เอกสารประกอบการสอน
รายวิชาการเขียนโปรแกรมภาษาคอมพิวเตอร์ 1

สัมฤทธิ์ เสนภาศ

คณะเทคโนโลยีสารสนเทศ
มหาวิทยาลัยราชภัฏเทพสตรี
2548

เอกสารประกอบการสอน
รายวิชาการเขียนโปรแกรมภาษาคอมพิวเตอร์ 1

สัมฤทธิ์ เสนกาศ
ค.อ.ม. คอมพิวเตอร์และเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศ
มหาวิทยาลัยราชภัฏเทพสตรี
ISBN 974-9607-15-5
2548

คำนำ

เอกสารประกอบการสอน รายวิชา 4121202 การเขียนโปรแกรมภาษาคอมพิวเตอร์ 1 เป็นเอกสารใช้เป็นเอกสารประกอบการสอนในหมวดวิชาเฉพาะด้าน ของนักศึกษาที่เรียนวิชาเอกคอมพิวเตอร์ ได้แก่ วิทยาการคอมพิวเตอร์ คอมพิวเตอร์ศึกษา และ เทคโนโลยีสารสนเทศ ของมหาวิทยาลัยราชภัฏ กล่าวถึงการศึกษาความรู้ในด้านการเขียนคำสั่งให้คอมพิวเตอร์ทำงาน ในลักษณะโปรแกรมภาษา เนื้อหาของเอกสาร กล่าวถึง คำสั่งพื้นฐานที่จำเป็นในการเขียนโปรแกรม ข้อมูลและการจัดเก็บข้อมูลชนิด ต่าง ๆ ที่นำมาใช้ในโปรแกรมคอมพิวเตอร์ การทำงานของโปรแกรมคอมพิวเตอร์ในลักษณะต่าง ๆ จนถึง การเก็บข้อมูลด้วยหน่วยความจำภายนอก เอกสารนี้ใช้ภาษาซี พลัสพลัส เป็นภาษาที่ใช้ประกอบการฝึกเขียนโปรแกรม แบ่งเนื้อหาออกเป็น 6 บทเรียน เริ่มจากเนื้อหาขั้นพื้นฐาน ไปจนถึงการเขียนคำสั่งชั้นสูง เพื่อใช้ใน การเก็บข้อมูลด้วยหน่วยความจำภายนอก นอกจากที่จะใช้เป็นเอกสารประกอบการสอนแล้ว ยังเหมาะสำหรับการศึกษาภาษาซี พลัสพลัส สำหรับผู้เริ่มต้น ผู้เขียนหวังว่า เอกสารเล่มนี้ คงจะมีประโยชน์สำหรับผู้อ่านไม่มากก็น้อย

ขอกราบขอบพระคุณ บุพการี ครู อาจารย์ และผู้ที่เกี่ยวข้องกับการเขียนเอกสารเล่มนี้ ด้วยความจริงใจ คุณงามความดีของเอกสารเล่มนี้ ขออุทิศเป็นส่วนกุศลให้ คุณพ่อผู้ล่วงลับ และเจ้ากรรมนายเวรทั้งหลาย ส่วนข้อบกพร่องผู้เขียนขอน้อมรับไว้แต่เพียงผู้เดียว

สัมฤทธิ์ เสนกาศ

2 สิงหาคม 2548

สารบัญ

สารบัญภาพ

หน้า

สารบัญภาพ

สารบัญภาพ

สารบัญภาพ

สารบัญตาราง

แผนบริหารการสอนประจำวิชา

รหัสวิชา 4121202

ชื่อวิชา การเขียนโปรแกรมภาษาคอมพิวเตอร์ 1 3(2-2)

เวลาเรียน 64 ชั่วโมง/ภาคเรียน

คำอธิบายรายวิชา

ศึกษาหลักการเขียนรูปแบบไวยากรณ์ประกอบภาษาคอมพิวเตอร์เกี่ยวกับ คำสั่ง I/O ชนิดของข้อมูลแบบต่างๆ operations, looping โปรแกรมย่อยและ ฟังก์ชัน ต่างๆ และการใช้แฟ้มข้อมูลเบื้องต้น โดยการใช้ภาษาคอมพิวเตอร์ ภาษาใดภาษาหนึ่ง เช่น Pascal, Cobol, C, etc. ในการฝึกเขียนและพัฒนาโปรแกรม

วัตถุประสงค์ทั่วไป

1. เพื่อให้ผู้เรียนมีความรู้ถึงการเขียน และ รูปแบบของไวยากรณ์ ของภาษาคอมพิวเตอร์เกี่ยวกับคำสั่ง รับข้อมูล และ แสดงผลข้อมูล
2. เพื่อให้ผู้เรียนมีความรู้เกี่ยวกับข้อมูล และ ชนิดของข้อมูล แบบต่าง ๆ
3. เพื่อให้ผู้เรียนมีความรู้เกี่ยวกับเครื่องหมาย ที่ใช้ในการสั่งงานต่าง ๆ
4. เพื่อให้ผู้เรียนมีความรู้เกี่ยวกับคำสั่ง ทำซ้ำ
5. เพื่อให้ผู้เรียนมีความรู้เกี่ยวกับการใช้โปรแกรมย่อย และ ฟังก์ชัน
6. เพื่อให้ผู้เรียนมีความรู้เกี่ยวกับแฟ้มข้อมูลในระบบคอมพิวเตอร์
7. เพื่อให้ผู้เรียนสามารถออกแบบและเขียนโปรแกรมคอมพิวเตอร์ภาษาซีได้

เนื้อหา

ฟังก์ชันเกี่ยวกับสตริง
ฟังก์ชันทั่ว ๆ ไป

บทที่ 4 คำสั่งควบคุมการทำงาน 14 ชั่วโมง

การทำงานแบบลำดับ
การทำงานแบบเลือกทำ
การทำงานแบบทำซ้ำ

บทที่ 5 โปรแกรมย่อย 12 ชั่วโมง

ความหมายของโปรแกรมย่อย
ประโยชน์ของโปรแกรมย่อย
โปรแกรมย่อยในภาษาซี
การส่งค่าผ่านพารามิเตอร์
ฟังก์ชันที่ไม่มีการส่งค่ากลับ
ฟังก์ชันที่มีการส่งค่ากลับ
ฟังก์ชันที่มีการส่งค่าไปกลับหลายค่า

บทที่ 6 ข้อมูลแบบโครงสร้าง 14 ชั่วโมง

โครงสร้างข้อมูลชนิดแถวลำดับ
โครงสร้างข้อมูลชนิดเรคคอร์ด
โครงสร้างข้อมูลชนิดแฟ้มข้อมูล

กิจกรรมการเรียนการสอน

1. ศึกษาเอกสารประกอบการสอน
2. บรรยาย
3. สาธิตการใช้คำสั่ง
4. ฝึกปฏิบัติ
5. แบบฝึกหัด

สื่อการเรียนการสอน

1. เอกสารประกอบการสอน
2. เครื่องคอมพิวเตอร์
3. บทเรียนบน Internet

การวัดและการประเมินผล

1. การวัดผล

1.1 คะแนนระหว่างภาค 60 คะแนน

- แบบฝึกหัด 10 คะแนน
- การมีส่วนร่วมในกิจกรรมการเรียน 10 คะแนน
- สอบกลางภาค 40 คะแนน

1.2 คะแนนปลายภาค 40 คะแนน

- สอบปลายภาค 40 คะแนน

2. การประเมินผล ใช้เกณฑ์ค่าระดับคะแนนดังนี้

80-100 คะแนน	ได้ระดับคะแนน	A	=	4.0
75-79 คะแนน	ได้ระดับคะแนน	B+	=	3.5
70-74 คะแนน	ได้ระดับคะแนน	B	=	3.0
65-69 คะแนน	ได้ระดับคะแนน	C+	=	2.5
60-64 คะแนน	ได้ระดับคะแนน	C	=	2.0
55-59 คะแนน	ได้ระดับคะแนน	D+	=	1.5
50-54 คะแนน	ได้ระดับคะแนน	D	=	1.0
0-49 คะแนน	ได้ระดับคะแนน	E	=	0

แผนบริหารการสอนประจำบทที่ 1

เนื้อหาประจำบท

คำจำกัดความของคอมพิวเตอร์

ประเภทของคอมพิวเตอร์

คุณสมบัติของเครื่องคอมพิวเตอร์

หน่วยความจำ

หน่วยวัดข้อมูลในระบบคอมพิวเตอร์

หน่วยวัดความเร็ว

โครงสร้างของเครื่องคอมพิวเตอร์

องค์ประกอบพื้นฐานระบบงานคอมพิวเตอร์

ภาษาคอมพิวเตอร์

โปรแกรมและหลักการพัฒนาโปรแกรม

หลักเกณฑ์ทั่ว ๆ ไปในการเขียนผังงาน

สรุป

คำถามทบทวน

วัตถุประสงค์เชิงพฤติกรรม

1. เพื่อให้ผู้เรียนสามารถอธิบายถึงความรู้เบื้องต้นในการใช้คอมพิวเตอร์ได้
2. เพื่อให้ผู้เรียนสามารถอธิบายความหมายของคอมพิวเตอร์และระบบคอมพิวเตอร์ได้
3. เพื่อให้ผู้เรียนสามารถอธิบายองค์ประกอบพื้นฐานระบบงานคอมพิวเตอร์ได้
4. เพื่อให้ผู้เรียนสามารถอธิบายเกี่ยวกับภาษาคอมพิวเตอร์ได้
5. เพื่อให้ผู้เรียนสามารถอธิบายความหมายโปรแกรมและหลักการพัฒนาโปรแกรมได้

วิธีสอนและกิจกรรมการเรียนการสอน

1. สอนแบบบรรยาย นำเข้าสู่บทเรียนด้วยการนำเสนอการใช้งานคอมพิวเตอร์ทั่วๆไปในปัจจุบัน และ วิธีการสั่งให้คอมพิวเตอร์ทำงาน

2. กิจกรรมการเรียนการสอน

2.1 แสดงตัวอย่างการใช้งานคอมพิวเตอร์ ในชีวิตประจำวัน

2.2 ฟังบรรยาย

2.3 ทำแบบฝึกหัดท้ายบท

สื่อการเรียนการสอน

1. เอกสารประกอบการเรียนบทที่ 1
2. เพาเวอร์พอยท์ พรีเซนเตชั่น (power point presentation) ประกอบการบรรยาย

การวัดและการประเมินผล

1. สังเกตจากการตอบคำถาม และการตั้งคำถาม ในชั้นเรียน
2. วัดเจตคติจากการสังเกตพฤติกรรม ความกระตือรือร้นในการทำกิจกรรม
3. ความถูกต้องและคุณภาพของการทำแบบฝึกหัดท้ายบทเรียน

บทที่ 1
ความรู้เบื้องต้นเกี่ยวกับคอมพิวเตอร์

ในปัจจุบันคอมพิวเตอร์เข้ามามีบทบาทกับชีวิตประจำวันของผู้คนอย่างมากมายและยากที่จะหลีกเลี่ยงได้ การใช้คอมพิวเตอร์เพื่อช่วยผ่อนแรงสมองของผู้คนเป็นสิ่งที่นิยมเป็นอย่างยิ่ง ด้วยเหตุผลที่ว่า คอมพิวเตอร์มีการจัดเก็บข้อมูลอย่างเป็นระเบียบ และมีระบบการทำงานที่เป็นระบบระเบียบ ดังนั้นการหาคำตอบที่ยากสำหรับมนุษย์เรา จึงเป็นเรื่องที่ง่ายสำหรับเครื่องคอมพิวเตอร์ ทั้งนี้ไม่ได้หมายความว่า เครื่องคอมพิวเตอร์นั้นจะสามารถทำงานแบบที่ว่าได้ด้วยตัวเอง หากแต่ว่ามนุษย์ต่างหากที่เป็นผู้สั่งการอยู่เบื้องหลัง การออกคำสั่งให้คอมพิวเตอร์ทำตามนั้นเป็นเรื่องที่จำเป็นสำหรับผู้ที่ต้องการใช้งานดังกล่าว เนื่องจากคอมพิวเตอร์เป็นเครื่องจักร ทางอิเลคทรอนิกส์ ที่ใช้คำสั่งแตกต่าง ไปจากภาษา ที่เราเคยสื่อสารกัน หลังจากที่มีการใช้งานคอมพิวเตอร์ มาเป็นเวลานาน จึงมีความพยายามที่จะทำให้การสั่งงานให้ง่ายขึ้น เพื่อให้ใกล้เคียงกับภาษาที่คนเราใช้มากที่สุด จึงมีวิวัฒนาการทางด้านการสั่งงานที่ดีขึ้นเรื่อย ๆ มา จนถึงยุคปัจจุบัน เราสามารถนำเอาเครื่องคอมพิวเตอร์ ไปเชื่อมต่อในลักษณะของเครือข่าย สามารถแลกเปลี่ยนข้อมูลข่าวสารกันได้ ทั่วโลก ในเวลาอันสั้น จึงเป็นเรื่องที่น่าสนใจเป็นอย่างยิ่งในปัจจุบัน ที่เราคงปฏิเสธว่าจะไม่ เกี่ยวข้องกับคอมพิวเตอร์ไม่ได้ บทเรียนนี้จะกล่าวถึง พื้นฐานของเครื่องคอมพิวเตอร์ตลอดจนถึงวิธีการสั่งงานในปัจจุบัน ซึ่งทำได้ทั้งการเขียนโปรแกรม และการใช้งานโปรแกรมสำเร็จรูป แต่จะเน้นไปในทางการสั่งงานด้วยการเขียนโปรแกรมคอมพิวเตอร์ ตลอดจนถึง การออกแบบและพัฒนาโปรแกรมคอมพิวเตอร์

