Lab 1: Input & output

```
s = input('Enter input: ')
```

ตัวแปร s เป็น string

```
first,last = s.split(' ') หรือ s.split() ก็ได้
```

ตัด string s เป็นท่อน (□ หมายถึง space) เช่น

John□Snow ตัดได้ 2 ท่อน คือ 'John' และ 'Snow'

John□□Snow ตัดได้ 3 ท่อน คือ 'John' และ '' และ 'Snow'

```
print('Hello', "I'm", first, last)
print('Hello ', "I'm ", first, ' ', last, sep='')
```

คำสั่ง print ทั้งสองบรรทัดได้ผลลัพธ์เหมือนกัน ค่า default ของ sep คือ ' ' (space)

Lab 1: Type conversion

```
s = input('Enter age: ')
```
สมมติว่าพิมพ์ 18↵

ตัวแปร s เป็น string

```
n = int(s)
```
หรือเขียนรวบไปเลยว่า
```
n = int(input('Enter age: '))
```
ตัวแปร n เป็น integer number

```
s = input('Enter price: ')
```
สมมติว่าพิมพ์ 35.75↵

ตัวแปร s เป็น string

```
p = float(s)
```
ตัวแปร p เป็น floating-point number

Lab 1: Substring

```
s = '2565'
n = int(s)
```

แปลง string s ให้เป็น integer n

ถ้าแปลง integer เป็น string ใช้ฟังก์ชัน str(…)

```
print(s[0:2])
```
ได้ 2 ตัวแรก คือ '25'

```
print(s[1:3])
```
ได้ 2 ตัวกลาง คือ '56'

```
print(s[-2:])
```
ได้ 2 ตัวท้าย คือ '65'

หรือจะใช้วิธี `n // 100` ก็ได้

หรือจะใช้วิธี `(n % 1000) // 10`

หรือจะใช้วิธี `n % 100` ก็ได้

Q1: จงเขียนโปรแกรมที่รับราคาสินค้าและค่าขนส่งจากผู้ใช้ แล้วคำนวณยอดรวมที่ลูกค้าต้องจ่าย แล้วแสดงผลลัพธ์

ข้อนี้นิสิตต้องใช้ฟังก์ชัน input() และฟังก์ชัน print()

ต้องแปลง string เป็น int ก่อน ถึงจะนำมาคำนวณ เช่น + - × ÷ ได้

Q2: จงเขียนโปรแกรมที่รับชื่อและนามสกุลจากผู้ใช้ แล้วนำมาสร้างเป็นสตริงใหม่ 2 สตริง โดย

สตริงหนึ่งเป็นชื่อต่อด้วยนามสกุลและมีช่องว่าง (blank) คั่นกลาง

อีกสตริงหนึ่งเป็นนามสกุลต่อด้วยชื่อและมี , ต่อด้วยช่องว่างคั่นกลาง

ข้อนี้นิสิตต้องใช้ฟังก์ชัน split() เพื่อแยกสตริง 1 ตัวให้เป็นชื่อและนามสกุล

Q3: จงเขียนโปรแกรมที่รับชื่อและอายุจากผู้ใช้ แล้วคำนวณปีเกิดของผู้ใช้และแสดงผลลัพธ์

ข้อนี้นิสิตต้องใช้ฟังก์ชัน split() ชื่อเป็น string อายุต้องแปลงเป็น integer ปีเกิดคือ ปี พ.ศ. ปัจจุบันลบด้วยอายุ

Q4: จงเขียนโปรแกรมที่รับชื่อและอายุจากผู้ใช้ แล้วคำนวณปีเกิดของผู้ใช้เพื่อสร้าง email address ของผู้ใช้โดย เอาชื่อมาต่อด้วยเลข 2 หลักหลังของปีเกิดและต่อด้วย @chula.ac.th แสดงผลลัพธ์

ข้อนี้ไม่ต้องใช้ split() เพราะรับอินพุต 2 ครั้ง ครั้งแรกรับชื่อ ครั้งที่สองรับอายุ

จะตัด 2 หลักหลังของปี พ.ศ. ยังไง ทำได้ 2 วิธี คือ 1) ตัด substring จากตัวแปร string หรือ

2) ใช้วิธี mod จากตัวแปร integer

Lab 2: Boolean variables

```
flag = True
flag = False

a = 3
b = 4
c = 5

flag = a < b
flag = a < b and b < c
flag = (a < b) and (b < c)
flag = a*a + b*b == c*c

print(a)
print(flag)
```

"และ" ใช้ and "หรือ" ใช้ or

ตอน print จะทำ type conversion เป็น string ให้อัตโนมัติ

Lab 2: Quadratic equation

Q1 (3 points)

จงเขียนโปรแกรมที่คำนวณหาผลเฉลยของสมการกำลังสอง (quadratic equation) โดยให้รับสัมประสิทธิ์ a, b, c ในสมการ $ax^2 + bx + c = 0$ จากผู้ใช้แล้วคำนวณหาผลเฉลยจากสูตร $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ แล้ว แสดงผลลัพธ์ดังตัวอย่างข้างล่างนี้ (a ต้องไม่เป็น 0 และ $b^2 \geq 4ac$)

หมายเหตุ: การหารากที่สองทำได้โดยใช้การยกกำลัง ½ เช่น 2**0.5 หรือ ใช้ฟังก์ชัน sqrt ในโมดูล math ซึ่ง ต้องใช้คำสั่ง import math เพื่อระบุว่าโปรแกรมนี้ใช้ฟังก์ชันในโมดูล math และ เรียกใช้ฟังก์ชัน sqrt ดัง ตัวอย่างนี้

```
import math

z = math.sqrt(2) # value in z is 1.4142135...
```

