



Samsung Innovation Campus

| Coding, Programming & Data Science

Together for Tomorrow!
Enabling People

Education for Future Generations

Chapter 3.

Effective Python Programming

- Function, Closure และ Class

Coding, Programming & Data Science

Chapter Description

៤ គ្រឿងការងារសំខាន់សំខាន់

- ✓ ផ្សេងៗសាមាតាំនៅក្នុងខ្លួន និង ពេកអាជីវកម្មប្រចាំថ្ងៃ គឺជាបញ្ហាដែលត្រូវបានដោះស្រាយ។ ក្នុងការបង្កើតកម្មវិធី និងការរៀបចំកម្មវិធី គឺជាបញ្ហាដែលត្រូវបានដោះស្រាយ។

Chapter contents

- ✓ Unit 17. Function
 - ✓ Unit 18. Recursion Function Call
 - ✓ Unit 19. Lambda
 - ✓ Unit 20. Closure
 - ✓ Unit 21. Class

Unit 17.

Function

ឧបតម្លៃកម្មវិធី

- ✓ មិត្តភាពទូទៅនៃកម្មវិធីខ្លួន និង សាមាតត្រូវ user-defined functions.
- ✓ សាមាតត្រូវខ្លួនដោយប្រើប្រាស់ statement និង ចំណាំខ្លួនដោយប្រើប្រាស់ខ្លួន.
- ✓ សាមាតត្រូវជាប្រព័ន្ធដែលអាចរក្សាទុកដាក់ការងារបាន។
- ✓ ខ្លួន និង សាមាតប្រើប្រាស់ខ្លួន ដើរបានការងារបាន។
- ✓ សាមាតប្រើប្រាស់ខ្លួនជាប្រព័ន្ធដែលអាចរក្សាទុកដាក់ការងារបាន និង បានប្រើប្រាស់ខ្លួនជាប្រព័ន្ធដែលអាចរក្សាទុកដាក់ការងារបាន នៅពេលទូទៅនៃកម្មវិធីខ្លួន។

ພາບລວມຂອງປິດຮຽນ

- ✓ ຮຽນຮູ້ກ່ຽວກັບການນຳໃຊ້ built-in functions.
- ✓ ເຂົ້າໃຈກ່ຽວກັບ user-defined function ແລະ ການກຳນົດມັນ.
- ✓ ຮຽນຮູ້ ກຳນົດ ແລະ ເຄີ່ນໃຊ້ user-defined function ໂດຍນຳໃຊ້ def statement.
- ✓ ຮຽນຮູ້ກ່ຽວກັບຕົວດຳເນີນການເພື່ອຜ່ານຄ່າເຂົ້າໄປຟ້າຂັ້ນ.
- ✓ ຮຽນຮູ້ກ່ຽວກັບປະເພດ type errors ທີ່ເກີດຂຶ້ນ ເວລາໃຊ້ຕົວດຳເນີນການ ແລະ ຈັດການກັບມັນເວລາມີ errors ເກີດຂຶ້ນ.
- ✓ ຮຽນຮູ້ກ່ຽວກັບການຜ່ານຄ່າທີ່ມີປະສິດທິຜົນ ໂດຍນຳໃຊ້ຕົວດຳເນີນການຕົວປ່ຽນ, ຕົວດຳເນີນການເລີ່ມຕົ້ນ ແລະ ຕົວດຳເນີນການ keyword.
- ✓ ຮຽນຮູ້ function return statements ໂດຍນຳໃຊ້ return.

សំខាន់ជាប្រព័ន្ធឌីជីថល

- ✓ ការណាំໃຊ់ធម្មតានំ input និង output.
- ✓ ការណាំໃຊ់ main string methods ទីផ្សារៗ ជាដឹងស្រាវជ្រាវ និងការរាយការណ៍ ដូចជា join, split និង format methods, ឬអមពីរណាំໃຊ់ Python built-in functions ទីផ្សារៗ ដូចជា sum និង len.
- ✓ ការណាំໃຊ់ loop statements និងការណាំໃຊ់ break, continue ដើម្បីបញ្ចប់ការវិភាគនៃ control loops.

Keywords

Function

Parameter

Type Error

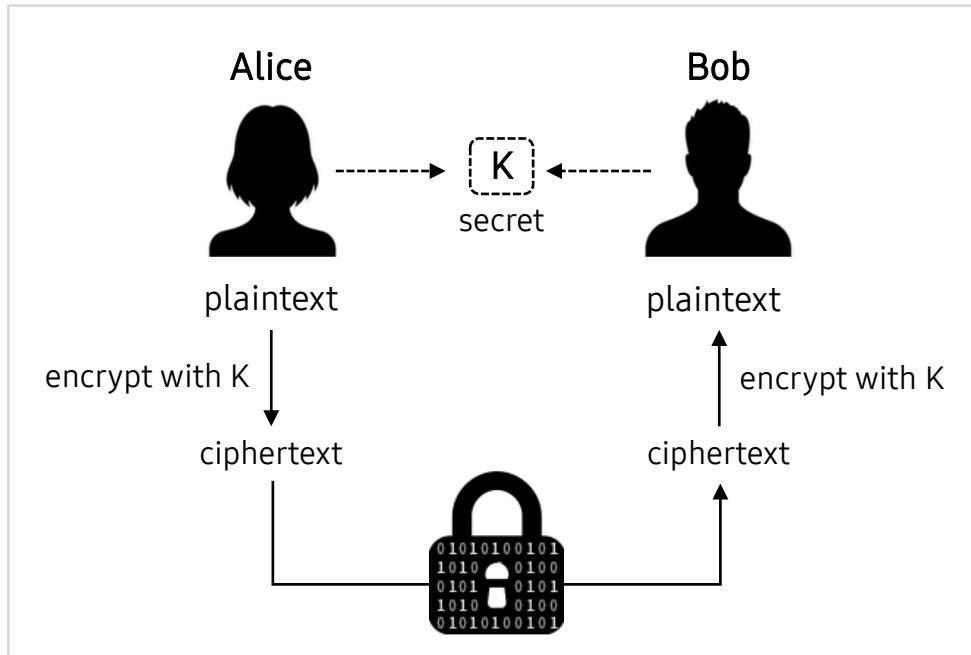
Variable, Default,
Keyword Parameter

Return

Mission

1. บันຫາຂອງໄລກແຫ່ງຄວາມເປັນຈິງ

1.1 ໄລກແຫ່ງບຸກດີຈິດຕອນ ແລະ ການເຂົ້າລະຫັດ



- ໃນໄລກແຫ່ງບຸກດີຈິດຕອນ, ມັນມີຄວາມສໍາຄັນຫຼາຍໃນການພິສູດຕົວຕືນ ແລະ ເຂົ້າເຖິງຂໍ້ມູນພວກເຮົາ.
- ໃນຂະນະດຽວກັນ, ການຂໍມູນຂໍ້ທາງດ້ານດີຈິດຕອນກາຍເປັນບັນຫາຮ້າຍແຮງ, ເຊັ່ນ: ການປອມແປງຕົວຕືນຂອງບຸກຄົນ ຫຼື ການເຂົ້າເຖິງຂໍ້ມູນສາຫາລະນະ ໂດຍບໍ່ໄດ້ຮັບອໍານາດຕາມຊອບທຳ.
- ເນື້ອສອງຄົນມີການສໍ້ສານກັນແບບສ່ວນຕົວ ເຊິ່ງຄົນທັງສອງນີ້ຢູ່ຄົນລະ ສະຖານທີ່, ພວກເຂົາ ສາມາດສື່ງ ແລະ ຮັບຂໍ້ມູນທີ່ຢູ່ໃນຮູບແບບການເຂົ້າ ລະຫັດ.

1. ບັນຫາຂອງໂລກແຫ່ງຄວາມເປັນຈິງ

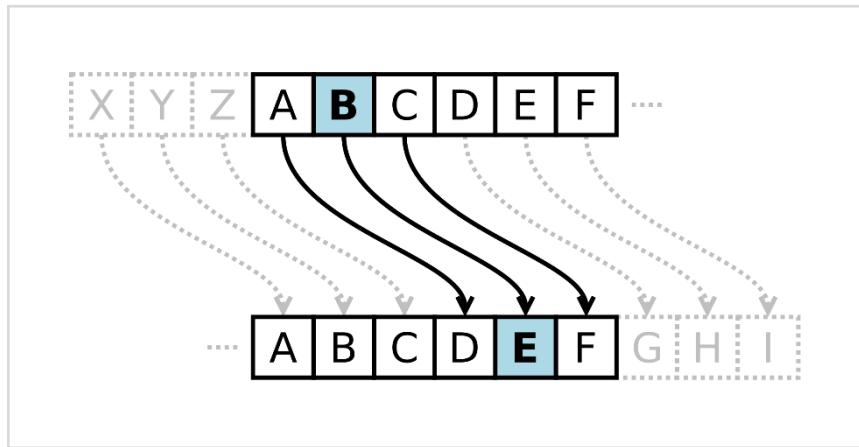
1.1 ໂລກແຫ່ງບຸກດີຈິດຕອນ ແລະ ການເຂົ້າລະຫັດ



- ▶ ເມື່ອມີການ log in ເຂົ້າສູ່ອິນເຕີເນັດ, ເວັບໄຊ ຈະໃຫ້ຜູ້ໃຊ້ປ້ອນລະຫັດຜ່ານ ແລະ ອື່ນໆເພື່ອພິສູດຕົວຕິນ.
- ▶ ໃນອິນເຕີເນັດ ແລະ ລະບົບການສື່ສານສ່ວນໜາຍແມ່ນ “ເຂົ້າລະຫັດ” ກ່ອນມີການແລກປ່ຽນຂໍ້ມູນ.
- ▶ ເຂົ້າລະຫັດຂໍ້ມູນດ້ວຍວິທີການເຂົ້າລະຫັດ Caesar cipher method.
- ▶ ກຽນຮູ້ວິທີການກຳນົດ ແລະ ເອັນໃຊ້ຟັງຊັນເພື່ອແກ້ໄຂບັນຫາ.

1. บันทາຂອງໂລກແຫ່ງຄວາມເປັນຈິງ

1.2. Caesar Cipher

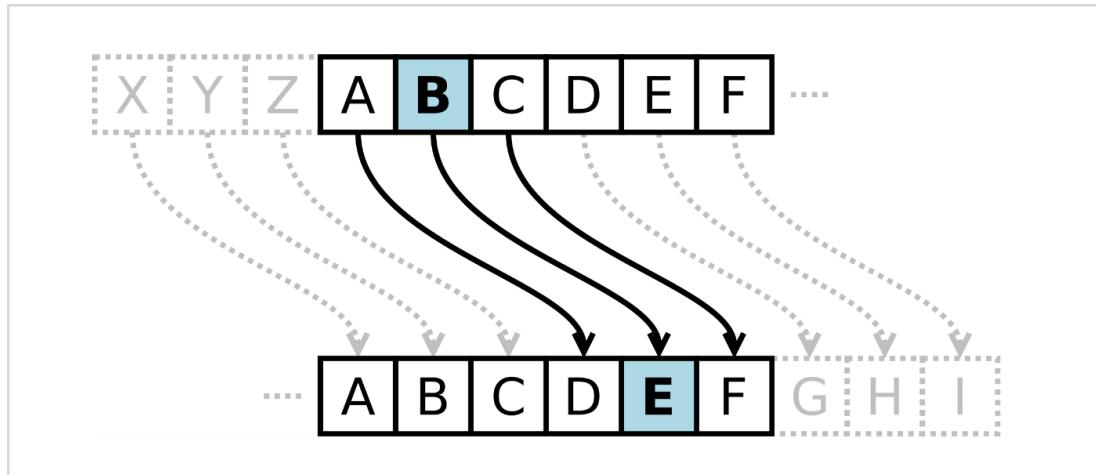


- ▶ ລະຫັດ Caesar ແມ່ນປະເພດລະຫັດການທິດແທນທີ່ງ່າຍໃນການເຂົ້າລະຫັດ.
- ▶ ໃນຄວາມເປັນຈິງ Caesar ແລະ Roman emperor ຍັງນຳໃຊ້ລະຫັດ Caesar. ລະຫັດ Caesar ແມ່ນວິທີການເຂົ້າລະຫັດດ້ວຍການປ່ຽນແທນ alphabet.
- ▶ ຕົວຢ່າງ, ຖ້າຕ້ອງການເຂົ້າລະຫັດຂໍຄວາມ 'COME TO ROME' ດ້ວຍ Caesar ຈະ ເຮັດໃຫ້ຂໍຄວາມດັ່ງກ່າວປ່ຽນເປັນ 'FRPH WR URPH'.
- ▶ ລະຫັດ Caesar ແມ່ນຖືກສ້າງປະມານ 100 BC. ມັນແມ່ນລະຫັດທີ່ສ້າງດ້ວຍ Roman general Caesar ເພື່ອສ້າງກັບ allies.
- ▶ ລະຫັດ Caesar ເປັນລະຫັດທີ່ງ່າຍ ແລະ ມັນຖືກນຳໃຊ້ໂດຍຄົນທ່ວໄປ. ເຖິງຢ່າງໃດກໍ ຕາມ, ແຕ່ມັນກຳມົດອ່ອນ ເນື່ອງຈາກມັນສາມາດແກ້ໄຂໄດ້ງ່າຍໂດຍການໃຊ້ຄວາມທີ່ຂອງການສະກິດຄໍາ ແລະ ຄໍາທີ່ໃຊ້ເລື່ອຍ່າງ.

1. ບັນຫາຂອງໄລກແຫ່ງຄວາມເປັນຈິງ

1.2. Caesar Cipher

- | ຮຽນເພີ່ມຕີມກ່ຽວກັບ Caesar cipher ໄດ້ທີ່ເວັບໄຊ https://en.wikipedia.org/wiki/Caesar_cipher.
- | ເຕັກນິກການເຂົ້າລະຫັດນີ້ ເປັນໜຶ່ງໃນເຕັກນິກການເຂົ້າລະຫັດທີ່ເກົ່າແກ່ທີ່ສຸດ.



1. បំណងការកិច្ចការណ៍ទៅកាន់ការបែងចុះ

1.3. Web Service សំគាល់ការងារ Caesar Cipher

- | បាន web services ដែលមែនជាផ្លូវការកិច្ចការណ៍ទៅកាន់ការបែងចុះ Caesar cipher.
- | តើតុលាតាមរបៀបនៃ plain text ដឹងទីនេះ ciphertext.

