Java Structure Programming

การเขียนโปรแกรมแบบโครงสร้างด้วยภาษาจาวา

จาวาฉบับพื้นฐาน ด้วยการเขียนโปรแกรมแบบโครงสร้าง

- ความรู้เบื้องต้นของภาษาจาวา
- คำสั่งแสดงผล
- คำสั่งรับข้อมูล
- ตัวดำเนินการ
- คำสั่งเงื่อนไข
- คำสั่งวนซ้ำ
- อาร์เรย์
- สตริง
- เมธอดและการเรียกใช้คลาส
- การดักจับข้อผิดพลาด
- แฟ้มข้อมูล
- ความรู้เบื้องต้นของการเขียนโปรแกรมเชิงวัตถุ

โดย เชาวลิต ขันคำ

คำนำ

ตำราเล่มนี้เขียนขึ้นเพื่อใช้สอนรายวิชา 4121204 การเขียนโปรแกรมภาษาคอมพิวเตอร์ (Computer Programming Language) ในหลักสูตรหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิทยาการคอมพิวเตอร์ เนื้อหาในตำราเล่มนี้นำเสนอหลักการเขียนโปรแกรมภาษาคอมพิวเตอร์ด้วยโปรแกรมภาษาจาวาที่เป็นที่นิยมในปัจจุบัน ลำดับการนำเสนอเนื้อหาเริ่มจากการติดตั้งโปรแกรมภาษาจาวา รูปแบบไวยากรณ์ของภาษา คำสั่งป้อนข้อมูล คำสั่งและรูปแบบการแสดงผลข้อมูล ชนิดของข้อมูลแบบต่าง ๆ ตัวดำเนินการ คำสั่งควบคุมแบบเงื่อนไขและการวนซ้ำ การใช้งานอาร์เรย์ เมธอดและการใช้งานคลาสเบื้องต้น สตริง การใช้งานแฟ้มข้อมูล การดักจับข้อผิดพลาด และหลักการเขียนโปรแกรมเชิงวัตถุเบื้องต้น

ผู้ที่ควรจะศึกษาจากตำราเล่มนี้เป็นอย่างยิ่งคือ นักเรียน นักศึกษา ที่เป็นผู้เริ่มศึกษาการเขียนโปรแกรมด้วยภาษาจาวา และ ครู อาจารย์ ที่ต้องการสอนการเขียนโปรแกรม เนื่องจากเนื้อหาในตำรา แสดงแนวคิดเชิงทฤษฎีให้กระชับพอเข้าใจ และเน้นตัวอย่างที่มีรหัสต้นฉบับและผลการรันโปรแกรมพร้อมกันในแต่ละตัวอย่างตามเนื้อหาที่นำเสนอไว้ ทั้งนี้ยังมีแบบฝึกหัดที่เปิดประเด็นสำหรับนำไปศึกษาและฝึกเขียนโปรแกรมเพื่อต่อยอดในระดับสูงขึ้นไปอีกด้วย

ท้ายนี้ผู้เขียนหวังเป็นอย่างยิ่งว่า ผู้ที่ได้ศึกษาจากตำราเล่มนี้จะมีทัศนคติที่ดีต่อการเขียนโปรแกรมจนพัฒนาตนเองให้เป็นโปรแกรมเมอร์ที่เก่ง สามารถคิดและวางแผนการเขียนโปรแกรมอย่างเป็นระบบ เพื่อพัฒนาประเทศไปสู่ความเป็นผู้นำทางด้านการพัฒนาซอฟต์แวร์ในอนาคต

เชาวลิต ขันคำ

9 กันยายน 2555

สารบัญ

หน้า

หน้า

หน้า

หน้า

หน้า

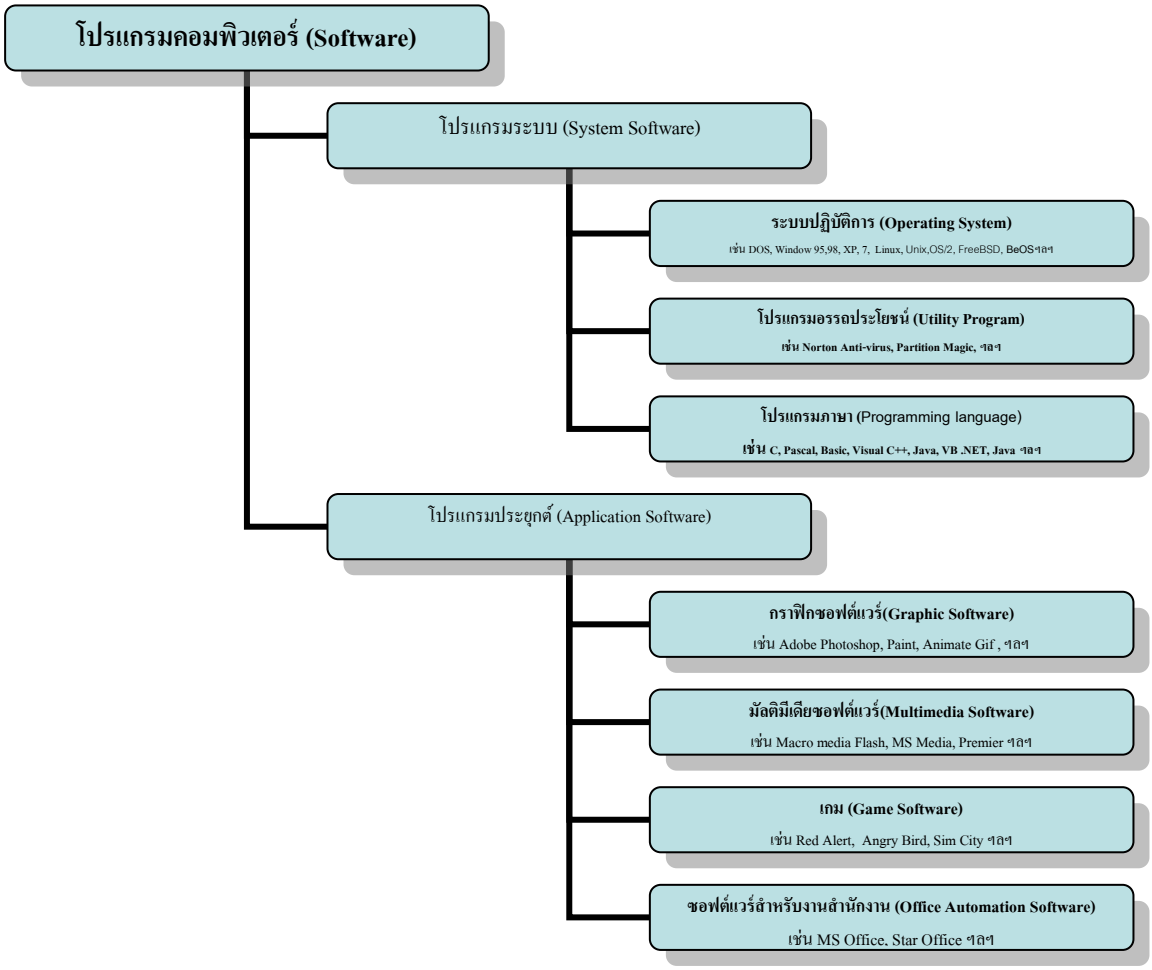
บทที่ 1
การเขียนโปรแกรมคอมพิวเตอร์

คอมพิวเตอร์ (Computer) คือ อุปกรณ์อิเล็กทรอนิกส์ที่สามารถประมวลผลข้อมูลให้ได้โดยอัตโนมัติตามโปรแกรมที่มนุษย์ได้ป้อนเข้าไปเพื่อสั่งให้ทำงาน (สมโภชน์ ชื่นเอี่ยม, 2553, หน้า 11) องค์ประกอบที่ทำให้คอมพิวเตอร์ประมวลผลดังกล่าวได้นั้น ได้แก่ ฮาร์ดแวร์ (Hardware) และซอฟต์แวร์ (Software) ฮาร์ดแวร์ คือ ตัวเครื่องคอมพิวเตอร์และอุปกรณ์รายรอบ (Peripheral Devices) ซึ่งสามารถสัมผัสได้จับต้องได้ ส่วนซอฟต์แวร์ คือ โปรแกรมที่ใช้สั่งงานเครื่อง (วัชราภรณ์ สุริยาภิวัฒน์, 2553, หน้า 57) เรียกว่า โปรแกรมคอมพิวเตอร์ซึ่งไม่สามารถสัมผัสได้จับต้องได้ การพัฒนาซอฟต์แวร์ต้องอาศัยโปรแกรมภาษาคอมพิวเตอร์เป็นเครื่องมือ ดังนั้นจำเป็นอย่างยิ่งที่ต้องเรียนรู้การเขียนโปรแกรมเพื่อสั่งให้คอมพิวเตอร์ทำงานตามความต้องการ รายละเอียดของบทนี้กล่าวถึง โปรแกรมคอมพิวเตอร์ โปรแกรมภาษาคอมพิวเตอร์ การพัฒนาโปรแกรมโดยใช้โปรแกรมภาษาคอมพิวเตอร์ ขั้นตอนการพัฒนาโปรแกรมด้วยภาษาคอมพิวเตอร์และการวิเคราะห์ปัญหาก่อนการเขียนโปรแกรม

โปรแกรมคอมพิวเตอร์

โปรแกรมคอมพิวเตอร์ (ซอฟต์แวร์) คือ ชุดคำสั่งที่สั่งให้คอมพิวเตอร์สามารถทำงานตามที่ต้องการ โดยแปลงคำสั่งให้เป็นสัญญาณทางอิเล็กทรอนิกส์ไปควบคุมฮาร์ดแวร์ โปรแกรมคอมพิวเตอร์แบ่งออกเป็น 2 ประเภท คือ โปรแกรมระบบและโปรแกรมประยุกต์ (สมโภชน์ ชื่นเอี่ยม, 2553, หน้า 187) โปรแกรมระบบทำหน้าที่ควบคุมการทำงานของคอมพิวเตอร์รวมถึงอุปกรณ์ โปรแกรมเหล่านี้ได้แก่ ระบบปฏิบัติการ (Operating system) โปรแกรมอรรถประโยชน์ (Utility program) โปรแกรมภาษาคอมพิวเตอร์ (Programming languages) เป็นต้น ส่วนโปรแกรมประยุกต์ คือ โปรแกรมที่เขียนขึ้นเพื่อสั่งให้คอมพิวเตอร์ทำงานเฉพาะอย่าง (วัชราภรณ์ สุริยาภิวัฒน์, 2553, หน้า 143) เช่น โปรแกรมที่ใช้ในการจัดการกราฟิก โปรแกรมเกม โปรแกรมสำหรับสำนักงาน เป็นต้น ผู้ที่มีบทบาทสำคัญที่สุดต่อการเขียนโปรแกรม คือ โปรแกรมเมอร์ (Programmer) ซึ่งเป็นผู้เขียนชุดคำสั่งต่าง ๆ เพื่อทำให้คอมพิวเตอร์ทำงานได้ตามที่ผู้ใช้ต้องการ

