



# Samsung Innovation Campus

| Coding, Programming & Data Science

Together for Tomorrow!  
**Enabling People**

Education for Future Generations



# Coding, Programming & Data Science

## ພາບລວມຂອງຫຼັກສູດ (Course Overview )

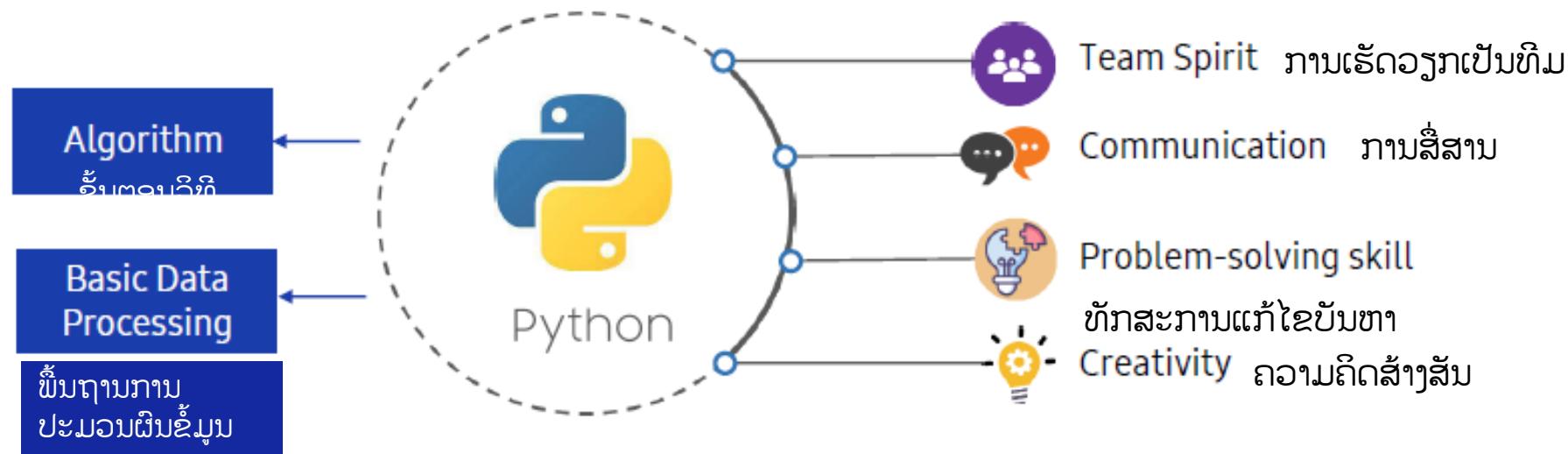
# Coding, Programming & Data Science

## ພາບລວມຂອງຫຼັກສູດ (Course Overview )

### ເປົ້າໝາຍຂອງຫຼັກສູດ(Course Goal):

ໃນຫຼັກສູດ C,P ແລະ D ນັກສຶກສາຮຽນກ່ຽວກັບທັກສະການຂຽນໂປຣແກຣມພາສາ Python, ຂັ້ນຕອນວິທີ (Algorithm) ແລະ ພື້ນຖານການປະມວນຜົນຂໍ້ມູນດ້ວຍພາສາ Python. ນັກສຶກສາບໍ່ພຽງແຕ່ເສີມຂະຫຍາຍທັກສະດ້ານເຫັກໂນໂລຊີ (Python) ເທົ່ານັ້ນ, ແຕ່ຍັງມີທັກສະໃນການນຳໃຊ້ Python ເຊົ້າໃນການຜະລິດໃນຊີວິດຕົວຈິງ.

ໃນຂະນະທີ່ຮຽນຫຼັກສູດນີ້, ນັກສຶກສາຮຽນຮູ້ວິທີແກ້ໄຂບັນຫາ ແລະ ປະສົບການໃນການເຮັດວຽກຮ່ວມກັນກັບໜຸ່ຮ່ວມຫ້ອງ.



# Coding, Programming & Data Science

## ພາບລວມຂອງຫຼັກສູດ (Course Overview )

## Course Module

### Module 1: ພື້ນຖານ Python (Python Basic)

ຮຽນກ່ຽວກັບພື້ນຖານພາສາ Python ແລະ ແນວດວາມຄິດທີ່ສໍາຄັນກ່ຽວກັບ ວິທະຍາສາດຄອມພິວເຕີ ແລະ ການຂຽນ ໂປຣແກຣມ

Chapter 1	Programming Basic Concept and Starting Python
Chapter 2	Python Programming Basic
Chapter 3	Effective Python Programming

### Module 2: Algorithm ຂັ້ນຕອນວິທີ

ເຊົ້າໃຈແນວຄົດພື້ນຖານຂອງຂັ້ນຕອນວິທີຕ່າງໆ (algorithms) ແລະ ຜິກຊ້ອມການນຳໃຊ້ຂັ້ນຕອນວິທີຕ່າງໆກັບຕົວຢ່າງຕົວຈິງໃນໂລກດ້ວຍພາສາ Python

Chapter 4	Algorithm I: Data Structure
Chapter 5	Algorithm II: Sorting Algorithms
Chapter 6	Algorithm III: Problem Solving with Algorithms

### Module 3: Data Analysis Basic ພື້ນຖານການວິເຄາະຂໍ້ມູນ

ຮຽນພື້ນຖານການວິເຄາະຂໍ້ມູນດ້ວຍໂມດຸນ Pandas ໃນພາສາ Python ນີ້ແມ່ນຫັກສະທິ່ມປະໂຫຍດສໍາລັບພະນັກງານລະດັບເລື່ມຕົ້ນ ໃນຕະຫຼາດແຮງງານ.

Chapter 7	Data Processing and Descriptive Statistics, Data Visualization
Chapter 8	Data Analysis and Visualization – Mini Project

Module 3 ບໍ່ຕ້ອງໃຊ້ຄະນິດສາດລະດັບ ມະຫາວິທະຍາໄລ.

# Coding, Programming & Data Science

## ພາບລວມຂອງຫຼັກສູດ (Course Overview )

- Course Module

**Module 1**

Course Contents	Time
Chapter 1. Programing Basic Concept and Starting Python	18H
Unit 1. Sequential programming	2H
Unit 2. Planning for programming	2H
Unit 3. Basic of Numeric Data Types and Arithmetic Operation	2H
Unit 4. Variables and Inputs	2H
Unit 5. Logic and Comparison Operators	2H
Unit 6. Conditional Statement-1: Conditions and Decision Making	2H
Unit 7. Conditional Statement -2: Making decisions in two directions and applying conditional statement	2H
Unit 8. Loop-1	2H
Unit 9. Loop-2	2H
Chapter 2. Python Programing Basic - Sequence Data type in Python	14H
Unit 10. Lists and Tuple Data Types	2H
Unit 11. Dictionary Data Type	2H
Unit 12. Addressing Sequence Types	2H
Unit 13. Two-Dimensional List	2H
Unit 14. Dictionary Method 1	2H
Unit 15. Dictionary Method 2	2H
Unit 16. Set Data Types	2H
Chapter 3. Effective Python Programing - Function, Closure and Class	10H
Unit 17. Function	2H
Unit 18. Recursion Function Call	2H
Unit 19. Lambda	2H
Unit 20. Closure	2H
Unit 21. Class	2H

**Module 2**

Course Contents	Time
Chapter 4. Algorithm 1- Data Structures	10H
Unit 22. Stack and Queue	2H
Unit 23. Queue	2H
Unit 24. Linear Search	2H
Unit 25. Binary Search	2H
Unit 26. Harsh Table	2H
Chapter 5. Algorithm 2 -Sorting Algorithms	6H
Unit 27. Bubble, Selection, and Insertion Sort	2H
Unit 28. Merge Sort	2H
Unit 29. Quick Sort	2H
Chapter 6. Algorithm 3– Problem Solving with Algorithms	8H
Unit 30. Greedy Approach	2H
Unit 31. divide-and-conquer	2H
Unit 32. Dynamic Programming	2H
Unit 33. Backtracking	2H
Chapter 7. Data Processing and Descriptive Statistics, Data Visualization	10H
Unit 34. Using Python Modules	2H
Unit 35. Pandas Series for Data Processing	2H
Unit 36. Pandas DataFrame for Data Processing	2H
Unit 37. Data Tidying	2H
Unit 38. Time Series Data	2H
Chapter 8. Data Analysis and Data Visualization - Mini Project	4H
Unit 39. Financial Data Analysis Mini Project	2H
Unit 40. Global Corona Pandemic Analysis Mini Project	2H

**Module 3**

# Coding, Programming & Data Science

## ພາບລວມຂອງຫຼັກສູດ (Course Overview )

### Class Schedule

ຫຼັກສູດ C,P ແລະ D ໃຊ້ເວລາ 87 ຊົ່ວໂມງ, ເຊິ່ງປະກອບມີການບັນລະຍາຍແຕ່ລະບົດ (unit lecture), ບົດຝຶກຫັດ (exercise) ແລະ ບົດທິດສອບ (quiz).

Course Contents	Time	Course Contents	Time
Chapter 1. Python Basics	19H	Chapter 4. Algorithm 1– Data Structures	11H
Unit 1. Sequential programming	2H	Unit 22. Stack and Queue	2H
Unit 2. Planning for programming	2H	Unit 23. Queue	2H
Unit 3. Basic of Numeric Data Types and Arithmetic Operation	2H	Unit 24. Linear Search	2H
Unit 4. Variables and Inputs	2H	Unit 25. Binary Search	2H
Unit 5. Logic and Comparison Operators	2H	Unit 26. Harsh Table	2H
Unit 6. Conditional Statement-1: Conditions and Decision Making	2H	Quiz	1H
Unit 7. Conditional Statement -2: Making decisions in two directions and applying conditional statement	2H	Chapter 5. Algorithm 2 -Sorting Algorithms	7H
Unit 8. Loop-1	2H	Unit 27. Bubble, Selection, and Insertion Sort	2H
Unit 9. Loop-2	2H	Unit 28. Merge Sort	2H
Quiz	1H	Unit 29. Quick Sort	2H
Chapter 2. Key of Python Basic	15H	Quiz	1H
Unit 10. Lists and Tuple Data Types	2H	Chapter 6. Algorithm 3– Problem Solving with Algorithms	9H
Unit 11. Dictionary Data Type	2H	Unit 30. Greedy Approach	2H
Unit 12. Addressing Sequence Types	2H	Unit 31. divide-and-conquer	2H
Unit 13. Two-Dimensional List	2H	Unit 32. Dynamic Programming	2H
Unit 14. Dictionary Method 1	2H	Unit 33. Backtracking	2H
Unit 15. Dictionary Method 2	2H	Quiz	1H
Unit 16. Set Data Types	2H	Chapter 7. Data Processing and Descriptive Statistics, Data Visualization	11H
Quiz	1H	Unit 34. Using Python Modules	2H
Chapter 3. Python Intensive	11H	Unit 35. Pandas Series for Data Processing	2H
Unit 17. Function	2H	Unit 36. Pandas DataFrame for Data Processing	2H
Unit 18. Recursion Function Call	2H	Unit 37. Data Tidying	2H
Unit 19. Lambda	2H	Unit 38. Time Series Data	2H
Unit 20. Closure	2H	Quiz	1H
Unit 21. Class	2H	Chapter 8. Data Analysis and Data Visualization - Mini Project	4H
Quiz	1H	Unit 39. Financial Data Analysis Mini Project	2H
		Unit 40. Global Corona Pandemic Analysis Mini Project	2H
		Quiz	1H
		Total Class Hours	80H

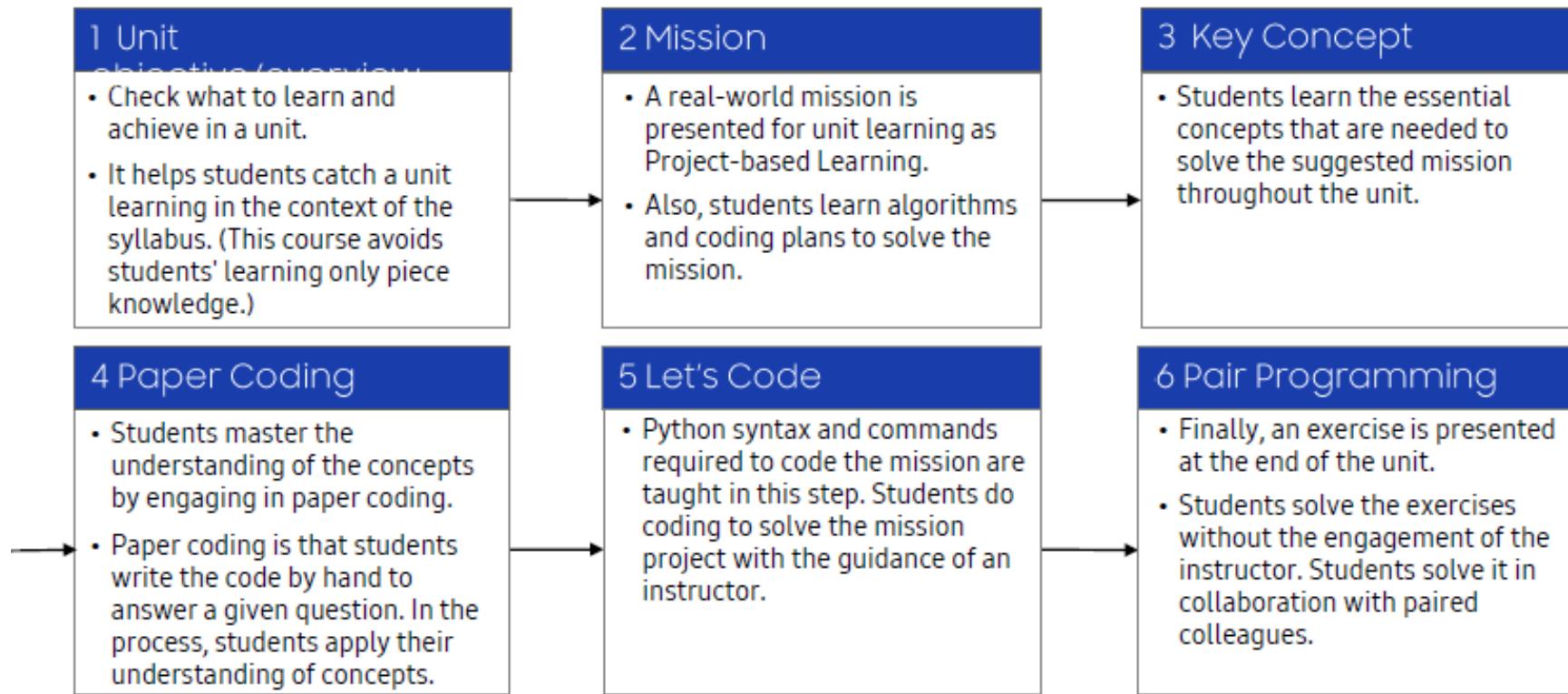
# Coding, Programming & Data Science

## ພາບລວມຂອງຫຼັກສູດ (Course Overview )

### Unit Lesson Flow ລຳດັບການດຳເນີນບົດຽນ

ແຕ່ລະບົດຽນ (unit) ແມ່ນໃຊ້ເວລາ 2 ຊົ່ວໂມງ. ຄຸນິການສອນແຕ່ລະຂັ້ນຕອນຂອງແຕ່ລະ Unit ໄດ້ອະທິບາຍໃນ slide ດັ່ງລຸ່ມນີ້

\* It is recommended to follow the step-by-step guide by opening the Unit 5 as a sample next to you for better understanding.



# Coding, Programming & Data Science

## ພາບລວມຂອງຫຼັກສູດ (Course Overview )

Assessment Breakdown and Certificate Qualification

### ເງື່ອນໄຂໃນການຮັບໃບຢັ້ງຢືນ

ການປະເມີນຈະດຳເນີນຕະຫຼອດຫຼັກສູດໂດຍຜູ້ສອນ

1). ການເຂົ້າຮຽນປີກະຕິ, 2). ການມີສ່ວນຮ່ວມເຮັດວຽກທັດ, 3). ຜົນລວມຂອງການປະເມີນຜົນທັງໝົດ  
ຈະຖືກນຳມາໃຊ້ໃນການປະເມີນນັກສຶກສາແຕ່ລະຄົນເພື່ອຮັບໃບຢັ້ງຢືນ ແລະ ຈັດລຽງຕາມການປະຕິບັດອັນດັບ  
ສູງສຸດ.

ໃບຢັ້ງຢືນການສໍາເລັດຈະຖືກອອກໃຫ້:

- ນັກສຶກສາທີ່ເຂົ້າຮຽນເກີນ 90% ຂອງຊື່ວໂມງຮຽນ
- ນັກສຶກສາທີ່ມີຜົນການປະເມີນທັງໝົດເກີນ 50 ຄະແນນຂຶ້ນໄປ

ການຄັດເລືອກຜູ້ທີ່ມີຜົນການຮຽນດີ

- ນັກສຶກສາຜູ້ທີ່ໄດ້ຄະແນນສູງສຸດຈາກຄະແນນປະເມີນທັງໝົດ.

Chapter 1. ບົດທີ1

# Programing Basic Concept and Starting Python

ແນວຄວາມຄິດພື້ນຖານກ່ຽວກັບການຂຽນໂປຣແກຣມ ແລະ ການເລີ່ມຕົ້ນ Python

Coding, Programming & Data Science

# Chapter Description

---

## ◆ Chapter objectives จุดประสงค์ของบิ๊ด

นักศึกษาจะสามารถตั้งค่าและพิมพ์ผลลัพธ์ของการประมวลผล Python และ รู้วิธีเขียนภาษา Python ของโปรแกรม. นักเรียนจะถูกฝึกให้รู้สึกถึงภาษา Python ด้วย: การป้อนข้อมูล/ผู้ใช้ได้รับ, ประมวลผลข้อมูล และ ตัวบ่งชี้. นอกจากนี้, พิเศษเช่นเดียวกันจะได้รับทักษะในการแก้ไขข้อบกพร่องในภาษา Python code ผ่านตัวบ่งชี้การดำเนินการ (operators), เงื่อนไข (conditionals), และ loops.

## ◆ Chapter contents เมื่อในของบิ๊ด

- ✓ Unit 1. Sequential Programming
- ✓ Unit 2. Planning for Programming
- ✓ Unit 3. Basic of Numeric Data Types and Arithmetic Operation
- ✓ Unit 4. Variables and Inputs
- ✓ Unit 5. Logic and Comparison Operations
- ✓ Unit 6. Conditional Statement-1: Conditions and Decision Making
- ✓ Unit 7. Conditional Statement -2: Making decisions in two directions and applying conditional statements
- ✓ Unit 8. Loop-1
- ✓ Unit 9. Loop-2

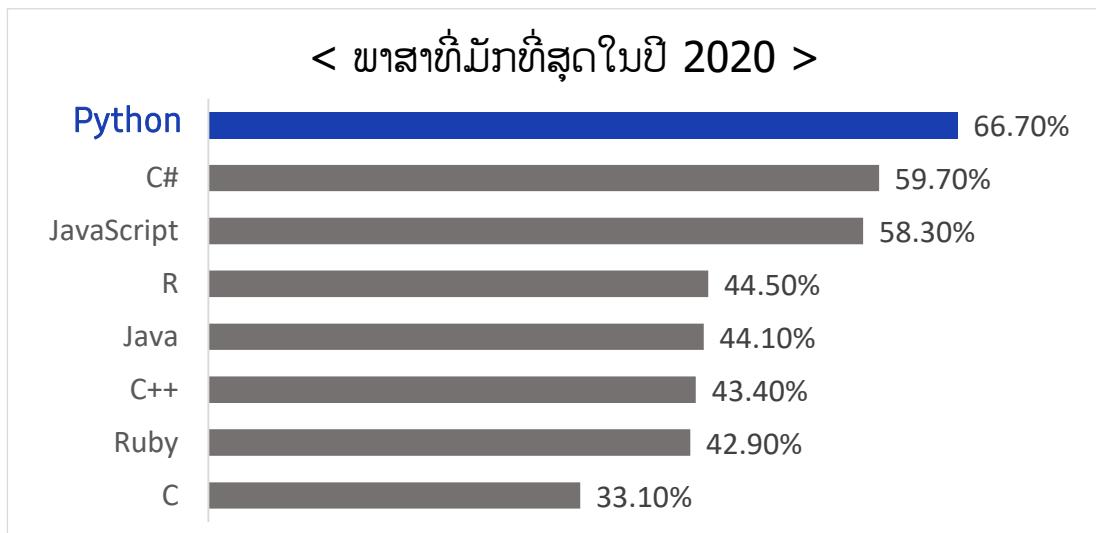
Orientation

# Introduction to Python

# ແນະນຳກ່ຽວກັບ Python

## Python ແມ່ນຫຍັງ

- | Python ເປັນພາສາໂປຣແກຣມລະດັບສູງ ແລະ ເປັນໜຶ່ງໃນພາສາໂປຣແກຣມທີ່ໄດ້ຮັບຄວາມນິຍົມຫຼາຍໆທີ່ສຸດໃນໂລກ.
- | Python ເປັນພາສາທີ່ສາມາດໃຊ້ໄດ້ຢ່າງກວ້າງຂວາງທຸກຂີ່ງເຂດ. ດັ່ງນັ້ນ, Python ຖືກນຳໃຊ້ຢ່າງກວ້າງຂວາງເພື່ອສ້າງໂປ່ງແນວ, ວິທະຍາສາດຂໍ້ມູນ, ການຝັດທະນາເວັບ, ແລະ ອັດຕະໂນມັດຈິນເຖິງການເຮັດສິ່ງຕ່າງໆໂດຍທົ່ວໄປ.
- | ເນື່ອງຈາກພາສາ python ເປັນພາສາທີ່ເຂົ້າໃຈງ່າຍ, ດັ່ງນັ້ນ Python ຈຶ່ງຖືກນຳໃຊ້ໂດຍນັກຝັດທະນາ ແລະ ຜູ້ບໍ່ແມ່ນນັກຝັດທະນາ.