The screenshot shows a web-based Caesar cipher tool with three main sections:

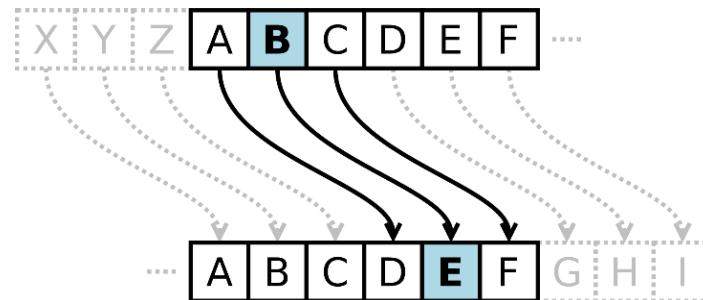
- Enter plain text:** A text input field containing "ATTACK TONIGHT". Below it is a dropdown menu set to "Plaintext" and a "VIEW" button.
- Enter substitution rules:** A central panel titled "Caesar cipher". It includes:
 - A "SHIFT" slider currently at 3, with options to increase or decrease the shift value.
 - An "ALPHABET" dropdown showing the standard English alphabet: abcdefghijklmnopqrstuvwxyz.
 - "CASE STRATEGY" dropdown set to "Maintain case".
 - "FOREIGN CHARS" dropdown set to "Include".
 - A message at the bottom stating "→ Encoded 14 chars".
- Return ciphertext:** A text output field displaying "DWWDFN WRQLJKW". Below it is a dropdown menu set to "Ciphertext" and a "VIEW" button.

<https://cryptii.com/pipes/caesar-cipher>

2. Mission

2.1 ส້າງ Caesar Cipher ໂປຣແກຣມ

- | ນໍາໃຊ້ຕັກນິກການເຂົ້າລະຫັດ Caesar ເພື່ອສ້າງໂປຣແກຣມເຂົ້າລະຫັດຂໍ້ຄວາມ.
- | ເພື່ອແກ້ບັນຫາ, ໃຫ້ import string ແລະ ໃຫ້ ascii_uppercase ດັ່ງໂຄດຄໍາສັງລຸ່ມນີ້.
- | ໃນກໍລະນີນີ້, ໃຫ້ນໍາໃຊ້ ການຕັດຂໍ້ຄວາມ ຫຼື string slicing.



```
1 import string  
2 src_str = string.ascii_uppercase  
3 dst_str = src_str[3:] + src_str[:3]
```

```
src_str = ABCDEFGHIJKLMNOPQRSTUVWXYZ  
dst_str = DEFGHIJKLMNOPQRSTUVWXYZABC
```

2. Mission

2.2. ການປະຕິບັດງານຂອງ Caesar Cipher

```
1 import string
2
3 def cipher(a):
4     idx = src_str.index(a)
5     return dst_str[idx]
6
7 src_str = string.ascii_uppercase
8 dst_str = src_str[3:] + src_str[:3]
9
10 src = input('Enter a sentence: ')
11 print('Encrypted text : ', end='')
12
13 for ch in src:
14     if ch in src_str:
15         print(cipher(ch), end='')
16     else:
17         print(ch, end='')
18
19 print()
```

Enter a sentence: ATTACK TONIGHT!

Encrypted text : DWWDFN WRQLJKW!

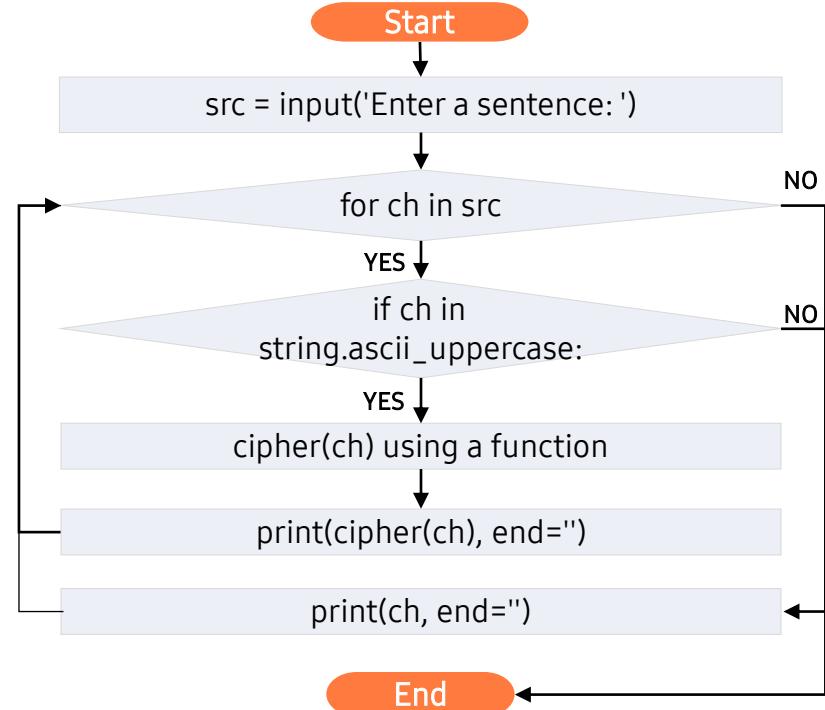
2. Mission

2.3. ການວາງແຜນໂປຣແກຣມ

Pseudocode

```
[1] Start
[2] Enter a sentence
[3] Repeat for the number of characters in the string do{
[4]     if the entered character is uppercase alphabet, then{
[5]         Get the third next order of characters in the
            alphabet you entered
[6]         print the character of ch }
[7]     else{ print the character as it is entered}
[8] End
```

Flowchart



2. Mission

2.4. ໂປຣເກສດ Caesar Cipher: final code

```
1 import string
2
3 def cipher(a):
4     idx = src_str.index(a)
5     return dst_str[idx]
6
7 src_str = string.ascii_uppercase
8 dst_str = src_str[3:] + src_str[:3]
9
10 src = input('Enter a sentence: ')
11 print('Encrypted text : ', end='')
12
13 for ch in src:
14     if ch in src_str:
15         print(cipher(ch), end='')
16     else:
17         print(ch, end=' ')
18
19 print()
```

| Key concept

1. ໂຄງສ້າງພື້ນຖານຂອງຝ່າຍຊັນ

1.1. ຝ່າຍຊັນແມ່ນການສ້າງ blocks ພື້ນຖານຂອງໂປຣແກຣມ

- | ໃນ unit ນີ້, ພວກເຮົາຈະຮຽນກ່ຽວກັບຝ່າຍຊັນທີ່ມີຄວາມສໍາຄັນໃນໂປຣແກຣມ
- | ຝ່າຍຊັນແມ່ນຊຸດຄໍາສັ່ງຂອງໂປຣແກຣມ. ມັນມີລັກສະນະພິເສດໃນການປະຕິບັດງານ.
- | ຝ່າຍຊັນແມ່ນການສ້າງ block ຂອງໂປຣແກຣມ.
- | ຊອບແວທີ່ພວກເຮົາໃຊ້ຈະປະກອບມີທັງ functions ແລະ modules.
- | ຄິດຄົກກັບການປະກອບ Legos ເພື່ອໃຫ້ໄດ້ຮູບຮ່າງທີ່ຕ້ອງງານ, ເຊັ່ນດຽວກັນສໍາລັບການສ້າງໂປຣແກຣມ, ການສ້າງໂປຣແກຣມຂະໜາດໃຫຍ່ ທີ່ມີໂຄດຄໍາສັ່ງຫຼາຍພັນແຖວ, ເຊິ່ງຝ່າຍຊັນຕ້ອງໄດ້ມີການຈັດບັນດາແຖວເລື່ອນັ້ນໃຫ້ເປັນລຳດັບຢ່າງລະອຽດ.



Functions, Modules, and Classes



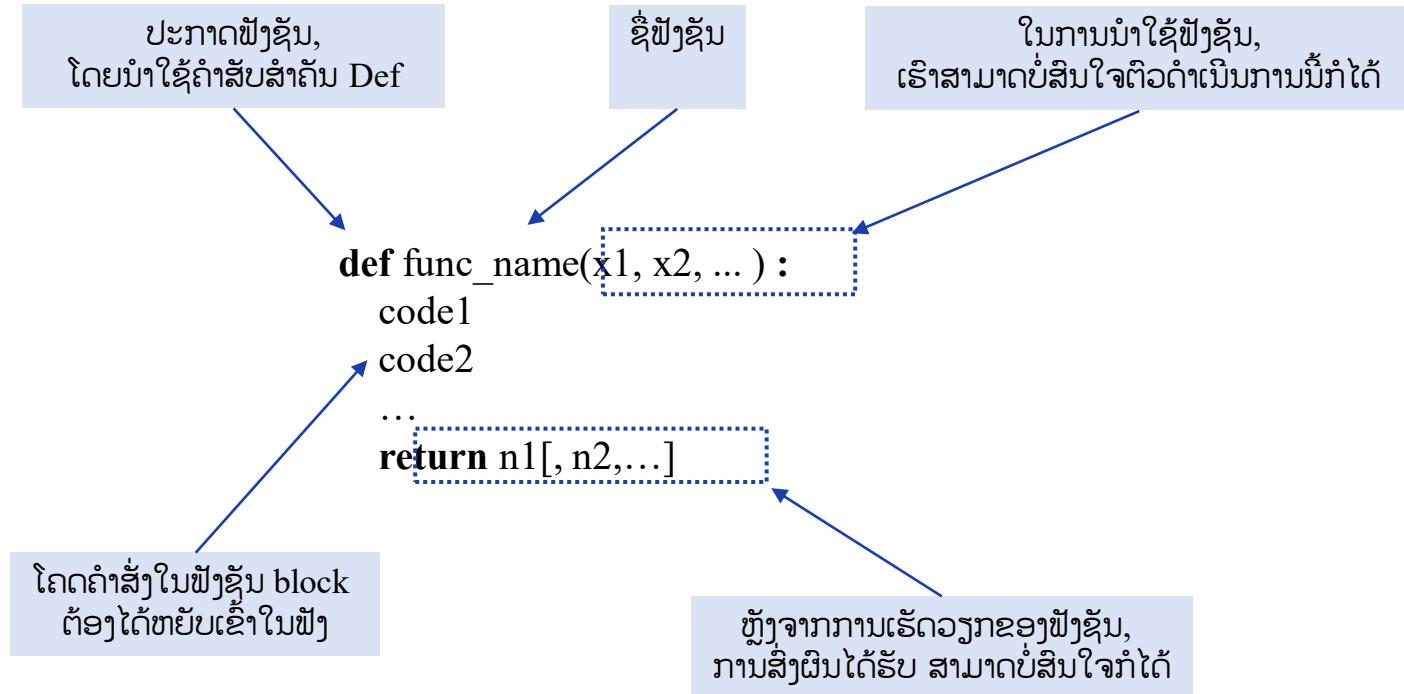
Software

ເຊັ່ນດຽວກັບການປະກອບລົດດ້ວຍ Legos, ພວກເຮົາສາມາດສ້າງໂປຣແກຣມດ້ວຍການລວມຝ່າຍຊັນຕ່າງໆ

1. ໂຄງສ້າງພື້ນຖານຂອງຟັງຊັນ

1.2. ໂຄງສ້າງພື້ນຖານຂອງຟັງຊັນ

| ຮຽນໂຄງສ້າງພື້ນຖານຂອງຟັງຊັນທີ່ຈໍາເປັນໃນໂປຣແກຣມ.



1. ໂຄງສໍາງພື້ນຖານຂອງຝຶກຊັນ

1.3. ວິທີການກຳນົດ ແລະ ເອັນໃຊ້ຝຶກຊັນໃນໂປຣແກຣ

| ຮຽນຮູ້ການກຳນົດຝຶກຊັນທີ່ງໆ ແລະ ເອັນໃຊ້

```
1 def print_star(): # Define a function to output an asterisk
2     print('*'*10)
3
4 print_star()      # Call a function to output an asterisk
```

Line 1, 2, 4

- def print_star(): ເປັນຝຶກຊັນທີ່ນຳໃຊ້ດຳເສັບສຳຄັນ def. ຫຼັງຈຶ່ງໃຊ້ຝຶກຊັນຕ້ອງໄດ້ໃສ່ສອງຈຳ ":" ຕໍ່ທ້າຍ. ເຊິ່ງ ":" ມີຄວາມໝາຍວ່າ ດຳສັ່ງທີ່ຈະ ຂຽນຕ້ອງບັນຈຸຢູ່ໃນ block ມີ. ມັນຄືກັບ if, for, ແລະ while ທີ່ໄດ້ຮູນໃນ unit ຜ່ານມາ.
- ພຶກຊັນ print ທີ່ໃຊ້ໃນ block ຕ້ອງໄດ້ຫຍັບເຂົ້າໄປ ແລະ ມັນຈະປະມວນຜົນພາຍໃນ block ດັ່ງກ່າວ.
- ແຖວທີ່ 4 ເປັນການເອັນໃຊ້ຝຶກຊັນ print_star() ແລະ ປະມວນຜົນ.

1. ໂຄງສ້າງພື້ນຖານຂອງຝຶງຊັນ

1.4. ການເອັນໃຊ້ຝຶງຊັນ

| ພຶງຊັນ print_star ແມ່ນ ກຸ່ມ ຫຼື block ຂອງຂອງຄໍາສັ່ງເພື່ອຮັດວຽກບາງຢ່າງ. ເຮົາສາມາດກວດສອບການຮັດວຽກຂອງກຸ່ມເລື່ອນ໌ໄດ້ຕະຫຼອດ ດ້ວຍການເອັນນີ້ພາຍນອກ block ດັ່ງກ່າວ.

```
1 def print_star():
2     print('*****')
3
4 print_star() # Asterisk output function call 1
5 print_star() # Asterisk output function call 2
6 print_star() # Asterisk output function call 3
7 print_star() # Asterisk output function call 4
```

```
*****
*****
*****
*****
```

Line 1, 2

- ກໍານົດຝຶງຊັນ print_star ເພື່ອພິມດາວ.
- ແຖວທີ 4 ເຖິງ 7 ເປັນການເອັນໃຊ້ຝຶງຊັນ print_star .

1. ໂຄງສ້າງພື້ນຖານຂອງຝຶງຊັນ

1.5. ຂໍ້ດີຂອງຝຶງຊັນ

| ຂໍ້ດີພື້ນຖານຂອງຝຶງຊັນທີ່ຕ້ອງໄດ້ຮຽນ.

1. ຖ້າໂຄງສ້າງຂອງໂປຣແກຣມມີຂະໜາດໃຫຍ່ ເຮົາສາມາດແບ່ງອອກເປັນຍ່ອຍໄດ້.
2. ມັນເປັນໄປໄດ້ທີ່ຈະນຳໃຊ້ຝຶງຊັນຕຽວກັນໄປໃຊ້ໃນໂປຣແກຣມອື່ນໆ.
3. ການອ່ານໂຄດຄໍາສັ່ງແມ່ນມີການເພີ່ມຂຶ້ນ.
4. ເວລາມີການແກ້ໄຂໂປຣແກຣມ ແມ່ນງ່າຍ ເພາະວ່າພຽງແຕ່ແກ້ບາງຝຶງຊັນທີ່ຕ້ອງການ.
5. ທ້າມີການໃຊ້ຝຶງຊັນເພື່ອພັດທະນາໂປຣແກຣມ ຈະຮັດໃຫ້ປະຢັດເວລາ ແລະ ຄ່າໃຊ້ຈ່າຍ.

