สถาบันไอทีจีเนียส
IT GENIUS INSTITUTE

COURSE ID
WDC-88
PROGRAMMING / PYTHON

COURSE OUTLINE · รายละเอียดหลักสูตรอบรม

Python OOP

คิดและออกแบบโค้ด Python แบบ OOP ตั้งแต่ Class จนถึง Design Pattern

- 18 ชั่วโมง
- 3 วัน
- พื้นฐาน **เขียน Python ได้มาแล้วระดับหนึ่ง**
- ระดับ **Intermediate**



ระยะเวลา: 3 วัน
18 ชั่วโมง (3 วัน วันละ 6 ชม.) · 09:00-16:00 น.

รูปแบบ: Classroom / Hybrid
Onsite หรือ Online ผ่าน Microsoft Teams · Public / In-house

รายละเอียดหลักสูตร
www.itgenius.co.th

การเขียนโปรแกรมเชิงวัตถุ (OOP) ช่วยให้โค้ด Python ที่โตขึ้นเรื่อย ๆ ยังอ่านง่าย แก้ไขง่าย และนำกลับมาใช้ซ้ำได้ เพราะแยกข้อมูลกับพฤติกรรมที่เกี่ยวข้องไว้ด้วยกันเป็น Object ที่มีหน้าที่ชัดเจน เฟรมเวิร์กที่ใช้กันมากอย่าง Django, FastAPI และ pytest ก็ออกแบบด้วยแนวคิดนี้ ผู้ที่เข้าใจ OOP จึงอ่านและต่อยอดโค้ดของเฟรมเวิร์กได้เร็วขึ้น

หลักสูตรนี้พาผู้ที่เขียน Python ได้แล้วมาฝึกคิดและออกแบบแบบ Object Oriented ตั้งแต่การแยก Object จากโจทย์ เขียน Class และ Object ใช้หลัก Abstraction, Encapsulation, Inheritance และ Polymorphism ใช้ dataclass, Abstract Class และ Protocol ตรวจชนิดข้อมูลด้วย Type Hint จนถึงหลัก SOLID และ Design Pattern ที่ใช้บ่อยในแบบที่เหมาะกับ Python แล้วปิดท้ายด้วยการปรับสคริปต์เดิมให้เป็นโค้ดแบบ OOP พร้อม Test (อบรม 3 วัน วันละ 6 ชั่วโมง รวม 18 ชั่วโมง ระดับ Intermediate)

เครื่องมือ
- Py Python 3.14
- uv uv
- pt pytest
- my mypy
- DP Design Patterns

ราคาค่าอบรม · PUBLIC / IN-HOUSE

9,500 บาท / ท่าน

สมัครผ่านเว็บไซต์ ลด 10%

ราคายังไม่รวมภาษีมูลค่าเพิ่ม (VAT 7%) · รวมเอกสารประกอบการอบรม Lab Guide ซอร์สโค้ดครบทุก Lab และ Certificate · รับจัด In-house ขอใบเสนอราคาได้

เพิ่มเพื่อน LINE

02-570-8449 · 088-807-9770 · @itgenius · www.itgenius.co.th

01 วัตถุประสงค์ Objectives

- เข้าใจเหตุผลและหลักการคิดแบบ Object Oriented เทียบกับการเขียนแบบ Procedural
- ออกแบบ Class จากโจทย์ และเขียน Class กับ Object ใน Python ได้
- ใช้หลัก Abstraction, Encapsulation, Inheritance และ Polymorphism ได้ถูกที่
- ใช้ dataclass, Dunder Method, Abstract Class และ Protocol ได้
- ตรวจชนิดข้อมูลของโค้ด OOP ด้วย Type Hint และ mypy ได้
- นำหลัก SOLID และ Composition มาออกแบบโค้ดที่แก้ไขง่าย
- เลือกและเขียน Design Pattern ที่ใช้บ่อยในแบบที่เหมาะกับ Python ได้
- เขียน Test ด้วย pytest เพื่อปรับโครงสร้างโค้ดได้อย่างมั่นใจ

02 กลุ่มเป้าหมาย Target Audience

- Developer ที่เขียน Python มาสักระยะและต้องการจัดโครงสร้างโค้ดให้ดีขึ้น
- ผู้ที่ผ่านหลักสูตร Python เบื้องต้นและต้องการต่อยอดไปพัฒนาเว็บแอปพลิเคชัน
- IT Manager และหัวหน้าทีมที่ต้องรีวิวและวางมาตรฐานโค้ด Python
- นักศึกษาและผู้สอนที่ต้องการเข้าใจ OOP และ Design Pattern ด้วย Python

