เอกสารประกอบการสอน

รายวิชา SIT304 หลักการเขียนโปรแกรมเชิงวัตถุ
(Object-Oriented Programming Principles)

อาจารย์ไชย มีหนองหว้า
หลักสูตรเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี

มหาวิทยาลัยราชภัฏวไลยอลงกรณ์ ในพระบรมราชูปถัมภ์
2569

หนังสือ เอกสารประกอบการสอน รายวิชา SIT304 หลักการเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming Principles)

เขียนโดย ดร.ไชย มีหนองหว้า

ปีที่เผยแพร่ กรกฎาคม พ.ศ. 2569

พิมพ์ครั้งที่ 1 (หนังสืออิเล็กทรอนิกส์ / e-Book Edition)

ข้อมูลทางบรรณานุกรมของหอสมุดแห่งชาติ

ไชย มีหนองหว้า.
เอกสารประกอบการสอน รายวิชา SIT304 หลักการเขียนโปรแกรมเชิงวัตถุ = Object-Oriented Programming Principles.-- ปทุมธานี : [ม.ป.พ.], 2569.
172 หน้า.

1. การโปรแกรมเชิงวัตถุ. 2. การเขียนโปรแกรม (คอมพิวเตอร์). I. ชื่อเรื่อง.

005.117
ISBN 978-616-636-833-8

จัดทำโดย ดร.ไชย มีหนองหว้า
77/17 เดอะทรัสต์ กาญจนาภิเษก-หทัยราษฎร์
ซอย 3 หมู่ที่ 17 ถนนหทัยราษฎร์ ตำบลบึงคำพร้อย อำเภอลำลูกกา จังหวัดปทุมธานี 12150

(สงวนลิขสิทธิ์ตามพระราชบัญญัติลิขสิทธิ์ พ.ศ. 2550 และฉบับปรับปรุงเพิ่มเติม ห้ามลอกเลียนแบบ คัดลอก รีโปรดักชัน หรือดัดแปลงส่วนใดส่วนหนึ่งของเนื้อหาและซอร์สโค้ดในหนังสือเล่มนี้ ไม่ว่าจะด้วยกระบวนการทางอิเล็กทรอนิกส์หรือทางกายภาพใด ๆ เว้นแต่จะได้รับอนุมัติเป็นลายลักษณ์อักษรจากเจ้าของลิขสิทธิ์อย่างเป็นทางการเท่านั้น)

คำนำ

เอกสารประกอบการสอนรายวิชา SIT304 หลักการเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming Principles) เล่มนี้ เรียบเรียงขึ้นเพื่อใช้เป็นเครื่องมือหลักในการจัดการเรียนการสอนสำหรับนักศึกษาในระดับปริญญาตรี หลักสูตรเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏวไลยอลงกรณ์ ในพระบรมราชูปถัมภ์ ซึ่งวิชานี้ถือเป็นหมวดวิชาบังคับทางแกนหลักที่นักศึกษาหลักสูตรเทคโนโลยีสารสนเทศทุกคน จำเป็นต้องมีความรู้ความเชี่ยวชาญอย่างลึกซึ้ง เพื่อนำไปใช้เป็นรากฐานในการก้าวสู่การเป็นนักพัฒนาระบบมืออาชีพในระดับสากล

เนื้อหาภายในเอกสารเล่มนี้ได้รับการออกแบบและปรับปรุงให้มีความทันสมัยสอดคล้องกับแนวทางการพัฒนาซอฟต์แวร์ในยุคปัจจุบัน โดยเปลี่ยนผ่านวิธีคิดจากการเขียนโปรแกรมเชิงกระบวนคำสั่งแบบดั้งเดิม (Procedural Programming) ไปสู่กระบวนทัศน์เชิงวัตถุ (Object-Oriented Paradigm) อย่างเป็นลำดับขั้นตอน มุ่งเน้นการปูพื้นฐานการจัดการข้อมูลและตรรกะในภาษา Python บนแพลตฟอร์ม Google Colab ผ่านคำอธิบายที่เข้าใจง่าย ควบคู่ไปกับแบบจำลองโครงสร้างและแผนภาพลายเส้นเชิงตรรกะ (Schematic Diagram) ที่ช่วยให้ผู้เรียนมองเห็นภาพการไหลเวียนของข้อมูลและการจัดสรรพื้นที่ในหน่วยความจำ (RAM) ได้ด้วยตนเองอย่างราบรื่น

นอกจากนี้ ตัวเล่มยังครอบคลุมไปถึงเสาหลักทางสถาปัตยกรรมซอฟต์แวร์ขั้นสูง ไม่ว่าจะเป็นการห่อหุ้มข้อมูล (Encapsulation) การสืบทอดคุณลักษณะ (Inheritance) การพหุสัณฐาน (Polymorphism) และการจัดทำนามธรรม (Abstraction) เพื่อมุ่งสู่การออกแบบระบบในลักษณะ Plug-and-Play รวมถึงการบูรณาการระบบนิรภัยดักจับข้อผิดพลาด (Exception Handling) และการเชื่อมโยงความสัมพันธ์ระหว่างวัตถุ (Composition & Aggregation) เพื่อสร้างระบบซอฟต์แวร์ที่มีความเสถียร ทนทาน และง่ายต่อการขยายตัวในอนาคต โดยซอร์สโค้ดโปรแกรมและกิจกรรมปฏิบัติการ (Hands-on Lab) ทุกตัวอย่าง ได้รับการจัดฟอร์แมตให้มีเลขกำกับบรรทัดพร้อมคำอธิบายแบบบรรทัดต่อบรรทัดอย่างละเอียดถี่ถ้วน เพื่อเอื้ออำนวยความสะดวกสูงสุดให้แก่ผู้เรียนในการศึกษาค้นคว้าด้วยตนเอง (Self-paced Learning)

