for loop

for

Using range

- Function range
- for … in range(…)

Files

- Open and close files
- Read a line from a file
- For loop to read each line from a file

Using range

Function range

for … in range(…)

Function range

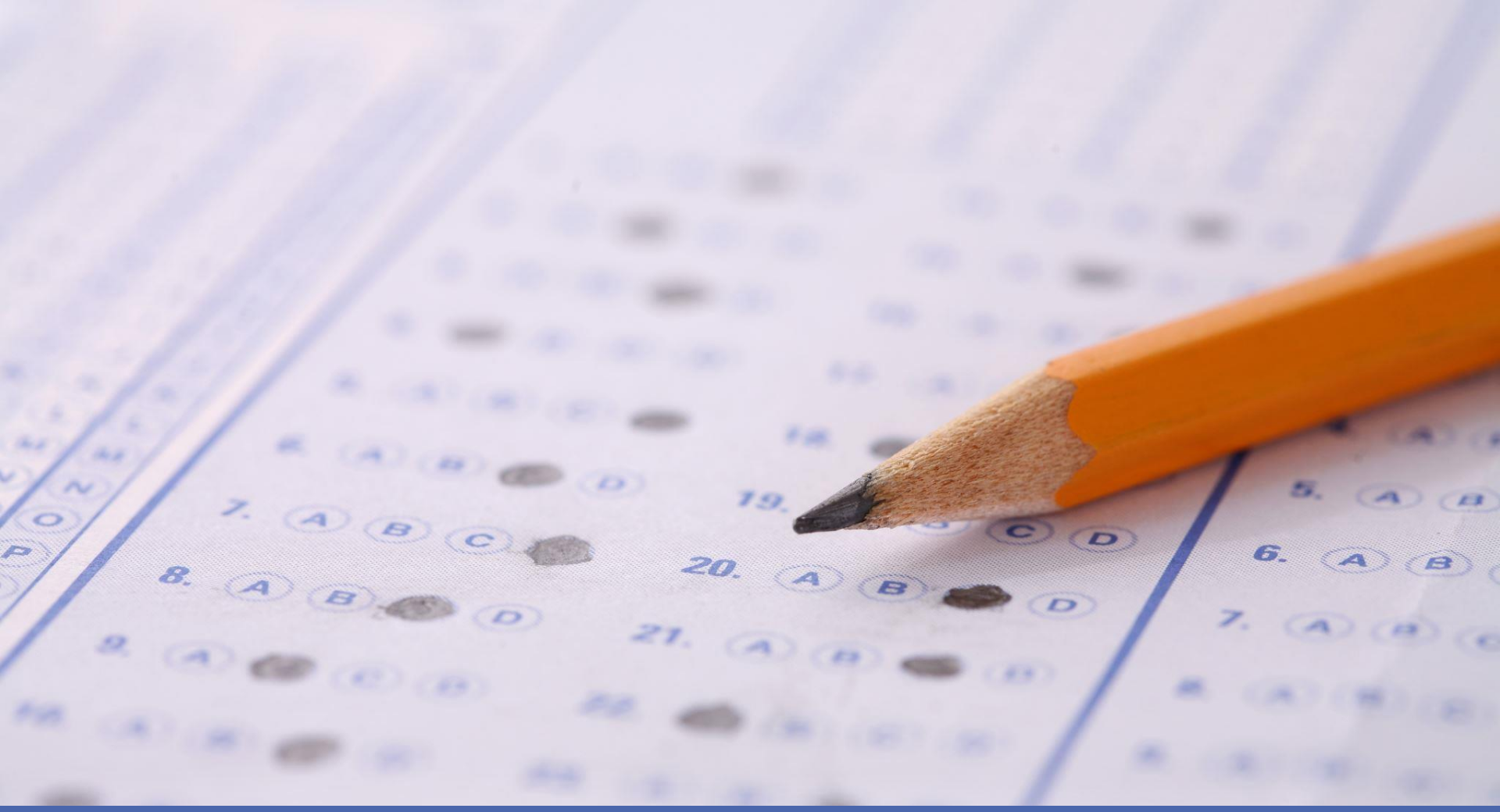
ฟังก์ชัน range

- ฟังก์ชัน **range** เป็นฟังก์ชันแจงนับ ของจำนวนเต็ม โดยจะแจกแจงจำนวนตั้งแต่ค่าเริ่มต้น (start) ไปจนถึงจำนวนเต็มสุดท้ายก่อนค่าขอบปลาย (end)
- ค่าที่ได้อยู่ในช่วง [start, end)
- **range**(end) เริ่มต้นที่ 0 และสิ้นสุดที่ end-1
- **range**(start, end) เริ่มต้นที่ start และสิ้นสุดที่ end-1
- **range**(start, end, step) เริ่มต้นที่ start ค่าเพิ่มขึ้นทีละ step และสิ้นสุดก่อนถึง end

ตัวอย่าง range

- **range**(end) เริ่มต้นที่ 0 และสิ้นสุดที่ end-1
- **range**(start, end) เริ่มต้นที่ start และสิ้นสุดที่ end-1
- **range**(start, end, step) เริ่มต้นที่ start ค่าเพิ่มขึ้นทีละ step และสิ้นสุดก่อนถึง end

- range(10) 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- range(1, 10) 1, 2, 3, 4, 5, 6, 7, 8, 9
- range(1, 10, 2) 1, 3, 5, 7, 9
- range(10, 0, -1) 10, 9, 8, 7, 6, 5, 4, 3, 2, 1
- range(-5, 5) -5, -4, -3, -2, -1, 0, 1, 2, 3, 4
- range(1, 2)
- range(1, -1)
- range(1, -1, -1)