ข้อนี้สมมติว่าอินพุตเข้าเงื่อนไข `a ≠ 0` และ `b²  ≥ 4ac` เสมอ นิสิตไม่ต้องตรวจสอบอินพุตว่าตรงกับเงื่อนไขหรือไม่ สังเกตว่าใน `test` ก็ไม่มีอินพุตที่ไม่เข้าเงื่อนไข และโจทย์ก็ไม่ได้บอกว่าถ้ามีอินพุตที่ไม่เข้าเงื่อนไข จะให้ทำอย่างไร

Lab 2: Quadratic equation

Q2 (3 points)

จงเขียนโปรแกรมที่หาว่าสามารถหาผลเฉลยของสมการกำลังสองจากสูตร $x = -b \pm \sqrt{b^2 - 4ac}$ ได้หรือไม่

โดยตรวจสอบว่า $b^2 - 4ac \geq 0$ และ $a \neq 0$ แล้วแสดงผลลัพธ์ดังตัวอย่างข้างล่างนี้

ข้อนี้นิสิตต้องเขียนนิพจน์เพื่อคำนวณค่า `True` หรือ `False` ได้

เพื่อหาว่า `(1) a ≠ 0` มีค่าเป็นเป็น `True` หรือ `False`

เพื่อหาว่า `(2) b²  ≥ 4ac` มีค่าเป็นเป็น `True` หรือ `False`

แล้วเชื่อมด้วย `logical and` ว่า `(1)` และ `(2)` เป็น `True` หรือ `False`

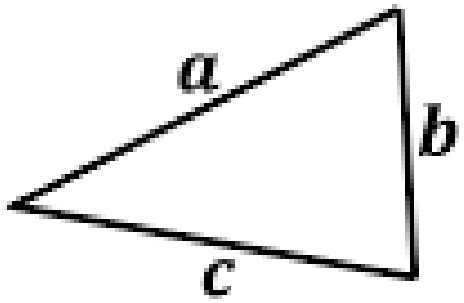
แล้ว `print` ค่านั้นออกมา

Lab 2: Triangle

Q3 (3 points)

จงเขียนโปรแกรมที่รับความยาวด้าน 3 ด้านของสามเหลี่ยมจากผู้ใช้ แล้วตรวจสอบว่าเป็นด้านของสามเหลี่ยม หรือไม่ และแสดงผลลัพธ์ดังตัวอย่างข้างล่างนี้

ข้อนี้นิสิตต้องรู้ Triangular Inequality Theorem
ด้านทั้งสามคือ a, b, c ต้องเข้าเงื่อนไขนี้ ถึงจะเป็นสามเหลี่ยม



$$a + b > c$$
$$a + c > b$$
$$b + c > a$$

Lab 2: Pythagorean theorem

Q4 (3 points)

จงเขียนโปรแกรมที่รับความยาวด้าน 3 ด้านของสามเหลี่ยมจากผู้ใช้ แล้วตรวจสอบว่าเป็นด้านของสามเหลี่ยม มุมฉากหรือไม่ และแสดงผลลัพธ์ดังตัวอย่างข้างล่างนี้

ข้อนี้รับอินพุต `a, b, c` แต่ไม่ได้บอกว่าด้านใดจะเป็นด้านที่ยาวที่สุด ถ้าดูใน `test` ด้านที่ยาวที่สุดอาจจะเป็น `a` หรือ `b` หรือ `c` ก็ได้ ดังนั้นถ้าทึกทักว่าด้านที่ยาวที่สุดคือ `c` โปรแกรมก็อาจจะทำงานผิดได้ ผมลองเสนอ 2 วิธี เลือกวิธีใดวิธีหนึ่ง

วิธีที่ 1: หาด้านที่ยาวที่สุดก่อน (c)

แล้วเช็คเงื่อนไข $a^2 + b^2 = c^2$

ไม่บอกว่าทำยังไง ลองคิดดูเอง

วิธีที่ 2: ทำทุกด้าน

เช็คเงื่อนไข $a^2 + b^2 = c^2$ หรือ

เช็คเงื่อนไข $a^2 + c^2 = b^2$ หรือ

เช็คเงื่อนไข $b^2 + c^2 = a^2$

ในทางปฏิบัติ มี rounding error เช็คว่าเท่ากับไม่ได้ เช็คว่าคลาดเคลื่อนไม่เกิน ทศนิยมตำแหน่งที่ 6 แทน

$$|a^2 + b^2 - c^2| < 0.000001$$

ถ้าเงื่อนไขนี้เป็นจริง ก็พอที่จะประมาณได้ว่าเป็นสามเหลี่ยมมุมฉาก (c เป็นด้านตรงข้ามมุมฉาก)

Lab 3: if-else

Q1 (4 points)

จงเขียนโปรแกรมที่คำนวณหาผลเฉลยของสมการกำลังสอง (quadratic equation) โดยให้รับสัมประสิทธิ์ a, b, c ในสมการ $ax^2 + bx + c = 0$ จากผู้ใช้แล้วคำนวณหาผลเฉลย จากสูตร $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ ได้ให้ คำนวณและพิมพ์ผลเฉลย

ถ้าไม่สามารถหาได้ให้พิมพ์ข้อความ 'No real solution.' ดังตัวอย่างข้างล่างนี้

```
if there are real solutions :
    คำนวณผลเฉลยตรงนี้
    print('x = ', …)
else:
    print('No real solution.')
```

Lab 3: if-elif-else

Q2 (7 points)

จงเขียนโปรแกรมที่รับน้ำหนัก (kg.) และส่วนสูง (m.)

