การเขียนโปรแกรมคอมพิวเตอร์

การวิเคราะห์ปัญหาและแนวทางที่ใช้ในการไขปัญหา

อาจารย์ โอภาส วงษ์ทวีทรัพย์ (อ.โอ๊ต)
ศูนย์ปฏิบัติการวิจัย และพัฒนาระบบสารสนเทศอัจฉริยะ
ภาควิชาคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากร
E-Mail :: oatcomster@gmail.com

ขอบเขตของเนื้อหา

- ทบทวนนิยามและความหมายของ “คอมพิวเตอร์”
- ทบทวนการทำงาน และพฤติกรรมของโปรแกรมคอมพิวเตอร์
- ขั้นตอนในการพัฒนาโปรแกรมคอมพิวเตอร์ ตั้งแต่การวิเคราะห์ปัญหา – การบำรุงรักษาโปรแกรม
- วิธีการในการแก้ไขปัญหาทางคอมพิวเตอร์
- ประสิทธิภาพของโปรแกรม

นิยามของคอมพิวเตอร์

คอมพิวเตอร์เป็นอุปกรณ์อิเล็กทรอนิกส์อย่างหนึ่ง
ที่สามารถรับโปรแกรมและข้อมูลในรูปแบบที่เครื่อง
สามารถจะรับได้ แล้วทำการคำนวณ เคลื่อนย้ายข้อมูล
ทำการเปรียบเทียบจนกระทั่งได้ผลลัพธ์ตามที่ต้องการ

พฤติกรรมของโปรแกรมคอมพิวเตอร์

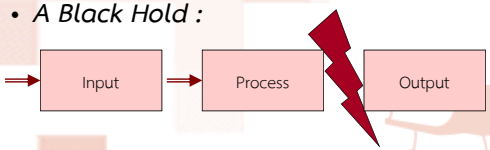
- *A Normal Program :*

is when a process has inputs

and produces complete outputs.

พฤติกรรมของโปรแกรมคอมพิวเตอร์

- *A Black Hold :*

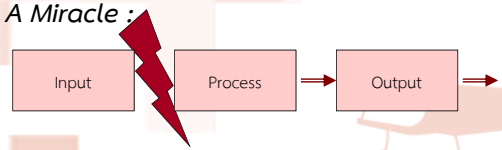


is when a process has inputs

but no outputs.

พฤติกรรมของโปรแกรมคอมพิวเตอร์

- *A Miracle :*



is when a process has outputs

but no inputs.

พฤติกรรมของโปรแกรมคอมพิวเตอร์

- *A Gray Hold :*

is when the inputs of a process

are insufficient to produce the outputs.

ขั้นตอนการพัฒนาโปรแกรมคอมพิวเตอร์

- การวิเคราะห์ปัญหา ***(Analysis the Problem)***
- ออกแบบโปรแกรม ***(Design a Program)***
- การเขียนโปรแกรมด้วยคอมพิวเตอร์ ***(Coding Program)***
- การตรวจสอบข้อผิดพลาดของโปรแกรม ***(Testing and Debugging)***

ขั้นตอนการพัฒนาโปรแกรมคอมพิวเตอร์

- การทดสอบความถูกต้องของโปรแกรม ***(Testing and Validating)***
- การทำเอกสารประกอบโปรแกรม ***(Documentation)***
- การบำรุงรักษาโปรแกรม ***(Program Maintenance)***

วิธีการวิเคราะห์ปัญหา

- จะมีรูปแบบในการวิเคราะห์ปัญหาอยู่ 2 แนวทาง คือ
  - *Analytical*
  - *Algorithmic*

วิธีทาง Analytical

- เป็นวิธีที่นักคณิตศาสตร์และนักฟิสิกส์นิยมใช้กัน
- โดยมีหลักการในการแก้ปัญหา โดย
  - แยกแยะค่าที่ให้มา
  - พิจารณาว่าต้องการหาอะไรหรือต้องการแก้ไขปัญหาอะไร
  - หาสูตรต่างๆ เพื่อนำมาประยุกต์ใช้
  - ทำการคำนวณตามสูตรดังกล่าว
  - ได้ผลลัพธ์ตามต้องการ

