หน่วยการเรียนรู้ที่ 2 เรื่อง เขียนโปรแกรมภาษาขั้นพื้นฐาน

สาระที่ 3 เทคโนโลยีสารสนเทศและการสื่อสาร

มาตรฐาน ง 3.1 เข้าใจ เห็นคุณค่าและใช้กระบวนการเทคโนโลยีสารสนเทศในการสืบค้นข้อมูล การเรียนรู้ การสื่อสาร การแก้ปัญหา การทำงานและอาชีพอย่างมีประสิทธิภาพ ประสิทธิผล และมีคุณธรรม

ตัวชี้วัด ม.2 เขียนโปรแกรมภาษาขั้นพื้นฐาน

สาระการเรียนรู้แกนกลาง : หลักการพื้นฐานในการเขียนโปรแกรม

- แนวคิดและหลักการโปรแกรม โครงสร้างโปรแกรม ตัวแปร การลำดับ คำสั่ง การตรวจสอบเงื่อนไข การควบคุมโปรแกรม คำสั่งแสดงผล และรับข้อมูล การเขียนโปรแกรมแบบง่าย ๆ
- การเขียนสคริปต์

เขียนโปรแกรมภาษาขั้นพื้นฐาน

สาระสำคัญ

"การเขียนโปรแกรมคอมพิวเตอร์ ทุกภาษานั้นเหมือนกัน "สิ่งที่แตกต่างกันของแต่ละภาษา คือโครงสร้างภาษา (Syntax) แต่สิ่งที่เหมือนกันของทุกภาษา คือ การใช้ประสบการณ์จากภาษาหนึ่งไปใช้ในอีกภาษาหนึ่งได้ ด้วยการซึมซับเรื่องของ โครงสร้างการเขียนโปรแกรม (Structure Programming) จนเข้าใจ เพื่อควบคุมในสิ่งที่คล้าย ๆ กัน คือ ข้อมูลเข้า (input) กระบวนการทำงาน (process) และ แสดงผลข้อมูล (output)

ดังนั้น ถ้าเขียนโปรแกรมเพื่อกำหนดการทำงานในภาษาหนึ่งได้แล้ว การเขียนโปรแกรมแบบนั้นในภาษาอื่นย่อมไม่ใช่เรื่องยากเพียงแต่ต้องศึกษารูปแบบการเขียนของภาษาใหม่เพิ่มเติม แล้วนำประสบการณ์ที่เคยเขียน ไปสั่งให้ภาษาใหม่ทำงานตามต้องการได้

ความหมายภาษาคอมพิวเตอร์

ภาษาคอมพิวเตอร์ หมายถึง ภาษาใด ๆ ที่ผู้ใช้งานใช้สื่อสารกับคอมพิวเตอร์ หรือคอมพิวเตอร์ด้วยกัน แล้วคอมพิวเตอร์สามารถทำงานตามคำสั่งนั้นได้ ภาษาคอมพิวเตอร์มีมากมายหลายพันภาษา แต่ภาษาที่สั่งให้คอมพิวเตอร์ทำงานได้จริงนั้นมีภาษาเดียว คือ ภาษาเครื่อง (Machine language) ซึ่งเป็นภาษาที่บังคับการทำงานของเครื่องอย่างแท้จริง ข้อเสียของภาษาเครื่อง คือ ภาษาของแต่ละเครื่องจะไม่เหมือนกัน ทั้งนี้แล้วแต่การออกแบบระบบเครื่องว่าจะเป็นแบบใด นอกจากนี้เครื่องคอมพิวเตอร์แต่ละเครื่องที่สร้างขึ้นมานั้นส่วนใหญ่จะมีการแก้ไขปรับปรุงให้ดีขึ้นอยู่เสมอ ซึ่งจะทำให้ลักษณะของภาษาเครื่องเปลี่ยนแปลงแตกต่างออกไปอีก แต่ภาษาที่ไม่ใช่ภาษาเครื่องอาจจะยังเหมือนเดิมได้

ประเภทของภาษาคอมพิวเตอร์

ในปัจจุบันได้มีการพัฒนาภาษาโปรแกรมคอมพิวเตอร์ มากมายหลายภาษา และให้เหมาะกับการใช้งานประเภทต่างๆ โดยแบ่งระดับของภาษาโปรแกรมคอมพิวเตอร์ออกเป็น 5 ยุคคือ

ยุคที่ 1 ภาษาเครื่อง (Machine Language)