จากผู้ใช้มาคำนวณ bmi จาก น้ำหนัก (kg.)/ส่วนสูง (m.) ยกกำลัง 2 แล้วแสดงผลลัพธ์ว่าผู้ใช้มีรูปร่างในระดับใดดังนี้

ต่ำกว่า 18.5 ผอม

ไม่น้อยกว่า 18.5 แต่ต่ำกว่า 23.0 รูปร่างปกติ

ไม่น้อยกว่า 23.0 แต่ต่ำกว่า 25.0 รูปร่างอ้วน

ไม่น้อยกว่า 25.0 แต่ต่ำกว่า 30.0 อ้วนระดับ 1

ไม่น้อยกว่า 30.0 อ้วนระดับ 2

```
if bmi < 18.5:
  print('ผอม')
elif bmi < 23.0:
  print('รูปร่างปกติ')
elif bmi < 25.0:
  print('รูปร่างอ้วน')
elif bmi < 30.0:
  print('อ้วนระดับ 1')
else:
  print('อ้วนระดับ 2')
```

ไม่จำเป็น

~~18.5 <= bmi and~~ bmi < 23.0

สูตรคำนวน **ดัชนีมวลกาย (BMI)**

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

```
BMI = W / (H ** 2)
```

Lab 3: Is (x,y) in a rectangle (x_1,y_1), (x_2,y_2)?

$x_1 < x_2$
$y_1 < y_2$

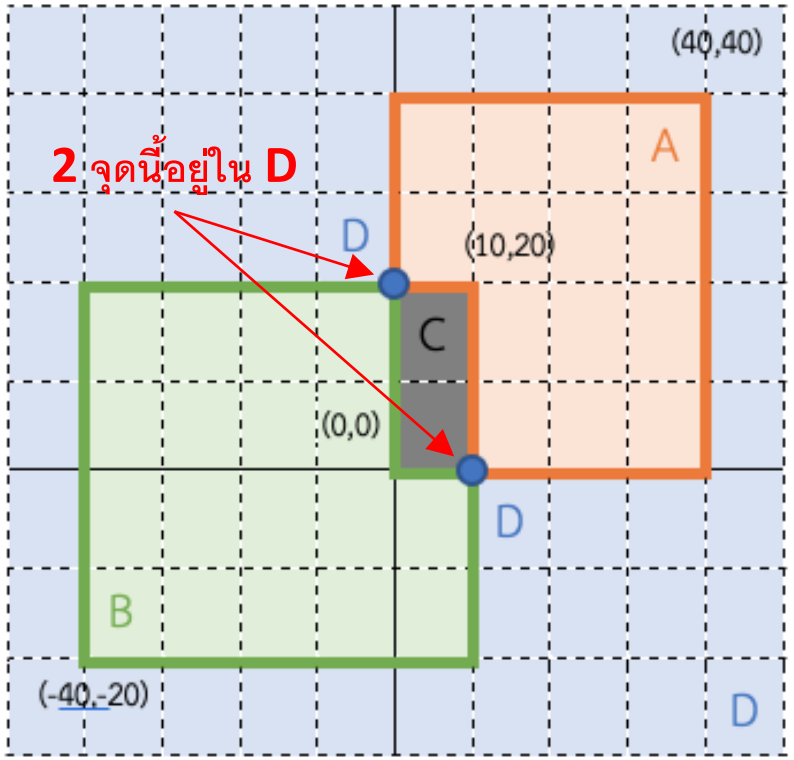
Q3 (6 points)

จงเขียนโปรแกรมที่รับพิกัด (x,y) ของจุดหนึ่ง ในรูปแบบจำนวนเต็ม แล้วตรวจสอบว่าจุดนั้นอยู่ในบริเวณ A หรือ B หรือ C หรือ D ในรูปข้างล่าง

```
(x1 < x and x < x2) and (y1 < y and y < y2)
```
ไม่นับจุดบนเส้นขอบ

```
(x1 ≤ x and x ≤ x2) and (y1 ≤ y and y ≤ y2)
```
นับจุดบนเส้นขอบ

```
if (x,y) in C:
    print('(x,y) is in C')
elif (x,y) in A:
    print('(x,y) is in A')
elif (x,y) in B:
    print('(x,y) is in B')
else:
    print('(x,y) is in D')
```



Lab 3: check multiple conditions

Q4 (5 points)

จงเขียนโปรแกรมที่รับรหัสนิสิต แล้วตรวจสอบว่าเป็นรหัสที่ใช้ได้ คือ

- เป็นเลข 10 หลัก
- เลข 2 หลักแรกเป็นปีที่เข้าเรียน (50-66)
- เลขหลักที่ 3 (จากหน้า) เป็น 3, 4 หรือ 7
- เลข 2 หลักสุดท้ายของรหัสนิสิตเป็นรหัสคณะ คือ 21 22 23 24 25 26 27 28

แนะนำว่าใช้ตัวแปร boolean (c1, c2, c3, c4) เก็บเงื่อนไขทั้ง 4 ข้อ ว่าเป็น True หรือ False ลองเขียนโค้ดให้ดู 2 แบบ