2. Function Parameter

2.1. Key ពិកអ៉ារីយុទ្ធម៌

| នរណា ត្រូវរាយកាប់ key ដើម្បីពិកអ៉ារីយុទ្ធម៌.

► **Parameter** : ឈរបញ្ជីរាយកាប់ដើម្បីពិកអ៉ារីយុទ្ធម៌ និង method header. មានលក្ខណៈថាអាចបានបញ្ជីរាយកាប់ដើម្បីពិកអ៉ារីយុទ្ធម៌ តុលាតិត្យ។

ពិវឌ្ឍន៍ m និង n នៃ def foo(m, n):

► **Argument** : ឈរបញ្ជីរាយកាប់ដើម្បីពិកអ៉ារីយុទ្ធម៌ និង method តុលាតិត្យ។ មានលក្ខណៈថាអាចបានបញ្ជីរាយកាប់ដើម្បីពិកអ៉ារីយុទ្ធម៌ តុលាតិត្យ។

ពិវឌ្ឍន៍ 3 និង 4 នៃ foo(3, 4)

2. Function Parameter

2.1. Key ពិកអ័យអិរីដ្ឋាន

⌚ **ចិត្ត** ពេលខ្លួនដឹងដ្ឋាន, parameter ត្រូវបានផ្តល់ទៅនូវវត្ថុសំខាន់ដោយ **argument**.

ពិគរម្យ m, n កាបតា 3 និង 4 ទៅសំខាន់ Parameter

```
def foo(m, n):  
    code  
    ...  
    return n1[, n2,...]
```

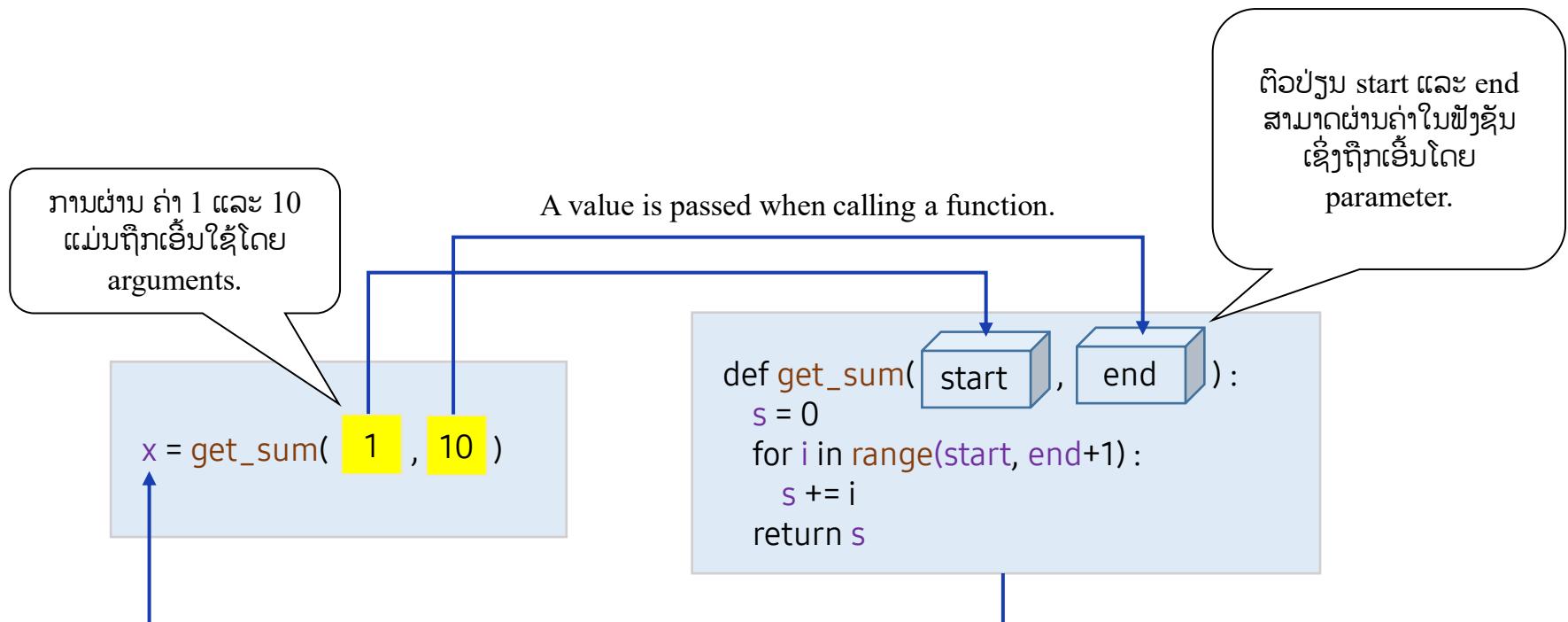
foo(3, 4)

សំខាន់ដោយដ្ឋាន និង ខ្លួន ធ្វើឱ្យ foo 3, 4: Arguments

2. Function Parameter

2.1. Key terms related to functions

- | ຈາກຕົວຢ່າງແມ່ນການສິ່ງຜົນບວກຂອງຟັງຊັນ ທີ່ເປັນຈຳນວນທຸວນຈາກ 1 ເຖິງ 10
 - ▶ ພັງຊັນຈະຄືດໄລ່ຜົນບວກຈຳນວນທຸວນຈາກ 1 ເຖິງ 10 ແລະ ສິ່ງຄ່າກັບ.



ຟັງຊັນຈະຮັດວຽກ ແລະ ສິ່ງຜົນໄດ້ຮັບກັບມາດ້ວຍ return statement.

2. Function Parameter

2.2. ការពិមណេវខ្សែដោយ parameter

- | ស៉ាងដ៏ខ្សែដោយពិមណេវដោយ parameter និងផ្លូវដោយ argument.
- | មើលតាមរបៀបដោយផ្តល់លទ្ធផល

```
1 # Program that repeats the asterisk output for parameter n times
2 def print_star(n):
3     for _ in range(n):
4         print('******')
5
6 print_star(4) # Give the argument value of 4 for asterisk output
```

```
*****
*****
*****
*****
```

Line 2, 3, 4, 6

- ការពិមណេវ parameter n ដោយផ្តល់លទ្ធផល print_star.
- មានចំណាំលេងលេង ចំនួន n នៃក្នុងនិងពិមណេវរបៀបនៅក្បែង.
- ការពិមណេវតាមរបៀបដោយផ្តល់លទ្ធផល print_star.

3. Parameter และ Type Error

3.1. ກໍລະນີ A ມາໃຊ້ parameter

| ຄວບຄຸມຈໍານວນຄັງການສະແດງ ‘Hello’ ໂດຍຜ່ານ argument ດ້ວຍສະແດງລຸ່ມນີ້. n ທີ່ຢູ່ໃນຝັງຂັນແມ່ນ parameter ທີ່ຮັບ argument.

```
1 def print_hello(n):          # Repeat output using parameters
2     print('Hello ' * n)
3
4 print('It prints Hello twice.')
5 print_hello(2)
6 print('It prints Hello 3 times.')
7 print_hello(3)
8 print('It prints Hello 4 times.')
9 print_hello(4)
```

It prints Hello twice.

Hello Hello

It prints Hello 3 times.

Hello Hello Hello

It prints Hello 4 times.

Hello Hello Hello Hello

3. Parameter และ Type Error

3.2. ກໍລະນີ B ນຳໃຊ້ parameter

- | ຊອກຫາຜົນບວກຂອງຄ່າ argument ໂດຍນຳໃຊ້ parameter ແລະ ພິມມັນອອກມາ.
- | ບວກ arguments ທີ່ມີຄ່າເທົ່າ 10, 20 ແລະ 100, 200 ສາມາດຄົດໄລ່ໃນຝັງຊັນ.

```
1 def print_sum(a, b):          # Function with two parameters
2     result = a + b
3     print('The sum of' a, 'and', b, 'is', result)
4
5 print_sum(10, 20)
6 print_sum(100, 200)
```

The sum of 10 and 20 is 30.

The sum of 100 and 200 is 300.

Line 1, 2, 3, 5

- ພັງຊັນ print_sum ມີ Parameter a ແລະ b.
- ກໍານົດຜົນບວກ a + b ເກັບໄວ້ໃນ result ແລະ ພິມມັນອອກມາ.
- ເອັນໃຊ້ພັງຊັນ print_sum ໂດຍ ກໍານົດຄ່າໃຫ້ argument ເປັນ 10 ແລະ 20 ທີ່ສິ່ງໄປພັງຊັນດັ່ງກ່າວ

3. Parameter และ ประเภท Error

3.3. ประเภท error ของ Parameter

- | ถ้าใส่ argument ที่ไม่ต่อเวลาอันให้ฟังชัน, จะเหตุให้เกิด TypeError ขึ้น, และ ฟังชันจะบล็อกการทำงานได้ เมื่อจาก จำนวน parameter บวกกับที่กำหนดไว้ในฟังชัน.
- | กรณีบล็อกหัวใจ argument และ parameter แม่นมีความสำคัญมาก.

TypeError

```
1 def print_sum(a, b):  
2     result = a + b  
3     print('The sum of' a, 'and', b, 'is', result)  
4  
5 print_sum(10) # error
```

ในกรณีนี้, ตัวเล็กของ argument แม่นบล็อกหัวใจ

```
TypeError Traceback (most recent call last)  
<ipython-input-13-ec8bc3c39141> in <module>  
      3     print('The sum of' a, 'and', b, 'is', result)  
      4  
----> 5 print_sum(10) # error  
  
TypeError: print_sum() missing 1 required positional argument: 'b'
```

4. Arbitrary Parameter

4.1. ການຜ່ານ Arbitrary argument

| ດຽວຮູ້ກ່ຽວກັບການຜ່ານ arbitrary argument.

- ▶ arbitrary argument ແມ່ນ argument ທີ່ບໍ່ມີການກຳນົດຈຳນວນ.
- ▶ ຖ້າເຮົາບໍ່ສາມາດກຳນົດຈຳນວນ argument ແມ່ນໃຫ້ໃສ່ເຄື່ອງໝາຍດາວ (*) ໄວ້ເຕີ່ໜ້າ parameter. ຕົວປ່ຽນໃນຝັງຊັນຈະຮັບຄ່າ arbitrary argument.
- ▶ Arbitrary parameter ສາມາດນຳໃຊ້ໃນ for - in statement ຄ້າຍຄືກັບ tuples ແລະ lists.

4. Arbitrary Parameter

4.1. ການຝ່ານ Arbitrary argument

| ສ້າງຟັງຂັ້ນທີ່ປະກອບດ້ວຍ 3 ຫຼື 2 argument ໂດຍຝ່ານ arbitrary argument ດັ່ງສະແດງລຸ່ມນີ້

```
1 def greet(*names):
2     for name in names:
3         print('Hello', name, '!')
4
5 greet('A', 'B', 'C')      # 3 arguments
6 greet('James', 'Thomas')  # 2 arguments
```

```
Hello A !
Hello B !
Hello C !
Hello James !
Hello Thomas !
```

Line 1, 2, 3

- ໃຊ້ arbitrary argument ເຮັນ names.
- ແຕ່ລະອົງປະກອບໃນ names ແມ່ນໃກໝົມ ໂດຍໃຊ້ for statement.

4. Arbitrary Parameter

4.1. ການຜ່ານ Arbitrary argument

| ນັບຈຳນວນຕົວເລກ ຜ່ານ arguments ໂດຍນໍາໃຊ້ຟັງຂັນ len

▶ ລຸ່ມນີ້ເປັນການພິມຕົວເລກຜ່ານ arbitrary argument ເພື່ອຜ່ານຄ່າເຂົ້າໄປດ້ວຍການນໍາໃຊ້ຟັງຂັນ len.

```
1 def foo(*args):
2     print('The number of arguments:', len(args))
3     print('Arguments :', args)
4
5 foo(10, 20, 30)
```

```
The number of arguments: 3
Arguments : (10, 20, 30)
```

Line 1, 2, 3

- arbitrary argument ຖືກເອັນໂດຍ args.
- ພິມຈຳນວນຕົວເລກ ໂດຍໃຊ້ຟັງຂັນ len.
- ພິມອີງປະກອບຂອງ args.
- ຜົນທີ່ໄດ້ຮັບຂອງ arguments ແມ່ນ 10, 20, 30 ໂດຍມີຈະນິດຂຶ້ນເປັນ tuple.

4. Arbitrary Parameter

4.2. ການຄິດໄລ່ຜົນບວກຂອງຕົວເລກ

- | ສ້າງໂປຣແກຣມເພື່ອຄິດໄລ່ຜົນບວກຂອງຈຳນວນ ໂດຍການສຶ່ງຄ່າ argument.
- | ຖ້າສຶ່ງຈຳນວນຜ່ານຟັງຊັນ sum_nums , ພັງຊັນດັ່ງກ່າວນີ້ຈະບໍ່ຮູ້ລວງໝໍາວ່າມີຈຳນວນ arguments ຫັງໝົດເທົ່າໃດ ດັ່ງນັ້ນ arbitrary parameter ເຊັ່ນ: *numbers ຈະຜ່ານ arbitrary arguments ເຂົ້າໄປໃນຟັງຊັນ.

```
1 def sum_nums(*numbers):
2     result = 0
3     for n in numbers:
4         result += n
5     return result
6
7 print(sum_nums(10, 20, 30))      # Print the sum of arguments of 10, 20, 30
8 print(sum_nums(10, 20, 30, 40, 50)) # Print the sum of arguments of 10, 20, 30, 40, 50
```

60
150

Line 1, 2, 3, 4, 5

- ໃນຟັງຊັນ sum_nums, ບັນດາຕົວເລກຖືກເອີ້ນໃຊ້ດ້ວຍ arbitrary parameter.
- ຜ່ານ for loop, ອີງປະກອບຂອງ numbers ແມ່ນເກັບໄວ້ໃນ result.
- Return result.