ตัวอย่างวิธี Analytical

- สามเหลี่ยมมุมฉาก มีเส้นประกอบมุมฉาก 2 เส้น ที่มีความยาวเท่ากับ 3 และ 4 หน่วย ให้หาความยาวของเส้นตรง ที่อยู่ตรงข้ามมุมฉาก
- ใช้สูตร $c^2 = a^2 + b^2$
- แทนค่าในสูตรด้วยค่าของ a ด้วย 3 และ b ด้วย 4
- ทำให้ได้คำตอบ คือ $c = 5$

อัลกอริทึม (Algorithm)

- อัลกอริทึมเป็นขั้นตอนวิธีการที่จะถูกสร้างขึ้นมาจากกลุ่มของกฎเกณฑ์ที่มีอยู่
- กฎเกณฑ์ที่สร้างอัลกอริทึมนั้น ต้องชัดเจน ไม่คลุมเครือ
- การประมวลผลของ ***operations*** ที่กำหนดโดยกฎเกณฑ์นั้นต้องเป็นลำดับที่แน่นอน
- กระบวนวิธีการต้องให้เป็นผลลัพธ์ตามที่กำหนดในปัญหา และมีคุณสมบัติ ***generality***
- อัลกอริทึมต้องอยู่ในรูปของขั้นตอนวิธีการที่มีการสิ้นสุด

อัลกอริทึม (Algorithm)

- คือ รายการของคำสั่งที่ระบุคำบรรยายอย่างชัดเจน ว่าจะต้องดำเนินการอย่างไร ในลำดับก่อนหลัง
- เป็นขั้นตอนของกระบวนการที่รับประกันได้ว่าจะสิ้นสุดหลังจากทำงานตามจำนวนขั้นตอนที่จำกัด
- โดยมีผลลัพธ์ที่ถูกต้องสำหรับทุกกรณีที่อาจเกิดขึ้นของปัญหาเชิงอัลกอริทึมอันหนึ่ง

อัลกอริทึม (Algorithm)

- แก้ปัญหาด้วยลำดับของขั้นตอนหรือลำดับของคำสั่ง
- คล้ายกับวิธีการประมวลผลของ Computer ที่ต้องทำงานเรียงลำดับของคำสั่งว่า คำสั่งใดทำก่อน-ทำหลัง
- ยกตัวอย่าง เช่น ขั้นตอนในการทำเค้ก
- Input : ส่วนผสม, Processing :ขั้นตอนการทำ, Output : เค้ก

ลักษณะพื้นฐานของอัลกอริทึม

- ถูกต้องแน่นอน (“...ระบุคำบรรยายที่แม่นยำ”)
- ได้ผล (“...รับประกันว่าจะให้คำตอบที่ถูกต้อง”)
- รับประกันการหยุด (“...หยุดหลังจากจำนวนขั้นตอนที่จำกัด”)
- ใช้การได้ทั่วไป (“...สำหรับทุกกรณี”)

วิธีการสร้างอัลกอริทึม

- มีรูปแบบขั้นตอนในการทำ 3 รูปแบบ คือ
- การบรรยาย ***(Narrative description)***
- ผังงาน ***(Flowchart)***
- รหัสจำลอง ***(Pseudocode)***

วิธีการบรรยาย (Narrative Description)

- ***ตัวอย่าง*** ให้เขียนอัลกอริธึมเพื่อหาคะแนนเฉลี่ยของการสอบย่อย 4 ครั้ง แล้วให้เกรดวัดผล โดยถ้าคะแนนต่ำกว่า 50 สอบไม่ผ่าน แต่ถ้าคะแนนตั้งแต่ 50 ขึ้นไป สอบผ่าน
- Algorithm : Read a set of four marks. Compute their average by summing them and divided by 4. If this average is below 50, display the grade with a failing message, otherwise display the grade with a passing message.

ผังงาน (Flow Chart)

- เป็นวิธีการที่ใช้แสดงลำดับของขั้นตอนการทำงานต่างๆ
- โดยใช้รูปของภาพสัญลักษณ์ต่างๆ ในการแทนความหมายของการทำงานหนึ่งๆ
- ช่วยให้เข้าใจ และมองเห็นขั้นตอนของการทำงานเหล่านั้นได้อย่างชัดเจน