03 จุดเด่นของหลักสูตร Highlights

- **Python 3.14 รุ่นปัจจุบัน** จัดการโปรเจกต์ด้วย uv
- **OOP แบบ Pythonic** dataclass, Protocol และ Duck Typing
- **Type Hint ที่ตรวจได้จริง** Generic แบบใหม่และ mypy
- **Design Pattern ที่ใช้บ่อย** พร้อมทางเลือกแบบ Python
- **ปรับโค้ดจริงพร้อม Test** Workshop Refactor ด้วย pytest

04 ความรู้พื้นฐาน Prerequisites

- เขียนโปรแกรมภาษา Python ได้ เช่น ตัวแปร เงื่อนไข ลูป ฟังก์ชัน List และ Dictionary
- เคยเขียน Python มาสักระยะ หรือผ่านหลักสูตร Python เบื้องต้นมาแล้ว
- ใช้ Command Line และ VS Code เบื้องต้นได้
- นำโน้ตบุ๊กที่ติดตั้งโปรแกรมได้มาด้วย

05 สิ่งที่ต้องเตรียมและเครื่องมือ Setup & Tools

รายการ	รายละเอียด
อุปกรณ์	Notebook เชื่อมต่ออินเทอร์เน็ต พร้อมสิทธิ์ติดตั้งโปรแกรม
ภาษาและเครื่องมือ	Python 3.14, uv, VS Code และ Git
คุณภาพโค้ด	Ruff สำหรับ Lint และ Format, mypy สำหรับตรวจชนิดข้อมูล
การทดสอบ	pytest
ไลบรารีมาตรฐาน	dataclasses, abc, typing, enum และ functools

หมายเหตุ: เครื่องมือทั้งหมดเป็นโอเพนซอร์สหรือใช้ฟรี หลักสูตรนี้ปรับปรุงใหม่ปี 2026 ย้ายมาใช้ Python 3.14 จัดการโปรเจกต์ด้วย uv เพิ่ม dataclass, typing.Protocol, Generic แบบใหม่ของ Python 3.12 ขึ้นไป, Decorator @override และการตรวจชนิดข้อมูลด้วย mypy เพิ่มหลัก SOLID และ Composition และสอน Design Pattern ในแบบที่เหมาะกับ Python เช่น ใช้ฟังก์ชันแทนคลาสใน Strategy และใช้ Module แทน Singleton แทนการแปลงโค้ดแบบ Java มาตรงตัว พร้อม Workshop ปรับโค้ดจริงด้วย pytest

06 ภาพรวมการเรียน 3 วัน Big Picture

วัน	หัวข้อหลัก
Day 1 Section 1-5	เตรียมเครื่องมือด้วย uv, คิดแบบ Object Oriented และออกแบบ Class จากโจทย์จริง, เขียน Class และ Object, ห่อหุ้มข้อมูลด้วย Property แล้วใช้ Dunder Method กับ dataclass
Day 2 Section 6-10	สืบทอด Class และ MRO, Polymorphism กับ Duck Typing, Abstract Class ด้วย abc และ Protocol, Composition กับหลัก SOLID แล้วตรวจชนิดข้อมูลด้วย Type Hint และ mypy
Day 3 Section 11-15	Design Pattern แบบ Pythonic ทั้ง Singleton, Factory Method, Composite, Adapter, Observer, Strategy, Command และ State แล้วปิดท้ายด้วย Workshop ปรับสคริปต์เดิมเป็น OOP พร้อม Test

DAY 1

คิดแบบ OOP และเขียน Class ให้ดี

เนื้อหาการอบรม · Section 1-5

รวม
6 ชั่วโมง

1 Lab: เตรียมเครื่องมือและทำไมต้อง OOP

- ติดตั้ง uv และให้ uv ติดตั้ง Python 3.14 พร้อมสร้างโปรเจกต์
- ตั้งค่า VS Code, Ruff และ pytest สำหรับงานทั้งหลักสูตร
- เขียนแบบ Procedural เทียบกับ OOP ข้อดี ข้อเสีย และงานที่เหมาะ
- ปัญหาของสคริปต์ที่โตขึ้นโดยไม่มีโครงสร้าง
- Lab: อ่านสคริปต์ระบบร้านค้าแบบเดิมและหาจุดที่แก้ยาก

2 คิดแบบ Object และออกแบบ Class

- Object คืออะไร ข้อมูล (State) และพฤติกรรม (Behavior)
- แยก Object และหน้าที่ของแต่ละตัวจากโจทย์
- หลัก Abstraction เลือกเฉพาะสิ่งที่ระบบต้องรู้
- อ่านและวาด Class Diagram แบบ UML เบื้องต้น
- Workshop: ออกแบบ Class ของระบบร้านค้าบนกระดาษ