5. ຄ່າເລີ່ມຕົ້ນຂອງ Parameter

5.1. ຄ່າເລີ່ມຕົ້ນ parameter ແລະ ໂປຣແກຣມ

| ສ້າງໂປຣແກຣມດ້ວຍການກຳນົດຄ່າເລີ່ມຕົ້ນໃຫ້ກັບ parameter. ແຕ່ຖ້າບໍ່ກຳນົດຄ່າເລີ່ມຕົ້ນ ກໍຈະເຮັດໃຫ້ເກີດ error ດັ່ງລຸ່ມນີ້

❗ TypeError

```
1 def print_star(n):      # One argument required
2     for _ in range(n):
3         print('*****')
4
5 print_star() # An error occurs because there is no argument
```

```
TypeError                                     Traceback (most recent call last)
<ipython-input-18-6d1876493ea4> in <module>
      3     print('*****')
      4
----> 5 print_star() # An error occurs because there is no argument

TypeError: print_star() missing 1 required positional argument: 'n'
```



```
1 def print_star(n = 1): # The parameter n has a default value of 1
2     for _ in range(n):
3         print('*****')
4
5 print_star() # Execute without error even if there is no argument
```

ຖ້າຄ່າ default parameter ເທົ່າ 1 , ຈະບໍ່ເກີດ error ຂຶ້ນ.

5. ຄ່າເລີ່ມຕົ້ນຂອງ Parameter

5.2. ການເອັນໃຊ້ຝັງຊັນທີ່ບໍ່ມີ argument

| ຮຽນຮູ້ກ່ຽວກັບການເອັນໃຊ້ຝັງຊັນໄດ້ປັດສະຈາກ argument.

- ▶ ໃນລຳດັບການເຮັດວຽກຂອງຝັງຊັນ, ມັນມີຄວາມຈໍາເປັນທີ່ຈະຕ້ອງໄດ້ຮັບ argument ທີ່ຖືກຕ້ອງ. ບາງຄັ້ງ ຈໍາເປັນຕ້ອງໄດ້ນຳໃຊ້ຄ່າ default ເພື່ອປ້ອງກັນຄວາມຜິດພາດທີ່ຈະເກີດຂຶ້ນ.
- ▶ ໃນກໍລະນີນີ້ default parameter ແມ່ນຖືກນຳໃຊ້
- ▶ ໂຄດຄໍາສັ່ງລຸ່ມນີ້, ເຮົາສາມາດກຳນົດຄ່າ default ເທົ່າກັບ 1 ໃຫ້ກັບ parameter n.

```
1 def print_star(n = 1): # The parameter n has a default value of 1
2     for _ in range(n):
3         print('*'*n)
4
5 print_star() # Execute without error even if there is no argument
```

- ▶ ເຖິງແມ່ນວ່າ ການເອັນໃຊ້ຝັງຊັນຈະບໍ່ມີ argument ກໍຕາມ ຄ່າ default ເທົ່າກັບ 1 ຈະຜ່ານ parameter n ແລ້ວພິມດາວອກມາຫາງໜ້າຈຳຕາມປຶກກະຕິ.

5. ຄ່າເລື່ມຕົ້ນຂອງ Parameter

5.2. ການເອີ້ນໃຊ້ພັງຊັນທີ່ບໍ່ມີ argument



ຖ້າມີການໃສ່ຄ່າ argument, ຈະຮັດໃຫ້ຄ່າ default ຖືກຍິກເລີກ.

- ▶ ຖ້າໃສ່ຄ່າ argument ເທົ່າກັບ 2 ເວລເອີ້ນໃຊ້ພັງ print_star ຈະຮັດໃຫ້ຄ່າ default ເທົ່າກັບ 1 ຖືກຍິກເລີກ, ເນື່ອງຈາກມັນຈະໄປຮັດຄ່າ argument ເທົ່າກັບ 2 ແທນ.
- ▶ ໃນກໍລະນີລຸ່ມນີ້ ຈະພິມດາວອອກມາ 2 ແຫວ້ວ ແຊ່ງຈະຮັດໃຫ້ຄ່າ Default ຖືກຍິກເລີກ.

```
1 def print_star(n = 1): # The parameter n has a default value of 1
2     for _ in range(n):
3         print('*' * n)
4
5 print_star(2) # Since the argument value is 2, the default parameter n=1 is not performed
```

5. ຄ່າເລີ່ມຕົ້ນຂອງ Parameter

5.3. ທິດລອງການນຳໃຊ້ຄ່າເລີ່ມຕົ້ນຂອງ parameter

- | ສ້າງຝັງຊັບໜຶ່ງຂັ້ນມາທີ່ຊື່ div ປະກອບມີ 2 parameter.
- | ກໍານົດຄ່າເລີ່ມຕົ້ນໃຫ້ກັບ parameter ຫີສອງເທົ່າ 2.

```
1 def div(a, b = 2):  
2     return a / b  
3  
4 print('div(4) =', div(4))  
5 print('div(6, 3) =', div(6, 3))
```

```
div(4) = 2.0  
div(6, 3) = 2.0
```

Line 1, 2

- a ຕ້ອງຮັບຄ່າ argument , ແຕ່ b has ມີຄ່າ assigned ເທົ່າກັບ 2 ທີ່ໄດ້ກໍານົດໄວ້ໃນເບື້ອງຕົ້ນແລ້ວ, ຖ້າບໍ່ມີການຮັບຄ່າ argument ໃໝ່ຈະຖືເວົາ ຄ່າທີ່ກໍານົດໃນເບື້ອງຕົ້ນນີ້.
- Returns a / b.
- ກໍານົດຄ່າໃຫ້ argument ເທົ່າ 4, ສ່ວນ b ແມ່ນກໍານົດໃຫ້ເປັນ 2 (ເປັນຄ່າເລີ່ມຕົ້ນ).

5. ຄ່າເລີ່ມຕົ້ນຂອງ Parameter

5.3. ທິດລອງນຳໃຊ້ຄ່າເລີ່ມຕົ້ນຂອງ parameter

 SyntaxError: argument ທີ່ບໍ່ມີຄ່າເລີ່ມຕົ້ນ ຈະປະຕິບັດຕາມ argument ທີ່ມີຄ່າເລີ່ມຕົ້ນ

- ▶ ມັນຈະເກີດຫຍຸງຂຶ້ນຖ້າກໍານົດຄ່າເລີ່ມຕົ້ນໃຫ້ກັບ parameter ບໍ່ເອີດ, ສ່ວນ parameter ທີ່ສອງແມ່ນບໍ່ໄດ້ກໍານົດ?

```
1 def div(a = 2, b):  
2     return a / b
```

```
File "<ipython-input-22-48c2fea5df1e>", line 1  
    def div(a = 2, b):  
           ^
```

```
SyntaxError: non-default argument follows default argument
```

- ▶ ຄ່າເລີ່ມຕົ້ນຄວນກໍານົດໃຫ້ທຸກ parameters ຫຼື ຕົວປ່ຽນທີ່ຢູ່ໃນຝຶງຊັ້ນ.

5. ຄ່າເລີ່ມຕົ້ນຂອງ Parameter

5.3. ທິດລອງນຳໃຊ້ຄ່າເລີ່ມຕົ້ນຂອງ parameter

- | ຮຽນຮູ້ການເອີ້ນໃຊ້ statement ໂດຍພິຈາລະນາຄ່າເລີ່ມຕົ້ນ parameter ແລະ ຄ່າ argument
- | ຈາກຕາຕະລາງລຸ່ມນີ້, `div()` ແມ່ນຄືກັບ `div(1, 2)`. ຕົວເລກສີໜ້າ 1 ແລະ 2 ແມ່ນຄ່າເລີ່ມຕົ້ນຂອງ argument. ເວລາເອີ້ນໃຊ້ຝຶ່ງຊັ້ນຫວ່າງເປົ່າ ໝາຍວ່າບໍ່ມີການກຳນົດຄ່າ argument ໃໝ່ ມັນຈະສົ່ງຄ່າເລີ່ມຕົ້ນອອກມາ.
- | ແຕ່ຖ້າ ກຳນົດຄ່າ argument ໃໝ່ເຊົ້າໄປໃນຝຶ່ງຊັ້ນ ເຊັ່ນ `div(4)` ແລະ `div(4, 2)` ມັນຈະສົ່ງຄ່າດັ່ງກ່າວອອກມາດັ່ງສະແດງລຸ່ມນີ້.

```

1 def div(a = 1, b = 2):
2     return a / b
3
4 print('div() =',div())
5 print('div(4) =',div(4))
6 print('div(6, 3) =',div(6, 3))

```

`div() = 0.5`
`div(4) = 2.0`
`div(6, 3) = 2.0`

Function Call Statement	Arguments passed during actual execution (the blue argument is the default value)	Return Value
<code>div()</code>	<code>div(1, 2)</code>	0.5
<code>div(4)</code>	<code>div(4, 2)</code>	2.0
<code>div(6, 3)</code>	<code>div(6, 3)</code>	2.0

6. Keyword Parameter

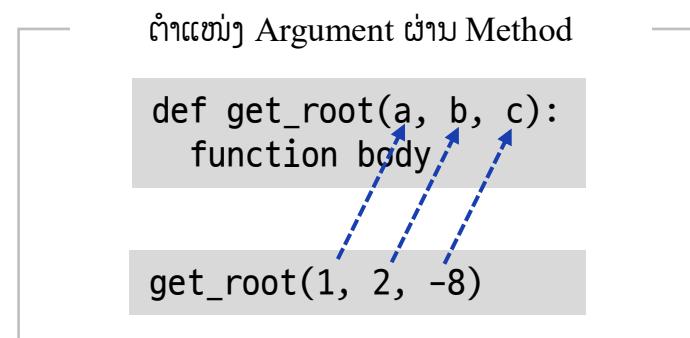
6.1. Keyword parameter และ ການຜ່ານ argument ໃນ Method

I ການຜ່ານ Keyword parameter ໃນ method

- ຄ່າເລີ່ມຕົ້ນຂອງ argument ຜ່ານ method ໃນ Python ແມ່ນວິທີການເອັນດຳແຫ່ງຂອງ argument.

```
1 def get_root(a, b, c):
2     r1 = (-b + (b ** 2 - 4 * a * c) ** 0.5) / (2 * a)
3     r2 = (-b - (b ** 2 - 4 * a * c) ** 0.5) / (2 * a)
4     return r1, r2
5
6 # When calling a function, use a constant value as an argument
7 # Return the result value using result 1 and result 2
8 result1, result2 = get_root(1, 2, -8)
9 print('The result value is', result1, 'or', result2)
```

The result value is 2.0 or -4.0



6. Keyword Parameter

6.2. ສິ່ງທີ່ຈໍາເປັນຂອງ keyword parameter

I ເປັນຫຍັງພວກເຮົາຈຶ່ງຕ້ອງການ keyword parameter.

- ▶ ເວລາເອັນໃຊ້ຝັງຊັນ, ມັນບໍ່ພຽງແຕ່ຜ່ານຄ່າ argument, ມັນຍັງມີການກຳນົດຊື່ສະເພາະ ເຊິ່ງເອັນວ່າ keyword parameter.
- ▶ ໃນໂຄດຄໍາສັ່ງລຸ່ມນີ້, ຖ້າກຳນົດຄ່າ argument ແມ່ນ -8, 2, 1, ຜົນໄດ້ຮັບຈະມີຄວາມແຕກຕ່າງກັນກັບການກຳນົດ 1, 2, -8 .

```
1 result1, result2 = get_root(-8, 2, 1)
2 print('The result value is', result1, 'or', result2)
```

The result value is -0.25 or 0.5



ເວລາສິ່ງຄ່າຜ່ານ parameter, ມັນຈະລຽງຄ່າດັ່ງກ່າວໄປຕາມລຳດັບ a, b, c. ຕໍາແໜ່ງຂອງ argument ແມ່ນມີຄວາມສໍາຄັນຕໍ່ການສິ່ງຄ່າຜ່ານ.

6. Keyword Parameter

6.3. ខ្លឹមខោយ keyword parameter

- Keyword parameter សាមសួរបានចាប់ផ្តើមបានដូចតាំងខាងក្រោម។
 - keyword parameter ແມ័ែនវិធានការណ៍ថា argument នេះទីំនួយត្រូវបានចាប់ផ្តើមដោយនូវឈ្មោះនូវឈ្មោះនៃវិធានការណ៍។

```
1 # All three codes below are a=1, b=2, c=-8, so the result is the same
2 result1, result2 = get_root(a = 1, b = 2, c = -8)
3 print('The result value is', result1, 'or', result2)
```

The result value is 2.0 or -4.0

```
1 result1, result2 = get_root(a = 1, c = -8, b = 2)
2 print('The result value is', result1, 'or', result2)
```

The result value is 2.0 or -4.0

```
1 result1, result2 = get_root(c = -8, b = 2, a = 1)
2 print('The result value is', result1, 'or', result2)
```

The result value is 2.0 or -4.0

- ជឿនខ្លឹមខោយត្រូវបានចាប់ផ្តើមដោយនូវឈ្មោះនូវឈ្មោះនៃវិធានការណ៍។
- ឡើង keyword arguments ពីការណ៍ដែលបានបង្ហាញ។

Positional Argument Passing Method

```
def get_root(a, b, c):
    function body
get_root(a=1, c=-8, b=2)
```

6. Keyword Parameter

6.4. ການນຳໃຊ້ keyword parameter

 **Focus** ສັງເກດການຈັດລຳດັບຂອງຕຳແໜ່ງ ແລະ keyword parameter.

- | ເວລາຜ່ານargument ໄປທີ່ຟັງຊັ້ນໃນ Python ຈະມີ 2 ວິທີ ຄື: ວິທີຜ່ານຕຳແໜ່ງ argument ແລະ ວິທີຜ່ານດ້ວຍ keyword argument. ແຕ່ສາມາດລວມທັງສອງວິທີນີ້ເຂົ້າກັນໄດ້.
- | ເວລາລວມທັງສອງວິທີເຂົ້າກັນ, ຕຳແໜ່ງ argument ຕ້ອງປະຕິບັດຕາມ keyword argument

SyntaxError: ຕຳແໜ່ງຂອງ argument ຕ້ອງປະຕິບັດຕາມ keyword argument

- ▶ ເວລາ keyword argument ແລະ ຕຳແໜ່ງ argument ຫຼືກນຳໃຊ້ຮ່ວມກັນ, ຕຳແໜ່ງຂອງ argument ຕ້ອງປະຕິບັດຕາມ keyword argument. (ນີ້ຮັດດັ່ງລຸ່ມນີ້ ຈະເກີດ error ຂຶ້ນ.)

```
1 # An error occurs when:  
2 result1, result2 = get_root(c = -8, b = 2, 1)  
3 print('The result value is', result1, 'or', result2)
```

```
File "<ipython-input-29-e61e9d57ef12>", line 2  
    result1, result2 = get_root(c = -8, b = 2, 1)  
               ^
```