<https://insights.stackoverflow.com/survey/2020>

# ເປັນຫຍັງ Python ຈຶ່ງເປັນທີ່ນີ້ຍືມ?

| Python ເປັນທີ່ນີ້ຍືມໃນຂຶ້ງເຂດການສຶກສາເພາະວ່າມັນຈ່າຍຕໍ່ການອ່ານ, ຂຽນ ແລະ ຮຽນຮູ້.

- ▶ Code ຂອງພາສາ Python ສາມາດຂຽນ ແລະ ຕິດວາມໝາຍຄ້າຍຄືກັນກັບພາສາຂອງມະນຸດ, ດັ່ງນັ້ນມັນຈຶ່ງສາມາດອ່ານໄດ້ ແລະ ຈ່າຍຕໍ່ການຮຽນຮູ້ສໍາລັບຜູ້ເລີ່ມຕົ້ນ. ນັກຮຽນສາມາດອອກແບບການເຮັດວຽກຂອງໂປຣແກຣມໃນຮູບແບບພາສາຂອງມະນຸດ ແລະ ປັບປຸງເປັນ code ພາສາ Python ໄດ້ຢ່າງຈ່າຍດາຍ.
- ▶ Syntax ຂອງ Python ສາມາດທຳຄວາມເຂົ້າໃຈ ແລະ ຈ່າຍຕໍ່ການຂຽນ, ຊ່ວຍໃຫ້ຜູ້ເລີ່ມຕົ້ນຂຽນໂປຣແກຣມໃໝ່ບໍ່ຕ້ອງເສຍເວລາກັບ syntax ທີ່ສັບສົນ.

Python syntax    `if 4 in [1,2,3,4]: print("There is 4.")`

Interpreted    If there is 4 among 1, 2, 3, and 4, print "There is 4."

## ເປັນຫຍັງ Python ຈຶ່ງເປັນທີ່ນີ້ຍືມ

| Python ຍັງເປັນທີ່ນີ້ຍືມຈາກບໍລິສັດ ແລະ application ໃຫ່ຍໆ ເນື່ອງຈາກມີຄວາມສາມາດຮອບດ້ານ.

- ▶ Python ເປັນທີ່ຕ້ອງການບໍ່ພຽງແຕ່ໃນຂຶ່ງເຂດການສຶກສາເທົ່ານັ້ນ, ແຕ່ຍັງມີບໍລິສັດໃຫຍ່ໆ ເຊັ່ນ: Google, Intel, eBay, Netflix, Instagram, Dropbox, ແລະ Slack, ນຳໃຊ້ Python ສໍາລັບການພັດທະນາ ແລະ ຮັກສາ applications ຂອງພວກເຂົາ.

< ບໍລິສັດໃຫຍ່ໆ ທີ່ໃຊ້ Python >



- ▶ **Google** ນຳໃຊ້ Python ໃນການພັດທະນາ software ເພື່ອເພີ່ມຄວາມສາມາດຂອງເຄື່ອງມືຕົ້ນຫາ
- ▶ **Netflix** ນຳໃຊ້ Python ເພື່ອແນະນຳໜັງ, ລາຍການໂທລະພາບ ແລະ ສາລະຄະດີ ຕາມປະຫວັດຂອງຜູ້ໃຊ້.
- ▶ **Instagram** ນຳໃຊ້ Python ເພື່ອຕົກແຕ່ງສ່ວນການສໍາຫຼວດສໍາລັບຜູ້ໃຊ້.
- ▶ **Spotify** ນຳໃຊ້ Python ເພື່ອແນະນຳເພິ່ນໃຫ້ແກ່ຜູ້ໃຊ້
- ▶ **Reddit** ນຳໃຊ້ Python ສໍາລັບພັດທະນາເວັບເພື່ອຫຼຸດຜ່ອນຄວາມຊັບຊັອນຂອງສ່ວນຖາມ-ຕອບ.

<https://www.botreetechnologies.com/blog/pros-and-cons-of-python/>

# პანთაკ Python ჯერპანთისმი

| ადგინდებ უძრავი მოყვარული Python ინსტანციები მასში მარტივი და მაღალ მოძრაობის მქონე დანართების გადასატანი.

- ▶ Python შეიცავს უზრუნველყოფილი დანართების კოდების დანართების გადასატანი. მაგალითად შემდეგი დანართები მომიჯნავე არის:



<https://light-it.net/blog/top-10-python-libraries-for-machine-learning/>

- ▶ **Pandas** წარმოადგენს სტრუქტურულ და დანართულ მონაცემთა სისტემას, რომელიც გადასატანი და მომიჯნავე არის.
- ▶ **NumPy** წარმოადგენს მარტივ და მაღალ მოძრაობის მქონე დანართების გადასატანი.
- ▶ **Tensor Flow** წარმოადგენს მარტივ და მაღალ მოძრაობის მქონე დანართების გადასატანი.
- ▶ **Keras** წარმოადგენს მარტივ და მაღალ მოძრაობის მქონე დანართების გადასატანი.

## ເປັນຫຍັງ Python ຈຶ່ງເປັນທີ່ນີ້ຍືມ?

- | Python ເປັນພາສາໂປຣແກຣມຫຼັກທີ່ນຳໄປສູ່ການປະຕິວັດອຸດສາຫະກຳ 4.0 ດ້ວຍ libraries ເຫຼື້ນໍ້າ.
- ▶ ການປະຕິວັດອຸດສາຫະກຳ 4.0 ແມ່ນສຸມໃສ່ວຽກງານອັດຕະໂນມັດ, ການສື່ສານທີ່ດີຂຶ້ນ ແລະ ການກວດສອບເຄື່ອງຈັກອັດສະລິຍະດ້ວຍຕົນເອງ. ໂດຍສະເພາະກ່ຽວກັບ machine learning ແລະ deep learning ພວມເຕີບໃຫຍ່ຂະຫຍາຍຕົວຢ່າງໄວ.
- ▶ ດັ່ງທີ່ກ່າວໄວ້ໃນ slide ທີ່ຜ່ານມາ Python ຮອງຮັບ libraries ຫຼາກຫຼາຍທີ່ມີສ່ວນຮ່ວມໃນການຮຽນຮູ້ຂອງຄອມພິວເຕີ ແລະ ການຮຽນຮູ້ແບບເລີກເຊິ່ງກ່ຽວກັບປັນຍາປະດິດ. ມັນໄດ້ຮັບການພັດທະນາໄປພ້ອມກັບຄວາມຄືບໜັ້ນຂອງການປະຕິວັດອຸດສາຫະກຳ 4.0 ແລະ ມີປິດບາດລໍາຄັນໃນການເສີມຂະຫຍາຍການຂະຫຍາຍຕົວຢ່າງວ່ອງໄວ.
- | Libraries ແລະ frameworks ຂອງ Python ຊ່ວຍໃຫ້ສາມາດພັດທະນາ full stack ໃນ workplaces ດັ່ງ
- ▶ Python ຍັງມີ Python framework ປະເພດຕ່າງໆ, ເຊັ່ນ Django, Flask ແລະ FastAPI. ດ້ວຍການໃຊ້ປະໂຫຍດຈາກ frameworks ແລະ libraries ເຫຼື້ນໍ້າ, Python ສາມາດນຳໃຊ້ໄດ້ທັງການຊຽນໂປຣແກຣມແບບ front-end ແລະ back-end. ຊ່ວຍໃຫ້ສາມາດພັດທະນາ **full stack**, ເຊິ່ງເປັນທີ່ຕ້ອງການໃນຫຼາຍຕໍ່ແຫ່ນໆງານໃນອຸດສາຫະກຳທີ່ກ່ຽວຂ້ອງ.

# Python ນຳໃຊ້ເພື່ອຫຍັງ?

ປັນຍາປະດິດ  
&  
ການຮຽນຮູ້ຂອງຄອມພົວ  
ຕີ

ວິທະຍາສາດຂໍ້ມູນ  
&  
ການສະແດງຂໍ້ມູນ  
(Data Visualization)

ການພັດທະນາເວັບໄຊ  
(Web Development)

ການພັດທະນາເກມ  
(Game Development)

ລະບົບອັດຕະໂນມັດ  
(Automation)  
ເຮັດວຽກງ່າຍໆ ໃນຄອມພົວຕີ  
ແບບອັດຕະໂນມັດ

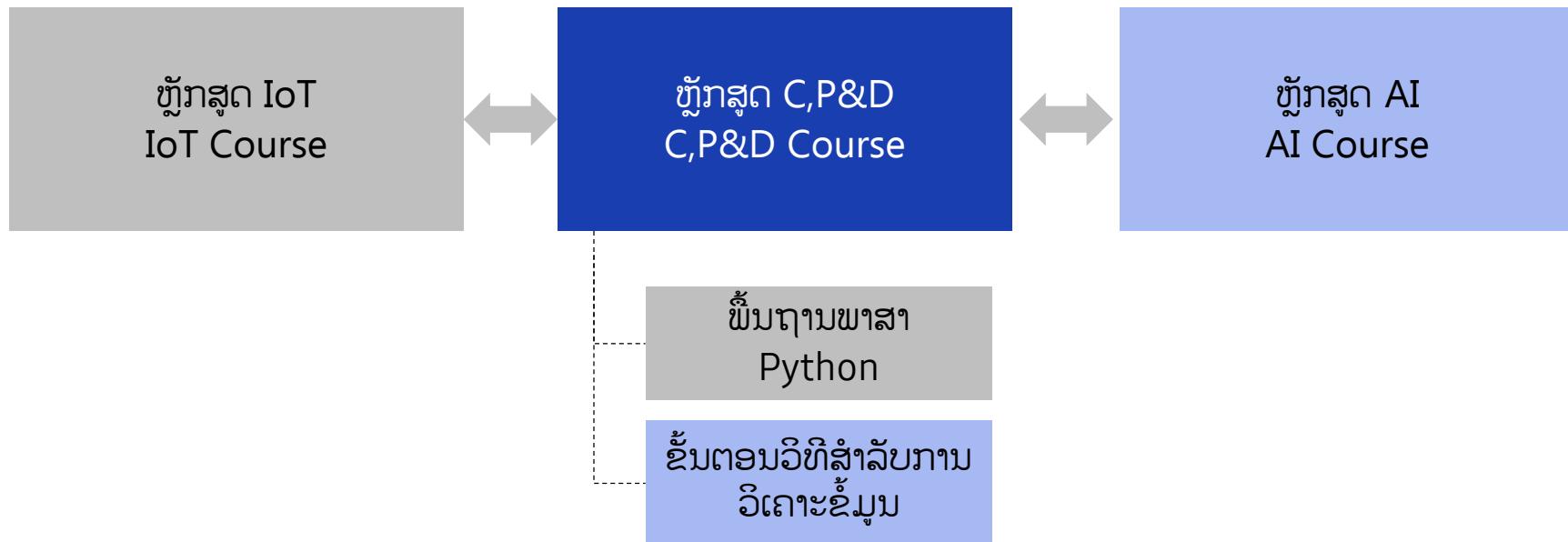
ການທຶດສອບຂໍອບແວ  
ແລະ ການສ້າງຕົ້ນແບບ  
(Software Testing and  
Prototyping)

ຫຼຸດຄວາມຊ້າຊ້ອນຂອງວຽກປະຈຳວັນຜ່ານລະບົບ  
ອັດຕະໂນມັດ

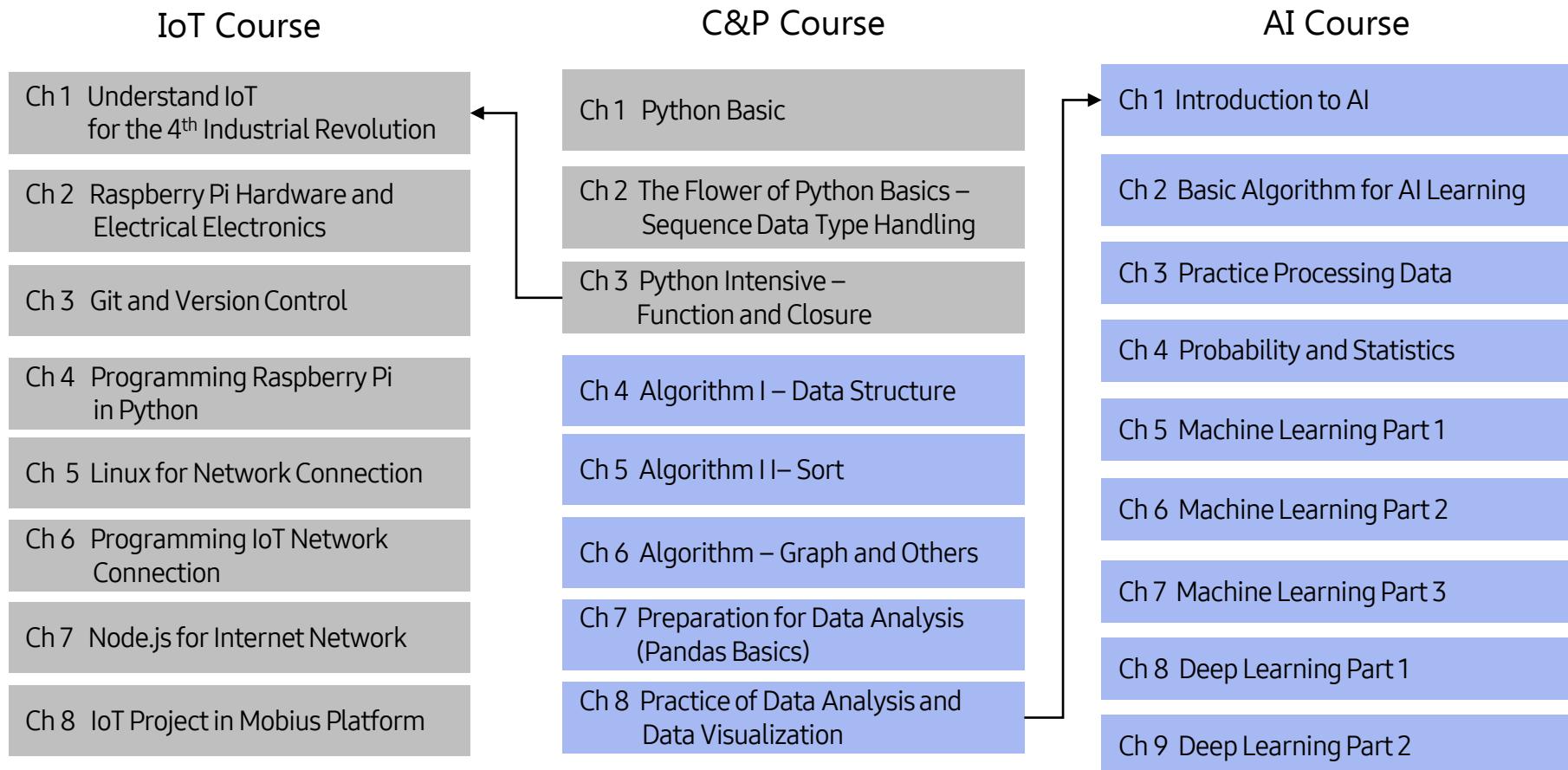
- ▶ ອັບເດດລາຍການຊື້ເຕື່ອງໃຊ້
- ▶ ການປ່ຽນຊື່ File ຈຳນວນຫຼາຍ
- ▶ ການປ່ຽນ files ຂໍຄວາມເປັນ spreadsheets
- ▶ ການຕື່ມແບບຟອມອອນໄລນ໌ອັດຕະໂນມັດແລະອື່ນໆ.

# ໃຊ້ຫຼັກສູດ Coding, Programming & Data Science ແນວໃດ?

- | ຫຼັກສູດນີ້ຖືກອອກແບບມາເພື່ອໃຫ້ນັກສຶກສາມີພື້ນຖານພາສາ Python ຢ່າງໜັກແໜ້ນ. ເນື່ອຈາກເປັນແຄນຫຼັກຂອງໂຄງການ SIC, ຈຶ່ງຖືກແຍກສ່ວນເພື່ອໃຫ້ເຊື່ອມໄປຢ່າງມີປະສິດທິພາບກັບຫຼັກສູດອື່ນງໜອງ SIC, ແລ້ວ: ຫຼັກສູດ AI ແລະ IoT.
- | ເນື່ອສໍາເລັດຫຼັກສູດນີ້, ນັກສຶກສາມາສາດນຳໃຊ້ຄວາມຮູ້ ແລະ ທັກສະພາສາ Python ຂັ້ນພື້ນຖານໄດ້.



# ໃຊ້ຫຼັກສູດ Coding, Programming & Data Science ແນວໃດ?



Unit 1.

# Sequential Programming

## ภาษาชีวน์โปรแกรมแบบลำดับ

## Learning objectives

- ✓ ຜູ້ຮຽນຈະສາມາດນິຍາມ ແລະ ຮຸ້າເຖິງຄວາມຈຳເປັນຂອງການດິດແບບຄໍານວນ.
- ✓ ຜູ້ຮຽນຈະສາມາດຕິດຕັ້ງຊອບແວທີ່ຈຳເປັນສໍາລັບ Python ຕັ້ງແຕ່ເລີ່ມຕົ້ນ.
- ✓ ຜູ້ຮຽນຈະສາມາດຕັ້ງຄ່າສະພາບແວດລ້ອມຈຳລອງ ນອກຈາກທີ່ຮັດການຕິດຕັ້ງ Python 3.x ແລະ ຍັງສາມາດຕິດຕັ້ງ Python ລຸ້ນຕ່າງໆໄດ້.
- ✓ ຜູ້ຮຽນຈະສາມາດສະຫຼຸບລຳດັບການຂຽນ code ໂດຍການເບິ່ງຜົນໄດ້ຮັບໂດຍໃຊ້ພຽງແຕ່ຟັງຊັນການພິມ print()
- ✓ ຜູ້ຮຽນຈະສາມາດຈຳແນກຄວາມຜິດໄດ້ລະຫວ່າງດ syntax ແລະ runtime

## Learning overview

- ✓ ຮຽນຮູ້ນິຍາມ ແລະ ຄວາມຈຳເປັນຂອງການຄິດແບບຄໍານວນ
- ✓ ກຳນົດຄ່າສະພາບແວດລ້ອມພື້ນຖານສໍາລັບ Python ໂດຍການຕິດຕັ້ງ Anaconda.
- ✓ ເຊົ້າໃຈຄວາມຕ້ອງການສະພາບແວດລ້ອມຈໍາລອງ ແລະ ການກຳນົດຄ່າ, ການລຶບ ແລະ ການນຳໃຊ້ສະພາບແວດລ້ອມຈໍາລອງ ຂຶ້ນກັບ ຈຸດປະສົງຂອງການນຳໃຊ້
- ✓ ຮຽນຮູ້ UI ແລະ ຄໍາສັ່ງຂອງ Jupyter Notebook
- ✓ ຜິກການຂຽນໂປຣແກຣມແບບລໍາດັບ ແລະ ນຳໃຊ້ຝັງຊັນ print()
- ✓ ຮຽນຮູ້ແວດຄວາມຄິດຫຼັກຂອງວິທີການ Pytonic , ເຊັ່ນ: ການຫຍື້ຫນ້າ , ຄໍາອະທິບາຍ , ລະຫັດທີ່ສາມາດອ່ານໄດ້ , ຄໍາສັບສໍາຄັນ ແລະ ຫຼັກການຕັ້ງຊື່ຕົວປ່ຽນ
- ✓ ຮຽນຮູ້ຈໍາແນກລະຫວ່າງຄວາມຜິດແບບ syntax ແລະ ຄວາມຜິດແບບ runtime.

