

# Gifted School 2024

Foundation of Programming

## สรุปเนื้อหา Week 01 - 03

### Week 01

#### บทนำสู่การเขียนโปรแกรมภาษาไพธอน (บทที่ 1)

- วัตถุประสงค์ของบทเรียน:
  - อธิบายองค์ประกอบพื้นฐานของคอมพิวเตอร์
  - อธิบายหลักการเขียนโปรแกรมเบื้องต้น
  - อธิบายประวัติของภาษาไพธอน
- คอมพิวเตอร์:
  - เครื่องคอมพิวเตอร์คืออุปกรณ์ที่จัดการกับข้อมูล เช่น การจัดเก็บข้อมูล การค้นหาข้อมูล และการประมวลผลข้อมูล
- โครงสร้างของเครื่องคอมพิวเตอร์:
  - อุปกรณ์นำเข้าข้อมูล (Input unit)
  - อุปกรณ์แสดงผลข้อมูล (Output unit)
  - หน่วยความจำ (Memory)
  - หน่วยประมวลผลกลาง (CPU)
  - หน่วยเก็บข้อมูลสำรอง (Secondary storage unit)
- ซอฟต์แวร์:
  - ชุดคำสั่งที่ใช้ในการสั่งการทำงานของคอมพิวเตอร์ มี 2 ประเภท: ซอฟต์แวร์สำหรับงานทั่วไป (Application software) และซอฟต์แวร์ระบบ (System software)
- การเขียนชุดคำสั่ง:

- การออกแบบและพัฒนาขั้นตอนการทำงานที่คอมพิวเตอร์เข้าใจและดำเนินการเพื่อให้ได้ผลลัพธ์ที่ต้องการ
- โครงสร้างของโปรแกรม:
  - โครงสร้างเชิงเส้น (Sequential structure): ดำเนินการตามลำดับ
  - โครงสร้างทางเลือก (Selection structure): เลือกเส้นทางตามเงื่อนไข
  - โครงสร้างการทำซ้ำ (Repetitive structure): ทำซ้ำชุดคำสั่งตามเงื่อนไข

## ภาษาไพธอน (Python) (บทที่ 2)

- ประวัติของภาษาไพธอน:
  - พัฒนาขึ้นในปี 1989 โดย Guido van Rossum ที่ CWI
  - ผสมผสานข้อดีของภาษาโปรแกรมหลายภาษา เช่น ABC และ Modula-3
- คุณสมบัติของภาษาไพธอน:
  - ภาษาโปรแกรมระดับสูง
  - ซอร์สโค้ดแบบเปิด (Open source)
  - ใช้งานง่าย (User friendly)
  - ไม่ขึ้นกับระบบปฏิบัติการ (Platform independent)

## การทำงานของภาษาไพธอน

- การย้ายโปรแกรมจากหน่วยความจำสำรองมายังหน่วยความจำ
- หน่วยประมวลผลกลางทำการประมวลผลโปรแกรมและข้อมูลบนหน่วยความจำจนได้ผลลัพธ์

## ข้อมูลและตัวแปร

- ข้อมูล (Data):
  - ข้อมูลที่เกิดจากการวัด สังเกต หรือประมวลผลทางสถิติ
- ตัวแปร (Variable):
  - พื้นที่ในหน่วยความจำที่มีการกำหนดชื่อเพื่อให้สามารถอ้างอิงและเปลี่ยนแปลงข้อมูลได้

## การเขียนโปรแกรมเชิงเส้น (Sequential Programming)

- การออกแบบโปรแกรม: IPO Pattern:

- แบ่งกระบวนการออกเป็น 3 ชุด: Input, Process, Output
- **วิธีการเขียนโปรแกรม:**
  - แยกปัญหาออกเป็นปัญหาย่อยๆ และแต่ละกิจกรรมย่อยจะถูกเรียงเรียงให้สอดคล้องกับแนวทางการแก้ไขปัญหา
- **การออกแบบด้วยผังไหล (Flowchart):**
  - แสดงการไหลของการทำงานของโปรแกรมโดยใช้สัญลักษณ์ต่างๆ เช่น ทรงกลม ขนาน และสี่เหลี่ยม

## สรุปกรณีศึกษา

- กรณีศึกษาการคำนวณค่าแรงจากเวลาทำงานและค่าแรงต่อชั่วโมง โดยการรับข้อมูลเข้า (Input) กระบวนการทำงาน (Process) และการแสดงข้อมูล (Output)