Quiz

Quiz

คำสั่งต่อไปนี้ได้ผลลัพธ์เป็นกลุ่มของจำนวนเต็ม จงแจกแจงสมาชิกในกลุ่มตามลำดับ

range(1)

range(1, 1)

range(0)

range(-10)

range(1,10,3)

range(10,1)

range(10,1,-2)

for … in range(…)

โครงสร้าง **for**

- โครงสร้าง **for** เป็นการวนซ้ำโดยที่การทำงานแต่ละรอบจะดึงสมาชิกในข้อมูลแบบกลุ่ม(collection) ออกมา
- การทำงานมีลักษณะเป็นการวนซ้ำ มีตัวแปรหนึ่งตัว สำหรับเชื่อมกับค่าของสมาชิกในกลุ่ม
- ในแต่ละรอบของการวนซ้ำ ตัวแปรเชื่อมค่าจะเปลี่ยนการเชื่อมค่ากับสมาชิกไปทีละตัว จากตัวแรกไปจนถึงตัวสุดท้าย
- ดัชนี 0 ถึงดัชนี **len**(กลุ่ม) – 1 (ส่วนนี้ เก็บไว้เรียนในบท list)

ฟังก์ชัน **range** กับโครงสร้าง **for**

```
for i in range(10):
        ...
```

อ่านได้ว่า สำหรับ i ในช่วง [0,10)

ในแต่ละรอบ ตัวแปร i จะเชื่อมกับค่า 0 1 2 3 4 5 6 7 8 9

ตามลำดับ และจบการทำงาน

```
for i in range(1, 11, 2):
        ...
```

ในแต่ละรอบ i จะเชื่อมกับค่า

การนับรอบ

สามารถใช้ for ร่วมกับ range ทำงานได้เหมือนกับการเขียน while แบบ loop n times

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

```
for x in range(6):
        print (x)
```

จะมีการวนซ้ำทั้งหมด 6 รอบ และจบการทำงาน ในแต่ละรอบ ตัวแปร x จะมีค่า 0 - 5 ตามลำดับ

```
x=1
while x<=5:
        print (x)
        x=x+1
```

