หน่วยที่ 1 หลักการเขียนโปรแกรมขั้นต้น

คอมพิวเตอร์ เป็นอุปกรณ์ทางอิเล็กทรอนิกส์อย่างหนึ่ง ซึ่งไม่สามารถทำงานด้วยตนเองได้ แต่จะสามารถทำงานได้ตามชุดคำสั่งในโปรแกรมที่ป้อนเข้าสู่เครื่อง ซึ่งจะทำงานตามคำสั่งทีละคำสั่ง (Step by Step) โดยคำสั่งที่เครื่องคอมพิวเตอร์สามารถเข้าใจได้ จะต้องอยู่ในรูปแบบของภาษาเครื่อง (Machine Language) แต่ถ้ามีการเขียนด้วยภาษาอื่นที่ไม่ใช่ภาษาเครื่อง หรือที่เรียกว่า ภาษาชั้นสูง (High-level Language) ก็จะต้องมีตัวแปลภาษา เช่น คอมไพเลอร์ (Compiler) หรือ อินเตอร์พรีเตอร์ (Interpreter) ทำการแปลภาษาชั้นสูงนั้นให้เป็นภาษาเครื่องอีกทีหนึ่งในการเขียนโปรแกรมหรือภาษาคอมพิวเตอร์นี้ โดยทั่วไปแล้วแต่ละภาษาจะมีหลักเกณฑ์ในการเขียนและการออกแบบโปรแกรมเหมือนกัน ซึ่งสามารถที่จะแบ่งขั้นตอนการเขียนโปรแกรมออกได้เป็น 7 ขั้นตอน ดังนี้

1. ขั้นตอนการวิเคราะห์ปัญหา (Analysis the Problem)
2. ขั้นตอนการออกแบบโปรแกรม (Design a Program)
3. ขั้นตอนการเขียนโปรแกรม (Coding)
4. ขั้นตอนการตรวจสอบข้อผิดพลาดของโปรแกรม (Testing and Debugging)
5. ขั้นตอนการทดสอบความถูกต้องของโปรแกรม (Testing and Validating)
6. ขั้นตอนการทำเอกสารประกอบโปรแกรม (Documentation)
7. ขั้นตอนการบำรุงรักษาโปรแกรม (Program Maintenance)

ขั้นตอนที่ 1 ขั้นตอนการวิเคราะห์ปัญหา (Analysis the problem)

ขั้นตอนนี้เป็นขั้นตอนแรกสุดที่นักเขียนโปรแกรมจะต้องทำก่อนที่จะลงมือเขียนโปรแกรมจริง ๆ เพื่อทำความเข้าใจกับปัญหาที่เกิดขึ้น และค้นหาจุดมุ่งหมายหรือสิ่งที่ต้องการ ในขั้นตอนนี้จะมีองค์ประกอบอยู่ 3 องค์ประกอบที่จะช่วยในการวิเคราะห์ปัญหา ได้แก่

1. การระบุข้อมูลเข้า (Input) ต้องรู้ว่ามีข้อมูลอะไรบ้างที่จะต้องป้อนเข้าสู่คอมพิวเตอร์พร้อมกับโปรแกรม เพื่อให้โปรแกรมทำการประมวลผลและออกผลลัพธ์
2. การระบุข้อมูลออก (Output) จะพิจารณาว่างานที่ทำมีเป้าหมายหรือวัตถุประสงค์อะไร ต้องการผลลัพธ์ที่มีรูปร่างหน้าตาเป็นอย่างไร โดยจะต้องคำนึงถึงผู้ใช้เป็นหลักในการออกแบบผลลัพธ์
3. กำหนดวิธีการประมวลผล (Process) ต้องรู้วิธีการประมวลผลเพื่อให้ได้ผลลัพธ์ตามที่ต้องการ

2. ขั้นตอนการออกแบบโปรแกรม (Design a Program)

หลังจากวิเคราะห์ปัญหาแล้ว ขั้นตอนถัดไปคือ การออกแบบโปรแกรม โดยใช้เครื่องมือมาช่วยในการออกแบบ ในขั้นตอนนี้ยังไม่ได้เป็นการเขียนโปรแกรมจริง ๆ แต่จะช่วยให้การเขียนโปรแกรมทำได้ง่ายขึ้น โดยสามารถเขียนตามขั้นตอนที่ได้ออกแบบไว้ในขั้นตอนนี้ และช่วยให้การเขียนโปรแกรมมีข้อผิดพลาดน้อยลง ช่วยตรวจสอบการทำงานของโปรแกรม ทำให้ทราบขั้นตอนการ