คำจำกัดความของคอมพิวเตอร์

จากที่คนในยุคโบราณเคย คิดค้นเครื่องมือต่าง ๆ ขึ้นมา เช่น ล้อเลื่อน คาน รอก และเครื่องมืออื่น ๆ ล้วนแล้วแต่เป็นเครื่องมือที่ช่วยผ่อนแรงกาย เพื่อให้ใช้พลังงานน้อยลง และได้เนื้องานมากขึ้น จนโลกมีวิวัฒนาการขึ้นเรื่อย ๆ จนถึงยุคที่คนเราต้องออกมา ทำงานนอกบ้านเพื่อการดำรงชีวิต การที่คนเราได้มาร่วมงานกันจึงได้มีการจัดระเบียบทางสังคม เพื่อให้อยู่ร่วมกันอย่างสงบสุข ดังนั้นสิ่งที่ตามมาอีกเรื่องหนึ่งก็คือ ข้อมูลข่าวสารของแต่ละบุคคลว่า มีความเป็นมาอย่างไร มีความสำคัญต่อสังคมอย่างไร จึงเกิดข้อมูลจำนวนมหาศาลขึ้นมา ให้คนเราทำการวิเคราะห์ เพื่อหาข้อเท็จจริงให้ทันกับเหตุการณ์ เครื่องคอมพิวเตอร์จึงเป็นเครื่องมือที่คนเราออกแบบขึ้นมา เพื่อช่วยในการวิเคราะห์ข้อมูลจำนวน มากมายดังกล่าว ให้เสร็จสิ้นภายในเวลาอันรวดเร็ว ทั้งนี้เนื่องจากการแข่งขันทางด้านการควบคุมเศรษฐกิจ เพื่อความอยู่รอดของมวลมนุษยชาติ คอมพิวเตอร์ จึงเป็น

เครื่องมือที่ออกแบบขึ้นมาเพื่อช่วยผ่อนแรง ทางด้านการใช้สมองมนุษย์ เป็นเครื่องมือที่ช่วยวิเคราะห์ และ ทำการประมวลผล ที่มนุษย์ทำได้ยาก ให้เป็นเรื่องง่าย ถูกต้อง และ รวดเร็ว ดังมีผู้ให้ทัศนะเกี่ยวกับคอมพิวเตอร์ว่า

ทักษิณา สวนานนท์ (2544, หน้า 120) กล่าวว่า คอมพิวเตอร์ หรือ คณิตกรณ์ แปลว่า ผู้คำนวณ คือ อุปกรณ์ชนิดหนึ่งที่ทำงานด้วยระบบอิเลกทรอนิกส์ สามารถจำข้อมูลและคำสั่งได้ ทำให้สามารถทำงานไปได้โดยอัตโนมัติด้วยอัตราความเร็วที่สูงมาก ใช้ประโยชน์ในการคำนวณ หรือการทำงานต่าง ๆ ได้เกือบทุกชนิด มี 3 ขนาด คือ ขนาดใหญ่ (mainframe computer) ขนาดกลาง (mini computer) และขนาดเล็กที่กำลังได้รับความนิยมทั่วไปในขณะนี้ เรียกว่าไมโครคอมพิวเตอร์ (microcomputer) หรือ คอมพิวเตอร์ส่วนบุคคล (personal computer) ที่เรียกกันย่อ ๆ ว่าคอมพิวเตอร์ประเภท PC ปัจจุบันการใช้ระบบเครือข่ายทำให้เราสามารถใช้คอมพิวเตอร์เป็นที่ ค้นหาข้อมูลต่างๆ ได้ สื่อสารได้ นอกเหนือไปจากการใช้เพื่อการคำนวณตามวัตถุประสงค์ดั้งเดิมของผู้ประดิษฐ์

สถาบันส่งเสริมการสอนวิทยาศาสตร์และเทคโนโลยี (2539, หน้า 2) ให้คำจำกัดความว่าเครื่องคอมพิวเตอร์มีขั้นตอนการทำงาน 3 ขั้นตอน คือ

1. รับโปรแกรม และข้อมูล หมายถึง ชุดของคำสั่งที่จะให้คอมพิวเตอร์ทำงาน เรียกว่าโปรแกรมคอมพิวเตอร์ ส่วนข้อมูลอาจเป็นตัวเลขหรือตัวหนังสือที่ต้องการให้คอมพิวเตอร์ ทำการประมวลผล

2. ทำการประมวลผล หมายถึง การจัดระเบียบแบบแผนของข้อมูล เพื่อให้ได้ผลลัพธ์ตามที่ต้องการ ทำได้โดยการคำนวณ เปรียบเทียบ วิเคราะห์โดยใช้สูตรทางวิทยาศาสตร์หรือคณิตศาสตร์ วิธีการต่าง ๆ เหล่านี้ทำได้โดยอาศัยคำสั่งหรือโปรแกรมที่เขียนขึ้น

3. แสดงผลลัพธ์ หมายถึง การนำผลลัพธ์ที่ได้จากการประมวลผลเสร็จเรียบร้อยแล้วแสดงออกมาในรูปแบบต่าง ๆ ที่ผู้ใช้เข้าใจ นำไปใช้ประโยชน์ได้

จากคำจำกัดความต่าง ๆ พอจะสรุปได้ว่า เครื่องคอมพิวเตอร์ เป็นเครื่องมือที่ออกแบบขึ้นมาเพื่อช่วยเหลือคนเราทางด้านความจำ ความคิด ที่มีความสามารถวิเคราะห์ และ ประมวลผลข้อมูล ได้อย่างรวดเร็ว และ ถูกต้อง อีกทั้งยังมีความสามารถทางด้านอื่น ๆ ที่คนเราสามารถนำไปประยุกต์ให้เข้ากับ การดำรงชีวิตได้

ประเภทของคอมพิวเตอร์

การแบ่งประเภทของคอมพิวเตอร์ เพื่อให้เหมาะสมกับงานที่จะนำไปใช้งานนั้น สามารถแบ่งคอมพิวเตอร์ออกเป็นหลายประเภท ตามลักษณะต่าง ๆ เช่นงานที่มีปริมาณข้อมูลน้อย หรืองาน

มีข้อมูลมาก และ ต้องใช้ความละเอียดในการรับส่งข้อมูลสูง เรื่องคอมพิวเตอร์ที่จะนำมาใช้งานย่อมมีความแตกต่างกันออกไป

วาสนา สุขกระสานติ (2540: หน้า 1-4) กล่าวว่า คอมพิวเตอร์ในปัจจุบันสามารถแบ่งเป็นประเภทต่างๆ โดยใช้ความแตกต่างจากขนาดของเครื่องความเร็วในการประมวลผล รวมทั้งราคาเป็นหลักคือ

1. ซูเปอร์คอมพิวเตอร์ เป็นคอมพิวเตอร์ที่มีขนาดใหญ่ที่สุด ทำงานได้รวดเร็ว และมีประสิทธิภาพสูง มีหน่วยประมวลผลเป็นร้อยตัวเพื่อทำงานพร้อมกัน นิยมนำมาใช้ในงานที่มีการคำนวณซับซ้อน เช่นการคำนวณทางวิทยาศาสตร์ การบิน อุตสาหกรรมน้ำมันเป็นต้น

2. เมนเฟรมคอมพิวเตอร์ จัดเป็นเครื่องที่มีประสิทธิภาพรองลงมาจากเครื่อง ซูเปอร์คอมพิวเตอร์ มีหน่วยประมวลผลน้อยกว่า รองรับผู้ใช้ได้หลายร้อยคนพร้อมกัน เหมาะสำหรับองค์กรขนาดใหญ่

3. มินิคอมพิวเตอร์ เป็นคอมพิวเตอร์ขนาดกลาง เหมาะสำหรับองค์กรขนาดกลาง ที่มีผู้ใช้งานพร้อมกันประมาณไม่เกินสองร้อยคน ความเร็วในการทำงานจะช้ากว่าเครื่องเมนเฟรม รวมไปถึงสื่อต่าง ๆ ที่ใช้ในการเก็บข้อมูลจะมีความจุน้อยกว่า

4. ไมโครคอมพิวเตอร์ เป็นคอมพิวเตอร์ขนาดเล็ก ออกแบบมาสำหรับใช้งานส่วนตัว จึงเรียกได้อีกอย่างว่า คอมพิวเตอร์พีซี เป็นคอมพิวเตอร์ที่มีราคาถูก ปัจจุบันสามารถเชื่อมต่อเป็นเครือข่ายได้ จึงเป็นเครื่องคอมพิวเตอร์ที่นิยมใช้กันมากที่สุด

5. คอมพิวเตอร์สำหรับเครือข่าย (network computers) เป็นคอมพิวเตอร์ที่ออกแบบมาเพื่อใช้งานด้านการเชื่อมต่อเข้ากับเครือข่ายคอมพิวเตอร์โดยเฉพาะ เช่นการเชื่อมต่อเครือข่ายอินเตอร์เน็ต คอมพิวเตอร์ประเภทนี้ จะไม่มีการเก็บข้อมูลสำรองในตัว แต่จะมีการเก็บข้อมูลที่เครื่องคอมพิวเตอร์แม่ข่าย (server) ทำให้มีราคาต่ำกว่าไมโครคอมพิวเตอร์ทั่วไป นิยมใช้ในองค์กรที่มีการใช้คอมพิวเตอร์ในปริมาณมาก

6. คอมพิวเตอร์แบบฝัง (embedded computers) เป็นคอมพิวเตอร์ที่ฝังในอุปกรณ์ต่างๆ ที่มีหน้าที่ทำงานเฉพาะด้านเช่น ระบบการเติมน้ำมัน เตาไมโครเวฟ อุปกรณ์เล่นเกมส์ เป็นต้น

ศิริรัตน์ ชำนาญรบ และคณะ (2539, หน้า 14) กล่าวว่า เครื่องคอมพิวเตอร์โดยทั่วไป มี 3 แบบ คือ

1. อนาลอกคอมพิวเตอร์ (analog computer) คือ คอมพิวเตอร์ที่รับข้อมูลในรูปของปริมาณที่วัดต่อเนื่องกันแล้วแปลงปริมาณนั้นให้เป็นตัวเลข เช่น เครื่องวัดอุณหภูมิของอากาศ อุปกรณ์จ่ายน้ำมันในสถานีบริการน้ำมัน จึงเหมาะกับงานด้านวิทยาศาสตร์ คณิตศาสตร์ และวิศวกรรมศาสตร์

2. ดิจิตอลคอมพิวเตอร์ (digital computer) คือคอมพิวเตอร์ที่มีการทำงานแบบมีการคำนวณโดยการนับจำนวนโดยตรง ซึ่งมีลักษณะการจัดเก็บข้อมูลด้วยระบบเลขฐานสอง คือ 0 กับ 1 ขั้นตอนในการประมวลผลเป็นไปอย่างต่อเนื่อง มีหน่วยความจำ เป็นที่เก็บข้อมูล และ แลกเปลี่ยนข้อมูลได้ตลอดเวลา คอมพิวเตอร์ทั่วๆไปมักเป็นดิจิตอลคอมพิวเตอร์เป็นส่วนใหญ่ เพราะคอมพิวเตอร์ประเภทนี้สามารถนำไปประยุกต์ใช้งานได้อย่างกว้างขวาง เช่น คอมพิวเตอร์ส่วนบุคคล คอมพิวเตอร์ที่ใช้ในการสำรองที่นั่งของบริษัทการบิน และคอมพิวเตอร์ที่ใช้ในการควบคุมการจ่ายเงินสำหรับตู้ ATM เป็นต้น

3. ไฮบริดจ์คอมพิวเตอร์ (hybrid computer) คือ การผสมผสานกันระหว่างอนาลอกคอมพิวเตอร์ กับดิจิตอลคอมพิวเตอร์ มีลักษณะสร้างขึ้นเพื่อใช้งานเฉพาะกิจในงานด้านวิทยาศาสตร์ หรืองานควบคุมกระบวนการทางอุตสาหกรรม ด้านการแพทย์ เช่น เครื่องตรวจวัดสายตา การจำลองเหตุการณ์ของเครื่องฝึกบิน เป็นต้น

