


Class period 1

บทที่ 2 ตัวแปรและโครงสร้างข้อมูลของภาษาไพธอน

Basic python

python101


1



Variables คืออะไร

- Variables คือตัวแปรที่ใช้เก็บข้อมูล ยกตัวอย่างกำหนดตัวแปร pi เก็บค่า $\pi = 3.14159265359$ ไม่ต้องพิมพ์ยาว ใช้ตัวแปร pi ที่กำหนดไว้แล้วแทน
- หลักการตั้งชื่อตัวแปรเบื้องต้น
 - ตั้งให้สื่อ
 - ภาษาอังกฤษ
 - ใช้ตัวเลขได้แต่ห้ามขึ้นต้นด้วยตัวเลข
 - ห้ามเว้นวรรค
 - ห้ามตั้งชื่อตัวแปรที่ซ้ำกับชื่อฟังก์ชันต่างๆ (def, for, range, etc.)


2



ชนิดของตัวแปร

- Int : ตัวเลขจำนวนเต็ม เช่น $a = 10$
- float : จำนวนจริง (ทศนิยม) เช่น $b = 10.0$
- ตัวอักษร (char (character)) ข้อความ (text หรือ string) เช่น $c = \text{'ธนพงศ์'}$
- ตัวเลขที่เป็น string ไม่สามารถเอามา บวก ลบ คูณ หาร กับตัวเลขได้ เช่น $d = \text{'10'}$


3



variable casting (การเปลี่ยนชนิดของข้อมูล)

- กรณีต้องใช้ข้อมูลที่นำมาจากที่อื่น เราสามารถเปลี่ยนชนิดของข้อมูลตามที่เรากำลังใช้งานได้ โดยการกำหนดชนิดของข้อมูลที่ต้องการเปลี่ยนไว้หน้าตัวแปร
- int(d)
- float(d)
- str(d)


4



Operation (การเอาตัวแปร 2 ตัวมาทำอะไรกัน) (Operators + , - , * , / , %)

- การบวก
- การลบ
- การคูณ
- การหาร
- การหารแบบ % เครื่องหมาย modulo คือการหารเอาเศษ

5



คำสั่ง print แบบพิเศษ (การ format string)

- แบบที่ 1 คำสั่ง print พื้นฐาน เช่น `print(ตัวแปร)` สิ่งที่อยู่ข้างในวงเล็บคือ ตัวแปร หรือ string
- แบบที่ 2 การเพิ่มข้อความที่ต้องการนอกจากรากตัวแปร คือการเพิ่ม f หน้า 'string' และใช้ { } ใส่ code เช่น `print(f'% คือการหารเอาเศษ เช่น 7%3 = {7%3}')`
- `\n` คือการขึ้นบรรทัดใหม่
- `\\` ใช้ในการตัด text แยกใน code และ code จะถูกอ่านปกติ

6

DATA STRUCTURE (โครงสร้างข้อมูล)

- List คือ การเอาตัวแปรหลายๆตัวมาเรียงกัน สามารถสร้างได้ 2 แบบ ดังนี้
- แบบที่ 1 `list_a = []`
- แบบที่ 2 `list_a = list()`
- `list_b = [1,5,V]`
- ลำดับที่อยู่ใน list มีความสำคัญ ลำดับใน list เริ่มจาก 0,1,2,...
- อยากรู้สมาชิกของ list_b ตัวที่ 1 ให้เขียน `list_b[1]` คือ 5

7

append() การเพิ่มสมาชิกเข้าใน list

- คำสั่ง `.append()` ตามด้วยค่าที่ต้องการเพิ่มใน () สามารถเพิ่มสมาชิกเข้าใน list ที่ต้องการได้ เช่น
- `list_b.append('u')` ต่อมาลอง `print(list_b)`
- `[1, 5, 'v', 'u']` จะเห็นว่า 'u' ถูกเพิ่มเข้ามาใน list_b
- คำสั่ง `.pop()` ใช้สำหรับดึงสมาชิกที่สุดท้ายออกจาก list
- `list_b.pop()` ต่อมาลอง `print(list_b)`
- `[1, 5, 'v']` จะเห็นว่า 'u' ถูกดึงออกจาก list_b