ทำงานของโปรแกรมได้อย่างรวดเร็ว โดยไม่ต้องไปไล่ดูจากตัวโปรแกรมจริง ๆ ซึ่งถ้าเปรียบเทียบการเขียนโปรแกรมเหมือนกับการสร้างบ้านแล้ว ในขั้นตอนการออกแบบโปรแกรมนี้ ก็เปรียบเหมือนการสร้างแปลนบ้านลงในกระดาษไว้ ซึ่งในการสร้างบ้านจริง ก็จะอาศัยแปลนบ้านนี้เป็นต้นแบบในการสร้างนั่นเอง

ในขั้นตอนการออกแบบโปรแกรมนี้ เป็นการออกแบบการทำงานของโปรแกรม หรือขั้นตอนในการแก้ปัญหา ซึ่งผู้ออกแบบสามารถเลือกใช้เครื่องมือมาช่วยในการออกแบบได้ โดยเครื่องมือที่ใช้ในการออกแบบโปรแกรมมีอยู่หลายอย่าง ซึ่งวิธีการซึ่งเป็นที่นิยมสำหรับใช้ในการออกแบบโปรแกรม เช่น

- อัลกอริทึม (Algorithm)
- ผังงาน (Flowchart)
- รหัสจำลอง (Pseudo-code)
- แผนภูมิโครงสร้าง (Structure Chart)

อัลกอริทึมเป็นเครื่องมือที่ช่วยในการออกแบบโปรแกรม โดยใช้ข้อความที่เป็นภาษาพูดในการอธิบายการทำงานของโปรแกรมที่เป็นลำดับขั้นตอน จะข้ามไปข้ามมาไม่ได้ นอกจากจะต้องเขียนสั่งไว้ต่างหาก ตัวอย่างอัลกอริทึมง่าย ๆ ที่พบเห็นในชีวิตประจำวัน ได้แก่ อัลกอริทึมการสระผม

เริ่มจากการทำผมให้เปียกโดยการราดน้ำ เมื่อผมเปียกแล้วจึงใส่แชมพูสระผมลงบนศีรษะ แล้วขยี้ให้มีฟองเกิดขึ้น หลังจากนั้นก็ล้างออกด้วยน้ำ แล้วเริ่มทำใหม่อีกครั้ง

ในการเขียนอัลกอริทึมนี้ แม้จะมีความชัดเจนอยู่ในตัวแล้ว แต่ก็มีจุดอ่อนอยู่ที่ ข้อความอธิบายค่อนข้างเยิ่นเย้อ และถ้าผู้เขียนใช้สำนวนที่อ่านยาก ก็อาจทำให้ผู้อ่านไม่เข้าใจขั้นตอนการทำงานของโปรแกรมได้ ดังนั้น จึงมีการคิดค้นเครื่องมืออื่นที่ช่วยในการออกแบบโปรแกรมแทนอัลกอริทึม ได้แก่ ผังงาน รหัสจำลอง แผนภูมิโครงสร้าง

ผังงานเป็นเครื่องมือที่ช่วยในการออกแบบโปรแกรม โดยใช้สัญลักษณ์รูปภาพ แสดง ขั้นตอนการเขียนโปรแกรม หรือขั้นตอนในการแก้ปัญหาทีละขั้น และมีเส้นที่แสดงทิศทางการไหลของข้อมูลตั้งแต่จุดเริ่มต้นจนกระทั่งได้ผลลัพธ์ตามที่ต้องการ ซึ่งจะทำให้ผู้อ่านสามารถอ่านและทำความเข้าใจได้โดยง่าย