การแบ่งประเภทของคอมพิวเตอร์นั้น มีหลักการในการแบ่งประเภทหลายแบบ ให้เหมาะสมกับการใช้งาน ถ้ามองในแง่ของการใช้งาน สรุปได้ว่า คอมพิวเตอร์แบ่งได้ 2 ประเภทคือ แบบใช้งานทั่วไป และ แบบใช้งานเฉพาะด้าน มองในแง่ของข้อมูลที่ใช้ในระบบคอมพิวเตอร์ แบ่งออกเป็น ประเภทอนาลอก ประเภทดิจิตอล และประเภทไฮบริดจ์ และถ้ามองในแง่ของความเร็วและประสิทธิภาพแบ่งได้เป็น ซูเปอร์คอมพิวเตอร์ เมนเฟรม มินิคอมพิวเตอร์ และไมโครคอมพิวเตอร์

คุณสมบัติของเครื่องคอมพิวเตอร์

เครื่องคอมพิวเตอร์ เป็นเครื่องมือ ที่ทำงานโดยระบบอัตโนมัติ หลังจากที่เราได้ออกคำสั่งที่ถูกต้องให้ทำการประมวลผล สามารถจดจำข้อมูลจำนวนมากไว้ในหน่วยความจำและเรียกใช้อย่างง่ายและถูกต้อง มีความเร็วในการประมวลผลข้อมูลสูงทำให้ได้ปริมาณงานมาก ขณะที่ใช้เวลาน้อย สามารถติดต่อสื่อสารกันได้เหมือนกับสิ่งมีชีวิตอื่น ๆ ทั้งที่เป็นสิ่งไม่มีชีวิต และเป็นเครื่องมือที่เชื่อถือได้ในด้านผลลัพธ์ที่ทำการประมวลผล

สถาบันส่งเสริมการสอนวิทยาศาสตร์และเทคโนโลยี (2539, หน้า 2-3) ได้ให้คำจำกัดความเกี่ยวกับคุณสมบัติของเครื่องคอมพิวเตอร์ไว้ว่า เครื่องคอมพิวเตอร์มีคุณสมบัติสำคัญ 4 อย่าง คือ

1. ทำงานโดยอัตโนมัติ ถ้าสังเกตการทำงานของคอมพิวเตอร์จะเห็นว่าอุปกรณ์ทุกอย่างของคอมพิวเตอร์ทำงานได้เองโดยอัตโนมัติโดยที่คนที่ไม่ได้เข้าไปควบคุมไม่ว่าจะเป็นการอ่านข้อมูล คำนวณ หรือพิมพ์ผลลัพธ์ อย่างไรก็ตามเครื่องจักรก็ยังคงเป็นเครื่องจักร ไม่มีชีวิตจิตใจคิดและทำด้วยตัวเองเหมือนกับสิ่งมีชีวิตทั่วไปไม่ได้ การที่อุปกรณ์ทุกส่วนของคอมพิวเตอร์ทำงาน

ต่อเนื่องกันโดยอัตโนมัติได้นั้น แท้จริงแล้วจะต้องอาศัยโปรแกรมที่เขียนขึ้น โดยโปรแกรมนั้นจะบอกขั้นตอนโดยละเอียดว่า ให้อุปกรณ์ส่วนไหนของคอมพิวเตอร์ทำอะไรก่อน และทำอย่างไรจึงจะได้ผลลัพธ์ตามที่ต้องการ ดังนั้นความเป็นอัตโนมัติของคอมพิวเตอร์จึงอยู่ที่ความสามารถทำงานตามคำสั่งของมนุษย์

2. ทำงานได้เอนกประสงค์ เครื่องคอมพิวเตอร์จัดอยู่ในประเภทเอนกประสงค์ เพราะทำงานได้หลายชนิดขึ้นอยู่กับโปรแกรมที่ใช้ เช่น ถ้าใส่โปรแกรมบัญชีเงินเดือน ก็จะสามารถใช้คำนวณ และพิมพ์รายการเงินเดือนที่ได้รับ ตลอดจนรายการต่าง ๆ ที่ถูกหักของแต่ละเดือน ถ้าเปลี่ยนเป็นโปรแกรมคิดคะแนนสอบของนักเรียนเมื่อสิ้นภาคการศึกษา คอมพิวเตอร์จะทำการอ่านคะแนนสอบแต่ละวิชา ทำการคำนวณและพิมพ์ผลการสอบของนักเรียนแต่ละคน เป็นต้น

3. เป็นอุปกรณ์อิเลคทรอนิกส์ ทุกคนคงคุ้นเคยอยู่กับอุปกรณ์ที่เป็นเครื่องจักรกล เช่น เครื่องยนต์ จักรเย็บผ้า เครื่องสูบน้ำ ซึ่งจะสังเกตเห็นว่าในขณะที่เครื่องทำงาน จะมีการเคลื่อนไหวของชิ้นส่วนต่าง ๆ ส่วนอุปกรณ์ที่นำมาประกอบกันเข้าเป็นเครื่องคอมพิวเตอร์นั้นล้วนแต่เป็นอุปกรณ์ทางด้านอิเลคทรอนิกส์ เช่น ทรานซิสเตอร์ วงจรไอซี (IC ย่อมาจาก integrated circuit) เป็นต้น การทำงานของระบบคอมพิวเตอร์ เป็นระบบอิเลคทรอนิกส์จึงทำงานด้วยความเร็วสูงมาก

4. เป็นระบบดิจิตอล คำว่าดิจิตอล (digital) มาจากคำว่าดิจิต (digit) ซึ่งหมายถึง ตัวเลข เครื่องคอมพิวเตอร์ส่วนใหญ่ทำงานโดยใช้ระบบตัวเลข ข้อมูลทุกชนิดไม่ว่าจะเป็น ตัวเลข ตัวหนังสือ หรือเครื่องหมายอื่นใดที่ใช้ในทางคณิตศาสตร์ วิทยาศาสตร์ หรือทางธุรกิจ เมื่อส่งเข้าไปยังเครื่องรับข้อมูลทางคอมพิวเตอร์แล้วจะถูกเปลี่ยนเป็นตัวเลขทั้งหมด ซึ่งการทำงานในหน่วยประมวลผลข้อมูลของคอมพิวเตอร์ จะใช้ระบบตัวเลขเท่านั้น

จากคำจำกัดความสรุปได้ว่า เครื่องคอมพิวเตอร์ เป็นเครื่องอิเลคทรอนิกส์ ที่ทำงานได้เอนกประสงค์ แล้วแต่ผู้ใช้จะประยุกต์ใช้ ทำงานด้วยคำสั่งที่เรียกว่าโปรแกรม คำสั่งที่ใส่ลงไปทั้งหมดจะถูกแปลงให้เป็นระบบตัวเลขภายในหน่วยความจำ และทำงานได้แบบอัตโนมัติ

หน่วยความจำ

การเก็บข้อมูลในคอมพิวเตอร์นั้น จำเป็นต้องมีหน่วยความจำ (memory) เพื่อให้เครื่องนำข้อมูลไปเก็บ ในแต่ละโปรแกรมจะมีการนำเอาข้อมูลต่าง ๆ ไปเก็บยังหน่วยความจำ ก่อนที่จะนำเอาข้อมูลเหล่านั้นมาใช้ประโยชน์ ดังกล่าวแล้วว่าข้อมูลต่าง ๆ ที่ใส่ลงไปในเครื่องคอมพิวเตอร์จะถูกเปลี่ยนให้อยู่ในระบบตัวเลข ซึ่งตัวเลขที่เป็นตัวแทนของ การมี และ ไม่มี กระแสไฟฟ้า คือ เลขฐานสอง (binary digit) ดังนั้นเมื่อมีการป้อนข้อมูลเข้าไปในหน่วยความจำ ข้อมูลที่ป้อนเข้าไปจะถูกแปลงให้อยู่ในรูปแบบของกระแสไฟฟ้า ปัจจุบัน ใช้ 8 ช่องสัญญาณ ต่อตัวอักษร 1 ตัว และ

เพื่อให้คอมพิวเตอร์สามารถสื่อสารกันได้ จึงมีการกำหนดรหัสของข้อมูลให้เหมือนกัน โดยทุกวันนี้ เราใช้ระบบของการแปลงข้อมูลให้เป็นรหัสตัวเลข ที่เรียกว่า รหัสแอสกี้ (ASCII ย่อมาจาก American Standard Code for Information Interchange) หน่วยความจำที่ใช้ภายในเครื่องคอมพิวเตอร์แบ่งตามลักษณะการใช้งานออกเป็น 2 ประเภทคือ

1. หน่วยความจำแบบอ่านอย่างเดียว (read only memory) หรือเรียกว่า รอม (ROM) เป็นหน่วยความจำที่ผู้ผลิตเครื่องคอมพิวเตอร์เป็นผู้สร้างและกำหนดการทำงานของเครื่องคอมพิวเตอร์ลงไปเพื่อให้คอมพิวเตอร์ทำงาน ที่จำเป็นตามต้องการ ส่วนใหญ่จะเป็นการทำงานขณะที่เริ่มเปิดเครื่อง ซึ่งขณะนั้นเครื่องจะต้องมีคำสั่งให้สำรวจอุปกรณ์รอบข้างต่าง ๆ ว่าอยู่ในลักษณะพร้อมใช้งานหรือไม่ ถ้าทุกอย่างที่จำเป็นพร้อมใช้งาน โปรแกรมสั่งงานที่อยู่ภายในหน่วยความจำส่วนนี้ จะสั่งให้เริ่มทำงานได้ โดยการสั่งให้ทำงานตามคำสั่งของระบบปฏิบัติการ หน่วยความจำส่วนนี้เป็นหน่วยความจำที่ไม่สามารถเข้าไปเปลี่ยนแปลงได้โดยผู้ใช้

2. หน่วยความจำสำหรับผู้ใช้ (random access memory) หรือเรียกว่า แรม (RAM) เป็นหน่วยความจำ ที่มีไว้ให้เก็บข้อมูลสำหรับผู้ใช้คอมพิวเตอร์ มีลักษณะเสมือน ห้องพักของโรงแรม มีพนักงานบริการนำข้อมูลไปเก็บ และนำข้อมูลกลับมาใช้ ในเวลาอันรวดเร็ว การเก็บข้อมูลในส่วนนี้จะมีลักษณะการนำข้อมูลเข้าไปเก็บ ตามหมายเลขของหน่วยความจำ ซึ่งเรียกว่า แอดเดรส (address) สามารถนำข้อมูลไปเก็บได้แบบเข้าถึงโดยตรง (direct access) ถ้าหากมีการนำข้อมูลใหม่เข้ามาเก็บยังตำแหน่งเดิมที่เคยเก็บ เครื่องจะทำการบันทึกข้อมูลใหม่ลงไปแทนที่ข้อมูลเก่า ทำให้ข้อมูลเก่า ณ ตำแหน่งหน่วยความจำนี้ หายไป ข้อจำกัดของหน่วยความจำชนิดนี้คือ ใช้ได้เฉพาะตอนที่มีกระแสไฟฟ้าเท่านั้น ดังนั้นถ้าหากเราปิดเครื่อง ข้อมูลต่าง ๆ ที่อยู่ในหน่วยความจำส่วนนี้ก็จะหายไป

หน่วยวัดข้อมูลในระบบคอมพิวเตอร์

เมื่อเราใส่ข้อมูลลงไปในระบบคอมพิวเตอร์ เครื่องจะทำการเปลี่ยนให้เป็นระบบไฟฟ้าทั้งหมด ดังนั้นข้อมูลที่ป้อนลงไปในระบบ จะต้องมีการเข้ารหัสข้อมูล เพื่อแปลงเป็นกระแสไฟฟ้าตามที่ต้องการของเครื่อง และเมื่อถึงคราวที่เครื่องจะแสดงผลลัพธ์ให้ผู้ใช้ได้รับทราบ ก็จะทำการแปลงสัญญาณของกระแสไฟฟ้าเหล่านั้นมาเป็น ข้อมูลที่คนเรารับรู้ได้ เพื่อความเข้าใจระบบการจัดเก็บข้อมูลของเครื่องคอมพิวเตอร์ จึงต้องเรียนรู้ถึงเรื่องหน่วยวัดข้อมูลในเครื่องคอมพิวเตอร์ก่อน หน่วยวัดข้อมูลที่อยู่ในหน่วยความจำของเครื่องคอมพิวเตอร์ มีหน่วยวัดดังนี้

1. บิต (bit) คือ ส่วนที่เล็กที่สุดของหน่วยความจำ มาจากคำว่า binary digit หมายถึงหลักของเลขฐานสอง ใช้สัญลักษณ์แทนค่าอยู่ 2 ตัวคือ 0 กับ 1 เราจึงนำมาใช้แทนสถานะทาง ไฟฟ้า