ผู้เขียนหวังเป็นอย่างยิ่งว่า เอกสารประกอบการสอนเล่มนี้จะเป็นประโยชน์และเป็นคู่มือคู่ใจที่ทรงคุณค่าสำหรับนักศึกษา ตลอดจนผู้สนใจทั่วไปในการทลายกำแพงความรู้ และเปิดประตูบานใหญ่สู่โลกแห่งการออกแบบและพัฒนาซอฟต์แวร์เชิงวัตถุระดับมืออาชีพได้อย่างยั่งยืน

อาจารย์ไชย มีหนองหว้า
กรกฎาคม 2569

สารบัญ

เนื้อหา **หน้าที่**

คำนำ 2

สารบัญ 3

แผนบริหารการสอนประจำรายวิชา (ตามแบบ มคอ.3) 13

บทที่ 1 17

แนะนำ GOOGLE COLAB **และทบทวนพื้นฐานภาษา** PYTHON 17

(INTRODUCTION TO COLAB & PYTHON REVIEW) 17

แผนบริหารการสอนประจำบทที่ 1 17

จำนวนชั่วโมงที่สอน *17*

หัวข้อเนื้อหา *17*

คำสำคัญ *17*

วัตถุประสงค์เชิงพฤติกรรม *17*

หน่วยการเรียนที่ 1.1 การใช้งาน GOOGLE COLAB เบื้องต้น 17

โครงสร้างและการทำงานของหน้าต่าง Google Colab *18*

หน่วยการเรียนที่ 1.2 ทบทวนการเขียนฟังก์ชัน (FUNCTIONS) 18

โครงสร้างไวยากรณ์และการนับบรรทัด *18*

หน่วยการเรียนที่ 1.3 โครงสร้างข้อมูลแบบ LIST (รายการ) 19

ตัวอย่างการเขียนโค้ดและการดึงข้อมูล *19*

หน่วยการเรียนที่ 1.4 โครงสร้างข้อมูลแบบ DICTIONARY (พจนานุกรม) 20

ตัวอย่างการเขียนโค้ดและการจับคู่ข้อมูล *20*

ประกอบร่าง: สถาปัตยกรรมข้อมูลที่ทำงานร่วมกันแบบเต็มระบบ *21*

คำถามทบทวน / กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 22

โจทย์กิจกรรมปฏิบัติการ *22*

เฉลยกิจกรรมปฏิบัติประจำบท (Lab Assignment) *23*

ซอร์สโค้ดภาษา Python สำเร็จรูป *23*

ตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 24
ตัวอย่างที่ 1 24
ตัวอย่างที่ 2 25
ตัวอย่างที่ 3 26
ตัวอย่างที่ 4 26
ตัวอย่างที่ 5 27
เอกสารอ้างอิงประจำบท 28

บทที่ 2 30

แนวคิดและหลักการเขียนโปรแกรมเชิงวัตถุ 30

(CONCEPTS AND PRINCIPLES OF OBJECT-ORIENTED PROGRAMMING) 30

แผนบริหารการสอนประจำบทที่ 2 30
หน่วยการเรียนที่ 2.1 ทำไมต้องเปลี่ยนมาเป็น OOP? (ขีดจำกัดของโปรแกรมแบบดั้งเดิม) 30
หน่วยการเรียนที่ 2.2 ความสัมพันธ์ระหว่าง คลาส (CLASS) และ อ็อบเจกต์ (OBJECT) 31
หน่วยการเรียนที่ 2.3 ส่วนประกอบภายในวัตถุ: ATTRIBUTES และ METHODS 31
หน่วยการเรียนที่ 2.4 การเขียนโค้ดสร้างคลาสแรกในภาษา PYTHON 32
ตัวอย่างซอร์สโค้ดระบบจัดการข้อมูลรถยนต์ 32
กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 33
โจทย์กิจกรรมปฏิบัติการ 33
เฉลยกิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 34
ซอร์สโค้ดภาษา Python 34
ชุดตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 36
ตัวอย่างที่ 1: การเข้าถึงข้อมูลระดับ Public (สาธารณะ) 36
ตัวอย่างที่ 2: ตัวแปรระดับ Protected (การใช้ขีดล่าง 1 ตัว) 37
ตัวอย่างที่ 3: ตัวแปรระดับ Private (การใช้ขีดล่าง 2 ตัว) 38
ตัวอย่างที่ 4: กลไก Name Mangling เบื้องหลังระบบ Private 38
ตัวอย่างที่ 5: การใช้งาน @property เพื่อควบคุมข้อมูลอย่างมืออาชีพ (Pythonic Way) 39
เอกสารอ้างอิงประจำบท 40