## Week 02

### การดำเนินการในภาษาไพธอน (บทที่ 3)

- **วัตถุประสงค์ของบทเรียน:**
  - อธิบายหลักการลำดับขึ้นการประมวลผลของการดำเนินการ
  - ใช้การดำเนินการในการเขียนโปรแกรมได้
- **ลำดับขึ้นการประมวลผล (Precedence):**
  - ใช้เครื่องหมาย ( ) เพื่อกำหนดลำดับความสำคัญสูงสุด
  - ใช้เครื่องหมาย \*, /, modulus ซ้ายไปขวา
  - ใช้เครื่องหมาย +, - ซ้ายไปขวา
- **การหาเศษ (Modulus):**
  - เป็นการดำเนินการหาเศษของจำนวนโดยใช้เครื่องหมาย %
- **การดำเนินการเชิงตรรกะ (Boolean Operators):**
  - **NOT:** พลิกค่าความจริง
  - **OR:** ถ้าค่าหนึ่งเป็นจริง ผลลัพธ์จะเป็นจริง

- **AND:** ถ้าทั้งสองค่าเป็นจริง ผลลัพธ์จะเป็นจริง
- **การดำเนินการเชิงความสัมพันธ์ (Relational Operators):**
  - ตรวจสอบความสัมพันธ์ระหว่างค่า 2 ค่า เช่น น้อยกว่า (<), มากกว่า (>), เท่ากับ (==), ไม่เท่ากับ (!=)

## การตัดสินใจในโปรแกรม (บทที่ 5)

- **วัตถุประสงค์ของบทเรียน:**
  - อธิบายหลักการเขียนโปรแกรมเพื่อการตัดสินใจ
  - ใช้โครงสร้างตัดสินใจในการออกแบบโปรแกรม
  - พัฒนาโปรแกรมที่มีโครงสร้างตัดสินใจได้
- **ทำไมต้องใช้โครงสร้างการตัดสินใจ:**
  - เพื่อให้โปรแกรมสามารถเลือกทางเลือกได้ตามเงื่อนไข
  - ป้องกันการคำนวณที่ผิดพลาดจากข้อมูลที่ไม่สอดคล้องกัน
- **วิธีการออกแบบโครงสร้างการตัดสินใจ:**
  - กำหนดทางเลือกที่เป็นไปได้ทั้งหมด
  - กำหนดเงื่อนไขในการเลือกแต่ละทางเลือก
  - เขียนเงื่อนไขให้อยู่ในประโยคเดียว
- **การเขียนโครงสร้างการตัดสินใจโดยใช้ผังไหล (Flowchart):**
  - ใช้สัญลักษณ์ Diamond เพื่อแสดงเงื่อนไขและเส้นทางที่ออกจาก Diamond จะมีค่าความจริงกำกับ

## กรณีศึกษา

1. **การคำนวณราคาสินค้ารวมภาษีมูลค่าเพิ่ม:**
  - ข้อมูลนำเข้า: ราคาสินค้า
  - กระบวนการ: คำนวณราคาสินค้ารวมภาษี
  - ผลลัพธ์: แสดงราคาสินค้ารวมภาษี
2. **การตัดสินใจในการเลือกทานอาหาร:**

- เงื่อนไข: ถ้ามีเงินมากกว่า 1000 บาท จะไปร้าน ก ถ้า มีเงินนั้นจะไปร้าน ข ไม่

### 3. การตัดสินใจในการเข้าร่วมสอบปลายภาค:

- เงื่อนไข: นักศึกษาที่ขาดเรียนมากกว่า 20% จะไม่สามารถเข้าร่วมสอบได้

## โครงสร้างทั่วไปของเงื่อนไข

- ใช้ตัวดำเนินการตรรกศาสตร์ (Boolean Operators) เช่น AND, OR, NOT
- ใช้ตัวดำเนินการเชิงคณิตศาสตร์ (Arithmetic Operators) เช่น +, -, \*, /, %
- ใช้ตัวดำเนินการเชิงความสัมพันธ์ (Relational Operators) เช่น <, >, ==, !=

เนื้อหานี้เป็นพื้นฐานการเขียนโปรแกรมภาษาไพธอนที่ครอบคลุมการดำเนินการต่างๆ และการตัดสินใจเพื่อออกแบบโปรแกรมที่สามารถทำงานได้อย่างมีประสิทธิภาพตามเงื่อนไขที่กำหนด