ภาษาเครื่อง เป็นภาษาโปรแกรมคอมพิวเตอร์ระดับต่ำที่สุด ซึ่งคอมพิวเตอร์เข้าใจได้ทันทีโดยไม่ต้องผ่านตัวแปลภาษาเพราะเขียนคำสั่งและแทนข้อมูลด้วยเลขฐานสอง (Binary Code) ทั้งหมด ซึ่งเป็นการเขียนคำสั่งด้วยเลข 0 หรือ 1 ดังตัวอย่างคำสั่งภาษาเครื่อง ดังนี้

คำสั่งภาษาเครื่อง (Machine Code)	ความหมาย
0010 0000	โหลดข้อมูลจากหน่วยความจำ
0100 0000	ดำเนินการบวกข้อมูล
0011 0000	เก็บข้อมูลลงในหน่วยความจำ

คอมพิวเตอร์แต่ละเครื่องจะมีภาษาเครื่องแตกต่างกันขึ้นอยู่กับชนิดของเครื่องคอมพิวเตอร์ และหน่วยประมวลผลกลาง (Central Processor Unit: CPU) โดยมีรูปแบบคำสั่งเฉพาะเครื่อง จึงไม่นิยมที่จะเขียนโปรแกรมด้วยภาษาเครื่อง เพราะทำการแก้ไข และเขียนโปรแกรมได้ยากทำให้เกิดยุ่งยากในการจดจำ และเขียนคำสั่งต้องใช้เวลามากในการเขียนโปรแกรม รวมทั้งการหาข้อผิดพลาดจากการทำงานของโปรแกรม และโปรแกรมที่เขียนขึ้นทำงานเฉพาะคอมพิวเตอร์ที่มีฮาร์ดแวร์เดียวกันเท่านั้น (Machine Dependent)

ข้อดีของภาษาเครื่อง คือสามารถเขียนโปรแกรมควบคุมการทำงานคอมพิวเตอร์ได้โดยตรง และสั่งงานให้คอมพิวเตอร์ทำงานได้อย่างรวดเร็ว

ยุคที่ 2 ภาษาแอสเซมบลี (Assembly Language)

ภาษาแอสเซมบลี จัดอยู่ในภาษาระดับต่ำ และเป็นภาษาที่พัฒนาต่อมาจากภาษาเครื่องในปี ค.ศ. 1952 ภาษาแอสเซมบลีมีความใกล้เคียงกับภาษาเครื่องมาก คือ 1 คำสั่งของภาษาแอสเซมบลีจะเท่ากับ 1 คำสั่งของภาษาเครื่อง โดยที่ภาษาแอสเซมบลีจะเขียนคำสั่งเป็นตัวอักษรภาษาอังกฤษ เพื่อใช้แทนคำสั่งภาษาเครื่อง ทำให้นักเขียนโปรแกรมสามารถเขียนโปรแกรมได้ง่ายขึ้น โดยการจดจำรหัสคำสั่งสั้น ๆ ที่จำได้ง่าย ซึ่งเรียกว่า นิวมอนิกโค้ด (Mnemonic code) เช่น

คำสั่งนิวมอนิกโคด (Mnemonic code)	คำสั่งภาษาเครื่อง	ความหมาย
LOAD	0010 0000	โหลดข้อมูลจากหน่วยความจำ
ADD	0100 0000	ดำเนินการบวกข้อมูล
SUB	1101 0000	ดำเนินการลบข้อมูล
MOV	1001 0000	ย้ายข้อมูลเข้าออกจากหน่วยความจำ
STROE	0011 0000	เก็บข้อมูลไว้ในหน่วยความจำ

ตัวอย่างของคำสั่งภาษาแอสเซมบลี ดังตัวอย่าง เช่น

```
CALL My Sub; transfer of control
MOV AX, 5; data transfer
ADD AX, 20; arithmetic
RET; return
```

เมื่อนักเขียนโปรแกรมเขียนโปรแกรมด้วยภาษาแอสเซมบลีแล้ว ต้องใช้ตัวแปลภาษาที่เรียกว่า **แอสเซมเบลอ** (Assembler) เพื่อแปลภาษาแอสเซมบลีให้เป็นภาษาเครื่อง จึงจะสามารถสั่งงานคอมพิวเตอร์ให้ทำงานได้ สรุปคำสั่งที่เขียนด้วยภาษาโปรแกรมคอมพิวเตอร์ ในยุคที่ 1 และที่ 2 จะต้องใช้เทคนิคการเขียนโปรแกรมสูง เพราะมีความยืดหยุ่นในการเขียนน้อยมาก และมีความยากในการเขียนคำสั่งสำหรับผู้เขียนโปรแกรม แต่สามารถควบคุมและเข้าถึงการทำงานของเครื่องคอมพิวเตอร์ได้โดยตรง และมีความรวดเร็วกว่าการใช้ภาษาระดับอื่นๆ