บทที่ 3 41

การสร้างคลาสและอ็อบเจกต์ในภาษา PYTHON 41

(CREATING CLASSES AND OBJECTS IN PYTHON) 41

แผนบริหารการสอนประจำบทที่ 3 41

หัวข้อเนื้อหา *41*

คำสำคัญ (Keywords) *41*

วัตถุประสงค์เชิงพฤติกรรม *41*

หน่วยการเรียนที่ 3.1: แนวคิดของคลาสและอ็อบเจกต์ในโลกจริง 41

หน่วยการเรียนที่ 3.2: การสร้างคลาสและการกำหนดคุณลักษณะใน PYTHON 42

โครงสร้างไวยากรณ์เบื้องต้น *42*

หน่วยการเรียนที่ 3.3: กลไกของคอนสตรัคเตอร์(CONSTRUCTOR) และเมธอด __INIT__ 43

หน่วยการเรียนที่ 3.4: หน้าที่และบทบาทของพารามิเตอร์ SELF 43

โครงสร้างไวยากรณ์เมื่อประยุกต์ใช้ร่วมกัน *44*

หน่วยการเรียนที่ 3.5: ตารางเปรียบเทียบความแตกต่างเชิงพฤติกรรม 44

กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 45

เฉลยกิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 45

ซอร์สโค้ดภาษา Python สำเร็จรูป *45*

เจาะลึกตรรกะและการประมวลผลโค้ดตัวอย่างทีละบรรทัด: *46*

คำถามทบทวน/กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 47

ส่วนที่ 1 การจำลองข้อมูลนักศึกษา (Student Class) *47*

ส่วนที่ 2 การจำลองระบบร้านหนังสือ (Book Class) *47*

เฉลยกิจกรรมปฏิบัติประจำบท (สำหรับผู้สอน) *47*

ตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 49

ตัวอย่างที่ 1 ..... *49*

ตัวอย่างที่ 2 ..... *50*

ตัวอย่างที่ 3 ..... *51*

ตัวอย่างที่ 4 ..... *52*

ตัวอย่างที่ 5 ..... *53*

เอกสารอ้างอิงประจำบท 54

บทที่ 4 55

ความสัมพันธ์ระหว่างคลาสและการสืบทอดคุณสมบัติ 55

(CLASS RELATIONSHIPS AND INHERITANCE) 55

แผนบริหารการสอนประจำบทที่ 4 55

หัวข้อเนื้อหา *55*

คำสำคัญ (Keywords) *55*

วัตถุประสงค์เชิงพฤติกรรม *55*

หน่วยการเรียนที่ 4.1: โครงสร้างความสัมพันธ์ระหว่างคลาส 55

1. ความสัมพันธ์แบบมีส่วนประกอบ (Composition / Aggregation) *55*

2. ความสัมพันธ์แบบสืบทอดคุณสมบัติ (Inheritance) *56*

หน่วยการเรียนที่ 4.2: กลไกการสืบทอดคุณสมบัติ (INHERITANCE) 56

ตัวอย่างโค้ด: การสร้างคลาสแม่และคลาสลูกพื้นฐาน *56*

หน่วยการเรียนที่ 4.3: การใช้งานคำสั่ง SUPER() และการเพิ่มคุณสมบัติเฉพาะตัว 57

ตัวอย่างโค้ด: การใช้ super() เพื่อสร้างคอนสตรักเตอร์ขยายความสามารถ *57*

หน่วยการเรียนที่ 4.4: การเขียนทับเมธอด (METHOD OVERRIDING) 58

ตัวอย่างโค้ด: การเขียนทับเมธอดและการนำเมธอดเดิมมาต่อยอด *58*

กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 59

เฉลยกิจกรรมปฏิบัติประจำบท (Lab Assignment) *60*

คำถามทบทวน/กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 62

เฉลยกิจกรรมปฏิบัติประจำบท (สำหรับผู้สอน) 63

ชุดตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 64

ตัวอย่างที่ 1 ..... *64*

ตัวอย่างที่ 2 ..... *66*

ตัวอย่างที่ 3 ..... *67*

ตัวอย่างที่ 4 ..... *68*

ตัวอย่างที่ 5 ..... *69*

เอกสารอ้างอิงประจำบท 70

บทที่ 5 71

การสืบทอดคุณลักษณะและความสัมพันธ์ระหว่างคลาส 71

(INHERITANCE AND CLASS RELATIONSHIPS) 71
แผนบริหารการสอนประจำบทที่ 5 71
หัวข้อเนื้อหา *71*
คำสำคัญ (Keywords) *71*
วัตถุประสงค์เชิงพฤติกรรม *71*
หน่วยการเรียนที่ 5.1: แนวคิดและหลักการของการสืบทอดคุณลักษณะ 71
ทำไมต้องมีการสืบทอด? *71*
หน่วยการเรียนที่ 5.2: การสร้างคลาสแม่และคลาสลูกในภาษา PYTHON 72
ตัวอย่างไวยากรณ์พื้นฐาน *72*
หน่วยการเรียนที่ 5.3: การเขียนทับเมธอด (METHOD OVERRIDING) และการใช้ SUPER() 73
ตัวอย่างโปรแกรมการต่อยอดแอตทริบิวต์และพฤติกรรม *73*
กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 75
เฉลยกิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 75
ซอร์สโค้ดภาษา Python สำเร็จรูป *75*
ชุดตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 78
ตัวอย่างที่ 1 ..... *78*
ตัวอย่างที่ 2 ..... *80*
ตัวอย่างที่ 3 ..... *81*
ตัวอย่างที่ 4 ..... *82*
ตัวอย่างที่ 5 ..... *84*
เอกสารอ้างอิงประจำบท 85
บทที่ 6 86
การพหุสัณฐานและการกำหนดรูปแบบนามธรรม 86
(POLYMORPHISM AND ABSTRACT DESIGN) 86
หัวข้อเนื้อหา *86*
คำสำคัญ (Keywords) *86*
วัตถุประสงค์เชิงพฤติกรรม *86*
หน่วยการเรียนที่ 6.1: แนวคิดของ POLYMORPHISM (การพหุสัณฐาน) 86

