Unit 8.

Loop-1

Learning objectives

- ✓ ทำความเข้าใจจุดประสงค์ของคำสั่งวนซ้ำ และ ข้อดีของคำสั่งวนซ้ำ.
- ✓ ทำความเข้าใจแนวความคิดของฟังชัน range และ การนำใช้เพื่อวนซ้ำ.
- ✓ ทำความเข้าใจ และ นำใช้ for syntax และ list.

Lesson overview

- ✓ ເຂົ້າໃຈຄວາມຈຳເປັນຂອງການວົນຊ້ຳ ແລະ ຮູ້ວິທີການນຳໃຊ້
- ✓ ການປະຕິບັດຊ້ຳຕາມຈຳນວນຄັ້ງທີ່ກຳນົດໄວ້ ໂດຍໃຊ້ for loops
- ✓ ຮຽນຮູ້ວິທີການຂຽນການຊ້ຳແບບຕ່າງໆ ໂດຍນຳໃຊ້ຄຳສັ່ງ for ແລະ range
- ✓ ຮຽນຮູ້ວິທີການສ້າງລຳດັບຂອງຕົວເລກຕ່າງໆ ໂດຍໃຊ້ຟັງຊັນ range ແລະ lists

ແນວຄວາມຄິດທີ່ເຈົ້າຈະຕ້ອງຮູ້ຈາກບົດຮຽນທີ່ຜ່ານມາ

- ✓ ທຳຄວາມເຂົ້າໃຈຄຳສັ່ງທີ່ເງື່ອນໄຂຕ່າງໆເຊັ່ນ: ຄຳສັ່ງ if ແລະ ຄຳສັ່ງ if-elif-else
- ✓ ທຳຄວາມເຂົ້າໃຈກ່ຽວກັບໂຄງສ້າງຂອງຄຳສັ່ງທີ່ປະຕິບັດການໄດ້ແບບບໍ່ຕໍ່ເນື່ອງ, ເຊິ່ງ Block ດ້ານລຸ່ມຈະຖືກດຳເນີນການເມື່ອກົງກັບເງື່ອນໄຂທີ່ກຳໜົດ.

Keyword

- for
- in
- iteration
- range

Mission

1. Real world problem

1.1. ໂລກອ້ວນເປັນສາເຫດຂອງພະຍາດຕ່າງໆ ແລະ ເພີ່ມພາລະຂອງຄ່າໃຊ້ຈ່າຍທາງການແພດໃນສັງຄົມໂດຍລວມ

http://www.bio-ethiks.com/index.php/2018/08/27/the-obesity-crisis-and-social-justice-part-1/

- ໂດຍທົ່ວໄປ, ໂລກອ້ວນແມ່ນເກີດຈາກການເພີ່ມຂຶ້ນຂອງໄຂມັນໃນຮ່າງກາຍ ເມື່ອໄດ້ຮັບພະລັງງານສູງກວ່າການນຳໃຊ້ພະລັງງານ. ໂລກອ້ວນແບບທີສອງ ແມ່ນເກີດມາຈາກການສືບພັນ, ພະຍາດຕ່ອມບໍ່ມີທໍ່ (ກຸ່ມອາການຮວຍໄຂ່ມີຖົງ ນ້ຳຫຼາຍໜ່ວຍ, ສາຍພັນອິນຊູລິນ (insulin) ແລະ ອື່ນໆ), ຢາເສບຕິດ ແລະ ອື່ນໆ.
- ໂລກອ້ວນບໍ່ໄດ້ຢຸດຢູ່ພຽງແຕ່ເປັນໂລກອ້ວນເທົ່ານັ້ນ, ແຕ່ສາມາດເປັນຕົ້ນເຫດ ຂອງພະຍາດອື່ນໆຫຼາກຫຼາຍ ແລະ ຍັງສາມາດນຳໄປສູ່ການເປັນພະຍາດທາງ ຈິດ.
- ພະຍາດເບົາຫວານປະເພດ 2, ພາວະໄຂມັນໃນເສັ້ນເລືອດຜິດປົກະຕິ, ຄວາມດັນເລືອດສູງ, ໄຂມັນຫຸ້ມຕັບ, ພະຍາດຕ່ອມນ້ຳບີ, ພະຍາດເສັ້ນເລືອດ ໃນຫົວໃຈຕີບ (ພະຍາດເສັ້ນເລືອດໃນຫົວໃຈຕີບ, ກ້າມເນື້ອຫົວໃຈຕາຍ), ໂລກ ຫຼອດເລືອດຕັນໃນສະໝອງ, ພາວະຢຸດຫາຍໃຈເວລານອນ, ໂລກເກົ້າ, ຂໍ່ເຂົ່າ ເຊື່ອມ, ປະຈຳເດືອນຜິດປົກກະຕິ, ມະເຮັງລຳໄສ້ໃຫຍ່ ແລະ ມະເຮັງເຕົ້ານົມ ເປັນຕົ້ນ.
- ໂລກອ້ວນເປັນທັງບັນຫາສ່ວນບຸກຄົນ ແລະ ເປັນບັນຫາສັງຄົມ ທີ່ເພີ່ມພາລະ ຄ່າປິ່ນປົວໃຫ້ກັບສັງຄົມໂດຍລວມ.

1. Real world problem

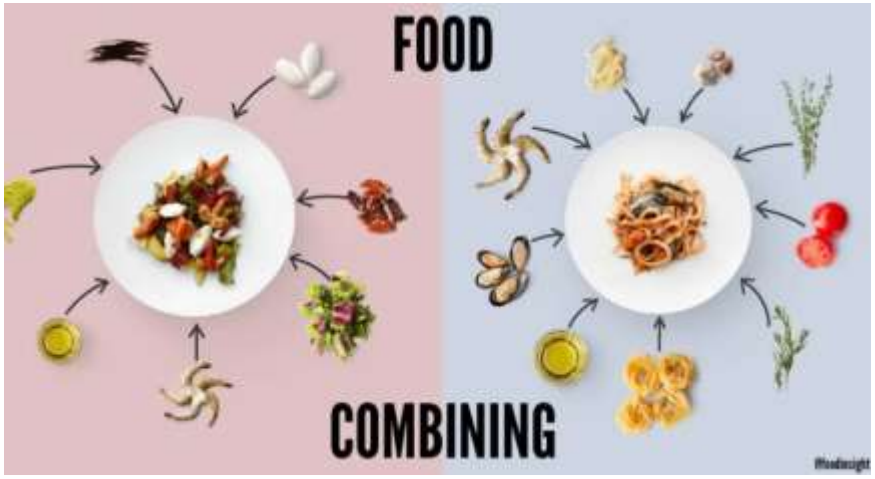
1.2. ວິທີການປ້ອງກັນໂລກອ້ວນ

https://timesofindia.indiatimes.com/life-style/food-news/10-food-items-you-should-avoid-to-stay-healthy/articleshow/68558987.cms

- ສິ່ງທຳອິດທີ່ແຕ່ລະຄົນສາມາດເຮັດໄດ້ເພື່ອຫຼີກເວັ້ນການເປັນໂລກອ້ວນແມ່ນຫຼີກລ້ຽງການກິນອາຫານຫຼາຍເກີນໄປ, ຫຼຸດຜ່ອນອາຫານສຳເລັດຮູບ, ໃຊ້ວັດຖຸດິບສົດ ແລະ ໃໝ່ ໃນອາຫານ.
- ໃນພາລະກິດນີ້, ພວກເຮົາຈະຄົ້ນຫາສູດການປະສົມສ່ວນປະກອບຕ່າງໆທີ່ສາມາດສະໜອງໃຫ້ບຸກຄົນ ເພື່ອແກ້ໄຂບັນຫາໂລກອ້ວນ.
- ເປົ້າໝາຍສຸດທ້າຍຂອງພາລະກິດນີ້ແມ່ນການຕັ້ງຮ້ານຂາຍແຊນວິດ (sandwich) ແລະ ສ້າງສ່ວນປະສົມຂອງແຊນວິດທຸກຢ່າງທີ່ເປັນໄປໄດ້ດ້ວຍສ່ວນປະສົມທາງຮ້ານສະໜອງໃຫ້ ລວມທັງເຂົ້າຈີ່ອົມຊີຕ່າງໆ, ຜັກ, ໄສ້ກອກ, ນ້ຳຊັອດ ແລະ ອື່ນໆ.

2. Solution

2.1. ການປະສົມອາຫານແມ່ນຫຍັງ?



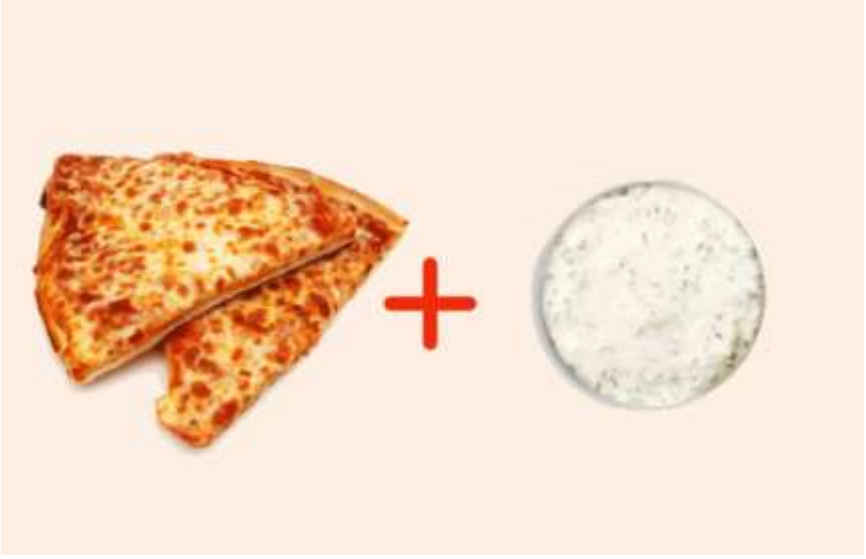
https://foodinsight.org/what-is-food-combining/

- ຊະນິດຂອງຜັກ ແລະ ໄສ້ກອກ ທີ່ໃຊ້ໃນການເຮັດອາຫານແຕກຕ່າງກັນ ແລະ ມີປະຫວັດຄວາມເປັນມາອັນຍາວນານ.
- ການປະສົມປະສານຂອງຜັກຕ່າງໆ ແລະ ໄສ້ກອກ ເຮັດໃຫ້ລົດຊາດເຂັ້ມຂຸ້ນດີ.
- ການຫຼຸດຜ່ອນການບໍລິໂພກນ້ຳຕານ, ໂດຍການບໍລິໂພກ ລົດຊາດທີ່ເຂັ້ມຂຸ້ນນີ້ຈະເປັນການນຳໃຊ້ວິທີການປຸງແຕ່ງອາຫານທີ່ດີ.