3 Lab: Class และ Object ใน Python

- ประกาศ Class, เมธอด __init__ และความหมายของ self
- Instance Attribute เทียบกับ Class Attribute
- เมธอดปกติ, @classmethod และ @staticmethod
- ใส่ Type Hint ให้ Attribute และเมธอด
- Lab: Class สินค้า ลูกค้า และตะกร้าสินค้า

4 Lab: Encapsulation และ Property

- หลักการห่อหุ้มข้อมูลและแนวทางตั้งชื่อ _ และ __ ของ Python
- ควบคุมการอ่านและแก้ค่าด้วย @property และ Setter
- ตรวจความถูกต้องของข้อมูลและแจ้ง Error ที่ชัดเจน
- Object ที่แก้ค่าไม่ได้ (Immutable) และเมื่อไรควรใช้
- Lab: ป้องกันราคาติดลบและสต็อกไม่พอในตะกร้าสินค้า

5 Lab: Dunder Method และ dataclass

- Python Data Model กับ __repr__, __str__ และ __eq__
- เปรียบเทียบและเรียงลำดับ Object ด้วย __lt__ และ total_ordering
- ทำ Object ให้วนลูปได้ด้วย __len__ และ __iter__ และใช้กับ with
- @dataclass กับ field, frozen, slots และ kw_only
- Lab: เขียน Class สินค้าใหม่ด้วย dataclass ให้สั้นลง

DAY 2

Inheritance, Polymorphism และการออกแบบที่ยืดหยุ่น

เนื้อหาการอบรม · Section 6-10

รวม 6 ชั่วโมง

6 Lab: Inheritance และ MRO

- สืบทอด Class และเรียก Class แม่ด้วย super()
- Override เมธอดพร้อม Decorator @override จาก typing
- Multiple Inheritance และลำดับการค้นหาเมธอด (MRO)
- Mixin สำหรับเพิ่มความสามารถเล็ก ๆ ให้หลาย Class
- Lab: ลูกค้าทั่วไปกับลูกค้าสมาชิกที่คิดส่วนลดต่างกัน

7 Polymorphism และ Duck Typing

- Polymorphism คืออะไร และการเรียกเมธอดเดียวกันกับต่าง Object
- Duck Typing แนวคิดที่ Python ใช้แทนการบังคับชนิด
- Operator Overloading ด้วย __add__ และเพื่อน
- แยกการทำงานตามชนิดข้อมูลด้วย singledispatchmethod และ match
- Lab: ช่องทางชำระเงินหลายแบบที่เรียกใช้ด้วยโค้ดชุดเดียว

8 Lab: Abstract Class และ Protocol

- Abstract Class ด้วยโมดูล abc, ABC และ @abstractmethod
- ความต่างระหว่าง Class ทั่วไปกับ Abstract Class
- Interface แบบ Python ด้วย typing.Protocol
- ตรวจขณะรันด้วย @runtime_checkable และข้อจำกัด
- Lab: กำหนดสัญญาของระบบจัดส่งสินค้าแล้วเขียนหลายแบบ

9 Composition และหลัก SOLID

- Composition over Inheritance และสัญญาณว่าสืบทอดมากเกินไป
- Single Responsibility และ Open/Closed
- Liskov Substitution, Interface Segregation และ Dependency Inversion
- ส่ง Dependency เข้าทาง Constructor เพื่อให้ทดสอบง่าย
- Workshop: ปรับระบบคิดค่าส่งให้เพิ่มเงื่อนไขใหม่โดยไม่แก้โค้ดเดิม

10 Lab: Type Hint สำหรับโค้ด OOP

- Generic Class แบบใหม่ เช่น class Repository[T] และคำสั่ง type
- Self, ClassVar, Final และ Enum สำหรับค่าคงที่
- การประเมิน Annotation แบบเลื่อนเวลาของ Python 3.14
- ตรวจโค้ดทั้งโปรเจกต์ด้วย mypy และอ่านข้อความเตือน
- Lab: Repository แบบ Generic สำหรับสินค้าและลูกค้า

DAY 3

Design Pattern แบบ Pythonic และ Workshop

เนื้อหาการอบรม · Section 11-15

รวม 6 ชั่วโมง

11 Design Pattern และ Creational Pattern

- Design Pattern คืออะไร ที่มาจาก Gang of Four และ 3 กลุ่มหลัก
- เมื่อไรควรใช้ Pattern และเมื่อไรภาษา Python มีทางที่ง่ายกว่า
- Singleton ด้วย __new__ เทียบกับ Object ระดับ Module และข้อควรระวัง
- Factory Method และ Registry ด้วย Dictionary ของ Class
- Lab: สร้างตัวเชื่อมต่อการชำระเงินจากค่าตั้งค่า