แบบที่ 1

ถ้าเอาเงื่อนไขทั้งหมดมาโค้ดไว้ตรงนี้จะเขียนยาวมาก เลยให้เก็บไว้ในตัวแปร c1, c2, c3, c4 แทน

```
if c1 and c2 and c3 and c4:
    print('Valid ID')
else:
    print('Invalid ID')
```

แบบที่ 2

```
if not c1:
    print('Invalid ID')
elif not c2:
    print('Invalid ID')
elif not c3:
    print('Invalid ID')
elif not c4:
    print('Invalid ID')
else:
    print('Valid ID')
```

Lab 4: if-elif-else

รับ input จาก keyboard

```
if circle:
      … … …
elif square:
      … … …
elif rectangle:
      … … …
else:
      … … …
```

Choose circle, square or rectangle : *circle*↵
square↵
rectangle↵

Length of the radius of the circle : *3.6*↵
Area is 40.71504079052372

Length of the side of the square : *3.6*↵
Area is 12.96

Length of two sides of the rectangle : *2.5, 3.6*↵
Area is 9.0

Lab 4: if-elif-else

Weight : *61.3 kg*↵

Height : *1.65 m*↵

รูปร่างปกติ

Weight : *159.3 lbs*↵

Height : *4.92 ft*↵

อ้วนระดับ 2

Weight : *110.8 lbs*↵

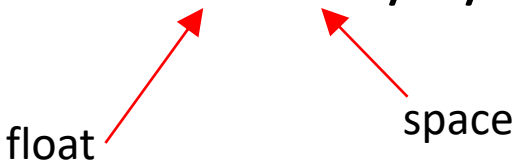
Height : *167 cm*↵

ผอม

Assume ว่าป้อน input ถูกต้องเสมอ

xxx kg/lbs

xxx m/ft/cm



1 kg = 2.205 lbs

100 cm = 1 m = 3.2808399 ft

Lab 4: if-elif-else

	กลุ่ม 1				กลุ่ม 2			
	ปริญญาตรี		บัณฑิตศึกษา		ปริญญาตรี		บัณฑิตศึกษา	
รหัสนิสิตขึ้นต้นด้วย	ภาค 1/2	ภาค 3	ภาค 1/2	ภาค 3	ภาค 1/2	ภาค 3	ภาค 1/2	ภาค 3
50 – 55	18,000	4,500	26,000	7,000	14,500	4,500	19,000	7,000
56 เป็นต้นไป	21,000	5,250	31,000	7,750	17,000	5,250	23,000	7,750

เลขหลักที่ 3 (จากหน้า) ของรหัสนิสิตระดับปริญญาบัณฑิต คือ 3, 4 และบัณฑิตศึกษา คือ 7

เลข 2 หลักสุดท้ายของรหัสนิสิตเป็นรหัสคณะ

รหัสคณะในกลุ่ม 1 คือ 21 23 25 28

รหัสคณะในกลุ่ม 2 คือ 22 24 26 27

ตัวอย่างการทำงาน:

Enter student ID : **6033235423**
Enter semester : **1**
Registration fee : 21000

Id นิสิตที่ valid
- เป็นเลข 10 หลัก
- เลข 2 หลักแรก ≥ 50
- เลขหลักที่ 3 จากหน้า เป็น 3 หรือ 4 หรือ 7 เท่านั้น
- เลข 2 หลักสุดท้าย เป็น 21 - 28 เท่านั้น

semester ที่ valid คือต้องเป็น 1, 2, 3 เท่านั้น
เช็ค string เลยก็ได้ว่าเป็น '1' หรือ '2' หรือ '3' ไม่ต้องแปลงเป็น int

Write easy-to-read code

- เรียบเรียงความคิดก่อน อย่าด้นสด แบบโค้ดไปคิดไป
- ตั้งชื่อตัวแปรให้สื่อความหมาย เช่น `x, y, a, b, n, id, sid`
  `studentId, numStudents`
- ถ้าโค้ด 1 บรรทัดยาวเกินไป ให้เขียนเป็นหลาย ๆ บรรทัดแทน

Otherwise you can do something like this:

```
if (a == True and
    b == False):
```

or with explicit line break:

```
if a == True and \
   b == False:
```

Using parentheses, your example can be written over multiple lines:

```
a = ('1' + '2' + '3' +
    '4' + '5')
```

The same effect can be obtained using explicit line break:

```
a = '1' + '2' + '3' + \
    '4' + '5'
```

Lab 5: while loop *n* times (*n* is known)

```
i = 0
while (i < 5):
    i = i + 1
    print(i)
```

```
1
2
3
4
5
```

```
for i in range(1, 6):
    print(i)
```

```
1
2
3
4
5
```

Lab 5: while loop *n* times (*n* is not known)

```
while True:
    name = input("Input: ")
    if name == "quit" or name == "q":
        break
```

Ken
Lim
Pim
quit

Lab 5: initialize and then iterate

```
n = int(input("Number of students : "))
i = 0
while i < n:
    i = i + 1
    score =  float(input("Student " + str(i) + " : "))
```

Number of students : 5
Student *1* : 76
Student *2* : 84
Student *3* : 97
Student *4* : 80
Student *5* : 65

```
min = ค่ามากที่สุด
while …:
  if x < min:
    min = x
```

```
max = ค่าน้อยที่สุด
while …:
  if x > max:
    max = x
```

```
sum = 0
while …:
  sum = sum + x
avg = sum / n
```

Lab 6: loop again

ตัวอย่างการทำงาน:

Next word : **My**
Next word : **name**
Next word : **is**
Next word : **next**
Next word : **word**
Next word : **.**
Sentence: My name is next word

loop จนกว่าจะเจอ fullstop

ตัวอย่างการทำงาน:

Enter an integer : **12**
12 is divisible by:
1
2
3
4
6
12

loop 1 to n (n = 12)

ตัวอย่างการทำงาน:

Day 1 : **32.6**
Day 2 : **34.4**
Day 3 : **35.0**
Day 4 : **34.6**
Temperature dropped on day 4
Day 5 : **35.2**
Day 6 : **35.6**
Day 7 : **34**
Temperature dropped on day 7

loop 1 to 7

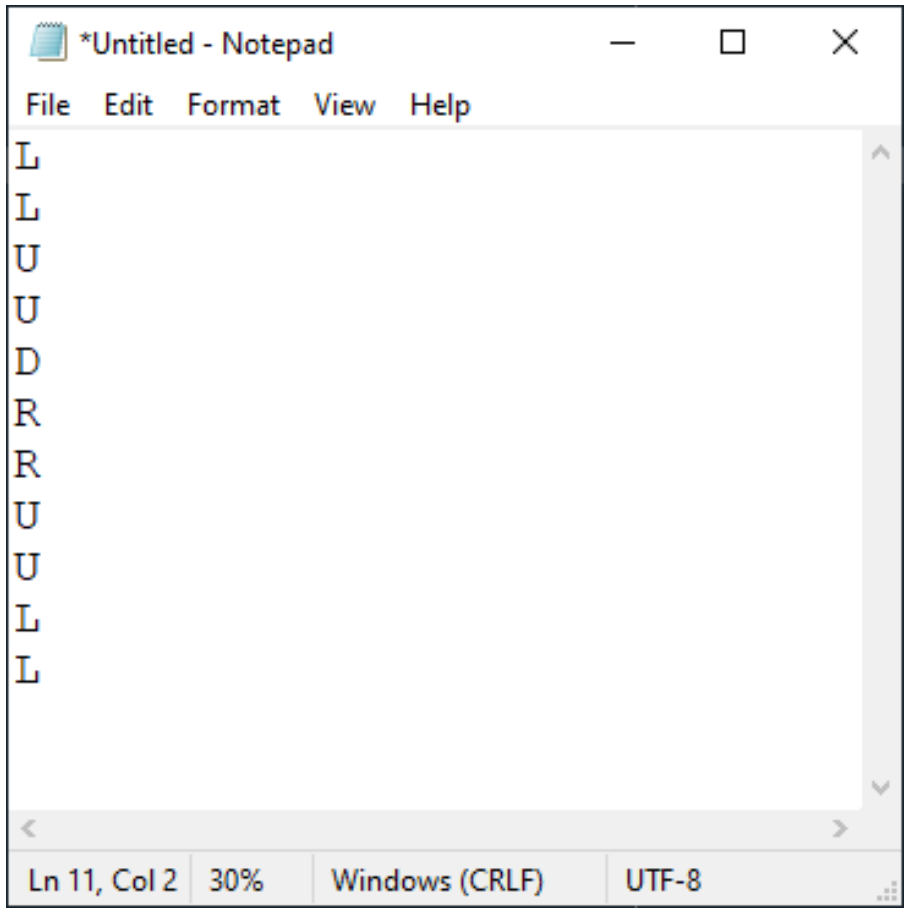
ตัวอย่างการทำงาน:

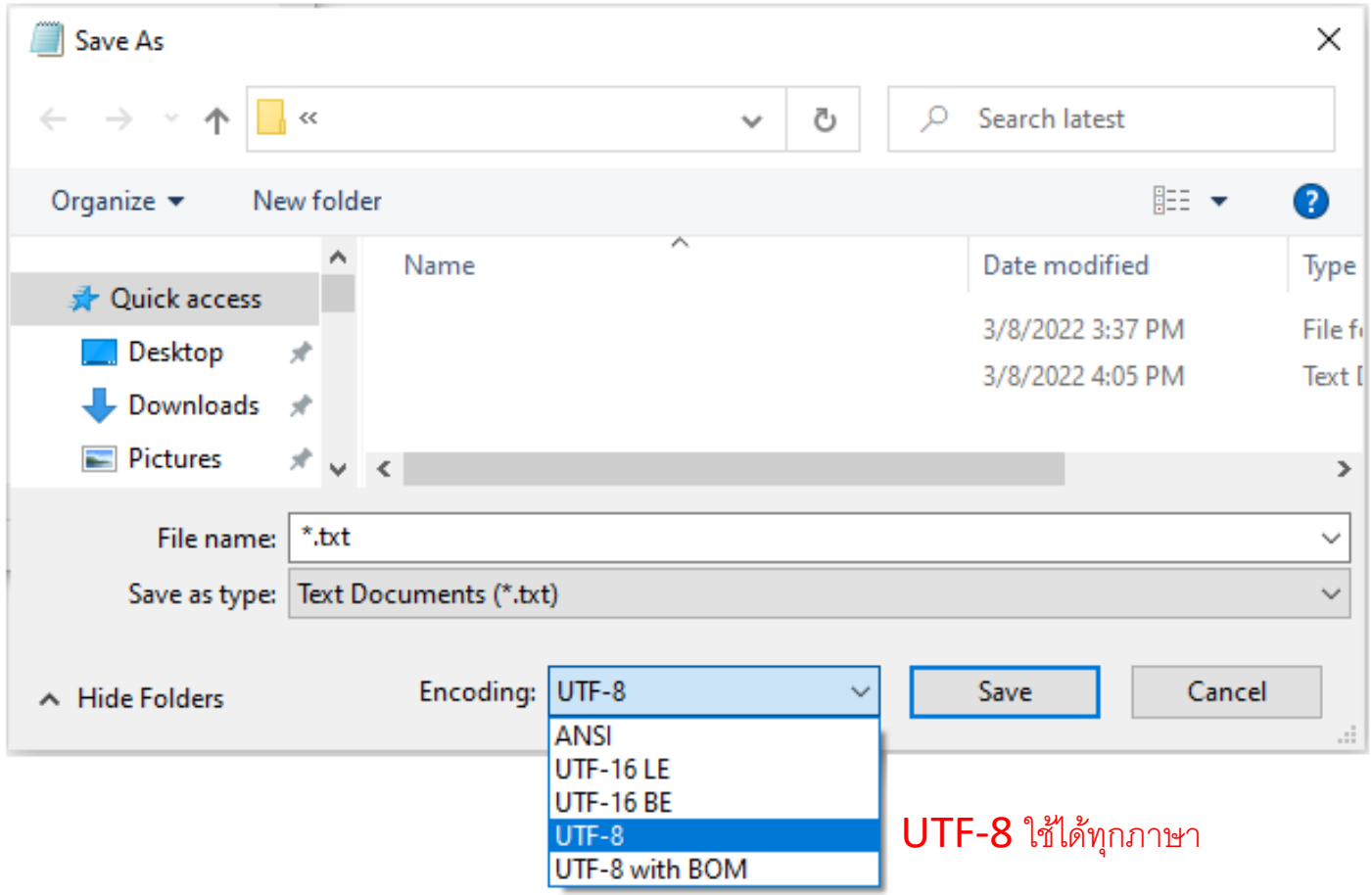
withdraw : **60000**
Insufficient fund.
withdraw : **40000**
withdraw : **20000**
Insufficient fund.
withdraw : **10000**
Balance is 0

loop จนกว่า balance = 0

Lab 7: Read a File

ใช้โปรแกรม Text Editor เช่น Notepad เพื่อสร้างไฟล์อินพุต





UTF-8 ใช้ได้ทุกภาษา

```
f = open('C:\\Users\\Chat\\Documents\\robot.txt', 'r')
for x in f:
```

ในแต่ละรอบ `x` จะมีค่าเท่ากับ `L` หรือ `R` หรือ `U` หรือ `D` และมี `\r\n` ต่อท้าย

carriage return อยู่ในเครื่องพิมพ์ดีด ปัจจุบันไม่มีแล้ว

Windows ใช้ \r\n
Linux ใช้ \n

\r carriage return
\n line feed

ใช้ \ นำหน้าเพื่อบอกว่าเป็นตัวอักษรพิเศษ ดังนั้นเมื่อจะหมายถึงตัวอักษร backslash จึงเขียนเป็น \\

```
f = open('robot.txt', 'r')
for x in f:
```

ในแต่ละรอบ x จะมีค่าเท่ากับ L หรือ R หรือ U หรือ D และมี \r\n ต่อท้าย

ถ้าไม่ต้องการ \r\n ให้ใช้ x.strip() เพื่อเอา white-space character ด้านหน้าและด้านหลังออก

แต่ถ้าต้องการอ่านไฟล์แบบไม่เอา \r\n มาด้วย เขียนแบบนี้

```
f = open('robot.txt', 'r')
lines = f.read().splitlines()
for x in lines:
```

ในแต่ละรอบ x จะมีค่าเท่ากับ L หรือ R หรือ U หรือ D ไม่มี \r\n ต่อท้าย

ฟังก์ชัน read() จะอ่านทั้งไฟล์มาเป็น string ชิ้นเดียว แล้ว split ด้วยฟังก์ชัน splitlines() เป็นแต่ละบรรทัด
โดยไม่มี \r\n ติดมาด้วย แต่ถ้าใช้ฟังก์ชัน split('\n') จะผิด ถ้าบรรทัดสุดท้ายในไฟล์มี \r\n ต่อท้าย จะ split ได้เกินมา 1 บรรทัด
แต่ถ้าใช้ splitlines() จะไม่เจอปัญหานี้

Lab 8: List

```
items = ['apple', 'banana', 'cherry']
for x in items:
    print(x)

for i in range(len(items)):
     print(items[i])                    #index i = 0, 1, 2

items = ['1.1', '2.2', '3.3']           #list of string
print(items)                            #['1.1', '2.2', '3.3']
items = [float(x) for x in items]       #list of float
print(items)                            #[1.1, 2.2, 3.3]

f = open('robot.txt', 'r')
```

```
lines = f.read().splitlines()        #ได้ลิสต์ [line1, line2, …]
items = lines[0].split(' ')          #lines[0] คือบรรทัดแรก
```

ถ้า f.read().split(' ') มันจะติด \n มากับ item สุดท้าย
ต้อง f.read().strip().split(' ')

เวลา split ด้วย space เช่น a□b□c□□d จะได้ลิสต์ที่มีของ 5 ชิ้น
เพราะได้ empty string ที่อยู่ระหว่าง □ และ □ มาด้วย

```
items = text.split(' ')
items = [x for x in items if x != '']    #exclude empty string

items = [x.lower() for x in items if len(x) > 3]
items = […(2.จะทำอะไร)… for x in items if …(1.เงื่อนไข)…]
```

