Python Programming: Beginner (Integrative-Generative AI Edition)

Variables and Data Types • Progrrams • Functions • Collections

Student Price Book Center

ai

คำนำ

ในโลกปัจจุบันที่เทคโนโลยีก้าวหน้าอย่างไม่หยุดยั้ง **"ภาษา Python"** ได้กลายเป็นหนึ่งในภาษาการเขียนโปรแกรมที่ได้รับความนิยมสูงสุด ด้วยความเรียบง่าย อ่านเข้าใจง่าย และสามารถนำไปประยุกต์ใช้งานได้หลากหลาย ไม่ว่าจะเป็นด้าน **วิทยาการข้อมูล (Data Science), ปัญญาประดิษฐ์ (AI), การพัฒนาเว็บ, การวิเคราะห์ข้อมูล, ไปจนถึงการควบคุมฮาร์ดแวร์** โดยไม่จำเป็นต้องมีพื้นฐานด้านการเขียนโปรแกรมมาก่อนก็สามารถเริ่มต้นได้

หนังสือเล่มนี้จึงถูกเขียนขึ้นด้วยวัตถุประสงค์สำคัญคือ **“พาผู้อ่านเดินทางจากระดับผู้เริ่มต้น (Beginner) ไปสู่ระดับมืออาชีพ (Professional) อย่างเป็นระบบ”** โดยมีการจัดเรียงเนื้อหาตามลำดับความยากง่าย พร้อมตัวอย่างประกอบที่เข้าใจง่ายและสามารถนำไปทดลองได้ทันที
ในส่วนแรกของหนังสือนี้จะครอบคลุมเนื้อหาในระดับ **ผู้เริ่มต้น (Beginner Level)** ได้แก่:

- การติดตั้ง Python และเครื่องมือที่ใช้พัฒนา (IDE)
- การเขียนโปรแกรม Python ไฟล์แรก
- การแสดงผลและรับข้อมูลจากผู้ใช้
- การใช้ตัวแปรและชนิดข้อมูลพื้นฐาน เช่น int, float, str, bool
- โครงสร้างควบคุมโปรแกรม เช่น if, for, while
- ฟังก์ชันพื้นฐานและการส่งค่าระหว่างฟังก์ชัน
- การใช้งานคอลเลกชัน (list, tuple, set, dict) อย่างเข้าใจ

ทุกบทจะมาพร้อม **ตัวอย่างโค้ด คำอธิบายผลลัพธ์ และแนวคิดเบื้องหลัง** เพื่อให้ผู้อ่านสามารถเรียนรู้ได้ด้วยตนเองอย่างมั่นใจ

หวังเป็นอย่างยิ่งว่าหนังสือเล่มนี้จะเป็น **คู่มือสำคัญในการเริ่มต้นก้าวแรกของคุณในโลกแห่งการเขียนโปรแกรม** และจะช่วยให้คุณต่อยอดความรู้สู่การประยุกต์ใช้ในระดับสูงได้อย่างมั่นคง
ขอให้ทุกการเรียนรู้ของคุณเต็มไปด้วยความเข้าใจ ความสนุก และแรงบันดาลใจ

ด้วยความเคารพ
ศูนย์หนังสือราคานักเรียน

สารบัญ

หน้า

บทที่ 1 พื้นฐาน Python(Python Basic)..... 1
- ●ความสำคัญของภาษาโปรแกรม Python
- ●ประวัติของภาษา Python
- ●การเติบโตของภาษา Python และบทบาทในแต่ละยุค
- ●สถานะของภาษา Python ในปัจจุบัน (ปี 2025)
- ●หน่วยงาน องค์กร และบริษัท ที่ใช้งานภาษา Python
- ●พื้นฐานภาษา Python
- ●การติดตั้ง Python และ IDE (เช่น VSCode, PyCharm)
- ●การติดตั้งและใช้งาน Python ผ่าน Google Colab และ Jupyter Notebook
- ●การเขียนโปรแกรม Python ไฟล์แรก (hello.py)
- ● การแสดงผลและการรับข้อมูลใน Python (ฟังก์ชัน print() และ input())
- ●ตัวอย่างโปรแกรม Python เบื้องต้น

บทที่ 2 ตัวแปรและชนิดข้อมูลพื้นฐาน (Variables and Data Types).....24
- ●ตัวแปรและชนิดข้อมูลพื้นฐานในภาษา Python
- ●ข้อมูลเชิงลึก (In-Depth Explanation)
- ●ชนิดข้อมูล (Data Types) ในภาษา Python
- ●ประยุกต์ใช้ชนิดข้อมูลพื้นฐาน
- ●การแปลงชนิดข้อมูล (Type Conversion)
- ●การใช้ฟังก์ชันใน Python
- ●ฟังก์ชัน built-in
- •แนวทางการประยุกต์ใช้

บทที่ 3 โครงสร้างควบคุมโปรแกรม (Program Control Flow).....67
- ●โครงสร้างควบคุมโปรแกรม (Control Structures)
- ●โครงสร้างควบคุมโปรแกรม (Control Structures) ใน Python อย่างละเอียดเชิงลึก
- ●คำสั่งเงื่อนไข (if, elif, else)

●คำสั่งวนซ้ำใน Python ทั้งแบบ for และ while
●คำสั่งวนซ้ำ (for, while) ใน Python แบบละเอียด
●การใช้งานของ break, continue, และ pass ใน Python
บทที่ 4 ฟังก์ชัน (Function) 109
●ฟังก์ชัน (Functions) ใน Python
●ฟังก์ชัน (Functions) ใน Python — รายละเอียดเชิงลึก
●การสร้างและเรียกใช้ฟังก์ชัน (def) ใน Python
●การสร้างและเรียกใช้ฟังก์ชัน (def) ใน Python — แบบละเอียด
●การส่งค่าผ่าน parameter และ return ในฟังก์ชัน Python
●การส่งค่าผ่าน Parameter และการคืนค่า (Return) ใน Python - เชิงลึก
●Default arguments และ Keyword arguments ใน Python
●แนวทางบูรณาการใช้งานฟังก์ชันใน Python
บทที่ 5 คอลเลกชัน (Collections) 156
●คอลเลกชัน (Collections)
●คอลเลกชัน (Collections) ใน Python — เจาะลึกเชิงลึก
●list, tuple, set, dict
●บูรณาการ ตัวแปรและชนิดข้อมูล, โครงสร้างควบคุมโปรแกรม, ฟังก์ชัน, และ คอลเลกชัน (list, tuple, set, dict)
● การเพิ่ม (Add), ลบ (Delete), แก้ไข (Update), และวนลูป (Loop) ผ่าน Collections
●ตัวอย่างแนวประยุกต์ใช้งานจริง
●การ ใช้ list comprehension
●การประยุกต์ใช้งานจริงของ คอลเลกชันใน Python
บรรณานุกรม 205

บทที่ 1
พื้นฐาน Python
(Python Basic)