ยุคที่ 3 ภาษาระดับสูง (High-level Language)

ภาษาระดับสูงถือว่าเป็นภาษาโปรแกรมคอมพิวเตอร์ในยุคที่สาม (Third-generation language) ที่มีการใช้กันอย่างแพร่หลายในปี ค.ศ. 1960 โดยมีโครงสร้างภาษา และชุดคำสั่งเหมือนกับภาษาอังกฤษ รวมทั้งสามารถใช้นิพจน์ทางคณิตศาสตร์ในการคำนวณได้ด้วย ทำให้ผู้เขียนโปรแกรมสะดวกในการเขียนคำสั่ง และแสดงผลลัพธ์ได้ตามต้องการ ลดความยุ่งยากในการเขียนโปรแกรมลงได้มาก ทั้งยังทำให้เกิดการใช้งานคอมพิวเตอร์ เพื่อการประมวลผลเพิ่มขึ้น เช่น การควบคุมและสั่งงานเครื่องคอมพิวเตอร์เมนเฟรม การแก้ปัญหาเฉพาะด้านทางด้านอุตสาหกรรม เช่น การควบคุมเครื่องจักรกลต่าง ๆ เป็นต้น

การเขียนโปรแกรมด้วยภาษาระดับสูงจะต้องใช้ ตัวแปลภาษา ที่เรียกว่า คอมไพเลอร์ (Compiler) เพื่อแปลภาษาระดับสูงโดยการตรวจสอบไวยากรณ์ของภาษาระดับสูง ไปเป็นภาษาเครื่องเพื่อสั่งให้เครื่องคอมพิวเตอร์ทำงานต่อไป โดยคอมไพเลอร์ของภาษาระดับสูงแต่ละภาษาจะแปลเฉพาะภาษาของตนเอง และทำงานได้เฉพาะเครื่องคอมพิวเตอร์ชนิดเดียวกัน เท่านั้น เช่น คอมไพเลอร์ของภาษา COBOL บนเครื่องไมโครคอมพิวเตอร์ จะแปลภาษาเฉพาะคำสั่งของภาษา COBOL และจะทำงานได้บนเครื่องคอมพิวเตอร์ที่เหมือนกันเท่านั้น ถ้าต้องการนำไปใช้กับเครื่องคอมพิวเตอร์แบบอื่นๆ เช่น เมนเฟรม จะต้องใช้คอมไพเลอร์ของภาษา COBOL แบบใหม่ ตัวอย่างของภาษาคอมพิวเตอร์ระดับสูงได้แก่ **ภาษา BASIC ภาษา COBOL ภาษา FORTRAN และ ภาษา C** ที่ได้รับความนิยมมากเช่นกัน สามารถเขียนโปรแกรมแก้ปัญหาเฉพาะด้าน เช่น การควบคุมหุ่นยนต์ การสร้างภาพกราฟิก ได้เป็นอย่างดีเพราะมีความยืดหยุ่นและเหมาะกับการใช้งานทั่ว ๆ ไปได้

สรุปภาษาโปรแกรมคอมพิวเตอร์ในยุคที่ 3 มีการเขียนโปรแกรมที่ง่ายกว่าในยุคที่ 2 สามารถทำงานได้บนเครื่องคอมพิวเตอร์หลายระดับ (Machine Independent) โดยต้องใช้ควบคู่กับตัวแปลภาษา (Compiler or Interpreter) สำหรับเครื่องนั้น ๆ และมีความยืดหยุ่นในการแก้ปัญหาได้มากกว่าภาษาระดับต่ำ

ยุคที่ 4 ภาษาระดับสูงมาก (Very high-level Language)

ภาษาระดับสูงมาก เป็นภาษาโปรแกรมคอมพิวเตอร์ยุคที่สี่ (Fourth-generation language) ซึ่งเป็นภาษาที่ใช้ ในการเขียนโปรแกรมด้วยคำสั่งสั้น ๆ และง่ายกว่าภาษาในยุคก่อน ๆ มีการทำงานแบบไม่จำเป็นต้องบอกลำดับของขั้นตอนการทำงาน (Nonprocedural language) เพียงนักเขียนโปรแกรมกำหนดว่าต้องการให้โปรแกรมทำอะไรเท่านั้น โดยไม่ต้องทราบว่าทำได้อย่างไร ทำให้เขียนโปรแกรมได้ง่าย และรวดเร็ว กว่าภาษาระดับสูงในยุคที่ 3 ที่มีการเขียนโปรแกรมแบบบอกขั้นตอนการทำงาน (Procedural language) ภาษาระดับสูงมากทำงานเหมือนกับภาษาพูดว่าต้องการอะไร และเขียนเหมือนภาษาอังกฤษ ดังตัวอย่าง เช่น

```
TABLE FILE SALES
SUM UNITS BY MONTH BY CUSTOMER BY PRODUCT
ON CUSTOMMER SUBTOTAL PAGE BREAK
END
```