ที่มีความหมายว่า มีกระแส หรือ ไม่มีกระแส ดังนั้นเมื่อนำมาเก็บข้อมูล จะเก็บข้อมูลที่มีลักษณะเป็น 2 สถานะเช่น ใช่ หรือ ไม่ใช่ หญิง หรือ ชาย โสด หรือ สมรส เป็นต้น

2. ไบต์ (byte) คือ การนำเอา หน่วยความจำประเภท บิต มาประยุกต์ใช้ คือนำหลาย ๆ บิตมาเรียงกัน แล้ว เปิดปิดสัญญาณไฟฟ้าในแต่ละบิตให้แตกต่างกัน แล้วใส่ความหมายลงไปว่าสัญญาณการเปิดปิดในแบบนี้หมายถึงอะไร หรือเรียกว่าใส่รหัส ปัจจุบันได้นำบิตมาประกอบกัน 8 บิต หมายความว่า จะมีการเปิดปิดสัญญาณไฟที่เรียงกันอยู่ 8 สัญญาณนี้ โดยไม่ซ้ำกันได้ถึง 256 แบบ นั่นหมายความว่าเราสามารถให้เครื่องจำข้อมูลได้ถึง 256 ตัวโดยไม่ซ้ำกันเลย สัญญาณไฟฟ้า 8 สายสัญญาณหมายถึงตัวอักขระ 1 ตัวในหน่วยความจำของคอมพิวเตอร์ เรียกว่า 1 ไบต์ ดังนั้นถ้าต้องการเก็บข้อความว่า "RITS" จะต้องใช้พื้นที่ในหน่วยความจำเท่ากับ 4 ไบต์

3. กิโลไบต์ (kilo byte) เป็นหน่วยที่ใหญ่กว่า ไบต์ เขียนเป็นอักษรย่อว่า 'K' หรือ 'KB' มีค่าเท่ากับ 1,024 ไบต์ หรือเท่ากับ 2 ยกกำลัง 10 ไบต์

4. เมกะไบต์ (mega byte) เป็นหน่วยที่ใหญ่กว่า กิโลไบต์ เขียนเป็นอักษรย่อว่า 'MB' หมายถึง 1,024 KB หรือ $1,024^2$ ไบต์

5. กิกะไบต์ (giga byte) เป็นหน่วยที่ใหญ่กว่า เมกะไบต์ เขียนเป็นอักษรย่อว่า 'GB' หมายถึง 1,024 MB หรือ $1,024^3$ ไบต์

การวัดว่ามีข้อมูลอยู่ในปริมาณเท่าใดในหน่วยความจำ โดยทั่วไป เราใช้หน่วยวัด เป็นไบต์ ถ้าข้อมูลมีจำนวนมากขึ้น หน่วยวัดก็จะขยายใหญ่ขึ้น เพื่อสะดวกในการวัดปริมาณ เหมือนกับมาตราสำหรับวัดโดยทั่วไป เช่น กรัม กิโลกรัม และ เมตริกตัน เป็นต้น หน่วยวัดหน่วยความจำของเครื่องคอมพิวเตอร์ก็เหมือนหน่วยวัดทั่วไป แต่จะมีหน่วยเป็น ไบต์ กิโลไบต์ เมกะไบต์ กิกะไบต์ และ อื่น ๆ ที่สูงกว่านี้

หน่วยวัดความเร็ว

คอมพิวเตอร์ทำงานด้วยระบบไฟฟ้า จึงมีความเร็วในการประมวลผลสูงมาก การวัดความเร็วในการประมวลผลของเครื่องคอมพิวเตอร์ เทียบได้ในช่วงเวลา 1 วินาที หมายความว่าภายใน 1 ช่วงเวลา หรือ ใน 1 วินาที เครื่องคอมพิวเตอร์สามารถประมวลผลได้กี่คำสั่ง เครื่องคอมพิวเตอร์แต่ละเครื่องจึงมีความสามารถแตกต่าง กันออกไปในเรื่องของประสิทธิภาพ หลังจากให้แต่ละเครื่องได้ทดลองประมวลผลข้อมูลตามที่ต้องการ

ธงชัย สิทธิกรณ์ (2540, หน้า 3) ความเร็วเป็นจุดเด่นทางโครงสร้างของตัวเครื่องคอมพิวเตอร์ เป็นตัวบ่งชี้ประสิทธิภาพของเครื่องคอมพิวเตอร์ ความเร็วนี้อาจเปลี่ยนแปลงได้ตามขนาดของข้อมูล รูปแบบหรือลักษณะของการประมวลผล โดยพิจารณาจากความสามารถในการ

ประมวลผลซ้ำ ๆ ในช่วงเวลาหนึ่ง ๆ เรียกว่า ความถี่ และเนื่องจากมีความเร็วสูงมาก จึงนิยมเทียบความเร็วในหนึ่งวินาทีเสมอ (cycle/second) โดยใช้หน่วยวัดความเร็วเรียกว่า เฮิรตซ์ (hertz :Hz)

ความเร็วในการประมวลผลถูกกำหนดโดย อุปกรณ์ประมวลผล หรือ CPU ซึ่งปัจจุบันมีความเร็วเริ่มต้นตั้งแต่ แสนครั้ง/วินาที (1MHz = 10^6 Hz) ขึ้นไป เช่น CPU Pentium 133 MHz มีความเร็วในการประมวลผลเท่ากับ 133,000,000 ครั้ง/วินาทีเป็นต้น หน่วยวัดความเร็วในการประมวลผล สามารถจำแนกได้ดังตารางที่ 1.1

ตารางที่ 1.1 หน่วยวัดความเร็วในการประมวลผล

หน่วยความเร็ว	สัญลักษณ์	ค่าความเร็ว
Millisecond	ms	หนึ่งในพันของวินาที (1/1,000)
Microsecond	µs	หนึ่งในล้านของวินาที (1/1,000,000)
Nanosecond	ns	หนึ่งในพันล้านของวินาที (1/1,000,000,000)
Picosecond	ps	หนึ่งในล้านล้านของวินาที (1/1,000,000,000,000)

โครงสร้างของเครื่องคอมพิวเตอร์

โครงสร้างของเครื่องคอมพิวเตอร์ หมายถึง อุปกรณ์ภายในตัวเครื่องคอมพิวเตอร์ซึ่งประกอบไปด้วยอุปกรณ์ต่าง ๆ ทั้งที่มองเห็นจากภายนอก และ ส่วนที่อยู่ภายในตัวเครื่องคอมพิวเตอร์ แบ่งตามหน้าที่การทำงานออกเป็นหน่วยทำงาน ต่างๆ ดังนี้

1. หน่วยรับข้อมูล (input unit) คืออุปกรณ์ที่ทำหน้าที่รับข้อมูลโปรแกรมคำสั่งจากภายนอกเข้าสู่เครื่องคอมพิวเตอร์ โดยทำการแปลงข้อมูลหรือคำสั่งที่รับเข้ามาให้เป็นรูปแบบข้อมูลทางอิเลคทรอนิกส์ เพื่อทำการประมวลผลต่อไป เครื่องมือที่ใช้ในส่วนนี้เรียกว่า เครื่องป้อนข้อมูล หรือ เครื่องบันทึกข้อมูล เป็นหน่วยที่ทำหน้าที่เป็นผู้เชื่อมความสัมพันธ์ระหว่างมนุษย์กับเครื่องคอมพิวเตอร์นั่นเอง อุปกรณ์ในการป้อนข้อมูล แบ่งออกตามประเภท และ ลักษณะการนำข้อมูลเข้าได้ 2 ชนิดคือ

1.1 เครื่องป้อนข้อมูลที่ไม่สามารถอ่านข้อมูลต้นทางได้โดยตรง หมายถึง เครื่องมือที่ไม่สามารถนำข้อมูลที่มนุษย์สัมผัสได้ เช่นตัวอักษร หรือรูปภาพบนกระดาษ เข้าสู่เครื่องคอมพิวเตอร์ได้โดยตรง จำเป็นต้องอาศัยเครื่องเปลี่ยนตัวอักษรหรือภาพเหล่านั้นให้อยู่ในรูปของสัญญาณไฟทางไฟฟ้าก่อน ตัวอย่างของเครื่องป้อนข้อมูลชนิดนี้ และเป็นที่นิยมใช้งานทั่วไป คือ แป้นพิมพ์ เครื่องอ่าน/เขียนเทป หรือแผ่นจานเก็บข้อมูล เทอร์มินอล

1.2 เครื่องป้อนข้อมูลที่สามารถอ่านข้อมูลต้นทางได้โดยตรง หมายถึง เครื่องมือที่สามารถรับข้อมูลที่มนุษย์สัมผัสได้ เช่น ตัวอักษร หรือรูปภาพ เข้าสู่เครื่องคอมพิวเตอร์ได้โดยตรง ทำให้การประมวลผลรวดเร็ว เนื่องจากไม่ต้องเสียเวลาจากขบวนการเปลี่ยนข้อมูลจาก สื่อข้อมูลต่าง ๆ เครื่องมือนี้จึงเหมาะกับงานที่มีข้อมูลเข้า/ออกซ้ำซาก เช่น ธนาคาร บัญชี และ การเงิน รวมทั้งสถาปัตยกรรม และงานเพื่อความบันเทิง เป็นต้น ตัวอย่างของเครื่องมือในกลุ่มนี้ได้แก่ เครื่องอ่านตัวอักษรหมึกแม่เหล็ก เครื่องอ่านอักขระด้วยแสง เครื่องอ่านคะแนนด้วยแสง ปากกาแสง ดิจิไทเซอร์ เมาส์ เป็นต้น

2. หน่วยประมวลผลกลาง (central processing unit) หรือ CPU หมายถึง หน่วยที่ทำหน้าที่เป็นศูนย์กลางควบคุมการทำงานของฮาร์ดแวร์ โดยนำข้อมูลจากอุปกรณ์รับข้อมูลมาทำการประมวลผล ตามคำสั่งของโปรแกรม ตลอดจนทำการคำนวณหรือเปรียบเทียบค่าต่าง ๆ และส่งผลลัพธ์ที่ได้ออกไปที่หน่วยแสดงผล ในรูปแบบที่ผู้ใช้เข้าใจ เช่น ทางกระดาษพิมพ์ หรือบันทึกไว้ที่สื่อข้อมูล ที่สามารถนำมาใช้กับคอมพิวเตอร์ได้อีก เช่น แผ่นบันทึกเทปแม่เหล็ก หน่วยประมวลผลกลาง นี้สามารถทำการคำนวณ และโยกย้ายข้อมูลหรือเปรียบเทียบข้อมูลได้อย่างรวดเร็วมาก หน่วยประมวลผลกลางประกอบด้วยส่วนสำคัญ 3 ส่วน คือ หน่วยควบคุม หน่วยคำนวณและตรรกะ และหน่วยความจำ

2.1 หน่วยควบคุม (control unit) ทำหน้าที่ประสานงานและควบคุม การทำงาน ของเครื่องคอมพิวเตอร์ ควบคุมให้อุปกรณ์รับข้อมูลส่งข้อมูลไปที่หน่วยความจำ ติดต่อกับอุปกรณ์แสดงผลเพื่อสั่งให้นำข้อมูลจากหน่วยความจำไปยังอุปกรณ์แสดงผล โดยหน่วยควบคุมของคอมพิวเตอร์จะแปลความหมายของคำสั่งในโปรแกรมของผู้ใช้ และควบคุมให้อุปกรณ์ต่าง ๆ ให้ทำตามคำสั่งนั้น ๆ หน่วยนี้ทำงานคล้ายกับสมองคนซึ่งควบคุมให้ระบบอวัยวะต่าง ๆ ของร่างกายทำงานประสานกัน

2.2 หน่วยคำนวณและตรรกะ (arithmetic and logic unit) หรือที่นิยมเรียกว่า ALU ทำหน้าที่คำนวณทางเลขคณิต และ เปรียบเทียบทางตรรกะ เพื่อทำการตัดสินใจโดยรับข้อมูลจากหน่วยความจำมาไว้ยังที่เก็บข้อมูลชั่วคราวของ ALU ซึ่งเรียกว่า รีจิสเตอร์ (registers) เพื่อทำการคำนวณแล้วส่งผลลัพธ์กลับยังหน่วยความจำ หรือทำการเปรียบเทียบข้อมูล เพื่อตรวจสอบว่าปริมาณหนึ่งน้อยกว่า มากกว่า เท่ากับ หรือ มากกว่า อีกปริมาณหนึ่ง แล้วส่งผลลัพธ์จากการเปรียบเทียบที่มีคำตอบเป็น จริง หรือ เท็จ ไปที่หน่วยความจำเพื่อทำงานต่อไปตามขั้นตอนที่กำหนดไว้ในเงื่อนไข

