

Unit 12.

ປະເພດຂໍ້ມູນແບບ Sequence

Learning objectives

- ✓ ສາມາດຄື້ນຫາ ແລະ ນຳໃຊ້ຄ່າສະເພາະຂອງ ປະເພດຂໍ້ມູນແບບ sequence
- ✓ ສາມາດລວມສອງວັດຖຸໂດຍຜ່ານການເຊື່ອມໄຍງ່ sequence objects
- ✓ ຂຽນຮູ້ປະເພດຂໍ້ມູນ iteration ຈາກ sequence objects ແລະ ສາມາດພິມໂດຍນຳໃຊ້ iteration statements
- ✓ ສາມາດຄິດໄລ່ເລກທີເປັນອີງປະກອບຂອງ sequence object
- ✓ ສາມາດກຳນົດຂອບເຂດສະເພາະ sequence object index
- ✓ ຂຽນຮູ້ຟັງຊັນ len ໃນ sequence objects

Learning overview

- ✓ ຮຽນຮູ້ຝຶງຊັ້ນການຄົ້ນຫາຂໍ້ຄວາມໃນ tuple ແລະ list
- ✓ ຮຽນຮູ້ກ່ຽວກັບການເຊື່ອມໄຍງ sequence objects ແລະ ການຄໍານວນຄ່າຂອງອີງປະກອບຕ່າງໆພາຍໃນ sequence object
- ✓ ເຂົ້າໃຈປະເພດ ແລະ ລັກສະນະພິສະດຂອງ iteration data types ແລະ ນຳໃຊ້ມັນຢ່າງໝາຍະສົມ
- ✓ ຮຽນວິທີການນຳໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ sequence objects
- ✓ ຮຽນວິທີການຄໍານວນຄວາມຍາວຂອງ sequence objects ແລະ ການນຳໃຊ້ຝຶງຊັ້ນ len

Concepts You Will Need to Know From Previous Units

- ✓ ຕ້ອງເຂົ້າໃຈຢ່າງຖືກຕ້ອງກ່ຽວກັບແນວຄິດຂອງ list, range, tuple, ແລະ string ຊຶ່ງປັນ sequence objects
- ✓ ສາມາດນຳໃຊ້ ຕົວດຳເນີນການຂອງ sequence objects ປະກອບມີ in, not in, ແລະ ອື່ນໆ.
- ✓ ສາມາດພື່ມ ແລະ ລຶບ ອີງປະກອບໃນ list. ສາມາດນຳໃຊ້ slicing ແລະ indexing

Keywords

**Sequence
object**

Iteration

Searching, linking

**Finding a specific
element**

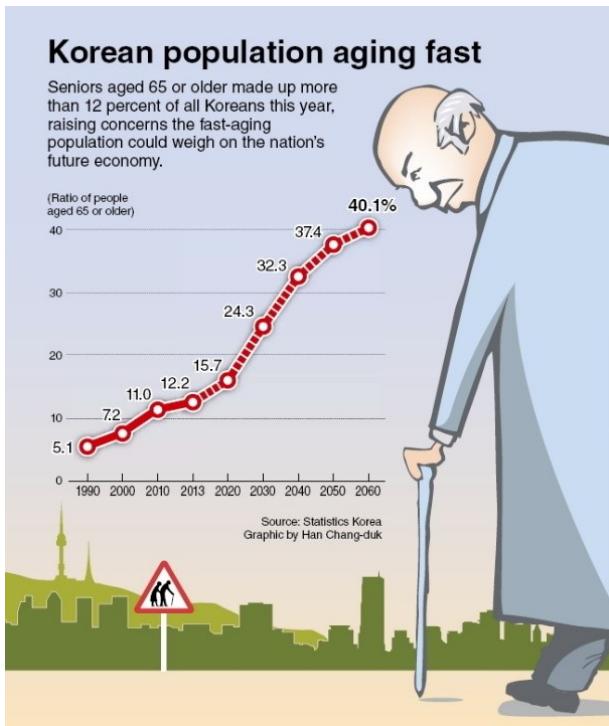
Index

len, range

Mission

1. Real world problem

1.1. ប័ណ្ណការងារ



- ▶ ຄົນສູງອາຍຸ ເປັນບັນຫາສັງຄົມທີ່ຮ້າຍແຮງໃນບາງປະເທດ ໂດຍສະເພາະ ປະເທດເກົ້າຫຼື ແລະ ປະເທດຢືນ.
 - ▶ ເປົ້າເຊັນຂອງຜູ້ສູງອາຍຸທຽບໃສ່ປະຊາກອນທັງໝົດ ແມ່ນນັບມື້ນບສູງຂຶ້ນ.
 - ▶ ບັນຫາໃຫຍ່ຂອງປະເທດທີ່ມີຜູ້ສູງອາຍຸຫາຍມີດັ່ງນີ້:
 - ຂາດແຮງງານພາຍໃນປະເທດ ແລະ ອາຫານສໍາລັບສູງອາຍຸ ບໍ່ພຽງພໍ
 - ຄ່າໃຊ້ຈ່າຍເພີ່ມຂຶ້ນຢ້ອນຕ້ອງໄດ້ດຸແລຄົນສູງອາຍຸ
 - ການຫາວຽກເຮັດງານທຳມືການແຂ່ງຂັນກັນສູງ
 - ສູງອາຍຸຖືກປ່ອຍໃຫ້ຢູ່ຕາມລຳພັງ
 - ▶ ລອງສໍາຫຼວດ ແລະ ຄິດໄລ່ອັດຕາສ່ວນຜູ້ສູງອາຍຸໃນເມືອງຂອງເຈົ້າ.

<http://www.inhapress.com/news/articlePrint.html?idxno=6982>

1. Real world problem

1.2. ຄິດໄລ່ເປີເຊັນຂອງຜູ້ສູງອາຍຸ



http://www.inhapress.com/news/articlePrint.html?i_dxno=6982

- ຂໍ້ມູນປະຊາກອນໃນເມືອງໜຶ່ງ ເກັບເປັນຂໍ້ມູນປະເພດ tuple, ໃນນີ້ ອີງປະກອບທີ 1 ແມ່ນຈຳນວນປະຊາກອນທີ່ມີອາຍຸຢູ່ລະຫວ່າງ 0 ຫາ 9 ປີ, ແລະ ອີງປະກອບທີ 2 ແມ່ນປະຊາກອນທີ່ມີອາຍຸຢູ່ລະຫວ່າງ 10~19 ປີ ເຊິ່ງເປັນໄວລຸ່ມ. ອີງປະກອບທີ 3 ແມ່ນປະຊາກອນທີ່ມີອາຍຸຢູ່ລະຫວ່າງ 20~29 ປີ, ແລະ ອີງປະກອບທີ i+1 ແມ່ນປະຊາກອນທີ່ມີອາຍຸຢູ່ລະຫວ່າງ 10i~10i+9 ປີ. ອີງປະກອບ 11 ແມ່ນອີງປະກອບສຸດທ້າຍ ເຊິ່ງແມ່ນປະຊາກອນທີ່ມີອາຍຸຫຼາຍກວ່າ 100 ປີຂຶ້ນໄປ.
- ສົມມຸດ ມີ 2 tuple ທີ່ເກັບຂໍ້ມູນປະຊາກອນສອງເມືອງ ຄື: ເມືອງ A ແລະ ເມືອງ B ດັ່ງສະແດງລຸ່ມນີ້.

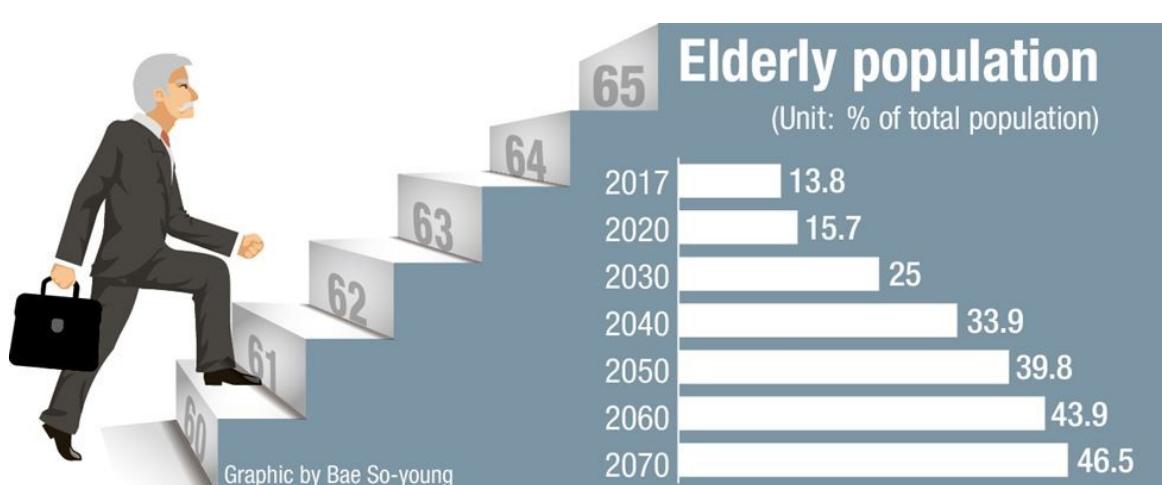
```
population_a = (100, 150, 230, 120, 180, 100, 140, 95, 81, 21, 4)
population_b = (300, 420, 530, 420, 400, 300, 40, 5, 1, 1, 1)
```