## Keywords

Computational thinking

Virtual Environment

Jupyter Notebook

Sequential Structure

Annotation

Naming Conventions

Expressions

Syntax Error

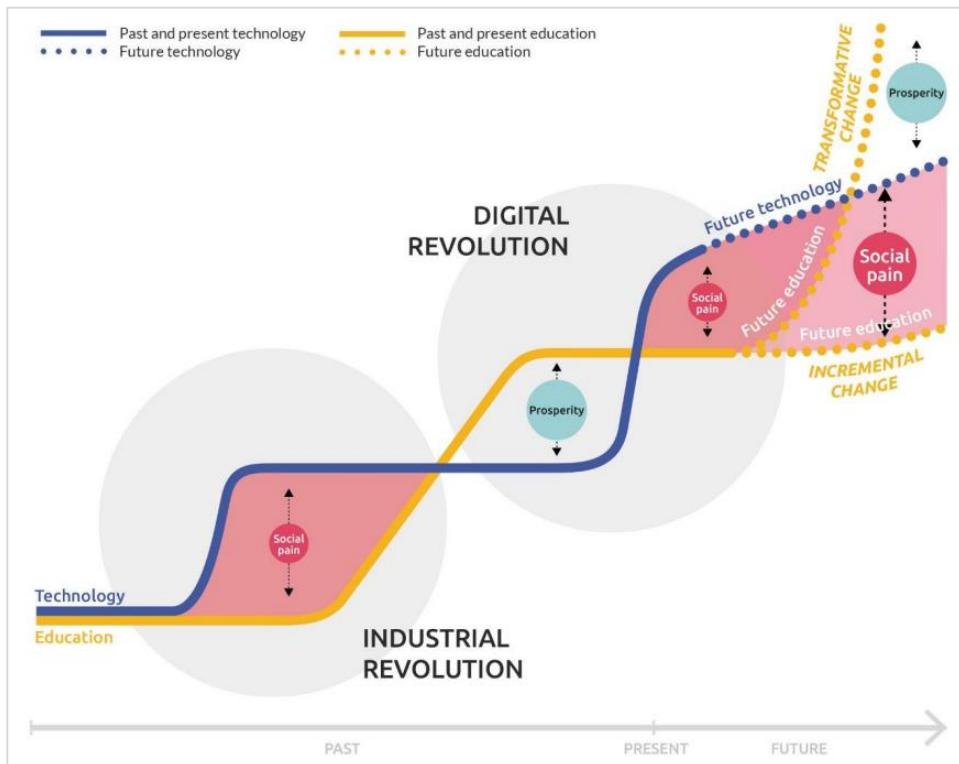
Runtime Error

# Mission

## 1. Real world problem

### 1.1. 47% of jobs will disappear in the next 20 years due to tech innovation.

47% ຂອງອາຊີບຈະຫາຍໄປໃນ 20 ປີຂ້າງໜ້າເນື່ອງຈາກນະວັດກຳເຕັກໂນໂລຢີ.

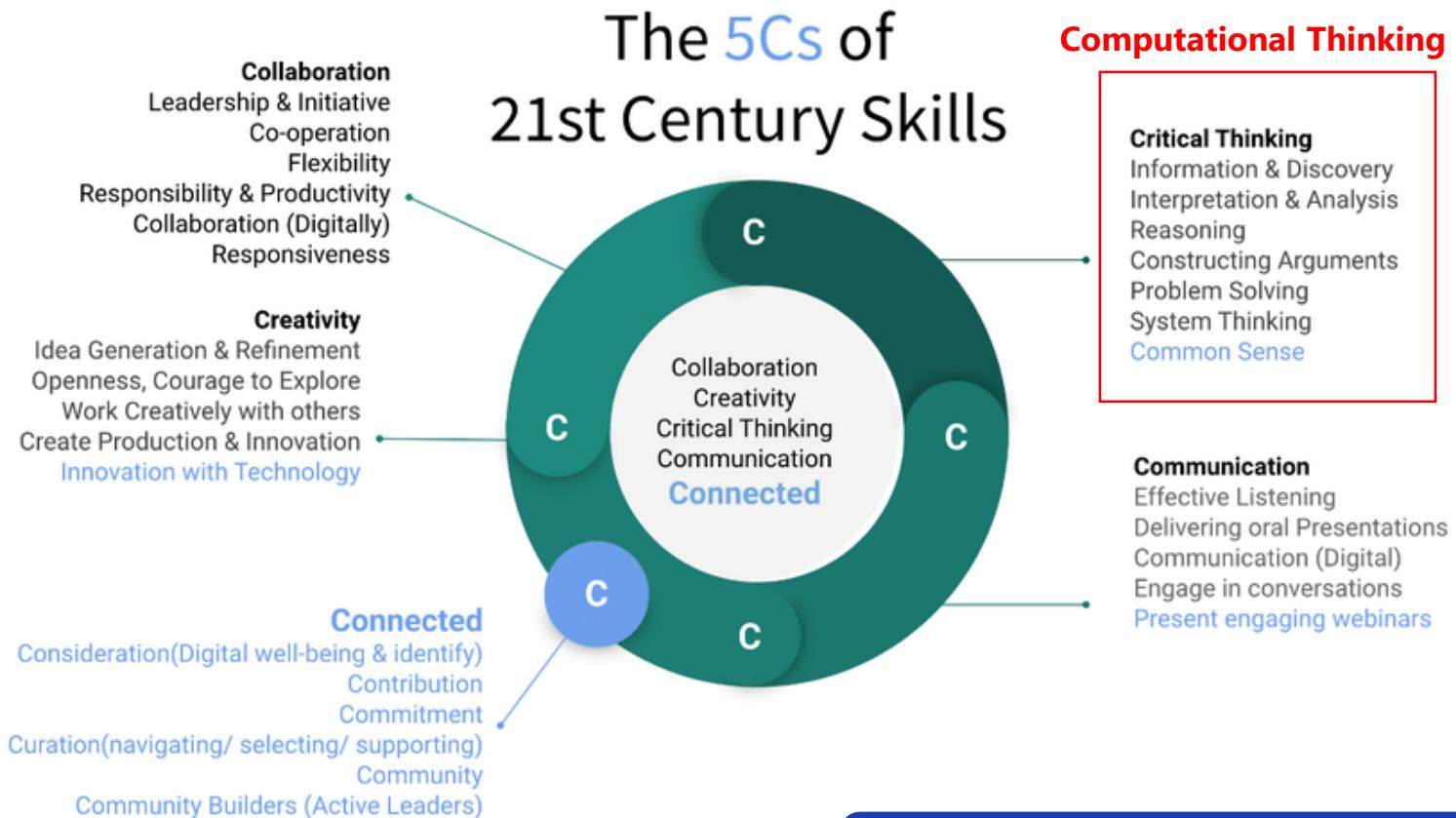


- ▶ ດ້ວຍການພັດທະນາຢ່າງວ່ອງໄວຂອງເຕັກໂນໂລຊີຄອມພິວເຕີ ແລະ ບັນຍາປະດິດ, ຄວາມສາມາດ້ານວິຊາຊີບທີ່ຕ້ອງການໃນຕະຫຼາດແຮງງານແມ່ນມີການປ່ຽນແປງ. ອີງຕາມການສຶກສາຂອງມະຫາວິທະຍາໄລ Oxford, ປະມານ 47% ຂອງອາຊີບຄາດວ່າຈະຖືກປ່ຽນແທນໄດຍເຄື່ອງຈັກ ຫຼື ຫາຍໄປພາຍໃນ 20 ປີຂ້າງໜ້າ.
- ▶ ດັ່ງນັ້ນ, ອາຊີບໃນອະນາຄົດຈະຂຶ້ນກັບຄວາມຄິດສ້າງສັນ, ຄວາມຮັບຜິດຊອບ ແລະ ຄວາມສາມາດໃນການຮຽນຮູ້ໄດຍຜ່ານປະສົບການ, ເຊິ່ງເປັນຄວາມສາມາດພິເສດຂອງມະນຸດ.
- ▶ ອີງການຮ່ວມມືທາງດ້ານເສດຖະກິດ ແລະ ການພັດທະນາ (OECD) ໄດ້ປະກາດຄວາມສາມາດຫຼັກ 4 ຢ່າງທີ່ຈໍາເປັນສໍາລັບຜູ້ມີຄວາມສາມາດໃນອະນາຄົດ.

<https://www.oecd.org/education/2030-project/teaching-and-learning/learning/>

## 1. Real world problem

### 1.2. The 5Cs of 21<sup>st</sup> Century Skills ຫັກສະ 5Cs ແຫ່ງສະຕະວັດທີ 21



<https://www.battelleforkids.org/networks/p21/frameworks-resources>

### 1. Real world problem

#### 1.3. ເຮົາຄວນກຽມຕົວແນວໃດສໍາລັບການຄິດແບບຄໍານວນ?



<https://www.barefootcomputing.org/>

"ການຄິດແບບຄໍານວນແມ່ນຂະບວນການຄິດທີ່ກ່ຽວຂ້ອງກັບການກຳນົດບັນຫາ ແລະ ວິທີແກ້ໄຂ, ເຊິ່ງວິທີແກ້ໄຂນັ້ນຖືກນຳສະເໜີໃນຮູບແບບທີ່ສາມາດປະຕິບັດໄດ້ຢ່າງມີປະສິດທິພາບ ດ້ວຍສັນຍາລັກການປະມວນຜົນຂໍ້ມູນ."

- Cuny, Snyder, Wing, 2010

- ຄອມພິວເຕີຄິດແນວໃດ? ແມ່ນການຄິດ ແລະ ແກ້ໄຂບັນຫາຄືກັບນັກວິທະຍາສາດຄອມພິວເຕີ.
- ມັນກ່ຽວຂ້ອງກັບແນວດ່ວຍການຄິດຕ່າງໆ, ເຊັ່ນ: ຂັ້ນຕອນວິທີ, ຮູບແບບ, ສິ່ງທີ່ເປັນນາມມະຫາ, ລັກສະນະທົ່ວໄປ, ການປະເມີນຜົນ ແລະ ການເຮັດວຽກແບບອັດຕະໂນມັດ .
- ການຂຽນ code ເປັນຄ່ອງມືທີ່ເໝາະສີມທີ່ສຸດສໍາລັບການຝຶກອົບຮົມ ແລະ ການຝຶກຊ້ອມການຄິດແບບຄໍານວນ ຫຼື ການຄິດແບບຄອມພິວເຕີ.
- ປັດຈຸບັນ, ຫ້າຍປະເທດໃນທົ່ວໄລກໄດ້ບັນຈຸການຂຽນໂປຣແກຣມຄອມພິວເຕີເຂົ້າໃນຫຼັກສູດເພື່ອພັດທະນາຊັບພະຍາກອນມະນຸດໃນອານາຄົດ
- ເຖິງແມ່ນວ່າການຂຽນໂຄດ (code) ບໍ່ແມ່ນຄ່ອງມືດຽວໃນການຮຽນຮູ້, ແຕ່ສໍາລັບການພັດທະນາການຄິດແບບຄໍານວນ, ມັນແມ່ນທີ່ໃນທາງເລືອກທີ່ດີທີ່ສູດ.

## 2. Solution

### 2.1. Thinking like a Computer คิดถึงกับคอมพิวเตอร์

\* เพื่อเบ่งคลิบวิดีโอ, เอา ကານูนຮອ ວາງເທິງ box ແລະ ບຸ່ມຫຼັນປະກິດຂຶ້ນ. ຕລິກາມນີ້ເພື່ອເບິ່ງ.



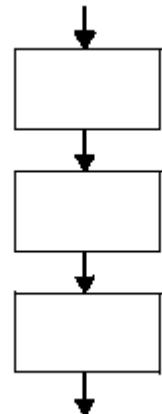
3 ວິທີໃນການຕັດສິນໃຈໄດ້ດີຂຶ້ນ - ໂດຍການຄິດຄັບຄອມພິວເຕີ | Tom Griffiths

## 2. Solution

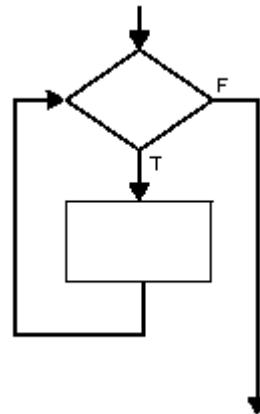
### 2.2. The 3 Structures of Programming (3 ໂຄງສ້າງຂອງການຂຽນໂປຣແກຣມ)

- | ເພື່ອຄິດ ແລະ ແກ້ໄຂບັນຫາຄືກັບນັກວິທະຍາສາດຄອມພິວເຕີ, ຈຳເປັນຕົ້ອງເຂົ້າໃຈພື້ນຖານວິທີການເຮັດວຽກຂອງການຂຽນໂປຣແກຣມ
  - | ໃນຂະນະທີ່ຮຽນ Python ຕະຫຼອດຫຼັກສູດນີ້, ຈະໄດ້ຮຽນຮູ້ການປັບປຸງປະສິດທິພາບໂດຍການຫຼຸດຜ່ອນວຽກງານທີ່ຊ້າກັນ, ປະຕິບັດການຕາມລຳດັບ ແລະ ການຕັດສິນໃຈໂດຍອີງໄສ່ເຖິງອື່ນໄຂທີ່ຊັດເຈນ.
- ວຽກທຳອິດແມ່ນນຳໃຊ້ຟັງຊັນ `print()` ເພື່ອຮຽນຮູ້ໂຄງສ້າງແບບລຳດັບ ແລະ ສ້າງໂປຣແກຣມໃຫ້ດຳເນີນການຕາມລຳດັບ.

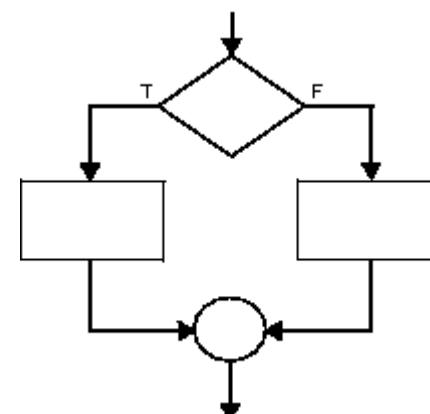
**Sequence**



**Iteration**



**Selection**

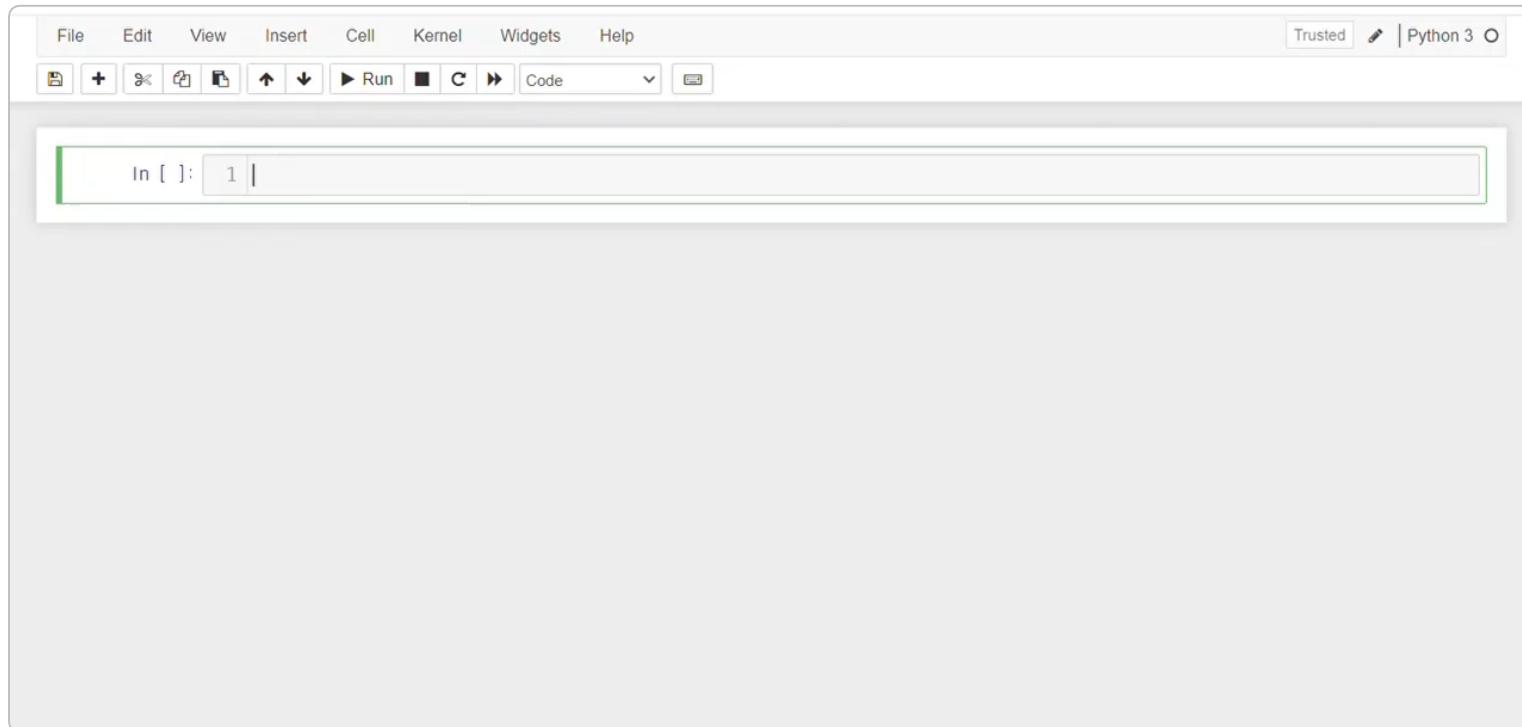


### 3. Mission

#### 3.1. Create a Program that Prints Your Resume ຂຽນໂປຣແກຣມພິມປະຫວັດຫຍໍ້

| ວິດີໂອນີສະແດງໃຫ້ເຫັນຜົນໄດ້ຮັບຂອງສິ່ງທີ່ຈະປະຕິບັດໃນ unit ນີ້. ເບິ່ງໂປຣແກຣມນີ້ ແລະ ຫາຄວາມຮູ້ທີ່ສາມາດໃຊ້ໄດ້ ແລະ ທ່ານຄວນຮຽນຮູ້ ແລະ ຄິດວິທີ ວາງແຜນເພື່ອສ້າງມັນຂຶ້ນມາ.

\* ເພື່ອເບິ່ງຄລິບວິດີໂອ, ເອົາ mouse ວາງເທິງ box ແລະ ບຸ່ມຫຼືນປະກິດຂຶ້ນ. ຄລິກມັນເພື່ອເບິ່ງ.