2. Solution

2.2. ຂໍ້ມູນເພີ່ມເຕີມສຳລັບການປະສົມອາຫານ

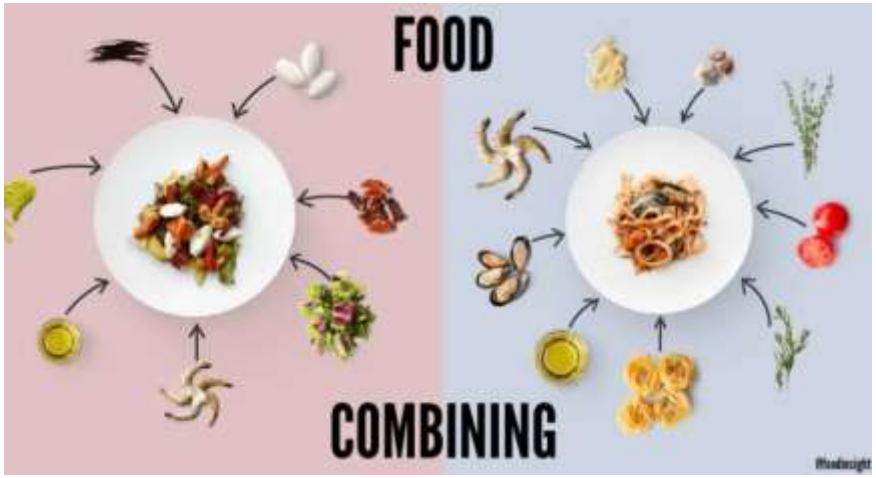
- ກວດເບິ່ງ https://www.eatthis.com/weird-food-combinations/ ສຳລັບຂໍ້ມູນເພີ່ມເຕີມກ່ຽວກັບການປຸງແຕ່ງອາຫານ
- ເວັບໄຊນີ້ບອກເລົ່າເຖິງຄວາມສະໜຸກສະໜານ ແລະ ຄວາມແຊບທີ່ພວກເຮົາບໍ່ໄດ້ຄິດເຖິງມາກ່ອນ.



https://www.eatthis.com/weird-food-combinations/

3. Mission

3.1. ວິທີປຸງແຕ່ງອາຫານໃຫ້ແຊບ



https://foodinsight.org/what-is-food-combining/

- David ເປີດຮ້ານ sandwich ໃໝ່. David ຕ້ອງການ ສູດແຊນວິດທຸກປະເພດທີ່ສາມາດເຮັດໄດ້ໂດຍການປະສົມ ເຂົ້າກັນຂອງເຂົ້າຈີ່, ຊີ້ນ, ຜັກ, ໄສ້ກອກ ແລະ ຊັອດ ທີ່ມີ ຢູ່ໃນຮ້ານຂອງລາວ.
- ປະເພດເຂົ້າຈີ່ຂອງ David ປະກອບມີ: "ເຂົ້າຈີ່ Rye“, “ແປ້ງສາລີ" ແລະ "ສີຂາວ“. ປະເພດຊີ້ນປະກອບມີ: "Meatball", "ໄສ້ກອກ", "ເອີກໄກ່“. ປະເພດຜັກປະ ກອບມີ: "ຜັກກາດ", "ໝາກເລັ່ນ", "ໝາກແຕງ" ແລະ ນ້ຳຊອດໄດ້ແກ່: "Mayonnaise”, "Honey Mustard", ແລະ "Chili".
- ສ່ວນປະສົມແຕ່ລະປະເພດສາມາດເລືອກໄດ້ພຽງແຕ່ຄັ້ງ ດຽວເທົ່ານັ້ນ. ຊຸດການປະສົມປະສານທີ່ເປັນໄປໄດ້ແມ່ນ ຫຍັງ? ໃຫ້ພິມການປະສົມທັງໝົດດັ່ງຕໍ່ໄປນີ້:

3. Mission

3.2. ขั้นตอนการปรุงแต่งอาหารแซบ

```
1  breads = ["Rye bread", "Wheat", "White"]
2  meats = ["Meatball", "Sausage", "Chicken breast"]
3  vegis = ["Lettuce", "Tomato", "Cucumber"]
4  sauces = ["Mayonnaise", "Honey Mustard", "Chili"]
5
6  print('Possible combinations of David's sandwich shop')
7  for b in breads:
8      for m in meats:
9          for v in vegis:
10             for s in sauces:
11                 print(b+ " + "+ m+ " + "+ v+ " + "+ s)
12
```

```
Possible combinations of David's sandwich shop
Rye bread + Meatball + Lettuce + Mayonnaise
Rye bread + Meatball + Lettuce + Honey Mustard
Rye bread + Meatball + Lettuce + Chili
Rye bread + Meatball + Tomato + Mayonnaise
Rye bread + Meatball + Tomato + Honey Mustard
Rye bread + Meatball + Tomato + Chili
Rye bread + Meatball + Cucumber + Mayonnaise
Rye bread + Meatball + Cucumber + Honey Mustard
```

3. Mission

3.3. แผนภาพขั้นโปรแกรม

Pseudocode

[1] Start

[2] Define list-type data containing food ingredients.

[3] the number of elements in for b in bread list :

[4] the number of elements in for m in meats list :

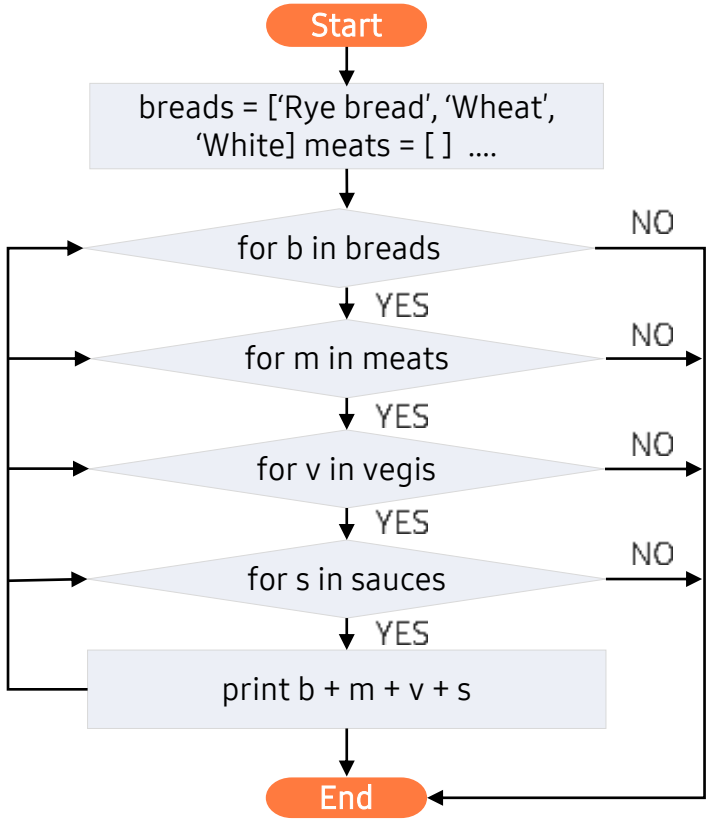
[5] the number of elements in for v in vegis list :

[6] the number of elements in for s in sauces list :

[7] print b + m + v + s.

[8] end

Flowchart



3. Mission

3.4. Code สุดท้ายในการปรุงแต่งอาหารที่แซบ

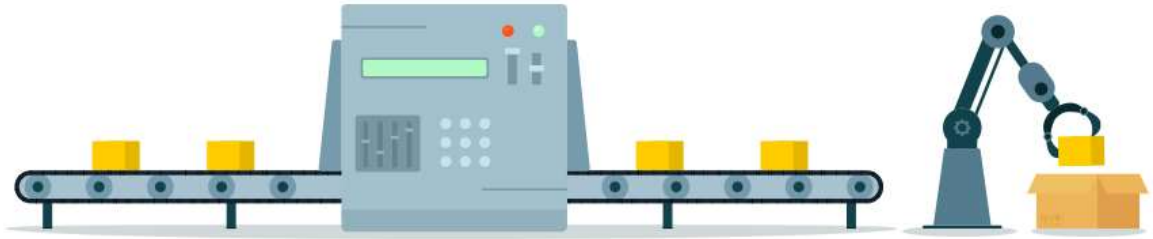
```
breads = ["Rye bread", "Wheat", "White"]
meats = ["Meatball", "Sausage", "Chicken breast"]
vegis = ["Lettuce", "Tomato", "Cucumber"]
sauces = ["Mayonnaise", "Honey Mustard", "Chili"]

print('Possible combinations of David's sandwich shop')
for b in breads:
    for m in meats:
        for v in vegis:
            for s in sauces:
                print(b+ " + "+ m+ " + "+ v+ " + "+ s)

```

Key concept

1. ພື້ນຖານຂອງຄຳສັ່ງ for

1.1. ຄວາມຈຳເປັນຂອງຄຳສັ່ງ for

- ການວົນຊ້ຳ code ຕາມເງື່ອນໄຂຈະຊ່ວຍແກ້ໄຂບັນຫາທີ່ສັບຊ້ອນໄດ້.
- ຕົວຢ່າງເຊັ່ນ ໂປຣແກຣມຈະມີຄວາມກະທັດຮັດຫຼາຍຂຶ້ນ ໂດຍການໃຊ້ໂຄງສ້າງແບບວົນຊ້ຳ ແທນການຄັດລອກ ແລະ ວາງຄຳສັ່ງ ບາງຢ່າງເພື່ອເຮັດວຽກເດີມຊ້ຳອີກ.
- ນອກຈາກນັ້ນ, ຄຳສັ່ງການວົນຊ້ຳສາມາດຫຼຸດເວລາໃນການຂຽນໂປຣແກຣມ.



1. ພື້ນຖານຂອງຄຳສັ່ງ for

1.1. ຄວາມຈຳເປັນຂອງຄຳສັ່ງ for

```
1 print('Welcome to everyone!!')
2 print('Welcome to everyone!!')
3 print('Welcome to everyone!!')
4 print('Welcome to everyone!!')
5 print('Welcome to everyone!!')
```

```
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
```

- ຂຽນຟັງຊັນການພິມ 5 ເທື່ອ. ເພື່ອພິມ 5 ເທື່ອຊ້ຳໆກັນ ໂດຍບ່ໃຊ້ຄຳສັ່ງວົນຊ້ຳ.
- ລອງເຮັດຊ້ຳວຽກງານນີ້ 1000 ເທື່ອ.
- ການເຮັດແບບນີ້ບ່ມີປະສິດທິພາບ.

1. ພື້ນຖານຂອງຄຳສັ່ງ for

1.1. ຄວາມຈຳເປັນຂອງຄຳສັ່ງ for

Why is repetition important?

```
1 for i in range(5):
2     print('Welcome to everyone!!')
```

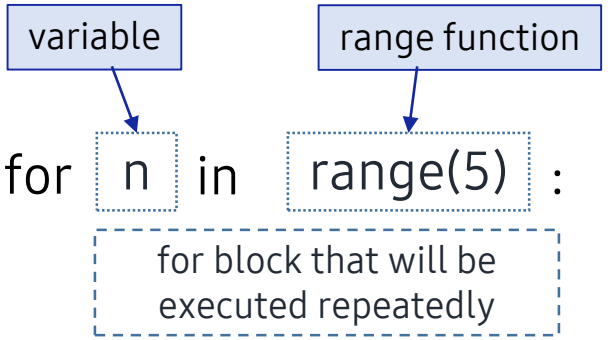
```
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
```

- ແນ່ນອນ ສຳລັບການທຳຊ້ຳພຽງບໍ່ເທົ່າໃດຄັ້ງ, ເຮົາສາມາດ "copy and paste" ຕາມຂ້າງເທິງໄດ້
- ແຕ່ເຮົາສາມາດເຮັດຊ້ຳອີກ 100 ເທື່ອໄດ້ ຫຼື ບໍ່?
- ໃນກໍລະນີນີ້, ສາມາດຂຽນ code ງ່າຍໆໄດ້ ໂດຍໃຊ້ຄຳສັ່ງ for.
- ຄຳສັ່ງວົນຊ້ຳເປັນຄຳສັ່ງຄວບຄຸມທີ່ສຳຄັນ ທີ່ຈັດການກັບວຽກງານຊ້ຳຊ້ອນທີ່ບໍ່ສະດວກ.