เนื้อหา

- ความสำคัญของภาษาโปรแกรม Python
- ประวัติของภาษา Python
- การเติบโตของภาษา Python และบทบาทในแต่ละยุค
- สถานะของภาษา Python ในปัจจุบัน (ปี 2025)
- หน่วยงาน องค์กร และบริษัท ที่ใช้งานภาษา Python
- พื้นฐานภาษา Python
- การติดตั้ง Python และ IDE (เช่น VSCode, PyCharm)
- การติดตั้งและใช้งาน Python ผ่าน Google Colab และ Jupyter Notebook
- การเขียนโปรแกรม Python ไฟล์แรก (hello.py)
- การแสดงผลและการรับข้อมูลใน Python (ฟังก์ชัน print() และ input())
- ตัวอย่างโปรแกรม Python เบื้องต้น

บทนำ

ในยุคปัจจุบันที่เทคโนโลยีสารสนเทศมีบทบาทสำคัญในทุกมิติของชีวิตประจำวัน การเรียนรู้ทักษะด้านการเขียนโปรแกรมจึงกลายเป็นพื้นฐานสำคัญที่บุคคลทั่วไปและผู้ที่ทำงานในสายอาชีพต่าง ๆ ไม่สามารถหลีกเลี่ยงได้ ภาษา Python ได้รับความนิยมอย่างแพร่หลายเนื่องจากมีโครงสร้างที่อ่านง่ายใกล้เคียงกับภาษามนุษย์ และเหมาะสมอย่างยิ่งสำหรับผู้เริ่มต้นที่ต้องการเข้าสู่โลกของการพัฒนาโปรแกรมอย่างมีประสิทธิภาพ

หนังสือเล่มนี้มุ่งเน้นการให้ความรู้พื้นฐานเกี่ยวกับภาษา Python โดยเริ่มต้นจากขั้นตอนการติดตั้งโปรแกรมและเครื่องมือพัฒนา เช่น Python Interpreter และโปรแกรมเขียนโค้ดยอดนิยมอย่าง Visual Studio Code (VSCode) และ PyCharm ซึ่งผู้อ่านจะได้เรียนรู้วิธีการติดตั้งอย่างเป็นขั้นตอน พร้อมคำแนะนำในการตั้งค่าสภาพแวดล้อมเบื้องต้นให้พร้อมใช้งานสำหรับการเขียนโปรแกรม หลังจากติดตั้งเครื่องมือเรียบร้อยแล้ว หนังสือจะพาผู้อ่านเขียนโปรแกรม Python ไฟล์แรกที่มีชื่อว่า hello.py ซึ่งเป็นจุดเริ่มต้นของการสร้างโปรแกรมง่าย ๆ ที่สามารถแสดงข้อความออกทางหน้าจอ การ

เริ่มต้นในระดับพื้นฐานนี้มีเป้าหมายเพื่อสร้างความเข้าใจในโครงสร้างของโปรแกรม Python อย่างเป็นระบบ และกระตุ้นให้ผู้อ่านเกิดความมั่นใจในการเรียนรู้ต่อไป

นอกจากนี้ ผู้อ่านจะได้เรียนรู้ฟังก์ชันสำคัญเบื้องต้นของ Python ได้แก่ print() สำหรับแสดงผลลัพธ์ และ input() สำหรับรับข้อมูลจากผู้ใช้ ซึ่งเป็นพื้นฐานของการโต้ตอบระหว่างโปรแกรมกับผู้ใช้งาน ฟังก์ชันเหล่านี้จะถูกนำไปประยุกต์ใช้ในโปรแกรมขนาดเล็กเพื่อสร้างความเข้าใจในการประมวลผลข้อมูลที่ผู้ใช้ป้อนเข้ามา และการตอบสนองของโปรแกรมอย่างเหมาะสม

ด้วยแนวทางการเรียนรู้แบบค่อยเป็นค่อยไป หนังสือเล่มนี้จึงเหมาะสำหรับผู้เริ่มต้นที่ไม่มีพื้นฐานด้านการเขียนโปรแกรมมาก่อน และสามารถใช้เป็นคู่มือในการเรียนรู้ด้วยตนเองหรือเป็นสื่อประกอบการสอนในชั้นเรียนได้อย่างมีประสิทธิภาพ จุดมุ่งหมายสำคัญคือการปูพื้นฐานที่มั่นคงเพื่อให้ผู้อ่านสามารถพัฒนาไปสู่ระดับที่สูงขึ้นในการเขียนโปรแกรมภาษา Python ได้อย่างมั่นใจและต่อเนื่อง.

ความสำคัญของภาษาโปรแกรม Python

ภาษา Python เป็นภาษาโปรแกรมระดับสูงที่มีความสำคัญอย่างยิ่งในโลกยุคดิจิทัล เนื่องจากมีโครงสร้างที่เรียบง่าย อ่านง่าย ใกล้เคียงกับภาษามนุษย์ ทำให้เหมาะสำหรับผู้เริ่มต้นและยังทรงพลังพอสำหรับนักพัฒนามืออาชีพ ด้วยคุณสมบัติที่สนับสนุนการพัฒนาได้ทั้งในด้านวิทยาศาสตร์ข้อมูล (Data Science), ปัญญาประดิษฐ์ (AI), เว็บแอปพลิเคชัน, การประมวลผลข้อมูลอัตโนมัติ (Automation) และการควบคุมฮาร์ดแวร์ (IoT) Python จึงกลายเป็นภาษาหลักในหลายองค์กรชั้นนำทั่วโลก อีกทั้งยังมีชุมชนผู้ใช้งานขนาดใหญ่ที่พร้อมแบ่งปันเครื่องมือ ไลบรารี และความรู้ต่าง ๆ อย่างต่อเนื่อง ทำให้การเรียนรู้และพัฒนาโปรแกรมด้วย Python เป็นเรื่องที่เข้าถึงได้ง่ายและทรงประสิทธิภาพในทุกระดับของผู้เรียน.

ภาษา Python มีความสำคัญและคุณค่าอย่างยิ่งในหลากหลายด้านของการพัฒนาโปรแกรมและเทคโนโลยีสมัยใหม่ ด้วยคุณสมบัติดังต่อไปนี้:

1. **ไวยากรณ์เรียบง่ายและอ่านเขียนง่าย**

   Python ถูกออกแบบให้มีโครงสร้างโค้ดที่ชัดเจน ลด “noise” ในการเขียนโปรแกรม เช่น การใช้ indentation แทนเครื่องหมายปีกกา ({}) ทำให้โค้ดอ่านเหมือนภาษาธรรมชาติ ส่งผลให้ผู้เริ่มต้นเรียนรู้ได้รวดเร็ว และนักพัฒนามืออาชีพสามารถทำงานร่วมกันได้ง่าย