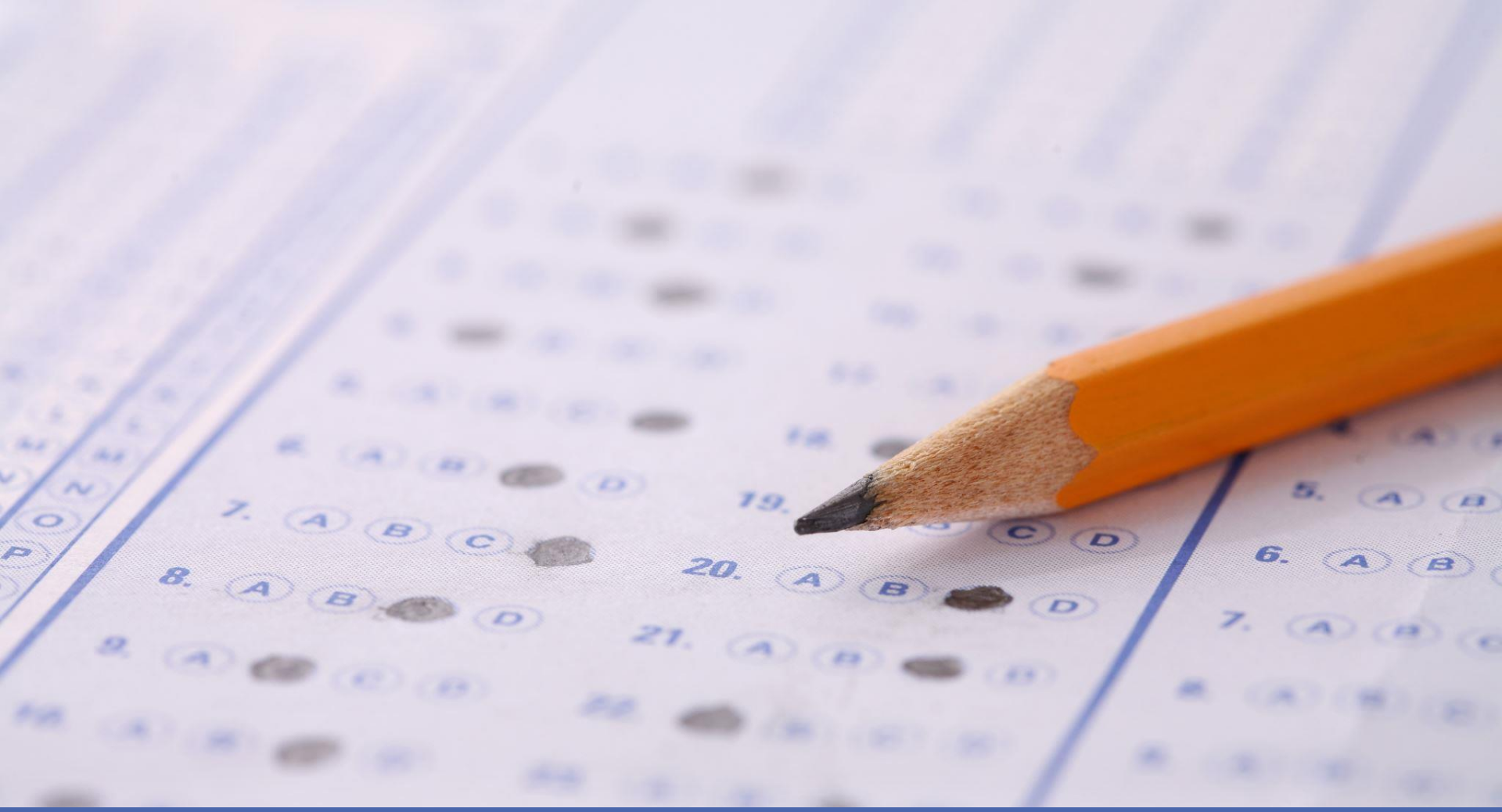
```
for x in range(1, 6):
        print (x)
```