รหัสจำลองจะมีการใช้ข้อความที่เป็นภาษาอังกฤษหรือภาษาไทยก็ได้ ในการแสดงขั้นตอนการแก้ปัญหา แต่จะมีการใช้คำเฉพาะ (Reserve words) ที่มีอยู่ในภาษาโปรแกรม มาช่วยในการเขียน โครงสร้างของรหัสจำลองจึงมีส่วนที่คล้ายกับการเขียนโปรแกรมมาก ดังนั้น รหัสจำลองจึงเป็นเครื่องมืออีกแบบที่เป็นที่นิยมใช้กันมากในการออกแบบโปรแกรม แผนภูมิโครงสร้างการใช้แผนภูมิโครงสร้าง จะเป็นการแบ่งงานใหญ่ออกเป็นโมดูลย่อย ๆ ซึ่งเรียกว่า การออกแบบจากบนลงล่าง (Top-Down Design) แต่ละโมดูลย่อยก็ยังสามารถแตกออกได้อีกจนถึงระดับล่างสุดที่สามารถเขียนโปรแกรมได้อย่างง่าย

3. ขั้นตอนการเขียนโปรแกรม (Coding)

ในขั้นตอนนี้ จะเป็นการนำเครื่องมือที่ถูกสร้างขึ้นจากขั้นตอนการออกแบบมาแปลให้เป็นโปรแกรมคอมพิวเตอร์ ซึ่งในการสร้างโปรแกรมคอมพิวเตอร์นั้น เราสามารถเลือกใช้ภาษาได้หลายภาษา ตั้งแต่ภาษาระดับต่ำ เช่น ภาษาแอสเซมบลี จนถึงภาษาระดับสูง เช่น ภาษาเบสิก ภาษาโคบอล ภาษาปาสคาล ภาษาซี ซึ่งแต่ละภาษาจะมีรูปแบบ โครงสร้าง หรือไวยากรณ์ของภาษาที่แตกต่างกันออกไปดังนั้น การเขียนโปรแกรมที่ดีนั้น ควรจะต้องทำตามขั้นตอนคือ เริ่มตั้งแต่วิเคราะห์ปัญหาให้ได้ก่อน แล้วทำการออกแบบโปรแกรมจึงจะเริ่มเขียนโปรแกรม ซึ่งในการเขียนโปรแกรมนั้น สำหรับผู้ที่ยังไม่มีประสบการณ์การเขียนโปรแกรมเพียงพอ ก็ควรจะทดลองเขียนลงในกระดาษก่อน แล้วตรวจสอบจนแน่ใจว่าสามารถทำงานได้แล้ว จึงทำการป้อนเข้าสู่เครื่องคอมพิวเตอร์ เพื่อเป็นการประหยัดเวลาและทำให้สามารถทำงานได้เร็วขึ้น

4. ขั้นตอนการตรวจสอบข้อผิดพลาดของโปรแกรม (Testing and Debugging)

หลังจากที่ทำการเขียนโปรแกรมเสร็จสิ้นแล้ว โปรแกรมนั้นจะต้องได้รับการตรวจสอบก่อนว่า มีข้อผิดพลาด (error) ในโปรแกรมหรือไม่ ซึ่งอาจเกิดจากการเขียนโปรแกรมที่ผิดหลักไวยากรณ์ของภาษาเป็นต้น โดยทั่วไปจะมีวิธีที่จะตรวจสอบข้อผิดพลาดของโปรแกรม 2 ขั้นตอน ดังนี้

1. ตรวจสอบด้วยตนเอง (Self Checking) เป็นการทดลองเขียนโปรแกรมลงบนกระดาษ แล้วใส่ตรวจสอบการทำงานของโปรแกรมทีละขั้นด้วยตนเอง ว่าโปรแกรมมีการทำงานที่ถูกต้อง ได้ผลลัพธ์ตรงตามความเป็นจริงหรือไม่

2. ตรวจสอบด้วยการแปลภาษา (Translating) หลังจากที่เขียนโปรแกรมเสร็ข และมีการตรวจสอบด้วยตนเองเรียบร้อยแล้ว ก็จะป้อนโปรแกรมเข้าสู่เครื่องคอมพิวเตอร์เพื่อทำการแปลโปรแกรม โดยจะต้องเรียกใช้ตัวแปลภาษาโปรแกรม ที่เรียกว่า คอมไพเลอร์ (Compiler) หรืออินเตอร์พรีเตอร์ (Interpreter) อย่างใดอย่างหนึ่ง ทำการแปลภาษาโปรแกรมให้เป็นภาษาเครื่อง การแปลนี้จะเป็นการตรวจสอบความผิดพลาดของโปรแกรมด้วย ซึ่งถ้ามีข้อผิดพลาดใด ๆ เครื่องคอมพิวเตอร์จะแจ้งให้ทราบทางหน้าจอ