2. **ไลบรารีมาตรฐานและสังคมผู้ใช้ขนาดใหญ่**

   Python มาพร้อมกับไลบรารีมาตรฐาน (Standard Library) ที่ครอบคลุมการทำงานทั่วไป เช่น การจัดการไฟล์, เครือข่าย, การเข้ารหัส, ฐานข้อมูล และอื่น ๆ อีกมากมาย นอกจากนี้ ชุมชนโอเพนซอร์สของ Python ยังสร้างไลบรารีเสริม (Third-Party Packages) จำนวนมหาศาล เช่น NumPy/Pandas (Data Science), TensorFlow/PyTorch (Machine Learning), Django/Flask (Web Development) ทำให้การพัฒนารวดเร็วและง่ายดาย

3. **รองรับการใช้งานหลายรูปแบบ (Versatility)**

   – **เว็บแอปพลิเคชัน**: ด้วยเฟรมเวิร์กชื่อดังอย่าง Django และ Flask

– **งานวิทยาศาสตร์ข้อมูลและ AI**: ไลบรารีสำหรับการวิเคราะห์ข้อมูล การสร้างโมเดลปัญญาประดิษฐ์ และการเรียนรู้เชิงลึก

– **สคริปต์อัตโนมัติ (Automation & Scripting)**: ใช้เขียนสคริปต์เพื่อจัดการงานซ้ำซ้อน เช่น การประมวลผลไฟล์แบทช์ หรือการตั้งเวลาเป้าหมาย

– **IoT & Embedded**: ใช้บนอุปกรณ์อย่าง Raspberry Pi เพื่อควบคุมฮาร์ดแวร์หรือสร้างโปรโตไทป์อุปกรณ์สมองกลฝังตัว

4. **รองรับการพัฒนาอย่างรวดเร็ว (Rapid Prototyping)**
   ด้วยการเป็นภาษาที่ตีความ (interpreted) และมีโครงสร้างโค้ดที่กระชับ Python ช่วยให้นักพัฒนาสามารถสร้างต้นแบบ (prototype) ของแอปพลิเคชันหรืออัลกอริทึมได้อย่างรวดเร็ว ลดเวลาวงจรการพัฒนา
5. **ข้ามแพลตฟอร์มและความสามารถในการขยาย (Cross-Platform & Extensibility)**
   โค้ด Python สามารถรันได้บนระบบปฏิบัติการยอดนิยม ทั้ง Windows, macOS, Linux โดยไม่ต้องปรับแต่งมากขึ้น อีกทั้งยังสามารถเชื่อมต่อกับภาษาอื่น เช่น C/C++ เพื่อเพิ่มประสิทธิภาพในส่วนที่ต้องใช้การประมวลผลหนัก
6. **ทรัพยากรการเรียนรู้และโอกาสทางอาชีพ**
   ด้วยความนิยมอย่างกว้างขวาง ทำให้มีคอร์สเรียน เอกสาร บทความ และชุมชนออนไลน์ให้ศึกษามากมาย การมีทักษะ Python จึงเป็นจุดเด่นในเรซูเม่ และเปิดโอกาสรับงานในหลากหลายสายงาน ตั้งแต่ Data Analyst, Backend Developer, DevOps Engineer ไปจนถึง Research Scientist ในสถาบันการศึกษาและองค์กรเทคโนโลยีชั้นนำทั่วโลก

ด้วยเหตุนี้ Python จึงไม่ได้เป็นเพียง “ภาษาโปรแกรมอีกตัวหนึ่ง” แต่เป็นเครื่องมือสำคัญที่ขับเคลื่อนนวัตกรรมทางเทคโนโลยีในยุคปัจจุบัน ช่วยให้นักพัฒนาสามารถสร้างสรรค์ผลงาน ร่วมมือในโครงการขนาดใหญ่ และเรียนรู้เทคโนโลยีใหม่ ๆ ได้อย่างมีประสิทธิภาพและต่อเนื่องยาวนานยิ่งขึ้น.

ประวัติของภาษา Python

ภาษา Python ถูกพัฒนาโดย **Guido van Rossum** โปรแกรมเมอร์ชาวเนเธอร์แลนด์ ในช่วงปลายปี ค.ศ. 1980 และเปิดตัวเวอร์ชันแรกอย่างเป็นทางการในปี ค.ศ. 1991 จุดประสงค์ของเขาคือการสร้างภาษาโปรแกรมที่มีความอ่านง่าย เข้าใจง่าย และลดความซับซ้อนของโค้ด เพื่อให้นักพัฒนาสามารถเขียนโปรแกรมได้อย่างรวดเร็วและมีประสิทธิภาพ

ชื่อ **"Python"** นั้น ไม่ได้มาจากชื่อของงู แต่ได้แรงบันดาลใจจากรายการตลกของอังกฤษที่ชื่อว่า *Monty Python's Flying Circus* ซึ่งสะท้อนถึงความตั้งใจของผู้พัฒนาในการสร้างภาษาโปรแกรมที่ "สนุก" และใช้งานง่าย

ในช่วงแรก Python ได้รับความนิยมเฉพาะในกลุ่มนักวิจัยและนักวิชาการ เพราะมีลักษณะเหมาะสำหรับการทดลองและการศึกษาขั้นพื้นฐาน ต่อมาเมื่อเข้าสู่ยุคอินเทอร์เน็ตและซอฟต์แวร์แบบ

โอเพนซอร์ส Python ก็เริ่มเติบโตอย่างรวดเร็ว โดยมีการพัฒนาไลบรารีและเฟรมเวิร์กต่าง ๆ ที่สนับสนุนงานด้านเว็บแอปพลิเคชัน, วิทยาศาสตร์ข้อมูล, ปัญญาประดิษฐ์, และงานระบบอัตโนมัติ

Python มีการพัฒนาอย่างต่อเนื่องผ่านเวอร์ชันต่าง ๆ โดยเฉพาะการเปลี่ยนแปลงจาก **Python 2** สู่ **Python 3** ซึ่งเปิดตัวในปี ค.ศ. 2008 เพื่อแก้ไขข้อจำกัดของเวอร์ชันเดิมและพัฒนาให้ทันสมัยยิ่งขึ้น แม้ว่าทั้งสองเวอร์ชันจะไม่เข้ากันแบบย้อนกลับ (backward compatibility) แต่การเปลี่ยนแปลงนี้ถือเป็นจุดเปลี่ยนสำคัญที่ทำให้ Python กลายเป็นภาษาที่เข้มแข็งและรองรับการใช้งานอย่างหลากหลาย ในปัจจุบัน Python กลายเป็นหนึ่งในภาษาโปรแกรมที่ได้รับความนิยมสูงสุดทั่วโลก ได้รับการจัดอันดับให้อยู่ในอันดับต้น ๆ ของภาษาโปรแกรมที่ใช้มากที่สุดจากแหล่งสำคัญ เช่น TIOBE Index และ Stack Overflow Developer Survey ด้วยเหตุนี้ Python จึงไม่เพียงแต่เป็นภาษาสำหรับผู้เริ่มต้นเท่านั้น แต่ยังเป็นเครื่องมือสำคัญในสายงานเทคโนโลยีชั้นสูงทั่วโลก.