ประเภทของผังงาน

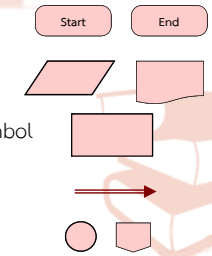
- **ผังงานนั้น สามารถแบ่งออกได้เป็น 2 ประเภท**
- *System Flowchart*
  - แสดงลำดับขั้นตอนการทำงานอย่างคร่าวๆทั้งระบบ
- *Program Flowchart*
  - แสดงลำดับขั้นตอนการทำงานอย่างละเอียด แสดงให้เห็นถึงคำสั่งแต่ละคำสั่งในขั้นตอนการประมวลผล

โครงสร้างควบคุมการทำงาน

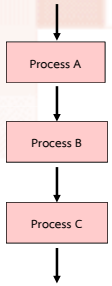
- โครงสร้างแบบเรียงลำดับ *(Sequence Structure)*
- โครงสร้างแบบมีทางเลือก *(Selection Structure)*
- โครงสร้างแบบทำซ้ำ *(Iteration Structure)*

อธิบายสัญลักษณ์ที่ใช้ในผังงาน

- Start/Stop (End) Symbol
- Read/Write (I/O) Symbol
- The Operation(Process) Symbol
- Lines and Arrow
- Connector, New page

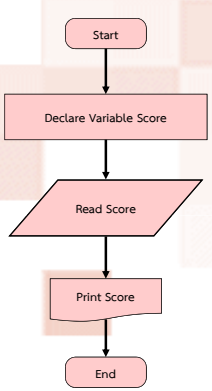


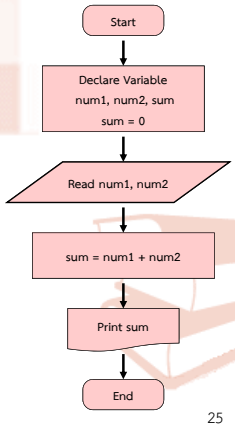
โครงสร้างแบบเรียงลำดับ



ตัวอย่างโจทย์

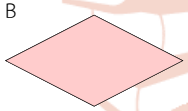
- เขียนผังงานแสดงขั้นตอนการอ่าน และพิมพ์คะแนนสอบของนักเรียน
- เขียนผังงานเพื่ออ่านเลขจำนวนเต็ม 2 จำนวนและคำนวณหาผลรวมของเลขสองจำนวนนั้น พร้อมทั้งพิมพ์ผลลัพธ์ที่ได้



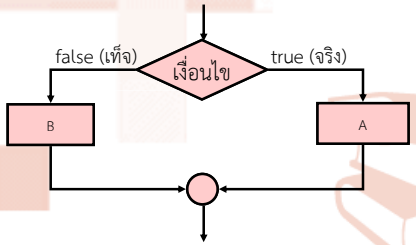


อธิบายโครงสร้างแบบมีทางเลือก

- Decision Symbol
- ภายในรูปเหลี่ยมนั้น จะมีข้อความแสดงถึงเงื่อนไขของการเปรียบเทียบ หรือการตัดสินใจ
- เช่น A> B , A >= B , A < B , A <= B
- A = B , A <> B etc.