SyntaxError: positional argument follows keyword argument

6. Keyword Parameter

6.4. ການນຳໃຊ້ keyword parameter

| ປະສົມ keyword argument ແລະ ຕໍາແໜ່ງ argument

- ▶ ເວລານຳໃຊ້ການປະສົມປະສານລະຫວ່າງ keyword arguments ແລະ ຕໍາແໜ່ງ arguments, ຕ້ອງໝັ້ນໃຈວ່າຂຽນຕໍາແໜ່ງກ່ອນ arguments.
- ▶ ກໍລະນີລຸ່ມນີ້, ດ້ວຍຈຳນວດວ່າ keyword argument ກາຍເປັນ -8 ເນື້ອງຈາກ $c = -8$. ຫມາຍວ່າຄ່າຂອງ a ແລະ b ແມ່ນຖືກຜ່ານໂດຍຕໍາແໜ່ງ argument method.

```
1 result1, result2 = get_root(1, 2, c = -8)
2 print('The result value is', result1, 'or', result2)
```

The result value is 2.0 or -4.0

```
def get_root(a, b, c):
    function body
```

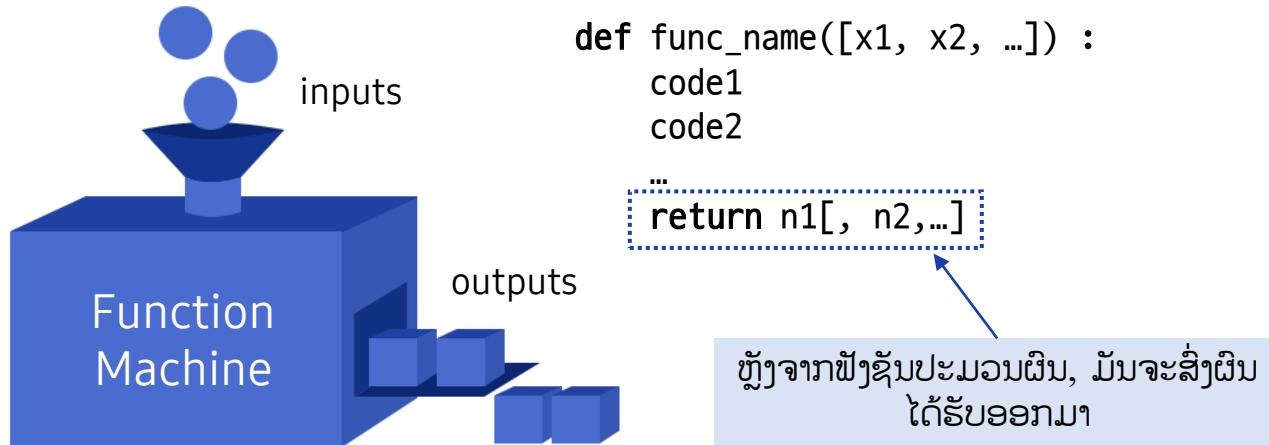
```
get_root(1, 2, c = -8)
```

7. Return

7.1. ការសៀវភៅ statement កំតិការនៃការសៀវភៅ

| ទ្វាននូវការរាយការនៃការសៀវភៅ statement.

- ▶ ត្រួយព័ត៌មាន, សិមមុដថា មានប្រព័ន្ធគ្មាន និងការសៀវភៅ.
- ▶ និងមានសាមាតប្រមុជនិង និងការសៀវភៅ ដែល ត្រួយបានការសៀវភៅ.
- ▶ ពីរប្រព័ន្ធដែលត្រួយបានការសៀវភៅ គឺ មានការសៀវភៅ ដោយប្រើប្រាស់ keyword return.



7. Return

7.1. งานสื้ง statement กำกົດການສ້າງຜົນຮັບ

- ▶ ພັນ get_sum ຈະສິ່ງເປີນບວກຂອງສອງຄ່າທີ່ເກັບໄວ້ໃນ result ອອກມາ
 - ▶ ເຊິ່ງເປີນບວກຂອງສອງຕົວເລກ a ແລະ b ເກັບໄວ້ໃນຕົວປ່ຽນ result ແລະ ສິ່ງ result ຜ່ານ return statement.
 - ▶ n1 ຈະສິ່ງຄ່າ 30, ດ້ວຍການບວກ 10 ແລະ 20, ສ່ວນ n2 ຈະສິ່ງຄ່າ 300, ດ້ວຍການບວກ 100 ແລະ 200.

```
1 def get_sum(a, b):      # A function that returns the sum of two numbers
2     result = a + b
3     return result        # Return the result using the return statement
4
5 n1 = get_sum(10, 20)
6 print('The sum of 10 and 20 =', n1)
7
8 n2 = get_sum(100, 200)
9 print('The sum of 100 and 200 =', n2)
```

The sum of 10 and 20 = 30

The sum of 100 and 200 = 300

7. Return

7.1. ការសៀវភៅ statement កំតិការាសៀវភៅជិនរប

| ប្រើប្រាប់ដៃងីខ្ពស់ get_sum ទីតាំងផិនបែកខូចខែងតែ

- ▶ គោលសៀវភៅតាមណា ធ្វើ return, មានសាមាតសៀវភៅតាមណា ធ្វើ expression ហួយតាំងវិញ return តែងតែលើមនេះ.
- ▶ លើមនេះមែនការណា ធ្វើខ្ពស់ get_sum និង ការសៀវភៅ statement ទាំងមីនាសៀវភៅបែកខូចខែងតាម ឬ expression.

```
1 def get_sum(a, b):  
2     return a + b  
3  
4 result = get_sum(100, 200)  
5 print('The sum of two numbers', result)
```

The sum of two numbers 300

- ▶ ឧបាស័យណាដែលវាទ់ ត្រូវតាំងវិញរបស់ខ្លួន និង ត្រូវតាំងវិញនៅ ត្រូវជិនរបតិវិក។

7. Return

7.2. ການ return statement ກຳສົດ ການ returns tuple

I ການ Return ສອງຄ່າດ້ວຍ tuple

- ອີກຫາງໝຶ່ງໃນການ return ສອງຄ່າດ້ວຍ tuple. ໃນກໍລະນີ້ນີ້, ການ return ດ້ວຍສາມາດຮັບຄ່າທີ່ເປັນ tuple ຈາກຂ້າງນອກໄດ້.

```
1 def get_root(a, b, c):
2     r1 = (-b + (b ** 2 - 4 * a * c) ** 0.5) / (2 * a)
3     r2 = (-b - (b ** 2 - 4 * a * c) ** 0.5) / (2 * a)
4     return r1, r2
5
6 # When calling a function, use a constant value as an argument
7 # Return the result value using result 1 and result 2
8 result1, result2 = get_root(1, 2, -8)
9 print('The result value is', result1, 'or', result2)
```

Returns the two roots r1 and r2 of the quadratic equation.

Result1 and result2 receive the values of r1 and r2.

The result value is 2.0 or -4.0

Line 2, 3

- The quadratic formula is expressed in Python modifiers.
- The quadratic formula is $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$.

Paper coding

- ຕ້ອງເຂົ້າໃຈແນວຄິດພື້ນຖານຂອງຫຼັກສູດນີ້ໃຫ້ຄົບຖ້ວນກ່ອນຈະໄປສູ່ຂັ້ນຕອນຕໍ່ໄປ.
- ການທີ່ບໍ່ເຂົ້າໃຈແນວຄິດພື້ນຖານ ຈະເພີ່ມພາລະໃນການຮຽນຮູ້ຂອງຫຼັກສູດນີ້ ແລະ ຈະຮັດໃຫ້ບໍ່ປະສົບຜົນສໍາເລັດ.
- ມັນອາດຈະເປັນເລື່ອງທີ່ຍາກຕອນນີ້, ແຕ່ຖ້າຢາກປະສົບຜົນສໍາເລັດໄດ້ນັ້ນ ພວກເຮົາຂໍແນະນຳໃຫ້ເຂົ້າໃຈແນວຄິດພື້ນຖານ ນີ້ຢ່າງເລີກເຊິ່ງ ແລະ ກ້າວໄປສູ່ຂັ້ນຕອນຕໍ່ໄປ.

Q1. ส້າງຟັງຊັນທີ່ຊື່ my_greet ເພື່ອພິມ "Welcome." ແລະ ເອັນໄຊຟັງຊັນສອງຄັ້ງເພື່ອພິມຜົນຮັບອອກມາທາງໜ້າຈຳ ດັ່ງສະແດງລຸ່ມນີ້.

Condition for Execution	Welcome. Welcome.
Time	5 minutes

Write the entire code and the expected output results in the note.

Q2.

ສ້າງຝຶກຊັນທີ່ຊື່ max2(m, n) ປະກອບດ້ວຍສອງ parameter ເຊັ່ນ m, n ແລະ returns ຄ່າໃຫຍ່ສຸດຂອງສອງ parameter ດັ່ງກ່າວ, ເຊັ່ນ
ດຽວກັນ ສ້າງ ພຶກຊັນ min2(m, n) ທີ່ມີສອງ parameter ຄື: m, n ແລະ returns ຄ່ານ້ອຍສຸດຂອງສອງ parameter ນີ້. ກຳນົດ 100 ແລະ
200 ໃຫ້ກັບ argument ເພື່ອເຈັ້ນໃຊ້ຝຶກຊັນດັ່ງກ່າວ ແລະ ໄດ້ຜົນດັ່ງສະແດງລຸ່ມນີ້.

Condition for Execution	The greater of 100 or 200 is : 200 The smaller of 100 or 200 is : 100
time	10 minutes



Write the entire code and the expected output results in the note.

Q3.

ພວກເຮົາຕ້ອງການປ່ຽນຄ່າ mile, ເຊິ່ງເປັນຄ່າທີ່ນີ້ຍືມໃຊ້ໃນອາເມລິກາ ໃຫ້ເປັນຄ່າ kilometer, ເຊິ່ງເປັນມາດຕະຖານສາກົນ. ສ້າງຝ່າງຂັ້ນທີ່ຊື່ mile2km(mi) ໃຫ້ຄ່າ mile ເປັນ parameter ແລະ returns ມັນເປັນ kilometers ຈາກນີ້ນ ເຊັ່ນຝ່າງຂັ້ນນີ້ເພື່ອສົງເຜີນໄດ້ຮັບຈາກ 1 ເຖິງ 5 miles ໃຫ້ເປັນ kilometer. ໃນກໍລະນີນີ້ ແມ່ນນຳໃຊ້ for - in range ເພື່ອເຮັດຊໍ້າ. (ກຳນົດ 1 mile ເທົາກັບ 1.61 km.)

Condition for Execution

Condition for Execution	1 mile = 1.61 kilometers 2 mile = 3.22 kilometers 3 mile = 4.83 kilometers 4 mile = 6.44 kilometers 5 mile = 8.05 kilometers
Time	5 minutes



Write the entire code and the expected output results in the note.

Q4.

ສ້າງຝຶ່ງຊັນຊື່ cel2fah(cel) ເພື່ອປ່ຽນອຸນຫະພູມ ອົງສາ Celsius ດ້ວຍ parameter ແລະ returns ມັນເປັນ ອົງສາ Fahrenheit. ຈາກນີ້ເອັນໃຊ້ຝຶ່ງຊັນນີ້ ເພື່ອປ່ຽນອົງສາ ຈາກ 10 ເຖິງ 50 degrees Celsius ແລະ ຜົນໄດ້ຮັບເປັນອຸນຫະພູມ ອົງສາ Fahrenheit ດັ່ງສະແດງຜົນໄດ້ຮັບຄຸ້ມື້.

Condition for Execution	10 degrees Celsius = 50.0 degrees Fahrenheit 20 degrees Celsius = 68.0 degrees Fahrenheit 30 degrees Celsius = 86.0 degrees Fahrenheit 40 degrees Celsius = 104.0 degrees Fahrenheit 50 degrees Celsius = 122.0 degrees Fahrenheit
Time	5 minutes

ສູດຄິດໄລ່ອີງສາ Fahrenheit = (Celsius × 9/5) + 32



• Write the entire code and the expected output results in the note.

| Let's code

1. ການປະມວນຜິນຂອງ String

1.1. ການປະມວນຜິນ functions ແລະ methods

- | ທໍາຄວາມເຂົ້າໃຈ string methods ເລີກເຊິ່ງທີ່ພວກເຮົາໄດ້ຮຽນໃນ unit 14.
- | ຮຽນ count method ທີ່ໃຊ້ໃນ string.
- | ນອກຈາກນີ້, ຍັງສາມາດຊອກຫາຄວາມຍາວຂອງ string ໂດຍນຳໃຊ້ ພັງຊັນ len.
 - ▶ ນັບ string ໂດຍນຳໃຊ້ count method ເພື່ອ returns ຈຳນວນ substring ໃນ string.

```
1 birth = '1998.05.13'      # Date of birth example
2 birth.count('.')           # Tell you how many times '.' occurs
```

2

- ▶ ນອກຈາກທີ່ໄດ້ເວົ້າມາຂ້າງເທິງ, ພັງຊັນ built-in ຍັງສາມາດນຳໃຊ້ສໍາລັບ string. ພັງຊັນ len ຈະ returns ຄວາມຍາວຂອງ string.

```
1 birth = '1998.05.13'
2 print('length of birth :', len(birth))    # Return the length of the string birth
```

length of birth : 10

1. ການປະມວນຜິນຂອງ String

1.2. ການນຳໃຊ້ຝັງຊັນ built-in ເພື່ອປະມວນຜິນ String

- | ເຮົາສາມາດນຳໃຊ້ຝັງຊັນ max ແລະ min ເພື່ອຊອກຫາຄ່າໃຫຍ່ສຸດ ແລະ ຄ່ານ້ອຍສຸດຂອງຕົວອັກສອນໃນ String. ມາດຖານໃຫຍ່ກວ່າ ຫຼື ນ້ອຍກວ່າ ແມ່ນ ອີງໃສ່ Unicode ຂອງ code value.
- | Unicode ແມ່ນລະບົບລະຫັດມາດຕະຖານອຸດສະຫະກຳ ເພື່ອກຳນົດຕົວອັກສອນທີ່ໃຊ້ໃນຄອມພິວເຕີ.
- | ພັງຊັນ ord ສາມາດຮັບຄ່າ Unicode ສໍາລັບຕົວອັກສອນ, ຄ່າ chr ຈະ returns ຕົວອັກສອນທີ່ສອດຄ່ອງກັບຄ່າ Unicode ທີ່ປ້ອນເຂົ້າໄປ.

```
1 birth = '1998.05.13'  
2 max(birth)
```

'9'

```
1 ord(max(birth)) # Return a Unicode value of 9
```

57

```
1 ord(min(birth)) # Return the Unicode value of the smallest Unicode value within the string birth
```

46

```
1 chr(57), chr(46) # Return the character corresponding to Unicode values 57 and 46
```

('9', '.')