การเติบโตของภาษา Python และบทบาทในแต่ละยุค

หลังจากเปิดตัว **Python เวอร์ชัน 1.0 ในปี 1991** ซึ่งประกอบด้วยฟีเจอร์สำคัญหลายประการ เช่น การจัดการโมดูล การใช้ชนิดข้อมูลพื้นฐาน และการควบคุมโฟลว์ (control flow) ก็มีการพัฒนาต่อเนื่องอย่างสม่ำเสมอ โดยในช่วง **ปลายยุค 1990 ถึงต้น 2000** Python เริ่มเป็นที่รู้จักในวงกว้าง โดยเฉพาะในกลุ่มนักวิจัย วิศวกร และนักศึกษาด้านวิทยาการคอมพิวเตอร์

ในปี **2000** ได้มีการเปิดตัว **Python 2.0** ซึ่งเพิ่มความสามารถใหม่ ๆ เช่น **List comprehensions**, **Garbage collection แบบอัตโนมัติ** และ **Unicode support** อย่างไรก็ตาม Python 2 มีข้อจำกัดที่ไม่สามารถแก้ไขได้โดยไม่ทำลายความเข้ากันได้ของโค้ดเก่า (backward compatibility) จึงนำไปสู่การออกแบบเวอร์ชันใหม่ที่ทันสมัยมากขึ้น

Python 3 และการเปลี่ยนผ่านครั้งสำคัญ

Python 3.0 เปิดตัวในปี **2008** โดยมีการปรับปรุงโครงสร้างของภาษาให้รองรับงานที่ซับซ้อนมากขึ้น เช่น

- การจัดการสตริงแบบ Unicode เป็นมาตรฐาน
- การเปลี่ยนแปลงการทำงานของ print จาก statement เป็น function
- การใช้ iterator และการจัดการหน่วยความจำที่ดีขึ้น

แม้ว่าในช่วงแรกจะมีการต่อต้านจากผู้ใช้ Python 2 ที่มีฐานโค้ดเดิมอยู่จำนวนมาก แต่เมื่อเวลาผ่านไป ชุมชน Python ได้ร่วมกันสนับสนุนการย้ายไปสู่ Python 3 และในปี **2020** Python 2 ก็ได้สิ้นสุดการซัพพอร์ตอย่างเป็นทางการ (End of Life) ทำให้ Python 3 กลายเป็นมาตรฐานใหม่อย่างสมบูรณ์

การใช้งานในอุตสาหกรรมและชุมชนโลก

Python ไม่ได้จำกัดอยู่เพียงงานเขียนโปรแกรมทั่วไปเท่านั้น แต่ยังถูกนำไปใช้ในภาคอุตสาหกรรมอย่างกว้างขวาง เช่น

- **Google, Facebook, Instagram, Dropbox, Netflix, NASA** ต่างก็ใช้ Python ในงานหลากหลายรูปแบบ
- ในวงการ **วิทยาศาสตร์ข้อมูล (Data Science), การเรียนรู้ของเครื่อง (Machine Learning), และ AI** Python ได้กลายเป็นภาษาหลัก เนื่องจากมีไลบรารีทรงพลัง เช่น NumPy, Pandas, Scikit-learn, TensorFlow และ PyTorch
- ด้าน **การศึกษาระดับมหาวิทยาลัย** Python ได้รับการบรรจุเป็นภาษาหลักในหลักสูตร CS101 ของมหาวิทยาลัยชั้นนำหลายแห่ง เช่น MIT, Harvard, Stanford

---

บทบาทในอนาคตและการพัฒนาอย่างต่อเนื่อง

ปัจจุบัน Python ได้รับการดูแลโดย **Python Software Foundation (PSF)** ซึ่งเป็นองค์กรไม่แสวงผลกำไรที่สนับสนุนการพัฒนาและขยายการใช้งานของภาษาอย่างต่อเนื่อง แม้ว่า **Guido van Rossum** จะประกาศวางมือจากบทบาท “Benevolent Dictator For Life (BDFL)” ในปี 2018 แต่เขายังเป็นบุคคลสำคัญในวงการ Python และกลับมาเข้าร่วมโครงการพัฒนาในด้านต่าง ๆ เช่น การปรับปรุงประสิทธิภาพของ Interpreter

---

สรุปความสำคัญของประวัติ Python

การเติบโตของ Python ตลอดสามทศวรรษที่ผ่านมาแสดงให้เห็นถึงศักยภาพของภาษาในการปรับตัวต่อการเปลี่ยนแปลงของเทคโนโลยีและความต้องการของผู้ใช้งาน ด้วยแนวคิดที่ยึดหลักความเรียบง่าย (Simplicity), ความชัดเจน (Clarity), และประสิทธิภาพในการทำงาน (Productivity) Python จึงกลายเป็นภาษาหลักในหลายภาคส่วน และจะยังคงเป็นแกนกลางของการเรียนรู้ การพัฒนา และนวัตกรรมในยุคดิจิทัลต่อไปอีกยาวนาน

สถานะของภาษา Python ในปัจจุบัน (ปี 2025)

ณ ปี 2025 ภาษา **Python** ยังคงเป็นหนึ่งในภาษาโปรแกรมที่ได้รับความนิยมสูงสุดทั่วโลก และเป็นภาษาหลักที่ใช้อย่างกว้างขวางทั้งในด้านการศึกษา อุตสาหกรรม และงานวิจัยขั้นสูง โดยมีลักษณะเด่นและความเคลื่อนไหวสำคัญดังต่อไปนี้:

☐ **1. ความนิยมและการใช้งานที่กว้างขวาง**

- Python ยังคงครองอันดับต้น ๆ ของภาษาโปรแกรมที่นิยมมากที่สุด จากการจัดอันดับของ **TIOBE Index**, **Stack Overflow Developer Survey** และ **RedMonk**
- ถูกใช้อย่างแพร่หลายในสายงานต่าง ๆ เช่น:

- **Data Science / Machine Learning / AI**
- **Web Development**
- **DevOps & Automation**
- **Cybersecurity**
- **Education & STEM**

☐ 2. เวอร์ชันล่าสุดและฟีเจอร์ใหม่ (Python 3.12 และ 3.13)

- Python 3.12 (เปิดตัวปลายปี 2023) และ Python 3.13 (คาดว่าจะเปิดตัวในปลายปี 2024) มีการปรับปรุงทั้งด้าน:
  - **ประสิทธิภาพการทำงาน (Performance Improvements)** โดยเฉพาะ PEP 659 และ PEP 683
  - **การจัดการหน่วยความจำดีขึ้น** และรองรับการทำงานกับ async มากขึ้น
  - **เครื่องมือ Debug และ Error Message ฉลาดขึ้น** (เช่น precise error location, enhanced tracebacks)
- ทีมพัฒนาเริ่มวางแผนสำหรับ Python 3.14 ด้วยแนวคิดใหม่ เช่น การใช้ **JIT Compiler** เพื่อเร่งความเร็วระดับ C-like