ข้อดีของภาษาคอมพิวเตอร์ในยุคที่ 4

- การเขียนโปรแกรมจะสั้นและง่าย เพราะเน้นที่ผลลัพธ์ของงานว่าต้องการอะไร โดยไม่สนใจว่าจะทำได้อย่างไร
- การเขียนคำสั่ง ทำได้ง่ายและแก้ไข เปลี่ยนแปลงโปรแกรมได้สะดวก ทำให้พัฒนาโปรแกรมได้รวดเร็วขึ้น
- ผู้เขียนโปรแกรมสามารถเขียนโปรแกรมได้เร็ว โดยไม่ต้องเสียเวลาอบรม หรือมีความรู้ด้านการเขียนโปรแกรม เพราะชุดคำสั่งเหมือนภาษาพูด
- ผู้เขียนโปรแกรมไม่จำเป็นต้องทราบถึงฮาร์ดแวร์ ของเครื่องและโครงสร้างคำสั่งของภาษาโปรแกรม

ตัวอย่างภาษาคอมพิวเตอร์ในยุคที่ 4

ประกอบด้วย Report Generators, Query Language, Application Generators และ Interactive Database Management System Programs

ภาษาที่ใช้สำหรับเรียกดูข้อมูลจากฐานข้อมูลได้ เรียกว่า ภาษาสอบถาม (Query languages) จัดเป็นภาษาในยุคที่ 4 ซึ่งสามารถใช้ค้นคืนสารสนเทศของฐานข้อมูล มาตรฐานของภาษาชนิดนี้ขึ้นอยู่กับฐานข้อมูลที่แตกต่างกัน ที่นิยมใช้กันมากที่สุดคือ SQL (Structured Query Language) และนอกจาก นี้ยังมีภาษา Query by Example หรือ QBE ที่ได้รับความนิยมในการใช้งาน

Report Generator หรือ Report Writer คือโปรแกรมสำหรับผู้ใช้ (End user) ที่ใช้สำหรับสร้างรายงาน อาจแสดงที่เครื่องพิมพ์หรือจอภาพก็ได้ อาจจะแสดงทั้งหมดหรือบางส่วนของฐานข้อมูลก็ได้ ท่านอาจจะกำหนดรูปแบบบรรทัด คอลัมน์ ส่วนหัวรายงาน และอื่น ๆ ได้

Application Generators คือเครื่องมือของผู้เขียนโปรแกรมที่ใช้ในการสร้างโปรแกรมประยุกต์ จากการอภิปรายปัญหาได้เร็วกว่าการเขียนโปรแกรมทั่ว ๆ ไป

ยุคที่ 5 ภาษาธรรมชาติ (Natural Language)

ภาษาธรรมชาติ เป็นภาษาโปรแกรมคอมพิวเตอร์ยุคที่ห้า (Fifth generation language) การเขียนคำสั่งหรือสั่งงานคอมพิวเตอร์ทำได้โดยใช้ภาษาธรรมชาติต่าง ๆ เช่น ภาพ หรือ เสียง โดยไม่สนใจรูปแบบไวยากรณ์ หรือโครงสร้างของภาษามากนัก ซึ่งคอมพิวเตอร์จะพยายามคิดวิเคราะห์ และแปลความหมายโดยอาศัยการเรียนรู้ด้วยตนเองและระบบองค์ความรู้ (Knowledge Base System) มาช่วยแปลความหมายของคำสั่งต่าง ๆ และตอบสนองต่อผู้ใช้งาน

ตัวอย่างภาษาคอมพิวเตอร์ในยุคที่ 5 เช่น

SUM SHIPMENTS BY STATE BY DATE

ข้อดีของภาษาคอมพิวเตอร์ในยุคที่ 5 คือผู้เขียนโปรแกรมสามารถเขียนโปรแกรมได้เร็ว โดยไม่ต้องมีความรู้ด้านการเขียนโปรแกรม แต่คอมพิวเตอร์ที่ใช้โปรแกรมต้องมีระบบรับคำสั่ง และประมวลผลแบบอัจฉริยะ สามารถตอบสนองและทำงานได้หลายแบบ