พหุสัณฐาน คืออะไรในโลกของโปรแกรมเมอร์? *86*
ลองเปรียบเทียบแนวคิด: สลัดคราบ If-Else สู่พลังพหุสัณฐาน *87*
ตัวอย่างโค้ดที่ 1: รูปแบบดั้งเดิมที่ผูกมัดแน่น (Procedural Approach - Code Smell) *87*
อธิบายทีละบรรทัด *89*
หน่วยการเรียนที่ 6.2: กลไก METHOD OVERRIDING (การเขียนทับเมธอด) 89
ตัวอย่างโค้ดการทำ Method Overriding *89*
อธิบายทีละบรรทัด *90*
หน่วยการเรียนที่ 6.3: นามธรรมด้วย ABSTRACT CLASS และ ABSTRACT METHOD 91
การทำสัญญาผูกมัดโครงสร้างซอฟต์แวร์ (The Software Contract) *91*
ตัวอย่างซอร์สโค้ดการทำพิมพ์เขียวเชิงนามธรรมที่สมบูรณ์ *91*
อธิบายทีละบรรทัด *92*
สถาปัตยกรรม PLUG-AND-PLAY ในชีวิตจริง (THE CLEAN ARCHITECTURE) 93
กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 93
โจทย์กิจกรรมปฏิบัติการ *93*
เฉลยกิจกรรมปฏิบัติประจำบท (Lab Assignment) *94*
ซอร์สโค้ดภาษา Python ฉบับสมบูรณ์และได้มาตรฐาน *94*
อธิบายทีละบรรทัด *96*
ชุดตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 97
ตัวอย่างที่ 1 ..... *97*
อธิบายทีละบรรทัด *98*
ตัวอย่างที่ 2 ..... *99*
อธิบายทีละบรรทัด *100*
ตัวอย่างที่ 3 ..... *100*
อธิบายทีละบรรทัด *101*
ตัวอย่างที่ 4 ..... *102*
อธิบายทีละบรรทัด *103*
ตัวอย่างที่ 5 ..... *103*
อธิบายทีละบรรทัด *104*
บทที่ 7 106

การจัดการข้อผิดพลาดและข้อยกเว้นในโครงสร้างเชิงวัตถุ 106

(EXCEPTION HANDLING IN OBJECT-ORIENTED STRUCTURES) 106

แผนบริหารการสอนประจำบทที่ 7 106

หัวข้อเนื้อหา *106*

คำสำคัญ (Keywords) *106*

วัตถุประสงค์เชิงพฤติกรรม *106*

หน่วยการเรียนที่ 7.1: ปรัชญาความเสี่ยงและประเภทข้อผิดพลาดในระบบซอฟต์แวร์ 106

หน่วยการเรียนที่ 7.2: โครงสร้างการควบคุมความเสี่ยงแบบ 4 ประสาน 107

อธิบายทีละบรรทัด *108*

หน่วยการเรียนที่ 7.3: การสร้างคลาสข้อยกเว้นขึ้นใช้งานเอง (CUSTOM EXCEPTION) 108

อธิบายทีละบรรทัด *109*

หน่วยการเรียนที่ 7.4: การผสานกลไก EXCEPTION HANDLING ในสถาปัตยกรรม OOP เต็มรูปแบบ .. 109

อธิบายทีละบรรทัด *111*

กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 112

โจทย์กิจกรรม *112*

เฉลยกิจกรรมปฏิบัติประจำบท (Lab Assignment) *112*

ซอร์สโค้ดภาษา Python *112*

อธิบายทีละบรรทัด *114*

ชุดตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 114

ตัวอย่างที่ 1 ..... *115*

อธิบายทีละบรรทัด *116*

ตัวอย่างที่ 2 ..... *116*

อธิบายทีละบรรทัด *118*

ตัวอย่างที่ 3 ..... *118*

อธิบายทีละบรรทัด *120*

ตัวอย่างที่ 4 ..... *120*

อธิบายทีละบรรทัด *122*

ตัวอย่างที่ 5 ..... *123*

อธิบายทีละบรรทัด *124*

บทที่ 8 126

การประยุกต์ใช้คลาสนามธรรมและอินเทอร์เฟซในระบบจริง 126

(PRACTICAL ABSTRACT CLASSES & INTERFACES IN PRODUCTION) 126

แผนบริหารการสอนประจำบทที่ 8 126

หัวข้อเนื้อหา *126*

คำสำคัญ (Keywords) *126*

วัตถุประสงค์เชิงพฤติกรรม *126*

หน่วยการเรียนที่ 8.1: จากทฤษฎีเชิงนามธรรม สู่สถาปัตยกรรมระบบจริง 126

หน่วยการเรียนที่ 8.2: การก้าวข้ามคำสั่งเงื่อนไขด้วยระบบสัญญาสากล (CONTRACT) 127

หน่วยการเรียนที่ 8.3: กิจกรรมปฏิบัติการและซอร์สโค้ดระดับสถาปัตยกรรม (PAYMENT HUB IMPLEMENTATION) 127