---

☐ 3. ระบบนิเวศที่เติบโตต่อเนื่อง

- ไลบรารีและเฟรมเวิร์กยังคงขยายตัว เช่น:
  - **Data/AI**: NumPy, Pandas, Scikit-learn, PyTorch, TensorFlow, Hugging Face Transformers
  - **Web**: Django, FastAPI, Flask
  - **Automation**: Selenium, PyAutoGUI, Ansible
  - **Visualization**: Matplotlib, Seaborn, Plotly
- Python มี **แพลตฟอร์มออนไลน์** มากมาย เช่น Kaggle, Google Colab, Jupyter Notebook ที่ช่วยให้การเรียนรู้และการทดลองทำได้รวดเร็ว

☐ 4. ความเปลี่ยนแปลงในโครงสร้างการพัฒนา

- หลังจากที่ **Guido van Rossum** วางมือจากตำแหน่งผู้นำหลักของภาษาในปี 2018 ชุมชน Python ได้เปลี่ยนไปใช้ระบบ “**Steering Council**” ซึ่งมีทีมพัฒนาหลายคนร่วมตัดสินใจอย่างเป็นระบบ
- Guido กลับมามีบทบาทในทีมพัฒนา **CPython Performance** กับ Microsoft โดยมุ่งเป้าพัฒนา Python ให้ทำงานเร็วขึ้น 5 เท่าในระยะยาว

☐ 5. บทบาทของ Python ในยุค AI และ Generative Technology

- Python เป็นภาษาหลักในการสร้างและใช้งานโมเดลปัญญาประดิษฐ์ยุคใหม่ เช่น GPT, Stable Diffusion, และ BERT
- เป็นภาษาหลักที่ใช้ควบคุม AI model ผ่าน API และ SDK ต่าง ๆ ที่เปิดให้นักพัฒนาทั่วโลกเข้าถึงได้อย่างง่ายดาย

☐ 6. บทบาทในระบบการศึกษา

- Python ยังคงเป็นภาษาหลักในหลักสูตรการเขียนโปรแกรมระดับพื้นฐานในมหาวิทยาลัยทั่วโลก
- มีคอร์สเรียนออนไลน์ทั้งฟรีและเสียเงินจำนวนมาก เช่น Coursera, edX, Udemy, Codecademy
- ใช้ในโรงเรียนมัธยมเพื่อสอนแนวคิดการเขียนโปรแกรมเบื้องต้น (เนื่องจากเป็นมิตรต่อผู้เริ่มต้น)

☐ สรุป

Python ในปัจจุบันคือภาษาที่ผสานความง่าย ความยืดหยุ่น และความสามารถขั้นสูงไว้ในตัวเดียว จึงเหมาะทั้งสำหรับผู้เริ่มต้นและผู้เชี่ยวชาญ ด้วยระบบนิเวศที่เติบโตอย่างต่อเนื่อง ชุมชนที่เข้มแข็ง และการตอบสนองต่อเทคโนโลยีใหม่ ๆ Python จึงยังคงเป็นหนึ่งในภาษาหลักของโลกโปรแกรมมิ่งอย่างแท้จริงในปี 2025 และมีแนวโน้มจะครองความนิยมต่อไปอีกยาวนาน

หน่วยงาน องค์กร และบริษัท ที่ใช้งานภาษา Python

☐ 1. บริษัทเทคโนโลยีชั้นนำ (Big Tech Companies)

☐ **Google**

- ใช้ Python มาตั้งแต่เริ่มต้นบริษัท โดยถือว่าเป็นหนึ่งในภาษาหลัก
- ใช้ในหลายบริการ เช่น **Google Search, YouTube, Google Cloud, Google Colab**
- พัฒนาไลบรารีสำคัญอย่าง TensorFlow (Machine Learning) ด้วย Python
- มีแนวคิด “Python is one of our official server-side languages.”

☐ **Facebook (Meta)**

- ใช้ Python ในงาน **Infrastructure, Machine Learning, Data Analysis และ DevOps**
- ระบบตรวจสอบและวิเคราะห์ข้อมูลแบบ real-time ภายในองค์กรจำนวนมากพัฒนาโดยใช้ Python
- ใช้ในงาน backend ร่วมกับภาษาอื่น เช่น PHP และ C++

☐ **Instagram**

- พัฒนา Backend เกือบทั้งหมดด้วย Python โดยใช้ **Django Framework**
- Instagram ให้ความสำคัญกับความสามารถของ Python ที่สามารถปรับขยายระบบ (scalable) รองรับผู้ใช้หลายร้อยล้านคน

☐ **Netflix**

- ใช้ Python ในงาน **Automation, Data Analytics, Monitoring** และ **Recommendation Algorithms**
- มีการใช้ Python Script เพื่อปรับแต่งระบบ Streaming และ Recommendation ให้เหมาะกับผู้ใช้แต่ละคน

☐ **Dropbox**

- เขียนโค้ดหลักของแอปพลิเคชันเดสก์ท็อปด้วย Python
- **Guido van Rossum** เคยทำงานกับ Dropbox และมีส่วนในการพัฒนาโครงสร้าง Python ให้เหมาะกับระบบจริง

☐ **Amazon & AWS**

- ใช้ Python ในระบบ **Web Services (AWS)** และงานด้าน AI/ML บนบริการเช่น SageMaker
- Python ถูกใช้สร้างและควบคุม Lambda functions และระบบอัตโนมัติใน AWS อย่างแพร่หลาย

☐ **2. องค์กรด้านวิทยาศาสตร์ เทคโนโลยี และงานวิจัย**

☐ **NASA (National Aeronautics and Space Administration)**

- ใช้ Python สำหรับการ **วิเคราะห์ข้อมูลทางวิทยาศาสตร์อวกาศ, การจำลองการบิน, และระบบควบคุมยานอวกาศ**
- โครงการเช่น **Data-Driven Modelling** และระบบแสดงภาพ 3 มิติของอวกาศใช้ Python

☐ **CERN (องค์การวิจัยนิวเคลียร์ยุโรป)**

- ใช้ Python ในการ **ควบคุมและวิเคราะห์ข้อมูลจาก Large Hadron Collider (LHC)**
- ใช้ร่วมกับไลบรารี NumPy, SciPy และ matplotlib

☐ **MIT, Stanford, Harvard**

- สถาบันวิจัยและมหาวิทยาลัยชั้นนำใช้ Python เป็นภาษาหลักในการเรียนการสอน CS101
- สนับสนุนการใช้ Jupyter Notebook และ Google Colab ในห้องเรียน
- มีการพัฒนาโมเดล AI และอัลกอริธึมวิจัยด้วย Python

☐ **3. ภาคธุรกิจ การเงิน และการตลาด**

☐ **JPMorgan Chase & Goldman Sachs**

- ใช้ Python ในการ **วิเคราะห์ข้อมูลทางการเงิน, การสร้างโมเดลความเสี่ยง, และการจัดการพอร์ตการลงทุนอัตโนมัติ**