ภาษาคอมพิวเตอร์กับการใช้

ภาษาคอมพิวเตอร์	การใช้งาน
BASIC (Beginner's All-purpose Symbolic Instruction Code)	สำหรับผู้เริ่มศึกษาการเขียนโปรแกรมภาษาคอมพิวเตอร์
COBOL (Common Business Oriented Language)	นิยมใช้ในงานธุรกิจบนเครื่องขนาดใหญ่
FORTRAN (FORmula TRANslator)	ใช้สำหรับงานด้านคณิตศาสตร์วิทยาศาสตร์วิศวกรรมศาสตร์
Pascal (ชื่อของ Blaise Pascal)	ใช้ในวิทยาลัย และมหาวิทยาลัย
C	สำหรับนักเขียนโปรแกรม และใช้ในวิทยาลัยมหาวิทยาลัย
C++	สำหรับผู้ผลิตซอฟต์แวร์
ALGOL (ALGOrithmic Language)	เริ่มต้นได้รับการออกแบบให้เป็นภาษาสำหรับงานทางวิทยาศาสตร์และต่อมามีการพัฒนาต่อเป็นภาษา PL/I และ Pascal
APL (A Programming Language)	ออกแบบโดยบริษัท IBM ในปีค.ศ. 1968 เป็นภาษาที่โต้ตอบกับผู้ใช้ทันที เหมาะสำหรับจัดการกับกลุ่มของข้อมูลที่สัมพันธ์กันในรูปแบบตาราง
LISP (LIST Processing)	ถูกออกแบบมาให้ใช้กับข้อมูลที่ไม่ใช้ตัวเลข ซึ่งอาจเป็นสัญลักษณ์พิเศษหรือตัวอักษรก็ได้ นิยมใช้ในด้านปัญญาประดิษฐ์ (Artifical Inelligence)
LOGO	นิยมใช้ในโรงเรียน เพื่อสอนทักษะการแก้ปัญหาให้กับนักเรียน

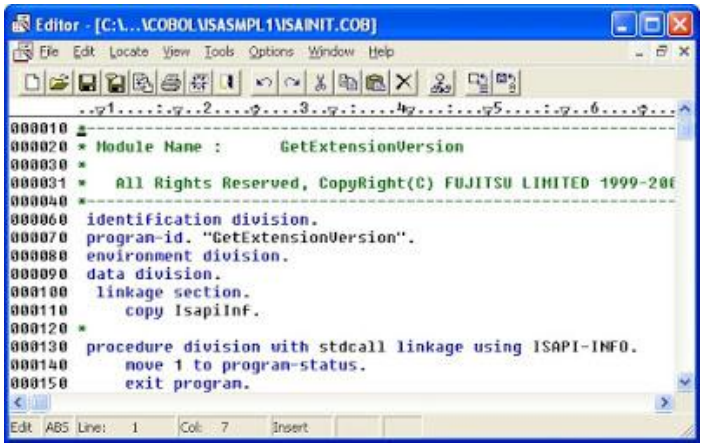
ตัวอย่างโครงสร้างภาษาคอมพิวเตอร์

ภาษาฟอร์เทรน (FORTRAN - Formula Translation) เป็นภาษาที่ออกแบบเพื่อใช้งานทาง วิทยาศาสตร์ วิศวกรรมศาสตร์ และด้านคณิตศาสตร์ ภาษาฟอร์เทรน จะประกอบด้วยข้อความ คำสั่ง ทีละ บรรทัด ซึ่งต้องอาศัยการคำนวณเป็นอย่างมาก ตัวอย่างการเขียนคำสั่งฟอร์เทรน สำหรับการคำนวณพื้นที่ ของรูปสี่เหลี่ยมผืนผ้า ดังนี้

```
C This program calculates the area of a rectangle.
PEAL WIDTH, HEIGHI, AREA
WIDTH = 10.0
HEIGHI = 234.1
AREA = WIDTH * HEIGHI
WRITE (*,*) ' AREA = ' , AREA
STOP

END
```

ภาษาโคบอล (COBOL: Common Business Oriented Language) เป็นภาษาสำหรับใช้ในงาน ธุรกิจภาษาแรกของโลกโดยคำสั่งของภาษา COBOL จะคล้ายกับภาษาอังกฤษทำให้สามารถอ่านและเขียน โปรแกรมได้ไม่ยากนัก ตัวอย่างการเขียนภาษาโคบอล ในการสั่งพิมพ์เลข 1 ถึง 10 โดยกำหนดค่าเริ่มต้น เป็น 0