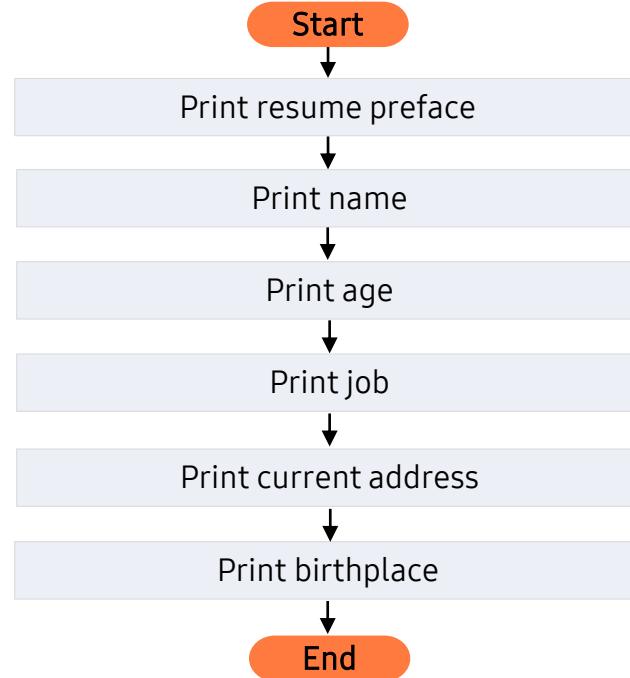
### 3. Mission

#### 3.2. Program Planning ການວາງແຜນໂປຣແກຣມ

##### Pseudocode

```
[1] # Self-introduction program
Start
[2] Print preface of the resume
[3] Print name
[4] Print age
[5] Print job
[6] Print the current address
[7] Print birth place
[9] End
```

##### Flowchart



### 3. Mission

#### 3.3. Resume Program: Final Codes (ໂປຣແກຣມພິມປະຫວັດຫຍໍ້: Codes ສຸດທ້າຍ)

| ນີ້ແມ່ນ code ສຸດທ້າຍຂອງໂປຣແກຣມທີ່ທ່ານຈະສ້າງໃນ unit ນີ້. ເປັນການກະຕຸນໃຫ້ນັກສຶກສາມີຈິນຕະນາການເຖິງຂະບວນການທີ່ຈໍາເປັນໂດຍຮູ້ຜົນໄດ້ຮັບສຸດທ້າຍ

```
1 print('Hello! I will introduce myself.')
2 print('Name: David Doe')
3 print('Age: 23')
4 print('Job: Data Scientist')
5 print('Address: Seoul, South Korea')
6 print('Place of Birth: Southern California, USA')
```

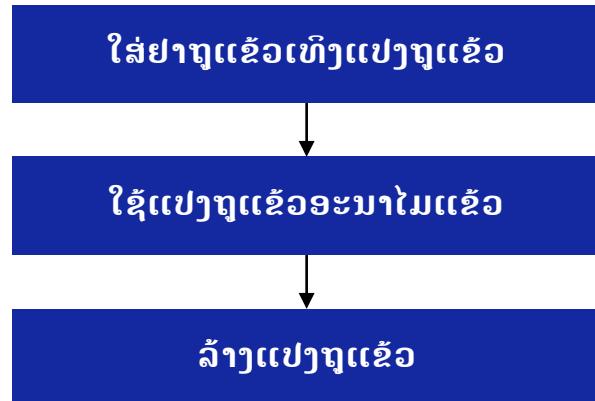
## | Key concept

## 1. ลำดับ (Sequence)

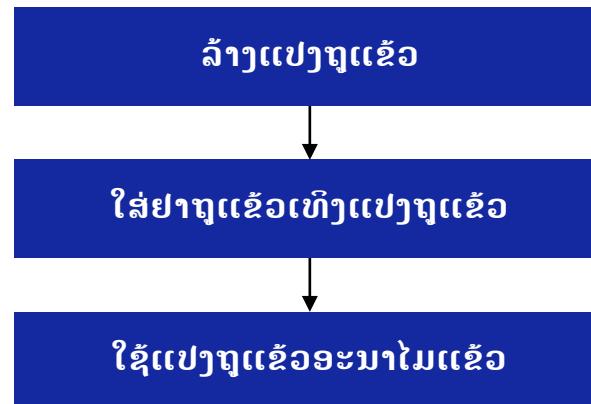
### 1.1. นิยาม และ ความสำคัญของลำดับ

- | Sequence ແມ່ນลำดับທີ່ຄອມພິວເຕີປະມວນຜົນ (executes) codes.
- | ເນື່ອງຈາກວ່າໂປຣແກຣມຮັດວຽກໄວໜ້າຍ, ຈຶ່ງອາດເບິ່ງຄືວ່າ codes ຖືກປະມວນຜົນພ້ອມງັກນ, ແຕ່ໃນຄວາມເປັນຈີງແລ້ວຄອມພິວເຕີດຳເນີນການຄໍາສັ່ງຕາມລຳດັບ.
- | ໂດຍທຳມະຊາດແລ້ວ, ເຖິງວ່າຈະແມ່ນຄໍາສັ່ງດຽວກັນແຕ່ຫາກລຳດັບປ່ຽນ ຜົນໄດ້ຮັບຂອງໂປຣແກຣມກໍ່ຈະແຕກຕ່າງຈາກເດີມຢ່າງແນ່ນອນ.
- | ເນື່ອຂຽນ codes ໃນພາສາ Python ຄວນພິຈາລະນາລຳດັບຄວາມສໍາຄັນ ແລະ ຈຸດປະສົງຂອງຄໍາສັ່ງກ່ອນທີ່ຈະຕັດສິນໃຈກຳນົດລຳດັບຂອງ codes.

ລຳດັບແມ່ນສໍາຄັນໃນຊີວິດປະຈຳວັນ



ປ່ຽນລຳດັບຈະສົ່ງຜົນຕໍ່ຜົນໄດ້ຮັບແນວໃດ?



## 2. คำศัพท์พื้นฐาน (Basic Terminology)

### 2.1. งานนำอั้ง glossary

- | Glossary เป็นเครื่องช่วยในการเรียนรู้. ตัวປະກິດມີคำศັບໃໝ່, ສິ່ງທີ່ສໍາຄັນແມ່ນຕ້ອງຈົດບັນທຶກ ແລະ ກວດສອບນິຍາມຂອງມັນ.
- | ການສອນຢູ່ໃນເວັບໄຊທາງການຂອງ Python ແມ່ນແນະນຳເຊັ່ນກັນ.

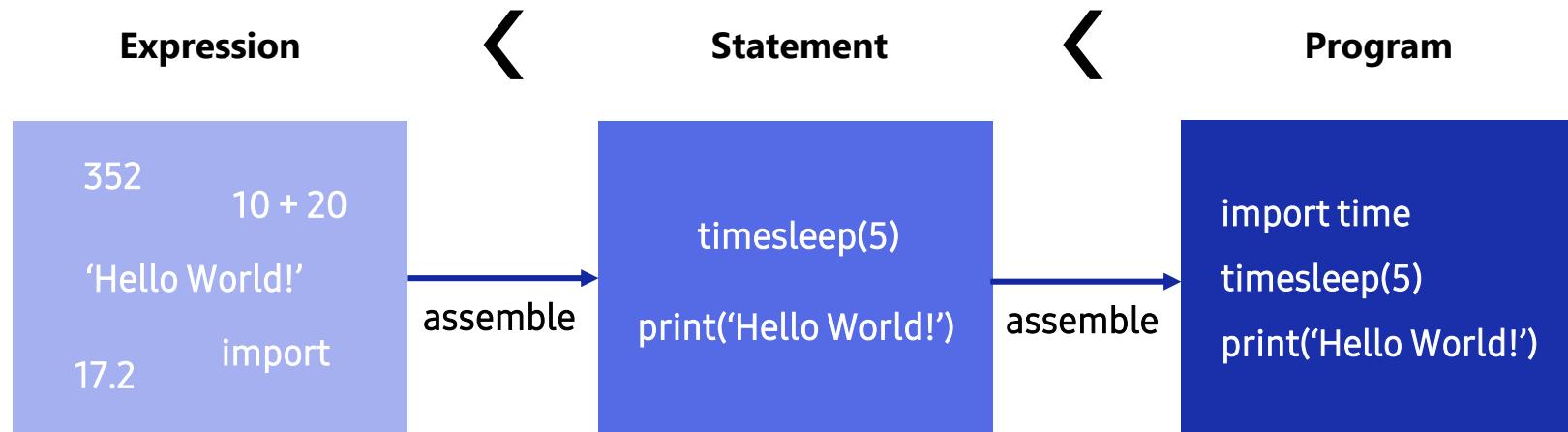
The screenshot shows the Python 3.9.7 documentation page. At the top, there's a navigation bar with links for 'Python »', 'English' (selected), '3.9.7' (selected), 'Documentation »', 'Quick search', 'Go', 'modules', and 'index'. On the left, a sidebar has sections for 'Download', 'Docs by version' (listing versions from 3.11 down to 2.7), and 'Other resources' (listing PEP Index, Beginner's Guide, Book List, and Audio/Visual Talks). The main content area features the title 'Python 3.9.7 documentation'. Below it is a welcome message: 'Welcome! This is the official documentation for Python 3.9.7.' It then lists several documentation sections: 'Parts of the documentation:', 'What's new in Python 3.9?' (with a link to 'or all "What's new" documents since 2.0'), 'Tutorial' (with a link to 'start here'), 'Library Reference' (with a link to 'keep this under your pillow'), 'Language Reference' (with a link to '...'), 'Installing Python Modules' (with a link to 'installing from the Python Package Index & other sources'), 'Distributing Python Modules' (with a link to 'publishing modules for installation by others'), and 'Extending and Embedding' (with a link to 'tutorial for C/C++ programmers').

<https://docs.python.org/3/>

## 2. คำศัพท์พื้นฐาน (Basic Terminology)

### 2.2. Expression, Statement และ Program

- I Expression ແມ່ນ code ທີ່ຢ່າຍດາຍປະກອບມີ ຕົວເລກ (numbers), ສູດ (formulas) ແລະ ຂໍຄວາມ (strings). ສ່ວນ Statement ປະກອບມີຫຼາຍກວ່າ 1 expressions ແລະ program ປະກອບດ້ວຍຫຼາຍກວ່າ 1 statement.



## 2. ຄຳສັບພື້ນຖານ (Basic Terminology )

## 2.3. Annotation

- | Annotation ແມ່ນປະໂຫຍກທີ່ໃຊ້ເພື່ອອະທິບາຍການເຮັດວຽກຂອງ code ໃນໂປຣແກຣມ. ຕົວແປພາສາ (interpreter) ຈະຂ້າມການດໍາເນີນງານຄໍາອະທິບາຍ.
  - | Annotation ແມ່ນມີຄວາມສໍາຄັນຫຼາຍໃນການຂຽນໂປຣແກຣມ. ເຊິ່ງຈະຊ່ວຍໃຫ້ນັກຂຽນໂປຣແກຣມເຂົ້າໃຈ code ກ່າຍຂຶ້ນ. ໂດຍສະເພາະ code ທີ່ມີຄວາມສັບຊ້ອນຫຼາຍ
  - | ມີສອງວິທີຫຼັກໃນການເພີ່ມ Annotation ໃນ Python. ວິທີທີ 1 ໃຊ້ຄື່ອງໝາຍ # ໃສ່ຈຸດເລີ່ມຕົ້ນຂອງປະໂຫຍກເພື່ອເປັນການອະທິບາຍ Annotation ໃນແຖວດຽງ. ດັ່ງລຸ່ມນີ້
  - | ເພີ່ມຄື່ອງໝາຍ # ໃສ່ບ່ອນເລີ່ມຕົ້ນຂອງປະໂຫຍກ, ປະໂຫຍກຢູ່ນັ້ນຈະກາຍເປັນຄໍາອະທິບາຍປະກອບ. ເຈົ້າຢັງບໍ່ໄດ້ຮູນ code ລຸ່ມນີ້ເຖີ່ງແຕ່ການອ່ານຄໍາອະທິບາຍຈະຊ່ວຍໃຫ້ເຈົ້າເຂົ້າໃຈໄດ້.

```
1 #ກ່າວນິດຕົວປັບນູ້ radius ມີຄ່າທີ່ 4  
2 radius = 4.0  
3  
4 #ສະແດງຄ່າຕົວປັບນູ້ radius, ຄ່ານີ້ທີ່ ແລະ ອ່າງອານຸຂອງຫຼົງມິນເທິງໝາດ  
5 print('Radius', radius)  
6 print('Area', 3.14 * radius * radius)      #ນຳໃຊ້ສູດຊອກຫາ ເນື້ອທີ່ຂອງຮູບວິທີມິນ  
7 print('Circumference', 2.0 * 3.14 * radius) #ນຳໃຊ້ສູດຊອກຫາ ອ່າງອານຸຂອງຮູບວິທີມິນ
```

Radius 4.0  
Area 50.24  
Circumference 25.13

## 2. ຄຳສັບພື້ນຖານ (Basic Terminology)

### 2.3. Annotation ຄໍາອະທິບາຍປະກອບ

| ວິທີທີ 2 ເພື່ອສ້າງຄໍາອະທິບາຍປະກອບຫຼາຍແຖວໃຫ້ເພີ່ມສັນຍາລັກ ““ ~ ~ ““ ຫຼື ““” ~ ~ ““” ສ່ວນທີ່ຢູ່ທາງໃນຂອງສັນຍາລັກຈະບໍ່ຖືກຄໍານວນ ດັ່ງສະແດງໃນຕົວຢ່າງລຸ່ມນີ້

1	” ”
2	You can create a multiple-line annotations using double quotation marks.
3	
4	It allows you to make multiple lines of comments.
5	” ”

1	““”
2	It allows you to make multiple lines of comments.
3	
4	It allows you to make multiple lines of comments.
5	““”

## 2. คำสับพื้นฐาน Basic Terminology

### 2.4. Reserved Word

I reserved words ຫຼື keywords ແມ່ນ ຄຳສັບທີ່ສະຫງວນໄວ້ໃນພາສາ python

Iເນື່ອງຈາກ keywords ສາມາດປະຕິບັດໜັ້ນທີ່ກໍານົດໄວ້ລ່ວງໜັ້ນໃນ Python, ບໍ່ສາມາດຖືກນຳໃຊ້ເປັນຊື່ identifiers ເຊັ່ນ ຊື່ຕົວປ່ຽນ ແລະ ຊື່ class. ຖ້າເຮົາໃຊ້ເປັນ identifier ພາລະບົດບາດຈະຫັບຊ້ອນກັບຄຳສັ່ງ ແລະ ບໍ່ສາມາດປະຕິບັດດໍາເນີນການໄດ້.

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

# Paper coding

ພະຍາຍາມເຊົ້າໃຈແນວຄວາມຄິດພື້ນຖານຢ່າງຄົບຖ້ວນກ່ອນທີ່ຈະຍ້າຍໄປສູ່ຂັ້ນຕອນຕໍ່ໄປ.

ການຂາດຄວາມເຊົ້າໃຈແນວຄວາມຄິດພື້ນຖານຈະເພີ່ມພາລະຂອງທ່ານໃນການຮຽນຮູ້ຫຼັກສູດນີ້, ເຊິ່ງອາດຈະເຮັດໃຫ້ທ່ານລຶ້ມເຫຼວໃນຫຼັກສູດ.

ຕອນນີ້ອາດຈະເປັນເລື່ອງຍາກ, ແຕ່ເພື່ອໃຫ້ສໍາເລັດຫຼັກສູດນີ້ ພວກເຮົາແນະນຳໃຫ້ທ່ານກຳຄວາມເຊົ້າໃຈແນວຄິດນີ້ຢ່າງທ່ອງແກ້  
ແລະ ກ້າວໄປສູ່ຂັ້ນຕອນຕໍ່ໄປ

**Q1.** Codes ຂອງ Python ຖືກດໍາເນີນການເປັນລຳດັບ. ຮູບຕາກະໄລຂະໜາດ  $3 \times 3$  ລຸ່ມນີ້. ເລີ່ມປະຕິບັດຈາກຕາກະໄລທີ 1, ແຕ່ລະຕາກະໄລຖືກທາສີຕາມຄໍາສັ່ງລຸ່ມນີ້. ຫຼື່ງໃນຫຼາຍ algorithms ທີ່ເປັນໄປໄດ້ ສະແດງດັ່ງລຸ່ມນີ້:

Start  
Here



<https://code.org/curriculum/course2/1/Teacher>

Move Right ຍ້າຍໄປທາງຂວາ,  
Fill-In Square ລະບາຍສີຕາກະໄລ,  
Move Right ຍ້າຍໄປທາງຂວາ,  
Move Down ຍ້າຍລົງລຸ່ມ  
Fill-In Square ລະບາຍສີຕາກະໄລ,  
Move Left ຍ້າຍໄປທາງຊ້າຍ,  
Move Left ຍ້າຍໄປທາງຊ້າຍ,  
Fill-In Square ລະບາຍສີຕາກະໄລ  
Move Down ຍ້າຍລົງລຸ່ມ,  
Move Right ຍ້າຍໄປທາງຂວາ,  
Fill-In Square ລະບາຍສີຕາກະໄລ,  
Move Right ຍ້າຍໄປທາງຂວາ

 Write the entire code and the expected output results in the note. ຂຽນ Code ທັງໝົດໃສ່ເຈັຍ ແລະ ຄາດເດືອນໄດ້ຮັບຈາກ Code ນັ້ນ.

**Q2.**

ໃຫ້ເລີ່ມຕົ້ນຈາກຕາກະໄລເບື້ອງຊ້າຍເທິງສຸດ, ຍ້າຍເທື່ອລະໜຶ່ງຕາກະໄລຕໍ່ຄັ້ງ ແລະ ລະບາຍສີຕາກະໄລດັ່ງຮູບລຸ່ມນີ້:

ໃຫ້ຄິດເຖິງວິທີດຳເນີນການທີ່ມີປະສິດທິພາບໝາຍຫີ່ສຸດ.

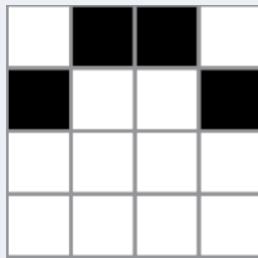


Image 1

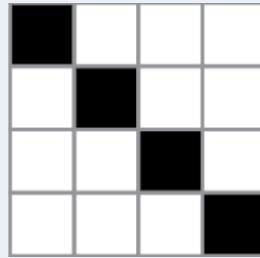


Image 2

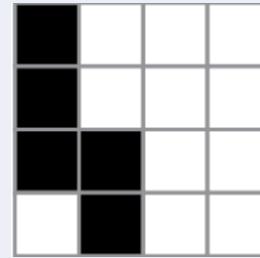


Image 3

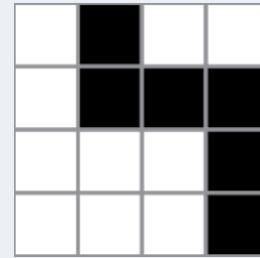


Image 4

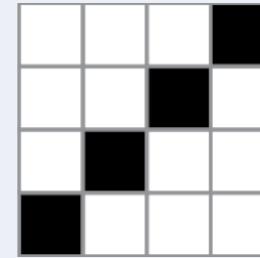


Image 5

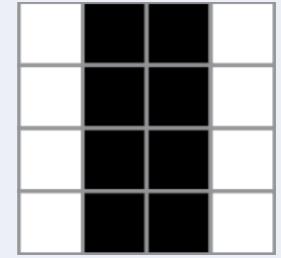


Image 6



Move One  
Square Right



Move One  
Square Left



Move One  
Square Up



Move One  
Square Down



Fill-In Square  
with Color

<https://code.org/curriculum/course2/1/Teacher>



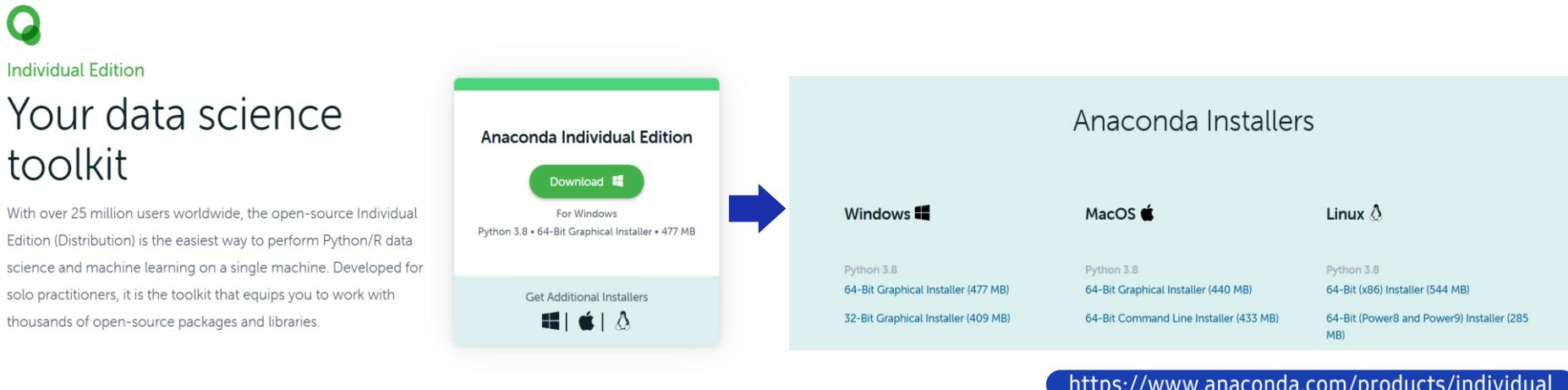
Write the entire code and the expected output results in the note. ຂຽນ Code ທັງໝົດໃສ່ເຈັຍ ແລະ ຄາດເດີເຜີນໄດ້ຮັບຈາກ Code ນັ້ນ.