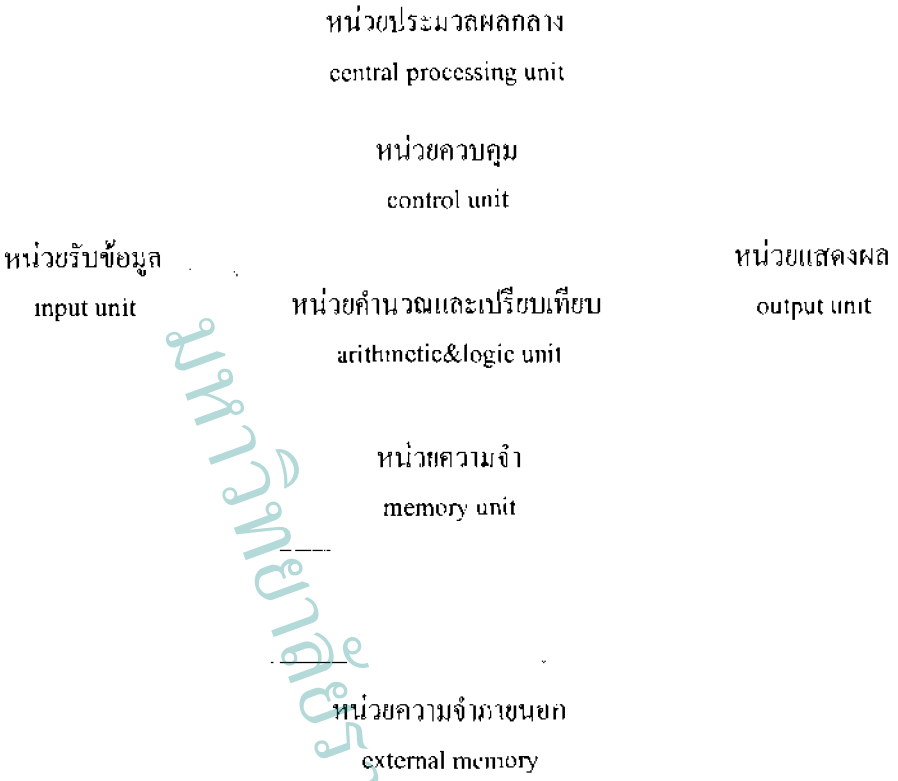
2.3 หน่วยความจำ (memory unit) ทำหน้าที่เก็บโปรแกรม ข้อมูลหรือคำสั่งที่ส่งมาจากหน่วยรับข้อมูล และผลลัพธ์ไว้ภายในคอมพิวเตอร์ ซึ่งรวมถึงสื่อข้อมูลที่ช่วยในการจดจำ เช่น

แผ่นบันทึก เป็นต้น จึงมีหน้าที่เช่นเดียวกับส่วนความจำในสมองของมนุษย์ ที่ควบคุมการทำงานของอวัยวะต่าง ๆ หน่วยความจำของคอมพิวเตอร์ประกอบด้วยวงจรอิเลคทรอนิกส์ ที่รับส่งสัญญาณไฟฟ้าในรูปแบบของรหัสโดยนิยมแทนด้วยตัวเลข 0 และ 1 ซึ่งแทนสถานะ การมีสัญญาณไฟฟ้า ต่ำ และ สูง แบ่งเป็น 2 ส่วนดังกล่าวไว้ในตอนต้น

3. หน่วยแสดงผลลัพธ์ (output unit) ทำหน้าที่แสดงผลจากการประมวลผลโดยนำผลที่ได้ออกมาจากหน่วยความจำหลัก แสดงให้ผู้ใช้ได้เห็นทางจอภาพ หรือ ในรูปของการบันทึกลงสื่อข้อมูล เรียกว่า อุปกรณ์แสดงผล (output device)ได้แก่ จอภาพ เครื่องพิมพ์ เครื่องขับแผ่นบันทึก เครื่องขับจานแม่เหล็ก เครื่องขับเทปแม่เหล็ก และ เครื่องวาดภาพ เป็นต้น

4. หน่วยความจำภายนอก (external memory) หมายถึง หน่วยความจำที่ ออกแบบมาช่วยเสริมให้เครื่องคอมพิวเตอร์ ได้จดจำข้อมูลต่าง ๆ ไว้ใช้หลังจากปิดเครื่องไปแล้ว เป็นการแก้ปัญหาหน่วยความจำภายในที่ไม่สามารถ จดจำข้อมูลได้ เมื่อทำการปิดเครื่อง อีกทั้งหน่วยความจำภายใน ยังมีราคาแพงกว่า หน่วยความจำภายนอก ไม่สามารถติดต่อได้โดยตรง ถ้าจะมีการติดต่อกับหน่วยความจำส่วนนี้ ต้องมีการออกคำสั่งโดยเฉพาะ เช่น สั่งเอาข้อมูลไปเก็บ (save) หรือ สั่งนำเอาข้อมูลเข้ามาใช้งาน (load) เป็นหน่วยความจำที่มีความสามารถใน การบันทึกข้อมูลและคำสั่งได้อย่างคงทน ถาวร เนื่องจากใช้สื่อ (media) ในการเก็บข้อมูล สื่อข้อมูล ที่นิยมใช้ในหน่วยความจำภายนอกที่ใช้กันมากในเวลานี้ คือ จานแม่เหล็ก (disk) เทปแม่เหล็ก (magnetic tape) และ ซีดีรอม (compact disk read only memory :CD–ROM)

โครงสร้างของเครื่องคอมพิวเตอร์แสดงเป็นภาพได้ดังภาพที่ 1.1



ภาพที่ 1.1 โครงสร้างของระบบคอมพิวเตอร์

องค์ประกอบพื้นฐานระบบงานคอมพิวเตอร์

องค์ประกอบพื้นฐานของระบบงานคอมพิวเตอร์ หมายถึง ส่วนประกอบต่าง ๆ ที่จำเป็นต่อระบบงานคอมพิวเตอร์ ที่จะสามารถทำให้คอมพิวเตอร์ ทำงานตามที่เราต้องการได้ องค์ประกอบที่สำคัญประกอบด้วยส่วนต่างๆ ทั้งหมด 5 ส่วน ดังนี้

1. อุปกรณ์ทางคอมพิวเตอร์ หรือเรียกว่า ฮาร์ดแวร์ (hardware)
2. โปรแกรมคอมพิวเตอร์ หรือ เรียกว่า ซอฟต์แวร์ (software)
3. บุคลากรทางคอมพิวเตอร์ หรือ เรียกว่า พีเพิลแวร์ (peopleware)
4. กระบวนวิธีการ (procedure)
5. ข้อมูล (data)

วรรณวิภา จำเริญดารารัศมี (2535, หน้า 21 – 25) กล่าวไว้ว่า การใช้งานคอมพิวเตอร์มีองค์ประกอบสำคัญในระบบดังนี้

1. ฮาร์ดแวร์ (hardware) หรือ ตัวเครื่อง หมายถึงอุปกรณ์ที่ประกอบกันเป็นเครื่องคอมพิวเตอร์ สัมผัสได้ ประกอบด้วย

1.1 อุปกรณ์รับข้อมูล (input devices)

1.2 หน่วยประมวลผล (processor unit)

1.3 อุปกรณ์แสดงผล (output devices)

2. ซอฟต์แวร์ (software) หมายถึง โปรแกรมหรือชุดคำสั่งที่เขียนเป็นภาษาใดภาษาหนึ่งเพื่อกำหนดให้ฮาร์ดแวร์ของระบบคอมพิวเตอร์ทำงานตามที่ต้องการ โปรแกรมดังกล่าวอาจเขียนด้วยภาษาเครื่อง (machine languages) ซึ่งคอมพิวเตอร์สามารถเข้าใจและทำงานได้โดยตรง หรืออาจเขียนขึ้นด้วยภาษาโปรแกรมอื่น ๆ ที่มนุษย์สามารถเข้าใจได้ง่าย ซอฟต์แวร์แบ่งประเภทตามหน้าที่ได้ดังนี้

2.1 ระบบปฏิบัติงาน (operating system :OS) หมายถึง โปรแกรมที่เขียนขึ้นเพื่อควบคุมการทำงานของเครื่องคอมพิวเตอร์ ให้ทำงานตามที่ต้องการได้อย่างมีประสิทธิภาพ

2.2 โปรแกรมประยุกต์ (application program) หมายถึง โปรแกรมที่ผู้ใช้สามารถนำเอามาประยุกต์ใช้กับงานที่ตนเองต้องการได้ เช่น งานด้านบัญชี งานด้านการพิมพ์เอกสาร งานด้านการตกแต่งรูปภาพ งานด้านทะเบียนประวัติพนักงาน เป็นต้น โปรแกรมประยุกต์แบ่งได้ 2 ประเภทคือ

2.2.1 โปรแกรมภาษา (language programming) หมายถึง การเขียนโปรแกรมสั่งให้คอมพิวเตอร์ทำงานโดยใช้ภาษาคอมพิวเตอร์เช่น BASIC ,C ,Pascal เป็นต้น

2.2.2 โปรแกรมสำเร็จรูป (package) หมายถึง โปรแกรมที่มีผู้สร้างขึ้นมาจากโปรแกรมภาษา โดยมีจุดประสงค์เพื่อ ทำให้การใช้งานง่ายขึ้น โดยผู้ใช้งานไม่ต้องมีความรู้ด้านคอมพิวเตอร์มาก ก็สามารถใช้งานได้ เช่น โปรแกรมด้านการจัดพิมพ์เอกสาร โปรแกรมด้านฐานข้อมูล โปรแกรมด้านการคำนวณ และโปรแกรมด้านรูปภาพ เป็นต้น

3. บุคลากร (people ware) หมายถึง ผู้ที่ปฏิบัติงานตามกระบวนวิธีการในกิจกรรมต่าง ๆ เช่น การสร้างหรือเก็บรวบรวมข้อมูล การพัฒนาซอฟต์แวร์ขึ้นมาใหม่ ๆ การสร้างระบบงานคอมพิวเตอร์ ตำแหน่งและหน้าที่ที่เกี่ยวข้องกับคอมพิวเตอร์ มีดังนี้

3.1 พนักงานเตรียมข้อมูล (data entry operator) ทำหน้าที่บันทึกข้อมูลลงในอุปกรณ์ต่าง ๆ ที่คอมพิวเตอร์สามารถรับเข้าไปทำงานได้

3.2 พนักงานควบคุมเครื่องคอมพิวเตอร์ (computer operator) ทำหน้าที่นำโปรแกรมและข้อมูลเข้าเครื่องคอมพิวเตอร์ เตรียมอุปกรณ์ที่จะใช้กับงานต่าง ๆ

3.3 บรรณารักษ์คอมพิวเตอร์ (computer librarian) ทำหน้าที่ดูแลรักษาอุปกรณ์ที่ใช้บันทึกข้อมูล โปรแกรมคอมพิวเตอร์ หนังสือและเอกสารต่าง ๆ เหมือนกับบรรณารักษ์ห้องสมุดทั่วไป

3.4 นักโปรแกรมคอมพิวเตอร์ (programmer) ทำหน้าที่เขียนโปรแกรมเพื่อให้คอมพิวเตอร์ทำงานด้านต่าง ๆ

3.5 นักวิเคราะห์ระบบ (system analyst) ทำหน้าที่วิเคราะห์และออกแบบระบบงานคอมพิวเตอร์ เช่น ระบบบัญชี ระบบงานฝ่ายบุคคล เป็นต้น

4. กระบวนวิธีการ (procedure) หมายถึง ขั้นตอนที่จะบอกเครื่องคอมพิวเตอร์ว่าจะต้องทำอะไร และกระทำกับข้อมูลที่ได้รับมาอย่างไร และสุดท้ายคือผลลัพธ์ที่ต้องการคืออะไร

5. ข้อมูล (data) หมายถึง ข้อเท็จจริง เกี่ยวกับเรื่องต่าง ๆ ที่เราให้ความสนใจ อาจอยู่ในรูปของ ตัวเลข ตัวหนังสือ หรือ รูปภาพ เช่น รหัสนักศึกษา ชื่อ นามสกุล ฯลฯ หรือเป็นรายละเอียดความจริงที่วัดจากการทดลองโดยตรง ซึ่งสิ่งเหล่านี้ล้วนเป็นข้อมูลทั้งสิ้น

องค์ประกอบพื้นฐานของระบบงานคอมพิวเตอร์ เป็นเรื่องจำเป็นสำหรับผู้ที่ต้องการที่จะใช้เครื่องคอมพิวเตอร์ ในการทำงานที่จะต้องเตรียมองค์ประกอบต่าง ๆ ให้ครบ องค์ประกอบที่สำคัญที่สุดก็ คือ บุคลากรทางคอมพิวเตอร์ ที่ต้องใช้เวลาในการพัฒนา มาก ดังนั้นจึงเป็นเรื่องที่หน่วยงานหรือ องค์กรต่าง ๆ จะต้องมีแผนงานในการพัฒนาบุคลากรทางด้านนี้ เพื่อรองรับสถานการณ์ในอนาคตที่ จะมีการใช้งานคอมพิวเตอร์มากขึ้น