หลังจากที่ทำการเขียนโปรแกรมเสร็จแล้ว เวลา 50-70% ของเวลาในการพัฒนาโปรแกรม จะถูกใช้ไปในการหาข้อผิดพลาดของโปรแกรมและการ แก้ไขข้อผิดพลาดนั้น

5. ขั้นตอนการทดสอบความถูกต้องของโปรแกรม (Testing and Validating)

ในบางครั้ง โปรแกรมอาจผ่านการแปล โดยไม่มีข้อผิดพลาดใด ๆ แจ้งออกมา แต่เมื่อนำโปรแกรมนั้นไปใช้งาน ปรากฏว่าได้ผลลัพธ์ที่ไม่เป็นจริง เนื่องจากอาจเกิดข้อผิดพลาดขึ้นได้ ดังนั้นจึงควรจะต้องมีขั้นตอนการทดสอบความถูกต้องของโปรแกรมอีกทีด้วยในการทดสอบความถูกต้องของข้อมูล จะมีอยู่หลายวิธี ดังต่อไปนี้

1. การใส่ข้อมูลที่ถูกต้อง (Valid Case) เป็นการทดสอบโปรแกรมเมื่อมีการรันโปรแกรม ให้ทำการใส่ข้อมูลที่ถูกต้องลงไปในโปรแกรม และดูว่าผลลัพธ์ที่ได้จากโปรแกรม ถูกต้องตามความเป็นจริงหรือตรงตามที่ต้องการหรือไม่

2. การใช้ขอบเขตและความถูกต้องของข้อมูลเป็นการทดสอบ โดยตรวจสอบขอบเขตของข้อมูลที่ป้อนเข้าสู่โปรแกรม เช่น ถ้าโปรแกรมให้มีการป้อนวันที่ ก็จะต้องตรวจสอบว่า วันที่ที่ป้อนจะต้องไม่เกินวันที่ 31 ถ้าผู้ใช้ป้อนวันที่ที่เป็นเลข 32 โปรแกรมจะต้องไม่ยอมให้ป้อนวันที่นี้ได้

3. การใช้ความสมเหตุสมผล ตัวอย่างเช่น ถ้าโปรแกรมมีการออกแบบให้ผู้ใช้ป้อนข้อมูลลงไปในฟอร์มที่มีข้อมูลที่เป็นเพศ (หญิง หรือ ชาย) และรายละเอียดส่วนตัวของคน ๆ นั้น เช่นเพศ วันลาคลอดชาย ต้องไม่มี (ห้ามใส่)หญิง อาจมีหรือไม่มีก็ได้

4. ข้อมูลที่เป็นตัวเลขและตัวอักษร เป็นการตรวจสอบว่า ถ้าโปรแกรมให้ผู้ใช้ป้อนข้อมูลในฟิลด์ที่ต้องรับข้อมูลที่เป็นตัวเลข อย่างเช่น ฟิลด์ที่เป็นจำนวนเงิน ก็ควรจะยอมให้ผู้ใช้ป้อนข้อมูลได้เฉพาะตัวเลขเท่านั้น ไม่อนุญาตให้ใส่ตัวอักษรในฟิลด์นั้นได้ หรือถ้าเป็นฟิลด์ที่รับข้อมูลที่เป็นตัวอักษร เช่น ฟิลด์ชื่อ-นามสกุล ก็จะป้อนข้อมูลได้เฉพาะตัวอักษรเท่านั้น จะป้อนตัวเลขไม่ได้

5. ข้อมูลเป็นไปตามข้อกำหนด ข้อมูลที่ป้อนในฟิลด์ ต้องเป็นไปตามที่กำหนดไว้แน่นอนแล้วเท่านั้น เช่น กำหนดให้ฟิลด์นี้ป้อนข้อมูลได้เฉพาะตัวเลขที่อยู่ในกลุ่ม 1,2,5,7 ได้เท่านั้น จะป้อนเป็นตัวเลขอื่นที่ไม่อยู่ในกลุ่มนี้ ไม่ได้