8

String > list of characters

- คำสั่ง `len()` คือคำสั่งตรวจสอบความยาวของ list (จำนวนสมาชิก)
- String มีค่าเป็น list เช่น
- `t = 'python is easy'`
- `len(t)` จะเท่ากับ 14 นับตามจำนวนตัวอักษรและวรรคหรือช่องว่างก็จะถูกนับ

9

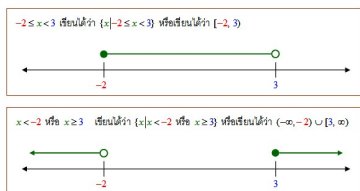
access a member of a list (list&string)

- สมาชิกของ list จะเริ่มนับจาก 0, -1 คือสมาชิกตัวสุดท้าย
- `t = 'python is easy'`
- `t[1]` จะเท่ากับ y คือสมาชิกตัวที่ 1 เริ่มนับ p เท่ากับ 0
- `t[-1]` จะเท่ากับ y คือสมาชิกตัวสุดท้าย

10

List slicing

- List slicing สามารถทำได้โดยใช้ colon `[a:b] -> [a,b)`
- ตัวที่อยู่ข้างใน [] เรียกว่า index(ตัวชี้)



11

ตัวอย่าง

- `t = 'python is easy'`
- `t[7:9]`
- ถ้าเว้นว่างหน้า : หมายความว่า เริ่มตั้งแต่ตัวแรก เช่น `t[:6]` คือ python
- ถ้าเว้นว่างหลัง : หมายความว่า ไปจนถึงตัวสุดท้าย เช่น `t[10:]` คือ easy
- ดังนั้น `t[7:9]` จะเท่ากับตัวที่ 7 ใน t ไปจนถึงตัวที่ 8 เพราะ 9 คือจุดจบ คือ is

12

ตัวอย่าง

- `t = 'python is easy'`
- `t[::2]` โดยคำสั่ง : ตัวที่สอง จะใช้กำหนด step(การกระโดด) ดังนั้น `t[::2]` step=2
- ผลลัพธ์จะได้ `pto ses`
- `list_a = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]`
- `list_a[::2]`
- ผลลัพธ์จะได้ `[0, 2, 4, 6, 8]`
- `t[2::2]` เริ่มต้นตัวที่ 2 โดยกำหนด step เป็น 2
- ผลลัพธ์จะได้ `to ses`

13

13

การนำ list มาต่อกัน

- สามารถนำ list มาต่อกันได้ด้วยการเติม + ตามด้วยค่าที่ต้องการต่อ
- string ต่อ string
- `t = 'python is easy'`
- `t + '??'` จะเท่ากับ `'python is easy??'`
- `list_b = [1, 5, 'v']`
- ไม่สามารถนำ list ปกติมาต่อกับ string ได้ เช่น `t + list_b` ไม่สามารถทำได้
- list ต่อ list
- `list_b + list_a` จะเท่ากับ `[1, 5, 'v', []]`

14

14

split string การแบ่ง string ตามสัญลักษณ์ที่กำหนด

- สามารถแบ่งได้โดยการเติม `.split` ตามด้วยสัญลักษณ์ที่ต้องการใน () เช่น
- `t = 'python is easy'`
- `t.split(' ')` หมายความว่า แบ่งข้อความในตัวแปร `t` โดยมีสัญลักษณ์ ' ' คือช่องว่าง ดังนั้นจะได้
- `['python', 'is', 'easy']`
- `time = '12:30:15'`
- `time.split(':')` หมายความว่า แบ่งข้อความในตัวแปร `time` โดยมีสัญลักษณ์ ':' ดังนั้นจะได้
- `['12', '30', '15']`

15

15

วิธีรวม string กลับ

- `t = 'python is easy'`
- `t_sp = t.split(' ')`
- `print(t_sp)` = `['python', 'is', 'easy']`
- สามารถรวมกลับได้โดยการกำหนดสัญลักษณ์ที่ต้องการ ตามด้วย `.join` ตามด้วยตัวแปรที่ต้องการรวมกลับใส่ใน ()
- `t_join = ''.join(t_sp)`
- `print(t_join)` = `python is easy`

16

16

Homework class period 1

คำนวณเวลาเป็นวินาทีของเวลาต่อไปนี้โดยใช้คำสั่ง `split()` ช่วย และ `print` ออกมาให้สวยงาม

- 12:30:15
- 13:41:07
- 12:53:15
- 00:59:25
- 11:11:11
- 16:06:09

17

17