ภาษาคอมพิวเตอร์

ภาษาคอมพิวเตอร์ หมายถึง ภาษาที่เรานำไปสั่งงานให้คอมพิวเตอร์ทำงานตามที่ต้องการ ภาษาที่ทำให้คอมพิวเตอร์ทำงานได้ ได้แก่ ภาษาเครื่อง แต่สำหรับคนเราเป็นการยาก ที่จะเรียนรู้ภาษาเครื่อง ภาษาคอมพิวเตอร์ จึงมีการพัฒนาขึ้นมาให้สามารถใช้งาน ได้ง่าย และ สะดวก จนถึงยุค ปัจจุบัน ภาษาคอมพิวเตอร์พัฒนาไปจนถึงภาษาระดับสูง ซึ่งเป็นภาษาที่ คนเราสามารถเข้าใจได้ง่าย จึงทำให้มีความสะดวกในการสั่งงาน จึงทำให้มีการใช้งานคอมพิวเตอร์อย่างแพร่หลาย

ธงชัย สิทธิกรณ์ (2540, หน้า 115 – 128) กล่าวว่า ภาษาคอมพิวเตอร์ หมายถึง ภาษาที่ใช้เขียน คำสั่ง (command) เพื่อนำไปประกอบเป็นชุดคำสั่ง (program) ให้เครื่องคอมพิวเตอร์ทำงาน ดังนั้น เมื่อต้องการสั่งให้คอมพิวเตอร์ทำงานด้วยภาษา จึงจำเป็นที่จะต้องเรียนรู้ถึงพื้นฐานของ

ภาษาคอมพิวเตอร์ว่ามีสิ่งจำเป็น มีข้อจำกัด และมีประโยชน์อย่างไรบ้าง ภาษาคอมพิวเตอร์ แบ่งตามลักษณะของภาษาได้ดังนี้

1. ภาษาเครื่อง (machine language) หมายถึง ภาษาที่สามารถติดต่อกับ ฮาร์ดแวร์ ของเครื่องคอมพิวเตอร์ได้โดยตรง เป็นภาษาที่ทำให้ระบบคอมพิวเตอร์สามารถทำงานตามคำสั่งได้ทันที อยู่ในรูปของเลขฐานสอง ("0" หรือ "1") เท่านั้น

2. ภาษาสัญลักษณ์ (assembly / symbolic language)หมายถึง ภาษาที่พัฒนามาจากภาษาเครื่อง เพื่อให้สะดวกและเข้าใจง่ายสำหรับผู้ใช้เครื่อง มีการใช้ รหัสช่วยจำ (mnemonic code) แทนคำในการทำงาน และใช้ชื่อ สัญลักษณ์ (symbolic name) หรือชื่อตัวแปรแทนตำแหน่งของส่วนความจำหลัก (address) ภาษานี้อยู่ในรูปของเลขฐานสอง เลขฐานแปด เลขฐานสิบหก หรือเลขฐานสิบ ทำให้สามารถสื่อความหมายกับมนุษย์ได้มากขึ้น ภาษาสัญลักษณ์นี้ ในปัจจุบันนิยมใช้กับงานควบคุม โดยเฉพาะงาน ควบคุมในโรงงานอุตสาหกรรม งานควบคุมสัญญาณไฟจราจร หรืองานด้านการศึกษาภาษาเครื่อง และการศึกษาโครงสร้างทาง ฮาร์ดแวร์เป็นต้น

3. ภาษาระดับสูง (high level language) หมายถึง ภาษาที่สามารถสื่อความหมายกับมนุษย์ได้โดยตรง สามารถจดจำหรือเข้าใจได้ง่าย เนื่องจากเป็นภาษาที่มีลักษณะใกล้เคียงกับ ภาษามนุษย์ ในที่นี้หมายถึง ภาษาอังกฤษ และ ใช้เลขเลขฐานสิบ เป็นหลัก ภาษาระดับสูงเป็นภาษาที่ไม่ขึ้นกับเครื่องคอมพิวเตอร์เครื่องใดเครื่องหนึ่ง ในการเขียนโปรแกรมด้วยภาษาระดับสูง ผู้เขียนไม่จำเป็นต้องทราบถึงการทำงานของอุปกรณ์ภายในเช่น การทำงานของ ซีพียู หรือไม่จำเป็นต้องรู้ระบบเลขฐานสอง ภาษาระดับสูงที่ใช้สำหรับเขียนคำสั่ง มีหลายภาษาดังนี้

3.1 ภาษาเบสิก (BASIC : beginners all-purpose symbolic instruction code) คือภาษาสารพัดประโยชน์ ที่ใช้ทั้งในงานธุรกิจและอื่น ๆ พัฒนาขึ้นใช้โดยสถาบันมาตรฐานแห่งชาติของสหรัฐอเมริกา (American National Standards Institute : ANSI) นิยมใช้กับงานที่มีการโต้ตอบกับผู้ใช้แบบทันทีทันใด แฟ้มข้อมูลที่เกิดจากการเขียนโปรแกรมโดยใช้คำสั่งภาษานี้ จะเป็นชนิดที่มีสกุลเป็น BAS

3.2 ภาษาฟอร์แทรน (Fortran: formula translator) คือภาษาที่พัฒนาขึ้นโดยบริษัทเมื่อปี พ.ศ. 2500 นิยมใช้แก้ปัญหาทางวิทยาศาสตร์ เดิมถูกใช้งานเพื่อให้เครื่องคอมพิวเตอร์ทำงานแบบกลุ่ม แต่ปัจจุบัน นิยมใช้กันน้อยลงมาก

3.3 ภาษาโคบอล (Cobol: common business oriented language) คือภาษาที่เน้นใช้ในงานธุรกิจ พัฒนาขึ้นในสหรัฐอเมริกา เมื่อปี พ.ศ. 2502 และในปี พ.ศ. 2511 สถาบันมาตรฐานแห่งชาติสหรัฐอเมริกา ได้ร่วมมือกับบริษัทผู้ผลิตเครื่องคอมพิวเตอร์หลายบริษัทพัฒนา Ansi Cobol

(American National Standard Cobol) หลังจากนั้นในปี พ.ศ. 2517 จึงได้พัฒนา รุ่นใหม่ออกใช้งานอีก ปัจจุบันยังมีการใช้งานอยู่บ้าง ส่วนใหญ่ใช้กับเครื่องขนาดใหญ่

3.4 ภาษาปาสคาล (Pascal) เป็นภาษาสารพัดประโยชน์ภาษาหนึ่ง สามารถทำงานในลักษณะโต้ตอบได้ดี นิยมใช้ในงานธุรกิจและงานคำนวณทางวิทยาศาสตร์ นอกจากนั้น ยังเป็นภาษาที่เรียนรู้ได้ง่าย และสามารถเขียนเป็นโปรแกรมแบบโครงสร้างได้ ใช้งานกับเครื่องระดับมินิ และไมโครคอมพิวเตอร์

3.5 ภาษาซี (C) คือภาษาที่ได้รับความนิยมมากภาษาหนึ่ง มีลักษณะเป็นโปรแกรมโครงสร้าง (structure program) สามารถเข้าถึง ฮาร์ดแวร์ ของระบบได้ง่ายกว่าภาษาอื่น เนื่องจากมีคำสั่งที่สามารถเข้าถึงในระดับ บิต และ ไบต์ ของเครื่องคอมพิวเตอร์ได้โดยตรง ภาษาซีมีความสามารถคล้ายภาษาสัญลักษณ์แต่เขียนง่ายกว่า และมีความเร็วในการทำงานสูง ภาษาซีได้ถูกนำไปใช้ในการเขียนโปรแกรมสำเร็จรูปหลายโปรแกรม เช่น Microsoft Excel, CU Writer เป็นต้น

ภาษาคอมพิวเตอร์ ได้รับการพัฒนาให้ สามารถใช้งานได้ง่าย สะดวก และมีความรวดเร็ว จนถึงยุคปัจจุบัน ภาษาคอมพิวเตอร์ ก็ยังมีหลากหลายภาษา แต่ละภาษาจะมีลักษณะเด่นเฉพาะตัว ดังกล่าวมาแล้ว ดังนั้นการพิจารณาที่จะใช้ภาษาใดต้องพิจารณา ถึงคุณสมบัติของภาษานั้น ๆ ก่อน ว่าเหมาะสมที่จะนำมาใช้งานของเราหรือไม่

โปรแกรมและหลักการพัฒนาโปรแกรม

การพัฒนาโปรแกรมคอมพิวเตอร์ หรือ การเขียนโปรแกรมคอมพิวเตอร์ ขึ้นมาเพื่อใช้งานแทนมนุษย์เรานั้น เราต้องนำเอากระบวนการและวามคิดต่าง ๆ ใส่ลงไปในโปรแกรม ให้เกิดการทำงานเหมือนมนุษย์ทำเอง เครื่องคอมพิวเตอร์ เป็นเพียงเครื่องมือที่ คอยปฏิบัติตามคำสั่ง จะถูกหรือ ผิด เครื่องคอมพิวเตอร์ จะไม่สามารถรับรู้ได้ เนื่องจาก คอมพิวเตอร์ ไม่มีความคิด แต่เครื่องคอมพิวเตอร์ มีความจำที่ดีเลิศ เป็นระบบระเบียบ สามารถเรียกใช้ได้โดยเร็ว และเชื่อถือได้ในด้านคำตอบ ถ้าหากข้อมูลที่อยู่ในหน่วยความจำนั้นเป็นข้อมูล ที่ถูกต้อง ดังนั้นการพัฒนาโปรแกรมคอมพิวเตอร์ จึงเป็นเรื่องยากที่ เราจะทำอย่างไร ให้สามารถนำเอาความคิด และ กระบวนการต่าง ๆ ในด้านการคิด และ การประมวลผล ใส่ลงไปในโปรแกรม อย่างครบถ้วน การพัฒนาโปรแกรม หรือ การเขียนโปรแกรม จึงจำเป็นต้องมีขั้นตอนในการพัฒนา เพื่อให้ใส่องค์ประกอบดังกล่าวลงไปอย่างครบถ้วนเพื่อให้โปรแกรมนั้น ๆ สามารถทำงานได้อย่างถูกต้องและมีประสิทธิภาพเหมือนกับที่คนทำเอง หลักในการพัฒนาโปรแกรม อยู่ 5 ขั้นตอนด้วยกันคือ

1. การวิเคราะห์ปัญหา (analysis the problem)
2. การออกแบบโปรแกรม (design a program)

3. การลงรหัส (coding)

4. การทดสอบโปรแกรม (testing)

5. การทำเอกสารประกอบโปรแกรม (documentation)

1. การวิเคราะห์ปัญหา คือขบวนการแรกที่มีความสำคัญที่สุดในการพัฒนาโปรแกรม เป็นขั้นตอนการศึกษาลักษณะ และรายละเอียดของปัญหา หรืองานที่ต้องทำการวิเคราะห์ระบบงาน หรือ วิเคราะห์ปัญหา เพื่อที่จะนำเอาโปรแกรมคอมพิวเตอร์เข้ามาใช้งาน มีหลักเกณฑ์ที่สำคัญ ดังนี้

1.1 วิเคราะห์รูปแบบผลลัพธ์ หมายถึง การศึกษาถึงรูปแบบและข้อมูลผลลัพธ์ที่ต้องการให้เครื่องแสดงว่า ผลลัพธ์ของโปรแกรมจะออกมาอย่างไร อยู่ในรูปแบบใด และจะต้องใช้อุปกรณ์ในการแสดงผลลัพธ์อะไรบ้าง เช่น เครื่องพิมพ์ หรือ จอภาพ เป็นต้น ผลลัพธ์ของโปรแกรมถือว่าเป็น ผลผลิตของการเขียนโปรแกรม ดังนั้น ผลลัพธ์ที่ถูกต้อง และ ได้ตามจุดประสงค์ จึงเป็นเรื่องสำคัญที่สุดในกระบวนการพัฒนาโปรแกรม

1.2 วิเคราะห์ข้อมูลนำเข้า หมายถึง การคิดวิเคราะห์ว่า จะต้องนำข้อมูลอะไรให้กับเครื่องคอมพิวเตอร์ ให้น้อยที่สุด เพื่อที่จะทำให้โปรแกรมนั้นเกิดความง่ายและสะดวกในการใช้งาน และผลลัพธ์ก็ต้องออกมาอย่างถูกต้องและเชื่อถือได้ ตามลักษณะงานที่ได้รับมา ตลอดจนวิเคราะห์ไปถึงว่า จะนำเข้าโดยวิธีการใด และใช้อุปกรณ์ในการนำเข้าข้อมูลอะไรบ้าง เช่น ใช้แป้นพิมพ์ ใช้เครื่องอ่านแถบบาร์โค้ด ฯลฯ เป็นต้น

