนินการ (Operator)
- ตัวอย่าง
  - result = a + b * c
    - Operand : a, b, c
    - Operator : +, *

ตัวดำเนินการทางคณิตศาสตร์

Operator	Meaning	Example	Result
==	equals	`1 + 1 == 2`	`True`
!=	does not equal	`3.2 != 2.5`	`True`
<	less than	`10 < 5`	`False`
>	greater than	`10 > 5`	`True`
<=	less than or equal to	`126 <= 100`	`False`
>=	greater than or equal to	`5.0 >= 5.0`	`True`

Operator	Example	Result
`and`	`9 != 6 and 2 < 3`	`True`
`or`	`2 == 3 or -1 < 5`	`True`
`not`	`not 7 > 0`	`False`

ตัวดำเนินการทางคณิตศาสตร์

- ทดลองรันโปรแกรมดังนี้

```
a = 3; b=4;
print(“a+b is”, a+b)
print(“a-b is”, a-b)
print(“a*b is”, a*b)
print(“a/b is”, a/b)
print(“a%b is”, a%b)
print(“a**b is”, a**b)
print(“a//b is”, a//b)
```

ตัวดำเนินการเปรียบเทียบ

- ทดลองรันโปรแกรมดังนี้

```
a = 3; b=4;
print("a==b is", a==b)
print("a!=b is", a!=b)
print("a<b is", a<b)
print("a<=b is", a<=b)
print("a>b is", a>b)
print("a>=b is", a>=b)
```

ตัวดำเนินการตรรกศาสตร์

- ตัวดำเนินการ and

op1	op2	op1 and op2
true	true	true
true	false	false
false	true	false
false	false	false

ตัวดำเนินการตรรกศาสตร์

- ตัวดำเนินการ or

op1	op2	op1 or op2
true	true	true
true	false	true
false	true	true
false	false	false

ตัวดำเนินการตรรกศาสตร์

- ตัวดำเนินการ not

op	not op
true	false
false	true

ตัวดำเนินการตรรกศาสตร์

- ทดลองรันโปรแกรมดังนี้

```
a = 3; b=4;
print("a==b or a!=b is", a==b or a!=b)
print("a==b and a!=b is", a==b and a!=b)
print("not (a==b) is", not (a==b))
```

OPERATORS AND PRECEDENCE

- Logical operators

not	unary negation
and	conditional and
or	conditional or

- Equality Operators

is	same identity
is not	different identity
==	equivalent
!=	not equivalent

OPERATORS AND PRECEDENCE

- Comparison Operators

<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to

- Arithmetic Operators

+	addition
–	subtraction
*	multiplication
/	true division
//	integer division
%	the modulo operator

OPERATORS AND PRECEDENCE

- Bitwise Operators

~	bitwise complement (prefix unary operator)
&	bitwise and
|	bitwise or
^	bitwise exclusive-or
<<	shift bits left, filling in with zeros
>>	shift bits right, filling in with sign bit

- Sequence Operators

s[j]	element at index *j*
s[start:stop]	slice including indices [start,stop)
s[start:stop:step]	slice including indices start, start + step, start + 2*step, . . . , up to but not equalling or stop
s + t	concatenation of sequences
k * s	shorthand for s + s + s + ... (k times)
val **in** s	containment check
val **not in** s	non-containment check

OPERATORS AND PRECEDENCE

- Operators for Sets

key **in** s	containment check
key **not in** s	non-containment check
s1 == s2	s1 is equivalent to s2
s1 != s2	s1 is not equivalent to s2
s1 <= s2	s1 is subset of s2
s1 < s2	s1 is proper subset of s2
s1 >= s2	s1 is superset of s2
s1 > s2	s1 is proper superset of s2
s1 | s2	the union of s1 and s2
s1 & s2	the intersection of s1 and s2
s1 − s2	the set of elements in s1 but not s2
s1 ^ s2	the set of elements in precisely one of s1 or s2

OPERATORS AND PRECEDENCE

- Operators for Dictionaries

d[key]	value associated with given key
d[key] = value	set (or reset) the value associated with given key
del d[key]	remove key and its associated value from dictionary
key **in** d	containment check
key **not in** d	non-containment check
d1 == d2	d1 is equivalent to d2
d1 != d2	d1 is not equivalent to d2

OPERATOR PRECEDENCE

- Ordered from highest precedence to lowest precedence.
- 5 + 2 * 3

Operator Precedence		
	Type	**Symbols**
1	member access	expr.member
2	function/method calls container subscripts/slices	expr(...) expr[...]
3	exponentiation	**
4	unary operators	+expr, −expr, ~expr
5	multiplication, division	*, /, //, %
6	addition, subtraction	+, −
7	bitwise shifting	<<, >>
8	bitwise-and	&
9	bitwise-xor	^
10	bitwise-or	|
11	comparisons containment	**is**, **is not**, ==, !=, <, <=, >, >= **in**, **not in**
12	logical-not	**not** expr
13	logical-and	**and**
14	logical-or	**or**
15	conditional	val1 **if** cond **else** val2
16	assignments	=, +=, −=, *=, etc.