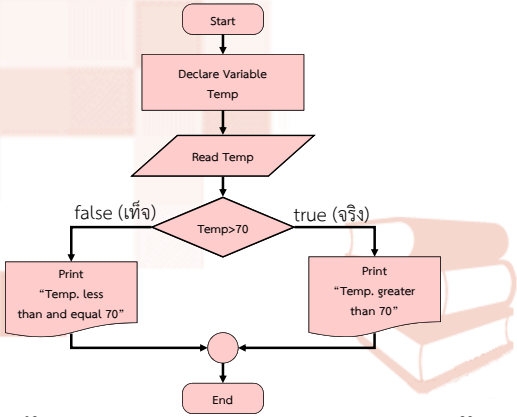


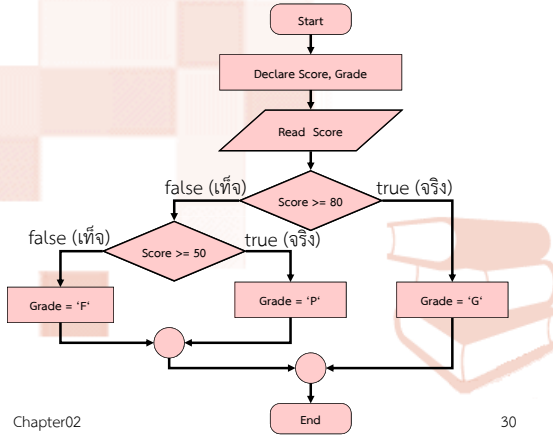
โครงสร้างแบบมีทางเลือก



ตัวอย่างโจทย์

- เขียนผังงานเพื่ออ่านอุณหภูมิประจำวัน ถ้าอุณหภูมิมากกว่า 70 พิมพ์ “Temp. greater than 70” นอกนั้นพิมพ์ “Temp. less than and equal 70”
- เขียนผังงานเพื่ออ่านคะแนนสอบของนักเรียน และทำการคิดเกรดตามเกณฑ์ที่กำหนด เกณฑ์ คะแนน >= 80 เกรด G คะแนน >= 50 เกรด P , คะแนน < 50 เกรด F

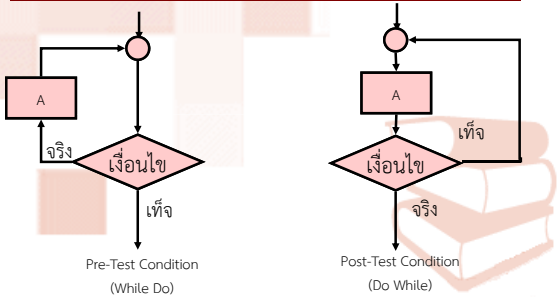




อธิบายโครงสร้างแบบทำซ้ำ

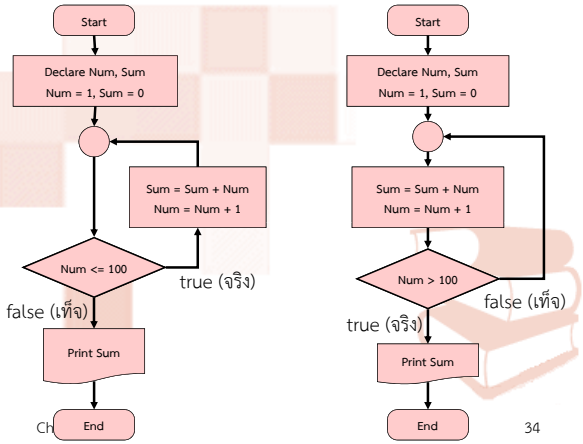
- **Fixed Count Condition คือ** กลุ่มคำสั่งทำซ้ำที่มีการทำซ้ำเท่ากับจำนวนครั้งที่ถูกกำหนดขึ้น ได้แก่ คำสั่ง for
- **Pre-Test Condition คือ** กลุ่มคำสั่งทำซ้ำที่จะทำการทดสอบเงื่อนไขก่อนจึงทำซ้ำ ได้แก่ คำสั่ง while-do
- **Post-Test Condition คือ** กลุ่มคำสั่งทำซ้ำที่จะทำงานก่อนจึงจะทำการทดสอบเงื่อนไข ได้แก่ คำสั่ง do-while

โครงสร้างแบบทำซ้ำ



ตัวอย่างโจทย์

- เขียนผังงานแสดงขั้นตอนการหาค่าผลบวกสะสม 1+2+3+...+98+99+100 และแสดงผลลัพธ์ที่ได้
- เขียนผังงานและแสดงขั้นตอนการอ่านชื่อ คะแนนสอบกลางภาค คะแนนสอบปลายภาค ของนักเรียน 100 คน และพิมพ์ผลลัพธ์



โค้ดเทียม (Pseudocode)

- ข้อแตกต่างที่เห็นได้ชัดเจนของผังงานและรหัสจำลองคือ รหัสจำลองไม่มีรูปภาพ แต่ใช้ภาษาอังกฤษเป็นคำอธิบาย โดยที่คำอธิบายนั้นจะใกล้เคียงกับภาษาโปรแกรมชั้นสูงภาษาใดภาษาหนึ่ง

การวิเคราะห์ออกแบบโปรแกรม

- **การระบุข้อมูลเข้า** ***(Input Specification)***
  - ข้อมูลที่นำเข้าสู่ระบบคอมฯ ประกอบด้วยอะไรบ้าง
- **การระบุข้อมูลออก** ***(Output Specification)***
  - กำหนดวัตถุประสงค์ของงาน
  - รูปแบบของผลลัพธ์

การวิเคราะห์ออกแบบโปรแกรม (ต่อ)

- **การกำหนดวิธีการประมวลผล *(Process Specification)***
  - ต้องทราบขั้นตอนวิธีการประมวลผล