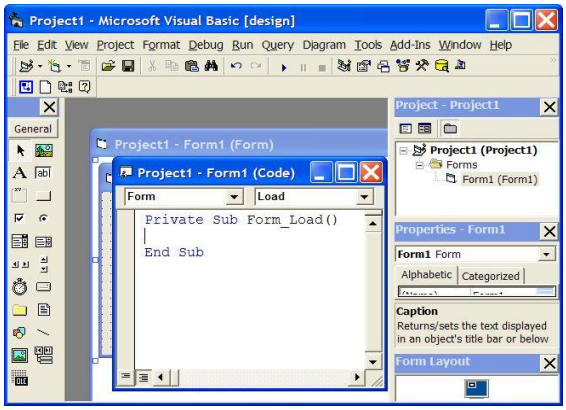


ตัวอย่างการเขียนโปรแกรมด้วยภาษา COBOL

```
IF SALES-AMOUNT IS GREATER THAN SALES-QUOTA
COMPUTE COMMISSION = MAX-RATE * SALES - AMOUNT
ELSE
COMPUTE COMMISSION = MIN-RATE * SALES - AMOUNT
```

ภาพจาก : http://jean-jeerattikul.blogspot.com/2013/08/cobol.html

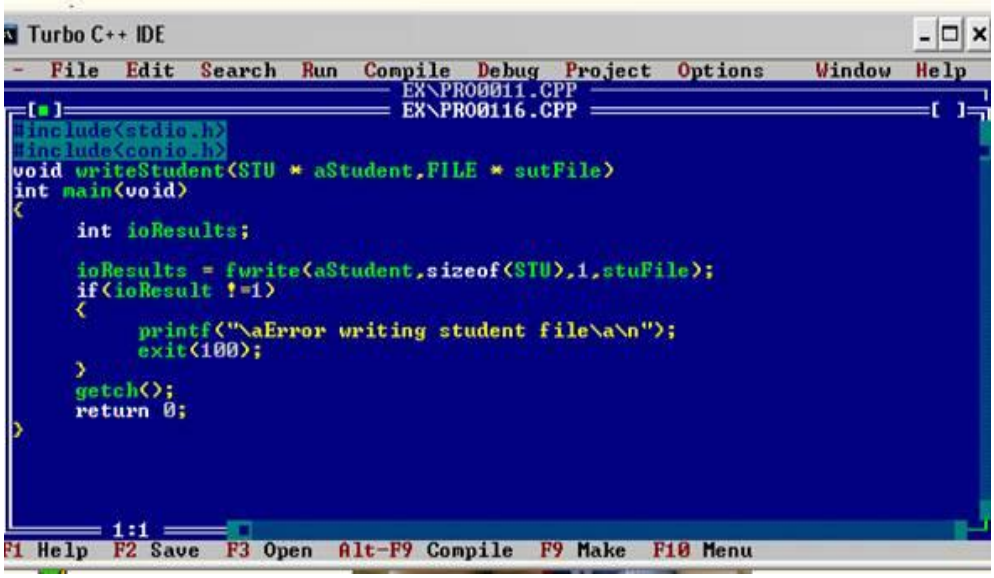
ภาษาเบสิก (BASIC) ภาษาโปรแกรมสำหรับผู้เริ่มต้น เป็นภาษาโปรแกรมที่เรียนรู้ง่าย ไม่ซับซ้อน เหมาะสำหรับใช้ในห้องเรียนสำหรับการเรียนการสอน มีคำสั่งที่ง่ายสามารถเข้าใจได้เร็ว




```
GraphicsWindow.Height = 200
GraphicsWindow.Width = 200
For i = 1 To 100
   GraphicsWindow.BrushColor = GraphicsWindow.GetRandomColor()
   Ex = Math.GetRandomNumber(200)
   Ey = Math.GetRandomNumber(200)
   GraphicsWindow.FillEllipse(Ex,Ey,20,20)
Endfor
```

ภาพจาก : https://www.google.co.th/search?

ภาษาซี (C) ภาษาสมัยใหม่ เป็นภาษาที่ใช้สำหรับเขียนโปรแกรมระบบปฏิบัติการ เหมาะสำหรับโปรแกรมเมอร์ที่มีความสามารถสูง



ภาพจาก : http://itd.htc.ac.th/st_it50/it5016/nidz/Web_C/unit12.html

ภาษาจาวา (Java) เป็นภาษาที่พัฒนาขึ้นล่าสุด แต่ได้รับความนิยมในการนำไปใช้ค่อนข้างมากเนื่องจากสามารถประมวลผลกับระบบคอมพิวเตอร์ได้ทุกประเภท จาวายังสามารถนำไปใช้เป็นภาษาสำหรับ อุปกรณ์แบบฝังต่าง ๆ เช่น โทรศัพท์และอุปกรณ์ขนาดมือถือแบบต่าง ๆ เป็นต้น