1. ພື້ນຖານຂອງຄຳສັ່ງ for

1.2. syntax ພື້ນຖານຂອງຄຳສັ່ງ range ແລະ loop

- ຟັງຊັນ range ສ້າງລຳດັບຈຳນວນຖ້ວນຂອງຊ່ວງເວລາໃດໜຶ່ງ.
- ສາມາດນຳໃຊ້ຈຳນວນວົນຊ້ຳໃນ for loop ໄດ້.



1. ພື້ນຖານຂອງຄຳສັ່ງ for

1.3. ຕົວຢ່າງຄຳສັ່ງ for

```
1 for i in range(10):
2     print('Welcome to everyone!!')
```

```
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
```

- ເພື່ອຕ້ອງການຊຸ້ມຊ້ຳ code ຂ້າງເທິງ 10 ເທື່ອ, ກໍໃຫ້ໃສ່ເລກ 10 ພາຍໃນວົງເລັບ ()
- code ທີ່ງ່າຍດາຍນີ້ໄດ້ຖືກປະຕິບັດຈຳນວນຫຼວງຫຼາຍ, ການດັດແກ້ code ແລະ ການບຳລຸງຮັກສາ ຈະງ່າຍຂຶ້ນ.
- ເພື່ອຕ້ອງການເຮັດຊ້ຳ 100 ເທື່ອ: ກໍປ່ຽນເປັນ range(100)

1. ພື້ນຖານຂອງຄຳສັ່ງ for

1.4. ຕົວປ່ຽນຄວບຄຸມ loop

ตัวอักสอม i, j, k, l, ... ถูกใช้เป็นตัวปรับควบคุมใน loop.

```
1 for i in range(10):
2     print(i, 'Welcome to everyone!!')
```

```
0 Welcome to everyone!!
1 Welcome to everyone!!
2 Welcome to everyone!!
3 Welcome to everyone!!
4 Welcome to everyone!!
5 Welcome to everyone!!
6 Welcome to everyone!!
7 Welcome to everyone!!
8 Welcome to everyone!!
9 Welcome to everyone!!
```

- ตัวปรับที่ใช้สำหรับคำสั่งการวนซ้ำ โดยทั่วไปแม่นถูกกำหนดด้วยตัวอักสอม i, j, k, l,...
- ตัวปรับเหล่านี้เรียกว่าตัวปรับการควบคุม loop ในพาสา C หรือ Java.

One More Step

ການລະບຸຊື່ຕົວປ່ຽນຄວບຄຸມ loop

- ເນື່ອງຈາກຕົວປ່ຽນ i ທີ່ຖືກກຳນົດໃໝ່ໃນຄຳສັ່ງວົນຊ້ຳຂ້າງເທິງ ແມ່ນຕົວປ່ຽນວົນຊ້ຳທີ່ບໍ່ຖືກນຳໃຊ້ໃນຄຳສັ່ງດຳເນີນການ, ສາມາດຍົກເລີກການເປີດເຜີຍຕົວປ່ຽນໄດ້, ໂດຍການປ່ຽນແທນມັນດ້ວຍເຄື່ອງໝາຍ underscore (_) ດັ່ງຕໍ່ໄປນີ້

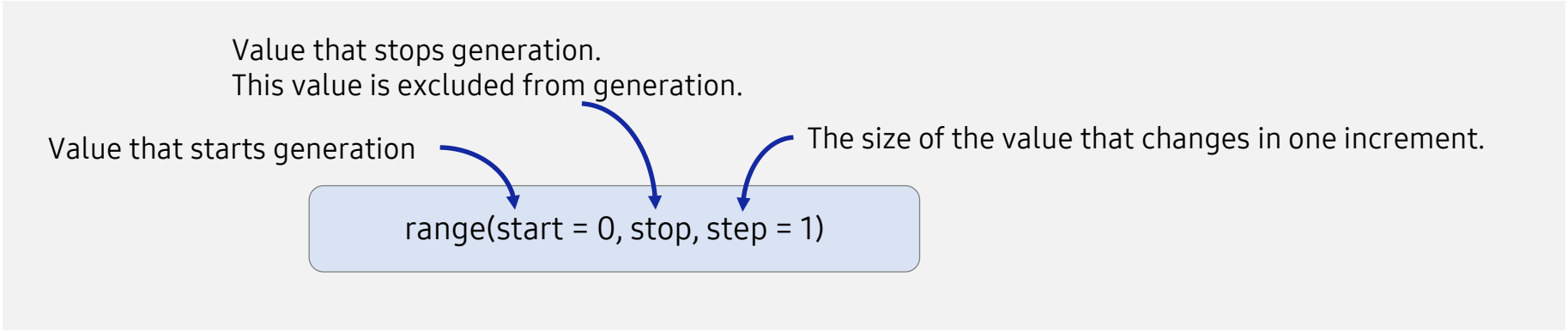
```
1 for _ in range(10):
2     print('Welcome to everyone!!')
```

```
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
Welcome to everyone!!
```

2. range

2.1. ໂຄງສ້າງຂອງ range

- ສາມາດສ້າງຈຳນວນຖ້ວນຕໍ່ເນື່ອງກັນລະຫວ່າງຄ່າເລີ່ມຕົ້ນ ແລະ ຄ່າສຸດທ້າຍ ເຊັ່ນ: ໃນ range(0, 5) ຫຼື ສາມາດໃສ່ຈຳນວນທີ່ຈະເພີ່ມຄ່າຂຶ້ນແຕ່ລະຄັ້ງໃນຕອນທ້າຍເຊັ່ນ: range(0, 5, 2)
- ໃນ range (0, 5, 1), 1 ເປັນຕົວເລກສຸດທ້າຍແມ່ນຄ່າເລີ່ມຕົ້ນທີ່ຈະເພີ່ມຂຶ້ນໃນແຕ່ລະຄັ້ງ.



- ຄ່າ range(0, stop, 1) ເປັນການສ້າງຈຳນວນຖ້ວນຈາກ 0 ຫາ (stop-1) ເພີ່ມຄ່າຂຶ້ນເທື່ອລະ 1.

One More Step

ອົງປະກອບຂອງຟັງຊັນ range ບໍ່ສາມາດເປັນຈຳນວນຈິງໄດ້.

- ฟังชัน range บ่สามาดมิ arguments เป็นจำนวนจิงได้. เถิงย่างใดกำตาม, เมื่อเอิ้นเอาฟังชัน arrange() จากโมดูน numpy (เซิ่งจะฮยนรู้ในพายหลัง) สามาดใช้ตัวเลกจำนวนจิงสำลับค่าเล่ิม ต้น และ ค่าสุดท้าย.

TypeError

```
1 for i in range(0.9):
2     print(i, 'Welcome to everyone!!')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-11-332ae3dce87d> in <module>
----> 1 for i in range(0.9):
      2     print(i, 'Welcome to everyone!!')

TypeError: 'float' object cannot be interpreted as an integer
```

2. range

2.2. สำหลับ range

สามาดสร้างลายทามได้ง่าย, โดยใช้ฟังชัน list ดั่งต่ำไปนี้

```
1 print(list(range(0, 5, 1)))
```

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

```
1 print(list(range(0, 5, 2)))
```

```
[0, 2, 4]
```

```
1 print(list(range(2, 5)))
```

```
[2, 3, 4]
```

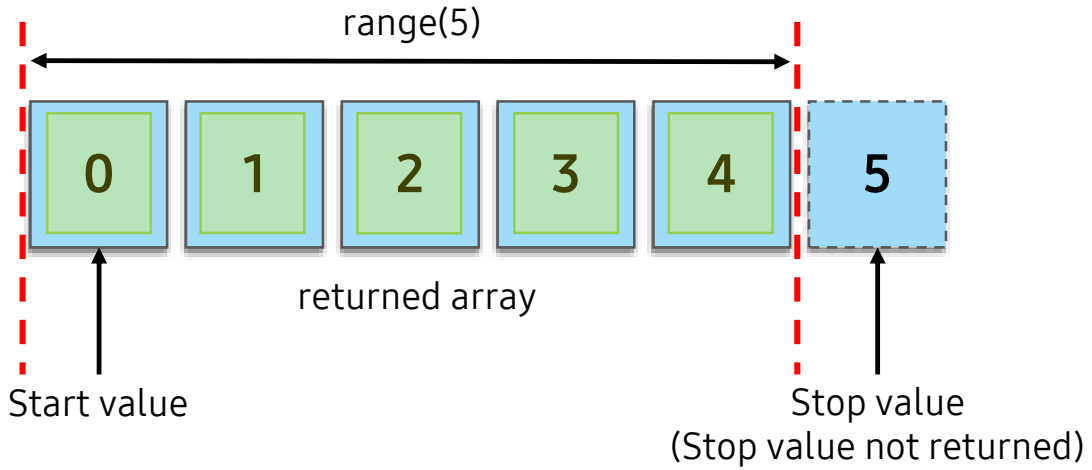
```
1 print(list(range(0, 10, 2)))
```

```
[0, 2, 4, 6, 8]
```

2. range

2.2. ສຳລັບ range

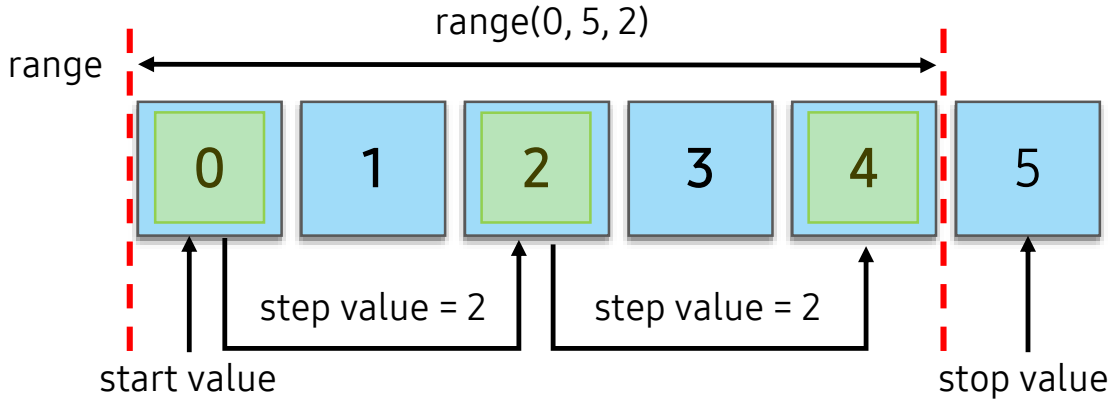
ค่าเริ่มต้น และ ค่าสุดท้ายของ range(5) และ ส่งค่ากับคืนเป็นข้อมูล array