## Week 03

### การทำซ้ำ (Loop) (บทที่ 6)

- วัตถุประสงค์ของบทเรียน:
  - อธิบายหลักการเขียนโปรแกรมแบบวนซ้ำ
  - ใช้โครงสร้างการทำซ้ำในการออกแบบโปรแกรม
  - พัฒนาโปรแกรมที่สามารถทำซ้ำได้
- เหตุผลในการใช้โครงสร้างการทำซ้ำ:
  - โครงสร้างเชิงเส้น (Sequential Structure) ทำงานเป็นเส้นตรง ทุกรอบของการประมวลผล
  - โครงสร้างเชิงเลือก (Selection Structure) เลือกทางเลือกตามเงื่อนไข
  - การผสมผสานทั้งสองแบบเพื่อออกแบบการแก้ไขปัญห
- โครงสร้างการทำซ้ำ (Loop Structure):
  - องค์ประกอบ 2 ส่วน:
    1. **เงื่อนไขหยุด (Stop Conditions):** นิพจน์ตรรกะที่ทำให้โครงสร้างการทำซ้ำหยุดการทำงาน

2. **เนื้อหาการทำงาน (Statements หรือ Loop Bodies):** ขั้นตอนการทำงานที่ทำซ้ำเมื่อเงื่อนไขการหยุดไม่เป็นจริง

- ประเภทของโครงสร้างการทำซ้ำ:
  - **Pre-test Loop:** ตรวจสอบเงื่อนไขก่อนทำงาน
  - **Post-test Loop:** ทำงานก่อนถึงตรวจสอบเงื่อนไข

### กรณีศึกษา #1: การแสดงข้อความ "Hello World"

- ผู้ใช้กรอกเลข 3 โปรแกรมจะแสดง "Hello World" สามครั้ง
- ผู้ใช้กรอกเลข 5 โปรแกรมจะแสดง "Hello World" ห้าครั้ง

### กรณีศึกษา #2: การคำนวณแฟกทอเรียล (Factorial)

- **ข้อมูลนำเข้า (Input):** จำนวนเต็มบวก 1 ค่า
- **ผลลัพธ์ (Output):** ค่าแฟกทอเรียลของข้อมูลนำเข้า
- **กระบวนการ (Process):** คำนวณแฟกทอเรียล เช่น  $3! = 3 \times 2 \times 1 = 6$

### การออกแบบโครงสร้างการทำงานซ้ำ

- การระบุเงื่อนไขการหยุด:
  - หยุดตามจำนวนรอบ
  - หยุดตามค่า
- การระบุเนื้อหาการทำงาน:
  - ขั้นตอนที่เกิดซ้ำ เช่น การคูณ

### การใช้ตัวแปรสะสมค่า (Cumulative Variable)

- การส่งต่อข้อมูลจากรอบก่อนหน้าไปยังรอบถัดไป
- ตัวแปรสะสมค่าต้องมีการกำหนดค่าเริ่มต้นที่สอดคล้องกับการดำเนินงาน เช่น การคูณกำหนดค่าเริ่มต้นเป็น 1

### โครงสร้างเชิงซ้อน (Nested Structure)

- การใช้โครงสร้างการเขียนโปรแกรมซ้อนในโครงสร้างการเขียนโปรแกรมอีกโครงสร้างหนึ่ง
- เช่น ใช้ for loop ซ้อนใน while loop หรือ if statement ซ้อนใน for loop

## ตัวอย่างการใช้โครงสร้างการทำซ้ำ

- **While Loop:** ใช้สำหรับการทำซ้ำจนกว่าเงื่อนไขจะเป็นเท็จ
- **For Loop:** ใช้สำหรับการทำซ้ำที่มีจำนวนรอบที่แน่นอน โดยใช้คำสั่ง `range()`

## ปัญหาทั่วไปของการใช้โครงสร้างการทำซ้ำ

- **Off-by-one Error (OBOE):** จำนวนรอบการทำงานเกินหรือขาดไป 1 รอบ
- **Infinite Loop:** เงื่อนไขการหยุดไม่มีทางเป็นเท็จ ส่งผลให้โปรแกรมทำงานไม่หยุด

## การติดตามการเปลี่ยนแปลงค่า (Program Tracing)

- วิธีการติดตามการเปลี่ยนแปลงของตัวแปรในการทำงานของโปรแกรม เพื่อตรวจสอบความถูกต้องและหา error