12 Lab: Structural Pattern

- Composite สำหรับข้อมูลแบบต้นไม้ เช่น หมวดหมู่สินค้า
- Adapter เชื่อมไลบรารีภายนอกที่ Interface ไม่ตรงกัน
- Decorator Pattern เทียบกับ Decorator ของ Python
- Facade ซ่อนระบบย่อยที่ซับซ้อนไว้หลังเมธอดเดียว
- Lab: หมวดหมู่สินค้าหลายชั้นและตัวแปลงข้อมูลขนส่ง

13 Lab: Observer และ Strategy

- Observer แจ้งเหตุการณ์ไปยังหลายส่วนโดยไม่ผูกกันแน่น
- Strategy สลับวิธีคำนวณขณะรันโปรแกรม
- ใช้ฟังก์ชันและ Callable แทน Class ในแบบ Python
- เลือกระหว่าง Strategy กับการสืบทอด
- Lab: แจ้งเตือนเมื่อสั่งซื้อ และส่วนลดหลายรูปแบบ

14 Behavioral Pattern อื่นที่ใช้บ่อย

- Command เก็บคำสั่งไว้ทำซ้ำหรือยกเลิก (Undo)
- State เปลี่ยนพฤติกรรมตามสถานะของคำสั่งซื้อ
- Template Method กำหนดขั้นตอนหลักให้ Class ลูกเติม
- Iterator ด้วย Generator และ yield
- Lab: สถานะคำสั่งซื้อตั้งแต่รอชำระเงินจนจัดส่งสำเร็จ

15 Workshop: Refactor สู่ OOP พร้อม Test

- เขียน Test ด้วย pytest และ Fixture ก่อนปรับโครงสร้าง
- ใช้ Test Double แทน Dependency ภายนอก
- ปรับสคริปต์ร้านค้าเดิมเป็น Class และ Pattern ทีละขั้น
- ใช้ AI Assistant ช่วยอธิบายและเสนอการ Refactor โดยตรวจผลเองทุกครั้ง
- Workshop: ระบบร้านค้าแบบ OOP ที่ผ่าน Test, Ruff และ mypy

07 เรียนจบแล้วได้อะไร Outcomes

- **คิดแบบ Object ได้** แยก Object และออกแบบ Class จากโจทย์
- **เขียน Class อย่างมืออาชีพ** Property, Dunder Method และ dataclass
- **ใช้หลัก OOP ครบ** Inheritance, Polymorphism และ Protocol
- **ออกแบบโค้ดที่แก้ง่าย** Composition และหลัก SOLID
- **เลือก Design Pattern ได้** Creational, Structural และ Behavioral
- **Refactor อย่างมั่นใจ** pytest, Ruff และ mypy

08 สิ่งที่ผู้เรียนจะได้รับ Deliverables

- เอกสารประกอบการอบรมครบทั้ง 3 วัน และ Lab Guide ทีละขั้นตอน
- ซอร์สโค้ดระบบร้านค้าแบบ OOP ครบทุก Lab พร้อม Test
- ตัวอย่าง Design Pattern แบบ Python ที่นำไปใช้ต่อได้
- ประกาศนียบัตรจบหลักสูตร (Certificate of Completion) และช่องทางสอบถามวิทยากรหลังจบหลักสูตร

09 ข้อมูลผู้สอน Instructor

อ.สามิตร โกยม ผู้สอนและนักพัฒนาที่ใช้ Python สร้างระบบงานจริงและดูแลโค้ดที่ทีมหลายคนแก้ร่วมกัน สอนผ่านระบบร้านค้าชุดเดียวที่ค่อย ๆ ปรับจากสคริปต์ให้เป็นโค้ดแบบ OOP อธิบายเหตุผลของแต่ละหลักการและแต่ละ Pattern ว่าใช้เมื่อไรและไม่ควรใช้เมื่อไร แล้วพาลงมือเขียนจนผู้เรียนออกแบบโค้ดของตัวเองได้

ราคาค่าอบรม 9,500 บาท/ท่าน · สนใจ IN-HOUSE ขอใบเสนอราคา

9,500 บาท / ท่าน (ยังไม่รวม VAT 7%)

สมัครผ่านเว็บไซต์ ลด 10%

02-570-8449 · 088-807-9770

@itgenius · contact@itgenius.co.th

เพิ่มเพื่อน LINE