2. range

2.2. ສຳລັບ range

- range(2, 5) ສ້າງ array ຈຳນວນຖ້ວນທີ່ໃຫຍ່ກວ່າ ຫຼື ເທົ່າກັບ 2, ແຕ່ນ້ອຍກວ່າ 5, ດັ່ງນັ້ນ ຈະໄດ້ [2, 3, 4]. range (0, 5, 2) ຈະເພີ່ມຂຶ້ນຄັ້ງລະ 2 ເມື່ອສ້າງຈຳນວນຖ້ວນຫຼາຍກວ່າ ຫຼື ເທົ່າກັບ 0 ແລະ ນ້ອຍກວ່າ 5. ດັ່ງນັ້ນ, ຈຶ່ງສ້າງຈຳນວນຖ້ວນ [0, 2, 4].
- ສຳລັບຄ່າຈຳນວນຖ້ວນບວກ, ຄ່າເລີ່ມຕົ້ນຈະຕ້ອງນ້ອຍກວ່າຄ່າສຸດທ້າຍ, ເຊັ່ນ: range(0, 5, 1). ສຳລັບຄ່າຈຳນວນຖ້ວນລົບ, ຄ່າເລີ່ມຕົ້ນຈະຕ້ອງໃຫຍ່ກວ່າຄ່າສຸດທ້າຍ ເຊັ່ນ: range(-2, -10, -2).



2. range

2.3. แก้ไขปัญหาโดยใช้ for ใน range

สามารถใช้คำสั่ง for loop ดังต่อไปนี้ เพื่อหาผลรวมของจำนวนถ้วนจาก 1 ถึง 10.

```
1 s = 0
2 for i in range(1, 11):
3     s += i
4 print('Sum of numbers from 1 to 10: ', s)
```

```
Sum of numbers from 1 to 10:  55
```

One More Step

ຄວາມສຳຄັນຂອງການເລີ່ມຕົ້ນໃນຄຳສັ່ງ loop, ການດຳເນີນການສະສົມ

Iteration	i value	s value	Iterate or not	Computing process of s (Initial s is 0)
1st	1	1	Iterate	0+**1**
2nd	2	3	Iterate	0+1+**2**
3rd	3	6	Iterate	0+1+2+**3**
4th	4	10	Iterate	0+1+2+3+**4**
5th	5	15	Iterate	0+1+2+3+4+**5**
6th	6	21	Iterate	0+1+2+3+4+5+**6**
7th	7	28	Iterate	0+1+2+3+4+5+6+**7**
8th	8	36	Iterate	0+1+2+3+4+5+6+7+**8**
9th	9	45	Iterate	0+1+2+3+4+5+6+7+8+**9**
10th	10	55	Stop Iterating	0+1+2+3+4+5+6+7+8+9+**10**

- ຖ້າຫາກວ່າຄຳສັ່ງ for ຖືກວົນຊ້ຳໃນໂປຣແກຣມ, ຄ່າ i ແລະ s ຈະມີການປ່ຽນແປງດັ່ງຕໍ່ໄປນີ້
- ຄ່າຂອງ i ປ່ຽນແປງໄປຫຼັງຈາກການວົນຊ້ຳແຕ່ລະຄັ້ງ ແລະ ດ້ວຍເຫດຜົນນີ້ຄ່າຂອງ s ຈຶ່ງເພີ່ມຂຶ້ນດ້ວຍ.
- "i" ເພີ່ມຂຶ້ນເທື່ອລະ 1, ກາຍເປັນ 1, 2, 3 ... ເນື່ອງຈາກ s = 0 ເປັນສະຖານະເລີ່ມຕົ້ນ, 0+1 ຖືກກຳນົດໃຫ້ເປັນຄ່າຂອງ s ໃນລະຫວ່າງການວົນຊ້ຳທຳອິດ, ແຕ່ເນື່ອງຈາກວ່າ code ດຳເນີນການ s+i ທຸກໆຄັ້ງ, ມັນຈຶ່ງຄິດໄລ່ 0+1, 0+1+2, 0+1+2+3, ...
- ຫຼັງຈາກການວົນຊ້ຳຄັ້ງທີ 10, s ຈະມີຄ່າເປັນ 55 ແລະ ການວົນຊ້ຳຈະຖືກຢຸດລົງ.

2. range

2.4 ການທັບຊ້ອນກັນຂອງຟັງຊັນ sum() ແລະ ຜົນລວມຂອງຊື່ຕົວປ່ຽນ

- ຟັງຊັນ sum() ເປັນຟັງຊັນພາຍໃນຂອງ Python ເພື່ອຄິດໄລ່ຜົນລວມຂອງອົງປະກອບໃນລາຍການ.
- ບາງຄັ້ງຕົວປ່ຽນຜົນລວມພາຍໃນ code ອາດສັບສົນກັບຜົນລວມຂອງຟັງຊັນ sum() ພາຍໃນພາສາ Python. ນັ້ນແມ່ນຂໍ້ຜິດພາດຈະເກີດຂຶ້ນຖ້າຕົວປ່ຽນ sum = 0 ແລະ sum(numbers) ຖືກເອີ້ນໃຊ້ພາຍໃນໂມດູນດຽວກັນ.
- code ຕໍ່ໄປນີ້ຈະພິມ 55 ໂດຍການເພີ່ມຄ່າທັງໝົດຂອງ 1, 2, 3, 4, 5, 6, 7, 8, 9, ແລະ 10 ທີ່ສ້າງຂຶ້ນໂດຍ range(1, 11) ໂດຍໃຊ້ຟັງຊັນ sum().

```
1 print(sum(range(1,11)))
```

```
55
```

- ฟังชัน sum() คิดไล่ผินลอมของอิงปะกอบในลายกาม. ดั่งนั้นจึ่งพิมผินลอมของตัวเลกจาก 1 ทา 10, ออกมาเป้น 55.

Overlapping use of sum() function and variable name

เมื่อจากฟังชัน sum ได้ถูกใช้เป็นชื่อของตัวปรับแล้ว, มันบ่สามาดเอิ้นใช้ได้ในพายหลัง.

```
1 sum = 0
2 for i in range(1, 11):
3     sum += i
4 print('The sum of numbers from 1 to 10: ', sum)
5 print(sum(range(1,11)))
```

```
The sum of numbers from 1 to 10 :  55
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-34-991f98d67989> in <module>
      3     sum += i
      4 print('The sum of numbers from 1 to 10: ', sum)
----> 5 print(sum(range(1,11)))

TypeError: 'int' object is not callable
```

- ถ้านำใช้ผินลอมตัวปรับพายใน code ดั่งที่สะแดงข้างเหิง, มันอาดจะสับสนกับผินลอมของฟังชัน sum() พายในของ Python.
- ข้ผิดพาดจะเกิดขึ้นเมื่อตัวปรับที่เอิ้นว่า sum = 0 และ ฟังชันที่เอิ้นว่า sum(numbers) ถูกเอิ้นใช้พายในโมดูนดียวกัน.
- ดั่งนั้น, ขำแนะนำให้ใช้ตัว s หื total เป็นชื่อของตัวปรับ, เพื่อบ่ให้มันทับซ้อนกันกับฟังชัน sum().

2. range

2.5. ภาษาคำนวน factorial

n factorial แม่นค่าที่ถืกคูนด้วยต๊อเลกทำมะชาดทั้งหมดจาก 1 ทา n เมื่อ n เป็นต๊อเลกทำมะชาดที่กำนิดตามใจ.

$$n! = \prod_{k=1}^{n} k = n \cdot (n-1) \cdot (n-2) \cdot \cdots \cdot 3 \cdot 2 \cdot 1$$

```
1 n = int(input('Enter a number: '))
2 fact = 1
3 for i in range(1, n+1):
4     fact = fact * i
5 print(str(n)+'!  = ', fact)
```

```
Enter a number :
5! = 120
```

One More Step

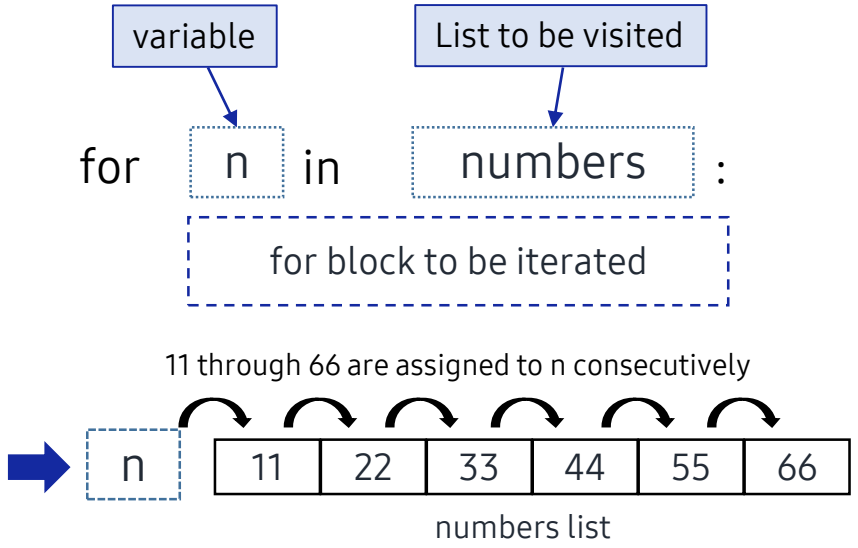
I. Python ແລະ ຂໍ້ຈຳກັດໃນ expressions ຈຳນວນຖ້ວນ

- ຖ້າເຮົາປ່ຽນຄ່າຂອງ n ໃນ code ຂ້າງເທິງເປັນ 50, ມັນຈະພິມຄ່າຕໍ່ໄປນີ້ອອກມາມະຫາສານ.
- 50! = 30414093201713378043612608166064768844377641568960512000000000000.
- ໃນພາສາໂປຣແກຣມເຊັ່ນ: C ແລະ Java, ບໍ່ສາມາດສະແດງຈຳນວນຖ້ວນທີ່ມີຂະໜາດໃຫຍ່ກວ່າຂະໜາດທີ່ກຳນົດໄດ້ ເພາະວ່າຂະໜາດຂອງຈຳນວນຖ້ວນຖືກກຳນົດເປັນ 4 byte. ຕົວຢ່າງ, ຈຳນວນຖ້ວນ long-type ຂອງ Java ຕັ້ງແຕ່ -9223372036854775808 ຫາ 9223372036854775807. ເຖິງຢ່າງໃດກໍຕາມໃນ Python ບໍ່ມີຂໍ້ຈຳກັດໃນສຳນວນຂອງຈຳນວນຖ້ວນ. ນີ້ເປັນຂໍ້ດີອີກອັນໜຶ່ງຂອງ Python.