จะมีการวนซ้ำทั้งหมด 5 รอบ และจบการทำงาน ในแต่ละรอบ ตัวแปร x จะมีค่า 1 - 5 ตามลำดับ

ตัวอย่าง : คำนวณผลรวมของจำนวนเต็มตั้งแต่ 1 ถึง 100 แสดงเป็นผลลัพธ์

ตัวอย่าง : รับค่า n แล้วคำนวณหาค่า n! แสดงเป็นผลลัพธ์

ตัวอย่าง : พิมพ์สูตรคูณแม่ 1-12 แสดงเป็นผลลัพธ์



Quiz

Quiz

1. จงเขียนโปรแกรมเพื่อหาผลบวกของอนุกรม

1-2+3-4+5-6+7-8+9-10 แสดงเป็นผลลัพธ์

(ผลลัพธ์ที่ได้คือ -5)

2. จงเขียนโปรแกรมเพื่อรับค่าจำนวนแถวเพื่อพิมพ์สามเหลี่ยมดังรูป เช่น 4 แถว แสดงผลลัพธ์เป็น

1

22

333

4444

การเปลี่ยนค่าตัวแจงนับของ **for**

- ในแต่ละรอบของการวนซ้ำ ตัวแปรเชื่อมค่าจะเปลี่ยนการเชื่อมค่ากับสมาชิกไปทีละตัว จากตัวแรกไปจนถึงตัวสุดท้าย
- ซึ่งจะเป็นการกำหนดค่าเตรียมไว้แล้วจนจบการทำงานของลูป
- หากภายในลูป มีการเปลี่ยนค่าของตัวแปรเชื่อมค่า เมื่อการทำงานย้อนกลับขึ้นไปที่ต้นลูป โปรแกรมจะใช้ค่าสมาชิกที่มันได้เตรียมแจงนับไว้ล่วงหน้าแล้ว เพื่อทำงานต่อไปตามปกติ

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

```
for x in range(1, 6):
    print (x)
    x = 10
    print('[', x, ']')
```

การเปลี่ยนค่าตัวแจงนับของ **for**

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

ผลรัน

```
1
2
3
4
5
```

```
for x in range(1, 6):
        print (x)
        x = 10
        print('[', x, ']')
```

ผลรัน

```
1
[ 10 ]
2
[ 10 ]
3
[ 10 ]
4
[ 10 ]
5
[ 10 ]
```

Files

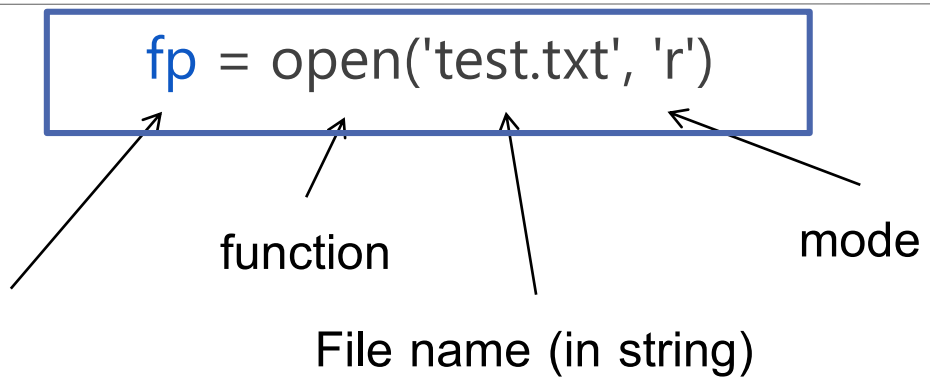
Open and close files

Read a line from a file

Write a string to a file

For loop to read each line from a file

การเปิดและปิดไฟล์



หลังจากคำสั่งนี้ อ้างถึงไฟล์ด้วยตัวแปรนี้

เปิด (open) ไฟล์ ก่อนใช้

Mode: r : read only ถ้าไม่ใส่ไว้ โปรแกรมจะถือว่า เปิดไฟล์เพื่ออ่านอย่างเดียว

w : write only (เขียนทับข้อมูลที่มีอยู่ก่อน)