1. ការបំរុលសំណង់ String

1.3. string method ទីនៅ

| នូវភារណ៍ string methods ពាយ្យ និង ការបំរុល។

```
1 a = 'I love python'  
2 a.count('o')      # Number of occurrences of the 'o' character
```

2

```
1 a.count('k')      # Number of occurrences of the 'k' character
```

0

```
1 a.find('o')      # Return the index when the 'o' character appears
```

3

```
1 a.find('on')     # Return the index when the 'on' character appears
```

11

```
1 a.startswith('I') # Does it start with the character 'I'?
```

True

```
1 a.endswith('python') # Does it end with the character 'python'?
```

True

1. ການປະມວນເຜີນຂອງ String

1.3. string method ອື່ນຕູ

| string methods ສໍາລັບ ຕົວອັກສອນໃຫຍ່/ຕົວອັກສອນນ້ອຍ

```
1 a = 'I Love Python'  
2 g = 'i love python'  
3 h = 'I LOVE PYTHON'
```

```
1 a.upper()      # Convert all uppercase  
'I LOVE PYTHON'
```

```
1 h.lower()      # Convert all lowercase  
'i love python'
```

```
1 a.swapcase()   # Convert uppercase to lowercase and lowercase to uppercase  
'i LOVE pYTHON'
```

```
1 a.istitle()    # Is it a title text?
```

True

1. ການປະມວນຜົນຂອງ String

1.4. ນໍາໃຊ້ໂມດຸນ string

| ພິມຕົວອັກສອນໃຫຍ່/ນ້ອຍໂດຍນໍາໃຊ້ string module

- ▶ Python ມີໂມດຸນທີ່ເອີ້ນວ່າ string ທີ່ຊ່ວຍໃນການປະມວນຜົນຂອງ string. ໃນໂມດຸນນີ້ ascii_uppercase ບັນຈຸ ຕົວອັກສອນໃຫຍ່ ແລະ ascii_lowercase ບັນຈຸຕົວອັກສອນນ້ອຍ.
- ▶ ພິມຕົວອັກສອນໃຫຍ່ໃນພາສາອັງກິດ ໂດຍນໍາໃຊ້ໂມດຸນ string.

```
1 import string  
2 src_str = string.ascii_uppercase  
3 print('src_str =', src_str)
```

src_str = ABCDEFGHIJKLMNOPQRSTUVWXYZ

- ▶ ພິມຕົວອັກສອນນ້ອຍໃນພາສາອັງກິດ ໂດຍນໍາໃຊ້ ໂມດຸນ string.

```
1 src_str = string.ascii_lowercase  
2 print('src_str =', src_str)
```

src_str = abcdefghijklmnopqrstuvwxyz

1. ການປະມວນເຜີນຂອງ String

1.4. ນໍາໃຊ້ ໂມດຸນ string

| ທິດລອງຕັດຕົວອັກສອນ

- ຖ້າຂໍ້ຄວາມ 'ABCDE..YZ' ຖືກຍ້າຍໄປເທົ່ອລະຕົວອັກສອນ, ຈະໃຊ້ວິທີໄດ້ເພື່ອສ້າງຂໍ້ຄວາມນີ້ 'BCDEF...ZA'?
- ໃນກໍລະນີນີ້, ພວກເຮົາສາມາດແກ້ໄຂບັນຫານີ້ດ້ວຍຕົວຢ່າງລຸ່ມນີ້.

```
1 src_str = string.ascii_uppercase  
2 dst_str = src_str[1:] + src_str[:1]  
3 print('dst_str =', dst_str)
```

dst_str = BCDEFGHIJKLMNOPQRSTUVWXYZA

Line 1, 2

- ກຳນົດຕົວອັກສອນໃຫຍ່ A ເຖິງ Z ໃຫ້ກັບຕົວປ່ຽນ src_str.
- ຕັດເອົາຕົວອັກສອນທີ່ຢູ່ຕຳແໜ່ງຂຶ້ມູນທີ 0 ເຊັ່ນ [:1] ຈະໄດ້ຕົວອັກສອນ A ແລະ [1:] ແມ່ນຕັດເອົາຕົວອັກສອນທີ່ຢູ່ຕຳແໜ່ງຂຶ້ມູນທີ 1 ຂັ້ນໄປ ຈະໄດ້ຕົວອັກສອນ B ເຖິງ Z ຈາກນີ້ເອົາມາລວມກັນ ດັ່ງນີ້ src_str[1:] + src_str[:1] ແລະ ຈະໄດ້ເຜີນຮັບເປັນດັ່ງສະແດງໃນໂຄດຄຳສັ່ງນີ້.

1. ການປະມວນຜິນຂອງ String

1.5. ນໍາໃຊ້ index method

- ຕໍາແໜ່ງຂອງຕົວອັກສອນ A ໃນ src_str ສາມາດຄົ້ນຫາໄດ້ ດ້ວຍ index method.
- ໃນຫາງົງກິງກັນຂ້າມ, ຖ້າຕົວປ່ຽນ n ນໍາໃຊ້ index ນີ້ ເພື່ອຄົ້ນຫາ dst_str ຕໍາແໜ່ງຂໍ້ມູນ, ມັນສາມາດຢືນຢັນໄດ້ວ່າຕົວອັກສອນຂອງ dst_str ສອດຄ່ອງກັບຕໍາແໜ່ງ 'A' ຂອງ src_str is 'B'.

```
1 n = src_str.index('A')
2 print('The index of src_str =', n)
3 print('The character in dst_str at position A in src_str =', dst_str[n])
```

The index of src_str = 0

The character in dst_str at position A in src_str = B

2. ឧបករាយវិធីរៀងដែលបានប្រើប្រាស់នូវសមិទ្ធភាពជាប្រព័ន្ធ

2.1. រៀងដែលបានប្រើប្រាស់នូវសមិទ្ធភាព

- | តាមរបៀបនេះមានវិធីរៀងដែលបានប្រើប្រាស់នូវសមិទ្ធភាពជាប្រព័ន្ធ។ សមឮត្តតាមរយៈរៀងនេះ។
- | ប៉ុណ្ណោះតាមរយៈរៀងនេះ a ≠ 0, ដូចតាំ b និង c មិនអាចសមិទ្ធភាពបាន។
- | សមឮត្តតាមរៀងដែលបានប្រើប្រាស់នូវសមិទ្ធភាពជាប្រព័ន្ធ។

$$ax^2 + bx + c = 0$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

```

1 def print_root(a, b, c):
2     r1 = (-b + (b ** 2 - 4 * a * c) ** 0.5) / (2 * a)
3     r2 = (-b - (b ** 2 - 4 * a * c) ** 0.5) / (2 * a)
4     print('The solution is', r1, 'or', r2)
5
6 # Solve quadratic equations with different coefficient values
7 print_root(1, 2, -8)
8 print_root(2, -6, -8)

```

The solution is 2.0 or -4.0

The solution is 4.0 or -1.0

- ផ្តល់នៅក្នុង arguments តាមរយៈ 3 parameter គឺ a, b និង c, និង ឲ្យបញ្ជូនតាមរយៈ r1 និង r2 នៃការបង្ហាញ។
- ការបង្ហាញនេះនឹងបង្ហាញការបង្ហាញនៃការបង្ហាញនេះ។

3. Multiple Return Statement

3.1. returns លើខ្លួន និង perimeter ខោចុះមិនិងផ្ទាល់ខ្លួន

- | បានដាក់តាមលក្ខណៈខ្លួន ដើម្បីទទួលឱ្យបានលើខ្លួន.
- | ផ្ទាល់ខ្លួន The circle_area_circum(radius) រួមម៉ោងសារខ្លួនមានផ្ទាល់ខ្លួន radius ទៅ 10 ដើម្បីបង្កើត parameter ខោចុះមិនិងផ្ទាល់ខ្លួន និង return តម្លៃ area និង cirum.

```
def circle_area_circum(radius):  
    area = 3.14 * radius ** 2  
    circum = 2 * 3.14 * radius  
    return area, circum
```

3. Multiple Return Statement

3.1. returns เม็ดที่ และ perimeter ของวงรีมินในฟังก์ชัน

- | กาน Return สອງ ຫຼື ຫ້າຍຄ່າ ເຕັມວ່າ multiple return statement.
 - | ໃນ multiple return statement, ຄ່າທີ່ return ແມ່ນຂັ້ນດ້ວຍຈຸດ ເຊິ່ງເປັນການ return ຫຼືປະເພດ tuple.

```
1 def circle_area_circum(radius):
2     area = 3.14 * radius ** 2
3     circum = 2 * 3.14 * radius
4     return area, circum # Returns a tuple - the return value (area, circum)
5
6 radius = 10
7 area, circum = circle_area_circum(radius) # Return the area and perimeter of a circle
8 print("The area of a circle of radius {} is {:.1f}, and the circumference of the circle is {:.1f})".format(radius, area, circum))
```

The area of a circle of radius 10 is 314.0, and the circumference of the circle is 62.8

 BYTE Line 1, 7-8

- ในโຄດຄໍາສັ່ງຂ້າງເທິງນີ້, ພັຈັນ `circle_area_circum` ຮັບ `radius` ທີ່ເປັນ parameter ແລະ `returns` ສອງຄ່າ ທີ່ເປັນເນື້ອທີ່ ແລະ ລວງຮອບຂອງວົງມິນ.
 - ຢູ່ນອກພັຈັນ ໃນເຖິງທີ່ 7-8, ພັຈັນ `circle_area_circum` ແມ່ນຖືກເອັນ ແລະ ຄິດໄລ່ຜົນຮັບເກັບໄວ້ໃນຕົວປ່ຽນ `area` ແລະ `circum` ພ້ອມທັງພິມ ມັນອອກໂນໂຍງຫຼັມຈໍາພາບ.

4. ពិវប្បធម៌ Global

4.1. រាយកិត្តភារវិវាបីពិវប្បធម៌ global

| នរណា នឹងរាយកិត្តភារវិវាបីពិវប្បធម៌ global.

- ▶ ពិវប្បធម៌ global មែនជាប្រព័ន្ធដែលអាចប្រើបានបានចំណាំ និងបានចំណាត់ថ្នាក់ នៃការប្រើបាន និងប្រើបាន នៅក្នុងខ្លួន។
- ▶ តាមសេចក្តីណាមត្រ និងរាយកិត្តភារវិវាបីពិវប្បធម៌ global នឹងត្រូវបានគោរព នៅក្នុង Python script file និងប្រើបាន នៅក្នុង print_sum.

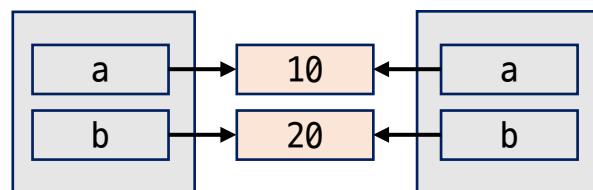
```

1 def print_sum():
2     result = a + b
3     print('print_sum() inside : the sum of', a, 'and', b, 'is', result)
4
5 a = 10
6 b = 20
7 print_sum()
8 result = a + b
9 print('print_sum() outside : the sum of', a, 'and', b, 'is', result)

```

print_sum() inside : the sum of 10 and 20 is 30
print_sum() outside : the sum of 10 and 20 is 30

Global variables a and b can be used in the entire code.



Python
script file

print_sum function
using global variables

4. ពិវប្បធន Global

4.1. នេរគតិការវរកបពិវប្បធន global

| ពិវប្បធន Global និងពិវប្បធនការណ៍សំខែ

- ▶ ផ្តល់មិភាពបច្ចាស់ពិវប្បធនជាប់នៃការបង្កើតការណ៍សំខែ។ តើអ្វីដោយទាំងនេះ?
- ▶ បង្ហាញនេរគតិការវរកបពិវប្បធន global និងពិវប្បធនការណ៍សំខែ។

```
1 def print_sum():
2     a = 100
3     b = 200
4     result = a + b
5     print('print_sum() inside : the sum of', a, 'and', b, 'is', result)
6
7 a = 10
8 b = 20
9 print_sum()
```

```
print_sum() inside : the sum of 100 and 200 is 300
```

4. ពិវប្បធម៌ Global

4.1. នេរគិតការរៀងរាល់ពិវប្បធម៌ global

| ប្រើប្រាស់ពិវប្បធម៌ global និង កវតសរបតែតាំង

- ▶ ប្រើប្រាស់ពិវប្បធម៌ global a និង b ទីយុទ្ធសាស្ត្រ និង កវតតាមខ្លួនរបស់ខ្លួន.

```
1 def print_sum():
2     a = 100
3     b = 200
4     result = a + b
5     print('print_sum() inside : the sum of', a, 'and', b, 'is', result)
6
7 a = 10
8 b = 20
9 print_sum()
10 result = a + b
11 print('print_sum() outside : the sum of', a, 'and', b, 'is', result)
```

ការណិតតាមខ្លួនដែលធ្វើឡើងក្នុងពិវប្បធម៌ a និង b តិ 100, 200

ការណិតតាមខ្លួនដែលធ្វើឡើងក្នុងពិវប្បធម៌ a និង b តិ 10 និង 20

```
print_sum() inside : the sum of 100 and 200 is 300
print_sum() outside : the sum of 10 and 20 is 30
```

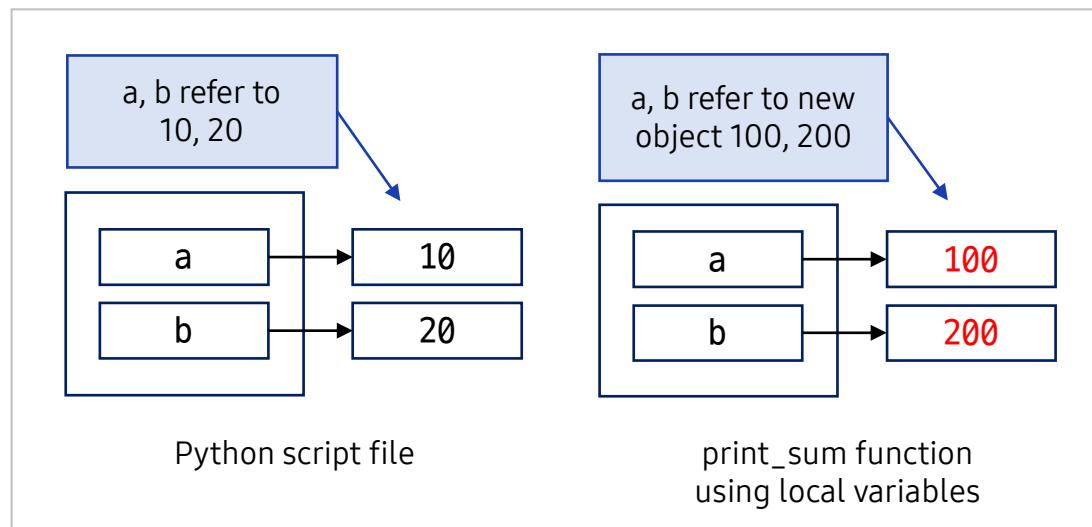
- ▶ ចូលរួមជាមួយនឹង print_sum បានពិចារណា, ការណិតតាមខ្លួនដែលធ្វើឡើងក្នុងពិវប្បធម៌ a និង b និង ពិនិត្យពីរបៀបរាយការណិតតាមខ្លួន, និង ពិនិត្យពីរបៀបរាយការណិតតាមខ្លួន.
- ប្រើប្រាស់ពិវប្បធម៌ a និង b ដើម្បី 100, 200 ទីយុទ្ធសាស្ត្រ
- ▶ កវតសរបតែតាមខ្លួន, តាមពិវប្បធម៌ a និង b ទីយុទ្ធសាស្ត្រ និង ពិនិត្យពីរបៀបរាយការណិតតាមខ្លួន.