3. ການວົນຊ້ຳແບບອັດຖຸຕາມລຳດັບ

3.1. ລຳດັບຂອງ Object

- ປະກາດຕົວປ່ຽນໃໝ່ n ທີ່ຈະຖືກກຳນົດລະຫວ່າງ for loop ແລະ in. ຫຼັງຈາກນັ້ນໃສ່ລາຍການຕົວເລກທີ່ມີອົງປະກອບ [11, 22, 33, 44, 55, 66] ຫຼັງຈາກ 'in'
- code ຂ້າງລຸ່ມນີ້ດຳເນີນການ ໂດຍໄປທີ່ຄ່າພາຍໃນລາຍການຕາມລຳດັບ, ຫຼັງຈາກນັ້ນຕົວປ່ຽນ n ຈະຖືກກຳນົດຕາມລຳດັບເປັນ 11, 22, 33, 44, 55, ແລະ 66.



3. ການວົນຊ້ຳແບບອັດຖຸຕາມລຳດັບ

3.1. ລຳດັບຂອງ Object

ປະກາດຕົວປ່ຽນໃໝ່ n ທີ່ຈະກຳນົດລະຫວ່າງ for loop ແລະ in, ໃສ່ລາຍການຕົວເລກຂອງອົງປະກອບ [11, 22, 33, 44, 55, 66] ຫຼັງຈາກ "in". ໃນຕອນນີ້ໃຫ້ໃຊ້ຄຳສັ່ງພິມ n ອອກມາ.

```
1 numbers = [11, 22, 33, 44, 55, 66]
2 for n in numbers :
3     print('n = ', n)  # different values are printed each time
```

```
n =  11
n =  22
n =  33
n =  44
n =  55
n =  66
```

3. ການວົນຊ້ຳແບບອັດຖຸຕາມລຳດັບ

3.2. ຕົວຢ່າງ code ການບວກສະສົມ

- ເພີ່ມລາຍການຈຳນວນຖ້ວນຂອງອົງປະກອບ list. ເກັບຜົນຮັບຂອງການບວກສະສົມ ໃນ s ແລະ ພິມ ອອກມາ.

```
1 numbers = [10, 20, 30, 40, 50]
2 s = 0
3 for n in numbers:
4     s = s + n
5 print('Sum of list items : ', s)
```

```
Sum of list items : 150
```

- ນີ້ແມ່ນໂປຣແກຣມເພື່ອຊອກຫາຜົນລວມຂອງຄ່າໃນລາຍການທີ່ເປັນຈຳນວນຖ້ວນໃນ list ໂດຍນຳ ໃຊ້ຟັງຊັນການບວກສະສົມ.

3. ການວົນຊ້ຳແບບອັດຖຸຕາມລຳດັບ

3.2. ຕົວຢ່າງ code ການບວກສະສົມ

ຜົນລວມສະສົມຂອງຄ່າຈຳນວນຖ້ວນໃນ list ສາມາດໃຊ້ຟັງຊັນ sum() ພຽງຢ່າງໄດ້.

```
1 numbers = [10, 20, 30, 40, 50]
2 print('Sum of list items : ', sum(numbers))
```

```
Sum of list items : 150
```

- ຜົນລວມຂອງອົງປະກອບ list ສາມາດຄິດໄລ່ໄດ້ງ່າຍ, ໂດຍໃຊ້ຟັງຊັນທີ່ສ້າງຂຶ້ນໃນຕົວ sum() ໂດຍບໍ່ຕ້ອງໃຊ້ຄຳສັ່ງ for.

3. ການວົນຊ້ຳແບບອັດຖຸຕາມລຳດັບ

3.3. ການແປງຂໍ້ມູນປະເພດ string ເປັນຂໍ້ມູນປະເພດ list.

ສາມາດແປງຂໍ້ມູນປະເພດ string ເປັນຂໍ້ມູນປະເພດ lists ໂດຍໃຊ້ຟັງຊັນ list

```
1 st = 'Hello'
2 print(list(st))
```

```
['H', 'e', 'l', 'l', 'o']
```

- ຂໍ້ມູນປະເພດ string ‘Hello’ ຈະກາຍເປັນລາຍຊື່ຕົວອັກສອນແຕ່ລະຕົວ.

3. ການວົນຊ້ຳແບບອັດຖຸຕາມລຳດັບ

3.4. ວິທີການນຳໃຊ້ຂໍ້ມູນປະເພດ string ໃນ for loops

```
1 for ch in 'Hello':
2     print(ch, end = ' ')
```

```
H e l l o
```

- ສາມາດແປງປະເພດຂໍ້ມູນ string 'Hello' ເປັນລາຍການຂອງອົງປະກອບຕົວອັກສອນໄດ້ ໂດຍການນຳໃຊ້ຟັງຊັນ list.
- ໃຫ້ວາງ string ໄວ້ຫຼັງຈາກ for ໃນສຳນວນ. ການດຳເນີນການນີ້ຈະແຍກ string ອອກເປັນຕົວອັກສອນແຕ່ລະຕົວ, ເຮັດໃຫ້ການວົນຊ້ຳສາມາດໄປຫາຕົວອັກສອນແຕ່ລະຕົວໄດ້.
- ເມື່ອແຊກ end =‘ ’ argument ພາຍໃນຟັງຊັນ print. ເປັນການສັ່ງໃຫ້ພິມຂໍ້ມູນໃນແຖວດຽວກັນ ໂດຍບໍ່ມີການລົງແຖວໃໝ່ຫຼັງຈາກພິມ ch.

One More Step

- ຕົວເລືອກຕ່າງໆຂອງຟັງຊັນ print()
- ຈົ່ງປະຕິບັດດັ່ງຕໍ່ໄປນີ້ເພື່ອໃສ່ຕົວອັກສອນທີ່ບໍ່ແມ່ນຊ່ອງຫວ່າງລະຫວ່າງຄ່າ sep ມາຈາກຄຳວ່າ separator.

```
1 print('s','e','p', sep='-')
```

```
s-e-p
```

- ຄຳຫຼັກຂອງການສິ້ນສຸດ argument ກຳນົດ string ທີ່ຈະພິມອອກຫຼັງຈາກຄ່າ output. ໃນ code ຂ້າງລຸ່ມນີ້, ພິມຊ່ອງຫວ່າງໜຶ່ງຊ່ອງ, ໂດຍໃຊ້ "end=" ຫຼັງຈາກ "My name is". ຖ້າພິມ string ຢ່າງອື່ນໃສ່, string ນັ້ນຈະຖືກພິມແທນຊ່ອງຫວ່າງ.

```
1 print("My name is", end=" ")
2 print("David")
```

```
My name is David
```

```
1 print("My name is", end=" : ")
2 print("David")
```

```
My name is : David
```

Paper coding

- ✓ ພະຍາຍາມເຂົ້າໃຈແນວຄວາມຄິດພື້ນຖານຢ່າງຖີ່ຖ້ວນກ່ອນທີ່ຈະກ້າວໄປສູ່ຂັ້ນຕອນຕໍ່ໄປ.
- ✓ ການທີ່ບໍ່ເຂົ້າໃຈແນວຄວາມຄິດພື້ນຖານຈະເພີ່ມພາລະຂໃນການຮຽນຮູ້ຫຼັກສູດນີ້, ເຊິ່ງອາດຈະເຮັດໃຫ້ລົ້ມເຫຼວໃນຫຼັກສູດ.
- ✓ ໃນຕອນນີ້ອາດຈະເປັນເລື່ອງຍາກ, ແຕ່ເພື່ອໃຫ້ສຳເລັດໃນຫຼັກສູດນີ້, ຂໍແນະນຳໃຫ້ເຂົ້າໃຈແນວຄວາມຄິດຢ່າງແທ້ຈິງ ແລະ ກ້າວໄປສູ່ຂັ້ນຕອນຕໍ່ໄປ.

Q1. ประกาศลายชื่อ bts = ['V', 'J-Hope', 'RM', 'Jungkook', 'Jin', 'Jimin', 'Suga']. ຫຼັງຈາກນັ້ນ, ຂຽນ code ທີ່ພິມລາຍການທັງໝົດໃນບັນຊີລາຍຊື່ນີ້ ໂດຍໃຊ້ຄຳສັ່ງ for.

Example Output	V J-Hope RM Jungkook Jin Jimin Suga
Time	5min

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

Q2. ใช้ฟาบอกสะสึมเพื่อคำนอบ และ พิมผืนลอมของๆจำนอมถ้อมแต่ 1 ทา 100.
(คำแนะนำ: เธัดให้ค่าที่พิมของฟังชัน range มีค่าตั้งแต่ 1 ทา 100.)

Example Output	`Sum of integers from 1 to 100 : 5050`
Time	5min

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

Q3. ใช้ค่า step ของฟังชัน range เพื่อชอกหาผินบอกของเลกคู่แต่ 1 ทา 100.
(คำแนะนำ: กำนิดค่าเลิ่มต้นของฟังชัน range เป้น 0 และ ค่า step เป้นสอง.)

Example Output	`Sum of even numbers from 1 to 100 : 2550`
Time	5min

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

Q4. ใช้ค่า step ของฟังชัน rang เพื่อชอกหาผินบอกของตัวเลกคีกจาก 1 หา 100.
((คำแนะนำ: กำนิดค่าเลิ่มต้นของฟังชัน rang เป็น 1 และ ค่า step เป็น 2)

Example Output	`Sum of odd numbers from 1 to 100 : 2500`
Time	5min

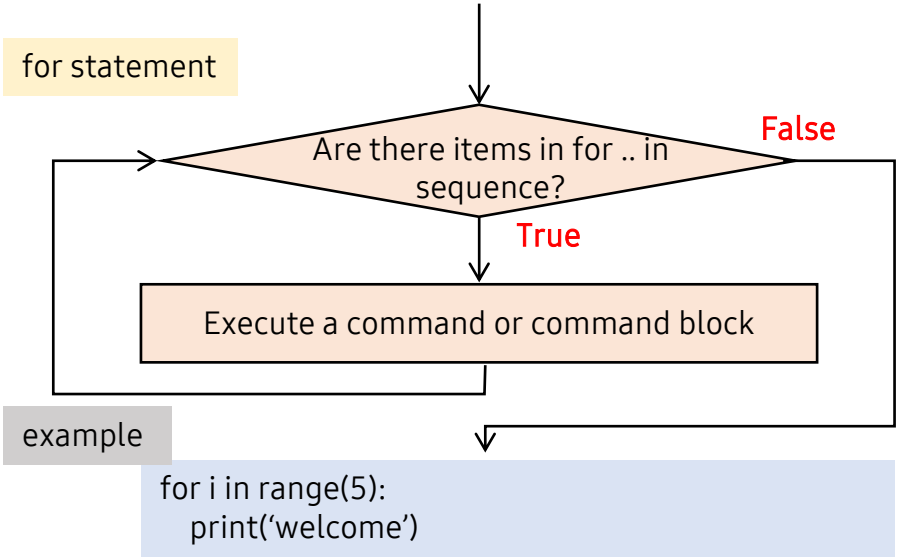
Write the entire code and the expected output results in the note.