1.3 วิเคราะห์วิธีการประมวลผล หมายถึง การวิเคราะห์ว่า การที่จะได้ผลลัพธ์ออกไปในแบบนั้น จะต้องมีการประมวลผลเกี่ยวกับเรื่องใดบ้าง ตามข้อมูลที่ได้รับเข้ามาจากการรับข้อมูล ค่าต่างๆ ที่ได้จากการประมวลผลนั้น เป็นค่าที่เราต้องนำไปใช้งานทั้งสิ้น การแก้ปัญหาโจทย์จำเป็นต้องมีขั้นตอน ดังนั้นการวิเคราะห์การประมวลผล ก็จะมีเป้าหมายหลักอยู่ที่ผลลัพธ์ของโปรแกรม ว่าจะประมวลผลอย่างไร เพื่อให้ได้ผลลัพธ์ตามที่ต้องการภายใต้เงื่อนไข ของข้อมูลที่ได้รับมา

1.4 กำหนดตัวแปรที่จะใช้งาน หมายถึง การสั่งให้เครื่องจดจำค่าต่าง ๆ ที่เกี่ยวข้องกับการทำงานของ โปรแกรมคอมพิวเตอร์ โดยทั่วไปโปรแกรมจะต้องมีการประมวลผลหลายขั้นตอน ข้อมูลที่ปรากฏขึ้นมาระหว่างการทำงานของโปรแกรม ก็เป็นเรื่องจำเป็นที่เครื่องคอมพิวเตอร์จะต้องจดจำค่าต่าง ๆ ที่เกิดขึ้น เพื่อให้เกิดความสะดวกในการเขียนโปรแกรม ข้อมูลทุกตัวที่เข้าไปอยู่ในหน่วยความจำของคอมพิวเตอร์ เราจำเป็นที่จะตั้งชื่อให้กับข้อมูลเหล่านั้นเพื่อประโยชน์ในการเรียกกลับเข้ามาใช้ใหม่ในโปรแกรม ชื่อของข้อมูลเหล่านั้นเราเรียกว่าตัวแปร เช่น ให้ W เป็นตัวแปรที่หมายถึง ความกว้างของรูปสี่เหลี่ยม ให้ L เป็นตัวแปรที่หมายถึงความยาวของรูปสี่เหลี่ยม ดังนั้นถ้าต้องการทราบ พื้นที่ของรูป สี่เหลี่ยม ก็ให้นำเอา W คูณกับ L ก็จะได้พื้นที่ของ

รูปสี่เหลี่ยมดังกล่าว การกำหนดตัวแปรจะต้องระบุด้วยว่า ข้อมูลที่จะเก็บในตัวแปรแต่ละตัวเป็นข้อมูลชนิดใด เช่น ตัวอักษร หรือ ตัวเลข เป็นต้น

2. การออกแบบโปรแกรม คือ กระบวนการที่นำรายละเอียดจากการวิเคราะห์ระบบงานจากขั้นตอนที่ 1 มาเขียนเป็นขั้นตอน อย่างละเอียด ว่าอะไรมาก่อน อะไรมาทีหลัง ซึ่งตามปกติแล้วโปรแกรมคอมพิวเตอร์ จะมีขั้นตอนเหมือนกับกระบวนการในการผลิต สินค้าของโรงงานอุตสาหกรรม ซึ่งประกอบด้วยกระบวนการรับข้อมูลเข้า กระบวนการประมวลผล และ สุดท้ายคือกระบวนการแสดงผลลัพธ์ การเขียนโปรแกรมนั้น จะต้องมีขั้นตอนอยู่หลายขั้นตอน กว่าที่จะได้มาซึ่งผลลัพธ์ของโปรแกรม เพื่อให้ง่ายในการเรียบเรียงลำดับขั้นตอนของการออกคำสั่งอย่างถูกต้อง จึงต้องมีการใช้เครื่องมือในการ ออกแบบ เครื่องมือที่นิยมใช้กันดังนี้

2.1 อัลกอริทึม (algorithm) เป็นการใช้ข้อความที่เป็นภาษาพูดในการอธิบายการทำงานของโปรแกรมอย่างเป็นลำดับขั้น ตามความถนัดภาษาของผู้เขียนโปรแกรม เช่น การเขียนโปรแกรมเพื่อรวมคะแนนสอบของนักศึกษาที่มีการสอบ 2 ครั้งใน 1 ภาคเรียน เขียนเป็น อัลกอริทึมได้ดังนี้

<u>อัลกอริทึมการรวมคะแนนสอบของนักศึกษา</u>

1. กำหนดค่าเริ่มต้นให้กับตัวแปร
2. รับข้อมูลรหัสนักศึกษาจากแป้นพิมพ์
3. รับข้อมูลชื่อนักศึกษาจากแป้นพิมพ์
4. รับข้อมูลคะแนนสอบครั้งที่ 1 จากแป้นพิมพ์
5. รับข้อมูลคะแนนสอบครั้งที่ 2 จากแป้นพิมพ์
6. คำนวณหาคะแนนรวมจากคะแนนครั้งที่ 1 รวมกับ คะแนนครั้งที่ 2
7. แสดงผลรหัสนักศึกษา
8. แสดงผลชื่อนักศึกษา
9. แสดงผลคะแนนสอบครั้งที่ 1
10. แสดงผลคะแนนสอบครั้งที่ 2
11. แสดงผลคะแนนรวม
12. จบ

2.2 ผังงาน (flowchart) หมายถึง สัญลักษณ์หรือรูปภาพต่าง ๆ ที่ใช้แทนข้อความหรือ คำพูด ที่ใช้ในวิธีการประมวลผล งานทุกชนิดที่ได้ผ่านการวิเคราะห์และกำหนดลำดับขั้นตอน

แล้ว เราสามารถนำมาเขียนเป็นผังงานได้ ไม่ว่าจะเป็นงานในชีวิตประจำวันทั่วไป หรืองานที่ต้องใช้คอมพิวเตอร์ประมวลผลก็ตาม อย่างไรก็ตาม การเขียนผังงานนิยมใช้การแสดงขั้นตอนในการแก้ปัญหาโดยคอมพิวเตอร์ เพื่อให้การเขียนโปรแกรมง่ายขึ้นและขจัดข้อผิดพลาดของงาน เนื่องจากคอมพิวเตอร์จะทำงานตามขั้นตอนซึ่งผู้เขียนโปรแกรมเป็นผู้กำหนดให้ ฉะนั้นผังงานก็ช่วยให้ผู้เขียนโปรแกรมมองเป็นขั้นตอนต่าง ๆ จากสัญลักษณ์ที่กำหนดไว้ เพื่อให้ง่าย และไม่เกิดความสับสน โดยเฉพาะปัญหาที่ซับซ้อนมาก ๆ ผังงานแบ่งเป็น 2 ประเภทด้วยกันคือ

2.2.1 ผังงานระบบ (system flowchart) เป็นผังงานที่แสดงให้เห็นถึงขั้นตอนของการทำงานภายในระบบนั้น ๆ โดยแสดงให้เห็นถึงความเกี่ยวข้องของส่วนต่าง ๆ ที่สำคัญในระบบการประมวลผลข้อมูลนั้น เช่น บุคลากร อุปกรณ์ วัสดุ โปรแกรม เป็นต้น โดยผังงานระบบจะแสดง ขั้นตอนตั้งแต่เริ่มต้น ว่ามีเอกสารเบื้องต้นจากส่วนใดของระบบงาน แล้วผ่านไปยังหน่วยงานใด มีกิจกรรมอะไรในหน่วยงานนั้น มีการส่งต่องานไปยังหน่วยงานใด จนกระทั่งงานเสร็จสิ้น ดังนั้นบางส่วนจะเกี่ยวข้องกับคน ส่วนที่เป็นวิธีการต้องใช้เครื่องคอมพิวเตอร์เท่านั้นที่เราจะนำมาแยกเขียนเป็นโปรแกรม โดยแสดงรายละเอียดการทำงานแยกออกมาเขียนแผนผังงานโปรแกรม สำหรับส่วนนั้น ๆ อีกทีหนึ่ง

2.2.2 ผังงานโปรแกรม (program flowchart) หมายถึง ผังงานที่แสดงลำดับขั้นตอนการทำงานในโปรแกรม โดยละเอียดว่าทำอะไร และ ทำอย่างไร ดังนั้นจึงมีส่วนแสดงการทำงานในขั้นตอนการรับข้อมูล การประมวลผล และ การแสดงผลลัพธ์ ผังงานโปรแกรมมักเรียกสั้น ๆ ว่า ผังงาน (flowchart)

การเขียนผังงานมีรายละเอียดอีกมากมาย และเป็นเครื่องมือที่จะใช้ประกอบกับเอกสารนี้ จึงขอกล่าวถึงรายละเอียดของการเขียนผังงานในหัวข้อต่อไป

2.3 รหัสจำลอง (pseudo code) เป็นการใช้ข้อความที่เป็นภาษาอังกฤษหรือภาษาไทยก็ได้ ในการแสดงขั้นตอนการแก้ปัญหา แต่จะมีการใช้คำเฉพาะ (reserve words) ที่มีอยู่ในภาษาโปรแกรมมาช่วยเขียน เช่น รหัสจำลองของการเขียน โปรแกรมเพื่อรวมคะแนนสอบของนักศึกษาที่มีการสอบ 2 ครั้งใน 1 ภาคเรียน เขียนเป็นรหัสจำลองได้ดังนี้

```
Initial value for test-1 test-2 and total score
input student-code
input student-name
input test-1 score
input test-2 score
```

compute total-score with test-1 + test-2

display student-code

display student-name

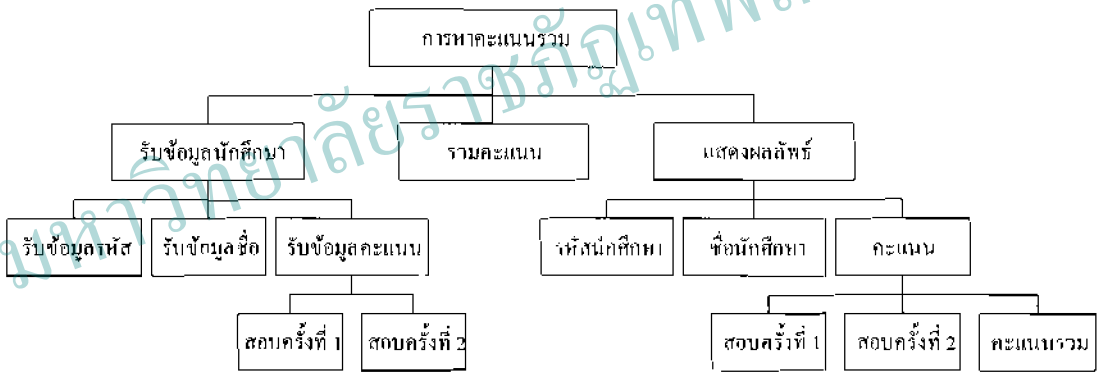
display test-1 score

display test-2 score

display total score

end

2.4 แผนภูมิโครงสร้าง (structure chart) เป็นการแบ่งงานใหญ่ออกเป็นส่วน ๆ ตามหน้าที่ของแต่ละส่วน แต่ละส่วนเรียกว่า โมดูล (module) การออกแบบด้วยวิธีนี้จะเริ่มต้นจากส่วนที่ใหญ่ที่สุดไปหาส่วนที่รองลงไป ซึ่งเรียกว่า การออกแบบจากบนลงล่าง (top-down design) เช่น การเขียนโปรแกรมเพื่อรวมคะแนนสอบของนักศึกษาที่มีการสอบ 2 ครั้งใน 1 ภาคเรียน แสดงได้ดังภาพที่ 1.2