6. ขั้นตอนการทำเอกสารประกอบโปรแกรม (Documentation)

การทำเอกสารประกอบโปรแกรม คือ การอธิบายรายละเอียดของโปรแกรมว่า จุดประสงค์ของโปรแกรมคืออะไร สามารถทำงานอะไรได้บ้าง และมีขั้นตอนการทำงานของโปรแกรมเป็นอย่างไร เครื่องมือที่ช่วยในการออกแบบโปรแกรมเช่น ผังงาน หรือรหัสจำลอง ก็สามารถนำมาประกอบกันเป็นเอกสารประกอบโปรแกรมได้โปรแกรมเมอร์ที่ดี ควรมีการทำเอกสารประกอบโปรแกรม ทุกขั้นตอนของการพัฒนาโปรแกรม ไม่ว่าจะเป็นขั้นตอนการออกแบบ การเขียนโปรแกรม หรือขั้นตอนการทดสอบโปรแกรม ซึ่งการทำเอกสารนี้จะมีประโยชน์อย่างมากต่อหน่วยงาน เนื่องจากบางครั้งอาจต้องการเปลี่ยนแปลงแก้ไขโปรแกรมที่ได้มีการทำเสร็จไปนานแล้ว เพื่อให้ตรงกับความต้องการที่เปลี่ยนไป จะทำให้เข้าใจโปรแกรมได้ง่ายขึ้นและจะเป็นการสะดวกต่อผู้ที่ต้องเข้ามารับช่วงงานต่อทีหลังเอกสารประกอบโปรแกรม โดยทั่วไปจะมีอยู่ด้วยกัน 2 แบบคือ

1. เอกสารประกอบโปรแกรมสำหรับผู้ใช้ (User Documentation)

จะเหมาะสำหรับผู้ใช้ที่ไม่ต้องเกี่ยวข้องกับการพัฒนาโปรแกรม แต่เป็นผู้ที่ใช้งานโปรแกรมอย่างเดียว จะเน้นการอธิบายเกี่ยวกับการใช้งานโปรแกรมเป็นหลัก ตัวอย่างเช่น

· โปรแกรมนี้ทำอะไร ใช้งานในด้านไหน
· ข้อมูลเข้า มีลักษณะอย่างไร
· ข้อมูลออกหรือผลลัพธ์มีลักษณะอย่างไร
· การเรียกใช้โปรแกรม ทำอย่างไร

· คำสั่งหรือข้อมูลที่จำเป็นให้โปรแกรมเริ่มทำงาน มีอะไรบ้าง
· อธิบายเกี่ยวกับประสิทธิภาพ และความสามารถของโปรแกรม

2. เอกสารประกอบโปรแกรมสำหรับผู้เขียนโปรแกรม (Technical Documentation) จะได้ออกได้เป็น 2 ส่วน

· ส่วนที่เป็นคำอธิบายหรือหมายเหตุในโปรแกรม หรือเรียกอีกอย่างหนึ่งว่า คอมเมนท์ (Comment) ซึ่งส่วนใหญ่มักจะเขียนแทรกอยู่ในโปรแกรม อธิบายการทำงานของโปรแกรมเป็นส่วน ๆ
· ส่วนอธิบายด้านเทคนิค ซึ่งส่วนนี้มักจะทำเป็นเอกสารแยกต่างหากจากโปรแกรม จะอธิบายในรายละเอียดที่มากขึ้น เช่น ชื่อโปรแกรมย่อยต่าง ๆ มีอะไรบ้าง แต่ละโปรแกรมย่อยทำหน้าที่อะไร และคำอธิบายย่อ ๆ เกี่ยวกับวัตถุประสงค์ของโปรแกรม

7. ขั้นตอนการบำรุงรักษาโปรแกรม (Program Maintenance)