Let's code

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.1. ໂຄງສ້າງການດຳເນີນການຂອງຄຳສັ່ງ for

ໃນຮູບຂ້າງລຸ່ມນີ້, range(5) ແມ່ນຟັງຊັນທີ່ສ້າງລຳດັບຕົວເລກເປັນ 0, 1, 2, 3 ແລະ 4 ແບບ ອັດຕະໂນມັດ.



- ດັ່ງນັ້ນ, ປະໂຫຍກນີ້ຈະຖືກວົນຊ້ຳຈົນກ່ວາການສົ່ງຕົວເລກກັບຄືນຈາກ 0 ຫາ 4 ສຳເລັດ.

One More Step

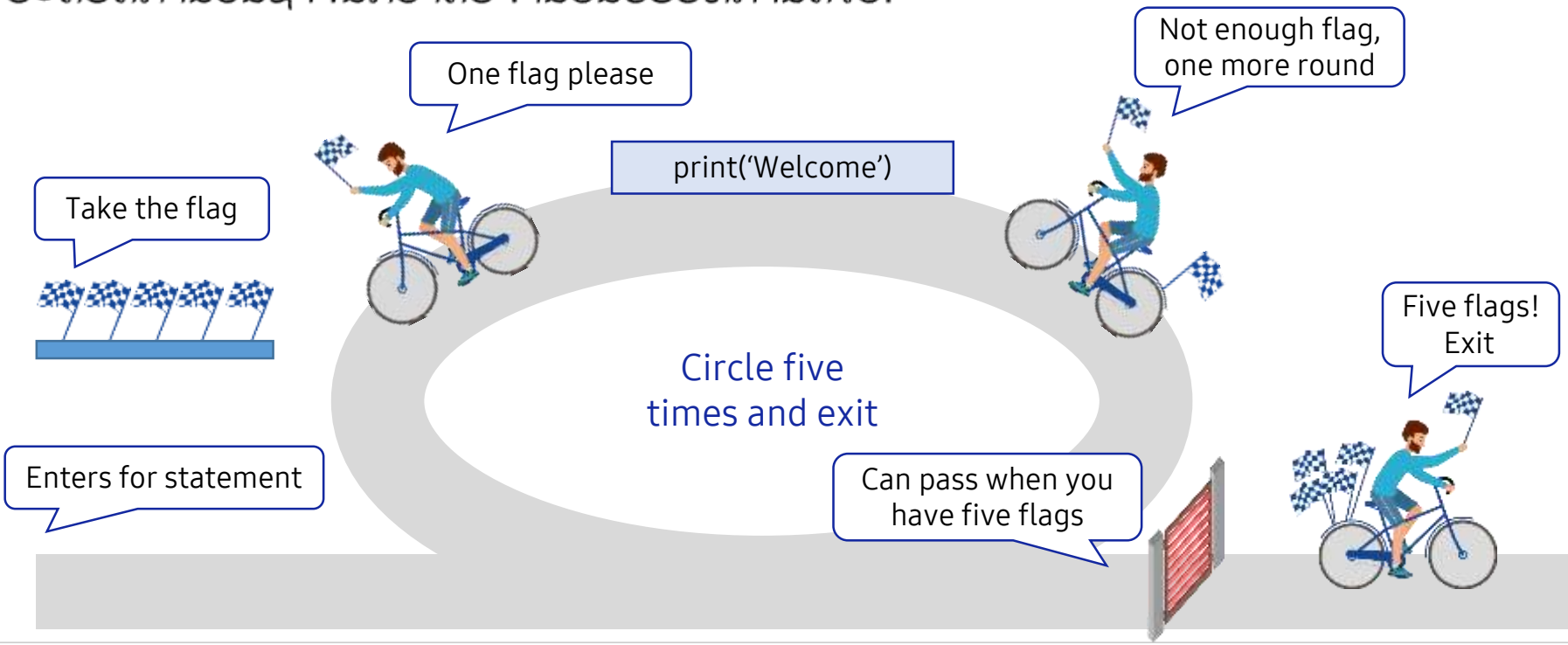
Loop

- ໃນການຂຽນໂປຣແກຣມ, ການວົນຊ้ำมักจะเอ้ิน่า loop, เช่ิงมีคอามหายอ่าเป้น วิงมินค้ายคี บ้อง. เน่ืองจากเม่ือโปรแกรมวินช้ำ, โปรแกรมจะย้อมวินกับไป ขั้นตอนก่อนหน้า, ค้ายคีกับวิงมิน. การวินช้ำกันโดยบ่ออกจากวิงนี้เอ้ินอ่า loop ที่บ่มีอันสิ้นสุด (infinite loop).

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.2. ນຳໃຊ້ for loop ເພື່ອການວົນຊ້ຳຕາມຈຳນວນຄັ້ງທີ່ຄວບຄຸມ

ໃນ Python, ສຳລັບ for loop ເປັນວິທີທີ່ສະດວກທີ່ສຸດສຳລັບການວົນຊ້ຳຂອງຕົວເລກທີ່ຄວບຄຸມ ດ້ວຍ for loop, ເຮັດໃຫ້ງ່າຍຕໍ່ການກຳນົດຈຳນວນຮອບຂອງການວົນຊ້ຳ ແລະ ປ່ອຍໃຫ້ code ທີ່ ປະຕິບັດການວົນຊ້ຳຈົນກ່ວາຄົບຈຳນວນຮອບທີ່ກຳນົດໄວ້.



General flow of loop execution using for loop statements.

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.2. ນຳໃຊ້ for loop ເພື່ອການວົນຊ້ຳຕາມຈຳນວນຄັ້ງທີ່ຄວບຄຸມ

- ເມື່ອໃຊ້ for loops, ໃຫ້ປະຕິບັດການຮ່ວມກັບການນຳໃຊ້ lists ດັ່ງຕໍ່ໄປນີ້. ສາມາດວົນຊ້ຳດ້ວຍ code ຕໍ່ໄປນີ້. ຂໍ້ມູນທີ່ຢູ່ໃນວົງຂໍ [...] ແມ່ນຂໍ້ມູນປະເພດ list.
- ເຊິ່ງຈະຖືກກ່າວເຖິງໃນລາຍລະອຽດເພີ່ມເຕີມໃນ unit ຕໍ່ໄປ. ສາມາດເຂົ້າໃຈໄດ້ວ່າມັນເປັນກະຕ່າສັ່ງຊື້ ສິນຄ້າທີ່ມີສິນຄ້າຫຼາຍລາຍການ. / ຄິດວ່າຈຳນວນຖ້ວນ 1, 2, 3, 4 ແລະ 5 ບັນຈຸຢູ່ໃນລາຍການ [1, 2, 3, 4, 5].

```
1 for i in [1, 2, 3, 4, 5]:    # There must be : at the end
2     print("Welcome.")            # Must indent
```

```
Welcome.
Welcome.
Welcome.
Welcome.
Welcome.
```

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.3. ການຫຍໍ້ໜ້າສຳລັບຄຳສັ່ງ for

- ຂໍ້ຄວາມທີ່ຈະຖືກປະຕິບັດຊ້ຳຈະຕ້ອງຖືກຫຍໍ້ໜ້າເຂົ້າ.
- ຄຳສັ່ງທີ່ຈະຖືກປະຕິບັດແມ່ນ Block ພາຍໃນຄຳສັ່ງ for. ດັ່ງສະແດງ code ທີ່ສົ່ງຜົນໄດ້ຮັບດັ່ງຕໍ່ໄປນີ້.

```
1 for i in [1, 2, 3, 4, 5]:    # There must be : at the end
2     print("Welcome.")        # Must indent
```

```
Welcome.
Welcome.
Welcome.
Welcome.
Welcome.
```

- ໃນການວົນຊ້ຳຄັ້ງທຳອິດ, ຄ່າຂອງຕົວປ່ຽນ i ຈະເປັນ 1, ຕົວເລກທຳອິດຂອງລາຍການ ແລະ ປະໂຫຍກ print(...) ຈະຖືກປະຕິບັດ.
- ໃນການວົນຊ້ຳຄັ້ງທີສອງ, ຄ່າຂອງຕົວປ່ຽນ i ຈະກາຍເປັນ 2, ຕົວເລກທີສອງໃນລາຍການ ແລະ ປະໂຫຍກ print(...) ຈະຖືກປະຕິບັດ.
- ໃນການວົນຊ້ຳຄັ້ງທີສາມ, ຄ່າຂອງຕົວປ່ຽນ i ຈະກາຍເປັນ 3, ຕົວເລກທີສາມໃນລາຍການ ແລະ ປະໂຫຍກ print(...) ຈະຖືກປະຕິບັດ.

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.3. ການຫຍໍ້ໜ້າສຳລັບຄຳສັ່ງ for

- ຂໍ້ຄວາມທີ່ຈະຖືກປະຕິບັດຊ້ຳຈະຕ້ອງຖືກຫຍໍ້ໜ້າເຂົ້າ.
- ຄຳສັ່ງທີ່ຈະຖືກປະຕິບັດແມ່ນບ Block ພາຍໃນຄຳສັ່ງ for. ດັ່ງສະແດງ code ທີ່ສົ່ງຜົນໄດ້ຮັບດັ່ງຕໍ່ໄປນີ້.

```
1 for i in [1, 2, 3, 4, 5]:    # There must be : at the end
2     print("Welcome.")        # Must indent
```

```
Welcome.
Welcome.
Welcome.
Welcome.
Welcome.
```

- ໃນການວົນຊ້ຳຄັ້ງທີສີ່, ຄ່າຂອງຕົວປ່ຽນ i ຈະກາຍເປັນ 4, ຕົວເລກທີສີ່ໃນລາຍການ ແລະ ປະໂຫຍກ print(...) ຈະຖືກດຳເນີນການ.
- ໃນການວົນຊ້ຳຄັ້ງທີຫ້າ, ຄ່າຂອງຕົວປ່ຽນ i ຈະກາຍເປັນ 5, ຕົວເລກທີຫ້າໃນລາຍການ ແລະ ປະໂຫຍກ print(...) ຈະຖືກດຳເນີນການ.

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.4. ລຳດັບຂອງ object ທີ່ຕາມມາສຳລັບ for-in

ลำดับ object เช่น list หรือ string ก็อาจจะติดตามมาก□า for-in.

- object ที่สามาถมิได้หลายลายกามเช่น: list หรือ string เอิ้นอ่าลำดับ object
- ใน Slide ที่ผ่ามมา, บ่มีกามนำใช้ถ่าของตัวปรฺม i เลย.
- เอลามื้, กำมิดลายกามของลำดับ object ให้กับตัวปรฺม i พายในถำสั่งวนซ้ำ. จากมั้นใช้ฟังชัน print เพื่อพิมถ่าของ i.

```
1 for i in [1, 2, 3, 4, 5]: # place the list after in and : after the list
2     print("i =", i) # print values of i
```

```
i = 1
i = 2
i = 3
i = 4
i = 5
```

1. โครงสร้างของคำสั่งวนซ้ำ

1.4. ลำดับของ object ที่ตามมาสำหรับ for-in

ลำสั่งนี้ยังสามาดนำใช้กับ strings ได้เช่นกัน.