List Comprehension

```
def calculate():
     return a1, a2, a3                    #return multiple values

def run():
     b1, b2, b3 = calculate()
```

กำหนดค่าเริ่มต้นของ studentScore และ itemScore ก่อน
แล้วพยายามคำนวณหาค่าให้ได้

```
studentScore = [0,0,0,0,0,0,0,0]    #มีนักเรียน 8 คน
itemScore = [0,0,0,0,0,0,0,0,0,0]   #มีจำนวนข้อ 10 ข้อ
```

สมมติว่า itemScore = [7, 5, 0, 1, 0, 2, 6, 3, 4, 8]

```
m = [x for x in itemScore if x == min(itemScore)]
m = [x for x in range(0, 8) if itemScore[x] == min(itemScore)]
m = [str(x) for x in range(1, 9) if itemScore[x - 1] == min(itemScore)]
```

ลิสต์ m จะมีค่าดังนี้

```
[0, 0]          #ได้ item ที่มีค่าน้อยสุดในลิสต์
[2, 4]          #ได้ index ของ item ที่มีค่าน้อยสุดในลิสต์
['3', '5']      #ได้ index + 1 ของ item ที่มีค่าน้อยสุดในลิสต์
                #ทำ type conversion เป็น string ด้วย เพื่อนำไป join
```

ฟังก์ชัน min() และ max()
ใช้หาค่า min และ max ในลิสต์

`' '.join(m)` จะมีค่าเท่ากับ `'3 5'`

ฟังก์ชัน `join` จะนำ `string ' '` มาเชื่อมของในลิสต์ `m`

เขียนให้ดูอีกแบบ ใช้ตัวแปร `space` แทน `string ' '`

```
space = ' '
```

`space.join(m)` จะมีค่าเท่ากับ `'3 5'` เหมือนกัน

ถ้าใช้ `loop` จะยากกว่า เช่น เมื่อเป็นของชิ้นที่ `2` ขึ้นไป ต้องมี `space` ด้วย

ดังนั้นจึงใช้ฟังก์ชัน `join()`

Lab 9: List Operations

```
a = ['apple', 'banana', 'cherry']
a.append('orange')

['apple', 'banana', 'cherry', 'orange']

==========================================================

a = ['apple', 'banana', 'cherry']
a.insert(1, 'orange')

['apple', 'orange', 'banana', 'cherry']
```

ถ้าจะไม่ให้ `insert` ซ้ำกับของที่มีอยู่แล้ว นิสิตต้องเช็คเอง

```
a = ['apple', 'banana', 'cherry']
a[1] = 'blackcurrant'

['apple', 'blackcurrant', 'cherry']

==========================================================

a = ['apple', 'banana', 'cherry']
a.remove('banana')

['apple', 'cherry']
```

```
a = ['apple', 'banana', 'cherry']
print(a[1])
```

```
Banana
```

==========

a = [] คือ empty list

len(a) คือ จำนวน items ใน a

List Methods

Python has a set of built-in methods that you can use on lists.

Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
extend()	Add the elements of a list (or any iterable), to the end of the current list
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position
remove()	Removes the item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list

```
cars = ['Ford', 'BMW', 'Volvo']
cars.sort()   หรือจะ print(sorted(cars)) ก็ได้
print(cars)
```

หา sum digits เช่น zz1xa2d3

พิจารณาทีละ 1 character ว่าเป็น decimal (0 - 9) หรือไม่

String Type	Example	Python `.isdecimal()`	Python `.isdigit()`	Python `.isnumeric()`
Base 10 Numbers	`'0123'`	True	True	True
Fractions and Superscripts	`'⅗'`, `'2²'`	False	True	True
Roman Numerals	`'Ð'`	False	False	True

```
Enter a number between [0,9]: 3
[3, 13, 23, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 43, 53, 63, 73, 83, 93]
```

ให้ loop 1 - 100 แล้วแปลงเป็น string ดูว่ามี character ที่เป็น 3 หรือไม่
ถ้าใช้ก็ append ลงใน list หรือจะใช้ list comprehension ก็ได้

วิธีเช็ค substring

```
'3' in '11234'          True
'23' in '11234'         True
'21' in '11234'         False

text = '11234'
s = '3'

'3' in text             True
s in text               True
```

```
if (item in item_list == False):
```

Bug! มันจะไม่เข้าเงื่อนไข if ต้องใช้ not หรือ วงเล็บ

```
if (item not in item_list):
if ((item in item_list) == False):
```

Lab 10: Dictionary operations

```
a = {
    '2301117': 'Calculus I',
    '2301118': 'Calculus II',
    '2301286': 'Probability'
}
```

a['2301117']) หรือ a.get('2301117') จะมีค่าเท่ากับ 'Calculus I'

a.keys() จะมีค่าเท่ากับ ['2301117', '2301118', '2301286']

a.values() จะมีค่าเท่ากับ ['Calculus I', 'Calculus II', 'Probability']

a.items() จะมีค่าเท่ากับ ลิสต์ของทูเปิ้ล (key, value)

```
 [('2301117', 'Calculus I'), ('2301118', 'Calculus II'), ('2301286', 'Probability')]
```

```
if '2301117' in a:
if 'Calculus I' in a.values():
```

```
a['2301117'] = 'Chemistry'          แก้ไข (change)
a['2301499'] = 'Senior Project'     เพิ่ม (add)
a.pop('2301117')                    ลบ (remove)

for k in a:
   print(a)                         พิมพ์ key ทุกตัวใน a

b = a.copy()

carts = {
   'customer1': { 'pen': 2, 'pencil': 1 },          # nested dictionaries
   'customer2': { 'pen': 3, 'ruler': 1 },
   'customer3': { 'pencil': 1, 'rubber': 1 }
}
```