ซอร์สโค้ดระบบประมวลผลธุรกรรมทางการเงิน (Production-Ready Template) *127*

อธิบายทีละบรรทัด *131*

คำถามทบทวน / กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 131

โจทย์กิจกรรมปฏิบัติการ *131*

เฉลยกิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 132

แนวคำตอบ (ซอร์สโค้ดภาษา Python สมบูรณ์) *132*

ชุดตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 134

ตัวอย่างที่ 1 ..... *134*

อธิบายทีละบรรทัด *135*

ตัวอย่างที่ 2 ..... *135*

อธิบายทีละบรรทัด *136*

ตัวอย่างที่ 3 ..... *136*

อธิบายทีละบรรทัด *137*

ตัวอย่างที่ 4 ..... *137*

อธิบายทีละบรรทัด *138*

ตัวอย่างที่ 5 ..... *139*

อธิบายทีละบรรทัด *139*

เอกสารอ้างอิงประจำบท 140

บทที่ 9 141

การจัดการข้อผิดพลาดในระบบเชิงวัตถุ 141

แผนบริหารการสอนประจำบทที่ 9 141

หัวข้อเนื้อหา *141*

คำสำคัญ (Keywords) *141*

วัตถุประสงค์เชิงพฤติกรรม *141*

หน่วยการเรียนที่ 9.1: แนวคิดเรื่องข้อผิดพลาดในโลกซอฟต์แวร์ 142

หน่วยการเรียนที่ 9.2: โครงสร้างบล็อกคำสั่ง TRY-EXCEPT-ELSE-FINALLY 142

อธิบายทีละบรรทัด *143*

หน่วยการเรียนที่ 9.3: การโยนข้อผิดพลาดและการสร้าง CUSTOM EXCEPTION (คลาสข้อผิดพลาดส่วนตัว) 143

ตัวอย่างสถาปัตยกรรมรหัสโค้ดระบบฝากถอนเงินอย่างปลอดภัย *144*

อธิบายทีละบรรทัด *145*

กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 145

เฉลยกิจกรรมปฏิบัติประจำบท (Lab Assignment) *146*

ซอร์สโค้ดภาษา Python สำเร็จรูป *146*

อธิบายทีละบรรทัด *148*

ชุดตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 149

ตัวอย่างที่ 1 ..... *149*

อธิบายทีละบรรทัด *149*

ตัวอย่างที่ 2 ..... *150*

อธิบายทีละบรรทัด *151*

ตัวอย่างที่ 3 ..... *151*

อธิบายทีละบรรทัด *152*

ตัวอย่างที่ 4 ..... *152*

อธิบายทีละบรรทัด *153*

ตัวอย่างที่ 5 ..... *153*

อธิบายทีละบรรทัด *154*

เอกสารอ้างอิงประจำบท 155

บทที่ 10 156

ความสัมพันธ์ระหว่างวัตถุและการออกแบบระบบ(OBJECT RELATIONSHIPS AND SYSTEM DESIGN) 156

แผนบริหารการสอนประจำบทที่ 10 156

หัวข้อเนื้อหา *156*

คำสำคัญ (Keywords) *156*

วัตถุประสงค์เชิงพฤติกรรม *156*

หน่วยการเรียนที่ 10.1: แนวคิดความสัมพันธ์ระหว่างวัตถุในโลกจริงสู่สถาปัตยกรรมซอฟต์แวร์ 157

โครงสร้างโค้ดจำลองระบบจัดการโครงข่ายเซนเซอร์ (Sensor Hub Network) *157*

อธิบายทีละบรรทัด *159*

คำถามทบทวน / กิจกรรมปฏิบัติประจำบท (LAB ASSIGNMENT) 160

เฉลยกิจกรรมปฏิบัติประจำบท (Lab Assignment) *161*

ซอร์สโค้ดภาษา Python สำเร็จรูป *161*

จำลองผลลัพธ์การรันหน้าจอโปรแกรม (Output Cases) *162*

อธิบายทีละบรรทัด *162*

ชุดตัวอย่างโปรแกรมเพื่อการเรียนรู้ด้วยตนเอง 163

ตัวอย่างที่ 1 ..... *163*

อธิบายทีละบรรทัด *164*

ตัวอย่างที่ 2 ..... *165*

อธิบายทีละบรรทัด *166*

ตัวอย่างที่ 3 ..... *166*

อธิบายทีละบรรทัด *168*

ตัวอย่างที่ 4 ..... *168*

อธิบายทีละบรรทัด *169*

ตัวอย่างที่ 5 ..... *170*

อธิบายทีละบรรทัด *171*

เอกสารอ้างอิงประจำบท 172

แผนบริหารการสอนประจำรายวิชา (ตามแบบ มคอ.3)

รายวิชา SIT304 หลักการเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming Principles)