| Let's code

## 1. ការព័ត៌មាន (Setting Up)

### 1.1. ការពិន្ទុកម្លាំង Anaconda

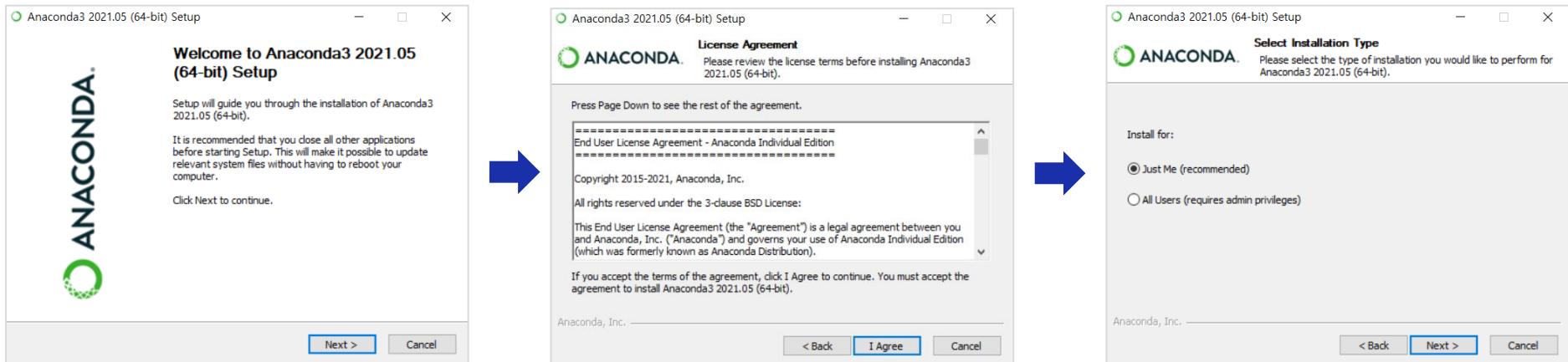
- | ធម្មរបស់ខ្លួនត្រូវក្រោមការពិន្ទុកម្លាំង Anaconda. ហាការណ៍នេះមិនមែនជាបញ្ហាសម្រាកទៅសំខាន់សំខាន់។ ដែលបានរាយការណ៍នេះមិនមែនជាបញ្ហាសម្រាកទៅសំខាន់សំខាន់។
- | កំណត់ថ្ងៃទី និងអេឡិចត្រូនិក ដើម្បីការពិន្ទុកម្លាំង Anaconda Individual Edition នៃកម្រិតនេះ។



# 1. ການຕັ້ງຄ່າ (Setting Up)

## 1.1. ການຕິດຕັ້ງ Anaconda

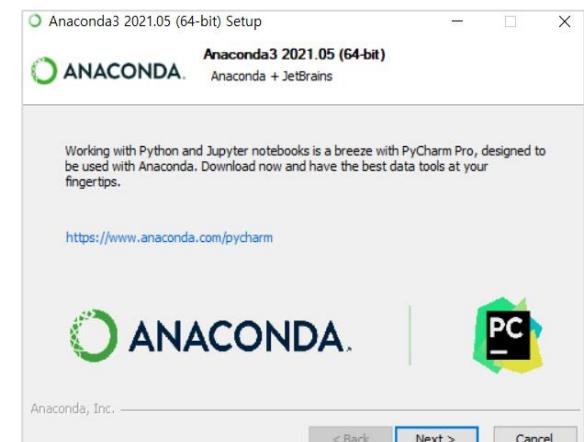
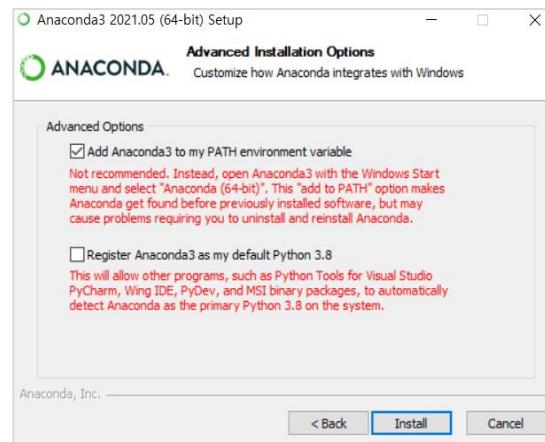
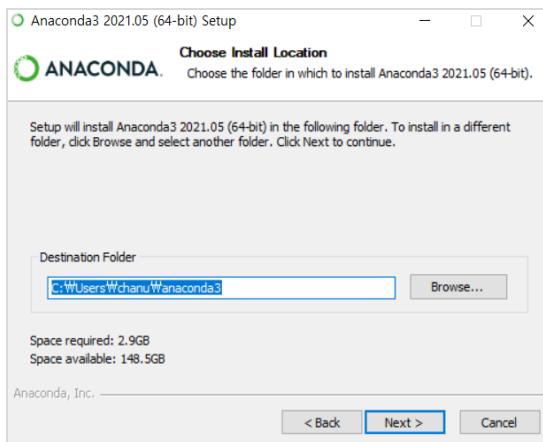
| ຫຼັງຈາກການດາວໂຫຼດ ເຮັດການຕິດຕັ້ງ anaconda, ປະຕິບັດຕາມຂັ້ນຕອນຂ້າງລຸ່ມນີ້ເພື່ອຕິດຕັ້ງໂປຣແກຣມ



# 1. ການຕັ້ງຄ່າ (Setting Up)

## 1.1. ການຕິດຕັ້ງ Anaconda

- ເວລາຕິດຕັ້ງ Anaconda, ຊື່ຜູ້ໃຊ້ຂອງຄອມພິວເຕີຕ້ອງຢັນພາສາອັງກິດ. ຖ້າບໍ່ດັ່ງນັ້ນ, ໂປຣແກຣມຈະບໍ່ສືບຕິດຕັ້ງ.
- ຖ້າເຄີຍຕິດຕັ້ງໂປຣແກຣມ Python ແຍກຕ່າງໆຫາກໄວ້ກ່ອນໜີ້ ໂປຣແກຣມນີ້ຈະທັບຊ່ອນກັບ Python.exe ຂອງ Anaconda. ໄປທີ່ Advanced Options ແລະ uncheck Add Anaconda3 to my PATH environment variable. ໃນກໍລະນີ້ສາມາດດຳເນີນການໂປຣແກຣມໂດຍການກົດໃສ່ Anaconda Prompt ໃນເມນຸ start.



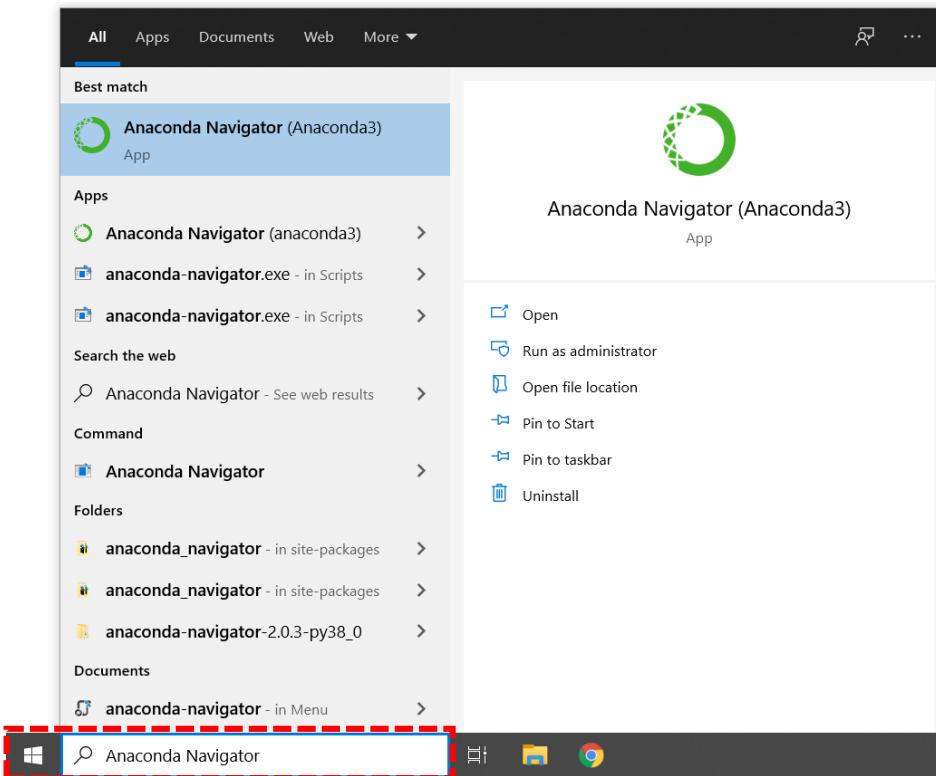
ວິທີຕິດຕັ້ງ <https://www.youtube.com/watch?v=ZEVDDBSuVXpk>

<https://www.youtube.com/watch?v=1Rmfq60fGT4&t=24s>

## 1. ການຕັ້ງຄ່າ (Setting Up)

### 1.2. ການຕິດຕັ້ງ Jupyter Notebook

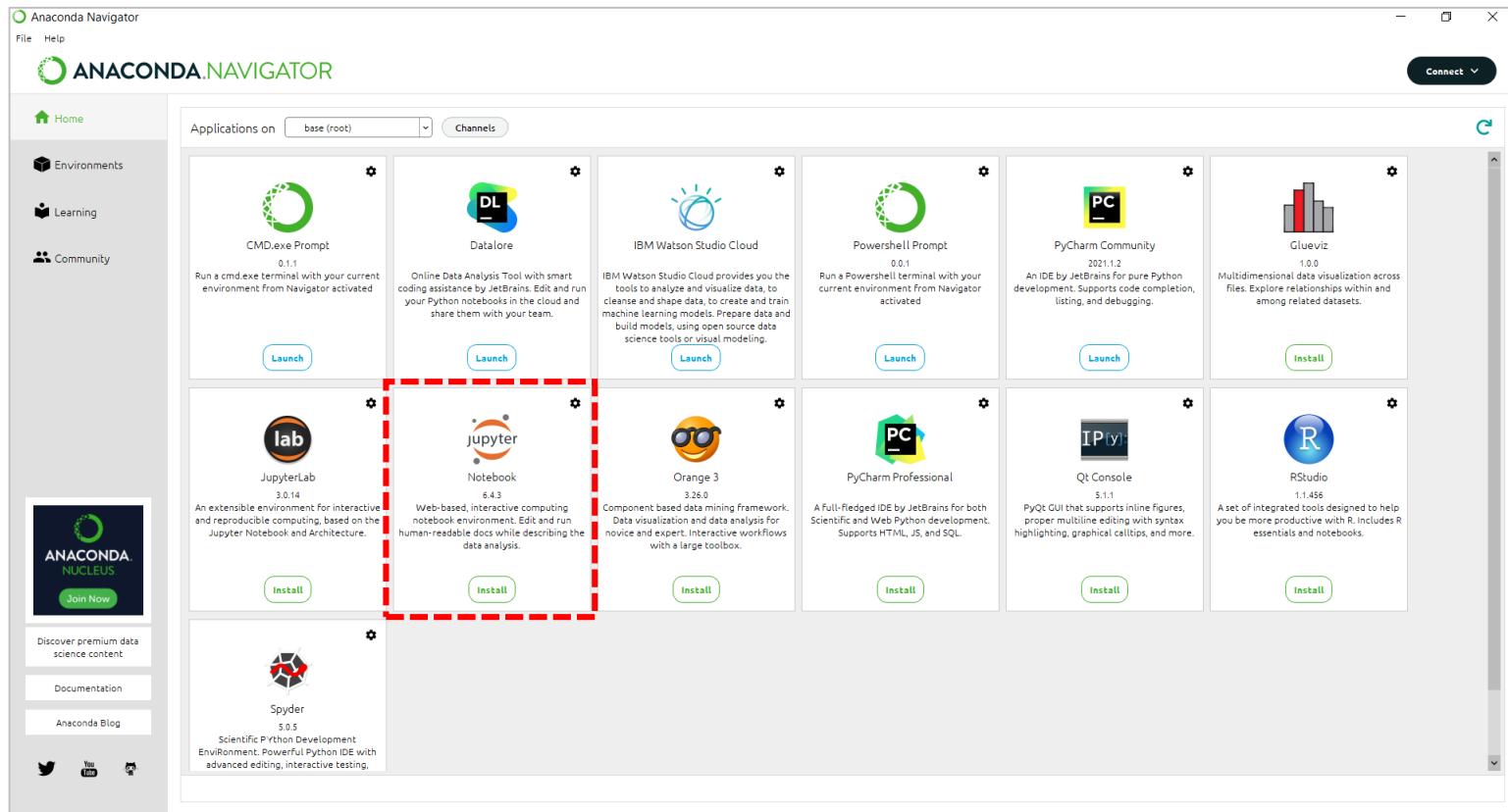
I ຫຼັງຈາກຕິດຕັ້ງ Anaconda, ລໍາດັບທັດໄປເຮົາຈະຕິດຕັ້ງ Jupyter Notebook. ຫໍາອິດ, ຂອກຫາ ແລະ ເປີດ Anaconda Navigator ໃນ Windows Start menu.



### 1. ການຕັ້ງຄ່າ (Setting Up)

#### 1.2. ການຕິດຕັ້ງ Jupyter Notebook

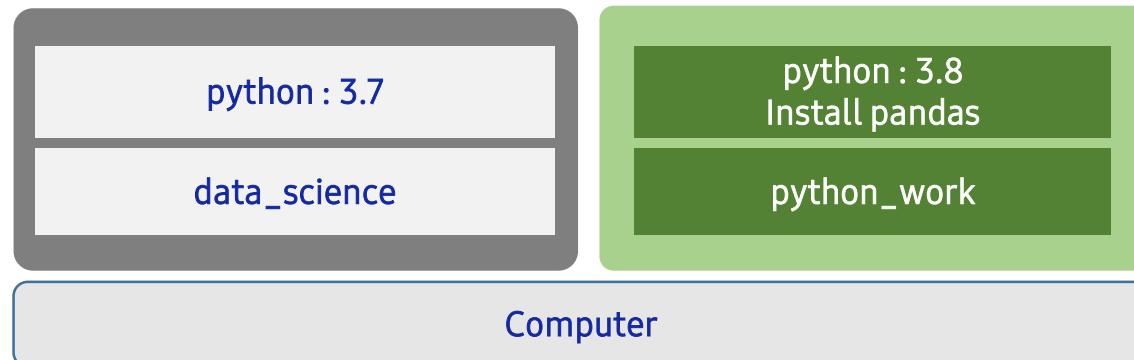
| ໃນ Anaconda Navigator ທີ່ສະແດງລຸ່ມນີ້, ກົດໃສ່ປຶ້ມ Notebook ການຕິດຕັ້ງແມ່ນສໍາລັດ



## 2. ການຕັ້ງຄ່າສະພາບແວດລ້ອມຈໍາລອງ (Virtual Environment Configuration)

### 2.1. ນິຍາມ ແລະ ຄວາມຈໍາເປັນຂອງສະພາບແວດລ້ອມສະໜີອນ

- | ເນື່ອຂຽນໂປຣແກຣມດ້ວຍ Python, ເຈົ້າອາດຈະໃຊ້ packages ພາຍນອກຕ່າງໆ. ຖ້າກໍາລັງ running ຫຼາຍໂປຣເຈັກ (project) ທີ່ໃຊ້ packages ພາຍນອກ ຫຼື ດໍາເນີນການ (running) ໂປຣເຈັກ (project) ໃໝ່ຕ້ອງການອັບເດດເວີຊັ້ນລ້າສຸດຂອງ Python ຫຼື ໃຊ້ packages ເວີຊັ້ນ ອື່ນ. ໃນກໍລະນີແບບນີ້, ການຈັດການໂມດຸນ ຫຼື ຄວບຄຸມເວີຊັ້ນຈະກາຍເປັນເລື່ອງຍາກ ແລະ ອາດເກີດບັນຫາການເຂົ້າກັນບໍ່ໄດ້ຂອງເວີຊັ້ນຕ່າງໆ.
- | ດ້ວຍເຫດນີ້ຈະໄດ້ຮຽນຮູ້ວິທີສ້າງສະພາບແວດລ້ອມຈໍາລອງ. ສະພາບແວດລ້ອມຈໍາລອງຂອງ Python ເປັນເຄື່ອງມືທີ່ສ້າງພື້ນທີ່ອິດສະຫຼະເພື່ອ ຈັດການໂປຣເຈັກ, ໂມດຸນ ແລະ ເວີຊັ້ນຕ່າງໆ. ອີກຄວາມໝາຍໜຶ່ງ, ແມ່ນມັນສ້າງພື້ນທີ່ເຮັດວຽກຕ່າງໆຢູ່ໃນເຄື່ອງແມ່ຂ່າຍ/ຄອມພິວເຕີ ໃນເຄື່ອງ ດຽວ.
- | ຕົວຢ່າງ, ໃນສະພາບແວດລ້ອມຈໍາລອງທີ່ເອັນວ່າ “data science” ທາງເບື້ອງຊ້າຍ, ອາດຈະໃຊ້ python 3.7. ໃນພື້ນທີ່ "python work" ຢູ່ເບື້ອງຂວາ, ອາດຈະໃຊ້ python 3.8 ແລະ pandas ຄືກັບກໍາລັງໃຊ້ຄອມພິວເຕີສອງເຕື່ອງ.



## 2. ການຕັ້ງຄ່າສະພາບແວດລ້ອມຈໍາລອງ (Virtual Environment Configuration)

### 2.2. ການສ້າງສະພາບແວດລ້ອມສະເໜີອນ Creating a Virtual Environment

- | ເພື່ອສ້າງສະພາບແວດລ້ອມຈໍາລອງ, ເປີດໃຊ້ Anaconda Prompt ຈາກເມນູ start ຂອງ Windows
- | ພິມຄໍາສັ່ງ `conda create -n python_work python=3.8`“ ໃນ prompt window ເພື່ອສ້າງພື້ນທີ່ເຮັດວຽກທີ່ເໝາະສົມສໍາລັບ Python 3.8. ສາມາດປັບແຕ່ງຊື່ພື້ນທີ່ເຮັດວຽກໄດ້ "python\_work".

Anaconda Prompt (Anaconda3)

```
(base) C:\Users\user>conda create -n python_work python=3.8
```

- | ເນື້ອການເຕືອນວ່າ "Proceeded ([y]/n)?" ປາກີດດັ່ງທີ່ສະແດງຢູ່ໃນຫນ້າຈຳຂ້າງລຸ່ມນີ້, ກົດແປ່ນພິມ y ເພື່ອດຳເນີນການ

```
python          pkgs/main/win-64::python-3.8.11-h6244533_1
setuptools     pkgs/main/win-64::setuptools-52.0.0-py38haa95532_0
sqlite         pkgs/main/win-64::sqlite-3.36.0-h2bbff1b_0
vc              pkgs/main/win-64::vc-14.2-h21ff451_1
vs2015_runtime pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
wheel          pkgs/main/noarch::wheel-0.37.0-pyhd3eb1b0_0
wincertstore   pkgs/main/win-64::wincertstore-0.2-py38_0
```

```
Proceed ([y]/n)? y
```

## 2. ການຕັ້ງຄ່າສະພາບແວດລ້ອມຈໍາລອງ Virtual Environment Configuration

### 2.2. ການສ້າງສະພາບແວດລ້ອມສະເໜີອນ Creating a Virtual Environment

- | ພິມຄໍາສັ່ງ "concrete-n data\_science python=3.7" ໃນ prompt window. ຊື່ຂອງພື້ນທີ່ເຮັດວຽກແມ່ນ "data\_science," ແລະ ພື້ນທີ່ເຮັດວຽກທີ່ເໝາະສີມກັບ Python ເວັ້ນ 3.7 ຖືກສ້າງຂຶ້ນ.

```
(base) C:\Users\user>conda create -n data_science python=3.7
```

- | ເນື່ອການເຕືອນວ່າ "Proceeded ([y]/n)?" ປາກີດດັ່ງທີ່ສະແດງຢູ່ໃນທຳນາຈຳຂ້າງລຸ່ມນີ້, ກີດແປ່ນພິມ y ເພື່ອດຳເນີນການ

```
python          pkgs/main/win-64::python-3.8.11-h6244533_1
setuptools     pkgs/main/win-64::setuptools-52.0.0-py38haa95532_0
sqlite         pkgs/main/win-64::sqlite-3.36.0-h2bbff1b_0
vc              pkgs/main/win-64::vc-14.2-h21ff451_1
vs2015_runtime pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
wheel          pkgs/main/noarch::wheel-0.37.0-pyhd3eb1b0_0
wincertstore   pkgs/main/win-64::wincertstore-0.2-py38_0
```

```
Proceed ([y]/n)? y
```

## 2. ການຕັ້ງຄ່າສະພາບແວດລ້ອມສະໜີອນ Virtual Environment Configuration

### 2.3. Anaconda Commands for Managing the Virtual Environment ຄໍາສັ່ງຂອງ Anaconda ສໍາລັບການຈັດການສະພາບແວດລ້ອມສະໜີອນ

| ຄໍາສັ່ງຕໍ່ໄປນີ້ສໍາລັບສະພາບແວດລ້ອມສະໜີອນ ອາດຈະຖືກນຳໃຊ້ໃນ Anaconda Prompt. ສະພາບແວດລ້ອມສະໜີອນສາມາດສ້າງ, ເປີດໃຊ້ງານ, ແລະ ຫຼັງຈາກນັ້ນປິດການນຳໃຊ້ ດ້ວຍຄໍາສັ່ງຕໍ່ໄປນີ້.