4. ពិវប្បធម៌ Global

4.2. របៀបគិតខាងពិវប្បធម៌ local

 Focus ជួយដាក់ទីនឹង និងការពិភាក្សាអម្ចាយខាងក្រោម។

- | មិនអាចពិវប្បធម៌ : $a = 100, b = 200$
- | ឬយើងដាក់ទីនឹង, មិនអាចពិវប្បធម៌ local ពីកសាងខ្លួន។
- | ពិវប្បធម៌ Global a និង b នឹងត្រូវការពិភាក្សាអម្ចាយខ្លួន 10 និង 20.
- | ឬយើងដាក់ទីនឹង `print_sum`, ពិវប្បធម៌ a និង b នឹងត្រូវការពិភាក្សាអម្ចាយខ្លួន 100 និង 200.



- ▶ នូវការសម្រេចការពិភាក្សាអម្ចាយខ្លួន ពិវប្បធម៌ global a និង b របស់យុទ្ធសាស្ត្រ នៃ Python script file.
- ▶ ពិវប្បធម៌ a និង b ឬយើងដាក់ទីនឹង `print_sum`, a, b នឹងត្រូវការពិភាក្សាអម្ចាយខ្លួន នៃ new object.

4. ពិវប្បធម៌ Global

4.3. global keyword

| នរណា ក្នុងការអំពីវប្បធម៌ global ត្រូវបានដើរជាធាសា global keyword.

```
1 def print_sum():
2     global a, b           # a and b use a and b declared outside the function
3     a = 100
4     b = 200               ← ពិវប្បធម៌ Global a និង b ត្រូវបានដើរជាធាសា
5     result = a + b
6     print('print_sum() inside : the sum of', a, 'and', b, 'is', result)
7
8     a = 10
9     b = 20
10    print_sum()
11    result = a + b
12    print('print_sum() outside : the sum of', a, 'and', b, 'is', result)
```

print_sum() inside : the sum of 100 and 200 is 300

print_sum() outside : the sum of 100 and 200 is 300

4. ពិវប្បធន Global

4.3. global keyword



Beware of SyntaxError

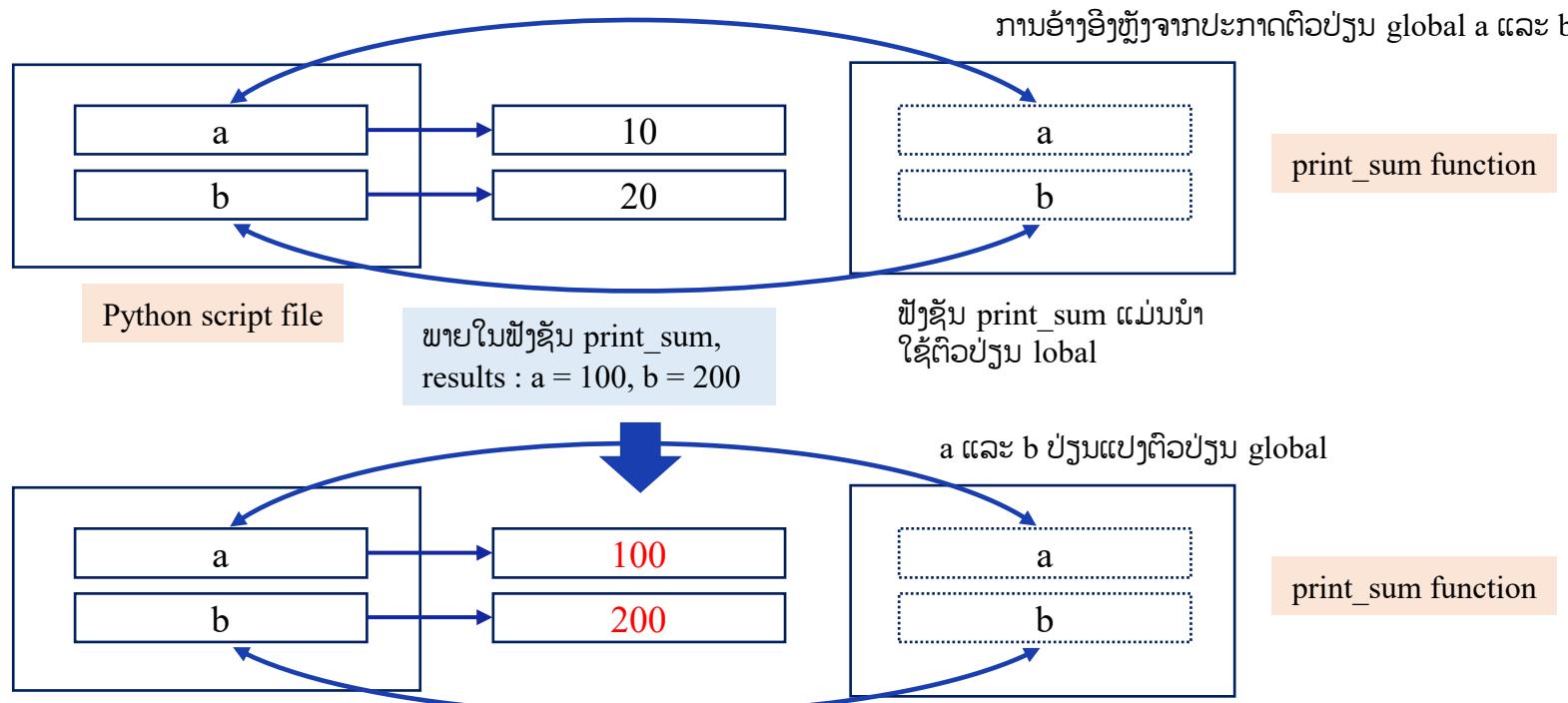
```
1 def print_sum():
2     global a = 0      # a and b use a and b declared outside the function
File "<ipython-input-55-15b3891832ee>", line 2
    global a = 0      # a and b use a and b declared outside the function
                                         ^
SyntaxError: invalid syntax
```

- ▶ Note: `global a = 0` จะເណើនមិ `syntax error`. យោងត្រូវបានរកចុចថា `global` ដែលមិនមែនចុចតាមរយៈការប្រើប្រាស់ការត្រួតពិនិត្យការអនុវត្តន៍.

4. ຕົວປ່ຽນ Global

4.4. ການອ້າງອີງໃນຕົວປ່ຽນ Global

| การประเมินโดยนำใช้ตัวปัจจุบัน global a และ b in a โดยมีการประเมินภาคตัวปัจจุบัน global ถัดไปนี้.



- ▶ ຖ້າປ່ຽນແປງຄ່າຂອງຕົວປ່ຽນທີ່ໄດ້ປະກາດເປັນ global a ຫຼື b, ຄ່າດັ່ງກ່າວຍັງຖືກປ່ຽນແປງຢູ່ນອກຟັງຊັ້ນ

4. ຕົວປັບປຸງ Global

4.5. ການນຳໃຊ້ຄ່າຄົງທີ່ໃນຕົວປັບປຸງ global



Global Variables vs. Global Constants

- ການນຳໃຊ້ global Variable ແມ່ນການປະຕິບັດການບໍ່ດີປານໃດໃນຫຼຸກພາສາໂປຣແກຣມ, ບໍ່ສະເພາະແຕ່ພາສາ Python. ເນື້ອໂຄດຄໍາສັ່ງມີຄວາມຍາວຈະຮັດໃຫ້ເກີດເປັນບັນຫາຫຼັກ.
- ໃນຫາງກົງກັນຂ້າມ, global constants ຈະບໍ່ເປັນຄື global Variable.
- global constant ແມ່ນຖືກປະກາດດ້ວຍ keyword global ດັ່ງສະແດງລຸ່ມນີ້. ນອກຈາກນີ້ຍັງສາມາດປະກາດນອກຝັງຊັ້ນ ໂດຍອ້າງອີງໂມດຸນ. ຄ່າຂອງ Global constant ຈະເປັນຕົວອັກສອນໃຫຍ່.
- ຄ່າຄົງທີ່ແມ່ນຕົວເລກທີ່ບໍ່ປັບປຸງແປງ.
- ເບິ່ງໂຄດຄໍາສັ່ງດ້ານລຸ່ມ, ຫຼັງຈາກນຳມືດຄ່າເປັນ 1024 ໃຫ້ຕົວປັບປຸງຊື່ GLOBAL_VALUE, ເຊິ່ງມັນໄດ້ປະກາດໃນຝັງຊັ້ນເປັນ global GLOBAL_VALUE ແລະ ເອັນໃຊ້ຄ່ານີ້ໃນຝັງຊັ້ນ foo function.

```
1 GLOBAL_VALUE = 1024 # Capitalizing global constants makes them easier to distinguish
2
3 def foo():
4     global GLOBAL_VALUE
5     a = GLOBAL_VALUE * 100
6     print(a)
7
8 foo()
```

102400

5. Function และ Method

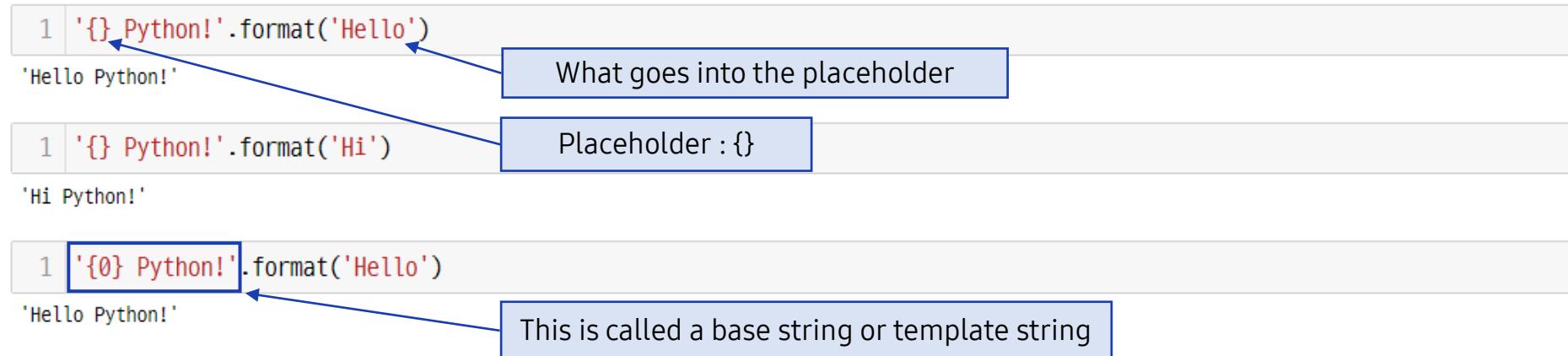
5.1. ការណែនាំនូវ function និង method

- | Functions និង methods ແມ່ນមີຄວາມត້າຍភິກັນ ພວກມັນທີ່ກຳນົດໃຫ້ຮັດວຽກສະເພາະໄດ້ឡື່ງ. ເຖິງຢ່າງໃດກຳຕາມ ແຕ່ພວກມັນກໍຍັງມີຄວາມពេញចិត្ត.
- | Function –ເປັນໂມດຸນຫຼື່ງໃນໂປຣແຣມ ຈະຮັບ parameter (ປະຕິເສດ) ເວລາທີ່ກາອັນໃຊ້ និង ຈະສິ່ງຄ່າກັບ ຖ້າຈໍາເປັນ.
- | Method - ເປັນຝັງຊັບທີ່ຜູກຕິດກັບ object ໃນ object-oriented programming.
 - ▶ ໂມດຸນ ສາມາດເອີ້ນ object ດ້ວຍຄ່າທີ່ពេញចិត្តກັນ ເອີ້ນວ່າ instances.
 - ▶ ໂມດຸນນີ້ຈະກວມເອົາໃນ class

5. Function และ Method

5.2. กานจัดรูบแบบ method

- ຽນຮູ້ກ່ຽວກັບການຈັດຮູບແບບສະແດງຜົນຂອງຂໍ້ຄວາມ. ການຈັດຮູບແບບສະແດງຜົນຂອງຂໍ້ຄວາມສາມາດເອີ້ນໃຊ້ string method.
- ເຄື່ອງໝາຍ {} ໃນ string ໃຊ້ເພື່ອ ສະແດງ argument ເຕັມວ່າ placeholder.
- ຖ້າ ໄສ argument ເຂົ້າໄປໃນ .format() method ດັ່ງສະແດງລຸ່ມນີ້ ຄ່າຂອງ argument ນີ້ ຈະສະແດງໃນ placeholder.



5. Function และ Method

5.2. กานจัดรูบแบบ method

| เริ่มมาดกุณภาพงานสระดูผิวถัวอย่างนี้ (index) พายในถืองมาย {} (ถัวอย่างนี้มันແມ່ນເລີ່ມຈາກ 0, 1, ..)

```
1 'I like {} and {}'.format('Python', 'Java')
```

```
'I like Python and Java'
```

```
1 'I like {0} and {1}'.format('Python', 'Java')
```

```
'I like Python and Java'
```

ໃສ່ index ແຕ່ໄປໃນຄໍອງໜາຍນີ້ ແຊ່ງເລີ່ມຈາກ 0, 1, ຂຶ້ນກັບຄຳຫຼັງຂຶ້ນ
ມູນທີ່ເຕືອງການ, Python ຫຼື Java ແມ່ນສາມາດເຮັດໄດ້ໂກັນ.

```
1 'I like {1} and {0}'.format('Python', 'Java')
```

```
'I like Java and Python'
```

'I like {} and {}'.format('Python', 'Java')



'I like {0} and {1}'.format('Python', 'Java')



'I like {1} and {0}'.format('Python', 'Java')



ກິດຂອງ placeholder ແລະ index

5. Function และ Method

5.2. กานจัดรูบแบบ method

- | เริ่มมาดใช้สะແດງຜົນດ້ວຍການນຳໃຊ້ {0}, {1}, {2}.
- | ການໃຊ້ placeholder ບໍ່ພຽງແຕ່ໃຊ້ໄດ້ກັບຂໍ້ຄວາມ ແຕ່ມັນຍັງສາມາດໃຊ້ກັບຕົວເລກ ຫຼື ຈຳນວນ ເຊັ່ນ 100 ຫຼື 200 ຫຼື ຈຳນວນຈິງອື່ນໆໄດ້.
- | ເຮົາສາມາດເປົ້າການສະແດງຜົນທີ່ແຕກຕ່າງກັນດັ່ງລຸ່ມນີ້.

```
1 '{0}, {0}, {0}! Python'.format('Hello')
```

```
'Hello, Hello, Hello! Python'
```

```
1 '{0}, {0}, {0}! Python'.format('Hello', 'Hi')
```

```
'Hello, Hello, Hello! Python'
```

```
1 '{0} {1}, {0} {1}, {0} {1}!'.format('Hello', 'Python')
```

```
'Hello Python, Hello Python, Hello Python!'
```

```
1 '{0} {1}, {0} {1}, {0} {1}!'.format(100, 200)
```

```
'100 200, 100 200, 100 200!'
```

‘{0}, {0}, {0}! Python’.format(‘Hello’, ‘hi’)



ເຮົາສາມາດເປົ້າການສະແດງຜົນດ້ວຍການພິມຕຳແໜ່ງຂຶ້ນໃນ placeholder.