คำอธิบายรายวิชา

ศึกษาแนวคิดและกระบวนทัศน์การเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming) ด้วยภาษา Python องค์ประกอบหลักของคลาส (Class) อ็อบเจกต์ (Object) คุณลักษณะ (Attributes) และพฤติกรรม (Methods) รวมถึงคุณสมบัติหลัก 4 ประการ ได้แก่ การห่อหุ้ม (Encapsulation) การสืบทอด (Inheritance) การพหุสัณฐาน (Polymorphism) และการจัดทำนามธรรม (Abstraction) การออกแบบโครงสร้างควบคุมเงื่อนไขด้วยหลักการ Polymorphism และ Design Patterns เบื้องต้น

จุดมุ่งหมายรายวิชา

1. เพื่อให้นักศึกษามีความรู้ความเข้าใจเกี่ยวกับแนวคิดและปรัชญาของการเขียนโปรแกรมเชิงวัตถุ
2. เพื่อให้นักศึกษาสามารถสร้าง Class, Object และประยุกต์ใช้คุณสมบัติหลักของ OOP ในภาษา Python ได้อย่างถูกต้องตามมาตรฐานสากล
3. เพื่อให้นักศึกษาสามารถปรับปรุงแก้ไขโค้ด (Refactoring) เพื่อกำจัดโครงสร้างเงื่อนไขที่ซับซ้อน (Code Smell) ให้กลายเป็นระบบโมดูลาร์ที่สะอาดและปลอดภัย
4. เพื่อให้นักศึกษาสามารถวิเคราะห์ ออกแบบ และดักจับข้อผิดพลาดของระบบ รวมถึงเชื่อมโยงวัตถุข้ามสายเลือดเพื่อแก้ไขปัญหาโจทย์เชิงธุรกิจและเทคโนโลยีในชีวิตจริงได้

วิธีสอนและกิจกรรม

เน้นการบรรยายภาคทฤษฎีเพื่อปรับกระบวนทัศน์ (Mindset) ควบคู่ไปกับการปฏิบัติการเขียนโปรแกรมจริงผ่านเครื่องคอมพิวเตอร์บนแพลตฟอร์ม Google Colab (Hands-on Lab) นำเสนอกรณีศึกษาจากระบบงานจริงบนโปรดักชัน (เช่น Payment Gateway และ Smart Farming) การทำกิจกรรมอภิปรายกลุ่ม (Group Discussion) และการมอบหมายโครงการออกแบบระบบเชิงวัตถุ เพื่อส่งเสริมการเรียนรู้ด้วยตนเองอย่างยั่งยืน

โครงสร้างแผนการประเมินผลสัมฤทธิ์ทางการเรียน

การวัดและประเมินผลในรายวิชานี้ จะเปลี่ยนผ่านจากเกณฑ์แบบอิงคะแนนดิบดั้งเดิม มาเป็นการประเมินผลตามสภาพจริงเพื่อวัดวิวัฒนาการของผู้เรียนในแต่ละมิติ โดยแบ่งสัดส่วนคะแนนดังนี้

คะแนนเก็บระหว่างภาคเรียน (Formative Assessment): 60%

- การมีส่วนร่วมในชั้นเรียนและกิจกรรมกลุ่ม (Class Participation): 10%
- งานปฏิบัติการประจำบทเรียน (Lab Assignments on Google Colab): 30%
- โครงงานออกแบบสถาปัตยกรรมซอฟต์แวร์กลุ่ม (OOP Project): 20%

คะแนนประเมินผลรวมรวบปลายภาค (Summative Assessment): 40%

- การสอบวัดความรู้ภาคทฤษฎีและปฏิบัติกลางภาค (Midterm Exam): 20%
- การสอบวัดความรู้ภาคทฤษฎีและปฏิบัติปลายภาค (Final Exam): 20%

เกณฑ์การประเมินผลในรูปแบบ Rubric Score (วัดวิวัฒนาการทักษะการเขียนโปรแกรม)

เพื่อให้ผู้เรียนทราบถึงระดับพัฒนาการและวิวัฒนาการทางทักษะการเขียนโปรแกรมเชิงวัตถุของตนเองอย่างละเอียด ตัวชี้วัดจะถูกแบ่งออกเป็น 4 มิติหลัก โดยมีเกณฑ์การให้คะแนนแบบแจกแจงรายละเอียด (Analytic Rubrics) ดังนี้

มิติที่ 1: ความถูกต้องทางไวยากรณ์และการจัดการข้อผิดพลาด (Syntax & Exception Robustness)