ການດໍາເນີນການ Execution	ຄໍາສັ່ງ Command
ເບິ່ງລາຍການສະພາບແວດລ້ອມສະໜີອນ	conda env list
ສ້າງສະພາບແວດລ້ອມສະໜີອນ	conda create -n [name of a virtual environment]
Clone ສະພາບແວດລ້ອມສະໜີອນ	conda create --clone [name of virtual environment to clone] -n [a new virtual environment]
ເປີດໃຊ້ສະພາບແວດລ້ອມສະໜີອນ	conda activate [name of a virtual environment]
ປິດນຳໃຊ້ສະພາບແວດລ້ອມສະໜີອນ	conda deactivate
ລຶບສະພາບແວດລ້ອມສະໜີອນ	conda remove --name [name of a virtual environment] --all

## 2. ການຕັ້ງຄ່າສະພາບແວດລ້ອມຈໍາລອງ (Virtual Environment Configuration)

### 2.3. Anaconda Commands for Managing the Virtual Environment

- | ບັນຄຳສັ່ງ "conda env list" ໃນ Anaconda Prompt ຈະສະແດງໃຫ້ທັນບັນຊີລາຍຊື່ຂອງສະພາບແວດລ້ອມຈໍາລອງຂອງ Python.
- | ຄ່າເລີ່ມຕົ້ນແມ່ນ base, ແລະ ສະພາບແວດລ້ອມຈໍາລອງ ທີ່ສ້າງຂຶ້ນກ່ອນໜີ້ນີ້, data\_science ແລະ python\_work, ຈະປາກິດຂຶ້ນ.

```
(base) C:\Users\user>conda env list
# conda environments:
#
base                  * C:\ProgramData\Anaconda3
C:\Users\user\.conda\envs\data_science
C:\Users\user\.conda\envs\python_work
```

## 2. ການຕັ້ງຄ່າສະພາບແວດລ້ອມຈໍາລອງ Virtual Environment Configuration

### 2.3. Anaconda Commands for Managing the Virtual Environment

- | ສາມາດຍ້າຍໄປພື້ນທີ່ເຮັດວຽກໃຫມ່ໂດຍໃຊ້ຄໍາສັ່ງ "conda activate data\_science". ຫຼຏາຕ່າງ prompt ຂອງ base ຈະປ່ຽນເປັນ (data\_science) ແລະ ຈະແຈ້ງໃຫ້ພື້ນທີ່ເຮັດວຽກໃຫມ່.

```
(base) C:\Users\user>conda activate data_science  
(data_science) C:\Users\user>
```

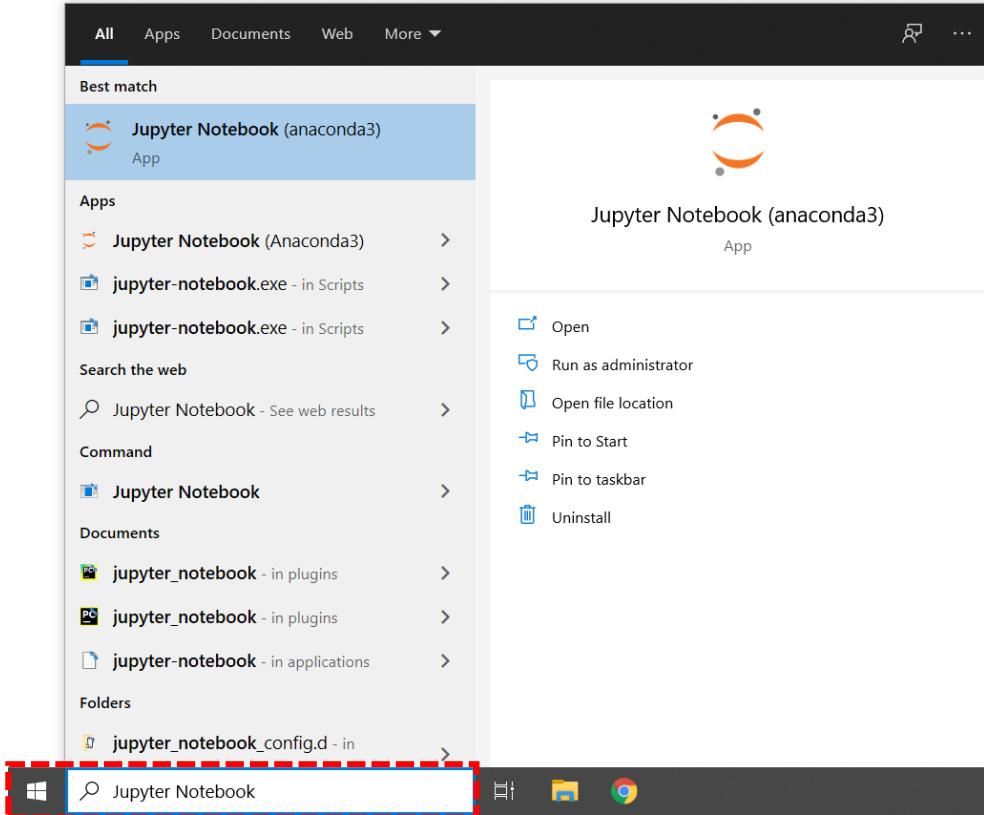
- | ລອງຍ້າຍ python\_work ໄປພື້ນທີ່ເຮັດວຽກອື່ນໂດຍໃຊ້ "conda activate python the work"

```
(data_science) C:\Users\user>conda activate python_work  
(python_work) C:\Users\user>
```

### 3. ວິທີການໃຊ້ Jupyter Notebook

#### 3.1. ການເປີດໃຊ້ Jupyter Notebook

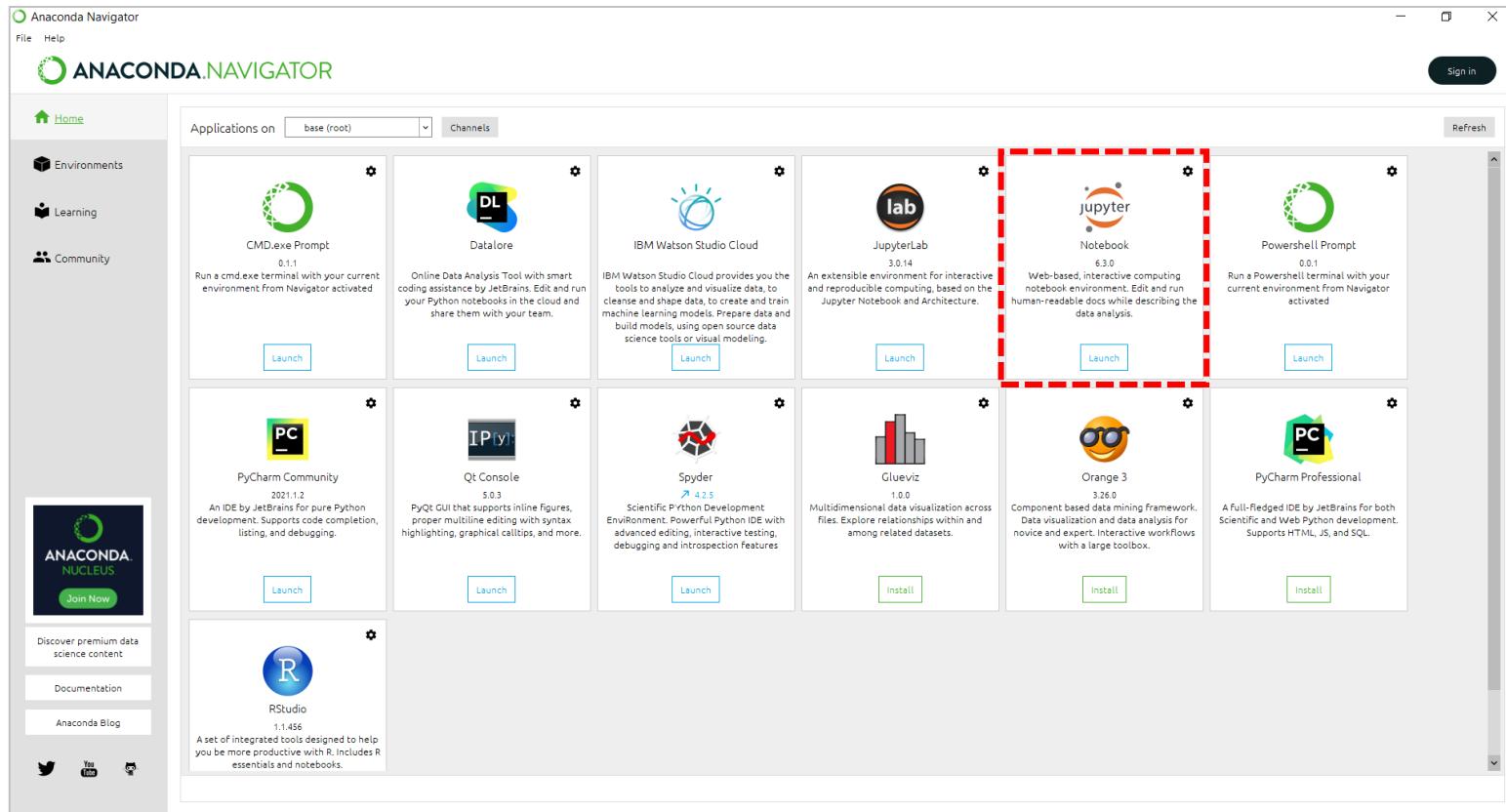
| ມີ 3 ວິທີຫຼັກທີ່ເຮົາສາມາດເປີດໃຊ້ Jupyter Notebook. ວິທີທີ່1, ໄປທີ່ Windows Start menu ແລະ ພຶມຊອກຫາ Jupyter Notebook.



### 3. ວິທີການໃຊ້ Jupyter Notebook

#### 3.1. ການເປີດໃຊ້ the Jupyter Notebook

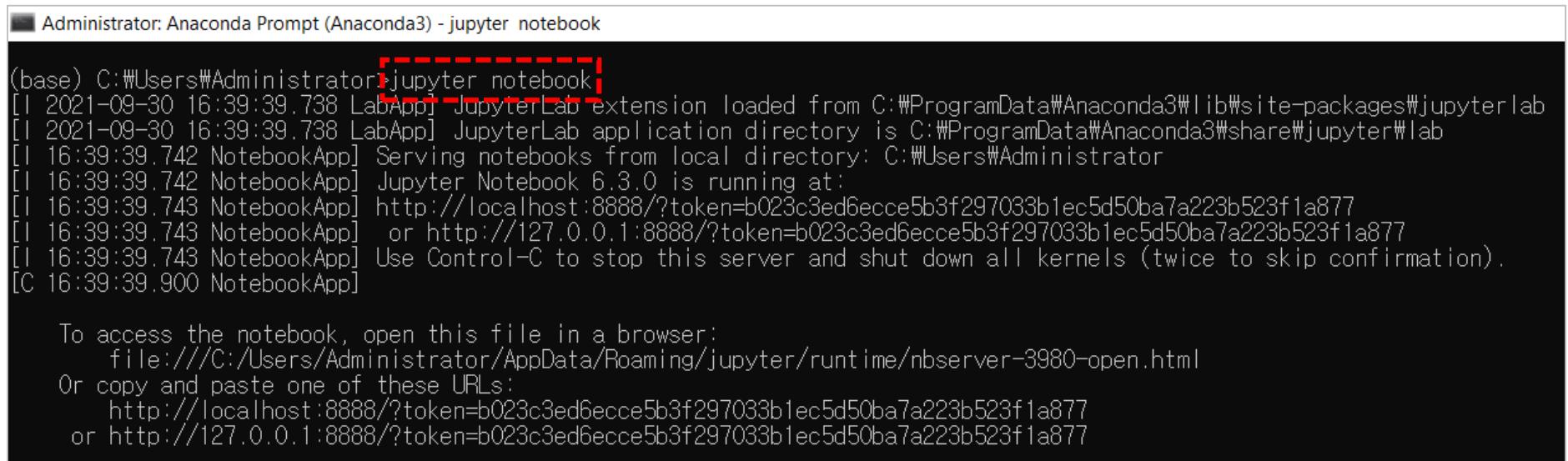
| ວິທີທີ2, ເລືອກ Jupiter Notebook ຈາກ Anaconda Navigator ແລະ ກົດຫຼືປົ້ມ Launch.



### 3. ວິທີການໃຊ້ Jupyter Notebook

#### 3.1. ການເປີດໃຊ້ Jupyter Notebook

| ວິທີທີ3, ພິມ "jupyter notebook" ໃນ Anaconda Prompt.



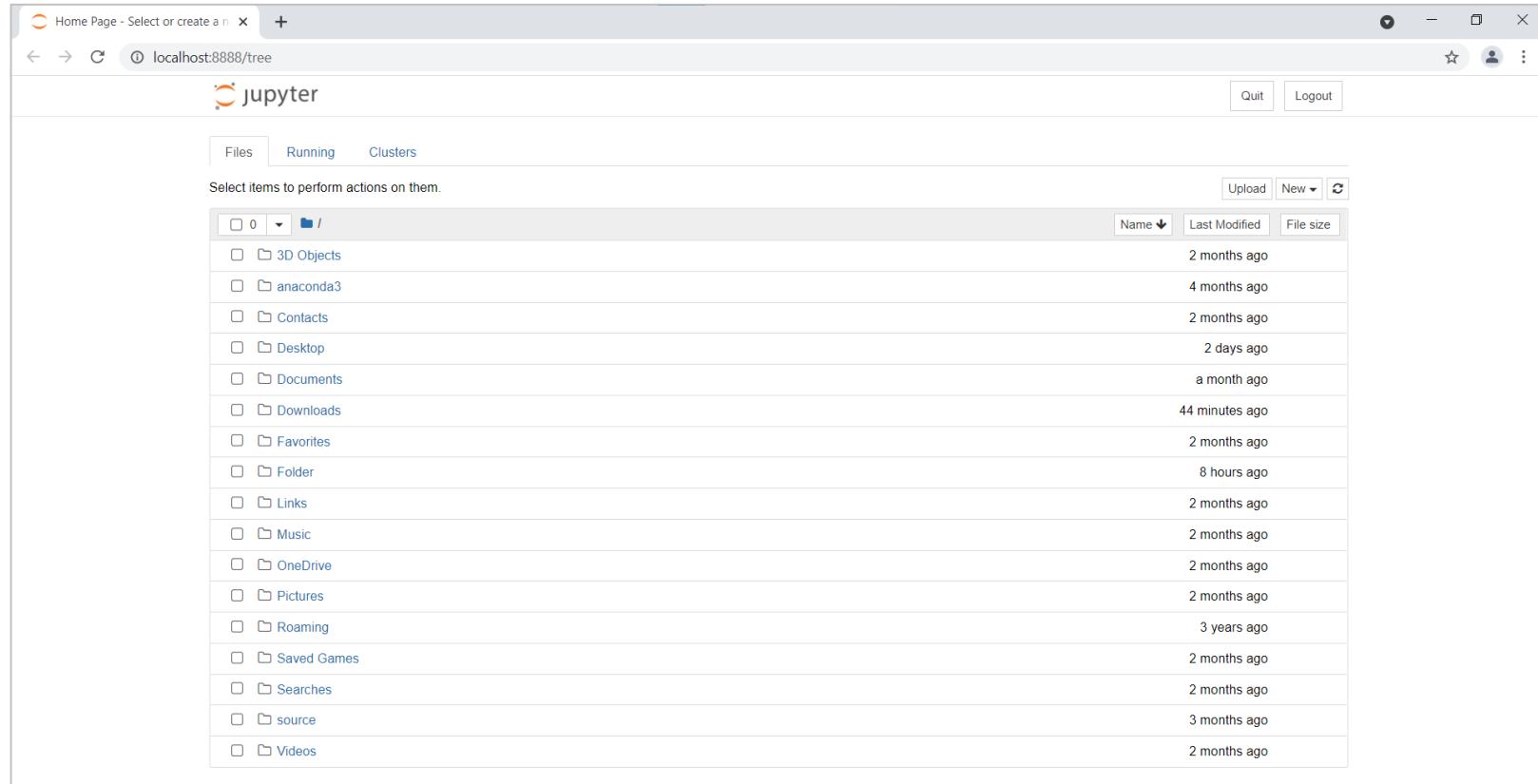
```
Administrator: Anaconda Prompt (Anaconda3) - jupyter notebook
(base) C:\Users\Administrator\jupyter notebook
[1 2021-09-30 16:39:39.738 LabApp] JupyterLab extension loaded from C:\ProgramData\Anaconda3\lib\site-packages\jupyterlab
[1 2021-09-30 16:39:39.738 LabApp] JupyterLab application directory is C:\ProgramData\Anaconda3\share\jupyter\lab
[1 16:39:39.742 NotebookApp] Serving notebooks from local directory: C:\Users\Administrator
[1 16:39:39.742 NotebookApp] Jupyter Notebook 6.3.0 is running at:
[1 16:39:39.743 NotebookApp] http://localhost:8888/?token=b023c3ed6ecce5b3f297033b1ec5d50ba7a223b523f1a877
[1 16:39:39.743 NotebookApp] or http://127.0.0.1:8888/?token=b023c3ed6ecce5b3f297033b1ec5d50ba7a223b523f1a877
[1 16:39:39.743 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 16:39:39.900 NotebookApp]

To access the notebook, open this file in a browser:
file:///C:/Users/Administrator/AppData/Roaming/jupyter/runtime/nbserver-3980-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=b023c3ed6ecce5b3f297033b1ec5d50ba7a223b523f1a877
or http://127.0.0.1:8888/?token=b023c3ed6ecce5b3f297033b1ec5d50ba7a223b523f1a877
```

### 3. ວິທີການໃຊ້ Jupyter Notebook

#### 3.2. UI ຂອງ Jupyter Notebook

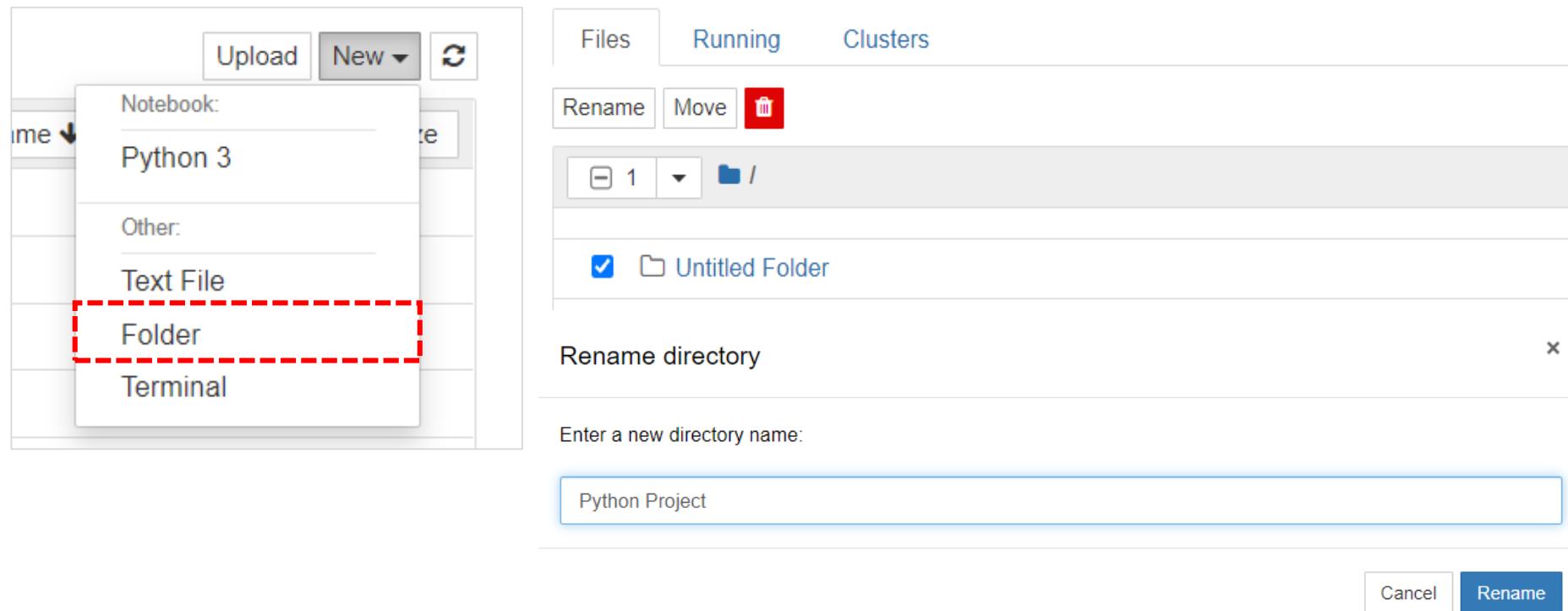
| ຈະເຫັນ screen ລຸ່ມນີ້ເມື່ອເຮົາເປີດ Jupyter Notebook.