a = {} คือ empty dictionary

len(a) คือ จำนวน (key,value) ใน dictionary

```
a = {
    'Bill': 15,
    'Carlos': 3,
    'Anne': 10
}

[x for x in a if a[x] == max(a.values())]          # หา key ที่ value มีค่ามากที่สุด
{k: v for k, v in sorted(a.items())}               # sort by key จากน้อยไปมาก
```

sort by value จากมากไปน้อย

```
{k: v for k, v in sorted(a.items(), key=lambda item: item[1], reverse=True)}
```

Tuples

ทูเปิ้ลมีลักษณะคล้ายกับลิสต์ แต่มีความยาวคงที่ และไม่สามารถแก้ไขข้อมูลในทูเปิ้ลได้ ข้อมูลในทูเปิ้ลอาจจะเป็น `number, string, boolean` อย่างใดอย่างหนึ่งหรือผสมกันก็ได้

ตัวอย่างการสร้างทูเปิ้ล

```
t1 = (1, 2)
t2 = (3.14, 'abc', True)
```

สร้างทูเปิ้ลจากตัวแปร

```
x = 10
y = 20
p1 = (x, y) # จุดในระบบพิกัดฉาก 2 มิติ
```

การอ่านค่าจากทูเปิ้ล

```
t = (3.14, 'abc', True)
a, b, c = t
```

`a` และ `t[0]` จะมีค่าเท่ากับ `3.14`

`b` และ `t[1]` จะมีค่าเท่ากับ `'abc'`

`C` และ `t[2]` จะมีค่าเท่ากับ `True`

ส่งทูเปิ้ลเป็นพารามิเตอร์ให้ฟังก์ชันได้

```
def test(t):
  a, b, c = t
  print(t[0], t[1], t[2])

test((3.14, 'abc', True))
```

Dictionary Methods

Python has a set of built-in methods that you can use on dictionaries.

Method	Description
clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and value
get()	Returns the value of the specified key
items()	Returns a list containing a tuple for each key value pair
keys()	Returns a list containing the dictionary's keys
pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
setdefault()	Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

```
'''ตรวจสอบฟังก์ชัน is equal'''
assert is_equal([1.0,2.0],[4.0,-5.1,-99.2]) == False
```

Test case ที่เป็น assert ใช้ตรวจสอบว่านิสิตได้เขียนฟังก์ชันที่โจทย์กำหนดไว้หรือไม่ เช่น กรณีนี้ต้องมีฟังก์ชัน is_equal() และถ้าใส่พารามิเตอร์เป็น [1.0,2.0] และ [4.0,-5.1,-99.2] แล้วจะต้องได้ return value เป็น False

Test case แบบนี้จะช่วยให้นิสิตได้คะแนนจากการเขียนโค้ดฟังก์ชันได้ถูกต้อง ถึงแม้ว่าโปรแกรมหลักจะทำงานผิด นิสิตก็ยังได้คะแนนบ้าง

ประกาศ

คาบเรียนที่ตรงกับวันหยุด ให้นิสิตทำแล็บส่งตามปกติ หากมีข้อสงสัยให้ถามทางกลุ่ม LINE หรือช่องทางอื่น ๆ ที่อาจารย์ผู้สอนแจ้งไว้

วันสอบมิดเทอม/ไฟนัล

ดูในประมวลรายวิชาและติดตามประกาศบน MyCourseVille

สอบมิดเทอม/ไฟนัล
นำโน้ต 1 แผ่น A4
หน้า/หลัง
เข้าได้ 1 แผ่น

ปีการศึกษา 2566/ภาคการศึกษาต้น

ทวิภาค

2301172 COMP PROG LAB

ปฏิบัติการคอมพิวเตอร์และการโปรแกรม

COMPUTER AND PROGRAMMING LABORATORY

คณะวิทยาศาสตร์ (ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์)

1.0 CREDIT HOURS = NL23 1.0 CR

(LAB 2.0 HR)

เงื่อนไขรายวิชา : COREQ 2301170

๑๐๐ ปี คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
100th Anniversary of The Faculty of Science, CU

หลักสูตรและโปรแกรมการศึกษา

เงื่อนไขรายวิชา

1. **รายวิชาที่ต้องสอบผ่าน (Prerequisite-Prer)** หมายถึง รายวิชาที่ต้องสอบผ่านก่อนการลงทะเบียนเรียนรายวิชานั้นๆ

ถอนวิชาแล็บ (1172) ไม่ต้องถอนวิชาเลคเชอร์ (1170)

2. **รายวิชาบังคับร่วม (Corequisite : Coreq)** หมายถึง รายวิชาที่ต้องเรียนก่อน หรือเรียนพร้อมกับรายวิชานั้น (ในกรณีที่เรียนก่อน ไม่จำเป็นต้องสอบผ่าน)

3. **รายวิชาควบ (Concurrent : Concur)** หมายถึง รายวิชาที่ต้องเรียนพร้อมๆ กับรายวิชานั้น

4. **รายวิชาที่คณะอนุญาตให้เรียน (Consent of Faculty: C.F.)** หมายถึง รายวิชาที่ต้องได้รับอนุญาตจากคณะก่อนการลงทะเบียน