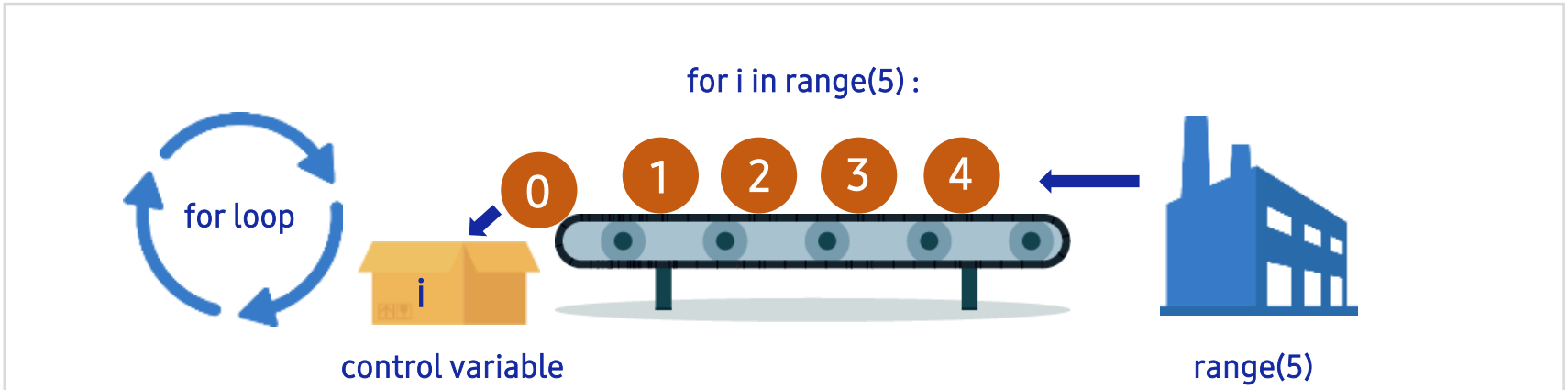
```
1 for i in "Hello":      # : is necessary at the end
2     print("i =", i)    # print i values
```

```
i = H
i = e
i = l
i = l
i = o
```

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.5. ຟັງຊັນ Range ທີ່ມາພ້ອມກັບຄຳສັ່ງ for.

- ໃນຕົວຢ່າງທີ່ຜ່ານມາ, ຈຳນວນຖ້ວນຖືກເກັບໄວ້ໃນ list ແລະ ວົນຊ້ຳເທື່ອລະລາຍການ. ເຖິງຢ່າງໃດກໍຕາມ, ກົນໄກນີ້ແມ່ນເປັນໄປບໍ່ໄດ້ຖ້າຫາກວ່າຈຳນວນຂອງການວົນຊ້ຳແມ່ນ 1,000 ຄັ້ງ.
- ສາມາດນຳໃຊ້ຟັງຊັນ range ສຳລັບ code ທີ່ມີການວົນຊ້ຳຈຳນວນຫຼາຍ.
- ຖ້າຫາກວ່າເຮົາສົ່ງຈຳນວນຂອງການວົນຊ້ຳໃຫ້ກັບຟັງຊັນ range ຟັງຊັນນັ້ນຈະສ້າງຈຳນວນຖ້ວນຂຶ້ນມາແບບອັດຕະໂນມັດ.
- range (5) ຂອງ code ຂ້າງລຸ່ມນີ້ຈະສ້າງລຳດັບຂອງ 0, 1, 2, 3 ແລະ 4 ແລະ ສົ່ງໄປເກັບໄວ້ໃນ i.



1. โຄງสร้างของคำสั่งวนซ้ำ

1.5. ฟังชัน Range ที่มาพ้อมกับคำสั่ง for.

สามาถเขียน code ที่พิม "Welcome to" และ "Python Corporation!" ห้าถ้ังกับคำสั่ง for ดังต่ำไปนี้.
(2) แม่น Block ทามวนซ้ำ.

```
1 for i in range(5):                # (1)
2     print('Welcome to')           # (2)
3     print("Python corporation!")  # (2)
```

```
Welcome to
Python corporation!
Welcome to
Python corporation!
Welcome to
Python corporation!
Welcome to
Python corporation!
Welcome to
Python corporation!
```

1. โครงสร้างของคำสั่งวนซ้ำ

1.6. คุณลักษณะของฟังชัน range

- ฟังชัน range(5) จะสิ่งลำดับของ 0, 1, 2, 3 และ 4. ถึงออกมาจากการวนซ้ำแต่ละครั้ง, มันจะกำหนดค่าเหล่านี้ให้กับตัวแปร i. นั่นคือ i กายเป็น 0 ในการวนซ้ำครั้งที่หนึ่ง และ 1 ในการวนซ้ำครั้งที่สอง. ในการวนซ้ำครั้งสุดท้าย i จะกายเป็น 4.
- สามารถเช็คจำนวนถ้วยที่สร้างขึ้น โดยฟังชัน range ถ้าหากว่านำใช้ฟังชัน list กับ ฟังชัน range. ด้วยวิธีนี้, ฟังชัน range จะสร้างค่าที่ต่อเนื่องกัน.

```
1 print(list(range(5)))
```

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

One More Step

ຄ່າທີ່ສົ່ງຄືນໂດຍຟັງຊັນ range ກໍຈະເປັນປະເພດ range.

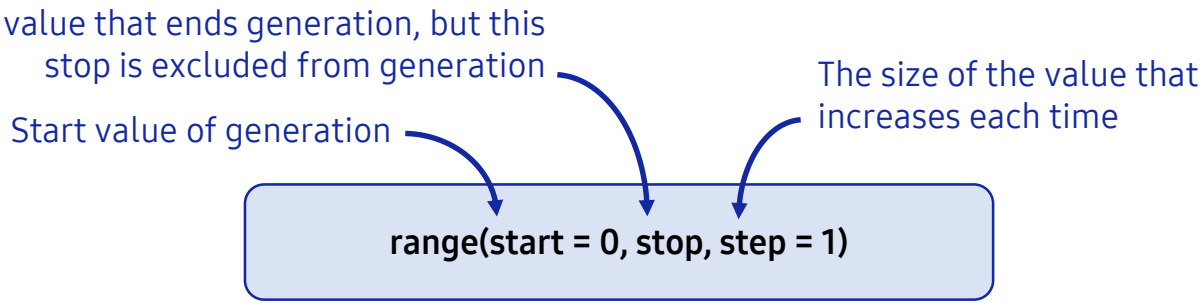
- ຟັງຊັນ range ຂອງ Python ຈະຄືນຄ່າຂໍ້ມູນເປັນ range-type
- ຂໍ້ມູນປະເພດ range-type ຈະສ້າງ ແລະ ສົ່ງຄ່າຄືນອອກມາຕິດຕໍ່ກັນ, ສຳລັບການເອີ້ນໃຊ້ແຕ່ລະຄັ້ງ. ດັ່ງນັ້ນ, ມັນຈຶ່ງໄດ້ໃຊ້ຮ່ວມກັບຄຳສັ່ງ for ເປັນຫຼັກ.

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.6. ຄຸນລັກສະນະຂອງຟັງຊັນ range

ฟังชัน range สามาถกำนิดค่า step ได้.

- ฟังชัน range สามาถสร้างลำดับจำนอนถ้อนที่ซ้ากซ้ายโดยการใช้ argument ซ้ายถือ และ รูบแบบทั่อไปของฟังชันมีดังต่ำไปนี้
- ถ้าซากว่าเรำเอิ้นใช้ range รูบแบบ range(start, stop, step), จำนอนถ้อนนับตั้งแต่เลิ่มต้นซา (stop-1) จะถูกสร้างขึ้น, โดยมีไลยะห่างเท่ำกับค่าของ step.
- For loop สำลับการวนซ้ำซ้ายรอบตามจำนอนของลำดับนี้.
- ในที่นี้, start และ step อาถจะถูกยกเว้น (บ่ต้องใส่ค่า), ในกำละนี start แม่น 0 และ step แม่น 1.
- เถิงย่างใดกำตาม, การวนซ้ำได้ถูกปะติบัดการสะเพาะ เมื่อมีการกำนิดค่ายุดเท่ำนั้น.



ถ้าบ่ได้ละบุค่า start และ step จะถือว่าเป้น 0 และ 1 ตามลำดับ ค่าที่ต้องละบุถือ stop.

ค่า ends ของการส้าง, แต่การ stop นี้บ่ได้ลอมยู่ในการส้าง, ค่าเลิ่มต้นของการส้าง, ขะหมาดของค่าที่เพิ่มขึ้นในแต่ละถั้ง, ถ้าบ่ได้ละบุค่า start และ step จะถือว่าเป้น 0 และ 1 ตามลำดับ ค่าที่ต้องละบุถือ stop.

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.7. ຕົວຢ່າງຂອງຟັງຊັນ range

- range (0, 5, 1) ຈະສົ່ງຄ່າຈຳນວນຖ້ວນ 0, 1, 2, 3 ແລະ 4. Range (5) ແມ່ນຄືກັນກັບ range (0, 5, 1) ເນື່ອງຈາກ Start ແລະ step ຖືກຂ້າມໄປ.
- ຖ້າຕ້ອງການວົນຊ້ຳຈາກ 1 (ບໍ່ແມ່ນ 0) ຫາ 5, ໃຊ້ range (1, 6, 1). ພິມຄ່າຂອງຕົວປ່ຽນ i ໃນຂະນະທີ່ວົນຊ້ຳ.

```
1 for i in range(1, 6, 1):
2     print(i, end=" ") # if you assign end = " ", it enumerates with spaces in between instead of changing li
```

```
1 2 3 4 5
```

- ຫາກຕ້ອງການພິມຕົວເລກຄີກລະຫວ່າງ 1 ຫາ 10, ໃຫ້ກຳນົດ 2 ໃຫ້ກັບຄ່າ step.

```
1 for i in range(1, 10, 2):
2     print(i, end=" ")
```

```
1 3 5 7 9
```

1. ໂຄງສ້າງຂອງຄຳສັ່ງວົນຊ້ຳ

1.7. ຕົວຢ່າງຂອງຟັງຊັນ range

ຖ້າຫາກຕ້ອງການພິມຈຳນວນຈາກ 1 ຫາ 10 ຕາມລຳດັບຈາກຫຼາຍໄປຫາໜ້ອຍ, ໃຫ້ໃຊ້ range(10, 0, -1). ພິມຄ່າຂອງຕົວປ່ຽນ i ໃນຂະນະທີ່ກຳລັງວົນຊ້ຳ.

```
1 for i in range(10, 0, -1):
2     print(i, end=" ")
```

```
10 9 8 7 6 5 4 3 2 1
```

One More Step

range(n) ແມ່ນຄືກັນກັບ range(0, n, 1).