โปรแกรมภาษาคอมพิวเตอร์อยู่ในกลุ่มของโปรแกรมระบบ มีหน้าที่ การพัฒนาโปรแกรมอื่น ๆ ผู้ที่ใช้ภาษาคอมพิวเตอร์คือโปรแกรมเมอร์ พิจารณาตามภาพที่ 1.1 แสดงตำแหน่งของโปรแกรมภาษาคอมพิวเตอร์ในโครงสร้างของซอฟต์แวร์ ซึ่งจะพบตำแหน่งของโปรแกรมที่ใช้ในการศึกษาในตำราเล่มนี้



ภาพที่ 1.1 แสดงผังโปรแกรมคอมพิวเตอร์

โปรแกรมระบบ คือ โปรแกรมที่สร้างขึ้นมาเพื่อควบคุมฮาร์ดแวร์ต่าง ๆ ของคอมพิวเตอร์โดยเฉพาะ ซึ่งโปรแกรมถูกแบ่งออกเป็น ระบบปฏิบัติการ โปรแกรมอรรถประโยชน์ และโปรแกรมภาษาคอมพิวเตอร์

- โปรแกรมระบบปฏิบัติการ เป็นโปรแกรมที่เขียนขึ้นมาเพื่อควบคุมให้คอมพิวเตอร์สามารถติดต่อสื่อสารกับมนุษย์ได้ โดยโปรแกรมจะจัดเตรียมคำสั่งต่าง ๆ เพื่อสื่อสารกับมนุษย์ โดย

ผ่านทั้งส่วนของข้อความ และกราฟิก และในปัจจุบันจะเน้นการติดต่อกับผู้ใช้ด้วยกราฟิกทั้งหมด โปรแกรมกลุ่มนี้ได้แก่ ระบบปฏิบัติการไมโครซอฟต์วินโดว์ (Microsoft Windows) ลีนุกซ์ (Linux) บีโอเอส (BeOS) เป็นต้น

- โปรแกรมอรรถประโยชน์ คือ โปรแกรมที่ใช้สำหรับจัดการระบบคอมพิวเตอร์เพื่ออำนวยความสะดวก และการจัดการปัญหาบางอย่างในการใช้คอมพิวเตอร์ เช่น โปรแกรมช่วยในการจัดการโปรแกรมไวรัส (Anti-Virus Program) โปรแกรมจัดการระบบของฮาร์ดดิสก์ เช่น นอร์ตันยูทิลิที (Norton Utility) โปรแกรมช่วยในการดาวน์โหลดข้อมูล เป็นต้น

- โปรแกรมภาษาคอมพิวเตอร์ เป็นโปรแกรมที่ใช้สำหรับนำมาเขียนโปรแกรมให้เกิดซอฟต์แวร์อื่น ๆ หรือใช้สำหรับเขียนโปรแกรมใช้งานตามความต้องการของผู้เขียน โปรแกรมภาษาคอมพิวเตอร์ เช่น แอสเซมลี (Assembly) ซี(C), ปาสคาล (Pascal), เบสิค (Basic), ไมโครซอฟต์วิสชวลซีพลัสพลัส(Microsoft Visual C++), จาวา (Java), ไมโครซอฟต์วิสชวลเบสิคดอทเน็ต (Microsoft Visual Basic .NET), โปรล็อก 2 (Prolog 2), เอดา (Ada) เป็นต้น ซึ่งโปรแกรมภาษาที่จะใช้สำหรับตำราเล่มนี้คือโปรแกรมภาษาจาวา โดยรายละเอียดเกี่ยวภาษาจาวาจะนำเสนอตั้งแต่บทที่ 2 เป็นต้นไป

โปรแกรมประยุกต์ คือ โปรแกรมที่สร้างขึ้นมาเพื่อใช้งานเฉพาะด้านอย่างใดอย่างหนึ่ง ดังตัวอย่างในภาพที่ 1.1 ที่แสดงงานของโปรแกรมประยุกต์บางส่วน ซึ่งอาจแบ่งออกเป็นโปรแกรมกราฟิก โปรแกรมทำงานเกี่ยวกับมัลติมีเดีย โปรแกรมสำหรับงานสำนักงาน และนอกจากนั้นในปัจจุบันยังมีโปรแกรมที่ทำงานเฉพาะด้านอื่น ๆ อีกมากมาย

โปรแกรมภาษาคอมพิวเตอร์

ภาษาคอมพิวเตอร์ คือ ภาษาที่ใช้สื่อสารกันระหว่างคอมพิวเตอร์กับมนุษย์ ธีรวัฒน์ ประกอบผล (2545, หน้า 3) และ เพ็ญศรี ปักกะสีนัง (2546, หน้า 97) กล่าวว่า ภาษาคอมพิวเตอร์เป็นการนำชุดคำสั่งแต่ละคำสั่งมาต่อกันเพื่อสั่งให้คอมพิวเตอร์ทำงาน การเขียนชุดคำสั่งนี้ไม่ว่าจะเขียนด้วยภาษาอะไรจะเรียกว่าโปรแกรมต้นฉบับ (Source program) หรือ รหัสต้นฉบับ (Source code)

ขณะที่ พนิดา พานิชกุล (2554, หน้า 2) กล่าวว่า ภาษาการโปรแกรมคอมพิวเตอร์ (Computer Programming Language) เป็นภาษาที่โปรแกรมเมอร์ใช้ในการเขียนชุดคำสั่งหรือเขียนโปรแกรมคอมพิวเตอร์เพื่อสั่งให้คอมพิวเตอร์ทำงาน

วิกีพีเดีย (Wikipedia, 2007) สารานุกรมออนไลน์ได้ให้ความหมายภาษาคอมพิวเตอร์เอาไว้ว่า **ภาษาคอมพิวเตอร์** หมายถึง ภาษาใด ๆ ที่ผู้ใช้งานใช้สื่อสารกับคอมพิวเตอร์หรือคอมพิวเตอร์ด้วยกัน แล้วคอมพิวเตอร์สามารถทำงานตามคำสั่งนั้นได้ คำนี้มักใช้เรียกแทนภาษาโปรแกรม

ระดับของภาษาคอมพิวเตอร์แบ่งได้ดังนี้

1. ภาษาระดับต่ำ ได้แก่ ภาษาเครื่องและภาษาแอสเซมบลี

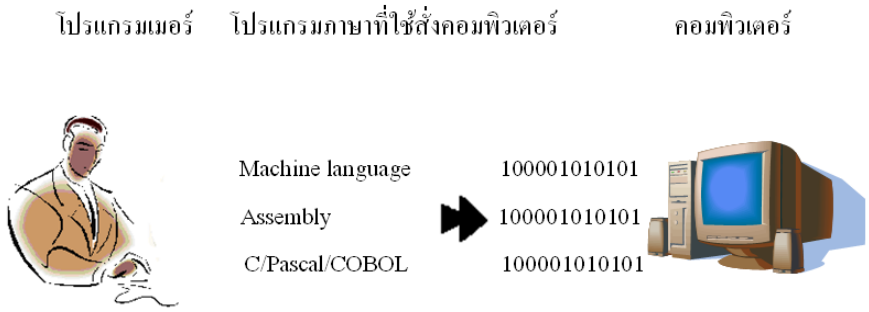
2. ภาษาระดับสูง ได้แก่ ภาษาที่ใช้อักขระหรือสัญลักษณ์ (Mnemonic) ที่ใกล้เคียงกับภาษามนุษย์ในการสื่อสารกับคอมพิวเตอร์

3. ภาษาสคริปต์ ได้แก่ ภาษายุคใหม่ที่ใช้สำหรับการเขียนโปรแกรมเพื่อนำเสนอข้อมูลผ่านระบบอินเทอร์เน็ต โดยเขียนคล้ายกับภาษาระดับสูงทั่วไป แต่จะเขียนรหัสต้นฉบับแทรกเข้าไปในภาษาเอชทีเอ็มแอล (HTML) โดยมุ่งเน้นสำหรับจัดการข้อมูลผ่านระบบเครือข่าย และเรียกใช้งานผ่านโปรแกรมบราวเซอร์ (Browser software)

โดยปกติแล้วแต่ละภาษาคอมพิวเตอร์จะมีรูปแบบเฉพาะของตนเอง ซึ่งหน้าที่หลักของภาษาคอมพิวเตอร์ คือ การสั่งการให้คอมพิวเตอร์ทำงานตามที่มนุษย์ต้องการ ดังนั้นในการที่สื่อสารกับคอมพิวเตอร์ด้วยภาษาใด ๆ นั้นผู้สั่งต้องเรียนรู้ภาษานั้น ๆ ก่อน และการที่จะควบคุมการสื่อสารดังกล่าวจะต้องผ่านเครื่องมือที่ใช้สั่งคอมพิวเตอร์ได้ ได้แก่โปรแกรมคอมพิวเตอร์หรือซอฟต์แวร์นั่นเอง

ดังนั้น **โปรแกรมภาษาคอมพิวเตอร์** คือ โปรแกรมที่ใช้เพื่อการพัฒนาหรือสร้างโปรแกรมอื่น ๆ โดยแบ่งออกเป็นหลายระดับ เช่น ระดับภาษาระดับต่ำ ภาษาระดับสูง และภาษาธรรมชาติ เป็นต้น ผู้ใช้ภาษาคอมพิวเตอร์ คือ โปรแกรมเมอร์ ซึ่งเป็นผู้ออกแบบและเขียนโปรแกรมให้ผู้อื่นใช้งาน โดยโปรแกรมเมอร์จะเขียนโปรแกรมด้วยภาษาคอมพิวเตอร์ตามความถนัดและเหมาะสม หลังจากนั้นจะแปลคำสั่งของโปรแกรมที่เขียนขึ้นด้วยคอมไพเลอร์ (Compiler) หรืออินเทอร์พรีเตอร์ (Interpreter) ให้เป็นคำสั่งที่ติดต่อกับคอมพิวเตอร์ได้ ซึ่งอยู่ในรูปแบบของภาษาเครื่อง เพื่อควบคุมให้คอมพิวเตอร์ทำงานได้ตามที่ต้องการ