- ປຽບທຽບລະດັບຜູ້ສູງອາຍຸໃນແຕ່ລະເມືອງ. ສົມມຸດວ່າ ລະດັບຄວາມສູງອາຍຸແມ່ນເປີເຊັນຂອງຜູ້ສູງອາຍຸທີ່ມີອາຍຸ 70 ປີ ຂຶ້ນໄປໃນຈຳນວນປະຊາກອນທັງໝົດ. ຈຶ່ງຂຽນໂປຣແກຣມເພື່ອໃຫ້ໄດ້ຜົນຮັບດັ່ງຕໍ່ໄປນີ້. (ນຳໃຊ້ slicing ເພື່ອເອົາສະເພາະສ່ວນສໍາຄັນຈາກຂໍ້ມູນ)

1. Real world problem

1.3. ខ័ណ្ឌមុនយើងពីរការកំណត់ស្តីសុំរបៀប

| សំលែកលាយលະទ្វាតកំណត់ស្តីសុំរបៀប, សាមាតខ៉ី ໄប់ប៉ែង ដោយ <https://m.koreatimes.co.kr/pages/article.asp?newsIdx=270287>



<https://m.koreatimes.co.kr/pages/article.asp?newsIdx=270287>

2. Mission

2.1. ວິທີຄິດໄລ່ອັດຕາສ່ວນຜູ້ສຸງອາຍຸທີ່ເຮັດວຽກ

```
1 population_a = (100, 150, 230, 120, 180, 100, 140, 95, 81, 21, 4)
2 population_b = (300, 420, 530, 420, 400, 300, 40, 5, 1, 1, 1)
3 oldA = sum(population_a[7:])
4 oldB = sum(population_b[7:])
5 sumA, sumB = sum(population_a), sum(population_b)
6
7 oldRateA, oldRateB = oldA/sumA, oldB/sumB
8
9 print('The degrees of aging in town A and B are {:.3f} and {:.3f} respectively.'.format(oldRateA, oldRateB))
```

ອັດຕາສ່ວນຂອງຜູ້ສຸງອາຍຸໃນເມືອງ A ແລະ B ແມ່ນ 0.165 ແລະ 0.003 ຕາມລຳດັບ.

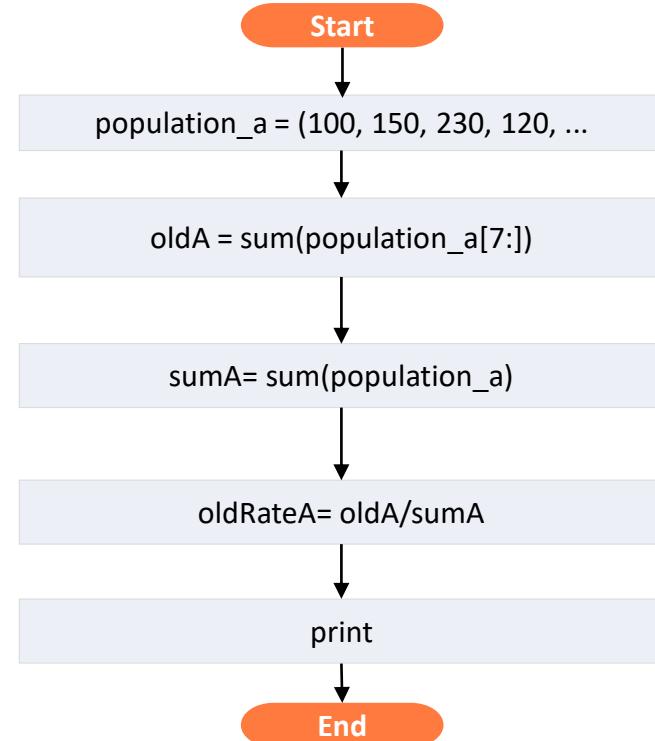
2. Mission

2.2. Programming plan

Pseudocode

- [1] Start
- [2] Enter the population data of the town.
- [3] Use slicing to only add the elderly who are 70 years old or older
- [4] Add the total population of the town
- [5] Divide the elderly population with the total population of the town.
- [6] Print the aging percentage.
- [7] End

Flowchart



2. Mission

2.3. final code ຂອງ ການຄິດໄລ່ເປີເຊັນຜູ້ສຸງອາຍຸ

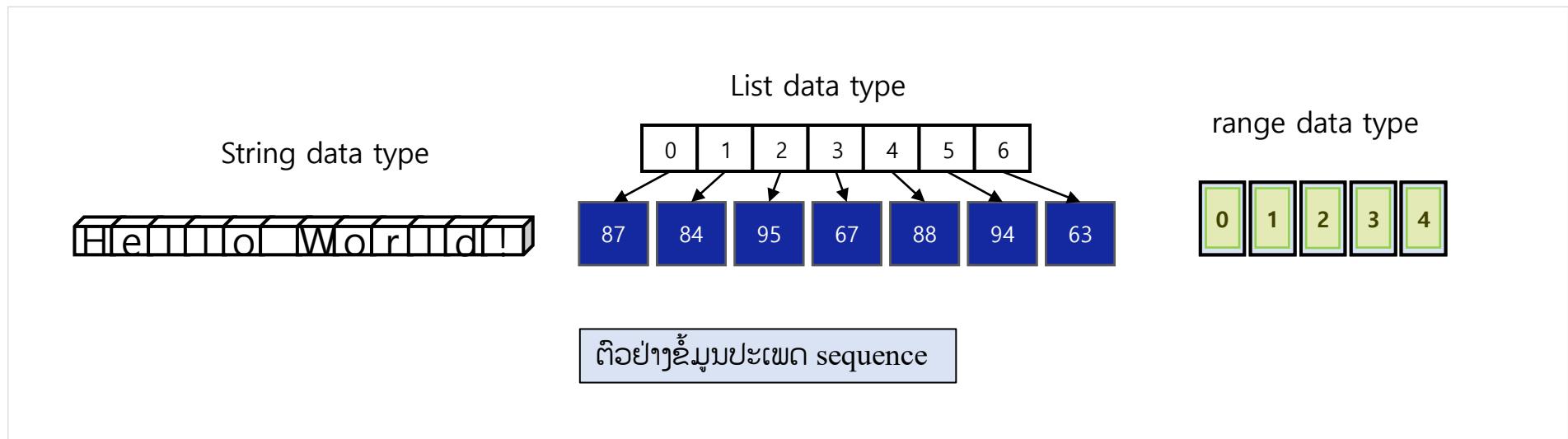
```
1 population_a = (100, 150, 230, 120, 180, 100, 140, 95, 81, 21, 4)
2 population_b = (300, 420, 530, 420, 400, 300, 40, 5, 1, 1, 1)
3 oldA = sum(population_a[7:])
4 oldB = sum(population_b[7:])
5 sumA, sumB = sum(population_a), sum(population_b)
6
7 oldRateA, oldRateB = oldA/sumA, oldB/sumB
8
9 print('The degrees of aging in town A and B are {:.3f} and {:.3f} respectively. '.format(oldRateA, oldRateB))
```

| Key concept

1. ការកើតហាក់សម្រេច

1.1. បរយោទីមុនបែបលាត់ប

- | list, tuple, range และ string ស្ថិតិវិញ្ញាណភាពកំណើ គ្នាតារាងទីក្រុងក្នុងខ្លួនដែលត្រូវបានបង្ហាញ។
- | ក្នុង Python, ខ្លួនបែប sequence ផ្ទុកជាប្រព័ន្ធឌូចជា list, tuple, range, string និងទៅជាតុលាការនៃ sequence types.
- | Objects តិចតាមតាមរយៈខ្លួនបែប ដើម្បីជាតុលាការនៃ sequence objects និង ការប្រើប្រាស់នៃ element ដើម្បីបង្ហាញខ្លួន។



1. ການຄົ້ນຫາຄ່າສະເພາະ

1.2. ຕົວດຳເນີນການ *in*, *not in*

- | “in” และ “not in” ແມ່ນຕົວດຳເນີນການທີ່ສິ່ງຄ່າ True ຫຼື False.
 - | ຕົວດຳເນີນການໃຊ້ເພື່ອກວດສອບອີງປະກອບຕ່າງໆຢູ່ໃນໂຄງສ້າງຂໍ້ມູນ, ເຊິ່ງໂຄງສ້າງຂໍ້ມູນປະກອບດ້ວຍ string, list, tuple.
 - | ຕົວດຳເນີນການ in ຈະສິ່ງຄ່າ True ຖ້າມີສະມາຊີກຢູ່ໃນໂຄງສ້າງຂໍ້ມູນ, ບໍ່ດັ່ງນັ້ນຈະສິ່ງຄ່າ False.
 - | ຕົວດຳເນີນການ not in ຈະສິ່ງຄ່າ False ບ້າມີສະມາຊີກຢູ່ໃນໂຄງສ້າງຂໍ້ມູນ, ບໍ່ດັ່ງນັ້ນຈະສິ່ງຄ່າ True.
 - | ຈາກຕົວຢ່າງລຸ່ມນີ້, ຍ້ອນວ່າ list1 ມີຄ່າ 10 ສະໜັບ 10 in list1 ຈະສິ່ງຄ່າ True ອອກມາ, ໃນຂະນະທີ່ 10 not in list1 ຈະສິ່ງຄ່າ False ອອກມາ.

```
1 list1 = [10, 20, 30, 40]
2 10 in list1
```

True

```
1 list1 = [10, 20, 30, 40]
2 10 not in list1
```

False

1. ການຄົ້ນຫາຄ່າສະເພາະ

1.3. ນໍາໃຊ້ຕົວດຳເນີນການ “in”

| เวลาถ้าต้องการเช็คค่าใน tuple, ให้ใช้ตัวคำนึง “in” .

```
1 tup = (1, 2, 3, 4)
2 3 in tup
```

True

| ນໍາໃຊ້ຕົວດໍາເນີນການ “in” ດ້ວຍຝ່າຊັ້ນ range.