- ใช้ในระบบ Quantitative Trading และการทำ Automation

☐ **Bloomberg**

- ใช้ Python ในการพัฒนา **ระบบวิเคราะห์ข้อมูลตลาดหุ้น และเศรษฐกิจแบบเรียลไทม์**
- Bloomberg Terminal รองรับภาษา Python ในการสร้าง Custom Scripts

☐ **Spotify**

- ใช้ Python เพื่อ **วิเคราะห์พฤติกรรมผู้ฟัง, สร้างระบบแนะนำเพลง, และวิเคราะห์ Big Data**
- ประมวลผลข้อมูลขนาดใหญ่ผ่านเครื่องมือเช่น Apache Beam + Python

---

☐ **4. บริษัทที่เกี่ยวกับอุปกรณ์และระบบอัตโนมัติ**

☐ **Tesla**

- ใช้ Python ในการควบคุมและวิเคราะห์ระบบซอฟต์แวร์ภายในรถยนต์ รวมถึงงานด้าน AI สำหรับ **Autopilot**
- ใช้ Python Scripts สำหรับ Simulation และ Automation ในการผลิต

☐ **Intel, IBM, Microsoft**

- ใช้ Python สำหรับการพัฒนา **Machine Learning, Quantum Computing SDK (เช่น Qiskit), Data Analytics**
- Microsoft สนับสนุน Python บน Visual Studio, Azure AI และ GitHub Copilot

---

☐ **5. หน่วยงานภาครัฐและรัฐวิสาหกิจ**

☐ **รัฐบาลสหรัฐ, ยุโรป และเอเชียหลายประเทศ**

- ใช้ Python สำหรับ **การวิเคราะห์ข้อมูลขนาดใหญ่, การทำแผนที่ GIS, และการจัดการข้อมูลทางเศรษฐกิจ**
- ตัวอย่างเช่น **CDC (ศูนย์ควบคุมโรค)** ใช้ Python ในการวิเคราะห์ข้อมูลทางระบาดวิทยา

☐ **ธนาคารแห่งประเทศไทย และองค์กรด้านเศรษฐกิจในไทย**

- มีการใช้ Python ร่วมกับเครื่องมือ BI (Business Intelligence) ในการวิเคราะห์แนวโน้มเศรษฐกิจ และพฤติกรรมการเงินของประชาชน

☐ **6. สตาร์ทอัปและองค์กรยุคใหม่**

- สตาร์ทอัปจำนวนมากเลือกใช้ Python เนื่องจากสามารถพัฒนาโปรแกรมได้เร็ว รองรับการขยายตัว และมีเครื่องมือพร้อมใช้
- เฟรมเวิร์กยอดนิยมเช่น **FastAPI, Flask, Django** ถูกเลือกใช้พัฒนาเว็บแอปและ API
- Python กลายเป็นภาษาหลักในการสร้าง MVP (Minimum Viable Product) ของหลายบริษัท

☐ **สรุป**

Python เป็นภาษาที่ถูกใช้งานในองค์กรทุกระดับ ตั้งแต่บริษัทเทคโนโลยีขนาดใหญ่ หน่วยงานวิจัย ไปจนถึงสตาร์ทอัป และหน่วยงานภาครัฐ ด้วยเหตุผลหลัก คือ ความเรียบง่าย ความสามารถในการปรับตัวได้สูง ไลบรารีที่ครอบคลุม และชุมชนที่สนับสนุนอย่างเข้มแข็ง การเลือกใช้ Python ไม่เพียงเป็นทางเลือกด้านเทคนิค แต่เป็นกลยุทธ์ที่ช่วยให้การพัฒนาเทคโนโลยีมีความยั่งยืนและต่อยอดได้ในระยะยาว

พื้นฐานภาษา Python

☐ **1.1 การติดตั้ง Python และ IDE**

☐ **การติดตั้ง Python:**

1. ไปที่เว็บไซต์หลัก: https://www.python.org/downloads
2. เลือกเวอร์ชันล่าสุด (เช่น Python 3.12.x) แล้วกด **Download**
3. เปิดตัวติดตั้ง (Installer) แล้วให้ **ติ๊กถูกที่ "Add Python to PATH"** เพื่อให้สามารถเรียกใช้คำสั่ง python จาก command line ได้
4. คลิก “Install Now” แล้วรอจนติดตั้งเสร็จ

☐ **การตรวจสอบว่าติดตั้งสำเร็จ:**

เปิด Command Prompt หรือ Terminal แล้วพิมพ์:

```
python --version
```

หรือในบางเครื่อง:

```
python3 --version
```

☐ **ติดตั้ง IDE ยอดนิยม:**

1. **VS Code (Visual Studio Code)** – ฟรี, เบา, รองรับหลายภาษา
   - ดาวน์โหลดจาก https://code.visualstudio.com
   - ติดตั้ง Extension ชื่อ Python โดย Microsoft
2. **PyCharm** (โดย JetBrains)
   - เหมาะกับงานขนาดกลางถึงใหญ่
   - ดาวน์โหลดเวอร์ชันฟรี (Community) ได้ที่ https://www.jetbrains.com/pycharm/

☐ **1.2 การเขียนโปรแกรม Python ไฟล์แรก (hello.py)**

☐ **สร้างไฟล์ .py แรก:**

1. เปิด VS Code หรือ PyCharm
2. สร้างไฟล์ชื่อ hello.py
3. พิมพ์โค้ดต่อไปนี้:

```
print("Hello, world!")
```

การรันโปรแกรม:

VS Code:

- คลิกขวาที่ไฟล์ ⟶ เลือก “Run Python File in Terminal”
- หรือเปิด Terminal แล้วพิมพ์:

```
python hello.py
```

PyCharm:

- คลิกขวาที่ไฟล์ ⟶ เลือก “Run hello”

ผลลัพธ์ที่แสดง:

```
Hello, world!
```

1.3 การแสดงผลและการรับข้อมูล

คำสั่ง print(): ใช้แสดงข้อความหรือค่าลงบนหน้าจอ

```
print("สวัสดี Python")
print("อายุของคุณคือ", 25)
```

ผลลัพธ์:

```
สวัสดี Python
อายุของคุณคือ 25
```

คำสั่ง input(): ใช้รับข้อมูลจากผู้ใช้ (รับเป็นข้อความเสมอ)

```
name = input("ป้อนชื่อของคุณ: ")
print("สวัสดี", name)
```

ตัวอย่างผลลัพธ์เมื่อรัน:

```
ป้อนชื่อของคุณ: สมชาย
สวัสดี สมชาย
```

แปลงชนิดข้อมูลจาก input:

เนื่องจาก input() คืนค่าเป็น string เสมอ ถ้าเราต้องการให้เป็นตัวเลข ต้องแปลงชนิด:

```
age = int(input("ป้อนอายุของคุณ: "))
print("อีก 10 ปี คุณจะอายุ", age + 10)
```

ผลลัพธ์:

ป้อนอายุของคุณ: 25
อีก 10 ปี คุณจะอายุ 35

สรุปสิ่งที่คุณได้เรียนรู้

- วิธีติดตั้ง Python และ IDE ยอดนิยม
- เขียนโปรแกรมแรก: print("Hello, world!")
- แสดงข้อความด้วย print()
- รับข้อมูลจากผู้ใช้ด้วย input()
- การแปลงข้อมูลจาก string เป็นตัวเลขด้วย int()

การติดตั้ง Python และ IDE (เช่น VSCode, PyCharm)

การติดตั้ง Python และ IDE (VSCode, PyCharm)

---

1. การติดตั้ง Python (เวอร์ชัน 3.x)

Python เป็นภาษาการเขียนโปรแกรมแบบ high-level ที่ใช้งานง่าย เหมาะสำหรับทั้งมือใหม่และมืออาชีพ

ขั้นตอนการติดตั้ง Python:

1. เข้าเว็บไซต์ทางการของ Python:
   - https://www.python.org/downloads
2. คลิกปุ่ม **“Download Python 3.x.x”** (แนะนำให้ใช้เวอร์ชันล่าสุด)
3. เปิดไฟล์ติดตั้งที่ดาวน์โหลดมา (เช่น python-3.12.3-amd64.exe)
4. **สำคัญมาก:** ให้ติ๊กเครื่องหมายถูกที่ **"Add Python to PATH"** ที่อยู่ด้านล่าง
5. คลิก **Install Now** ⟶ รอให้ติดตั้งเสร็จ
6. ทดสอบการติดตั้ง:
   - เปิด **Command Prompt (Windows)** หรือ **Terminal (macOS/Linux)**
   - พิมพ์:
   - python --version
     หรือ (ในบางเครื่อง)
     python3 --version
7. หากแสดงเวอร์ชัน เช่น Python 3.12.3 แสดงว่าติดตั้งสำเร็จแล้ว

---

2. การติดตั้ง VS Code (Visual Studio Code)

VS Code คือ **IDE (Integrated Development Environment)** ที่เบาและรวดเร็ว เหมาะกับการเขียน Python

☐ **ขั้นตอนการติดตั้ง VS Code:**

1. เข้าเว็บไซต์: ☐ https://code.visualstudio.com
2. คลิก “Download for Windows” (หรือระบบปฏิบัติการอื่น ๆ)
3. เปิดไฟล์ติดตั้ง ⟶ ติดตั้งตามขั้นตอนปกติ
4. หลังติดตั้งเสร็จ เปิด VS Code แล้ว:
    - ไปที่ **Extensions** (ไอคอนรูปสี่เหลี่ยม 4 ช่อง)
    - ค้นหา Python (โดย Microsoft)
    - คลิก “Install” เพื่อใช้งานภาษา Python

☐ **ทดสอบ Python บน VS Code:**

1. สร้างโฟลเดอร์ใหม่ (เช่น MyPythonProjects)
2. เปิดโฟลเดอร์ใน VS Code
3. สร้างไฟล์ใหม่ชื่อ hello.py
4. พิมพ์โค้ด:
5. print("Hello, Python with VS Code!")
6. คลิกขวา ⟶ “Run Python File in Terminal”

☐ คุณจะเห็นผลลัพธ์ใน Terminal ด้านล่าง

☐ **3. การติดตั้ง PyCharm (ทางเลือก)**

PyCharm คือ IDE ที่เน้นงาน Python โดยเฉพาะ เหมาะกับงานขนาดกลางถึงใหญ่

☐ **ขั้นตอนการติดตั้ง PyCharm:**

1. เข้าเว็บไซต์: ☐ https://www.jetbrains.com/pycharm/
2. คลิก “Download” ⟶ เลือกเวอร์ชัน **Community (ฟรี)**
3. ติดตั้งตามขั้นตอน (Next ⟶ Next ⟶ Install)
4. เมื่อติดตั้งเสร็จ เปิดโปรแกรม แล้ว:
    - คลิก “New Project”
    - ตั้งชื่อโปรเจกต์ ⟶ ตรวจสอบให้แน่ใจว่า Python Interpreter ถูกเลือกถูกต้อง
    - สร้างไฟล์ hello.py แล้วเขียน:
    - print("Hello from PyCharm!")
    - คลิกขวาที่ไฟล์ ⟶ “Run hello”

☐ **สรุป:**

เครื่องมือ	จุดเด่น
Python	ภาษาที่ใช้งานง่ายและทรงพลัง
VS Code	เบา รวดเร็ว ใช้งานได้หลายภาษา
PyCharm	เครื่องมือเฉพาะทางสำหรับ Python

การติดตั้งและใช้งาน Python ผ่าน Google Colab และ Jupyter Notebook

---

□ การติดตั้งและใช้งาน Python บน Google Colab และ Jupyter Notebook

---

□ 1. Google Colab (ไม่ต้องติดตั้ง Python)

□ คืออะไร?

Google Colaboratory (Colab) คือเครื่องมือจาก Google ที่ให้คุณเขียนและรัน Python บนคลาวด์ (Cloud) ได้ฟรี โดยไม่ต้องติดตั้ง Python หรือโปรแกรมใด ๆ บนเครื่อง

□ วิธีใช้งาน Google Colab:

1. ไปที่เว็บไซต์: □ https://colab.research.google.com
2. ถ้ายังไม่ได้ล็อกอิน ให้ล็อกอินด้วยบัญชี Google ของคุณ
3. คลิก “New Notebook” เพื่อเริ่มต้นไฟล์ Python ใหม่ (ไฟล์ .ipynb)
4. จะเห็นหน้าต่างที่แบ่งเป็นเซลล์ (Cell) สามารถเขียนโค้ด Python ได้ เช่น:

```
print("สวัสดีจาก Google Colab")
```

5. กดปุ่ม ▶□ (Run) หรือกด Shift + Enter เพื่อรัน

□ ไฟล์จะถูกเก็บไว้ใน Google Drive โดยอัตโนมัติ

□ ติดตั้งไลบรารีเพิ่มเติม (ผ่าน pip):

ใน Colab สามารถติดตั้งไลบรารีด้วยคำสั่ง:

```
!pip install numpy
```

หรือแม้แต่ไลบรารีขั้นสูง เช่น:

```
!pip install tensorflow
```

□ 2. Jupyter Notebook (ต้องติดตั้งบนเครื่อง)

□ คืออะไร?