### 3. ວິທີການໃຊ້ Jupyter Notebook

#### 3.2. UI ຂອງ Jupyter Notebook

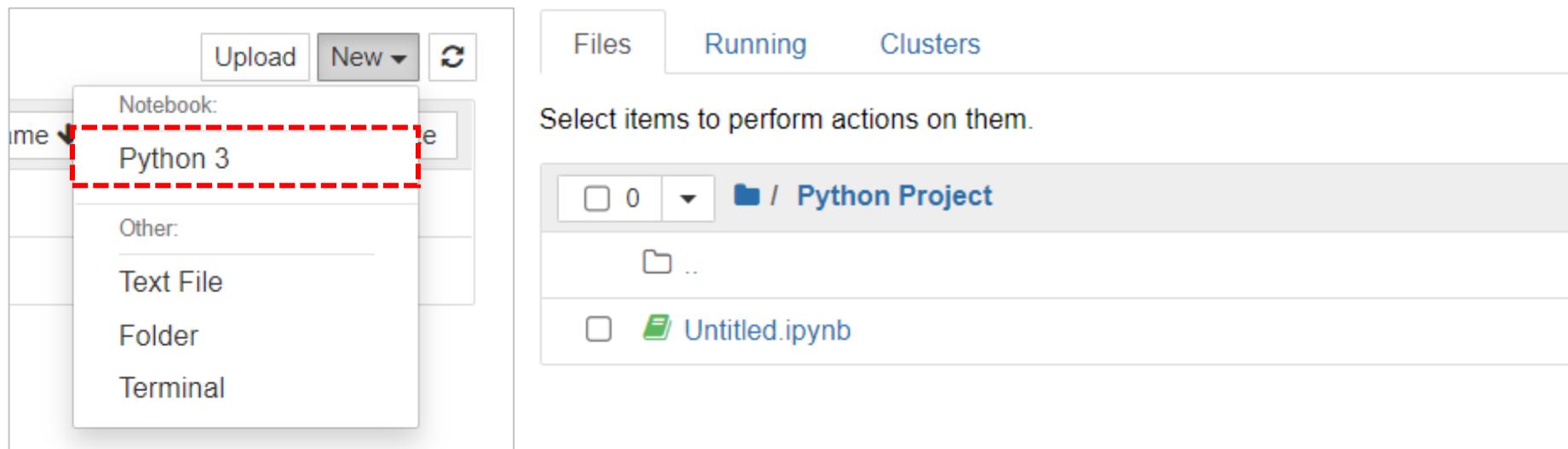
| ກົດປຸ່ມ New ຢູ່ແຈເທິງເບື້ອງຂວາເພື່ອໃຫ້ drop-down ເມນຸປະກິດອອກມາ. ກົດເລືອກ Folder ເພື່ອສ້າງ folder ໃໝ່. ເມື່ອ folder ໃໝ່ ທີ່ກໍສ້າງແລ້ວ, ກົດທີ່ check box. ແລ້ວປ່ຽນຊື່, ຍ້າຍ ຫຼື ລຶບ folder ນັ້ນ. ຕົວຢ່າງໃສ່ຊື່ folder ເປັນ “Python Project.”



## 3. ວິທີການໃຊ້ Jupyter Notebook

### 3.2. UI ຂອງ Jupyter Notebook

- | ເຂົ້າສູ່ folder ທີ່ສ້າງໃໝ່ຈາກນັ້ນກົດປຸ່ມ New ອີກຄັ້ງ. ກົດໃສ່ Python 3 ເພື່ອສ້າງ ipynb file ໃໝ່. ipynb ຫຍ້ມາຈາກ IPython Notebook, ແລະ IPython ແມ່ນສະພາບແວດລ້ອມການເຮັດວຽກແບບໂຕຕອບ.
- | ໃນຕົວຢ່າງລຸ່ມນີ້, ipynb file ໃໝ່ທີ່ເອັ້ນວ່າ Untitled ໄດ້ຖືກສ້າງຂຶ້ນພາຍໃຕ້ folder ຊື່ Python Project. ອີກທາງເລືອກໜຶ່ງ, ທ່ານສາມາດກົດທີ່ປຸ່ມ Upload ຢູ່ເຈຫິງເບື້ອງຂວາເພື່ອອັບໂລດ ipynb ພາຍຈາກ computer.



### 3. ວິທີການໃຊ້ Jupyter Notebook

#### 3.2. UI ຂອງ Jupyter Notebook

Icon ສີຂຽວປະກິດຂຶ້ນເມື່ອ ipynb Files ກໍາລັງເຮັດວຽກ. ຫາກຕ້ອງການຢຸດການເຮັດວຽກຂອງ file, ສາມາດ 1) ກິດເທິງ checkbox ແລະ ກິດປຸ່ມ Shutdown ຢູ່ກ້ອງ Files tab, ຫຼື 2) ກິດປຸ່ມ Shutdown ຢູ່ແຈລຸ່ມເບື້ອງຂວາຂອງ tab Running . ເຊິ່ງ tab Running ສະແດງບັນດາ files ທີ່ກໍາລັງເຮັດວຽກຢູ່ໃນປັດຈຸບັນ.

The screenshot shows the Jupyter Notebook interface with the 'Files' tab selected. At the top, there are three tabs: 'Files' (selected), 'Running', and 'Clusters'. Below the tabs is a toolbar with buttons for 'Duplicate', 'Shutdown' (highlighted in orange), 'View', 'Edit', and a trash icon. To the right of the toolbar are buttons for 'Upload', 'New', and a refresh icon. The main area displays a file tree under the path '/ Python Project'. A file named 'Untitled.ipynb' is selected, indicated by a checked checkbox next to its name. On the right side, there are columns for 'Name', 'Last Modified', and 'File size'. The file 'Untitled.ipynb' was last modified 'seconds ago' and has a size of '711 B'. There is also a 'Running' status indicator.

The screenshot shows the Jupyter Notebook interface with the 'Running' tab selected. At the top, there are three tabs: 'Files', 'Running' (selected), and 'Clusters'. Below the tabs is a section titled 'Currently running Jupyter processes'. It contains a 'Terminals' section with a message stating 'There are no terminals running.' and a 'Notebooks' section showing a list of running notebooks. The first notebook listed is 'Python Project/Untitled.ipynb', which is running in 'Python 3' and was last modified 'seconds ago'. There is a 'Shutdown' button next to the notebook entry.

## 3. ວິທີການໃຊ້ Jupyter Notebook

### 3.2. UI ຂອງ Jupyter Notebook

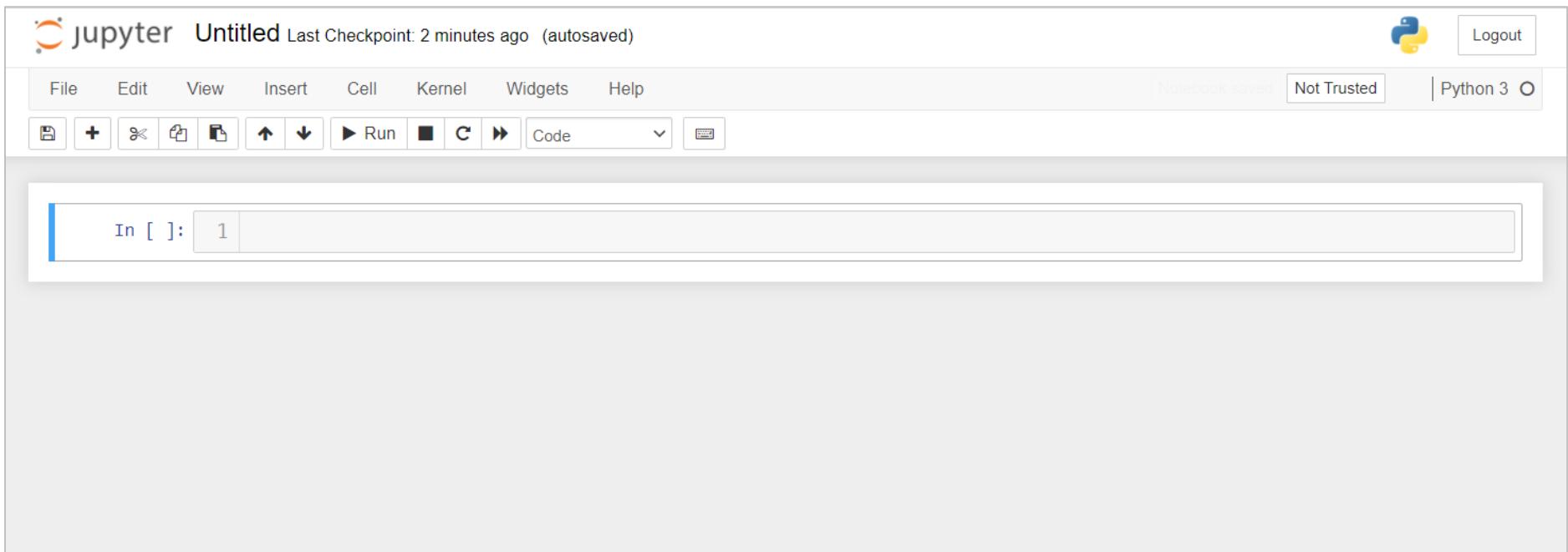
- | ເນື້ອກົດປຸ່ມ Shutdown, icon ຫຼັດຈາກ files ຈະປ່ຽນເປັນສີເຫຼີາ. tab Running ຈະສະແດງຂໍ້ຄວາມ “There are no notebooks running”.
- | ເນື້ອສັນສົດການດໍາເນີນການ file, ເຈົ້າສາມາດປ່ຽນຊື່, ຍ້າຍ ຫຼື Download ພາຍ ipynb ຈາກ Files tab ໄດ້



### 3. ວິທີການໃຊ້ Jupyter Notebook

#### 3.2. UI ຂອງ Jupyter Notebook

- | เมื่ອກິດໄຟ ipynb ຊື່ Untitled, ຈະເຫັນໜ້າຈຳທີ່ສາມາດຂຽນ code ໄດ້. ຂຽນ code ໃສ່ໃນ input window ຖັດຈາກ "In [ ]:" ແລະ ກິດປຸ່ມ Run ເພື່ອດຳເນີນການ. input window ມີເອັນວ່າ cell ແລະ ສາມາດໃຊ້ Edit tab ເພື່ອສໍາເນົາ (copy), ວາງ (paste), ລວມ(merge) ຫຼື ຍ້າຍ cells ຕ່າງໆໄດ້.
- | ເພື່ອເປີດນຳໃຊ້ໝາຍເລກແຖວຂອງ cell, ກິດ "Toggle Line Numbers" ເທິງ View tab.



## 4. ພັງຊັນ print( )

### 4.1. ພື້ນຖານ

print( ) code ແມ່ນຄໍາສັ່ງທີ່ສະແດງຄ່າຜົນໄດ້ຮັບໃນວົງເລັບ ( )

**print(“Hello World!”)**

ຄ່າຢູ່ໃນເຄື່ອງ “ ” ເຊັ່ນວ່າ **string** (ຂໍ້ຄວາມ)

ທັງຄື່ອງໝາຍ ‘ ’ ແລະ  
“ “ ສາມາດໃຊ້ໄດ້, ແຕ່ຕ້ອງເຂັ້ນ  
ຕົນ ແລະ ລົງທ້າຍດ້ວຍເຄື່ອງໝາຍ  
ດຽວກັນ



TIP

ບໍ່ຄືກັບພາສາໂປຣແກຣມອື່ນ, Python ຈະລົງແຖວໃໝ່ອັດຕະໂນມັດຫຼັງຈາກສະແດງຂໍ້ຄວາມ (ຜົນໄດ້ຮັບ) ດ້ວຍພັງຊັນ print().

## 4. ພັງຊັນ print( )

### 4.2. ການພິມຂໍ້ຄວາມ (String) ແລະ ຄ່າ (Value)

| ເພື່ອພິມຂໍ້ຄວາມນຳໃຊ້ພັງຊັນ print( ), ຈະຕ້ອງປິດທັງສອງດ້ານຂອງຂໍ້ຄວາມດ້ວຍເຄື່ອງໝາຍດຽວກັນ

```
1 print('Hello World!')  
2 print("Hello World!")
```

Hello World!  
Hello World!

| ເພື່ອພິມຕົວເລກ, ໃສ່ຕົວເລກທີ່ບໍ່ມີເຄື່ອງໝາຍວົງຢືມ (quotation mark), ຊະນິດຂໍ້ມູນຈະເປັນຕົວເລກ (integer)

```
1 print(10)  
2 print(7.5)
```

10  
7.5

| ຍັງສາມາດດຳເນີນການທາງຄະນິດສາດໃນພັງຊັນ print( ) ໄດ້. ການດຳເນີນການທາງຄະນິດສາດຂອງຂໍ້ມູນຊະນິດຕົວເລກຈະກ່າວເຖິງໃນ Unit 3.

```
1 print(10 + 7.5)  
2 print(10 - 7.5)
```

17.5  
2.5

### 4. ພັງຊັນ print( )

#### 4.3. ການເພີ່ມແຖວໃໝ່

- | Python ຈະລົງແຖວໃໝ່ອັດຕະໂນມັດຫຼັງຈາກພິມຂໍ້ຄວາມດ້ວຍພັງຊັນ print( ). ຫາກເປັນເຊັ່ນນັ້ນ, ສາມາດລົງແຖວປະໂຫຍກຕ່າງໆໃນພັງຊັນ print( ) ດຽວໄດ້ບໍ່?
- | ການໃຊ້ “ \n ” ພາຍໃນຂໍ້ຄວາມຈະເປັນການລົງແຖວໄປແຖວໃໝ່. “ \n ” ແມ່ນຕົວອັກສອນພິເສດໃຊ້ເພື່ອຄວບຄຸມການສະແດງຜົນຂໍ້ຄວາມ.

```
1 print('Hello\nWorld!')
```

Hello  
World!

## 4. ພັງຊັນ print( )

### 4.3. ການເພີ່ມແຖວໃໝ່

| ໃນພັງຊັນ print( ), ການເລີ່ມຕົ້ນ ແລະ ຈີບຂໍ້ຄວາມອາດຖືກຈັດກຸ່ມດ້ວຍເຄື່ອງໜາຍ ("") ຫຼື (""), ເຊິ່ງຈະສະແດງກຸ່ມຂອງຂໍ້ຄວາມທີ່ກວມເອົາ ການລົງແຖວພ້ອມ. ດັ່ງຕົວຢ່າງ code ແລະ ຜົນໄດ້ຮັບລຸ່ມນີ້:

```
1 print('''Hello  
2 World!''')
```

Hello  
World!

| ຢ່າງໃດກໍຕາມ, ຖ້າໃສ່ backslash (\) ໃນຕອນທ້າຍຂອງກຸ່ມຂໍ້ຄວາມທີ່ມີສາມວົງຍົມ, ຈະບໍ່ລົງແຖວ ແລະ ຂໍ້ຄວາມຢູ່ແຖວຖັດໄປຈະຖືກຕໍ່ ກາຍເປັນແຖວດຽວກັນ. ດັ່ງຕົວຢ່າງ code ແລະ ຜົນໄດ້ຮັບລຸ່ມນີ້:

```
1 print('''Hello\  
2 World!'''')
```

HelloWorld!

### 4. ພັງຊັນ print( )

#### 4.4. ຕົວດຳເນີນການ Comma(,) Additions(+), Multiplication(\*)

- | ຕົວດຳເນີນການ comma (,), addition (+) ຫຼື multiplication (\*) ສາມາດໃຊ້ພິມຂໍ້ຄວາມ ຫຼື ຄ່າຕົວເລກໄດ້ຫຼາກຫຼາຍ ແລະ ມີປະສິດທິພາບຫຼາຍຂຶ້ນ.
- | ເນື້ອໃຊ້ຕົວດຳເນີນການ (,) ຢູ່ລະຫວ່າງຄ່າທີ1 ແລະ ຄ່າທີ 2, ຕົວດຳເນີນການຈະເພີ່ມຊ່ອງຫວ່າງລະຫວ່າງສອງຄ່າ ແລ້ວລົງແຕວໃໝ່. ສາມາດພິມສອງ ຫຼື ຫຼາຍຈໍານວນຂຶ້ນໄປໂດຍບໍ່ກໍານົດປະເພດຂໍ້ມູນ ດັ່ງສະແດງໃນ code ແລະ ຜົນໄດ້ຮັບລຸ່ມນີ້:

```
1 print('Hello', 'World!')  
2 print(10, 20)
```

Hello World!

10 20

- | ອາດຈະລວມຂໍ້ຄວາມ ແລະ ຕົວເລກດ້ວຍຕົວດຳເນີນການ comma (,) ດັ່ງສະແດງໃນ code ແລະ ຜົນໄດ້ຮັບລຸ່ມນີ້:

```
1 print('Hello', 10)
```

Hello 10

## 4. ฟังชัน print( )

### 4.4. ติอคำเมินกาน Comma(,) Additions(+), Multiplication(\*)

| เมื่อข้อความที่กลมเข้ากับติอคำเมินกาน addition (+), ข้อความที่ 2 ทີກພິມຕໍ່ຫຼັງຂໍ້ຄວາມທີ 1 ໂດຍບໍ່ມີຊ່ອງຫວ່າງ ແລ້ວລົງແຖວໃໝ່. ເມື່ອຕົວດຳເນີນ ການ addition (+) ທີກໃຊ້ກັບຄ່າຕົວເລກ, ມັນຈະພິມຜົນຂອງການບວກຄ່າຕົວເລກຫຼັງຈາກນັ້ນ ລົງແຖວໃໝ່.

```
1 print('Hello' + 'World!')  
2 print(10 + 20)
```

HelloWorld!  
30

✿ เมื่อໃຊ້ຕົວດຳເນີນການ addition (+), ບໍ່ສາມາດລວມຂໍ້ຄວາມ ແລະ ຕົວເລກເຂົ້າກັນໄດ້. ເມື່ອລວມຂໍ້ຄວາມ ແລະ ຕົວເລກເຂົ້າກັນດຳເນີນການ addition (+), Type Error ຈະເກີດຂຶ້ນ.

```
1 print('Hello' + 10)
```

```
TypeError  
<ipython-input-3-62277d5ee2c1> in <module>  
----> 1 print('Hello' + 10)
```

```
TypeError: can only concatenate str (not "int") to str
```

## 4. ฟังก์ชัน print( )

### 4.4. ตัวดำเนินการ Comma(,), Additions(+), Multiplication(\*)

| เมื่อข้อความ และ เลขจำนวนนับที่กล่าวมีเข้ากันโดยใช้ตัวดำเนินการ multiplication (\*), ข้อความจะถูกพิมพ์มาต่อๆ กันโดยที่ไม่มีเครื่องหมายแบ่ง แต่จะมีเครื่องหมาย加 (+) หรือ乖 (\*) อยู่ระหว่างตัวอักษร เช่น คำว่า Hello World! จะถูกพิมพ์ออกมายังหน้าจอเป็น HelloWorld!

```
1 print('Hello' * 3)
2 print(10 * 20)
```

```
HelloHelloHello
200
```

 เมื่อข้อความ และ ข้อความที่กล่าวมีเข้ากันด้วย ตัวดำเนินการ multiplication (\*), Type Error จะเกิดขึ้น.

```
1 print('Hello' * 'World!')
```

```
TypeError                                 Traceback (most recent call last)
<ipython-input-6-b6a7f39bdc2f> in <module>
----> 1 print('Hello' * 'World!')
```

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

## 4. ផ្សេងៗ print( )

### 4.5. End និង Split Parameters

- | เมื่់វិភាគកម្មពិបាបផ្សេងៗ print() នៃ code, ការលើរពេល បែងពេទ្យឱ្យកើតឡើង. តើយើងឲ្យការណា ដឹងពីការណាដឹងជាបន្ទាល់ពីការណាដឹងនៅក្នុងការណាដឹង។
- | ការណាដឹងពិបាប 'Hello' និង 'World!' នូវការបែងពេទ្យឱ្យកើតឡើង។

```
1 print('Hello', end = '')  
2 print('World!')
```

HelloWorld!

```
1 print('Hello', end = ',')  
2 print('World!')
```

Hello,World!