- **ระดับดีมาก (4 คะแนน)**: โค้ดถูกหลักไวยากรณ์ Python 100% ไม่เกิดข้อผิดพลาดขณะรันโปรแกรม (Runtime Error) มีการติดตั้งบล็อกคำสั่ง try-except-else-finally ดักจับเหตุการณ์ไม่คาดฝันได้อย่างครอบคลุม และมีการใช้ Custom Exception แยกแยะเงื่อนไขทางธุรกิจอย่างเป็นระบบ
- **ระดับดี (3 คะแนน)**: โค้ดถูกหลักไวยากรณ์ รันผ่านได้ปกติ มีการใช้คำสั่ง try-except ดักจับข้อผิดพลาดพื้นฐานได้ดี แต่อาจขาดการจัดการในบล็อก else หรือ finally ในจุดที่ควรเคลียร์ ทรัพยากรระบบ
- **ระดับพอใช้ (2 คะแนน)**: โค้ดมีไวยากรณ์ที่ถูกต้องเป็นส่วนใหญ่ แต่ระบบมักพังล่ม (Crash) เมื่อผู้ใช้งานป้อนข้อมูลผิดประเภท หรือขาดการติดตั้งเกราะป้องกันข้อผิดพลาดในเมธอดสำคัญ
- **ระดับปรับปรุง (1 คะแนน)**: โค้ดเกิดข้อผิดพลาดเชิงไวยากรณ์ (Syntax Error) ทำให้ตัวแปลภาษาปฏิเสธการรันโปรแกรม หรือไม่มีการเขียนดักจับข้อผิดพลาดใด ๆ เลย

มิติที่ 2: การออกแบบโครงสร้างและการห่อหุ้มข้อมูล (Encapsulation & Architecture Design)

- **ระดับดีมาก (4 คะแนน)**: มีการแยกแยะคลาสและอ็อบเจกต์สะท้อนความจริงได้สมบูรณ์ เลือกใช้ระดับการเข้าถึงแอตทริบิวต์แบบ Private (__) เพื่อซ่อนข้อมูลสำคัญอย่างเข้มงวด และเข้าใช้งานผ่าน @property (Getter/Setter) ที่มีการตรวจสอบความถูกต้องของข้อมูล (Data Validation) ก่อนบันทึกจริง
- **ระดับดี (3 คะแนน)**: มีการห่อหุ้มข้อมูลโดยใช้ตัวแปรแบบ Private หรือ Protected แต่เมธอด Setter ยังขาดการเขียนเงื่อนไขตรวจสอบความปลอดภัยของข้อมูล หรือเขียนสลับไปมา
- **ระดับพอใช้ (2 คะแนน)**: ประกาศตัวแปรทุกตัวภายในคลาสเป็นแบบ Public ทั้งหมด ทำให้โค้ดภายนอกสามารถเข้ามาแก้ไขค่าแอตทริบิวต์ได้ตรง ๆ โดยไม่มีการควบคุมความปลอดภัย
- **ระดับปรับปรุง (1 คะแนน)**: ไม่เข้าใจความสัมพันธ์ระหว่างคลาสและอ็อบเจกต์ วางโครงสร้างตัวแปรสะเปะสะปะไม่ต่างจากการเขียนโปรแกรมเชิงกระบวนคำสั่งแบบดั้งเดิม

มิติที่ 3: การปฏิวัติความซ้ำซ้อนและการนำโค้ดกลับมาใช้ใหม่ (Inheritance & Code Reusability)

- **ระดับดีมาก (4 คะแนน)**: ออกแบบโครงสร้างสายเลือดคลาสลูกสืบทอดคุณสมบัติจากคลาสแม่ (Is-A Relationship) ได้อย่างสมเหตุสมผล ประยุกต์ใช้ฟังก์ชัน super().__init__() ส่งผ่านท่อข้อมูลได้อย่างถูกหลักวิศวกรรมซอฟต์แวร์ และทำ Method Overriding ได้อย่างกลมกลืนโดยไม่มีส่วนใดซ้ำซ้อน
- **ระดับดี (3 คะแนน)**: มีการใช้การสืบทอดคุณลักษณะเพื่อลดความซ้ำซ้อนของโค้ด แต่การใช้คำสั่ง super() หรือการเขียนทับเมธอดในคลาสลูกยังไม่สมบูรณ์ ทำให้บางแอตทริบิวต์ของคลาสแม่ไม่ถูกสร้างขึ้นในหน่วยความจำ
- **ระดับพอใช้ (2 คะแนน)**: มีการสืบทอดคลาส แต่คลาสลูกใช้วิธีการคัดลอกรหัสคำสั่ง (Copy-Paste) จากคลาสแม่มาเขียนใหม่ทั้งหมดในบล็อกตัวเอง ทำให้สูญเสียประโยชน์ของการนำโค้ดกลับมาใช้ซ้ำ
- **ระดับปรับปรุง (1 คะแนน)**: เขียนโค้ดแยกขาดจากกันทุกคลาส เกิดความซ้ำซ้อนของข้อมูลชุดเดิมในหลาย ๆ พิมพ์เขียว และไม่ปรากฏกลไกการสืบทอดคลาสเลย

มิติที่ 4: ความสะอาดของโค้ดและความยืดหยุ่นในการขยายระบบ (Polymorphism & Abstraction)