1 | 11 in range(10)

False

| ນໍາໃຊ້ຕົວດໍາເນີນການ “in” ເພື່ອຄົ້ນຫາຄ່າສະເພາະໃນ string. ໃນ Code ຄຳສັ່ງລົມນີ້, ເນື້ອຈາກ ‘a’ ຢ່າໃນ ‘abcd.’ ຈຶ່ງສິ່ງຄ່າ True ອອກມາ

```
1 'a' in 'abcd'
```

True

2. ການເຊື່ອມໄຍງ Sequence Object

2.1. ການເຊື່ອມໄຍງ sequence object

- | ສໍາລັບປະເພດຂໍ້ມູນແບບ sequence , ສາມາດນຳໃຊ້ຕົວດຳເນີນການບວກ (+).
- | ແຕ່ ພັງຊັນ range ບໍ່ສາມາດນຳໃຊ້ ຕົວດຳເນີນການບວກໄດ້.
- | ການເຊື່ອມໄຍງພາຍໃນ list

```
1 list1 = [11, 22, 33, 44]
2 list2 = [55, 66]
3 print(list1)
4 print(list1 + list2)
```

```
[11, 22, 33, 44]
[11, 22, 33, 44, 55, 66]
```

2. ການເຊື່ອມໄຍງ Sequence Object

2.2. ການເຊື່ອມໄຍງ sequence object

| ລຸ່ມນີ້ແມ່ນການເຊື່ອມ tuple. ໂດຍໃຊ້ຕົວດຳເນີນການບອກ (+) .

```
1 tup1 = (1, 2, 3)
2 tup2 = (4, 5, 6)
3 print(tup1 + tup2)
```

(1, 2, 3, 4, 5, 6)

| ລຸ່ມນີ້ແມ່ນການເຊື່ອມ string. ໂດຍໃຊ້ຕົວດຳເນີນການບອກ (+) .

```
1 str1 = 'hello '
2 str2 = 'world'
3 print(str1 + str2)
```

hello world

2. ການເຊື່ອມໄຍງ Sequence Object

2.3. ຂໍຜິດພາດຈາກການເຊື່ອມໄຍງ sequence object



TypeError

```
1 range(10) + range(10, 20)
```

```
TypeError                                 Traceback (most recent call last)
<ipython-input-14-b16ec941300e> in <module>
      1 range(10) + range(10, 20)

TypeError: unsupported operand type(s) for +: 'range' and 'range'
```

- ▶ ໃນປະພດຂໍ້ມູນ sequence , ຕົວດໍາເນີນການ + ບໍ່ສາມາດໃຊ້ໃນຝັງຂັນ range ໄດ້.

| ແຕ່ວ່າ ຖ້າຕ້ອງການເຊື່ອມໄຍງ ຝັງຂັນ range ຕ້ອງໄດ້ແປງ range ໃຫ້ເປັນ list ຫຼື tuple ກ່ອນ ຈຶ່ງສາມາດເຊື່ອມໄດ້

```
1 list(range(10)) + list(range(10,20))
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

```
1 tuple(range(10)) + tuple(range(10,20))
```

```
(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19)
```

3. ការង់តុលាកម្ម Sequence Object

3.1. ការង់តុលាកម្ម sequence object

| ពិវត្តន៍ការង់តុលាកម្ម sequence object របស់ខ្លួន (*)

- ▶ ប្រព័ន្ធសម្រាប់លំដាបសាមាតាំ ដើម្បីពិវត្តន៍ការង់តុលាកម្ម (*) ។ តើ តុលាកម្មរបស់ខ្លួន ត្រូវបានរាយបញ្ជាផលនៅក្នុងប្រព័ន្ធបណ្តុះបណ្តាល។
- ▶ ការង់តុលាកម្ម objects នៃប្រព័ន្ធសម្រាប់លំដាប ត្រូវបានរាយបញ្ជាផលនៅក្នុងប្រព័ន្ធបណ្តុះបណ្តាល ដើម្បីបង្ហាញការង់តុលាកម្ម របស់ខ្លួន ។
- ▶ ផ្ទាល់រាយបញ្ជាផលនៃការង់តុលាកម្ម: [sequence data type] * integer

| List iteration

```
1 list1 = [11, 22, 33, 44] * 2
2 print(list1)
```

```
[11, 22, 33, 44, 11, 22, 33, 44]
```

3. ການເຮັດຊໍາຂອງ Sequence Object

3.1. ການເຮັດຊໍາຂອງ sequence object

| Tuple iteration

```
1 tup1 = (1, 2, 3)
2 print(tup1 * 2)
```

(1, 2, 3, 1, 2, 3)

| String iteration

```
1 str2 = 'hello'
2 print(str2 * 3)
```

hellohellohello

3. ການເຮັດຊໍາຂອງ Sequence Object

3.2. ຂໍ້ຄວນລະວັງໃນການເຮັດຊໍາໃນ sequence object



ຂໍ້ຜິດພາດຈະເກີດຂຶ້ນ ຖ້ານໍາໃຊ້ຕົວດໍາເນີນການ ອຸນ (*) ໃນຝັງຊັ້ນ range ດັ່ງນີ້

```
1 range(10) * 3
```

```
TypeError                                 Traceback (most recent call last)
<ipython-input-26-90a7d224213c> in <module>
----> 1 range(10) * 3
```

TypeError: unsupported operand type(s) for *: 'range' and 'int'

- ສໍາລັບຂໍ້ມູນປະເພດ range , objects ບໍ່ສາມາດເຊື້ອມໄຍງໂດຍຕົວດໍາເນີນການ + .
- ເຊັ່ນດຽວກັນ ຂໍ້ມູນປະເພດ range ກໍບໍ່ສາມາດສ້າງການຊໍາກັນ ໂດຍຕົວດໍາເນີນການ *.

3. ການເຮັດຊໍາຂອງ Sequence Object

3.3. ການເຮັດໃຫ້ປະເພດຂໍ້ມູນ range ສາມາດໃຊ້ຕົວດຳເນີນການຄຸນ (*)

- ໃນປະເພດຂໍ້ມູນ range ບໍ່ສາມາດເຮັດໃຫ້ຊໍ້ກັນໂດຍໃຊ້ ຕົວດຳເນີນການ * ໄດ້.
- ດັ່ງນັ້ນ, ຈຶ່ງຕ້ອງປ່ຽນປະເພດຂໍ້ມູນດັ່ງກ່າວ ໃຫ້ເປັນ list ຫຼື tuple ຈຶ່ງສາມາດນຳໃຊ້ ຕົວດຳເນີນການຄຸນໄດ້.

```
1 ran = list(range(5)) * 3  
2 print(ran)
```

```
[0, 1, 2, 3, 4, 0, 1, 2, 3, 4, 0, 1, 2, 3, 4]
```

```
1 ran = tuple(range(5)) * 3  
2 print(ran)
```

```
(0, 1, 2, 3, 4, 0, 1, 2, 3, 4, 0, 1, 2, 3, 4)
```

3. ការគេលខ្លឹម Sequence Object

3.4. តាមពីរនៃការប្រើប្រាស់សម្រាប់ចិត្តរបាយការណ៍ (*)



តាមឯកសារមុនប្រព័ន្ធចារណ៍ list ចាប់ពីលទ្ធផល list ដើម្បីប្រើប្រាស់សម្រាប់ចិត្តរបាយការណ៍ * មានភារណ៍ syntaxic error

```
1 list1 = [11, 22, 33, 44]
2 list2 = [55, 66]
3 print(list1 * list2)
```

```
TypeError                                                 Traceback (most recent call last)
<ipython-input-21-01d94caf6e19> in <module>
      1 list1 = [11, 22, 33, 44]
      2 list2 = [55, 66]
----> 3 print(list1 * list2)
```

TypeError: can't multiply sequence by non-int of type 'list'

- ▶ ប៉ាសមាណាគោះ [sequence data type] * [sequence data type] នៃ list, tuple, និង string ໄດំ.
- ▶ និងប្រព័ន្ធមូលដ្ឋាន តាមចំណាំចិត្តរបាយការណ៍ [sequence data type] * [integer].

4. ການນັບອົງປະກອບສະເພາະໃນ Sequence Object

4.1. count method

- ▶ method ທີ່ຊື່ count ໃຊ້ເພື່ອນັບອົງປະກອບສະເພາະ ຫຼື ຄວາມຖືຂອງອົງປະກອບໃນຂໍ້ມູນປະເພດລຳດັບ.
 - ▶ count method ຈະສົ່ງຈຳນວນຄວາມຖືຂອງອົງປະກອບທີ່ມີໃນ sequence object.
 - ▶ ໃນໂຄດຄຳສັງລຸ່ມນີ້ແມ່ນການນຳໃຊ້ຝັງຊັນ count ເພື່ອນັບຄວາມຖືຂອງອົງປະກອບທີ່ມີຄ່າເທົ່າ 11 ໃນ list1 ເນື່ອງຈາກຄ່າດັ່ງກ່າວມີຫັງໜິດ 3 ຕົວສະນັ້ນ ມັນຈະສົ່ງຄ່າ 3 ອອກມາ.