การวิเคราะห์ออกแบบโปรแกรม (ต่อ)

- **<u>โจทย์</u> ต้องการหาค่าผลบวกของตัวเลข 2 ตัว**
- **การระบุข้อมูลเข้า *(Input Specification)***
  - *รับค่าของตัวเลขทั้ง 2 ตัว*
- **การระบุข้อมูลออก *(Output Specification)***
  - *แสดงผลบวกของเลข 2 ตัวที่รับเข้าไป*
  - *XXX + XXX = XXX*

การวิเคราะห์ออกแบบโปรแกรม (ต่อ)

- **การกำหนดวิธีการประมวลผล *(Process Specification)***
  - *รับค่าตัวเลขตัวที่ 1 (number1)*
  - *รับค่าตัวเลขตัวที่ 2 (number2)*
  - *ทำการคำนวณหาผลบวกของเลขตัวที่ 1 กับ 2 โดยใช้*
  - *ผลรวม = number1 + number2*

การวิเคราะห์ออกแบบโปรแกรม (ต่อ)

- **<u>โจทย์</u> ต้องการคำนวณค่าของพื้นที่ของรูปสามเหลี่ยม**
- **การระบุข้อมูลเข้า *(Input Specification)***
  - *รับค่าของส่วนสูงและความยาวของฐาน*
- **การระบุข้อมูลออก *(Output Specification)***
  - *แสดงผลของพื้นที่ของรูปสามเหลี่ยมที่คำนวณได้*
  - *Area of Triangle = XXX*

การวิเคราะห์ออกแบบโปรแกรม (ต่อ)

- **การกำหนดวิธีการประมวลผล *(Process Specification)***
  - *รับค่าของความสูง (height)*
  - *รับค่าของความยาวของฐาน (base)*
  - *ทำการคำนวณหาพื้นที่จากสูตร*
  - *Area = 1/2 x height x base*

การเขียนโปรแกรมด้วยคอมพิวเตอร์

- หลังจากขั้นตอนของการวิเคราะห์ปัญหา และการออกแบบโปรแกรมแล้ว
- เราก็ต้องตัดสินใจเลือกภาษาที่ใช้ในการเขียนโปรแกรมมา 1 ภาษา เพื่อเขียนโปรแกรมแก้ปัญหา ตามที่ได้วิเคราะห์ไว้
- จากนั้น ก็ต้องนำโค้ดโปรแกรมที่เขียนขึ้น มาตรวจสอบข้อผิดพลาดและความถูกต้องของโปรแกรม และจัดทำเอกสารประกอบโปรแกรมต่อไป

การเขียนโปรแกรมด้วยคอมพิวเตอร์ (ต่อ)

- วิธีการเลือกภาษาโปรแกรมมาใช้นั้น โปรแกรมเมอร์จะตอบปัญหานี้ได้ดีขึ้น เมื่อฝึกแก้ปัญหาและเขียนโปรแกรมมากขึ้น
- และการแก้ปัญหาเหล่านั้น ก็ได้ลองเขียนโปรแกรมไว้ในหลากหลายภาษา เพื่อเปรียบเทียบถึงข้อดี - ข้อเสีย และประสิทธิภาพที่ได้รับจากคำตอบของการแก้ไขปัญหา
- ซึ่งอาจเทียบในแง่ความเร็ว หรือขนาดหน่วยความจำที่ใช้

การตรวจสอบข้อผิดพลาดของโปรแกรม

- เป็นขั้นตอนในการตรวจสอบความถูกต้องของไวยากรณ์ ***(Syntax)*** ของภาษาโปรแกรมที่ได้เขียนขึ้น ว่าถูกต้องทั้งหมดและสามารถทำงานได้หรือไม่ โดยการแก้ไขให้ถูกต้องนั้น เราเรียกว่า ***การแก้บัก (Debugging)***
- ที่มาของศัพท์นี้ มาจากคำว่า ***Bug*** ที่แปลว่า ***แมลง*** เนื่องจากมีแมลงเข้าไปตายในเครื่องคอมพิวเตอร์ จึงทำให้ไม่สามารถทำงานได้ แต่เมื่อเอาแมลงออกกลับทำงานได้ตามปกติ

การตรวจสอบข้อผิดพลาดของโปรแกรม (ต่อ)