- **ระดับดีมาก (4 คะแนน)**: สลัดคราบหอคอยเงื่อนไข if-elif-else ออกไปจนหมดสิ้น (ไม่มี Code Smell) สั่งงานวัตถุหลากชนิดผ่านอินเทอร์เฟซสากลบรรทัดเดียวด้วยพลังพหุสัณฐาน เลือกใช้ Abstract Class ล็อกสัญญากฎโค้ดร่วมกับเดคคอเรเตอร์ @abstractmethod ได้อย่างสมบูรณ์แบบ ระบบรองรับการขยายตัวในอนาคตแบบ Plug-and-Play
- **ระดับดี (3 คะแนน)**: มีการใช้งานพหุสัณฐานและคลาสนามธรรม แต่ในระบบหลักยังมีคำสั่งเงื่อนไข if แอบนั่งดักเช็กประเภทของอ็อบเจกต์อยู่บางจุด ทำให้โค้ดไม่สะอาดร้อยเปอร์เซ็นต์
- **ระดับพอใช้ (2 คะแนน)**: มีการเขียนคลาสนามธรรม แต่คลาสลูกหลีกเลี่ยงสัญญาโดยไม่ยอมเขียนทับเมธอดนามธรรม ส่งผลให้ระบบแจ้งข้อผิดพลาดเมื่อพยายามสร้างวัตถุมาทำงาน
- **ระดับปรับปรุง (1 คะแนน)**: โค้ดเต็มไปด้วยโครงสร้างเงื่อนไขที่ยาวเหยียดและรุงรัง อ่านเข้าใจยาก บำรุงรักษายาก และระบบจะพังทันทีเมื่อมีการเพิ่มประเภทข้อมูลหรือฟีเจอร์ใหม่เข้าสู่ระบบ

เกณฑ์การตัดเกรด (แบบอิงเกณฑ์ตามมาตรฐานระดับมหาวิทยาลัย)

การแปลงผลคะแนนรวมรวมสรุปขั้นสุดท้ายจาก Rubric Score ทั้งหมดออกมาเป็นเกรดประจำวิชา จะใช้เกณฑ์มาตรฐานดังนี้ครับ

- **80 คะแนนขึ้นไป**: ได้ระดับผลการเรียน A (ความเป็นเลิศทางสถาปัตยกรรม OOP)
- **75 - 79 คะแนน:** ได้ระดับผลการเรียน B+
- **70 - 74 คะแนน:** ได้ระดับผลการเรียน B
- **65 - 69 คะแนน:** ได้ระดับผลการเรียน C+

- **60 - 64 คะแนน:** ได้ระดับผลการเรียน C
- **55 - 59 คะแนน:** ได้ระดับผลการเรียน D+
- **50 - 54 คะแนน:** ได้ระดับผลการเรียน D
- **ต่ำกว่า 50 คะแนน:** ได้ระดับผลการเรียน F (ไม่ผ่านเกณฑ์การประเมินขั้นต่ำ)

บทที่ 1
แนะนำ Google Colab และทบทวนพื้นฐานภาษา Python
(Introduction to Colab & Python Review)

แผนบริหารการสอนประจำบทที่ 1

จำนวนชั่วโมงที่สอน 4 ชั่วโมง | สัปดาห์ที่ 1

หัวข้อเนื้อหา

1. การใช้งาน Google Colab เบื้องต้น
2. ทบทวนการสร้างและการเรียกใช้งานฟังก์ชัน (Functions)
3. โครงสร้างข้อมูลแบบรายการ (Lists)
4. โครงสร้างข้อมูลแบบพจนานุกรม (Dictionaries)
5. กิจกรรมปฏิบัติการ สร้างสมุดโน้ต Colab แรก และเขียนโปรแกรมคำนวณภาษีแบบง่าย

คำสำคัญ

Google Colab, Jupyter Notebook, Function, def, List, Dictionary, Key-Value, Mutable, Immutable

วัตถุประสงค์เชิงพฤติกรรม

เมื่อผู้เรียนศึกษาบทเรียนนี้แล้วสามารถ

1. เข้าใช้งานและสร้างสมุดโน้ตบน Google Colab เพื่อเขียนและรันโค้ดภาษา Python ได้
2. เขียนโปรแกรมโดยใช้คำสั่ง def เพื่อสร้างฟังก์ชันสำหรับนำรหัสกลับมาใช้ซ้ำได้อย่างถูกต้อง
3. จัดการกลุ่มข้อมูลโดยประยุกต์ใช้โครงสร้างข้อมูลแบบ List และ Dictionary ได้อย่างเหมาะสม
4. ออกแบบและเขียนโปรแกรมคำนวณภาษีอย่างง่ายตามเงื่อนไขที่กำหนดได้

หน่วยการเรียนที่ 1.1 การใช้งาน Google Colab เบื้องต้น

ในการเริ่มต้นพัฒนาซอฟต์แวร์ อุปสรรคด่านแรกของนิสิตมักจะเป็นขั้นตอนการติดตั้งโปรแกรม (IDE) และการตั้งค่าสภาพแวดล้อมระบบปฏิบัติการที่ซับซ้อน รายวิชา SIT304 จึงเลือกใช้ Google Colaboratory (หรือ Colab) ซึ่งเป็นเครื่องมือพัฒนาโปรแกรมออนไลน์บนระบบคลาวด์ ข้อดีคือผู้เรียนสามารถเขียนโค้ดภาษา Python ผ่านเว็บเบราว์เซอร์ได้ทันทีโดยไม่ต้องตั้งค่าใด ๆ บนเครื่องคอมพิวเตอร์ส่วนตัว (Zero Setup Online IDE)

Colab ทำงานอยู่บนพื้นฐานของเทคโนโลยี Jupyter Notebook โดยจัดเก็บเอกสารในรูปแบบของ "สมุดโน้ต" (Notebook) ที่สามารถรวบรวมข้อความอธิบายทฤษฎี (Markdown) บล็อกรหัสคำสั่ง (Code) และผลลัพธ์