ภาพที่ 1.2 แสดงแนวคิดของการเขียนโปรแกรม ซึ่งโปรแกรมเมอร์จะออกแบบและเขียนโปรแกรมด้วยโปรแกรมภาษาที่แตกต่างกันตามความถนัดและเหมาะสมกับแต่ละงาน หลังจากนั้นจะแปลคำสั่งของโปรแกรมที่เขียนขึ้นด้วยคอมไพเลอร์ (Compiler) หรืออินเทอร์พรีเตอร์ (Interpreter) ให้เป็นคำสั่งที่ติดต่อกับคอมพิวเตอร์ได้ในลักษณะของภาษาเครื่องด้วยรหัสเลขฐานสอง เพื่อควบคุมให้คอมพิวเตอร์ทำงานได้ตามที่ต้องการ



ภาพที่ 1.2 แสดงความสัมพันธ์ระหว่างโปรแกรมภาษากับการเขียนโปรแกรม

จากภาพที่ 1.2 อธิบายว่าได้เมื่อโปรแกรมเรียนรู้รูปแบบก็นำสิ่งที่ต้องการแก้ไขปัญหามาเขียนด้วยคำสั่งตามรูปแบบของภาษานั้น ๆ หลังจากนั้นตัวแปลภาษาซึ่งเป็นโปรแกรมที่เขียนขึ้นมาโดยบริษัทผู้ผลิตคอมพิวเตอร์หรือบริษัทผู้ผลิตซอฟต์แวร์ (พนิดา พานิชกุล, 2554, หน้า 2) มาแปลให้เป็นภาษาเครื่องโดยรูปแบบการแปลนั้นเป็นแบบอินเทอร์พรีเตอร์หรือคอมไพเลอร์ ซึ่งจะตรวจสอบคำสั่งแล้วแสดงผลหรือดำเนินการให้ได้ดังที่ผู้เขียนต้องการให้คอมพิวเตอร์ดำเนินการ

ยุคของโปรแกรมภาษาคอมพิวเตอร์

พนิดา พานิชกุล (2554, หน้า 3-5) ได้แบ่งยุคของภาษาคอมพิวเตอร์ไว้ 5 ยุค คือ ยุคที่หนึ่งเป็นภาษาระดับล่างหรือภาษาเลขฐานสองเท่านั้นสำหรับสั่งให้คอมพิวเตอร์ทำงาน ยุคที่สองเป็นยุคของภาษาสัญลักษณ์ที่เขียนคำสั่งให้เข้าใจง่าย และสิ้นลงด้วยภาษาอังกฤษ ภาษายุคนี้ได้แก่ภาษาแอสเซมบลี (Assembly) ยุคที่สาม เป็นภาษาระดับสูงซึ่งใกล้เคียงภาษามนุษย์มากขึ้น ภาษายุคนี้ได้แก่ ซี (C) ปาสคาล (Pascal) โคบอล (COBOL) เป็นต้น ภาษายุคที่สี่ เป็นภาษาระดับสูงแต่มีความสะดวกรวดเร็วในการพัฒนาโปรแกรม และมีส่วนการติดต่อที่สวยงาม ตัวอย่างของภาษายุคนี้ได้แก่ ภาษาจาวา ภาษาวิชวลเบสิค เป็นต้น ส่วนภาษายุคที่ห้า คือภาษาที่ใช้พัฒนาซอฟต์แวร์สำหรับระบบผู้เชี่ยวชาญเน้นการใช้งานแบบปัญญาประดิษฐ์

ส่วน วัชราภรณ์ สุริยาภิวัฒน์ (2553, หน้า 149-153) แบ่งกลุ่มภาษาออกเป็นกลุ่มภาษาเชิงโพรซีเดอรัล (Procedural Programming Language) และโปรแกรมภาษาเชิงวัตถุ (OOP) โดยที่โปรแกรมเชิงโครงสร้างถูกแบ่งออกเป็นภาษาเครื่องจักร ภาษาสัญลักษณ์ และภาษาระดับสูง

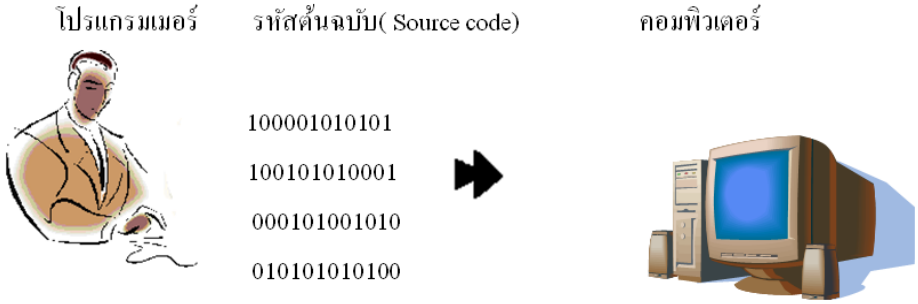
วิวัฒนาการของภาษาคอมพิวเตอร์จะถูกแบ่งออกเป็นยุค และถูกแบ่งออกเป็นระดับต่างกัน โดยแบ่งระดับของภาษาคอมพิวเตอร์ออกเป็น 5 ระดับ ได้แก่ ภาษาเครื่อง ภาษาแอสเซมบลี ภาษาระดับสูง ภาษาระดับสูงมาก และภาษาธรรมชาติ สำหรับในตำราเล่มนี้จะแบ่งโดยเน้นด้านการ

พัฒนาโปรแกรมภาษาตามการเขียนโปรแกรมโดยแบ่งดังเป็นยุคที่ 1 ถึงยุคที่ 5 โดยเน้นยุคที่ 4 (Fourth Generation Language: 4GL) ที่มีบทบาทในปัจจุบันและยุคที่ 5 (Fifth Generation Language: 5GL) ซึ่งจะมีความสำคัญมากในอนาคต รวมถึงแนวคิดการพัฒนาโปรแกรมคอมพิวเตอร์ในยุคที่ N ซึ่งการเขียนโปรแกรมอาจมิใช่เพียงโปรแกรมเมอร์เท่านั้น แต่มนุษย์ทุกคนจำเป็นต้องเขียนโปรแกรมด้วย รายละเอียดของแต่ละยุคแสดงได้พอสังเขป ดังนี้

ยุคที่ 1 ภาษาเครื่อง

การพัฒนาโปรแกรมคอมพิวเตอร์ในยุคแรก เรียกว่า ภาษาเครื่อง หรือรหัสเครื่อง (Machine language or Machine code) การเขียนโปรแกรมใช้การป้อนชุดคำสั่งโดยใช้รหัสเลขฐานสองคือ 0 กับ 1 ให้คอมพิวเตอร์โดยตรง ทำให้การเขียนโปรแกรมเป็นภาระหนักมากของโปรแกรมเมอร์ เนื่องจากไม่มีเครื่องมือที่ช่วยพัฒนาโปรแกรม ภาพที่ 1.3 แสดงแนวคิดของการเขียนโปรแกรมในยุคดังกล่าวนี้

ข้อดีของโปรแกรมภาษายุคนี้ คือ เครื่องคอมพิวเตอร์ไม่ต้องแปลรหัสต้นฉบับ (Source code) สามารถนำไปใช้สั่งงานคอมพิวเตอร์ได้ทันที ทำให้การทำงานเร็วมาก แต่ข้อเสีย คือ การเขียนโปรแกรมทำได้ยาก เพราะต้องจดจำคำสั่งและข้อมูลต่าง ๆ เป็นรหัสเลขฐานสอง จึงทำให้ต้องพัฒนาโปรแกรมภาษาในระดับที่สูงขึ้น

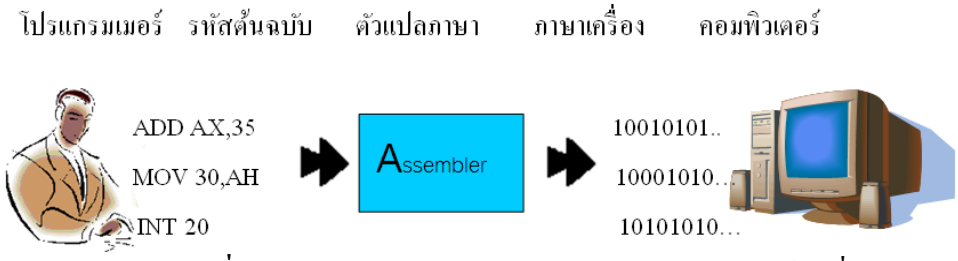


ภาพที่ 1.3 แสดงการพัฒนาโปรแกรมของโปรแกรมเมอร์ยุคที่ 1

ยุคที่ 2 ภาษาแอสเซมบลี

ยุคที่ 2 เริ่มขึ้นเมื่อประมาณต้นของช่วง 1950 ยุคนี้มีแนวคิดที่จะทำให้ภาษาคอมพิวเตอร์มีความใกล้เคียงภาษามนุษย์มากขึ้น โดยนำเอาภาษาอังกฤษมาสร้างเป็นคำสั่ง ภาษาในยุคนี้คือภาษาแอสเซมบลี (Assembly language) มีคำสั่งซึ่งเป็นคำภาษาอังกฤษง่าย ๆ เช่น ADD, SUB,MUL,DIV แทนการบวก ลบ คูณ และหาร เป็นต้น การเขียนโปรแกรมในยุคนี้ตัวแปลภาษาจะแปลจากภาษาอังกฤษมาเป็นภาษาเครื่องโดยใช้ตัวแปลภาษาที่เรียกว่า แอสเซมเบลอร์ (Assembler)

โปรแกรมเมอร์จะต้องมีความรู้ทั้งรูปแบบการใช้คำสั่งและระบบของฮาร์ดแวร์เป็นอย่างดี จึงจะสามารถเขียนโปรแกรมได้ดี แนวคิดในการเขียนโปรแกรมในยุคนี้แสดงดังภาพที่ 1.4



ภาพที่ 1.4 แสดงการพัฒนาโปรแกรมของโปรแกรมเมอร์ยุคที่ 2

เครื่องมือที่ใช้ในการพัฒนาโปรแกรมภาษาในยุคนี้มีให้เลือกมากขึ้น เช่นโปรแกรมเทอร์โบแอสเซมบลี (Turbo Assembly) แมโครแอสเซมลบลี (Macro Assembly) เป็นต้น