## 4. ផ្សេងៗនៃ print( )

### 4.5. End និង Split Parameters

- | កែវានៅក្បែងពីការប្រើប្រាស់លក្ខណៈជាការប្រើប្រាស់ការពិនិត្យការងារ។ ម៉ោងខ្លួនគឺជាការសរសៃការងារដែលបានបញ្ជាក់ដោយការប្រើប្រាស់លក្ខណៈជាការប្រើប្រាស់ការងារ។
- | កែវានៅក្បែងពីការប្រើប្រាស់ការងារដែលបានបញ្ជាក់ដោយការប្រើប្រាស់លក្ខណៈជាការប្រើប្រាស់ការងារ។

```
1 print('Hello', 'World!', sep = '')
```

HelloWorld!

```
1 print('Hello', 'World!', sep = ',')
```

Hello,World!

## 4. ພັງຊັນ print( )

### 4.6. ການສ້າງຊີວະປະຫວັດຫຍໍ່ Creating a Resume

| ລຸ່ມນີ້ແມ່ນຕົວຢ່າງ code ຂອງກິດຈະກ່າມື້ນີ້. ລອງຊ້ອມຂຽນ code ສະແດງປະຫວັດຫຍໍ່ນຳໃຊ້ຕົວປະຕິບັດການທີ່ຮຽນໃນບົດນີ້; comma (+), multiplication (\*) , end ແລະ split parameter.

```
1 print('Hello! I will introduce myself.')
2 print('Name: David Doe')
3 print('Age: 23')
4 print('Job: Data Scientist')
5 print('Address: Seoul, South Korea')
6 print('Place of Birth: Southern California, USA')
```

| ເຊັ່ນດຽວກັນ ລອງໃຊ້ເຄື່ອງໝາຍ asterisk (\*) ຫຼື minus (-) ເພື່ອຕົບແຕ່ງຊີວະປະຫວັດຫຍໍ່

```
1 print('*****')
2 print('-----')
```

## 5. ວິທີການແບບ Python ( Pythonic way)

### 5.1. ເປັນຫຍັງການຂຽນ code ແບບ Python ຈຶ່ງສໍາຄັນ?

- | ມັນເປັນສິ່ງສໍາຄັນທີ່ຈະສ້າງຮູບແບບການຂຽນ code ທີ່ດີ ແລະ ເຝັກສັນຄີຍຕັ້ງແຕ່ເລີ່ມຕົ້ນ.
- | ຈຸດປະສົງຫຼັກຂອງ Pythonic Coding ແມ່ນການຂຽນ code ທີ່ຊັດເຈນ ແລະ ກະທັດຮັດໂດຍໃຊ້ຄຸນລັກສະນະທີ່ແຕກຕ່າງຈາກພາສາໂປຣແກຣມອື່ນໆ.
- | ການນຳໃຊ້ conventions, ການຂຽນ code ພາສາ Python ທີ່ໃຊ້ໂດຍກຸ່ມນັກພັດທະນາຍັງຊ່ວຍໃນການຮ່ວມມືກັບຜູ້ພັດທະນາອື່ນໆ.
- | ມີຫຼາຍ conventions ສໍາລັບວິທີການ Pythonic, ແຕ່ຈະເນັ້ນໃສ່ສິ່ງທີ່ຈໍາເປັນສໍາລັບຜູ້ເລີ່ມຕົ້ນ.

#### PEP 8 -- Style Guide for Python Code

PEP:	8
Title:	Style Guide for Python Code
Author:	Guido van Rossum <guido at python.org>, Barry Warsaw <barry at python.org>, Nick Coghan <ncoghlan at gmail.com>
Status:	Active
Type:	Process
Created:	05-Jul-2001
Post-History:	05-Jul-2001, 01-Aug-2013

- ▶ ມີສອງຄຸ່ມຫຼັກ ກ່ຽວກັບຮູບແບບການຂຽນ Python
- ▶ PEP8 of Python's official community ແລະ Google's Python Style Guide.
- ▶ unit ມີໃຊ້ PEP8 guide.

<https://www.python.org/dev/peps/pep-0008/>

## 5. ວິທີການແບບ Python (Pythonic way)

### 5.2. ການຈັດວາງ Code (Code lay out)

| Python ອະນຸຍາດໃຫ້ຂຽນຫຼາຍປະໂຫຍກໃນແຖວດຽວໄດ້ໃຊ້ semicolons ດັ່ງທີ່ສະແດງຂ້າງລຸ່ມນີ້. ການຂຽນ code ລັກສະນະນີ້, ບໍ່ສາມາດອ່ານໄດ້ດີ.

```
1 a = 10; a += 10; print(a) # Bad case
```

20

| ເນື່ອຂຽນ code ດ້ວຍ Python, ແນະນຳໃຫ້ໃຊ້ຫນຶ່ງປະໂຫຍກໃນເຕັ້ນລະແຖວດັ່ງທີ່ສະແດງຂ້າງລຸ່ມນີ້.

```
1 a = 10      # Good case
2 a += 10
3 print(a)
```

20

## 5. ວິທີການແບບ Python (Pythonic way)

### 5.2. ການຈັດອາງ Code (Code lay out)

| Code ທີ່ຍາວງດັ່ງລຸ່ມນີ້ແມ່ນຍາກຫຼາຍໃນການອ່ານ.

```
1 if width == 0 and height == 0 and color == 'red' and emphasis == 'strong' or highlight > 100:  
2     print('Sorry, you lose.')
```

| ເນື້ອຂຽນ code ດ້ວຍ Python, ຫ້າປະໂຫຍກໄດ້ນຶ່ງຍາວເກີນໄປ, ໃຫ້ລອງຕັດແຖວໂດຍໃຊ້ backslash(\) ເພື່ອລົງແຖວ.

```
1 if width == 0 and height == 0 and color == 'red' and \  
2     emphasis == 'strong' or highlight > 100:  
3     print('Sorry, you lose.')
```

| ສຸດທ້າຍ, ຕົວປະຕິບັດການ (operators) ແລະ ຕົວຖືກດຳເນີນການ (operands) ຈະຕ້ອງມີຊ່ອງຫວ່າງລະຫວ່າງກັນ. ເນື້ອສະແດງລາຍການອີງປະກອບ, ໄສ່ຊ່ອງຫວ່າງຫຼັງເຄື່ອງໝາຍຈຸດ.

```
1 a=[1,2,3]      # Bad
```

```
1 a = [1, 2, 3]  # Good
```

## 5. វិທីការណ៍របៀប Python (Pythonic way)

### 5.3. ហ្វេកការណ៍ពីរខ្លួន (Naming convention)

- | ខ្លួនឯងនៅក្នុងការសរសៃរបៀបការងារ, តើម្លែង ពិវប្បធម៌ (variables), ផែនការ (functions), classes និងកម្រិត. ហ្វេកនេះអំពីថានឹងគោរពយោងទាំងអស់ជានឹងគោរពយោងទាំងអស់។
- | នៅក្នុងការសរសៃរបៀបការងារ, តើម្លែង ពិវប្បធម៌ និងផែនការ ត្រូវបានរំភ័យដោយរួចរាល់តាមរូបរាយលើខ្លួន។
- | នៅក្នុងការសរសៃរបៀប Python, តើម្លែង ពិវប្បធម៌ និងផែនការ ត្រូវបានរំភ័យដោយរួចរាល់តាមរូបរាយលើខ្លួន។

រូបរាយ (Style)	ការពិពណ៌នា (Description)	តើម្លែង
Snake	ពិវប្បធម៌ដែលត្រូវបានរំភ័យដោយរួចរាល់តាមរូបរាយលើខ្លួន។ មានក្នុងការសរសៃរបៀប Snake ដែលបានរំភ័យដោយរួចរាល់តាមរូបរាយលើខ្លួន។	hello_world
Camel	ពិវប្បធម៌ដែលត្រូវបានរំភ័យដោយរួចរាល់តាមរូបរាយលើខ្លួន។ មានក្នុងការសរសៃរបៀប Camel ដែលបានរំភ័យដោយរួចរាល់តាមរូបរាយលើខ្លួន។	helloWorld
Pascal	ពិវប្បធម៌ដែលត្រូវបានរំភ័យដោយរួចរាល់តាមរូបរាយលើខ្លួន។	HelloWorld

## 5. ວິທີການແບບ Python (Pythonic way)

### 5.3. ຫຼັກການຕັ້ງຊື່ (Naming convention)

| ສິນທີສັນຍາ (conventions) ຂອງ Python ສະລັບການຕັ້ງຊື່ identifiers ດັ່ງລຸ່ມນີ້:

1. ໂດຍທີ່ວໄປແລ້ວປະກອບດ້ວຍຕົວອັກສອນພາສາອັງກິດ, ຕົວເລກ, ແລະ ຂີດກ້ອງ ( \_ ).
2. ບໍ່ຄວນມີຂ່ອງຫວ່າງຢູ່ເຄີ່ງກາງຂອງຂໍ້ຄວາມ.
3. ຄໍາທຳອິດຕ້ອງເລີ່ມຕົ້ນດ້ວຍຕົວອັກສອນພາສາອັງກິດ ຫຼື ເຄື່ອງ ( \_ ).
4. ຕົວອັກສອນພິມນ້ອຍ ແລະ ພິມໃຫ່ຍແມ່ນແຕກຕ່າງກັນ. ຕົວຢ່າງ, Count ແລະ count ແມ່ນ identifiers ຕ່າງກັນ.
5. ຄວາມຍາວຂອງ identifier ບໍ່ມີຈຳກັດ.

#### 6. Keywords ບໍ່ສາມາດໃຊ້ເປັນ identifiers.

7. Identifiers ທີ່ເລີ່ມຕົ້ນດ້ວຍສອງຂີດ \_\_ (two underscores) ອ້າງເຖິງຄຸນລັກສະນະພື້ນຖານ ຫຼື ໃຊ້ພຽງແຕ່ເປັນຊື່ຂອງ methods ສະເພາະ. ດັ່ງນັ້ນ, ມັນບໍ່ໄດ້ຖືກນຳໃຊ້ເປັນຊື່ຂອງພື້ນຖານທີ່ໄປ ຫຼື ຕົວປ່ຽນ

## 5. Pythonic way ວິທີການແບບ Python

### 5.3. ຫຼັກການຕັ້ງຊື່

| ຕົວຢ່າງຂອງ identifiers ທີ່ຖືກຕ້ອງ

Identifiers ທີ່ຖືກຕ້ອງ	Reasons ເຫດຜົນ
number4	ຕົວເລກສາມາດໃຊ້ໄດ້ຫຼັງຈາກເລີ່ມຕົ້ນດ້ວຍຕົວອັກສອນພາສາອັງກິດ
my_list	Underscores ສາມາດໃຊ້ໄດ້ຫຼັກບ່ອນໃນ identifier, ເຊັ່ນດຽວກັບຕົວອັກສອນອື່ນໆທີ່ເຫຼືອ
__code__	ຊື່ຂອງ method ສະເພາະສາມາດເລີ່ມດ້ວຍ __ (two underscores).
for_loop	keyword ສາມາດໃຊ້ໄດ້ດ້ວຍການຕໍ່ຄໍາດ້ວຍ _ (underscore).

## 5. Pythonic way ວິທີການແບບ Python

### 5.3. ຫຼັກການຕັ້ງຊື່

| ຕົວຢ່າງການຕັ້ງຊື່ identifiers ທີ່ບໍ່ຖືກຕ້ອງ

Identifiers ທີ່ບໍ່ຖືກຕ້ອງ	Reasons ເຫດຜົນ
1st_variable	ບໍ່ສາມາດໃຊ້ໄດ້ເພະວ່າເລີ່ມຕົ້ນດ້ວຍຕົວເລກ
my list	ບໍ່ສາມາດຕັ້ງໄດ້ເພະວ່າມີຍະຫວ່າງ (space).
global	keywords ຂອງ Python ບໍ່ສາມາດໃຊ້ເປັນ identifiers ໄດ້
ver2.9	ຕົວອັກສອນພິເສດນອກຈາກ underscore ແມ່ນບໍ່ສາມາດໃຊ້ໄດ້
num&co	ຕົວອັກສອນພິເສດນອກຈາກ underscore ແມ່ນບໍ່ສາມາດໃຊ້ໄດ້

## 6. ខ្លឹមិតិណ៍ Error

### 6.1. ខ្លឹមិតិណ៍ឡើយាកន់

- | ធនាគារនខ្លួនប្រកបដករាយការណ៍ទីក្រុងខ្លឹមិតិណ៍ឡើយាកន់ និង conventions ទាំងអស់។ ការងារនខ្លួនទីនឹងបានស្តីពីការងារនខ្លួនដូចជា syntax error.
- | ផែនការរបាយការនខ្លឹមិតិណ៍ឡើយាកន់ គឺជាបញ្ជីការងារនខ្លួន និងការងារនខ្លួន និង code ខោរយ Python តាមរាយការណ៍ទីក្រុងខ្លឹមិតិណ៍ឡើយាកន់។
- | ផែនការរបាយការនខ្លឹមិតិណ៍ឡើយាកន់ គឺជាបញ្ជីការងារនខ្លួន និងការងារនខ្លួន និង code ខោរយ Python តាមរាយការណ៍ទីក្រុងខ្លឹមិតិណ៍ឡើយាកន់។

```
1 print('Hello')
File "<ipython-input-2-db8c9988558c>", line 1
    print('Hello')
^
SyntaxError: EOL while scanning string literal
```

## 6. ຂໍ້ຜິດພາດ (Error)

### 6.2. Runtime Error ຂໍຜິດພາດໃນລະຫວ່າງໂປຣແກຣມເຮັດວຽກ

- | ຕ່າງຈາກຂໍ້ຜິດພາດທາງໄວຍາກອນ, code ທີ່ຖືກຕ້ອງຕາມໄວຍະກອນຍັງສາມາດສົ່ງຜົນໃຫ້ເກີດຄວາມຜິດພາດໃນລະຫວ່າງການປະຕິບັດໄດ້. ປະເພດຂໍ້ຜິດພາດນີ້ເອີ້ນວ່າ runtime error (ຂໍ້ຜິດພາດໃນລະຫວ່າງໂປຣແກຣມເຮັດວຽກ).
- | ເພື່ອປ້ອງກັນຂໍ້ຜິດພາດໃນລະຫວ່າງໂປຣແກຣມເຮັດວຽກ (runtime errors), ມັກພັດທະນາຄວນພິຈາລະນາຄວາມເປັນໄປໄດ້ທີ່ຜູ້ໃຊ້ຈະປ້ອນຂໍ້ມູນທີ່ບໍ່ຖືກຕ້ອງເຂົ້າໄປ ແລະ ແນະນຳຄໍາຮູ້ອ່ານຂໍທີ່ເປັນມິດກັບຜູ້ໃຊ້ເພື່ອຊູກຍູ້ໃຫ້ປ້ອນຂໍ້ມູນທີ່ຖືກຕ້ອງ. ນອກຈາກນີ້, ມັກຈະມີກໍລະນີຂໍ້ຍົກເວັ້ນ (exception) ເກີດຂຶ້ນຫຼາຍກໍລະນີ, ດັ່ງນັ້ນ ຈຶ່ງຂໍແນະນຳເປັນຢ່າງຍິ່ງໃຫ້ທິດສອບ code ໃນເງື່ອນໄຂຕ່າງໆ.
- | ໃນ code ລຸ່ມນີ້, ຜູ້ໃຊ້ປ້ອນຂໍ້ຄວາມ “two” ແທນເລກ 2 ແລະ ຂໍຜິດພາດໃນລະຫວ່າງໂປຣແກຣມເຮັດວຽກ (runtime error) ເກີດຂຶ້ນ.

```
1 num = int(input('Input an integer: '))
```

```
Input an integer: two
```

```
-----  
ValueError                                                 Traceback (most recent call last)  
<ipython-input-3-2c1961434081> in <module>  
----> 1 num = int(input('Input an integer: '))  
  
ValueError: invalid literal for int() with base 10: 'two'
```



## One more step

### 1. Runtime Error and Resolution ຂໍຜິດພາດແບບ Runtime ແລະ ອີທີແກ້ໄຂ

- | ຕົວຢ່າງຂອງ runtime error ທີ່ພືບເຫັນເລື່ອຍ່າມມ່ນ Zero DivisionError, ຂໍຜິດພາດທີ່ເກີດຈາກການຫານຕົວເລກດ້ວຍເລກ 0. ເພື່ອແກ້ໄຂຂໍຜິດພາດແບບ runtime errors ດັ່ງກ່າວ, ຂໍຍົກເວັ້ນ (exceptions) ສາມາດເພີ່ມໂດຍນໍາໃຊ້ try ~ except statements.
- | ເນື້ອຜູ້ໃຊ້ປ້ອນເລກຈໍານວນເຕັມສອງໂຕຢ່າງຖືກຕ້ອງດັ່ງຕໍ່ໄປນີ້, ການດຳເນີນການການຫານທັງສອງຕົວເລກຈະດຳເນີນການໂດຍບໍ່ມີຂໍຜິດພາດຕາມທີ່ໂປຣແກຣມຕ້ອງການ.

```
1 try:  
2     a, b = input('Enter two integers.').split()  
3     result = int(a) / int(b)  
4     print('{}/{} = {}'.format(a, b, result))  
5 except :  
6     print('Check if the integers are correctly entered.')
```

Enter two integers. 10 2  
10/2 = 5.0

## 💡 อີກນິ້ງຂັ້ນຕອນ (One more step)

### 1. Runtime Error and Resolution ຂໍຜິດພາດແບບ Runtime ແລະ ອິທີແກ້ໄຂ

ຖ້າຜູ້ໃຊ້ປ້ອນຂໍ້ມູນບໍ່ຖືກຕ້ອງ ປ້ອນຂໍ້ຄວາມທີ່ແທນຈະປ້ອນເລກຈຳນວນຕົມ, ຫຼືເຮັດໃຫ້ເກີດກຳລະນີທີ່ຕົວເລກຫານດ້ວຍເລກ 0 ດັ່ງຕົວຢ່າງ  
ທີ່2 ລຸ່ມນີ້, except statement ຈະຈັດການກັບກຳລະນີ exception ດັ່ງກ່າວ.

```
1 try:  
2     a, b = input('Enter two integers.').split()  
3     result = int(a) / int(b)  
4     print('{} / {} = {}'.format(a, b, result))  
5 except :  
6     print('Check if the integers are correctly entered.')
```

Enter two integers. 10 two  
Check if the integers are correctly entered.

```
1 try:  
2     a, b = input('Enter two integers.').split()  
3     result = int(a) / int(b)  
4     print('{} / {} = {}'.format(a, b, result))  
5 except :  
6     print('Check if the integers are correctly entered.')
```

Enter two integers. 10 0  
Check if the integers are correctly entered.

Pair programing  
จับคู่เขียนโปรแกรม



# Pair Programming Practice

## | Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns “driving the mouse.” Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

## | Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a “divide-and-conquer” strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

## | Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.



# Pair Programming Practice

## | Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

## | Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor's question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students "divide and conquer." When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

**Q1.** ຂຽນ Code 7 ແກວຂອງພົງຊັນ print() ເພື່ອໃຫ້ໄດ້ຜົນໄດ້ຮັບດັ່ງລຸ່ມນີ້:

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

Samsung SW Qualification Test sample question  
ຄໍາຖາມຕົວຢ່າງການທິດສອບຄຸນສົມບັດ Samsung SW