```
1 list1 = [11, 11, 11, 22, 33, 44]  
2 print(list1.count(11))          # Number of specific elements
```

3

4. ການນັບອີງປະກອບສະເພາະໃນ Sequence Object

4.1. count method

ສະແດງການນຳໃຊ້ count method ໃນ tuple

- ▶ ມີຈຳນວນ 11 ຢູ່ 3 ຄ່າ ໃນ tup1.

```
1 | tup1 = (11, 11, 11, 22, 33, 44)           # Number of specific elements
2 | print(tup1.count(11))
```

3

| ສະແດງການນຳໃຊ້ count method ໃນ string

- ▶ ມີຕົວອັດສອນ ‘L’ ຢູ່ 3 ຕົວ ໃນ str1.

```
1 | str1 = 'hello world'                      # Number of specific elements
2 | print(str1.count('l'))
```

3

4. ການນັບອີງປະກອບສະເພາະໃນ Sequence Object

4.2. ຕົວຢ່າງ Code ຄໍາສັ່ງ count method

| ສິນທຽບ len ແລະ count ເພື່ອພິມເຕີນໄດ້ຮັບ ດັ່ງສະແດງລຸ່ມນີ້

```
1 ran = range(0, 5, 1)
2 print(len(ran))
3 print(ran.count(2))
```

5

1

Line 2, 3

- ran ມີ 5 ຄ່າຄື: 0, 1, 2, 3, 4. ຈໍານວນອີງປະກອບຖືກພິມດ້ວຍການນຳໃຊ້ຝັງຊັນ len ແລະ ເຕີນໄດ້ຮັບແມ່ນ 5.
- ໃນຫາງກົງກັນຂ້າມ, ເມື່ອໃຊ້ count method ເພື່ອສະແດງຄ່າ 2 ໃນ ran, ມັນກໍຈະພິມຕົວເລກ 1 ອອກມາ, ເນື່ອງຈາກອີງປະກອບທີ່ມີຄ່າເທົ່າ 2 ມີພຽງ 1 ຄ່າເທົ່ານັ້ນ.

5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.1. ຕຳແໜ່ງຂໍ້ມູນ ແລະ sequence object

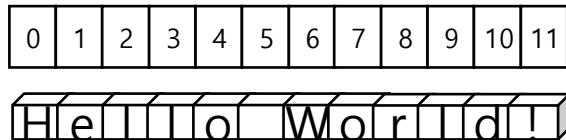
| ຕຳແໜ່ງຂໍ້ມູນແມ່ນຖືກນໍາໃຊ້ໃນ sequences objects .

| ຕຳແໜ່ງຂໍ້ມູນ:

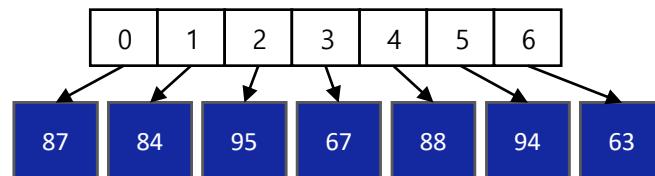
- ▶ ຕຳແໜ່ງຂໍ້ມູນຈະອ້າງອີງຄ່າຂອງອີງປະກອບໃນ list ຫຼື sequence.
- ▶ ທ້າວ່າ list ມີ n ອີງປະກອບ ສະນັ້ນ ຕຳແໜ່ງຂໍ້ມູນໃນ list ນີ້ ຈະມີຕຳແໜ່ງຈາກ 0 ເຖິງ n-1.

| ຕຳແໜ່ງຂໍ້ມູນໃນ sequence object ຈະເລີ່ມຕົ້ນຈາກ 0.

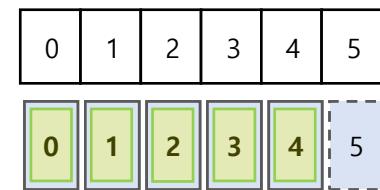
String



List



range(5)



ໃນຂໍ້ມູນປະເພດ sequence, ຕຳແໜ່ງຂໍ້ມູນຈະອ້າງອີງອີງປະອີງປະກອບ.

5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.2. ຕຳແໜ່ງຂໍ້ມູນໃນ sequence object

| ຕຳແໜ່ງຂອງຂໍ້ມູນໃນ list ຈະອ້າງອີງຄ່າຂອງມັນ

```
1 list1 = [11, 11, 11, 22, 33, 44]
```

```
1 list1[0]
```

11

```
1 list1[2]
```

11

| ຕຳແໜ່ງຂອງຂໍ້ມູນໃນ string ຈະອ້າງອີງຄ່າຂອງມັນ

```
1 str1 = 'hello world'
```

```
1 str1[2]
```

'l'

```
1 str1[7]
```

'o'

5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.2. ຕຳແໜ່ງຂໍ້ມູນໃນ sequence object

| ຄ່າຕ່າງໆໃນອີງປະກອບແມ່ນອ້າງອີງຈາກຕຳແໜ່ງຂໍ້ມູນໃນ tuple

```
1 tup1 = (1,1,1,2,3,3,4)
```

```
1 tup1[3]
```

2

| range(0, 5, 1) ມີຄວາມໝາຍວ່າ ອົງປະກອບໃນ range ມີຄ່າຕັ້ງແຕ່ 0 ເຖິງ 4 ເຊິ່ງຄ່າດັ່ງກ່າວຈະເພີ່ມຂຶ້ນເທື່ອລະ 1, ແຕ່ຖ້າ range(0, 5, 2) ສະແດງວ່າຄ່າໃນ range ຈະເພີ່ມຂຶ້ນເທື່ອລະ 2. ສາມາດອ້າງອີງຄ່າຂອງອົງປະກອບໂດຍໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ range ໄດ້ ດັ່ງສະແດງລຸ່ມນີ້