- ເຮົາມັກຈະຄິດວ່າຟັງຊັນ range(10) ຈະສ້າງຈຳນວນຖ້ວນຈາກ 1 ຫາ 10. ຈຳນວນຂອງການວົນຊ້ຳແມ່ນ 10. ເຖິງແນວໃດກໍຕາມ, ຈຳນວນຖ້ວນທີ່ສ້າງໄດ້ຄື 0 ຫາ 9. ນີ້ແມ່ນການຖືກຖຽງກັນມາດົນນານໃນການຄຳນວນ.
- ເລີ່ມຕົ້ນຈາກ 0 ໃນປັດຈຸບັນໄດ້ກາຍເປັນ trend ໄປແລ້ວ. ເຖິງແນວໃດກໍຕາມ, range(10) ຈະວົນຊ້ຳ 10 ຄັ້ງ ແລະ ຈະສ້າງຈຳນວນຖ້ວນແຕ່ 0 ເຖິງ 9. ຖ້າຕ້ອງການພິມຈຳນວນຖ້ວນຕັ້ງແຕ່ 1 ຫາ 10, ກໍໃຊ້ຄຳສັ່ງ range(1, 11).

2. ຮູບແບບຂອງຟັງຊັນ

2.1. ຕົວຢ່າງ code ຮູບແບບຂອງຟັງຊັນ

{} ຮູບແບບຟັງຊັນແມ່ນໃຊ້ເພື່ອຈັດຮູບແບບ strings ແລະ ຕົວເລກໃນຮູບແບບທີ່ກຳໜົດ.

```
# formatting integer value
print('{:5d}'.format(100))
```

```
100
```

```
# formatting integer value
print('{:5d}'.format(10))
```

```
10
```

```
# formatting real number value
print('{:6.3f}'.format(3.14))
```

```
3.140
```

```
# formatting real number value
print('{:6.2f}'.format(3.14))
```

```
3.14
```

- '{}'.format(100) ການເອີ້ນໃຊ້ຟັງຊັນການຈັດຮູບແບບທີ່ພິມ 100.
- {:5d}'.format(100), {:5d}'.format(10) ສ້າງ 5 ຊ່ອງຫວ່າງເພື່ອພິມ 100 ແລະ 10.
- '{:6.2f}'.format(3.14), ສ້າງ 6 ຊ່ອງຫວ່າງ ແລະ ພິມ 2 ຕົວເລກຢູ່ຫຼັງຈຸດທົດສະນິຍົມເພື່ອພິມຕົວເລກ 3.14.

Pair programming

Pair programming

ແນວທາງ, ກົນໄກ ແລະ ແຜນການສຸກເສີນ

ການກະກຽມຈັບຄູ່ຂຽນໂປຣແກຣມກ່ຽວຂ້ອງກັບການກຳໜົດແນວທາງແນະນຳ ແລະ ກົນໄກເພື່ອຊ່ວຍໃຫ້ນັກຮຽນຈັບຄູ່ຢ່າງຖືກຕ້ອງ ແລະ ໃຫ້ເຂົາເຈົ້າເປັນຄູ່. ຕົວຢ່າງ, ນັກຮຽນຄວນປ່ຽນ "driving the mouse." ການກະກຽມທີ່ມີປະສິດຕິພາບຈຳເປັນຕ້ອງມີແຜນການສຸກເສີນໃນ ກໍລະນີທີ່ຄູ່ຮ່ວມງານຄົນໃດໜຶ່ງບໍ່ຢູ່ ຫຼື ຕັດສິນໃຈທີ່ຈະບໍ່ເຂົ້າຮ່ວມດ້ວຍເຫດຜົນໃດກໍຕາມ. ໃນກໍລະນີເຫຼົ່ານີ້, ມັນເປັນສິ່ງສຳຄັນທີ່ຈະຕ້ອງໄດ້ລະບຸໃຫ້ ຊັດເຈນວ່າ ນັກຮຽນທີ່ມີຄວາມກະຕືລືລົ້ນຈະບໍ່ຖືກລົງໂທດ ເນື່ອງຈາກການຈັບຄູ່ບໍ່ໄດ້ຮັບຜົນດີ.

ການຈັບຄູ່ຄວາມສາມາດທີ່ຄ້າຍຄືກັນ, ບໍ່ຈຳເປັນຕ້ອງເທົ່າທຽມກັນ ໃນຖານະຄູ່ຮ່ວມງານ

ການຂຽນໂປຣແກຣມຈັບຄູ່ຈະມີປະສິດທິພາບເມື່ອນັກຮຽນທີ່ມີຄວາມສາມາດຄ້າຍຄືກັນ, ແຕ່ບໍ່ຈຳເປັນຕ້ອງມີຄວາມສາມາດເທົ່າທຽມກັນຈຶ່ງຈັບ ຄູ່ກັນ, ການຈັບຄູ່ນັກຮຽນທີ່ບໍ່ກົງກັນມັກຈະເຮັດໃຫ້ການມີສ່ວນຮ່ວມທີ່ບໍ່ສົມດຸນກັນ. ຄູສອນຕ້ອງເນັ້ນໜັກວ່າການຈັບຄູ່ຂຽນໂປຣແກຣມບໍ່ແມ່ນ ກົນລະຍຸດ "ແບ່ງປັນ ແລະ ເອົາຊະນະ", ແຕ່ເປັນຄວາມພະຍາຍາມຮ່ວມມືທີ່ແທ້ຈິງໃນທຸກໆຄວາມພະຍາຍາມສຳລັບໂຄງການທັງໝົດ. ຄູຄວນ ຫຼີກເວັ້ນການຈັບຄູ່ນັກຮຽນທີ່ອ່ອນຫຼາຍກັບນັກຮຽນທີ່ເກັ່ງຫຼາຍ.

ສ້າງແຮງຈູງໃຈໃຫ້ກັບນັກຮຽນໂດຍການໃຫ້ສິ່ງຈູງໃຈພິເສດ

ການສະເໜີແຮງຈູງໃຈພິເສດສາມາດຊ່ວຍກະຕຸ້ນໃຫ້ນັກຮຽນຈັບຄູ່, ໂດຍສະເພາະກັບນັກຮຽນທີ່ເກັ່ງ. ຄູບາງຄົນພົບວ່າການກຳໜົດໃຫ້ນັກຮຽນ ຈັບຄູ່ພຽງແຕ່ໜຶ່ງ ຫຼື ສອງວຽກນັ້ນມີປະໂຫຍດຫຼາຍ.

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

Pair Programming Practice

ປ້ອງກັນການໂກງໃນການເຮັດວຽກຮ່ວມກັນ (Prevent collaboration cheating)

ຄວາມທ້າທາຍສຳລັບຄູແມ່ນການຊອກຫາວິທີທີ່ຈະປະເມີນຜົນໄດ້ຮັບຂອງແຕ່ລະຄົນ, ໃນຂະນະທີ່ນຳໃຊ້ຜົນປະໂຫຍດຂອງການເຮັດວຽກຮ່ວມກັນ. ຈະຮູ້ໄດ້ແນວໃດວ່ານັກຮຽນຕັ້ງໃຈຮຽນ ຫຼື ຖືກໂກງ? ຜູ້ຊ່ຽວຊານແນະນຳໃຫ້ທົບທວນຄືນການອອກແບບຫຼັກສູດ ແລະ ການປະເມີນອີກຄັ້ງໜຶ່ງ ພ້ອມທັງປຶກສາຫາລືກັບນັກຮຽນຢ່າງຈະແຈ້ງ ແລະ ເປັນຮູບປະທຳ ກ່ຽວກັບພຶດຕິກຳທີ່ຈະຖືກຕີຄວາມໝາຍວ່າ ເປັນການໂກງ. ຜູ້ຊ່ຽວຊານຊຸກຍູ້ໃຫ້ຄູມີການມອບໝາຍວຽກໃຫ້ມີຄວາມໝາຍຕໍ່ນັກຮຽນ ແລະ ອະທິບາຍຄຸນຄ່າຂອງສິ່ງທີ່ນັກຮຽນຈະຕ້ອງໄດ້ຮຽນຮູ້ ເມື່ອເຮັດສຳເລັດ.

ສະພາບແວດລ້ອມຂອງການຮຽນຮູ້ຮ່ວມກັນ (Collaborative learning environment)

ສະພາບແວດລ້ອມການຮຽນຮູ້ຮ່ວມກັນເກີດຂຶ້ນໄດ້ທຸກເວລາ ທີ່ຜູ້ສອນຮຽກຮ້ອງໃຫ້ນັກຮຽນເຮັດກິດຈະກຳການຮຽນຮູ້ຮ່ວມກັນ. ສະພາບແວດລ້ອມການຮຽນຮູ້ຮ່ວມກັນສາມາດມີສ່ວນຮ່ວມທັງກິດຈະກຳທີ່ເປັນທາງການ ແລະ ບໍ່ເປັນທາງການ ແລະ ອາດຈະບໍ່ລວມເຖິງການປະເມີນແບບໂດຍກົງ. ຕົວຢ່າງ, ນັກສຶກສາຄູ່ໜຶ່ງເຮັດວຽກກ່ຽວກັບການຂຽນໂປຣແກຣມ; ນັກສຶກສາກຸ່ມນ້ອຍໆສົນທະນາຄຳຕອບທີ່ເປັນໄປໄດ້ຕໍ່ກັບຄຳຖາມຂອງອາຈານໃນລະຫວ່າງການບັນລະຍາຍ ແລະ ນັກຮຽນເຮັດວຽກຮ່ວມກັນນອກຫ້ອງຮຽນເພື່ອຮຽນຮູ້ແນວຄວາມຄິດໃໝ່ໆ. ການຮຽນຮູ້ຮ່ວມກັນແມ່ນແຕກຕ່າງຈາກໂຄງການທີ່ນັກຮຽນ "ແບ່ງປັນ ແລະ ເອົາຊະນະ." ເມື່ອນັກຮຽນແບ່ງວຽກ, ແຕ່ລະຄົນຈະຮັບຜິດຊອບພຽງແຕ່ສ່ວນໜຶ່ງຂອງການແກ້ໄຂບັນຫາ ແລະ ມີໂອກາດຈຳກັດຫຼາຍສຳລັບການເຮັດວຽກຜ່ານບັນຫາຮ່ວມກັບຄົນອື່ນ. ໃນສະພາບແວດລ້ອມທີ່ມີການເຮັດວຽກຮ່ວມກັນ, ນັກຮຽນມີສ່ວນຮ່ວມໃນການສົນທະນາທາງປັນຍາເຊິ່ງກັນ ແລະ ກັນ.

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

Q1. ອິງການ A ກຳລັງວາງແຜນທີ່ຈະຈອກປີ້ຄອນເສີດ ສຳລັບຄອນເສີດຂອງນັກຮ້ອງ idol. ໃນທີ່ນີ້, n ແມ່ນຈຳນວນຕົວເລກທີ່ input ແລະ ຈຳນວນບ່ອນນັ່ງຖືກຈັດລຽງດັ່ງຕໍ່ໄປນີ້. n* n ບ່ອນນັ່ງຖືກວາງໄວ້ເມື່ອ n ຖືກກຳນົດໃຫ້ເປັນ input. ດ້ານລຸ່ມນີ້ເອີ້ນວ່າ array matrix ທີ່ຮູບຮ່າງເໝືອນງວງຊ້າງ. ຈົ່ງຂຽນໂປຣແກຣມທີ່ສ້າງ array ຂອງຕົວເລກເຫຼົ່ານີ້.

Output Example

```
Enter n : 5
 1   2   3   4   5
10   9   8   7   6
11  12  13  14  15
20  19  18  17  16
21  22  23  24  25
```