- ข้อผิดพลาดของโปรแกรมนั้นสามารถเกิดขึ้นได้ 3 รูปแบบ คือ
  - ข้อผิดพลาดที่เกิดจากไวยากรณ์ของภาษา ***(Syntax Error)***
  - ข้อผิดพลาดที่เกิดในช่วงรันโปรแกรม ***(Run time Error)***
  - ข้อผิดพลาดที่เกิดจากการเขียนผิดความหมาย ***(Logical Error or Semantic Error)***

ข้อผิดพลาดที่เกิดจากไวยากรณ์ของภาษา

- การตรวจสอบนี้ เป็นหน้าที่หลักของตัวแปรภาษา ที่โปรแกรมเมอร์ใช้ หรือที่เราเรียกกันว่า คอมไพล์เลอร์ ***(Compiler)***
- แต่ตัวแปรภาษาจะไม่สามารถตรวจสอบข้อผิดพลาดที่เกิดจากการเขียนผิดความหมาย ***(Semantic)*** ของโปรแกรมได้

ข้อผิดพลาดที่เกิดจากไวยากรณ์ของภาษา(ต่อ)

- **ตัวอย่างของการทดสอบที่ต้องแจ้งให้ทราบว่าผิดพลาด เช่น**
  - if (a >< b) printf("What is this");
  - int a = b = c = d = 0;
- **ตัวอย่างของการทดสอบที่ไม่สามารถตรวจสอบได้ เช่น**
  - int pi = 22/7;
  - for(I = 0; I < 5; I--) printf("Show Result");

ข้อผิดพลาดที่เกิดในช่วงรันโปรแกรม

- ข้อผิดพลาดชนิดนี้ เป็นความผิดพลาดที่เกิดขึ้นในช่วงของการทำงานของโปรแกรม
- เป็นความผิดพลาดที่โปรแกรมเมอร์ไม่ได้เขียนโปรแกรมเพื่อทำการตรวจสอบไว้ เช่น
- การหารจำนวนด้วยส่วนที่มีค่าเป็นศูนย์ การเขียนโปรแกรมที่ให้รับค่าเป็นจำนวนเต็ม แต่ป้อนตัวอักขระเข้าไป เป็นต้น

ข้อผิดพลาดที่เกิดในช่วงรันโปรแกรม (ต่อ)

- ซึ่งการแก้ไขในส่วนนี้นั้น โปรแกรมเมอร์จะต้องเป็นผู้เขียนโค้ดขึ้นมาเอง เพื่อป้องกันความผิดพลาดที่อาจจะเกิดขึ้นได้
- if(count!=0) avg = sum/count;
- ต้องรับข้อมูลเข้าเป็น ***String*** แล้วทำการตรวจสอบให้แน่ใจว่าเป็น ***String*** ของตัวเลข แล้วจึงจะใช้ฟังก์ชัน ***atoi()*** ในการแปลง ***String*** ไปเป็น ***Integer*** เป็นต้น

ข้อผิดพลาดที่เกิดจากการเขียนผิดความหมาย

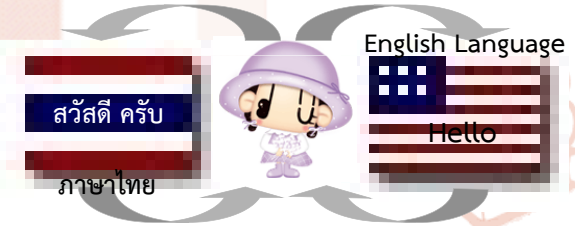
- การทดสอบความผิดพลาดในกรณีหลังนี้ จะไม่สามารถค้นหาเจอได้ ด้วยตัวแปรภาษา
- การตรวจสอบข้อผิดพลาดหรือการผิดความหมายแบบนี้ โปรแกรมเมอร์จะต้องเป็นผู้ค้นหาความผิดพลาดนี้เอง
- โดยใช้กระบวนการของการตรวจสอบความถูกต้องของโปรแกรม ซึ่งจะเป็นกระบวนการในขั้นตอนถัดไป

การตรวจสอบความถูกต้องของโปรแกรม

- โปรแกรมเมอร์จะต้องทำการตรวจสอบความถูกต้องของโปรแกรมจากชุดข้อมูลเข้า ที่คลอบคลุมกับทุกกรณีที่เป็นไปได้ว่าโปรแกรมสามารถ แก้ปัญหาหรือแสดงผลลัพธ์ได้ถูกต้องหรือไม่
- และโปรแกรมที่เขียนขึ้นนั้น จะต้องรองรับกับการแก้ไขปัญหาได้อย่างถูกต้อง ในทุกกรณี