เมื่อโปรแกรมผ่านการตรวจสอบตามขั้นตอนเรียบร้อยแล้ว และถูกนำมาให้ผู้ใช้ได้ใช้งาน ในช่วงแรกผู้ใช้อาจจะยังไม่คุ้นเคยก็อาจทำให้เกิดปัญหาขึ้นมาบ้าง ดังนั้นจึงต้องมีผู้คอยควบคุมดูแล และตรวจสอบการทำงาน การบำรุงรักษาโปรแกรมจึงเป็นขั้นตอนที่ผู้เขียนโปรแกรมต้องคอยเฝ้าดูแล และหาข้อผิดพลาดของโปรแกรมในระหว่างที่ผู้ใช้ใช้งานโปรแกรมและปรับปรุงแก้ไขโปรแกรมเมื่อเกิดข้อผิดพลาดขึ้น หรือในการใช้งานโปรแกรมไปนาน ๆ ผู้ใช้อาจต้องการเปลี่ยนแปลงการทำงานของระบบเดิมเพื่อให้เหมาะกับเหตุการณ์ เช่น ต้องการเปลี่ยนแปลงหน้าตาของรายงาน มีการเพิ่มเติมข้อมูลหรือลบข้อมูลเดิม นักเขียนโปรแกรมก็จะต้องคอยปรับปรุง แก้ไขโปรแกรมตามความต้องการของผู้ใช้ที่เปลี่ยนแปลงไปนั้น

คุณสมบัติของนักเขียนโปรแกรมที่ดี

นักเขียนโปรแกรมหรือเรียกอีกอย่างหนึ่งว่า โปรแกรมเมอร์นั้น ควรจะมีคุณสมบัติดังต่อไปนี้ จึงจะเรียกได้ว่าเป็นโปรแกรมเมอร์ที่ดี

· รักและชอบในการเขียนโปรแกรม
· มีความคิดริเริ่มสร้างสรรค์ และไฝ่ที่จะเรียนรู้
· มีความอดทนต่อการเขียนโปรแกรม ซึ่งบางครั้งอาจต้องใช้เวลานานในการเขียนโปรแกรม
· ต้องรู้จักการทำงานเป็นทีมหรือเป็นกลุ่มคณะ ซึ่งการพัฒนาโปรแกรมที่ใหญ่ ๆ อาจต้องมีการทำงานกันเป็นทีม ต้องมีการแบ่งงานกันทำเป็นส่วน ๆ แล้วจึงจะนำมารวมกันทีหลัง ผลงานที่ออกมาจะต้องเป็นผลงานส่วนรวมของทั้งทีม ไม่ใช่ของคนใดคนหนึ่ง ดังนั้นจึงต้องรู้จักการถ่ายทอดความรู้ ความคิดเห็นให้แก่คนในทีมงานเดียวกัน
· ต้องหมั่นทำเอกสารประกอบโปรแกรมไว้ตลอด เพื่อให้ง่ายต่อการพัฒนาต่อไปในภายหลัง

ลักษณะของโปรแกรมที่ดี

โปรแกรมที่ดี จะต้องมีลักษณะ ดังนี้

· สามารถอ่านแล้วมีความเข้าใจง่าย ควรหลีกเลี่ยงการใช้คำสั่ง GOTO เพื่อสั่งให้โปรแกรมกระโดดไปทำงานที่จุดนั้น จุดนี้ เพราะจะทำให้ ผู้อ่านโปรแกรมเกิดความสับสนได้ง่าย

· ควรจะเปิดโอกาสให้สามารถเข้าไปทำการแก้ไข หรือขยายโปรแกรมได้โดยง่าย นั่นคือควรจะมีการแบ่งการทำงานของโปรแกรมนั้นทั้งหมดออกเป็นส่วนย่อย ๆ ที่เรียกว่า โมดูล (Module) โดยแต่ละโมดูลก็จะมีหน้าที่การทำงานที่อิสระจากกัน แต่จะมีการส่งผ่านข้อมูลให้กันและกันได้ระหว่างโมดูล ดังนั้นผู้ที่จะเข้าไปทำการแก้ไขโปรแกรม ก็สามารถเลือกได้ว่า โมดูลไหนที่เกี่ยวข้องกับตน ก็จะแก้ไขเฉพาะ โมดูลนั้น โดยไม่จำเป็นต้องศึกษารายละเอียดของโปรแกรมทั้งหมด

· มีคำอธิบายโปรแกรมหรือคอมเมนท์ สอดแทรกอยู่ในแต่ละโมดูล เพื่อให้ง่ายต่อการทำความเข้าใจในการทำงานของโมดูลนั้น

· ควรทำงานได้อย่างถูกต้อง รวดเร็ว และมีประสิทธิภาพ