ข้อดีของการพัฒนาโปรแกรมแบบนี้ คือ การทำงานของโปรแกรมจะทำงานเร็วมาก แต่ข้อเสีย คือ โปรแกรมเมอร์ต้องเรียนรู้สถาปัตยกรรมของคอมพิวเตอร์ โดยเฉพาะเครื่องคอมพิวเตอร์แต่ละยี่ห้อ แต่ละรุ่นต่างมีสถาปัตยกรรมที่แตกต่างกันไป ทำให้แทนที่โปรแกรมเมอร์จะทุ่มเทให้กับการเขียนโปรแกรมอย่างเดียวกลับต้องศึกษาสถาปัตยกรรมคอมพิวเตอร์อีก จึงจำเป็นต้องพัฒนาโปรแกรมภาษาให้ง่ายต่อการเขียนโปรแกรมให้มากขึ้น

ยุคที่ 3 ภาษาระดับสูง

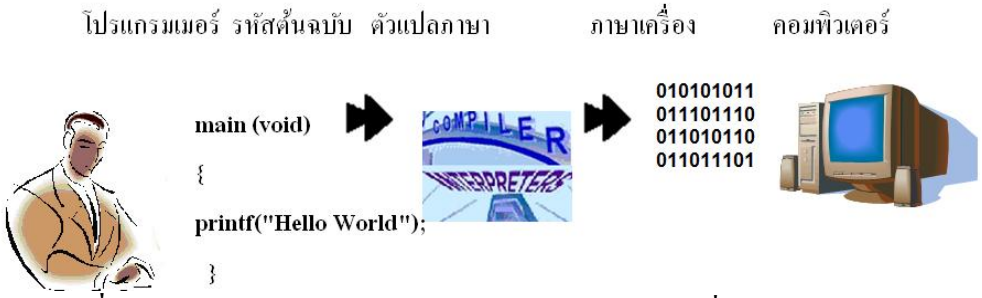
ยุคนี้เริ่มขึ้นเมื่อประมาณช่วงกลาง ค.ศ. 1950 ซึ่งถือว่าเป็นยุคแรกของภาษาระดับสูง (High level language) เพราะยุคที่ 1 และ 2 ถือว่าเป็นยุคของภาษาระดับต่ำ (Low level language) ที่ยังไม่เหมือนภาษามนุษย์เท่าใดนัก แต่ยุคนี้ภาษาคอมพิวเตอร์ใกล้เคียงกับภาษาของมนุษย์มากขึ้นอีก โดยใช้ประโยคหรือวลีในภาษาอังกฤษเพื่อเขียนโปรแกรม ข้อดีของภาษาในยุคนี้คือ

- การเขียนโปรแกรมใช้เวลาน้อยและต้นทุนต่ำลง
- โปรแกรมเมอร์ไม่จำเป็นต้องเรียนรู้สถาปัตยกรรมของคอมพิวเตอร์ ทำให้การเขียนโปรแกรมทำได้กว้างขวางมากขึ้น

การเขียนโปรแกรมยุคนี้มีชื่อเรียกหลากหลาย เช่นโปรแกรมแบบโครงสร้าง (Structure program) โปรแกรมแบบโมดูลหรือแบบโพรซีเดอรอล (Modular program or procedural program) มีหลายภาษาที่เกิดขึ้นในยุคนี้ การเขียนโปรแกรมยุคนี้โปรแกรมเมอร์ต้องเขียนรหัสต้นฉบับของคำสั่งทั้งหมดด้วยตนเอง

เครื่องมือช่วยพัฒนา ได้แก่โปรแกรมอิดิเตอร์ (Editor) ที่ติดมากับตัวแปรภาษาแต่ละภาษา โปรแกรมเมอร์จะเขียนรหัสต้นฉบับทั้งหมด จากนั้นใช้ตัวแปลภาษาซึ่งอาจเป็นคอมไพเลอร์หรือ

อินเตอร์พรีเตอร์ เพื่อแปลรหัสต้นฉบับให้เป็นภาษาเครื่องสำหรับสั่งให้คอมพิวเตอร์ทำงาน ภาษาคอมพิวเตอร์ยุคนี้มีหลากหลายโปรแกรมที่เหมาะกับงานที่แตกต่างกัน เช่น ฟอร์แทรน (FORTRAN) เหมาะกับงานคำนวณทางด้านวิทยาศาสตร์ ซี ปาสคาล เบสิค เหมาะกับงานทั่วไป ภาษาโคบอลเหมาะกับงานทางด้านธุรกิจ เป็นต้น ภาพที่ 1.5 แสดงแนวคิดของการพัฒนาโปรแกรมภาษาในยุคนี้



ภาพที่ 1.5 แสดงการพัฒนาโปรแกรมของโปรแกรมเมอร์ยุคที่ 3

อย่างไรก็ตาม ถึงแม้ว่าภาษายุคนี้มีความใกล้เคียงกับภาษามนุษย์มากแล้วก็ตาม แต่ปัจจัยทางด้านธุรกิจและความต้องการให้โปรแกรมทำงานมีความยืดหยุ่นเพื่อให้รองรับการทำงานได้กว้างขวางและสามารถเข้ากันได้กับโปรแกรมอื่น ๆ ได้ดีมากยิ่งขึ้น อีกทั้งแนวคิดในการนำกลับมาใช้ใหม่ (Reuse) มีความจำเป็นมากยิ่งขึ้น ดังนั้นจึงเป็นตัวผลักดันให้มีการพัฒนาโปรแกรมภาษาที่ง่ายขึ้นไปอีก จึงเกิดโปรแกรมภาษาในยุคที่ 4 ขึ้นในเวลาต่อมา

ยุคที่ 4 ภาษาระดับสูงมาก

จากความจำเป็นในยุคที่ 3 ที่กล่าวมาแล้วนั้น จึงพัฒนาโปรแกรมภาษาแนวใหม่เพื่อสนองความต้องการด้านต่าง ๆ เหล่านั้น ซึ่งเกิดขึ้นเมื่อประมาณช่วง ค.ศ. 1970 ยุคนี้อาจกล่าวได้ว่าเป็นยุคนอนโพรซีเดอรัล (Non-procedural) การพัฒนาเน้นให้ได้ผลลัพธ์ตามความต้องการอย่างรวดเร็วมากกว่าที่จะเน้นว่าจะต้องเขียนอย่างไร ในยุคก่อนหน้านี้โปรแกรมเมอร์ต้องรู้คำสั่งและวิธีการเขียนทุกสิ่งทุกอย่างด้วยตนเอง แต่ยุคนี้มีเครื่องมือช่วยพัฒนาได้รวดเร็วขึ้นกว่าเดิมมาก หากเปรียบการเขียนโปรแกรมเป็นการประกอบรถยนต์ 1 คัน ในยุคที่ผ่านมาผู้ประกอบต้องสร้างชิ้นส่วนทุกอย่างด้วยตนเองก่อนจึงประกอบเป็นรถยนต์ได้ แต่ยุคนี้การประกอบรถยนต์ทำได้ในเวลาอันรวดเร็ว เนื่องจากมีผู้ผลิตชิ้นส่วนทั้งหมดไว้แล้ว ผู้สร้างมีหน้าที่ประกอบเลือกเอาชิ้นส่วนมาประกอบกันให้เหมาะสมเท่านั้น

การเขียนโปรแกรมยุคนี้เรียกว่าการเขียนโปรแกรมเชิงวัตถุ (Object Oriented Programming: OOP) ซึ่งมองการเขียนโปรแกรมเหมือนการสร้างวัตถุเป็นชิ้น ๆ โดยมีเครื่องมือ

ช่วยเขียนโปรแกรม เรียกว่า ตัวสร้าง (Generator) ที่ติดมากับแต่ละภาษา ทำหน้าที่ติดต่อกับโปรแกรมเมอร์แบบกราฟิกอินเตอร์เฟส (Graphic interface) บางทีอาจเรียกว่าเป็นการเขียนโปรแกรมแบบวิสชวล (Visual programming) โปรแกรมเมอร์ใช้วิธีแดร็กแอนด์ดร็อพ (Drag and Drop) คือ การคลิกเอาวัตถุที่ใช้เป็นส่วนประกอบต่าง ๆ ของโปรแกรม เช่น ปุ่ม ฟอร์ม ไอคอน มาวางในพื้นที่เขียนโปรแกรม หลังจากนั้นกำหนดคุณสมบัติและหน้าที่การทำงานที่ควบคุมด้วยเหตุการณ์ที่เข้ามากระทำต่อวัตถุที่สร้างขึ้น หลังจากนั้นตัวสร้างจะแปลงส่วนการเขียนโปรแกรมให้เป็นรหัสต้นฉบับแล้วแปลด้วยคอมไพเลอร์หรืออินเตอร์พรีเตอร์ที่ติดมาเพื่อให้สามารถสั่งคอมพิวเตอร์ได้ ตัวอย่างภาษายุคนี้ เช่น วิสชวลเบสิค วิสชวลซีพลัสพลัส บอร์แลนเดลไฟ (Borland Delphi) บอร์แลนซีพลัสพลัสบิวเดอร์ (Borland C++ Builder) จาวา วิสชวลสตูดิโอดอทเน็ต (VisualStudio.NET) เป็นต้น ภาพที่ 1.6 แสดงแนวคิดของโปรแกรมภาษาในยุคนี้

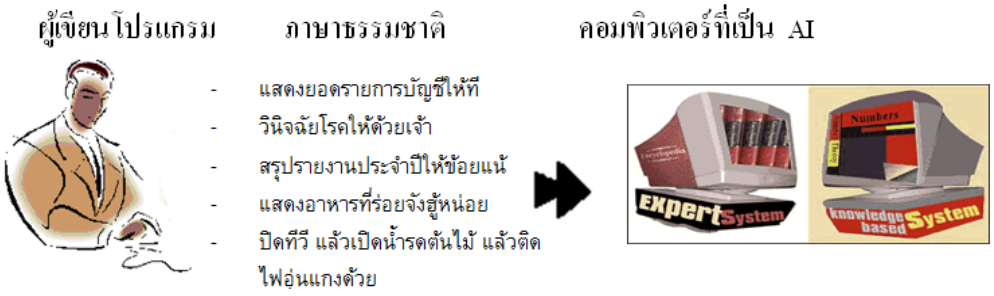
ข้อดีของโปรแกรมภาษาในยุคนี้ คือ การเขียนโปรแกรมทำได้รวดเร็วมาก ส่วนการติดต่อกับผู้ใช้มีความสวยงามน่าใช้มากขึ้น การใช้งานง่ายกว่าภาษายุคที่ 3 มาก แต่ข้อเสีย คือ โปรแกรมเมอร์ไม่สามารถรู้รายละเอียดบางอย่างที่ซ่อนเอาไว้โดยเครื่องมือช่วยพัฒนาเหล่านั้น โปรแกรมจะมีขนาดใหญ่มากขึ้น ฮาร์ดแวร์ที่ใช้จะต้องมีประสิทธิภาพสูงมากขึ้นด้วย เนื่องจากต้องมีการแปลคำสั่งหลายขั้นตอน