```
1 ran = range(0,5,1)
```

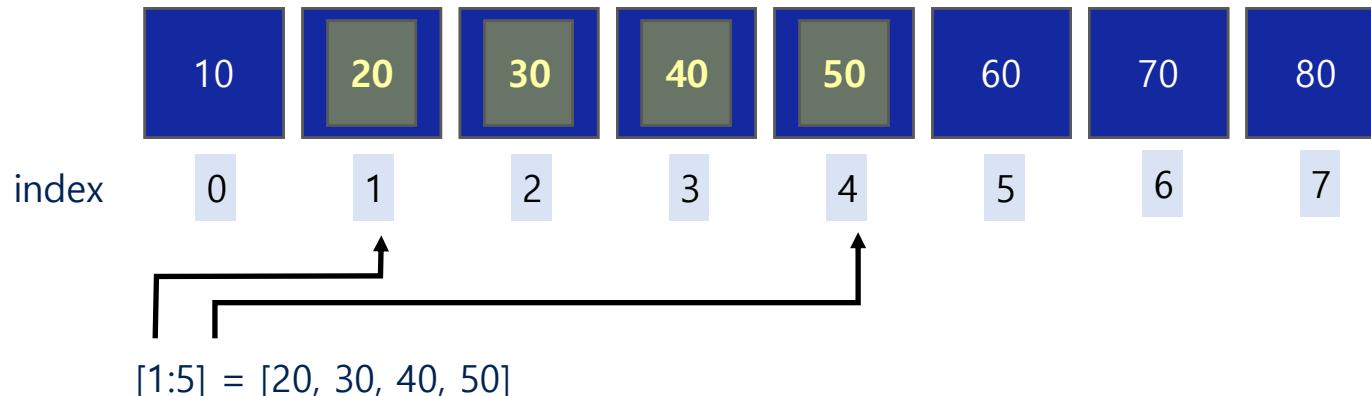
```
1 ran[4]
```

4

5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.3. ການຕັດໃນ sequence object

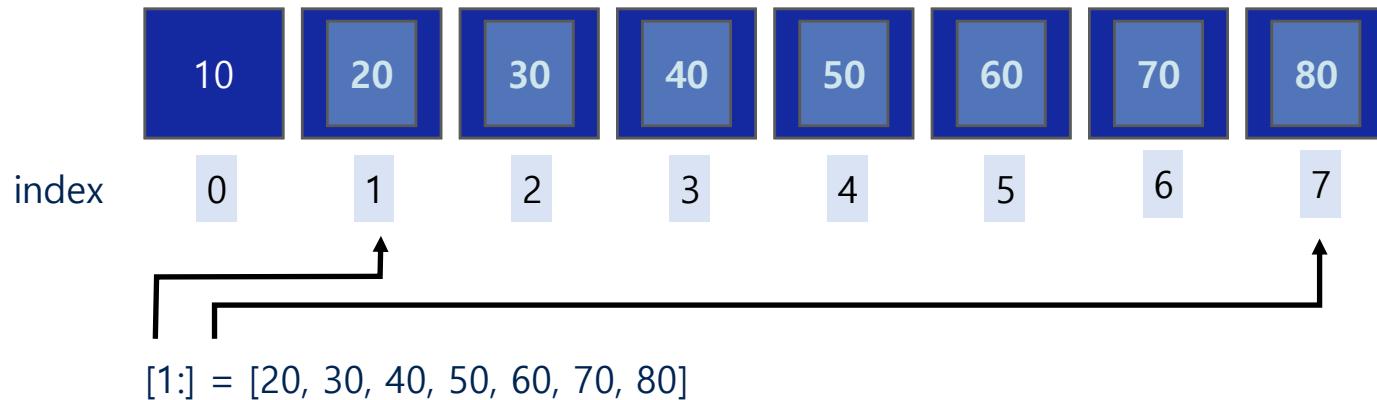
- ໄດຍທີ່ໄປແລ້ວ, ການຕັດໃນປະເພດຂໍ້ມູນແບບ sequence ຈະສາມາດຕັດເອົາສະເພາະສ່ວນທີ່ຕ້ອງການ ເຊິ່ງປະກອບດ້ວຍຕຳແໜ່ງເລີ່ມຕົ້ນ ແລະ ສຸດທ້າຍ.
- ທັງໝົດຂອງການຕັດຂໍ້ມູນແບບ sequence ຈະນໍາໃຊ້ ວິທີຕັດດັ່ງໃນລຸ່ມນີ້.
- ຖ້າກຳນົດ `[1:5]` ຈະເປັນການຕັດເອົາອີງປະກອບ ຈາກຕຳແໜ່ງຂໍ້ມູນທີ 1 ຫາ ຕຳແໜ່ງທີ 5-1 = 4.
- ແລະ ຈະໄດ້ຜົນຮັບຄື `[20, 30, 40, 50]`



5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.3. ການຕັດໃນ sequence object

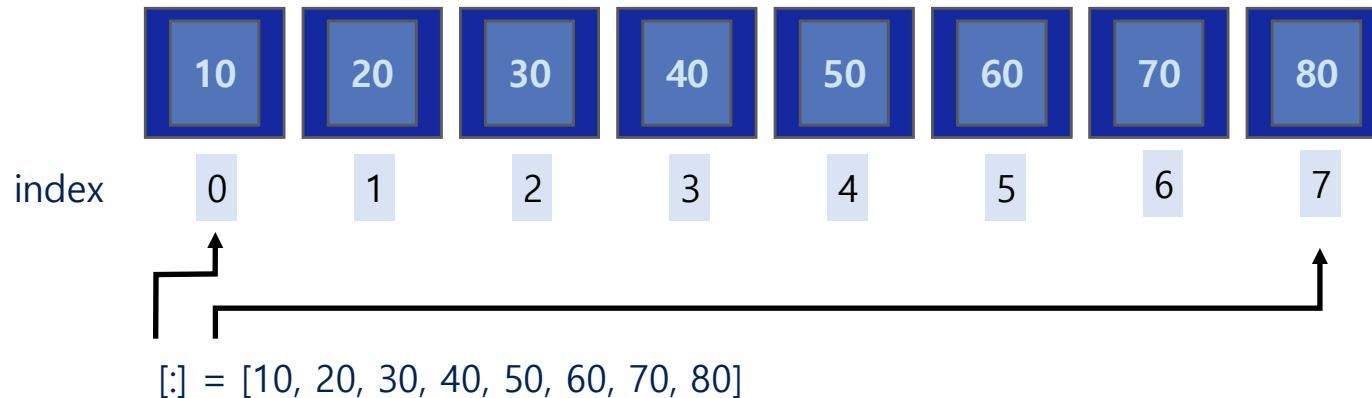
- | ເມືອບໍ່ມີການກຳນົດຕຳແໜ່ງສຸດທ້າຍຂອງອີງປະກອບ, ຈະເຮັດໃຫ້ຜົນໄດ້ຮັບມີຕຳແໜ່ງສຸດທ້າຍລວມຢູ່ນໍາ.
- | ເຊັ່ນວ່າ ຖ້າກຳນົດ [1:] ຫາຍວ່າ ແມ່ນການຕັດຈາກຕຳແໜ່ງຂໍ້ມູນທີ່ 1 ຈິນຮອດຕຳແໜ່ງສຸດທ້າຍ. ດັ່ງສະແດງລຸ່ມນີ້



5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.3. ການຕັດໃນ sequence object

- | ສາມາດຕັດເອົາທັງໝົດໄດ້ ຫມາຍວ່າ ຈາກຕຳແໜ່ງເລີ່ມຕົ້ນ ຈິນຮອດຕຳແໜ່ງສຸດທ້າຍໄດ້.
- | ທ້າຕ້ອງການຕັດເອົາທັງໝົດຕ້ອງກຳນົດເປັນ [:]



5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.4. ຕຳແໜ່ງຂໍ້ມູນທີ່ເປັນຄ່າລົບໃນ sequence object

| ຕຳແໜ່ງຂໍ້ມູນທີ່ເປັນຄ່າລົບຍັງສາມາດໃຊ້ໃນ sequence object.

| ຕຳແໜ່ງຂໍ້ມູນທີ່ເປັນຄ່າລົບດັ່ງສະແດງໃນຮູບລຸ່ມນີ້.



5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.4. ຕຳແໜ່ງຂໍ້ມູນທີ່ເປັນຄ່າລົບໃນ sequence object

| ຕຳແໜ່ງຂໍ້ມູນທີ່ເປັນຄ່າລົບໄດ້ຖືກນໍາໃຊ້ໃນ string object.

string object element

| | | | | | |
|------|------|------|------|------|------|
| 'a' | 'b' | 'c' | 'd' | 'e' | 'f' |
| [-6] | [-5] | [-4] | [-3] | [-2] | [-1] |

```
1 str1 = 'abcdef'  
2 print(str1[-1])
```

f

```
1 print(str1[-6])
```

a

5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.4. ຕຳແໜ່ງຂໍ້ມູນທີ່ເປັນຄ່າລົບໃນ sequence object

| ຕຳແໜ່ງຂໍ້ມູນເປັນຄ່າລົບທີ່ຖືກນໍາໃຊ້ໃນ tuple object.

| | | | | | | | |
|----------------------|------|------|------|------|------|------|------|
| tuple object element | 11 | 22 | 33 | 44 | 55 | 66 | 77 |
| Negative index | [-7] | [-6] | [-5] | [-4] | [-3] | [-2] | [-1] |

```
1 tup1 = (11, 22, 33, 44, 55, 66, 77)
2 print(tup1[-1])
```

77

```
1 print(tup1[-6])
```

22

5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.5. ຕຳແໜ່ງຂໍ້ມູນທີ່ເປັນຄ່າລົບໃນ sequence object

| ຕຳແໜ່ງຂໍ້ມູນເປັນຄ່າລົບທີ່ຖືກນໍາໃຊ້ໃນ range object.

| range object element | 1 | 2 | 3 | 4 | 5 | 6 |
|----------------------|--------|--------|--------|--------|--------|--------|
| Negative index | [-6] | [-5] | [-4] | [-3] | [-2] | [-1] |