ฟังก์ชันทางคณิตศาสตร์

Command name	Description
`abs(`***value***`)`	absolute value
`ceil(`***value***`)`	rounds up
`cos(`***value***`)`	cosine, in radians
`floor(`***value***`)`	rounds down
`log(`***value***`)`	logarithm, base *e*
`log10(`***value***`)`	logarithm, base 10
`max(`***value1***`, `***value2***`)`	larger of two values
`min(`***value1***`, `***value2***`)`	smaller of two values
`round(`***value***`)`	nearest whole number
`sin(`***value***`)`	sine, in radians
`sqrt(`***value***`)`	square root

Constant	Description
`e`	2.7182818...
`pi`	3.1415926...

หากต้องการใช้ฟังก์ชั่นเหล่านี้ ให้ทำการเพิ่มคำสั่งด้านล่าง ที่บรรทัดบนสุดของโปรแกรม

```
from math import *
```

EXERCISE

- จงเขียนโปรแกรมคำนวณอายุ โดยทำการรับค่าปี พ.ศ. ที่เกิด ผ่านทางคีย์บอร์ด
  - ตัวอย่างการทำงาน

    กรุณากรอกปีเกิดของคุณ (พ.ศ.) : 2548

    ปัจจุบันคุณอายุ 18 ปี
- จงเขียนโปรแกรมคำนวณค่าพื้นที่สามเหลี่ยม โดยทำการรับค่าความยาวฐานและความสูงของสามเหลี่ยมผ่านทางคีย์บอร์ด
  - ตัวอย่างการทำงาน

    กรุณากรอกความยาวฐานสามเหลี่ยม : 12

    กรุณากรอกความสูงสามเหลี่ยม : 10

    สามเหลี่ยมมีพื้นที่ 60.0 ตารางหน่วย

EXERCISE

- จงเขียนโปรแกรมเพื่อเปลี่ยนอุณหภูมิจากฟาเรนไฮน (รับค่าผ่านคีย์บอร์ด) ให้เปนเซลเซียล โดยมีสูตรดังนี้

  °C = (5/9) * (°F - 32)

  - ตัวอย่างการทำงาน

    กรุณากรอกค่าองศาฟาเรนไฮน์: 50

    ค่าองศาเซลเซียส คือ 10.0

- จงเขียนโปรแกรมเพื่อแปลงขอมูลที่ปอนเขามาเปนฟุตและนิ้ว ให้เปนหนวยเมตรและเซนติเมตร (ค่าประมาณ 1 ฟุต = 12 นิ้ว = 30 ซม.)
  - ตัวอย่างการทำงาน

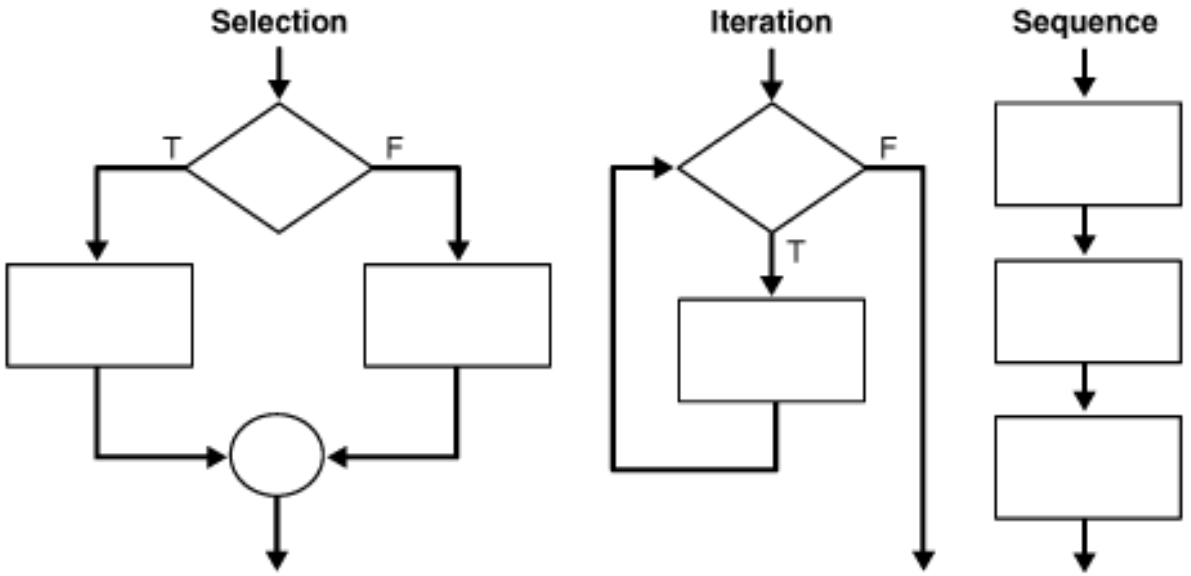
    กรุณากรอกค่าความยาวฟุต: 5

    กรุณากรอกค่าความยาวนิ้ว: 6

    แปลงค่าความยาว เท่ากับ 1 เมตร 65.0 เซนติเมตร

CONTROL FLOW

- โครงสร้างแบบตามลำดับ (Sequential Structure)
- โครงสร้างแบบเลือกทำ (Selection Structure)
- โครงสร้างแบบทำซ้ำ (Repetition Structure)