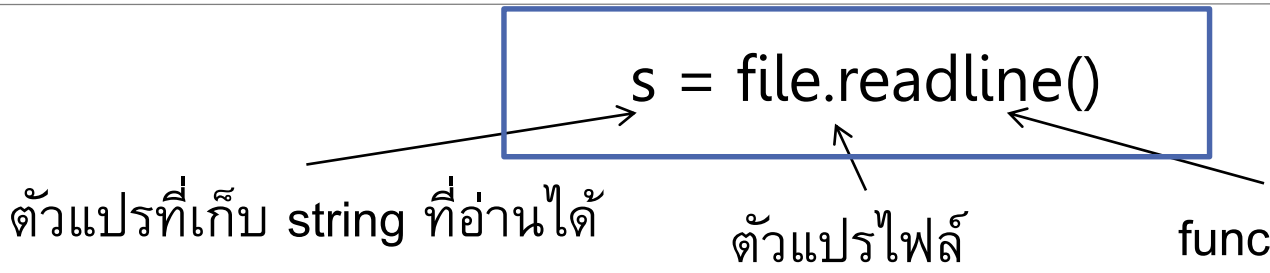
a : append (เขียนต่อท้ายจากข้อมูลที่มีอยู่ก่อน)

r+ : read and write

```
fp.close()
```

ปิด (close) ไฟล์ เมื่อเลิกใช้

การอ่านข้อมูลทีละบรรทัดจากไฟล์



test.txt

```
This is a test.\n
12345678\t9\n
```

```
fp = open('test.txt', 'r')
str1 = fp.readline()
str2 = fp.readline()
str3 = fp.readline()
fp.close()
```

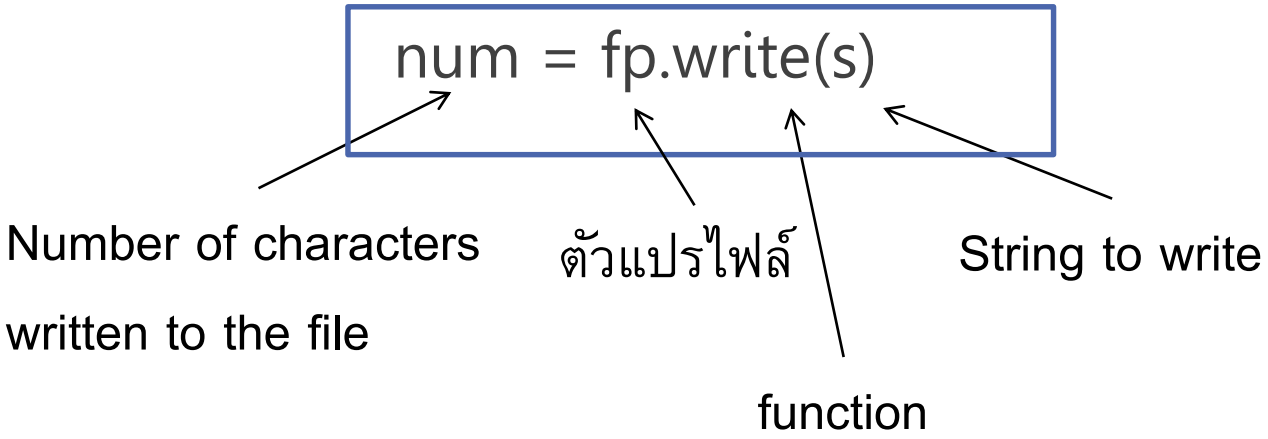
ค่าในตัวแปร

str1: This is a test.\n

str2: 12345678\t9\n

str3:

การเขียนสตริงเก็บลงไฟล์



เขียนสตริงลงในไฟล์ เมื่อ write ครั้งต่อไป จะเขียนต่อจากครั้งก่อน

- ถ้าเปิดไฟล์ด้วย mode w จะเริ่มเขียนจากต้นไฟล์ โดยทับข้อมูลเก่าที่มีอยู่
- ถ้าเปิดไฟล์ด้วย mode a จะเขียนต่อท้ายไฟล์เดิม

ตัวอย่าง

test.txt (before)	abcd\n

```
file = open('test.txt', 'w')
n1 = file.write('1234')
n2 = file.write('321\n')
n3 = file.write('abc')
file.close()
```

test.txt (after)	1234321\n abc

```
file = open('test.txt', 'a')
n1 = file.write('1234')
n2 = file.write('321\n')
n3 = file.write('abc')
file.close()
```

test.txt (after)	abcd\n 1234321\n abc

โครงสร้าง **for** กับการอ่านไฟล์

```
for line in file:
        ....
```

ตัวแปร line เก็บสตริงหนึ่งบรรทัดที่อ่านมาจากไฟล์ที่อ้างถึงด้วยตัวแปร file

- โครงสร้าง **for** ที่ใช้กับการอ่านไฟล์จะเป็นการวนซ้ำโดยที่การทำงานแต่ละรอบจะดึงข้อมูลในไฟล์ออกมาทีละบรรทัด
- การทำงานมีลักษณะเป็นการวนซ้ำ มีตัวแปรหนึ่งตัว สำหรับเชื่อมกับข้อมูลแต่ละบรรทัดในไฟล์
- ในแต่ละรอบของการวนซ้ำ ตัวแปรเชื่อมค่าจะเปลี่ยนการเชื่อมค่ากับข้อมูลในไฟล์ทีละบรรทัดจากบรรทัดแรกไปจนถึงบรรทัดสุดท้าย

ทบทวนฟังก์ชัน strip() ของสตริง

```
<สตริง1>.strip(<สตริง2>)
```

เราสามารถละ <สตริง2> เมื่อต้องการให้ลบ whitespace

```
<สตริง1>.strip()
```

ตัดช่องว่างทั้งหมดก่อนและหลังข้อความ

ช่องว่าง ได้แก่ เคาะวรรค แท็บ และอักขระระบุการขึ้นบรรทัดใหม่

ทบทวนตัวอย่างการใช้งาน strip

strip คือการตัดอักขระว่างต้นและท้ายข้อความ

เมื่ออ่านข้อความจากไฟล์

```
for line in f:
        print(line)
        print(line.strip())
```

testStrip.txt

```
Header
Number 1
Number 2
Number 2.1
```

Output

```
Header

Header
Number 1

Number 1
Number 2

Number 2
Number 2.1
Number 2.1
```

ทบทวนฟังก์ชัน split() ของสตริง

```
<สตริง1>.split(<สตริง2>)
```

การทำงานของฟังก์ชัน split

แบ่ง <สตริง1> เป็นสตริงย่อย โดยใช้ <สตริง2> เป็นตัวแบ่ง

```
<สตริง1>.split()
<สตริง1>.split(',')
<สตริง1>.split('23')
```

ผลลัพธ์ของ split เป็น List ของสตริง ในที่นี้ ยังไม่ได้เรียนเรื่อง list จึงขอใช้ ดังตัวอย่างต่อไปนี้

```
sname = ' John \t  Smith '
fn,ln = sname.split()
```

จะได้ fn='John', ln='Smith'

ทบทวนตัวอย่างการใช้งาน split

การใช้งาน split ที่พบบ่อย คือการตัดข้อความด้วยอักขระคั่น เช่น

ข้อความหนึ่งแถวประกอบด้วยหลายคอลัมน์คั่นคอลัมน์ด้วยการเคาะวรรค

line = ’5931234523 2301170 A’

print(line.split())