```
1 ran = range(1,7)
2 print(ran[-6])
```

1

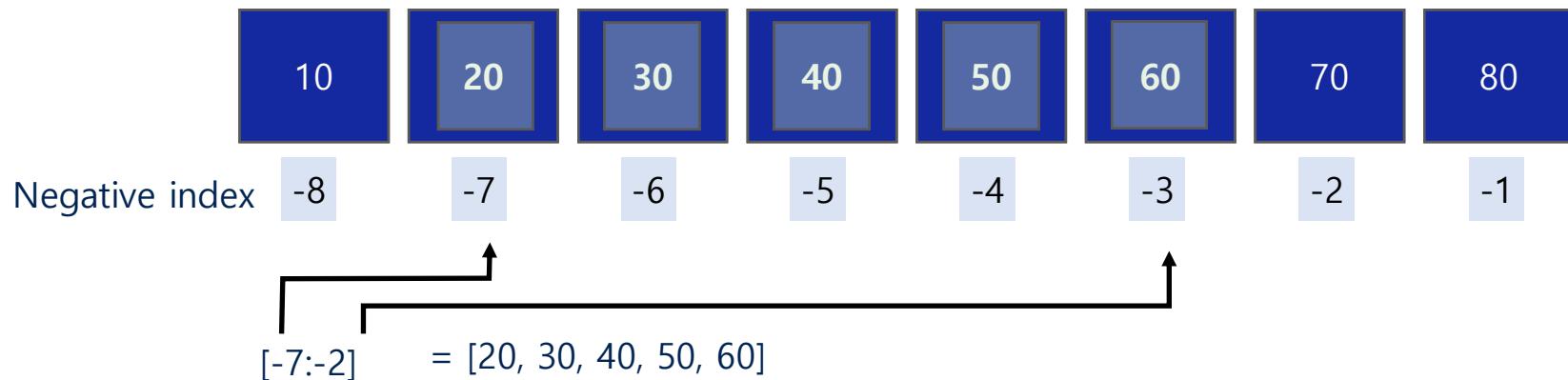
```
1 print(ran[-1])
```

6

5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.6. ການຕັດໂດຍໃຊ້ຕຳແໜ່ງຂໍ້ມູນເປັນຄ່າລົບ

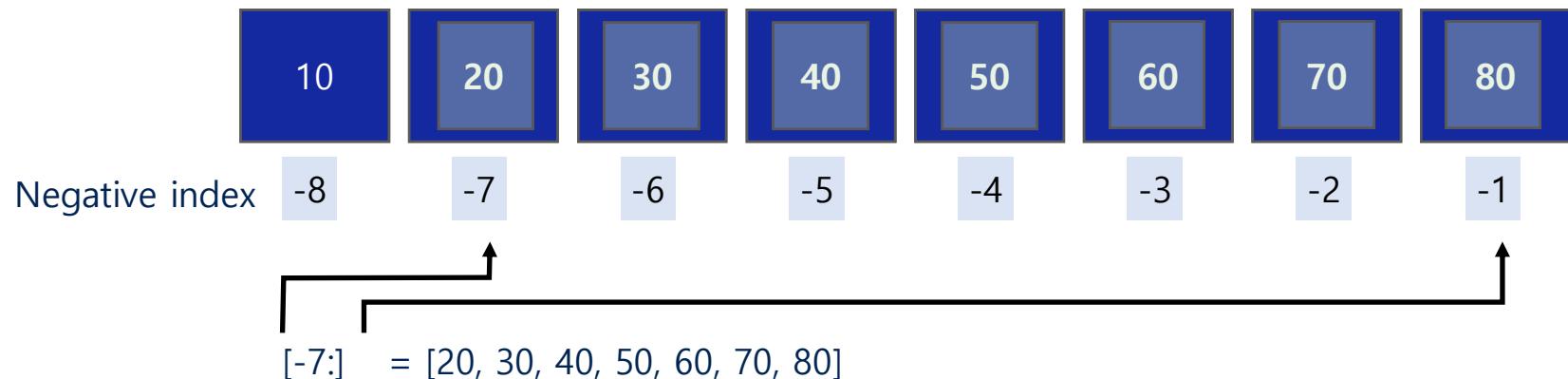
- | ລຸ່ມນີ້ ແມ່ນການຕັດໂດຍນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນເປັນຄ່າລົບ.
- | ທ້າຕັດຕຳແໜ່ງຂໍ້ມູນ $[-7:-2]$ ຈະໄດ້ຜົນຮັບແມ່ນ $[20, 30, 40, 50, 60]$.



5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.6. ການຕັດໂດຍໃຊ້ຕຳແໜ່ງຂໍ້ມູນເປັນຄ່າລົບ

| ລຸ່ມນີ້ມແມ່ນນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນເປັນຄ່າລົບ, ເພື່ອຕັດຕຳແໜ່ງຂໍ້ມູນຈົນຮອດຕຳແໜ່ງສຸດທ້າຍ, ດັ່ງສະແດງລຸ່ມນີ້.

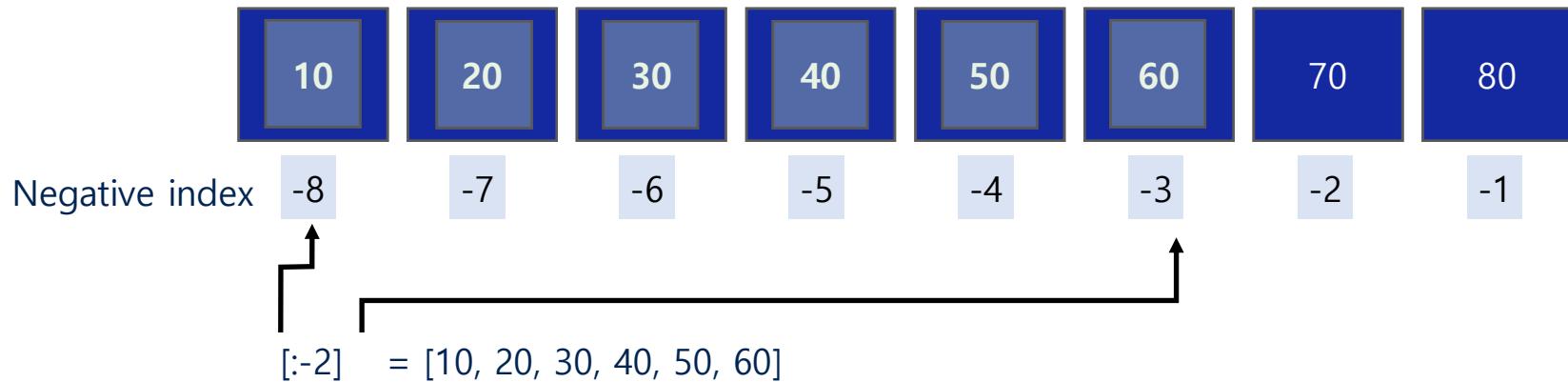


5. ນໍາໃຊ້ຕຳແໜ່ງຂໍ້ມູນໃນ Sequence Object

5.6. ການຕັດໂດຍໃຊ້ຕຳແໜ່ງຂໍ້ມູນເປັນຄ່າລົບ

| ມັນສາມາດກະໄດດຂ້າມຕຳແໜ່ງທີ່ອີດຂອງຂໍ້ມູນ ໂດຍນໍາໃຊ້ຕຳແໜ່ງທີ່ເປັນຄ່າລົບ.

| [: -2] ແມ່ນການຕັດເອົາຈິນຮອດຕຳແໜ່ງ -3



6. ການນຳໃຊ້ຝັງຊັນ len ແລະ range

6.1. ຕົວຢ່າງ Code ນຳໃຊ້ຝັງຊັນ range

| ສ້າງ list ທີ່ຂຶ້ນ even_list ໂດຍມີຄ່າຂຶ້ມູນຢູ່ລະຫວ່າງ 1 ເຖິງ 10 ແລ້ວ ນຳໃຊ້ຝັງຊັນ print ເພື່ອພິມຂໍ້ມູນອອກມາ ດັ່ງລຸ່ມນີ້.

```
1 even_list = list(range(2, 11, 2))
2 print('even_list =', even_list)
```

```
even_list = [2, 4, 6, 8, 10]
```

- ▶ ລຳດັບຂອງຕົວເລກຖືກສ້າງຈາກຝັງຊັນ range ບໍ່ຽນເປັນ list ຜ່ານ ພັງຊັນ list.
- ▶ ເນື່ອງຈາກວ່າອີງປະກອບທຳອິດຂອງ range ແມ່ນ 2, ຈະເພີ່ມເທືອ 2 ຈາກ 2 ເຖິງ 10. ສະນັ້ນຈະໄດ້ເປັນຮັບແມ່ນ 2, 4, 6, 8, 10.
- ▶ ການສ້າງ List ຈາກ range ແມ່ນເຮັດໄດ້ຢ່າຍ ດັ່ງສະແດງລຸ່ມນີ້.

```
1 list(range(10))
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
1 list(range(2, 10))
```

```
[2, 3, 4, 5, 6, 7, 8, 9]
```

```
1 list(range(2, 10, 3))
```

```
[2, 5, 8]
```

6. ການນຳໃຊ້ຟັງຊັນ len ແລະ range

6.2. ຕົວຢ່າງ Code ນຳໃຊ້ຟັງຊັນ len

| ສ້າງ list ທີ່ຂຶ້ນ "nations" ໂດຍມີອົງປະກອບແມ່ນ 'Korea', 'China', 'Russia', 'Malaysia' . ຈາກນີ້ນໃຫ້ ພິມ ອົງປະກອບຂອງ list ໂດຍນຳໃຊ້ຟັງຊັນ len ທີ່ຕໍ່າແໜ່ງຂໍ້ມູນ nations-1 ດັ່ງນີ້ len(nations)-1.

```
1 nations = ['Korea', 'China', 'Russia', 'Malaysia']
2 print('last element of nations :', nations[len(nations)-1])
```

last element of nations : Malaysia

| ໃຊ້ຊື່ປະເທດເປັນອົງປະກອບຂອງ list.
| ເອົາຄວາມຍາວຂອງ list ລົບ 1 ຈະໄດ້ຕໍ່າແໜ່ງສຸດທ້າຍຂອງ list ໂດຍໃຊ້ຟັງຊັນ len.

6. ການນຳໃຊ້ຟັງຊັນ len ແລະ range

6.3. ລະຫັດ ASCII

| ສິນມຸດ tuple ຊື່ a ມີອີງປະກອບເປັນຕົວອັກສອນແມ່ນ ('A','B','C') ແລະ tuple ຊື່ b ມີອີງປະກອບເປັນຕົວອັກສອນແມ່ນ ('A','B','D'), ໃນນີ້ ຕົວອັກສອນ C ມີຕຳແໜ່ງ 67 ແລະ D ມີຕຳແໜ່ງ 68. ສະນັ້ນ ຖ້າກວດສອບ $a > b$ ຈະສະແດງຄ່າ False , ແຕ່ກວດສອບ $a < b$ ຈະສະແດງຄ່າ True. ມັນແມ່ນການສົມທຽບຕຳແໜ່ງຂອງອີງປະກອບໃນ tuple.