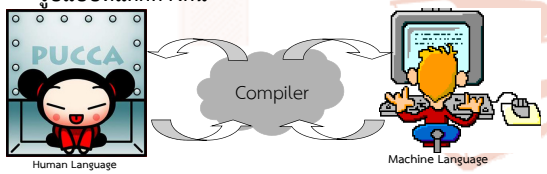
ตัวแปรภาษาคืออะไร???

- ทำไม??? ถึงต้องมี Compiler และ Compiler คืออะไร

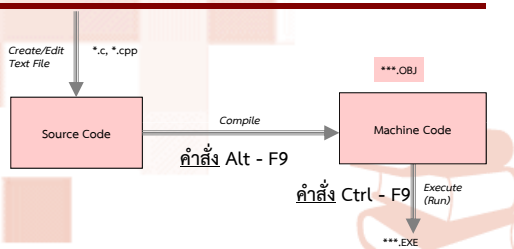


ตัวแปรภาษาคืออะไร???

- **ทำอย่างไร??? ให้คอมพิวเตอร์เข้าใจในภาษาของมนุษย์ เพราะเนื่องจากว่ามนุษย์และคอมพิวเตอร์มีภาษาที่ใช้ในรูปแบบที่แตกต่างกัน**



การแปลภาษาโปรแกรมเป็นภาษาเครื่อง



การจัดทำเอกสารประกอบโปรแกรม

- **เอกสารประกอบโปรแกรมนั้นแบ่งเป็น 2 ประเภท คือ**
  - เอกสารประกอบโปรแกรมสำหรับผู้ใช้งาน ***(User Documentation)***
  - เอกสารประกอบโปรแกรมสำหรับผู้พัฒนาโปรแกรม ***(Technical Documentation)***

รูปแบบของเอกสารประกอบโปรแกรม

- สำหรับเอกสารประกอบโปรแกรมสำหรับผู้ใช้งาน ***(User Documentation)*** นั้น
- จะประกอบไปด้วย รายละเอียดของโปรแกรมที่จัดทำขึ้น
- วิธีการติดตั้งโปรแกรม
- วิธีการใช้งานของโปรแกรม อธิบายถึงขั้นตอนการทำงานต่างๆ ของโปรแกรม

รูปแบบของเอกสารประกอบโปรแกรม

- สำหรับเอกสารประกอบโปรแกรมสำหรับผู้พัฒนาโปรแกรม ***(Technical Documentation)*** นั้น
- จะประกอบไปด้วย รายละเอียดของการออกแบบโปรแกรม อธิบายถึงภาษาที่ใช้ในการพัฒนา
- อธิบายหน้าที่หลักๆ ของฟังก์ชันแต่ละตัว รวมไปถึงเทคนิคที่โปรแกรมเมอร์คิดขึ้นเอง และข้อเด่น - ข้อจำกัดของโปรแกรมที่พัฒนาขึ้น

การบำรุงรักษาโปรแกรม

- หลังการใช้งานโปรแกรมไปนานๆ แล้วพบว่าต้องมีขั้นตอนของการบำรุงรักษาโปรแกรมเกิดขึ้นด้วย
- โดยอาจจะเป็นการปรับปรุงหรือแก้ไขโปรแกรมในบางส่วน เพื่อให้สอดคล้องกับความต้องการในปัจจุบัน
- หรืออาจจะเป็นการสร้างระบบขึ้นใหม่เพื่อ ทดแทนระบบเดิม ก็อาจเป็นไปได้

ประสิทธิภาพของการออกแบบโปรแกรม

- **ความถูกต้อง *(Correctness)***
  - ...ถูกต้องแค่ไหน รองรับกับข้อมูลเข้าทุกกรณีหรือเปล่า...
- **ความชัดเจน *(Clarity)***
  - ...วิธีที่แก้ไขปัญหานั้น มีความชัดเจน ไม่คลุมเครือ...
- **ประสิทธิภาพ *(Efficiency)***
  - ...ใช้เวลานานแค่ไหนในการได้มาซึ่งคำตอบ และเสียทรัพยากรของระบบไปเท่าไหร่...