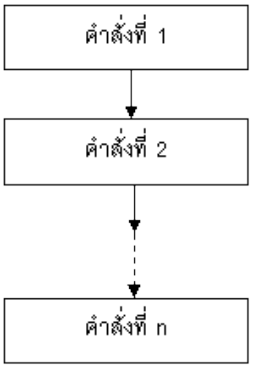
ตัวอย่างการเขียนภาษาจาวา การคิดเกรด

```
if (score < 50)
        messageBox.show("Your grade is F");
else if (score < 60)
        messageBox.show("Your grade is D");
else if (score < 70)
        messageBox.show("Your grade is C");
else if (score < 80)
        messageBox.show("Your grade is B");
else
       messageBox.show ("Your grade is A");
```

โครงสร้างการเขียนโปรแกรม (Structure Programming)

รูปแบบโครงสร้างภายในโปรแกรมที่ควรรู้จัก มีดังนี้

1. โครงสร้างแบบเรียงลำดับ



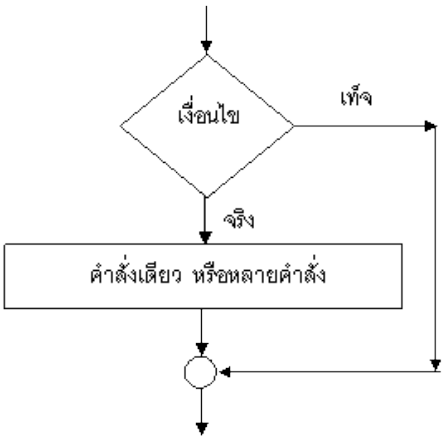
การทำงานของโครงสร้างแบบเรียงลำดับคือจะทำงานตามคำสั่งที่1,คำสั่งที่2, ⋯, คำสั่งที่nตามลำดับ

แสดงโครงสร้างแบบเรียงลำดับ

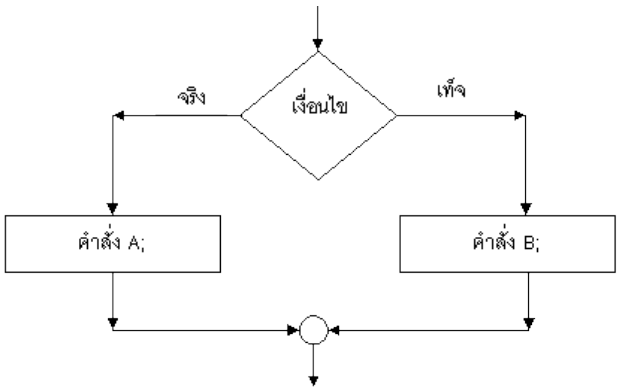
ที่มา : สมชายรัตนเลิศนุสรณ์, 2545 : 29.

2. โครงสร้างแบบทดสอบเงื่อนไข

โครงสร้างแบบทดสอบเงื่อนไข1ทาง

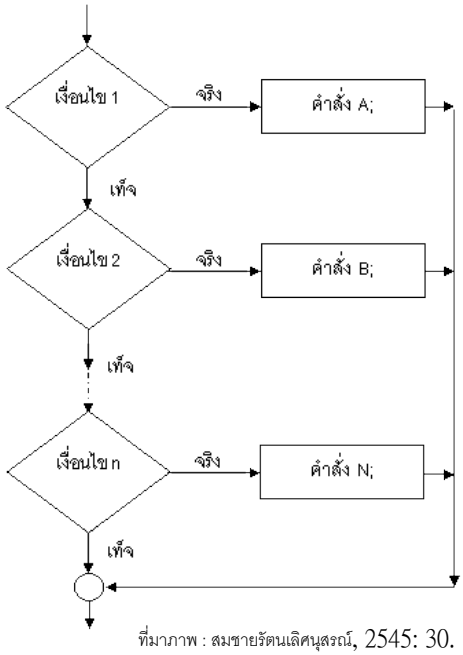


โครงสร้างแบบทดสอบเงื่อนไข2ทาง



ที่มาภาพ : สมชายรัตนเลิศนุสรณ์, 2545: 30.

โครงสร้างแบบทดสอบเงื่อนไขมากกว่า 2 ทางขึ้นไป



ที่มาภาพ : สมชายรัตนเลิศนุสรณ์, 2545: 30.