ภาพที่ 1.6 แสดงการพัฒนาโปรแกรมของโปรแกรมเมอร์ยุคที่ 4

ยุคที่ 5 ภาษาธรรมชาติ

ปัจจุบันเริ่มเข้าสู่ต้นยุคที่ 5 แล้ว ยุคนี้จะเน้นการนำปัญญาประดิษฐ์ (Artificial Intelligent: AI) มาใช้เป็นแนวคิดหลักของการเขียนโปรแกรม โดยเริ่มใช้ภาษาธรรมชาติของมนุษย์ (Natural language) ในแนวการเขียนโปรแกรมแบบตรรกะและโมเดล ภายใต้สภาวะแวดล้อมของโลกของความจริง สนับสนุนและยืดหยุ่นต่อพฤติกรรมที่เปลี่ยนไปภายใต้การจัดการของคอมพิวเตอร์ ปัจจุบันเราเริ่มเห็นหุ่นยนต์และซอฟต์แวร์ที่สามารถโต้ตอบกับมนุษย์ได้ ตัวอย่างภาษาช่วงเริ่มต้นของยุคนี้ เช่น ภาษาโปรลอก 2 (Prolog 2) เป็นโปรแกรมที่สามารถสอบถามข้อมูลด้วยประโยคที่เราใช้สื่อสารกัน ระบบผู้เชี่ยวชาญ (Expert System) ที่สามารถวินิจฉัยในเรื่องที่ระบบเชี่ยวชาญได้

ระบบฐานความรู้ (Knowledge Base System) ที่เก็บรวบรวมและประมวลผลข้อมูลอย่างชาญฉลาด เมื่อถึงยุคปลายของ ยุคนี้ทั้งฮาร์ดแวร์และซอฟต์แวร์จะเป็นยุคเอไอ (AI) อย่างสมบูรณ์ การเขียนโปรแกรมจะใช้ภาษาธรรมชาติ มนุษย์สามารถเขียนโปรแกรมคอมพิวเตอร์โดยภาษาที่ใช้สื่อสารกัน ภาพที่ 1.7 แสดงแนวคิดของโปรแกรมภาษาในยุคนี้



ภาพที่ 1.7 แสดงการพัฒนาโปรแกรมยุคที่ 5

ข้อดีของภาษายุคนี้ คือ การเขียนโปรแกรมทำได้โดยไม่ต้องเรียนรู้ภาษาคอมพิวเตอร์ให้ยุ่งยาก แต่ข้อเสีย คือ โปรแกรมเมอร์อาจถูกจำกัดความสามารถ เนื่องจากมีเครื่องมือช่วยในการเขียนโปรแกรมที่ฉลาดมากเกินไป ทำให้ไม่สามารถจัดการบางสิ่งบางอย่างตามที่ต้องการได้ กล่าวคือในระบบต่าง ๆ เหล่านั้นจะต้องทำการแปลคำสั่งเช่นเดียวกับยุคที่ผ่าน ๆ มา แต่ผู้เขียนโปรแกรมจะไม่รู้สึกว่าต้องทำเหมือนที่ผ่านมา เพราะระบบเหล่านั้นมีความฉลาด สามารถรับภาษาธรรมชาติแล้วแปลจนกระทั่งเป็นภาษาเครื่องได้

โปรแกรมภาษาคอมพิวเตอร์แบ่งเป็น ภาษาเครื่อง ภาษาระดับต่ำ ภาษาระดับสูง และภาษาธรรมชาติ ซึ่งต่างก็เป็นเครื่องมือสำหรับการเขียนโปรแกรมคอมพิวเตอร์ทั้งสิ้น ในหัวข้อถัดไปจะกล่าวแนวคิดในการเขียนโปรแกรมแบบโครงสร้างที่จะใช้งานในตำราเล่มนี้

การเขียนโปรแกรมคอมพิวเตอร์

การเขียนโปรแกรมคอมพิวเตอร์ คือ การสร้างกลุ่มคำสั่งเพื่อควบคุมให้คอมพิวเตอร์ประมวลผลตามที่ต้องการ โดยอาศัยโปรแกรมภาษาคอมพิวเตอร์เป็นเครื่องมือสร้างกลุ่มคำสั่งดังกล่าว สิ่งแรกที่โปรแกรมเมอร์ต้องทำ คือ ต้องเลือกโปรแกรมภาษาคอมพิวเตอร์ที่ต้องการใช้เป็นเครื่องมือ นำมาติดตั้งบนระบบปฏิบัติการที่มีอยู่ในเครื่องคอมพิวเตอร์ นอกจากนั้นโปรแกรมเมอร์จะต้องทำความเข้าใจกับขั้นตอนการเขียนโปรแกรมของแต่ละภาษาคอมพิวเตอร์ ในขณะเดียวกันจะต้องมีความรู้เกี่ยวกับรูปแบบของการเขียนโปรแกรม ส่วนนี้จะแสดงรายละเอียดดังกล่าวดังนี้

ธีรวัฒน์ ประกอบผล (2553, หน้า 11-14) และ วัชราภรณ์ สุริยาภิวัฒน์ (2553, หน้า 195) ต่างกล่าวตรงกันว่าขั้นตอนการเขียนโปรแกรมมีดังนี้

(1) การกำหนดและวิเคราะห์ปัญหา (Problem definition and problem analysis) ด้วยการกำหนดขอบเขตของปัญหา

(2) การเขียนผังงาน (Flow chart) และซูโดโค้ด (Pseudo coding) เป็นการแทนขั้นตอนต่าง ๆ ด้วยสัญลักษณ์

(3) การเขียนโปรแกรม (Programming) ด้วยการเขียนโปรแกรมตามรูปแบบของไวยากรณ์ตามรูปแบบของภาษาที่เลือกใช้

(4) การทดสอบและแก้ไขโปรแกรม (Program testing and debugging) เป็นการทดสอบว่าโปรแกรมที่เขียนขึ้นมานั้นถูกต้องตามรูปแบบที่เขียนหรือไม่ และทดสอบว่าโปรแกรมที่เขียนขึ้นเป็นไปตามที่วิเคราะห์ไว้หรือไม่

(5) การทำเอกสารและการบำรุงรักษาเป็นขั้นตอนของการทำเอกสารคู่มือ ทั้งคู่มือการใช้งานและคู่มือสำหรับโปรแกรมเมอร์

ขณะที่ กระทรวงศึกษาธิการ ได้กล่าวไว้ใน หลักการเขียนโปรแกรม คพ. ทป. 014 (2533, หน้า 46) ระบุขั้นตอนการเขียนโปรแกรมดังนี้

(1) การวิเคราะห์ปัญหา

(2) การออกแบบขั้นตอนการทำงานของโปรแกรม

(3) การลงรหัสโปรแกรม

(4) การทดสอบและแก้ไขโปรแกรม

(5) การทำเอกสารประกอบโปรแกรม

พนิดา พานิชกุล (2554, หน้า 9-10) ระบุว่าขั้นตอนการพัฒนาโปรแกรมคอมพิวเตอร์มีดังนี้

(1) การวิเคราะห์ปัญหา

(2) การออกโปรแกรม กำหนดส่วนนำเข้า ประมวลผล และส่วนการแสดงผล

(3) เขียนโปรแกรมโดยใช้ภาษาคอมพิวเตอร์

(4) การทดสอบและแก้ไขโปรแกรมเพื่อความถูกต้องก่อนนำไปใช้งานจริง

(5) การทำเอกสารและดูแลรักษาโปรแกรม

จากที่กล่าวมาข้างต้น ดังนั้นจึงสรุปขั้นตอนของการเขียนโปรแกรมได้ดังนี้

1. ศึกษาปัญหาและทำความเข้าใจสิ่งที่ต้องการของผู้ใช้

ขั้นตอนนี้เป็นขั้นตอนแรกสุด เพราะหากมองปัญหาไม่ออกก็ไม่สามารถที่จะเขียนโปรแกรมหรือสร้างงานออกมาได้อย่างถูกต้องตามความต้องการ ในขั้นตอนนี้สิ่งที่ต้องพิจารณาและทำความเข้าใจมีดังนี้

1.1 สิ่งที่โจทย์หรือผู้ใช้ต้องการ (Output) ของงาน ผู้เขียนจะต้องรู้อย่างชัดแจ้งว่าผู้ใช้ต้องการผลการทำงานอย่างไร แนวปฏิบัติของขั้นตอนนี้คือศึกษาจากโจทย์ที่ได้รับ หรือสอบถามรวบรวมข้อมูลจากผู้ปฏิบัติงานของผู้ใช้งานจริงให้ละเอียดที่สุด

1.2 สิ่งที่โจทย์ให้มาหรือปัจจัยต่าง ๆ ที่ทำให้เกิดผลลัพธ์ ซึ่งก็คือข้อมูลนำเข้า (Input) ดังนั้นผู้เขียนโปรแกรมจะต้องพิจารณาประเด็นนี้เป็นสิ่งสำคัญอันดับที่สอง โดยดูจากโจทย์กำหนดให้ว่ามีปัจจัยใดประกอบอยู่แล้วบ้าง และสิ่งใดจะต้องกำหนดหรือหาเพิ่มเติมเพื่อที่จะช่วยในการแก้ไขปัญหาดังกล่าวให้ได้ผลลัพธ์ตามต้องการ

1.3 วิธีการดำเนินงาน (Process) คือ ลำดับขั้นตอนการทำงาน เป็นกระบวนการเพื่อนำข้อมูลนำเข้ามาดำเนินการเพื่อให้ได้ผลลัพธ์