SEQUENTIAL STRUCTURE

- คำสั่งจะทำงานตามลำดับจากบนลงล่าง
- ตัวอย่าง

```
w = float(input("กรุณากรอกความยาวของสี่เหลี่ยม")) #คำสั่ง 1
h = float(input("กรุณากรอกความสูงของสี่เหลี่ยม")) #คำสั่ง 2
area = w*h #คำสั่ง 3
print("พื้นที่สี่เหลี่ยม เท่ากับ", area, "ตารางหน่วย") #คำสั่ง 4
```

- โปรแกรมจะไล่ทำงานทีละคำสั่ง เริ่มจากคำสั่งที่ 1 ไปจนถึงคำสั่งที่ 4 ตามลำดับ
- สังเกต คำสั่งจะถูกเขียนชิดซ้ายเหมือนกัน
- ทดลองเพิ่มช่องว่างหน้าคำสั่ง สังเกตผลลัพธ์ที่เกิดขึ้น

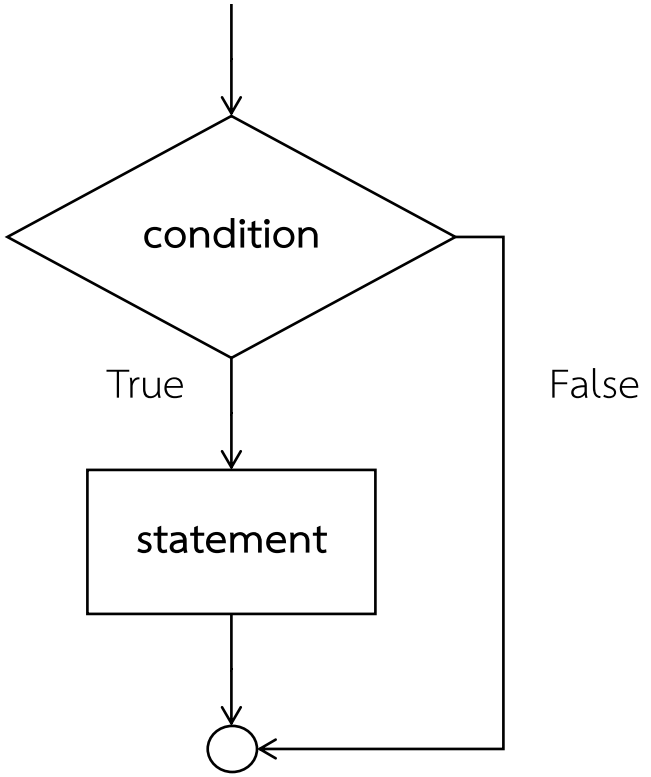
SELECTION STRUCTURE

```
if <condition>:
    <statements>
```

condition

```
x = 12
if x < 15:
    print("x is less than 15")
    print("Bye")
```