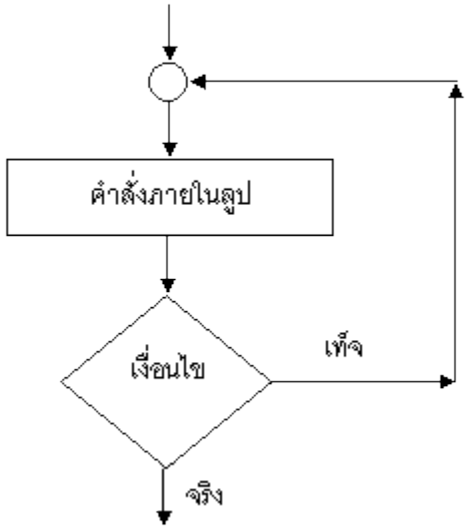
การทำงานของโครงสร้างแบบทดสอบเงื่อนไขมากกว่า 2 ทางขึ้นไป มีลักษณะดังต่อไปนี้

1. ทำการทดสอบเงื่อนไขที่ 1 ว่าเป็นจริงหรือเท็จ ถ้าเงื่อนไขเป็นจริง ให้ทำงานตามคำสั่ง A ที่กำหนดไว้แล้วออกจากการทำงาน แต่ถ้าเงื่อนไขเป็นเท็จ ให้ทำงานในข้อ 2.
2. ทำการทดสอบเงื่อนไขที่ 2 ว่าเป็นจริงหรือเท็จ ถ้าเงื่อนไขเป็นจริง ให้ทำงานตามคำสั่ง B ที่กำหนดไว้แล้วออกจากการทำงาน แต่ถ้าเงื่อนไขเป็นเท็จ ให้ทำงานในข้อ 3.
3. ทำการทดสอบเงื่อนไขใหม่อีก โดยทำซ้ำเช่นนี้ไปเรื่อย ๆ จนกว่าจะถึงเงื่อนไขสุดท้าย (เงื่อนไขที่ n) ถ้าเงื่อนไขเป็นจริง ให้ทำงานตามคำสั่ง N ที่กำหนดไว้แล้วออกจากการทำงาน แต่ถ้าเงื่อนไขเป็นเท็จ ให้ออกจากการทำงาน

3. โครงสร้างแบบทำงานวนลูป

โครงสร้างแบบทำงานวนลูป หมายถึง โครงสร้างของคำสั่งที่มีการทำงานซ้ำ ๆ เป็นวงจรปิด จนกว่าเงื่อนไขที่ทดสอบจะตรงกับค่าจริงหรือเท็จตามโครงสร้างที่ใช้ จึงสามารถออกจากการทำงานได้ ซึ่งสามารถแบ่งออกเป็น 2 ชนิด ดังนี้

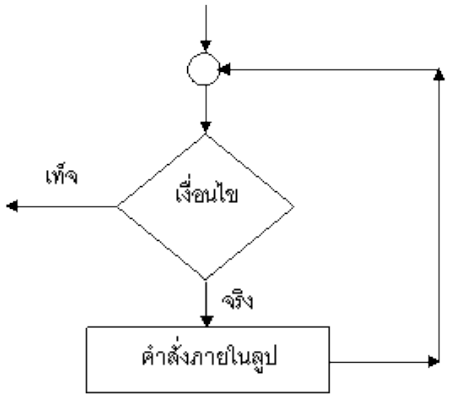
1) โครงสร้างแบบ DO UNTIL



การทำงานของโครงสร้างแบบ DO UNTIL คือเริ่มต้นด้วยการทำงานตามคำสั่งที่ต้องการภายในลูปจนหมดคำสั่งไป 1 รอบ จากนั้นจึงทำการทดสอบเงื่อนไขว่าเป็นจริงหรือเท็จ ถ้าเงื่อนไขเป็นเท็จ ให้กลับไปทำงานตามคำสั่งภายในลูปอีก ทำงานซ้ำ ๆ เช่นนี้จนกว่าเงื่อนไขเป็นจริง จึงออกจากการทำงานแบบนี้ได้ ลักษณะของโครงสร้างแบบ DO UNTIL นี้จะตรงกับคำสั่ง REPEAT UNTIL ในภาษาปาสคาล

ที่มา : สมชายรัตนเลิศนุสรณ์, 2545 : 31.

2) โครงสร้างแบบ DO WHILE



การทำงานของโครงสร้างแบบ DO WHILE คือเริ่มต้นด้วยการทดสอบเงื่อนไขว่าเป็นจริงหรือเท็จ ถ้าเงื่อนเป็นจริง ให้ทำงานตามคำสั่งที่ต้องการภายในลูปจนหมดคำสั่ง จากนั้นจึงย้อนกลับไปทำการทดสอบเงื่อนไขอีกครั้งว่าเป็นจริงหรือเท็จ ถ้าเงื่อนไขเป็นจริง ให้ทำงานตามคำสั่งภายในลูปอีก ทำงานซ้ำ ๆ เช่นนี้จนกว่าเงื่อนไขเป็นเท็จ จึงออกจากการทำงานแบบนี้ได้ ลักษณะของโครงสร้าง DO WHILE นี้จำตรงกับคำสั่ง while ในภาษา C

ที่มา : สมชาย รัตนเลิศนุสรณ์, 2545 : 32.