2. การออกแบบและกำหนดขั้นตอนการทำงานของโปรแกรม

ผู้ที่เพิ่งจะเริ่มเขียนโปรแกรมคอมพิวเตอร์นั้นจำเป็นอย่างยิ่งต้องมีแนวคิดเหมือนการแก้โจทย์หรือการแก้ปัญหาทางคณิตศาสตร์ กล่าวคือ ต้องคิดเป็นขั้นตอนและกำหนดลำดับการแก้ปัญหาให้ชัดเจนดังนี้คือ

2.1 ให้เขียนลำดับขั้นตอนการแก้ปัญหาเป็นขั้นตอนให้ชัดเจนที่สุดเท่าที่สามารถทำได้

2.2 แทนลำดับขั้นตอนการแก้ปัญหาดังกล่าวด้วยแผนภาพหรือผังงาน (Flowchart) คลาสไดอะแกรม (Class Diagram)

2.3 ลงรหัสโปรแกรม (Pseudo Code) หรือรหัสเทียมของโปรแกรม โดยรหัสเทียมดังกล่าวจะต้องดูจากผังการทำงานเป็นหลัก การลงรหัสเทียมนี้จะเขียนไว้เป็นกลาง ๆ อาจจะอิงโปรแกรมภาษาคอมพิวเตอร์ภาษาใดภาษาหนึ่งก็ได้หรือเขียนเป็นภาษาอังกฤษล้วน ๆ ก็ได้ หรือหากเป็นภาษาไทยก็ยังอนุโลมให้ใช้ได้เช่นกัน

3. ลงมือเขียนโปรแกรม

ในการเริ่มลงมือเขียนโปรแกรมคอมพิวเตอร์นั้น ผู้เขียนจะต้องดำเนินการดังต่อไปนี้

3.1 ศึกษารูปแบบไวยากรณ์ (Syntax) ของโปรแกรมภาษานั้น ๆ ซึ่งจะเป็นกฎเกณฑ์ข้อบังคับ หรือรูปแบบบังคับต่าง ๆ ของโปรแกรมภาษาที่ใช้ในการเขียนโดยโปรแกรมเมอร์ต้องศึกษาให้เข้าใจและแม่นยำ จึงจะสามารถเขียนโปรแกรมได้อย่างมีประสิทธิภาพ

3.2 ศึกษา คำสั่งปฏิบัติการ ตัวดำเนินการ ตัวแปร หรือฟังก์ชัน (Functions) ต่าง ๆ ที่มีในภาษาที่เลือก ซึ่งในขั้นตอนนี้ผู้ใช้จะต้องดำเนินการเขียนโปรแกรมจริง ๆ จึงจะสามารถเข้าใจและใช้งานได้อย่างถูกต้อง

4. ทดสอบโปรแกรม

เมื่อเขียนโปรแกรมแล้วการทดสอบการทำงานของโปรแกรมเพื่อตรวจสอบความถูกต้องของโปรแกรมมีขั้นตอนดังนี้

4.1 แปลรหัสต้นฉบับของโปรแกรมภาษาที่เลือกมาใช้เขียนอาจมีความแตกต่างกันและใช้คำสั่งที่ไม่เหมือนกัน แต่โดยทั่วไปจะขึ้นอยู่กับตัวแปลภาษาซึ่งถ้าเป็นคอมไพเลอร์ จะทำการแปลรหัสดังกล่าวเป็นภาษาเครื่องและอาจบันทึกเอาไว้ เพื่อให้สามารถเรียกใช้งานได้ในภายหลังโดยไม่ต้องทำการแปลคำสั่งใหม่ แต่ถ้าเป็นตัวแปลภาษาที่เป็นอินเตอร์พรีเตอร์จะต้องแปลทุกครั้งเมื่อต้องการรันโปรแกรมเพื่อใช้งาน

4.2 ทดสอบการรันของโปรแกรม หรือการเรียกใช้งานโปรแกรมแต่ละโปรแกรมภาษาจะมีการเรียกใช้ต่างกันบ้างแต่โดยทั่วไปจะทำได้หลังจากที่คอมไพล์แล้ว และโปรแกรมดังกล่าวไม่มีข้อผิดพลาด จึงจะสามารถเรียกใช้งานตามที่ต้องการผ่านระบบปฏิบัติการที่นำโปรแกรมไปติดตั้ง

5. จัดทำเอกสารและการบำรุงรักษา

5.1 การจัดทำเอกสารที่เกี่ยวข้อง เช่นคู่มือผู้ใช้ คู่มือการพัฒนาโปรแกรมและออกแบบระบบ และคู่มือการแก้ปัญหาสำหรับผู้ดูแลระบบหรือโปรแกรมเมอร์

5.2 บำรุงรักษาโปรแกรมด้วยการตรวจสอบความถูกต้องปรับปรุงให้ทำงานเร็วขึ้น หรือการปรับปรุงดัชนีของฐานข้อมูล หรือปรับปรุงรุ่นของซอฟต์แวร์ เป็นต้น

รูปแบบพื้นฐานของการเขียนโปรแกรมคอมพิวเตอร์

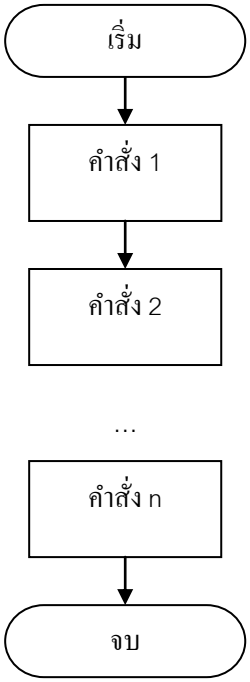
การพัฒนาโปรแกรมคอมพิวเตอร์ของโปรแกรมเมอร์จะต้องเลือกโปรแกรมภาษาที่จะใช้พัฒนาก่อนเป็นลำดับแรก หลังจากนั้นจะต้องศึกษารูปแบบการใช้งานของโปรแกรมภาษา ซึ่งแต่ละภาษาจะมีรูปแบบการเขียนโปรแกรมที่แตกต่างกันในแต่ละคำสั่ง ถึงแม้แต่ละภาษาจะมีรูปแบบของคำสั่งที่แตกต่างกันก็ตาม แต่จะมีรูปแบบการเขียนที่เป็นรูปแบบเดียวกันที่พอจะจำแนกได้ คือ การเขียนโปรแกรมแบบภาษาเครื่อง การเขียนโปรแกรมแบบภาษาโครงสร้าง การเขียนโปรแกรมแบบภาษาเชิงวัตถุ และการเขียนโปรแกรมด้วยภาษาธรรมชาติ แต่ในตำราเล่มนี้จะใช้การเขียนโปรแกรมแบบโครงสร้างซึ่งเป็นการเรียนรู้การเขียนโปรแกรมแบบง่ายและเหมาะสำหรับผู้เริ่มต้นเขียนโปรแกรมคอมพิวเตอร์ ดังนั้นต่อไปนี้จะนำเสนอเกี่ยวกับแนวคิดการเขียนโปรแกรมด้วยภาษาเชิงโครงสร้างและรูปแบบการเขียนโปรแกรมแบบนี้

กระทรวงศึกษาธิการ (2541, หน้า 9) ได้แบ่งรูปแบบการเขียนโปรแกรมออกเป็น เรียงลำดับ แบบทางเลือก และแบบวนซ้ำ ซึ่งรูปแบบการเขียนโปรแกรมแบบโครงสร้างเริ่มจากบนลงล่าง จากซ้ายไปขวาเสมอ หากแบ่งรูปแบบการพัฒนาอาจแบ่งรูปแบบการเขียนโปรแกรมเชิงโครงสร้างได้ดังนี้

1. โปรแกรมแบบเรียงลำดับ

การเขียนโปรแกรมแบบนี้ไม่มีความซับซ้อนใด ๆ โดยจะนำคำสั่งของโปรแกรมภาษาที่ใช้ในการเขียนมาเรียงต่อกันเป็นลำดับ ต่อเนื่องกันไปจากต้นจนจบ แสดงตัวอย่างแนวคิดการเขียนโปรแกรมดังกล่าวได้ดังภาพที่ 1.9

การพัฒนาโปรแกรมลักษณะนี้ไม่มีความซับซ้อนและเหมาะสำหรับผู้ที่เริ่มหัดเขียนโปรแกรม และผู้ที่ต้องการฝึกหัดใช้คำสั่งของภาษาคอมพิวเตอร์ภาษาใดภาษาหนึ่งในระยะเริ่มแรก ในตำราเล่มนี้บทต้น ๆ เป็นการเขียนโปรแกรมตามแนวทางนี้



ภาพที่ 1.8 แสดงตัวอย่างแนวคิดการเขียนโปรแกรมแบบตามลำดับ

ตัวอย่างต่อไปนี้แสดงลักษณะของงานที่ต้องเขียนโปรแกรมแบบตามลำดับ

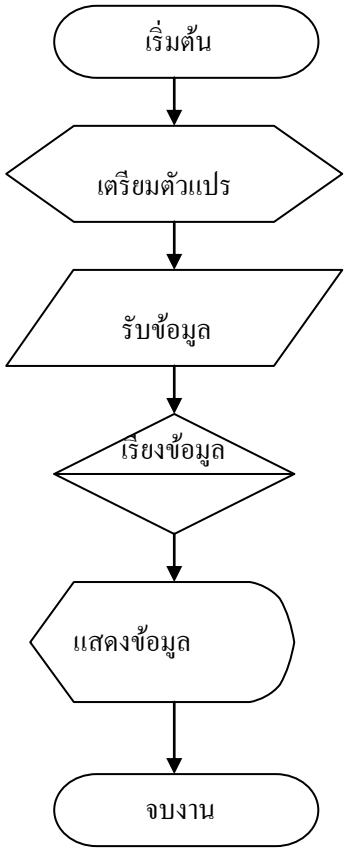
ตัวอย่างที่ 1.1 จงเขียนผังงานแบบตามลำดับของขั้นตอนต่อไปนี้

1. เริ่มต้น
2. เตรียมตัวแปรรับข้อมูล
3. รับข้อมูลจากผู้ใช้

4. คำนวณจัดเรียงข้อมูล

5. แสดงผลออกทางจอภาพ

6. จบงาน



ภาพที่ 1.9 แสดงตัวอย่างผังงานของตัวอย่าง 1.1

ตัวอย่างที่ 1.2 จงเขียนผังโปรแกรมรับข้อมูลชื่อ นามสกุล ที่อยู่ และอายุ หลังจากนั้นแสดงผลทางจอภาพ