```
1 a = ('A', 'B', 'C')
2 b = ('A', 'B', 'D')
```

```
1 ord('C')
```

67

```
1 ord('D')
```

68

```
1 a > b
```

False

```
1 a < b
```

True



One More Step

| ละหัด ASCII ຖືກນຳໃຊ້ໃນ strings ບໍ່?

- ▶ ละหัด ASCII ຫຍໍ້ມາຈາກ 'American Standard Code for Information Interchange.' ມັນກຳນົດ ແລະ ຈັດການຕົວເລກ ສໍາລັບແຕ່
ລະຕິວອກສອນ ແລະ ສັນຍາລັກ ແລະ ມັນແມ່ນ character code ທີ່ເປັນພື້ນຖານທີ່ສຸດ.
- ▶ ໃນການພັດທະນາຄອມພິວເຕີ, ວິທີສະແດງຕົວອກສອນ 'A' ຈະມີຄວາມແຕກຕ່າງກັນຕາມແຕ່ລະບໍລິສັດຜູ້ຜະລິດຄອມພິວເຕີ, ຕົວຢ່າງ ບໍລິສັດ
A ຈະໃຊ້ລະຫັດ 120, ສ່ວນບໍລິສັດ B ຈະໃຊ້ລະຫັດ 45 ເພື່ອສະແດງຕົວອກສອນ 'A.'
- ▶ ໃນຂະນະທີ່ອຸດສາຫະກຳຄອມພິວເຕີມີການພັດທະນາ, ມັນຈໍາເປັນຕ້ອງປ່ຽນຕົວອກສອນ ແລະ ສັນຍາລັກໃນການສະແດງຕົວເລກດຽວກັນເພື່ອ
ສື່ສານກັບ ໂປຣແກຣມ ຫຼື ຄອມພິວເຕີ. ສະນັ້ນ ລະຫັດ ASCII ໄດ້ຖືກພັດທະນາ ໂດຍນຳໃຊ້ 7 bits ເທົ່າກັບ 128 ຕົວເລກ ເພື່ອກຳນົດ ຕົວ
ອັກສອນ, ຕົວເລກ, ຕົວອັກສອນພິເສດ, ຕົວອັກສອນຄອບຄຸມ ແລະ ອື່ນໆ.

Paper coding

- ຕ້ອງເຂົ້າໃຈແນວຄິດພື້ນຖານຂອງຫຼັກສູດນີ້ໃຫ້ຄົບຖ້ວນກ່ອນຈະໄປສູ່ຂັ້ນຕອນຕໍ່ໄປ.
- ການທີ່ບໍ່ເຂົ້າໃຈແນວຄິດພື້ນຖານ ຈະເພີ່ມພາລະໃນການຮຽນຮູ້ຂອງຫຼັກສູດນີ້ ແລະ ຈະເຮັດໃຫ້ບໍ່ປະສົບຜົນສໍາເລັດ.
- ມັນອາດຈະເປັນເລື່ອງທີ່ຍາກຕອນນີ້, ແຕ່ຖ້າຢາກປະສົບຜົນສໍາເລັດໄດ້ນັ້ນ ພວກເຮົາຂໍແນະນຳໃຫ້ເຂົ້າໃຈແນວຄິດພື້ນຖານ ນີ້ຢ່າງລຶກເຊິ່ງ ແລະ ກ້າວໄປສູ່ຂັ້ນຕອນຕໍ່ໄປ.

Q1.

จากงานปั๊มตอนนี้ของ Code ถ้าสั่งลุ่มมี, จึงขวนผินได้รับของมันด้วยมี.

Conditions for Execution

จากงานปั๊มตอนนี้ของ Code ถ้าสั่งลุ่มมี, จึงขวนผินได้รับของมันด้วยมี.

Time

5 Min

Output example

```
1 t1 = 'a', 'b', 'c'
2 t2 = ('a', 'b', 'c')
3 t3 = ('d', 'e')
4
5 print(t1 == t2)
6
7 print(t1 > t3)
8
9 print(t1 < t3)
10
11 print(t2 + t3)
12
13 print([ t2 + t3 ])
14
15 print(t1)
```



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

Q2.

จากงานเก็บข้อมูลงานขายประจำวัน จำนวน 10 วัน. ให้เขียนໂຄດສ້າງເພື່ອພິມວ່າ ມີຈັກວັນທີການຂາຍຫຼຸດລົງ ຖ້າທຸຽບໃສ່ຂໍ້ມູນການຂາຍມີຜ່ານມາ. (ຂໍ້ແນະນຳ: ໃຫ້ສືມທຽບຄ່າ ດ້ວຍການເຮັດຊ້າຂອງ elements ກັບ iteration statement.)

Conditions for Execution

Daily sales record: (100, 121, 120, 130, 140, 120, 122, 123, 190, 125)
In the past 10 days, 3 days had reduced sales compared to the previous day.

Time

10 Min



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

| Let's code

1. Tuple เป็น object ที่บ่งบอกว่าได้

1.1. นิยาม tuple

- | Tuple ແມ່ນປະເພດຂໍ້ມູນທີ່ມີຫຼາຍອີງປະກອບ ຫຼື ເລື່ອນວ່າ item values.
- | ປະເພດຂໍ້ມູນທີ່ມີຫຼາຍອີງປະກອບ ເຊັ່ນ: list, tuple, dictionary ແລະ set.
- | ເຖິງຢ່າງໃດກໍຕາມ, ປະເພດຂໍ້ມູນແບບ Tuple ຈະແຕກຕ່າງກັບຂໍ້ມູນປະເພດ list ເນື່ອງຈາກວ່າ Tuple ບໍ່ສາມາດປ່ຽນລຳດັບອີງປະກອບ ຫຼັງຈາກມີການກຳນົດແລ້ວ. ດັ່ງຕົວຢ່າງນີ້,
- | ຖ້າມີການກຳນົດ Tuple ທີ່ມີອີງປະກອບດັ່ງນີ້ t = ('one', 'two', 'three') ຈະບໍ່ສາມາດປ່ຽນແປງອີງປະກອບໄດ້ ພາຍຄວາມວ່າ ບໍ່ສາມາດລືບ ຫຼື ແກ້ໄຂອີງປະກອບໄດ້.

1. Tuple เป็น object ที่บ่งบอกถึงความต่อเนื่องของข้อมูล

1.2. งานส้าง tuple

| งานส้าง tuple มีหลายวิธีที่ใช้แตกต่างกัน

| | |
|-------------------------------|---|
| ส้าง tuple หัวๆ | tuple0 = tuple() # ต้องใส่วงเล็บ |
| ส้าง tuple ที่มี 1 อยู่ประกอบ | tuple1 = (1,) # ต้องใช้แค่ 1 อย่างเดียว |
| ส้าง tuple ที่ใช้วงเล็บ | tuple2 = (1, 2, 3, 4) |
| ส้าง tuple แบบบ่วย | tuple3 = 1, 2, 3, 4 |
| ส้าง tuple จาก list | n_list = [1, 2, 3, 4] tuple4 = tuple(n_list) |

1. Tuple เป็น object ที่บ่งบอกว่ามีค่าได้

1.3. tup เป็นตัวบ่งบอกว่า tuple ที่มีชนิดเป็น Integer

- | ข้อควรระวังสำคัญ tuple ที่มีชนิดเป็น Integer
 - | สิ่งที่ต้องระวังคือ tuple ที่มีชนิดเป็น Integer ไม่ใช่ tuple เช่น tup = (100), มันจะถูกอ่านว่า tup = 100, ะนั้น tup จะถูกอ่านเป็นจำนวนเต็ม, ไม่ใช่ tuple.

```
1 tup = (100)
2 print(tup)
3 print(type(tup))
```

```
100
<class 'int'>
```

Line 1

- ถ้าเขียน tup = (100) จะได้รับเป็น tup = 100.
- ประเภทข้อมูลจะถูกอ่านเป็น int ไม่ใช่ tuple

1. Tuple เป็น object ที่บ่งบอกว่าตัวดำเนินการได้

1.4. tup เป็นตัวบ่งบอกว่า tuple เป็น tuple

| สังเคราะห์ tuple ที่ประกอบด้วยหนึ่งอิจงประกอบ, เช่น tup = (100,) เมื่ออิจงประกอบมีจุดยึดที่เดียว จะเรียกใช้ตัวบ่งบอก tup เป็นtuple.

```
tup=(100,  
print(tup)  
print(type(tup))
```

```
(100,  
<class 'tuple'>
```

Line 1

- ข้อมูล tup = (100,) ผ่านการรับเข้ามาเป็น tuple type.
- อิจงประกอบใน tup จะถูกจัดเป็น tuple.

1. Tuple เป็น object ที่บ่งบอกว่าได้

1.5. ความแตกต่างระหว่าง tuple และ list

- | Tuple บ่งบอกว่า Tuple บ่งบอกว่าได้ เช่น ตัวอย่างบ่งบอกว่าในตำแหน่งนี้มีสิ่งที่สูนจะเกิดความผิดพลาดขึ้น ถ้าจะแก้ไขมัน.
- | คุณสิ่งบังคับว่า Immutable.

TypeError

```
1 t = (0, 1, 2, 3, 4)
2 t[0] = 100
```

```
TypeError                                 Traceback (most recent call last)
<ipython-input-78-9d561bf19fbf> in <module>
      1 t = (0, 1, 2, 3, 4)
----> 2 t[0] = 100
```

TypeError: 'tuple' object does not support item assignment

1. Tuple เป็น object ที่บล็อกสามารถปูนแบบได้

1.6. Packing และ unpacking

- | Packing: หมายความว่าการพิมพ์ข้อมูลใน tuple ลงในตัวแปร.
- | Unpacking: หมายความว่าการอ่านข้อมูลจากตัวแปร tuple.

| Packing | Unpacking |
|--|---|
| <pre>1 a = (1, 2) # Tuple packing 2 a[0] # Reference for tuple item</pre> <p>1</p> | <pre>1 c = (3, 4) # Tuple packing 2 x, y = c # Assigning to 2 variables by unpacking tuple c 3 x</pre> <p>3</p> |
| <pre>1 a[1]</pre> <p>2</p> | <pre>1 y</pre> <p>4</p> |

1. Tuple เป็น object ที่บ้านมาดปรุงเปลี่ยนได้

1.7. Swap

- | a เก็บค่า 100 และ b เก็บค่า 200, สิ่งที่ต้องเรียกว่า swap ห้างสองค่าในตัวปูนดังกล่าว.
- | Code คำสั่งลุமน์แม่นวิธีการ swap ที่ว่าไปโดยมิใช้ในภาษา C หรือ Java.
- | สำลับงาน swap, ตัวปูน temp เช่นเป็นตัวปูนชื่อ temp ให้เพื่อเรียกหิดสอนบดังจะดูดี.
- | แต่ที่ใน Python จะใช้วิธีที่ง่ายกว่าภาษา C หรือ Java.