statements

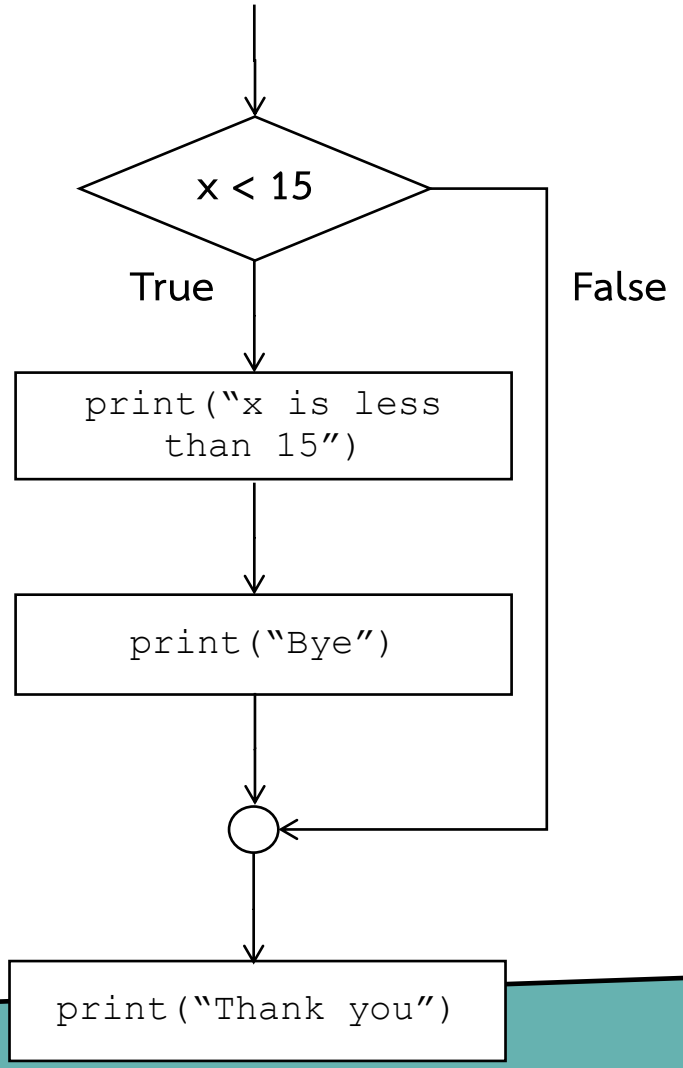


SELECTION STRUCTURE

- จงหาค่าผลลัพธ์ของโปรแกรมนี้

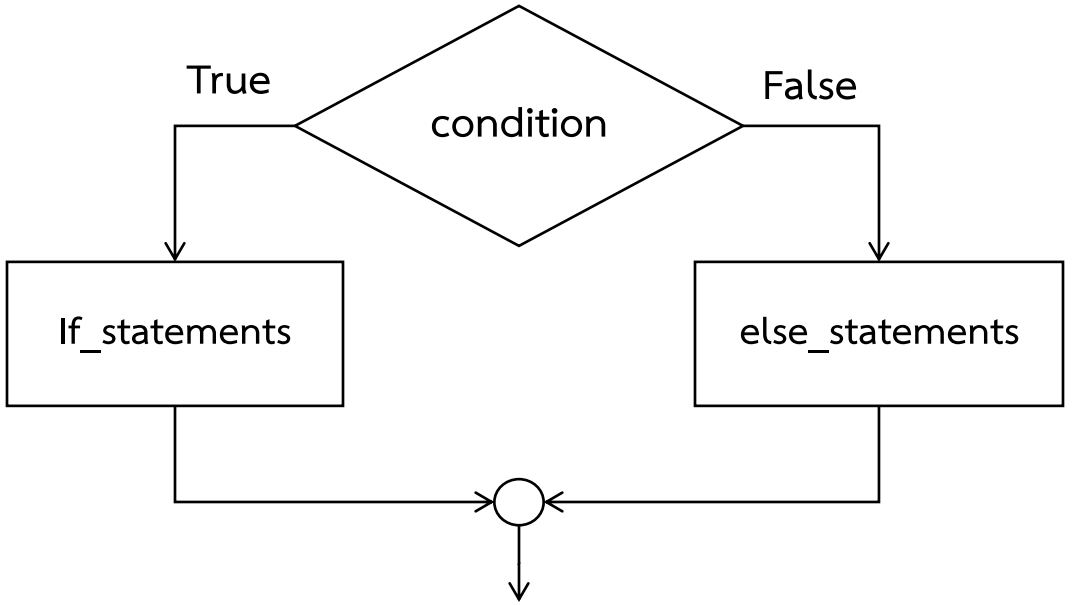
```
x = 12
if x < 15:
        print("x is less than 15")
        print("Bye")
print("Thank you")
```

- ถ้า x = 20 โปรแกรมจะแสดงผลลัพธ์อย่างไร



IF/ELSE STATEMENT

```
if <condition>:
      <if_statements>
else:
      <else_statements>
```



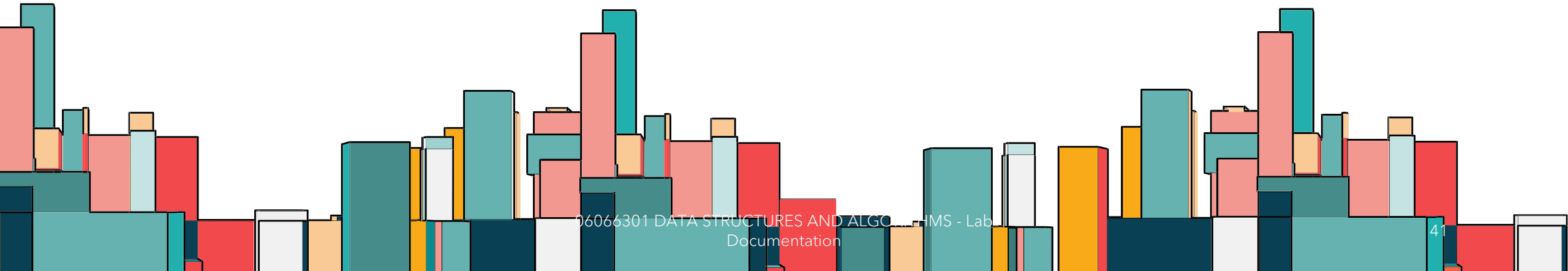
IF/ELSE STATEMENT

```
n = 5
if n == 10:
     print('n มีค่าเท่ากับ 10')
else:
     print('n ไม่ค่าไม่เท่ากับ 10')
```

```
money = 300
if money >= 350:
     print('You can buy this bag')
else:
     print('You don\'t have enough money to buy this bag')
```