วิธีคิด

จากโจทย์ ข้อมูลที่ต้องนำเข้า ได้แก่ ชื่อ สกุล และอายุ ข้อมูลออกแสดงผลคือข้อมูลเดียวกัน

อัลกอริทึม:

1. เริ่มต้น

2. กำหนดตัวแปร name , surname และ age

3. รับข้อมูล name

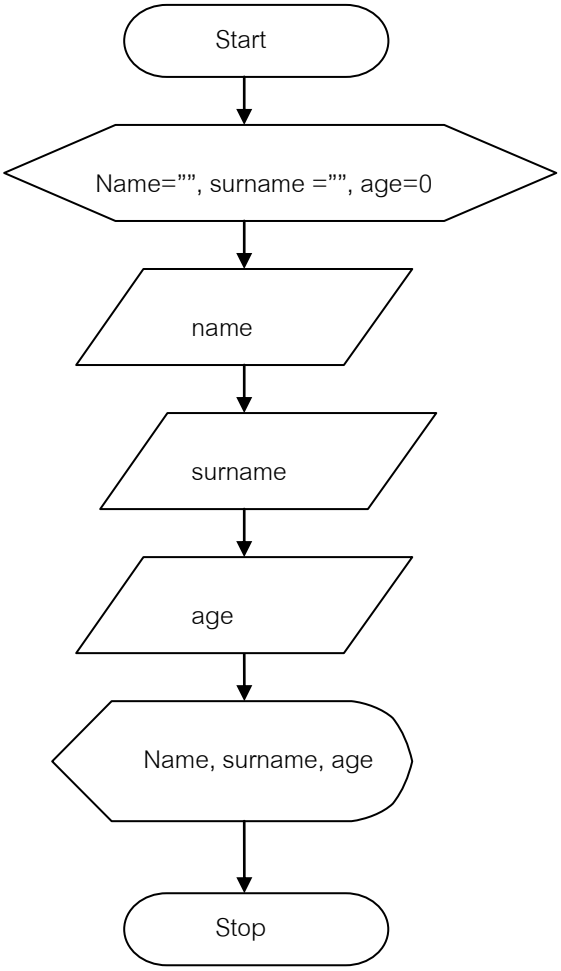
4. รับข้อมูล surname

5. รับข้อมูล age

6. แสดงผลข้อมูล name, surname, age

7. จบงาน

ผังโปรแกรม:

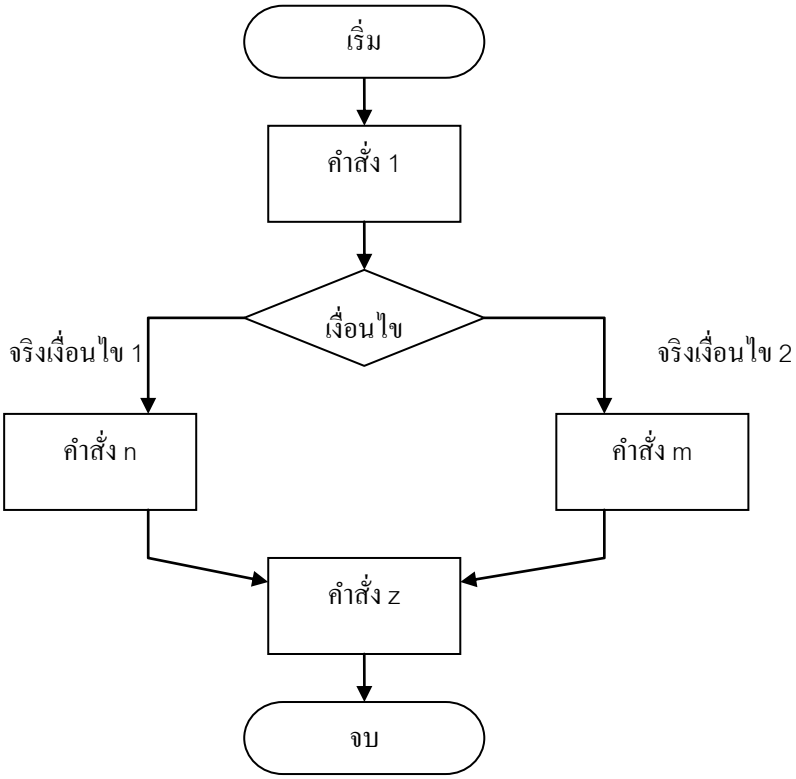


ภาพที่ 1.10 แสดงผังงานโปรแกรมของตัวอย่างที่ 1.2

2. การเขียนโปรแกรมแบบมีเงื่อนไข

การเขียนโปรแกรมแบบนี้ โดยหลักการทั่วไปจะเริ่มจากบนลงล่างตามแนวทางของโปรแกรมเชิงโครงสร้าง แต่ระหว่างทางจากบนลงล่างจะมีการตัดสินใจ โดยมีทางเลือกให้กับโปรแกรมเพื่อเลือกทำงานอย่างใดอย่างหนึ่งตามเงื่อนไขที่กำหนดไว้ ตัวอย่างแนวคิดแสดงดังภาพที่ 1.12

ลักษณะของโปรแกรมดังกล่าวนี้จะตรวจสอบเงื่อนไขว่าตรงตามเงื่อนไขใดแล้วจะไปทำคำสั่งตามที่กำหนดไว้ ซึ่งเงื่อนไขอาจมีมากกว่าหนึ่งเงื่อนไขก็ได้ และลักษณะของการตรวจสอบเงื่อนไขก็จะมีหลายรูปแบบเช่นกัน



ภาพที่ 1.11 แสดงแนวคิดของการเขียนโปรแกรมแบบมีเงื่อนไข

ตัวอย่างที่ 1.3 จงเขียนผังงานรับค่าอายุของผู้ใช้ หากอายุน้อยกว่าหรือเท่ากับ 20 แสดงว่าเป็นวัยรุ่น หากมากกว่านั้นแสดงว่าเป็นผู้ใหญ่

<u>วิธีคิด</u>

ข้อมูลนำเข้าคืออายุของผู้ใช้ และข้อมูลแสดงผล ได้แก่ ข้อความวัยรุ่น หรือวัยผู้ใหญ่

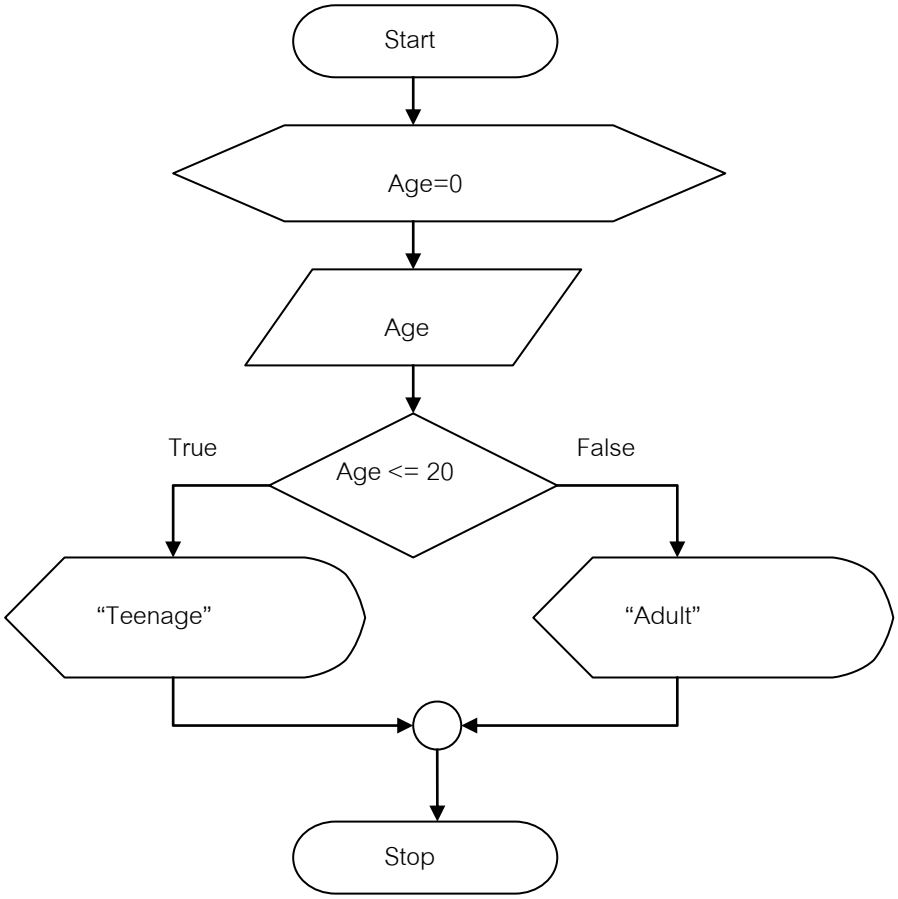
อัลกอริทึม:

1. เริ่มต้น
2. กำหนดตัวแปรรับข้อมูลนำเข้า age
3. รับข้อมูลอายุเก็บไว้ใน age
4. เปรียบเทียบว่า age น้อยกว่าหรือเท่ากับ 20 หรือไม่

1. ถ้าเป็นจริงแสดงข้อความ Teenage
2. ถ้าเป็นเท็จแสดงข้อความ Adult

5. จบงาน

ผังโปรแกรม:



ภาพที่ 1.12 แสดงผังโปรแกรมของตัวอย่างที่ 1.3

ตัวอย่างที่ 1.4 จงเขียนผังงานโปรแกรมของโปรแกรมให้ผู้ใช้เลือกเมนูดังนี้

เมนู 1 รับชื่อแล้วแสดงผลทางจอภาพ

เมนู 2 รับนามสกุลแล้วแสดงผลทางจอภาพ

เมนู 3 รับเงินเงินเดือนแล้วแสดงผล

เมนู 4 จบงาน

<u>วิธีคิด</u>

กำหนดข้อมูลนำเข้า มีตัวแปร 3 ตัวคือ name รับชื่อ surname เก็บนามสกุล และ salary เก็บเงินเดือน กำหนดตัวแปร menu แทนค่าที่จะใช้ในการตรวจสอบ

ข้อมูลแสดงผลได้แก่ตัวแปรทั้งสามที่รับเข้ามาแต่เลือกแสดงเฉพาะรายกรณี

อัลกอริทึม:

1. เริ่มต้น
2. กำหนดค่าตัวแปร name="", surname="", และ salary=0.0 , Menu=0
3. ให้ผู้ใช้ป้อนค่า Menu ทางอุปกรณ์แป้นพิมพ์
4. ตรวจสอบเงื่อนไข menu รายกรณี

Menu = 1

ให้ผู้ใช้ป้อนข้อมูลเก็บไว้ที่ name

แสดงผล name

Menu = 2

ให้ผู้ใช้ป้อนข้อมูลเก็บไว้ที่ surname

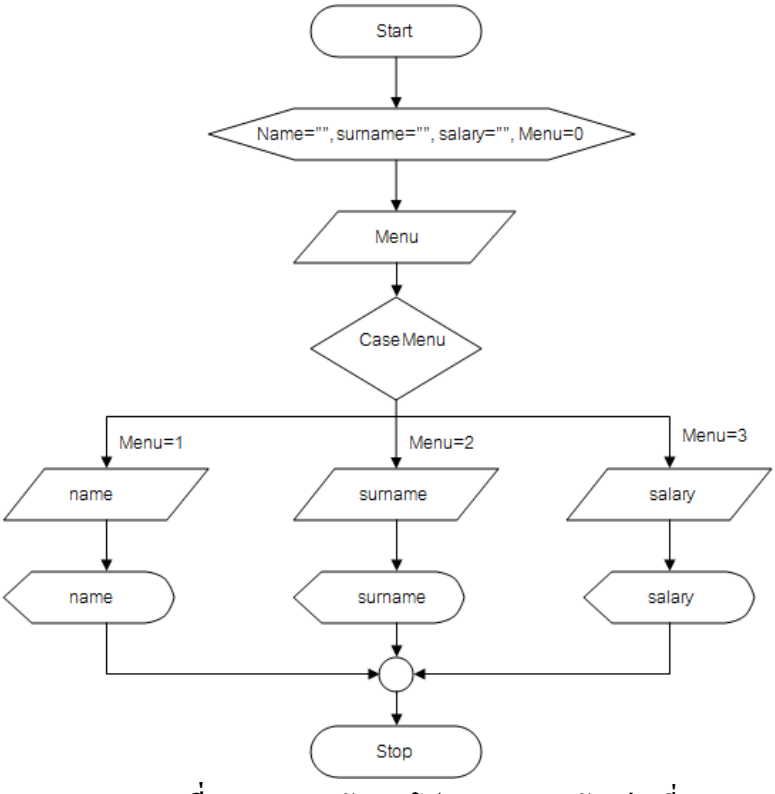
แสดงผล surname

Menu = 3

ให้ผู้ใช้ป้อนข้อมูลเก็บไว้ที่ salary

แสดงผล salary

5. จบงาน

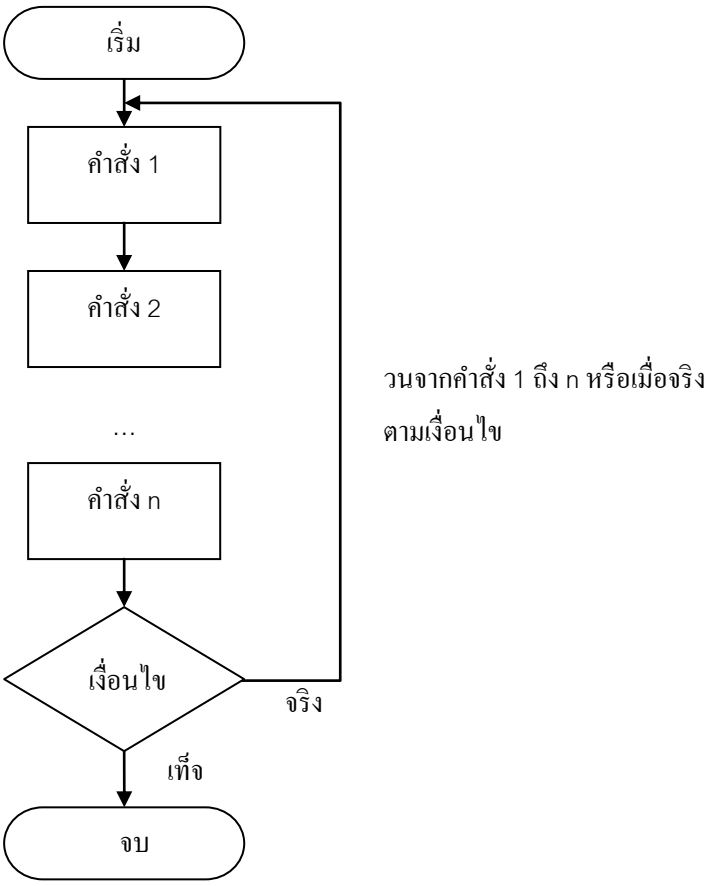


ภาพที่ 1.13 แสดงผังงานโปรแกรมของตัวอย่างที่ 1.4

3. การเขียนโปรแกรมแบบวนซ้ำ

การเขียนโปรแกรมแบบนี้ได้ใช้แนวคิดจากจุดเด่นของคอมพิวเตอร์ที่สามารถทำงานเดิมซ้ำ ๆ กันโดยไม่เบื่อหน่าย หลักการเขียนแบบนี้ยังคงเขียนจากบนลงล่างและเมื่อมีกลุ่มคำสั่งใดหรือโมดูลใดต้องทำงานซ้ำ ก็จะวนซ้ำตราบที่ตรงตามเงื่อนไขที่กำหนดไว้ เช่นกำหนดให้วนจนครบหนึ่งร้อยรอบ เป็นต้น แนวคิดดังกล่าวแสดงได้ดังภาพที่ 1.15

ลักษณะของการวนซ้ำหรือเงื่อนไขของการวนนั้นจะแตกต่างกันออกไป ขึ้นอยู่กับโปรแกรมภาษาที่ใช้ในการเขียนซึ่งจะทำให้รูปแบบการเขียนนั้นซับซ้อนแตกต่างกันออกไปด้วย



ภาพที่ 1.14 แสดงตัวอย่างการเขียนโปรแกรมแบบวนซ้ำ

ตัวอย่างที่ 1.5 จงเขียนผังงานโปรแกรมของโปรแกรมการป้อนค่าตัวเลข จนกระทั่งตัวเลขที่ป้อนเข้ามามากกว่า 100 ซึ่งถ้าค่าที่ป้อนเข้ามาน้อยกว่าหรือเท่ากับ 100 ให้แสดงข้อความ “It’s wrong number” แต่ถ้ามากกว่า 100 ให้แสดงข้อความ “That right” แล้วออกจากโปรแกรม

<u>วิธีคิด</u>

ข้อมูลนำเข้า คือตัวแปรที่จะรับค่าการป้อน จากผู้ใช้ กำหนดให้เป็น number

ข้อมูลออก คือ ข้อความที่แสดงว่า “That right” หรือ “It’s wrong number”

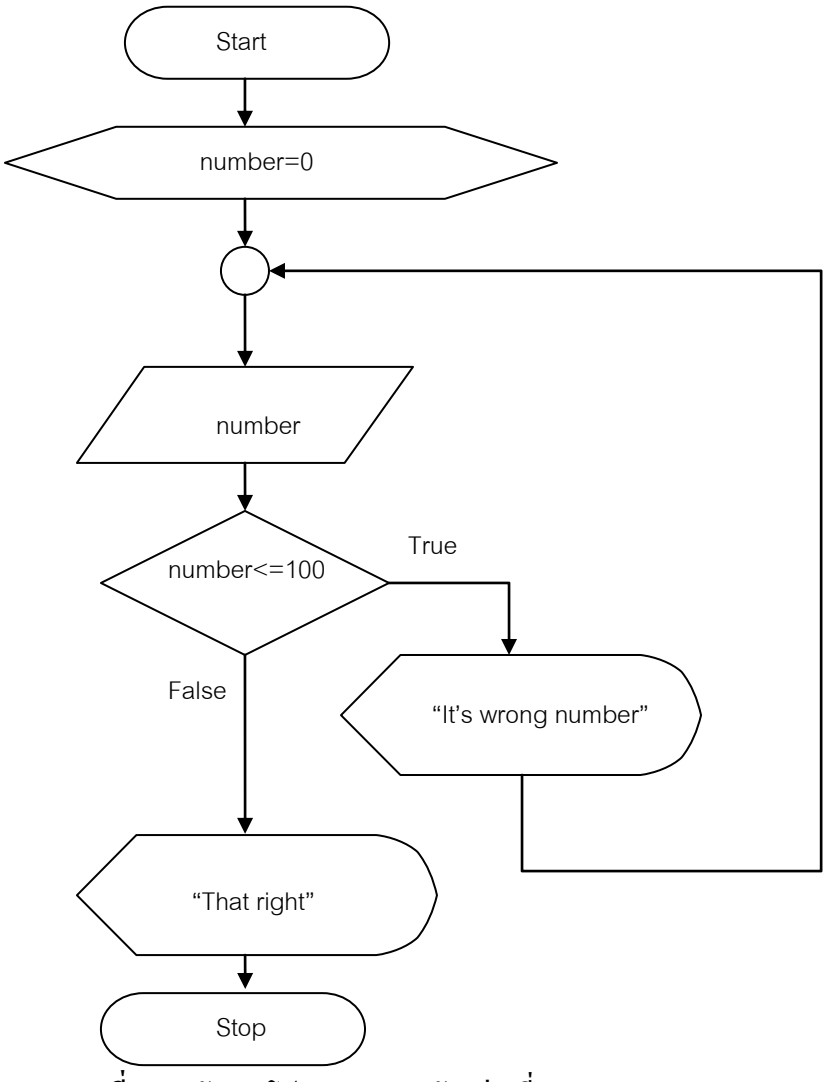
อัลกอริทึม:

1. เริ่มต้น
2. เตรียมตัวแปร number เพื่อรับค่าจากผู้ใช้
3. รับค่าการป้อนเข้ามาเก็บที่ตัวแปร number
4. เปรียบเทียบเงื่อนไขว่า number <= 100 หรือไม่

   หากเป็นจริงแสดงข้อความ “It’s wrong number” แล้ววนไปรับค่าการป้อนใหม่

   หากเป็นเท็จ ให้แสดงข้อความ “That right”
5. จบงาน

ผังโปรแกรม:



ภาพที่ 1.15 ผังงานโปรแกรมของตัวอย่างที่ 1.5