5. Function และ Method

5.2. ការចែករូបប្រព័ន្ធមេធុណា

| ໃນការចែករូបប្រព័ន្ធផីន នឹងសារមាតប្រសិទ្ធភាពសារការងារលទ្ធផលរបស់ការងារ និង ពិវប្បធម៌ដោយការ តំង់សារបញ្ជី។

```
1 greet = 'Hello'  
2 '{} World!'.format(greet)
```

'Hello World!'

```
1 name = 'David'  
2 print('My Name is {}!'.format(name))
```

My Name is David!

Line 1, 2

- រាយការណ៍ឱ្យពិនិត្យ name = 'David' .
- ផ្ទាយរាយការណ៍ថាអ្នកបានបង្កើតឱ្យពិនិត្យ name តាមរាយការណ៍ format method, តាមទីនេះ name ត្រូវបានគិតឡើងជាប៊ូលូហូលូ (placeholder).

5. Function และ Method

5.2. ការចំណាំនូវបញ្ជីជាអត្ថបទ method

Ex តារាងខាងក្រោមនេះແມ័ນខ្លួនដូចខាងក្រោមនេះ ត្រូវបានរៀបចំឡើងដើម្បីបង្ហាញពីការប្រើប្រាស់នូវការចំណាំនូវបញ្ជីជាអត្ថបទ (method).

```
1 name = input('Enter your name : ')
2 age = input('Enter your age : ')
3 job = input('Enter your job : ')
4
5 print('Your name is {}, age is {}, job is {}.'.format(name, age, job))
```

```
Enter your name : David
Enter your age : 23
Enter your job : Programmer
Your name is David, age is 23, job is Programmer.
```

- ▶ ត្រូវបានគ្រប់ការចំណាំនូវបញ្ជីជាអត្ថបទ (method) ដោយប្រើប្រាស់ការចំណាំនូវបញ្ជីជាអត្ថបទ (method) ដែលបានរៀបចំឡើង។
- ▶ ត្រូវបានគ្រប់ការចំណាំនូវបញ្ជីជាអត្ថបទ (method) ដែលបានរៀបចំឡើង។
- ▶ ត្រូវបានគ្រប់ការចំណាំនូវបញ្ជីជាអត្ថបទ (method) ដែលបានរៀបចំឡើង។

6. ການຄຸນກ່ຽມ

6.1. ແນວດການຄຸນກ່ຽມ

| ຮຽນກ່ຽວກັບການຄຸນກ່ຽມ, ເຮືຈະມາສິນທະນາກ່ຽວກັບກ່ຽມ.

- ▶ ສັງເກດການຄຸນກ່ຽມໂດຍນໍາໃຊ້ function ລຸ່ມນີ້.
- ▶ ການຄຸນກ່ຽມ A^3 , ແນວດພື້ນຖານຫາງດ້ານຄະນິດສາດແມ່ນ $A \times A \times A$.
- ▶ $A \times A \times A$ ສະແດງການຄຸນອີງປະກອບຕ່ັງລຸ່ມນີ້.

$$\begin{aligned}A \times A \times A &= (A \times A) \times A = \{(1, 1)(1, 3), (3, 1), (3, 3)\} \times \{(1, 3)\} \\&= \{((1, 1), 1), ((1, 1), 3), ((1, 3), 1), ((1, 3), 3), ((3, 1), 1), ((3, 1), 3), ((3, 3), 1), \\&\quad ((3, 3), 3)\}\end{aligned}$$

- ▶ ການປະຕິບັດການຂ້າງເທິງສາມາດຄໍານວນດ້ວຍການຄຸນຊີ້າຂອງແຕ່ລະກ່ຽມ ດ້ວຍ ພຶກຊັນ product_set.

6. ការតូនក្នុង

6.1. ແນວគិតការតូនក្នុង

- | តិតដល់ជើងតូនខ្លួចក្នុងក្នុងឡើងបាន។
- | ពិបទនរណ៍ការតូនក្នុងទាំងអស់នេះមានមាត្រាដែលស្របតាមការរៀបចំ។

```
1 def product_set(set1, set2) :  
2     res = set()  
3     for i in set1:  
4         for j in set2:  
5             res = res | {(i, j)}    # Product set using double for loop  
6     return res  
7  
8 def exp(input_set, exponent) : # A function that performs powers on input_set  
9     res = input_set      # res initialization  
10    for _ in range(exponent-1) : # Repeated for (exponent-1) to become a power  
11        res = product_set(res, input_set)  
12    return res  
13  
14 A = {1, 3}  
15 A3 = exp(A, 3) # Perform 3 powers on set A  
16 print(A3)
```

```
((3, 1), 3), ((3, 3), 1), ((3, 1), 1), ((1, 3), 3), ((1, 1), 1), ((3, 3), 3), ((1, 1), 3), ((1, 3), 1)]
```

Line 15

- When an argument is passed in the exp function, exp calls product_set to perform a multiplication set. That is, the third power is performed.

6. ກາມຄຸນກຸ່ມ

6.2. ຕົວເລກທີ່ເປັນໄປໄດ້ຫັງໝົດໃນການຄຸນກຸ່ມ

| ນໍາໃຊ້ຝັງຊັ້ນ product_set ເພື່ອນັບຈຳນວນໃນ cases.

```
1 def product_set(set1, set2) :
2     res = set()
3     for i in set1:
4         for j in set2:
5             res = res | {(i, j)}      # Product set using double for loop
6     return res
7
8 cases = { 1, 2, 3, 4, 5, 6 }          # Integers from 1 to 6 on the die
9 cases_2times = product_set(cases, cases) # Get product_set for case
10 print(cases_2times)
```

$\{(3, 4), (4, 3), (3, 1), (5, 4), (4, 6), (5, 1), (2, 2), (1, 6), (2, 5), (1, 3), (6, 2), (6, 5), (4, 2), (4, 5), (3, 3), (5, 6), (3, 6), (5, 3), (2, 4), (1, 2), (2, 1), (1, 5), (6, 1), (6, 4), (3, 2), (4, 1), (3, 5), (5, 2), (4, 4), (5, 5), (1, 1), (1, 4), (2, 3), (2, 6), (6, 6), (6, 3)\}$

Line 8, 9

- ในตัวปั้น cases แม่นกำหนด set type จาก 1 ถึง 6.
 - คุณสร้าง cases. จำแนก ล้อมตัวเล็กสอยถ้าเข้ม กำหนด set type. ท้ายมีด cases แม่น return ปั๊ะสะจากงานเข้าวัน.

| Pair programming



Pair Programming Practice

| ແນວທາງ, ກິນໄກ ແລະ ແຜນສຸກເສີນ

ການຈັບຄຸ້ຂຽນໂປຣແກຣມ ເປັນການຈັບຄຸ້ຂອງນັກຮຽນເພື່ອຮັດວຽກມອບໝາຍ, ນັກຮຽນຄວນມີແຜນ ແລະ ສາມາດປ່ຽນແທນກັນໄດ້ ໃນ ກໍາລະນີມີຜູ້ໃຫ້ນີ້ບໍ່ສາມາດເຂົ້າຮ່ວມຮັດວຽກມອບໝາຍໄດ້ບໍ່ວ່າໃນກໍາລະນີໃດກຳຕາມ ເຊິ່ງບັນຫາເລື່ອນັ້ນຕ້ອງຮັດໃຫ້ຈະເຈັ້ງ ແລະ ກຳບໍ່ແມ່ນ ຄວາມຜິດຂອງນັກຮຽນທີ່ຈັບຄຸ້ບໍ່ດີ.

| ຈັບຄຸ້ທີ່ຄ້າຍຄືກັນ, ບໍ່ຈໍາເປັນເຫົ້າທຽມກັນ, ຄວາມສາມາດເປັນຄຸ້ຮ່ວມງານ

ການຈັບຄຸ້ຂຽນໂປຣແກຣມ ຈະໄດ້ຮັບຜົນໃກ່ກໍາຕໍ່ເນື້ອນັກຮຽນມີຄວາມສາມາດຄ້າຍຄືກັນ ຫາຍວ່າ ບໍ່ຈໍາເປັນຕ້ອງມີຄວາມສາມາດຄືກັນກຳໄດ້, ແຕ່ວ່າ ການຈັບຄຸ້ ນັກຮຽນທີ່ມີຄວາມສາມາດແຕກຕ່າງກັນຫຼາຍ ກໍຈະຮັດໃຫ້ບໍ່ສົມດຸນກັນ. ຄຸສອນຮູ້ດີວ່າ ການຈັບຄຸ້ກັນບໍ່ແມ່ນ ຍຸດທະສາດ “ແບ່ງເພື່ອເອົາຊະນະ” ແຕ່ເປັນຄວາມ ພະຍາຍາມຮັດວຽກຮ່ວມກັນຂອງນັກຮຽນໃຫ້ປະສິບຜົນສໍາເລັດ. ຄຄວນທີ່ກາເວັ້ນການຈັບຄຸ້ກັນລະຫວ່າງນັກຮຽນອ່ອນ ແລະ ນັກຮຽນເກົ່າງ.

| ກະຕຸ້ນນັກຮຽນໂດຍການໃຫ້ສິ່ງຈຸງໃຈພື້ນເສດ

ຂໍສະເໜີແຮງຈຸງໃຈທີ່ຮັດໃຫ້ນັກຮຽນຈັບຄຸ້, ໂດຍສະເພາະນັກຮຽນທີ່ມີຄວາມສາມາດສຸງ. ບາງຄຸສອນໄດ້ພື້ນວ່າ ການຈັບຄຸ້ຮັດວຽກມອບໝາຍ ແມ່ນມີ ປະໂຫຍດ ສໍາລັບໜຶ່ງ ຫຼື ສອງວຽກມອບເທົ່ານັ້ນ



Pair Programming Practice

| ចំណាំការបំពេញនូវក្រុមខែងម៉ក្រុម

ສິ່ງທ້າທາຍສໍາລັບຄຸນແມ່ນເພື່ອຊອກຫາວິທີທີ່ຈະປະເມີນຜົນການຮຽນຂອງນັກຮຽນ, ຄຸນຫຼືບໍ່ວ່າ ນັກຮຽນ ໄດ້ຕັ້ງໃຈຮຽນ ຫຼື ບໍ່ຕັ້ງໃຈຮຽນ. ຜູ້ສ່ວວຊານໄດ້ແນະນຳໃຫ້ທົບທວນການອອກແບບຫຼັກສູດການຮຽນ ແລະ ຮູບແບບການປະເມີນ ພ້ອມທັງປີກສາຫາລືຢ່າງຈິງຈັງກັບນັກຮຽນ ກ່ຽວກັບພິດຕິກຳທີ່ຈະບໍ່ຕັ້ງໃຈຮຽນ ນອກຈາກນີ້ຢັ້ງໄດ້ແນະນຳມອບວຽກມອບໝາຍໃຫ້ນັກຮຽນ ພ້ອມທັງອະທິບາຍໃຫ້ເຂົ້າເຈົ້າຢ່າງຈະເຈັ້ງ

| ສະພາບແວດລ້ອມຂອງການຮຽນຮູ້ຮ່ວມກັນ

ສະພາບແວດລ້ອມການຮຽນຮູ້ຮ່ວມກັນເກີດຂຶ້ນໄດ້ທຸກເວລາທີ່ຜູ້ສອນຮຽກຮ້ອງໃຫ້ນັກຮຽນຮັດວຽກຮ່ວມກັນໃນກິດຈະກຳການຮຽນຮູ້ ເຊິ່ງອາດຈະເປັນກິດຈະກຳທີ່ເປັນທາງການ ແລະ ບໍ່ເປັນທາງການ ແລະ ອາດຈະບໍ່ລວມເຖິງການປະເມີນສິນການຮຽນໂດຍກິງ. ເຊັ່ນຕົວຢ່າງ ໃຫ້ນັກຮຽນຈັບຄຸ້ງກັນເພື່ອຮັດວຽກມອບໝາຍ ໂດຍນັກຮຽນຈະຕ້ອງທີ່ບໍ່ທວນກ່ຽວກັບການສອນຂອງອາຈານທີ່ຜ່ານມາ ແລະ ລະດົມແນວຄົດພາຍໃນກຸ່ມ ພ້ອມທັງມືການແບ່ງວຽກໃຫ້ແຕ່ລະຄົມຮັບຜິດຊອບ ຈາກນັ້ນກຳໃຫ້ມີການແລກປ່ຽນຄວາມຄົດເຫັນເຊິ່ງກັນ ແລະ ກັນ ເພື່ອຮັດວຽກມອບໝາຍໃຫ້ສໍາເລັດຕາມເປົ້າໝາຍທີ່ວ່າງໄວ້.

Q1.

ຈົງສ້າງ ແລະ ເຮັນໃຊ້ຟັງຊັນ, ໂດຍໃຫ້ຜູ້ໃຊ້ບ່ອນສາມຈຳນວນ a , b ແລະ c . ແລ້ວໃຫ້ພິມ ຄ່າສະເລ່ຍ, ຄ່າໃຫຍ່ສຸດ, ຄ່ານ້ອຍສຸດຂອງສາມຈຳນວນ ດັ່ງກ່າວດັ່ງລຸ່ມນີ້. ໃນກໍລະນີນີ້ $\text{mean3}(a, b, c)$, $\text{max3}(a, b, c)$, $\text{min3}(a, b, c)$ ແມ່ນມີສາມຕົວເລກເປັນ parameters ແລະ ສິ່ງ (return) ຄ່າສະເລ່ຍ, ຄ່າໃຫຍ່ສຸດ ແລະ ຄ່ານ້ອຍສຸດຂອງສາມຈຳນວນ.

Output Example

```
Enter three numbers : 9 2 6
The average value of 9, 2, 6 is 5.666666666666667
The maximum value of 9, 2, 6 is 9
The minimum value of 9, 2, 6 is 2
```