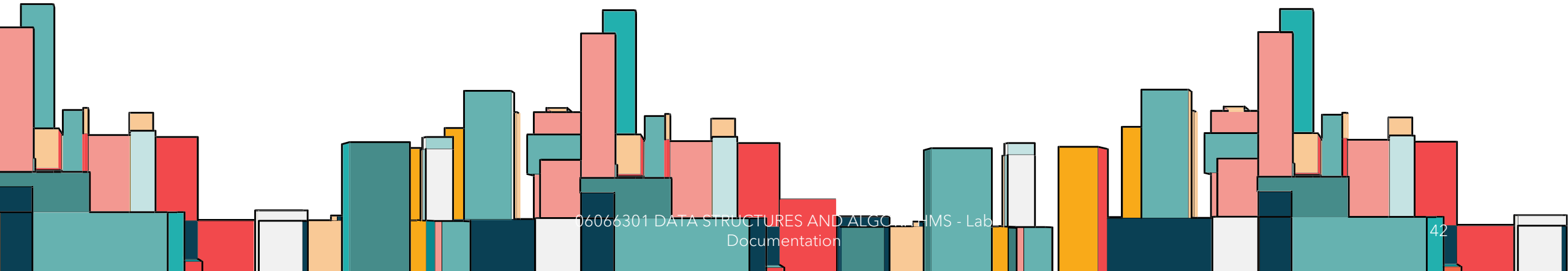
EXERCISE

- จงเขียนโปรแกรมคำนวณค่าพื้นที่สามเหลี่ยม โดยทำการรับค่าความยาวฐานและความสูงของสามเหลี่ยมผ่านทางคีย์บอร์ด
  - โดยมีการตรวจสอบว่า ค่าความยาวฐานและความสูง ต้องไม่เป็นค่าติดลบ
  - กรณีที่ป้อนค่าติดลบ ให้แสดงข้อความว่า
    - ค่าความยาวฐานและความสูงต้องเป็นจำนวนบวกเท่านั้น



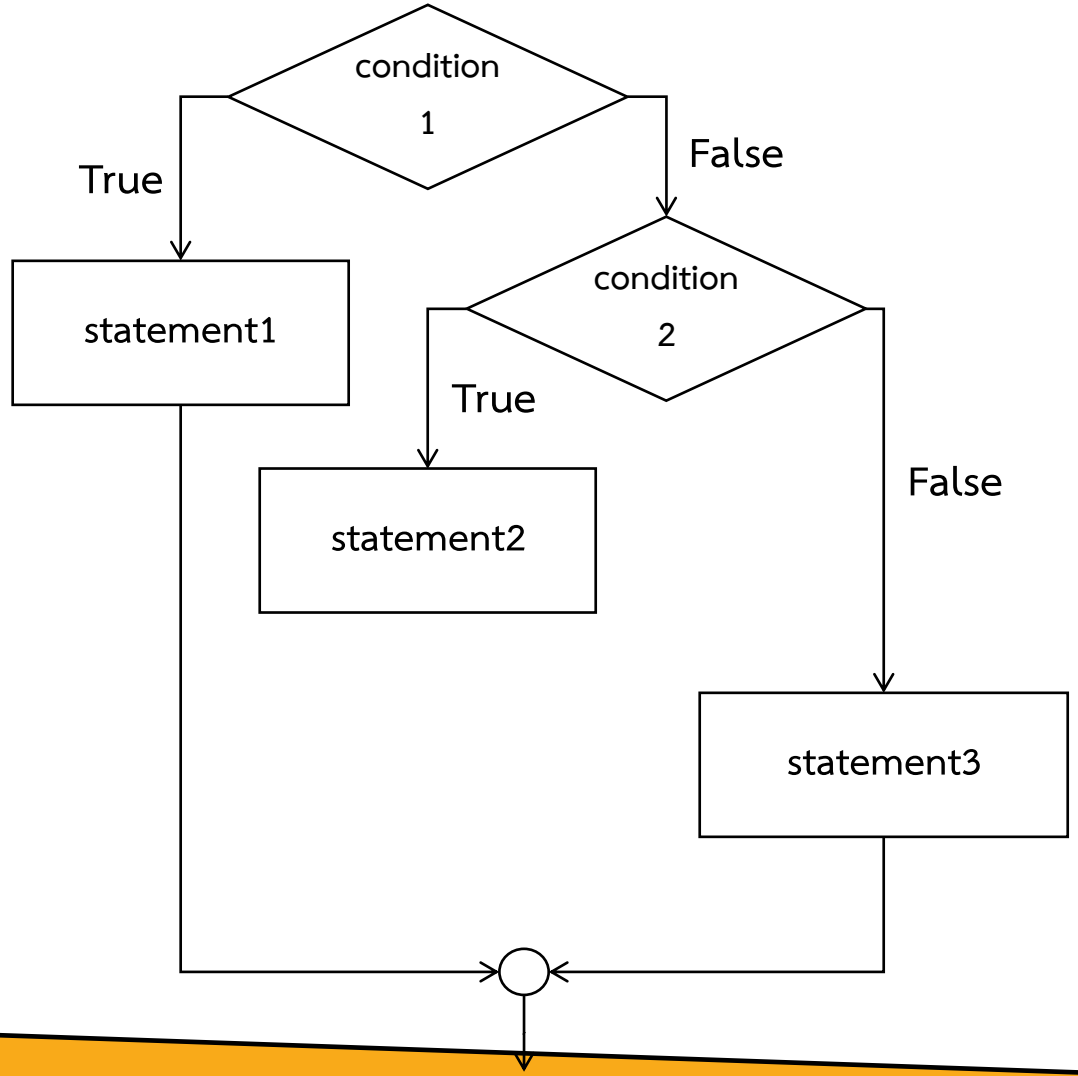
EXERCISE

- จงเขียนโปรแกรมตัดเกรด โดยทำการรับคะแนนผ่านทางคีย์บอร์ด
  - ถ้าคะแนน มากกว่าหรือเท่ากับ 50 คะแนน ให้แสดงข้อความ “Congrats!! You passed.”
  - ถ้าไม่ใช่ ให้แสดงข้อความ “Sorry!! You failed.”