```
1 a = 100
2 b = 200
3 print('before swap : a = ', a, 'b = ', b)
4 temp = a
5 a = b
6 b = temp
7 print('after swap : a = ', a, 'b = ', b)
```

```
before swap : a = 100 b = 200
after swap : a = 200 b= 100
```

1. Tuple เป็น object ที่บ่งบอกว่าได้

1.7. Swap

- | นำใช้ Python tuple เพื่อให้ง่ายสำหรับการเขียน Code คำสั่ง เพื่อ swap.
- | ใน Python, ในการ swap จะมีรูปแบบ a, b = b, a ดังจะแสดงลุமน์.
- | Code คำสั่งดังกล่าวจะเปลี่ยนรูปแบบ.

```
1 a = 100
2 b = 200
3 print('before swap : a = ', a, 'b = ', b)
4 a, b = b, a # very simple swap
5 print('swap result using tuple: a = ', a, 'b = ', b)
```

before swap : a = 100 b = 200

swap result using tuple: a = 200 b = 100

1. Tuple เป็น object ที่บໍ່ສາມາດປ່ຽນແປງຄ່າໄດ້

1.8. ການຈັດລຽງ

- | ເນື່ອງຈາກອີງປະກອບຂອງ tuple ບໍ່ສາມາດປ່ຽນແປງຄ່າໄດ້ໃນ, ສະນັ້ນ ໃນ Tuple ຈຶ່ງບໍ່ສາມາດຈັດລຽງຂໍ້ມູນໄດ້.
- | ຖ້າຕ້ອງການຈັດລຽງຂໍ້ມູນໃນ tuple, ຕ້ອງແປງ tuple ໃຫ້ເປັນ list ຈຶ່ງສາມາດຈັດລຽງໄດ້. ສໍາລັບການຈັດລຽງແມ່ນໃຊ້ພິ່ງຊັ້ນ sort.

```
1 tup = (1, 2, 5, 4, 3, 2, 9, 3, 7, 3, 9)
2 temp = list(tup)
3 temp.sort()
4 print(temp)
```

```
[1, 2, 2, 3, 3, 3, 4, 5, 7, 9, 9]
```

| Pair programming



Pair Programming Practice

| ແນວທາງ, ກິນໄກ ແລະ ແຜນສຸກເສີນ

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

| ຈັບຄຸ້ທີ່ຄ້າຢັກນັ້ນ, ບໍ່ຈໍາເປັນເຫົ້າທຽມກັນ, ຄວາມສາມາດເປັນຄຸ້ຮ່ວມງານ

ການຈັບຄຸ້ຂຽນໂປຣແກຣມ ຈະໄດ້ຮັບຜົນທີ່ກຳຕໍ່ເນື້ອນັກຮຽນມີຄວາມສາມາດຄ້າຢັກນັ້ນ ຫາຍວ່າ ບໍ່ຈໍາເປັນຕ້ອງມີຄວາມສາມາດຄ້າຢັກນັ້ນກໍໄດ້, ແຕ່ວ່າ ການຈັບຄຸ້ ນັກຮຽນທີ່ມີຄວາມສາມາດແຕກຕ່າງກັນຫຼາຍ ກ່າວຈະຮັດໃຫ້ບໍ່ສົມດຸນກັນ. ຄຸສອນຮູ້ດີວ່າ ການຈັບຄຸ້ກັນບໍ່ແມ່ນ ຍຸດທະສາດ “ແບ່ງເພື່ອເອົາຊະນະ” ແຕ່ເປັນຄວາມ ພະຍາຍາມຮັດວຽກຮ່ວມກັນຂອງນັກຮຽນໃຫ້ປະສົບຜົນສໍາເລັດ. ຄຄວນທີ່ກາເວັ້ນການຈັບຄຸ້ກັນລະຫວ່າງນັກຮຽນອ່ອນ ແລະ ນັກຮຽນເກົ່າງ.

| ກະຕຸ້ນນັກຮຽນໂດຍການໃຫ້ສິ່ງຈຸງໃຈພື້ນເສດ

ຂໍສະເໜີແຮງຈຸງໃຈທີ່ຮັດໃຫ້ນັກຮຽນຈັບຄຸ້, ໂດຍສະເພາະນັກຮຽນທີ່ມີຄວາມສາມາດສຸງ. ບາງຄຸສອນໄດ້ພື້ນວ່າ ການຈັບຄຸ້ຮັດວຽກມອບໝາຍ ແມ່ນມີ ປະໂຫຍດ ສໍາລັບໜຶ່ງ ຫຼື ສອງວຽກມອບເທົ່ານັ້ນ



Pair Programming Practice

| ចំណាំការបំពើនិងរុញខែងមករុញ

ສິ່ງທີ່ຫາຍສໍາລັບຄູແມ່ນເພື່ອຊອກຫາວິທີທີ່ຈະປະເມີນຜົນການຮຽນຂອງນັກຮຽນ, ຄູ້ຮັ້ງບໍ່ວ່າ ນັກຮຽນ ໄດ້ຕັ້ງໃຈຮຽນ ຫຼື ບໍ່ຕັ້ງໃຈຮຽນ. ຜູ້ສ່ວວຊານໄດ້ແນະນຳໃຫ້ທີ່ບໍ່ທວນການອອກແບບຫຼັກສຸດການຮຽນ ແລະ ຮູບແບບການປະເມີນ ພ້ອມທັງປີກສາຫາລືຢ່າງຈິງຈຳກັບນັກຮຽນ ກ່ຽວກັບພິດຕິກຳທີ່ຈະບໍ່ຕັ້ງໃຈຮຽນ ນອກຈາກນີ້ຢັ້ງໄດ້ແນະນຳມາອບວຽກມອບໝາຍໃຫ້ນັກຮຽນ ພ້ອມທັງອະທິບາຍໃຫ້ເຂົ້າເຈົ້າຢ່າງຈະເຈັ້ງ

| ສະພາບແວດລ້ອມຂອງການຮຽນຮັກ

ສະພາບແວດລ້ອມການຮຽນຮູ້ຮ່ວມກັນເກີດຂຶ້ນໄດ້ທຸກເວລາທີ່ຜູ້ສອນຮຽກຮ້ອງໃຫ້ນັກຮຽນເຮັດວຽກຮ່ວມກັນໃນກິດຈະກຳການຮຽນຮູ້ ເຊິ່ງອາດຈະເປັນກິດຈະກຳທີ່ເປັນທາງການ ແລະ ບໍ່ເປັນທາງການ ແລະ ອາດຈະບໍ່ລວມເຖິງການປະເມີນຜົນການຮຽນໂດຍກິງ. ເຊັ່ນຕົວຢ່າງ ໃຫ້ນັກຮຽນຈັບຄຸ້ງກັນເພື່ອເຮັດວຽກອບໝາຍ ໂດຍນັກຮຽນຈະຕ້ອງທີບທວນກ່ຽວກັບການສອນຂອງອາຈານທີ່ຜ່ານມາ ແລະ ລະດົມແນວຄິດພາຍໃນກຸ່ມ ພ້ອມທັງມືການແບ່ງວຽກໃຫ້ແຕ່ລະຄົນຮັບຜິດຊອບ ຈາກນັ້ນກໍໃຫ້ມີການແລກປ່ຽນຄວາມຄິດເຫັນເຊິ່ງກັນ ແລະ ກັນ ເພື່ອເຮັດວຽກອບໝາຍໃຫ້ສໍາເລັດຕາມເປົ້າໝາຍທີ່ວ່າງໄວ້.

Q1. จากผิบันไดรับในตัวอย่างลุ่มนี้, จึงชูน Code ถ้าสั่งเพื่อให้พิมค่าໃຫຍ່ສຸດອອກมาທາງໜ້າຈຳ.

Output example

ກຳນົດໃຫ້ tuples: (1, 2, 5, 4, 3, 2, 1, 4, 7, 8, 9, 9, 3, 7, 3, 9)
ຄ່າໃຫຍ່ສຸດຂອງອີງປະກອບແມ່ນ : 9

Q2.

ຈາກເປັນໄດ້ຮັບໃນຕົວຢ່າງລຸ່ມນີ້, ມີ tuple ບັນຈຸບັນດາອີງປະກອບທີ່ແຕກຕ່າງກັນເຊື່ອ: Tuple ຫວ່າງເປົ່າ, string ຫວ່າງເປົ່າ ແລະ list ຫວ່າງເປົ່າ.
ຈຶ່ງຂຽນ Code ເພື່ອລືບ tuple ຫວ່າງເປົ່າ, string ຫວ່າງເປົ່າ ແລະ list ຫວ່າງເປົ່າ ດັ່ງລຸ່ມນີ້. (ໂດຍບໍ່ໃຫ້ລືບ (,) tuple ເພາະວ່າ ມັນແມ່ນ tuple
ຫວ່າງເປົ່າ.)

Output example

ກຳນົດໃຫ້ tuples: (), (1,), [], 'abc', (), (), (1,), ('a'), ('a', 'b'), ((),), ''
ຜົນໄດ້ຮັບແມ່ນ: [(1,), 'abc', (1,), ('a'), ('a', 'b'), ((),)]