Jupyter Notebook คือเครื่องมือในการเขียนโค้ดแบบอินเทอร์แอคทีฟ (Interactive) ที่สามารถรันโค้ด Python แทรกรูปภาพ กราฟ ข้อความ Markdown ได้ในไฟล์เดียว เหมาะกับการสอน, ทดลอง, และทำ Data Science

---

□ **วิธีติดตั้ง Jupyter Notebook:**

□ **วิธีที่ 1: ใช้ Anaconda (แนะนำสำหรับผู้เริ่มต้น)**

Anaconda เป็นแพ็กเกจที่รวม Python + Jupyter + ไลบรารียอดนิยมไว้ครบถ้วน

1. ไปที่เว็บไซต์ □ https://www.anaconda.com/products/distribution
2. ดาวน์โหลด **Anaconda Individual Edition** สำหรับ Windows / macOS / Linux
3. ติดตั้งตามขั้นตอน (Next ⟶ Next ⟶ Finish)
4. หลังติดตั้งเสร็จ ให้เปิด **Anaconda Navigator**
5. คลิกที่ **“Launch” Jupyter Notebook**
   - □ เบราว์เซอร์จะเปิดหน้าต่างที่แสดงไฟล์และโฟลเดอร์ ⟶ คลิก New > Python 3
6. เริ่มเขียนโค้ด Python ได้เลย:

```
print("Hello from Jupyter Notebook")
```

□ **วิธีที่ 2: ติดตั้งผ่าน pip (กรณีไม่ได้ใช้ Anaconda)**

1. เปิด Command Prompt หรือ Terminal แล้วพิมพ์:

```
pip install notebook
```

2. เมื่อติดตั้งเสร็จ ⟶ พิมพ์คำสั่ง:

```
jupyter notebook
```

3. จะเปิดเบราว์เซอร์ที่หน้า Jupyter Notebook โดยอัตโนมัติ

□ **ความสามารถของ Jupyter Notebook:**

- เขียนโค้ด Python และรันแต่ละเซลล์แยกกันได้
- แทรก **ข้อความ Markdown** ได้ เช่น:

```
# หัวข้อใหญ่
## หัวข้อย่อย
**ตัวหนา** *ตัวเอียง*
```

- แสดงกราฟ, ตาราง, ภาพ ได้ในโน้ตบุ๊กเลย

□ **สรุปเปรียบเทียบ**

คุณสมบัติ	Google Colab	Jupyter Notebook

คุณสมบัติ	Google Colab	Jupyter Notebook
ติดตั้ง	☐ ไม่ต้องติดตั้ง	☐ ต้องติดตั้ง
ความสะดวก	ใช้ผ่านเว็บ	ใช้บนเครื่อง
เหมาะสำหรับ	มือใหม่, ใช้ทุกที่	Data Science, งานวิจัย
รองรับ GPU	☐ (ใช้ฟรีจาก Google)	☐ (ต้องมี GPU เอง)

หากเป็น **มือใหม่** ที่ต้องการเริ่มเรียน **ภาษา Python** อย่างรวดเร็วและไม่ยุ่งยากในการติดตั้งซอฟต์แวร์มากมาย — ทางเลือกที่ดีที่สุดคือ **Google Colab**
แต่หากคุณต้องการควบคุมสภาพแวดล้อมได้เองและเรียนรู้การเขียนโค้ดบนเครื่องของคุณ (offline) **Jupyter Notebook (ผ่าน Anaconda)** จะเป็นอีกทางเลือกที่เหมาะสม

☐ **คำแนะนำสำหรับมือใหม่: ใช้ Google Colab จะง่ายและสะดวกที่สุด**

☐ **เหตุผลที่แนะนำ Google Colab:**

จุดเด่น	รายละเอียด
☐ ไม่ต้องติดตั้ง	ใช้งานผ่านเว็บเบราว์เซอร์ได้ทันที
☐ ใช้ได้ทุกที่	แค่มีบัญชี Gmail และอินเทอร์เน็ต
☐ มี GPU ให้ใช้ฟรี	ใช้สำหรับเรียน Machine Learning, AI ได้เลย
☐ เก็บไฟล์บน Google Drive	ปลอดภัยและง่ายต่อการจัดการ
☐ รันโค้ด Python ได้ทันที	เพียงแค่คลิก "Run" ไม่ต้องติดตั้ง Python ในเครื่อง

☐ **เปรียบเทียบ 3 ทางเลือกยอดนิยมสำหรับมือใหม่**

รายการ	Google Colab	Jupyter Notebook (Anaconda)	VS Code
ต้องติดตั้ง?	☐ ไม่ต้อง	☐ ต้องติดตั้ง	☐ ต้องติดตั้ง
เหมาะกับมือใหม่	☐☐☐☐☐	☐☐☐	☐☐
ใช้งานง่าย	☐☐☐☐☐	☐☐☐☐	☐☐
มีโค้ดตัวอย่างในโน้ตบุ๊ก	☐ มี	☐ มี	☐ ไม่มีในตัว
รองรับไลบรารี	☐ เต็มรูปแบบ	☐ เต็มรูปแบบ	☐ เต็มรูปแบบ
ใช้งาน Offline	☐ ไม่ได้	☐ ได้	☐ ได้
รองรับ Markdown +	☐	☐	☐ ต้องติดตั้ง plugin

รายการ	Google Colab	Jupyter Notebook (Anaconda)	VS Code
โค้ด			เพิ่ม
การรันโค้ดแยกเซลล์	☐	☐	☐ (เป็น script ต่อเนื่อง)

☐ **สรุปคำแนะนำ:**

คุณเป็นใคร?	แนะนำใช้
☐☐ มือใหม่เริ่มหัด	☐ **Google Colab**
☐☐ สนใจ Data Science / AI / สอนในห้องเรียน	☐ Google Colab หรือ Jupyter Notebook
☐☐ นักพัฒนาอยากรัน Python บนเครื่องตัวเอง	☐ VS Code หรือ PyCharm
☐ อินเทอร์เน็ตไม่เสถียร	☐ Jupyter Notebook (ติดตั้งผ่าน Anaconda)

☐ **สรุปสุดท้าย (ภาษาง่าย ๆ):**

ถ้าคุณอยากเริ่มตอนนี้เลย ไม่อยากลงโปรแกรมเยอะ และอยากเขียน Python ได้เร็ว ⟶ **ไปที่ Google Colab** แล้วเริ่มเขียน print("Hello, Python!") ได้ทันที! ☐

การเขียนโปรแกรม Python ไฟล์แรก (hello.py)

☐ **1. ความเข้าใจพื้นฐาน**

โปรแกรม Python ไฟล์แรกของคุณมักจะเป็นคำสั่งง่าย ๆ เช่น print("Hello, world!") เพื่อแสดงข้อความบนหน้าจอ เรียกว่า “Hello World Program” ซึ่งเป็นจุดเริ่มต้นของทุกภาษาโปรแกรมทั่วโลก

☐ **2. ขั้นตอนการเขียนและรันโปรแกรม Python ไฟล์แรก (ชื่อ hello.py)**

☐ **A. เตรียมสิ่งต่อไปนี้ก่อน:**

- ติดตั้ง Python แล้ว (ดูวิธีติดตั้งในหัวข้อก่อนหน้า)
- ใช้เครื่องมือใดก็ได้ เช่น:
  - **VS Code** (แนะนำสำหรับมือใหม่)
  - **Notepad / Notepad++**
  - **PyCharm**
  - **Terminal / Command Prompt**
  - หรือแม้แต่บน **Google Colab** (แต่จะใช้ชื่อไฟล์เป็น .ipynb)