IF-ELIF STATEMENT

```
if <condition1>:
        <statement1>
elif <condition2>:
        <statement2>
else :
        <statement3>
```



IF-ELIF STATEMENT

```
level = input('Enter level (1 - 4): ')

if level == '1':
      print('Easy')
elif level == '2':
      print('Medium')
elif level == '3':
      print('Hard')
elif level == '4':
      print('Expert')
else:
      print('Invalid level selected')
```

EXERCISE

- จงเขียนโปรแกรมตัดเกรด โดยทำการรับคะแนนผ่านทางคีย์บอร์ด และมีเงื่อนไข ดังนี้
  - คะแนนต้องมีค่าตั้งแต่ 0 ถึง 100 เท่านั้น ไม่เช่นนั้นจะแสดงข้อความ “Your score is not in range.”
  - ถ้าคะแนนเป็น 80 คะแนนขึ้นไป ให้แสดงข้อความ “You get A.”
  - ถ้าคะแนนเป็น 70 คะแนนขึ้นไป แต่น้อยกว่า 80 ให้แสดงข้อความ “You get B.”
  - ถ้าคะแนนเป็น 60 คะแนนขึ้นไป แต่น้อยกว่า 70 ให้แสดงข้อความ “You get C.”
  - ถ้าคะแนนเป็น 50 คะแนนขึ้นไป แต่น้อยกว่า 60 ให้แสดงข้อความ “You get D.”
  - ถ้าคะแนนน้อยกว่า 50 คะแนน ให้แสดงข้อความ “Sorry!! You get F.”

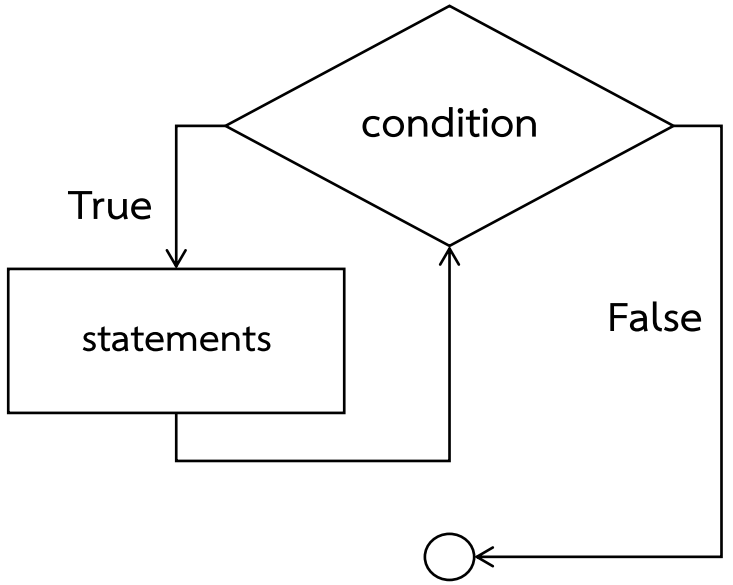
REPETITION STRUCTURE

- ใช้ในการสั่งให้ทำงานคำสั่งหรือชุดคำสั่งซ้ำๆ ตามเงื่อนไขที่กำหนด
  - การวนซ้ำโดยใช้ While Loop
  - การวนซ้ำโดยใช้ For Loop

WHILE STATEMENT

```
while <condition>:
      <statements>
```

```
i = 1
while i <= 10:
      print(i, end = ', ')
      i = i + 1
```



- การวนลูปจะเกิดขึ้นเมื่อเงื่อนไขเป็นจริง
- เหมาะกับการวนที่ขึ้นอยู่กับเงื่อนไข
- **<u>อย่าลืม</u> ควรมีคำสั่งที่ช่วยทำให้โปรแกรมหลุดออกจากการวนซ้ำด้วย เช่น การเปลี่ยนค่าตัวแปร**

EXAMPLE: WHILE STATEMENT

```
number = 1
while number < 200:
      print (number)
      number = number * 2
```

```
Output:
  1 2 4 8 16 32 64 128
```

EXERCISE

- จงปรับปรุงโปรแกรมตัดเกรดในข้อก่อนหน้านี้ ให้สามารถวนรับค่าคะแนน เพื่อแสดงผลเกรดไปเรื่อยๆ จนกว่าจะป้อนค่า -1 จึงจะหยุดทำงาน แล้วแสดงข้อความ “Thank you...”
- ตัวอย่าง

```
Please enter your score (Enter -1 for loop stopping): 65.9
You get C
Please enter your score (Enter -1 for loop stopping): 82.5
You get A
Please enter your score (Enter -1 for loop stopping): -1
Thank you...
```