จะได้ผลลัพธ์เป็นลิสต์

[’5931234523’, ’2301170’, ’A’]

เราใช้อย่างง่ายก่อน

id, crs, grade = line.split()

ตัวอย่าง : นับจำนวนวิชาที่ลงทะเบียนและจำนวนหน่วยกิตรวม

อ่านข้อมูลการลงทะเบียนจากไฟล์ register.txt โดยมีตัวอย่างข้อมูลดังนี้

2301117 4

2301170 3

2301172 1

2304103 3

ตัวอย่าง : นับจำนวนวิชาที่ลงทะเบียนและจำนวนหน่วยกิตรวม

```
f = open('register.txt')
cnt = 0
total = 0
for line in f:
    id,credit = line.split()
    cnt = cnt + 1
    total = total + int(credit)
f.close()
print('register', cnt, 'courses')
print('total credit : ', total)
```

register.txt

```
2301117 4
2301170 3
2301172 1
2304103 3
```

ผลการรันโปรแกรม

```
register 4 courses
total credit :  11
```

แบบฝึกหัด

1. จงเขียนผังงานและโปรแกรมเพื่อหาผลบวกของอนุกรม

$$\frac{1}{2}-\frac{2}{3}+\frac{3}{4}-\frac{4}{5}+\frac{5}{6}-\frac{6}{7}+\frac{7}{8}-\frac{8}{9}+\frac{9}{10}$$

แสดงเป็นผลลัพธ์ (ผลลัพธ์ที่ได้คือ 0.645635)

แบบฝึกหัด

2. ยอดเงินฝากถอน

กำหนดแฟ้มข้อมูลเข้าชื่อ bankbalance.txt เก็บข้อมูลเป็นการฝากและถอนเงิน บรรทัดละหนึ่งรายการ โดยการฝากเงินจะเป็นจำนวนจริงบวก และการถอนเงินเป็นจำนวนจริงลบ

ตัวอย่างข้อมูลภายในไฟล์

1000

-300

500

-1000

2.1 จงคำนวณยอดฝากเงินรวม

2.2 จงคำนวณยอดถอนเงินรวม

2.3 จงคำนวณยอดเงินสุทธิที่คงเหลืออยู่ในบัญชี

แบบฝึกหัด

3. คำนวณหาจำนวนนิสิตที่ได้รับทุนแบบต่าง ๆ และจำนวนงบประมาณที่ต้องใช้

กำหนดแฟ้มข้อมูลเข้าชื่อ grant.txt เก็บข้อมูลการให้ทุนนิสิต ประกอบด้วย รหัสนิสิต การได้ทุนค่าเทอม การได้ทุนค่าใช้จ่ายบรรทัดละหนึ่งคน

ตัวอย่างข้อมูลภายในไฟล์

6134355723 Y N

6134388823 Y Y

6232227823 N N

6232553423 Y Y

6332345623 N Y

6345535223 N Y

ทุนค่าเทอม คือ 21000 ทุนค่าใช้จ่ายคือ 8000 บาท ตามลำดับ

ผลการรันโปรแกรม

grant both : 2
grant tuition fee : 1
grant expense : 2
Not grant : 1
budget spent : 95000

แบบฝึกหัด ปรับจากโจทย์ข้อ 3 (เปลี่ยนข้อมูลในไฟล์)

4. คำนวณหาจำนวนนิสิตที่ได้รับทุนแบบต่าง ๆ และจำนวนงบประมาณที่ต้องใช้

กำหนดแฟ้มข้อมูลเข้าชื่อ grant2.txt เก็บข้อมูลการให้ทุนนิสิต ประกอบด้วย รหัสนิสิต การได้ทุนค่าเทอม การได้ทุนค่าใช้จ่าย บรรทัดละหนึ่งคน

ตัวอย่างข้อมูลภายในไฟล์

6134355723,21000,0

6134388823,21000,8000

6232227823,0,0

6232553423,21000,8000

6332345623,0,8000

6345535223,0,8000