FOR STATEMENT

- การวนลูปจะเกิดขึ้นเมื่อเงื่อนไขเป็นจริง
- เหมาะกับการวนที่ขึ้นอยู่กับจำนวน หรือช่วงของการวน (Range)

```
for variableName in groupOfValues:
        <statements>
```

```
for x in range(1, 6):
        print (x, 'squared is', x * x)
```

RANGE()

- เรามักจะใช้คำสั่ง for loop กับฟังก์ชัน range() ในการวนอ่านค่า
- range (ตัวเลขเริ่มต้น, ตัวเลขสุดท้าย, ค่าที่เปลี่ยนแปลงในลำดับของตัวเลข)

```
a = list(range(10))
b = list(range(1, 11))
c = list(range(0, 30, 5))
d = list(range(0, -10, -1))

print(a)
print(b)
print(c)
print(d)
```

EXAMPLE: FOR STATEMENT

```
for i in range(1, 11):
    print(i, end = ', ')
```

```
for i in range(10, 0, -1):
    print(i, end = ', ')
```

```
names = ['Mateo', 'John', 'Eric', 'Mark', 'Robert']
for i in range(len(names)):
    print(names[i], end = ', ')
```

EXERCISE

- จงเขียนโปรแกรมคำนวณค่า Factorial โดยรับค่าจำนวน Factorial ผ่านคีย์บอร์ด
- ตัวอย่าง

    Enter factorial number: 6

    6 ! = 720

EXERCISE

- จงเขียนโปรแกรมพิมพ์ * จำนวน n x n ตัว (n แถวๆ ละ n ตัว) โดยให้รับค่า n ผ่านคีย์บอร์ด
- ตัวอย่าง

```
Enter size: 5
* * * * *
* * * * *
* * * * *
* * * * *
* * * * *
```

EXERCISE

- จงเขียนโปรแกรมพิมพ์ * เป็นรูปสามเหลี่ยม ดังตัวอย่าง โดยให้รับค่าขนาดสามเหลี่ยมผ่านคีย์บอร์ด
- ตัวอย่าง

```
Enter size: 5
*
* *
* * *
* * * *
* * * * *
```

FUNCTION

- เราสามารถสร้างชุดคำสั่งไพธอนเพื่อทำงานอย่างใดอย่างหนึ่งให้ในรูปแบบฟังก์ชันได้ เพื่อลดความซ้ำซ้อนในการเขียนโปรแกรม และสามารถเรียกใช้ฟังก์ชันนั้นๆ ซ้ำได้
- รูปแบบการประกาศฟังก์ชัน

```
def function_name(args...):
    # statements


def function_name(args...):
    # statements
    return value
```

FUNCTION

- ตัวอย่างเช่น

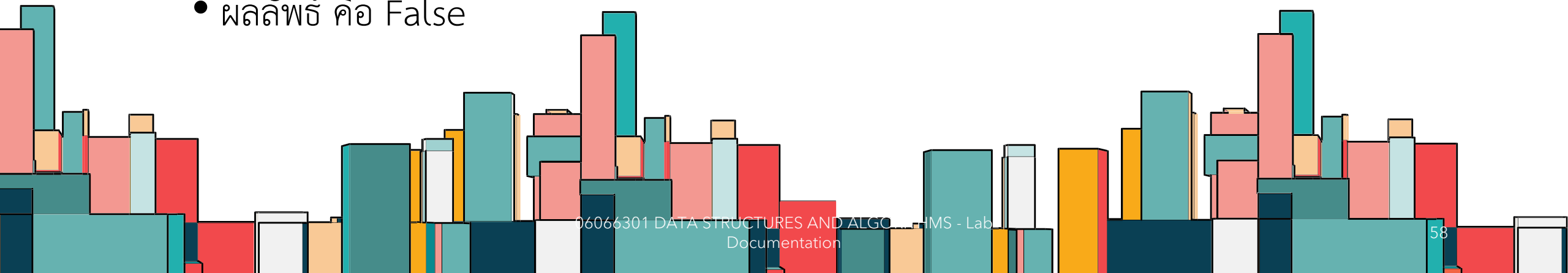
```
def hello(name):
    print("Hello", name)

def area(width, height):
    c = width * height
    return c
```

- การเรียกใช้ฟังก์ชัน (Calling a function)
  - hello(“IT-KMITL”)
  - my_area = area(10, 5)

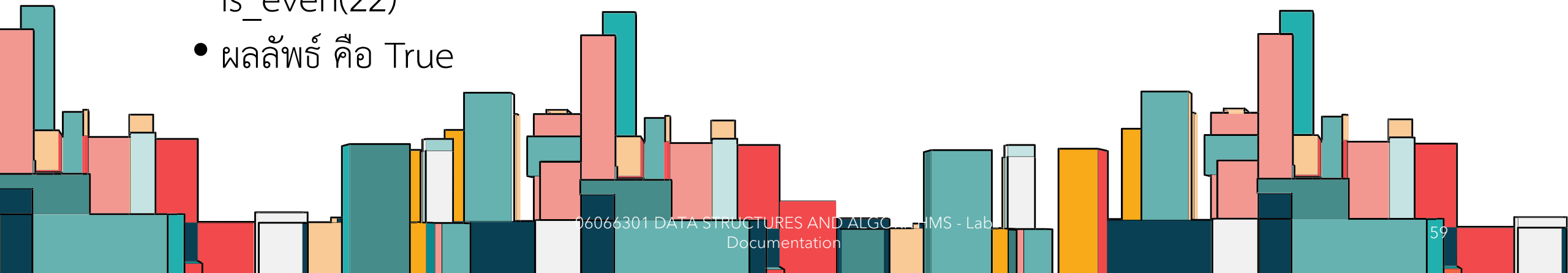
LAB 1.1

- จงเขียนฟังก์ชัน **is_multiple(n,m)** ที่รับค่าตัวเลขจำนวนเต็ม 2 ค่า (n และ m) และทำการคืนค่า (return) ผลลัพธ์เป็นค่า True หรือ False
  - คืนค่า True ก็ต่อเมื่อ n = mi โดยที่ i เป็นเลขจำนวนเต็ม
  - ไม่เช่นนั้น ให้คืนค่า False
- ตัวอย่าง
  - is_multiple(10, 3)
  - ผลลัพธ์ คือ False



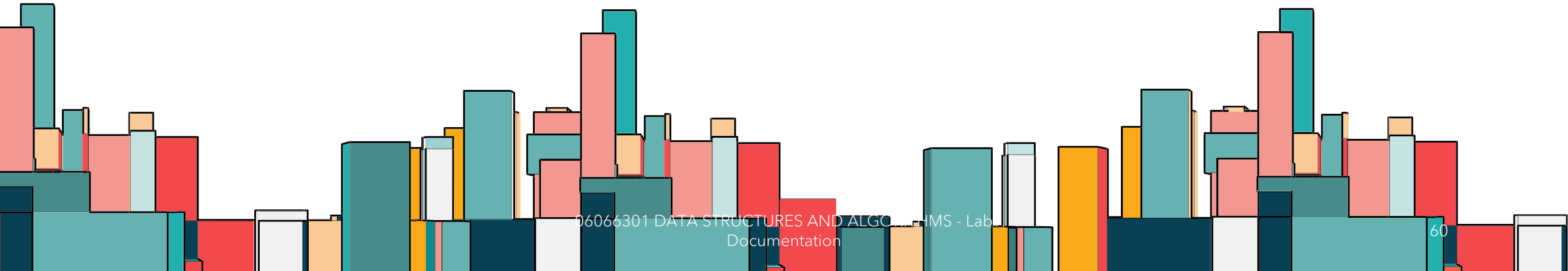
LAB 1.2

- จงเขียนฟังก์ชัน **is_even(k)** ที่รับค่าตัวเลขจำนวนเต็ม 1 ค่า (k)และทำการคืนค่า (return) ผลลัพธ์เป็นค่า True หรือ False
  - คืนค่า True ก็ต่อเมื่อ k เป็นเลขคู่
  - ไม่เช่นนั้น ให้คืนค่า False
- **ไม่อนุญาต**ให้ใช้โอเปอเรเตอร์สำหรับการคูณ การ mod หรือ การหาร
- ตัวอย่าง
  - is_even(22)
  - ผลลัพธ์ คือ True



LAB 1.3

- จงเขียนฟังก์ชัน **minmax(data)** ที่รับค่าลิสต์ข้อมูลตัวเลข (data) และทำการคืนค่า (return) ผลลัพธ์เป็นข้อมูล tuple ความยาว 2 ข้อมูล
- **ไม่อนุญาต**ให้ใช้ built-in function min(), max()
- ตัวอย่าง
  - minmax([22,54,7,87,12,9,63,55,48])
  - ผลลัพธ์ คือ (7, 87)



แบบฝึกหัดเสริม

- จงเขียนฟังก์ชัน ที่รับค่าตัวเลขจำนวนเต็มบวก n และทำการคืนค่า (return) ผลลัพธ์เป็นผลรวมของค่ายกกำลังสองของทุกๆ ค่าที่น้อยกว่า n ตัวอย่างเช่น
  - n = 5
  - ผลลัพธ์ที่ต้องการ คือ $1^2 + 2^2 + 3^2 + 4^2$
- จงเขียนฟังก์ชัน ที่รับค่าตัวเลขจำนวนเต็มบวก n และทำการคืนค่า (return) ผลลัพธ์เป็นผลรวมของค่ายกกำลังสองของทุกๆ ค่าที่เป็นเลขคี่และมีค่าน้อยกว่า n ตัวอย่างเช่น
  - n = 7
  - ผลลัพธ์ที่ต้องการ คือ $1^2 + 3^2 + 5^2$

REFERENCES

- Roberto T., Michael H. G. and Michael T. G. 2013. **Data Structures and Algorithms in Python**. Wiley.
- https://www.w3schools.com/python/