ภาพที่ 1.2 แผนภูมิโครงสร้างของโปรแกรมรวมคะแนนนักศึกษา

3. การลงรหัส หมายถึง การนำเอาภาษาคอมพิวเตอร์ภาษาใดภาษาหนึ่งมาเขียน เพื่อให้ได้ความหมายตามขั้นตอนที่ได้ลำดับไว้ในขั้นตอนที่ 2 หรือเรียกว่า การเขียนโปรแกรม ในการเขียนโปรแกรมผู้เขียนจำเป็นต้องมีความรู้เกี่ยวกับภาษาคอมพิวเตอร์ ที่จะใช้เขียนในระดับดี ต้องทราบถึงโครงสร้างของตัวโปรแกรม คำสั่งประเภทที่ใช้รับข้อมูล คำสั่งในการประมวลผล คำสั่งในการแสดงผล และคำสั่งในการควบคุมการทำงานอื่น ๆ ในการเขียนโปรแกรมของเอกสารนี้ จะใช้ภาษาซีพลัสพลัส เป็นภาษาที่ใช้เขียนโปรแกรม รายละเอียดต่าง ๆ จะกล่าวในบทต่อ ๆ ไป ตัวอย่างการลงรหัสเป็นภาษาซีพลัสพลัส เช่นการเขียนโปรแกรมเพื่อรวมคะแนนสอบของนักศึกษาที่มีการสอบ 2 ครั้งใน 1 ภาคเรียน เขียนเป็นโปรแกรมได้ดังนี้

```
1   // example 1
2   #include <iostream.h>
3   #include <conio.h>
4   int main()
5   {  char code[9],name[20];
6      int test1,test2,total;
7      clrscr();
8      cin>>code;
9      cin>>name;
10     cin>>test1;
11     cin>>test2;
12       total = test1 + test2;
13     cout<<code <<'\t'<<name <<'\t'
14           <<test1<<'\t'<<test2<<'\t'<<total;
15     getch();
16     return 0;
17  }
```


โปรแกรมในภาษาซี ไม่มีเลขบรรทัด แต่ในเอกสารนี้ ใส่เลขบรรทัดลงไปเพื่อให้ผู้อ่านได้เห็นว่าคำสั่งที่เขียนไปในแต่ละคำสั่ง อยู่ในบรรทัดใด เพื่อความสะดวกในการอธิบายโปรแกรม โปรแกรมตัวอย่างนี้ เป็นตัวอย่างของการลงรหัส ดังนั้นผู้เขียนจึงขออธิบายการทำงานในบทต่อไป

4. ทดสอบและแก้ไขโปรแกรม คือ ขั้นตอนการตรวจสอบความถูกต้องของโปรแกรม ว่าถูกต้องตามจุดประสงค์ของการเขียนโปรแกรม และ นำไปใช้งานได้ตามจุดประสงค์หรือไม่ การตรวจสอบโปรแกรมแบ่งได้เป็น 2 ขั้นตอนได้แก่

4.1 การตรวจสอบความถูกต้องตามหลักไวยากรณ์ของภาษาคอมพิวเตอร์ ในขั้นตอนนี้ โปรแกรมแปลภาษา จะทำหน้าที่ตรวจสอบและทำการแปลภาษา ให้โปรแกรมสามารถทำงานได้ แต่ถ้าหาก โปรแกรมนั้นยังมีส่วนใดส่วนหนึ่งของโปรแกรม ที่ยังเขียนไม่ถูกต้องตามหลักของการเขียนโปรแกรมในภาษานั้น ๆ โปรแกรมแปลภาษาจะส่ง ข่าวสารการผิดพลาดออกมาเพื่อให้ผู้เขียนโปรแกรม นำโปรแกรมไปทำการแก้ไขให้ถูกต้อง

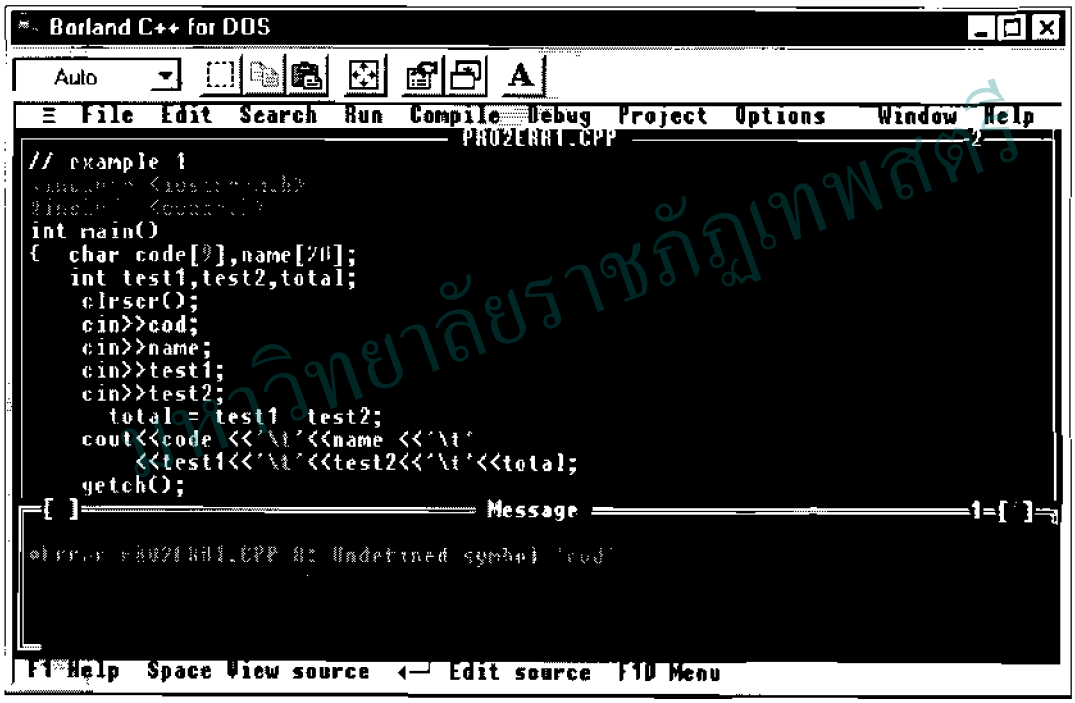
4.2 การตรวจสอบความถูกต้องของผลลัพธ์ เป็นขั้นตอนต่อจาก การแปลโปรแกรมจากภาษาระดับสูง ให้เป็นโปรแกรมภาษาเครื่อง นั่นหมายความว่า โปรแกรมไม่มีที่ผิดพลาดในด้านการเขียนภาษาแล้ว การตรวจสอบความถูกต้องของผลลัพธ์ ทำได้โดยการใช้ข้อมูลสมมุติ หรือ เรียกอีกอย่างหนึ่งว่า ข้อมูลเทียม ซึ่งหมายถึง ข้อมูลที่มีคำตอบของผลลัพธ์แล้ว เช่น เมื่อใส่ข้อมูลตามที่กำหนด โปรแกรมจะต้องได้ผลลัพธ์ถูกต้องตามที่ได้หาคำตอบไว้ล่วงหน้า ในการตรวจสอบผู้เขียนโปรแกรมต้องมีการเตรียมข้อมูลไว้ล่วงหน้า โดยที่ข้อมูลที่เตรียมไว้ จะต้องครอบคลุมถึง ความเป็นไปได้ ที่ผู้ใช้โปรแกรม จะป้อนลงไป เช่น ข้อมูลตามปกติ ข้อมูลที่มีค่าติดลบ ข้อมูลที่มีค่ามากเกินความจริง หรือข้อมูลที่ไม่ถูกต้องตามชนิดของการรับข้อมูล เช่น ป้อนข้อมูลตัวเลขลงชื่อคน หรือ ป้อนข้อมูลตัวอักษรให้กับเงินเดือนพนักงาน เป็นต้น ทั้งนี้เพื่อให้ทราบถึงความสามารถของโปรแกรม

การทดสอบโปรแกรมในขั้นตอนนี้หลังจากทดสอบแล้ว อาจเกิดความผิดพลาดของโปรแกรมขึ้น โดยเฉพาะผู้เขียนโปรแกรมที่เพิ่งจะหัดเขียน ถือเป็นเรื่องปกติของการเขียนโปรแกรมคอมพิวเตอร์ ความผิดพลาดที่เกิดขึ้นจากการทดสอบโปรแกรมแบ่งได้เป็น 3 ลักษณะดังนี้

1. ความผิดพลาดจากการใช้ไวยากรณ์ไม่ถูกต้อง (syntax error) หรือผิดพลาดจากการลงรหัสผิด ซึ่งโปรแกรมแปลภาษาสามารถตรวจสอบได้จากขั้นตอนการแปลให้เป็นภาษาเครื่องดังกล่าวแล้วในตอนต้น ดังนั้นผลลัพธ์ที่ได้จากการแปล จะเป็นไปได้อยู่ 2 สถานะคือ แปลแล้วไม่มีที่ผิดพลาด โปรแกรมจะแสดงข้อความในลักษณะที่ว่า ไม่มีข้อผิดพลาด และแปลแล้วพบข้อผิดพลาด ก็จะแสดงข้อผิดพลาดต่าง ๆ ที่เกิดขึ้นให้ผู้เขียนโปรแกรมได้รับทราบ เพื่อที่จะนำโปรแกรมไปแก้ไขต่อไป เช่น โปรแกรมต่อไปนี้ เมื่อทำการแปลแล้วจะพบข้อผิดพลาด ในบรรทัดที่ 8 และ บรรทัดที่ 12 ดังภาพที่ 1.3

```
1   // example 1
2   #include <iostream.h>
3   #include <conio.h>
4   int main()
5   { char code[9],name[20];
6     int test1,test2,total;
7       clrscr();
8       cin>>cod;
```

```
9       cin>>name;
10      cin>>test1;
11      cin>>test2;
12       total = test1  test2;
13      cout<<code <<'\t'<<name <<'\t'
14            <<test1<<'\t'<<test2<<'\t'<<total;
15      getch();
16     return 0;
17    }
```

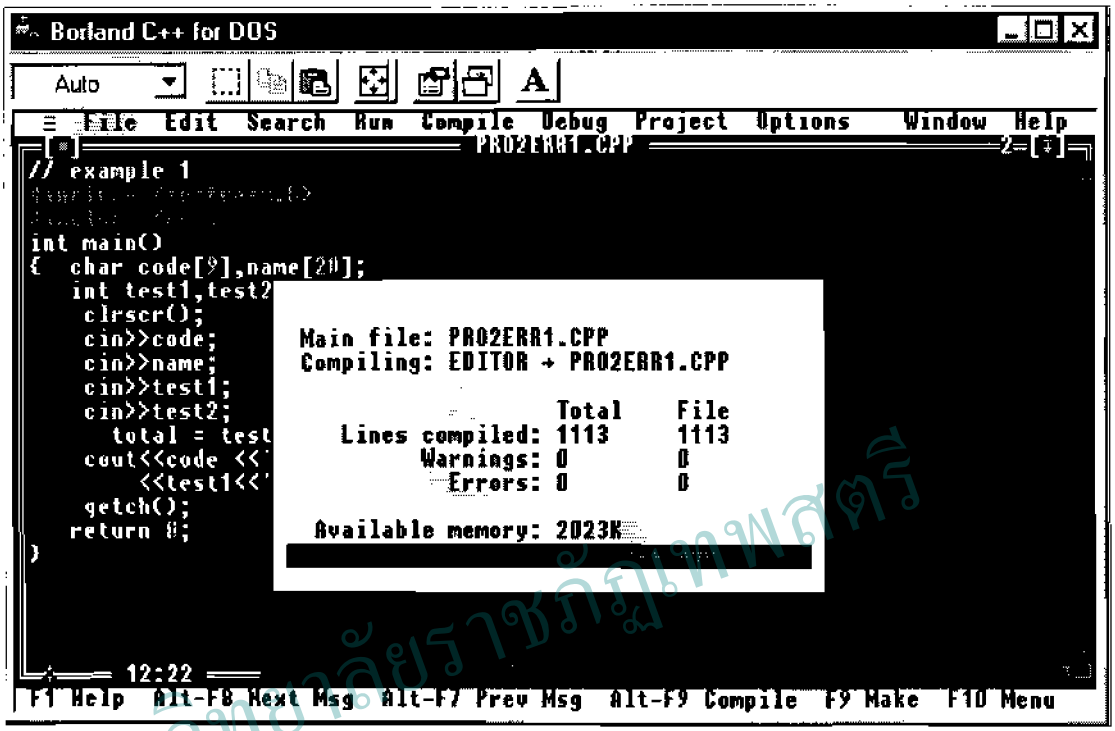


ภาพที่ 1.3 ข้อผิดพลาดของการเขียนคำสั่งผิดจากกฎเกณฑ์ของภาษา

จากภาพที่ 1.3 ข้อผิดพลาดเกิดขึ้นมีอยู่ 2 ที่ด้วยกันคือ

1. บรรทัดที่ 8 เกิดจาก ไม่ได้กำหนดสัญลักษณ์ หรือในที่นี้หมายถึงตัวแปร cod
2. บรรทัดที่ 12 เกิดจาก ประโยคคำสั่ง ไม่ถูกต้องในเรื่องของนิพจน์ทางคณิตศาสตร์

โปรแกรมนี้ ต้องทำการแก้ไขตามที่ โปรแกรมแปลภาษาแจ้งไว้ คือแก้ไขบรรทัดที่ 8 และ 12 ให้ถูกต้อง เมื่อทำการแก้ไขโปรแกรมในบรรทัดดังกล่าว ให้ถูกต้องแล้ว ทำการแปลโปรแกรมอีกครั้ง ผลลัพธ์ที่ได้แสดงดังภาพที่ 1.4



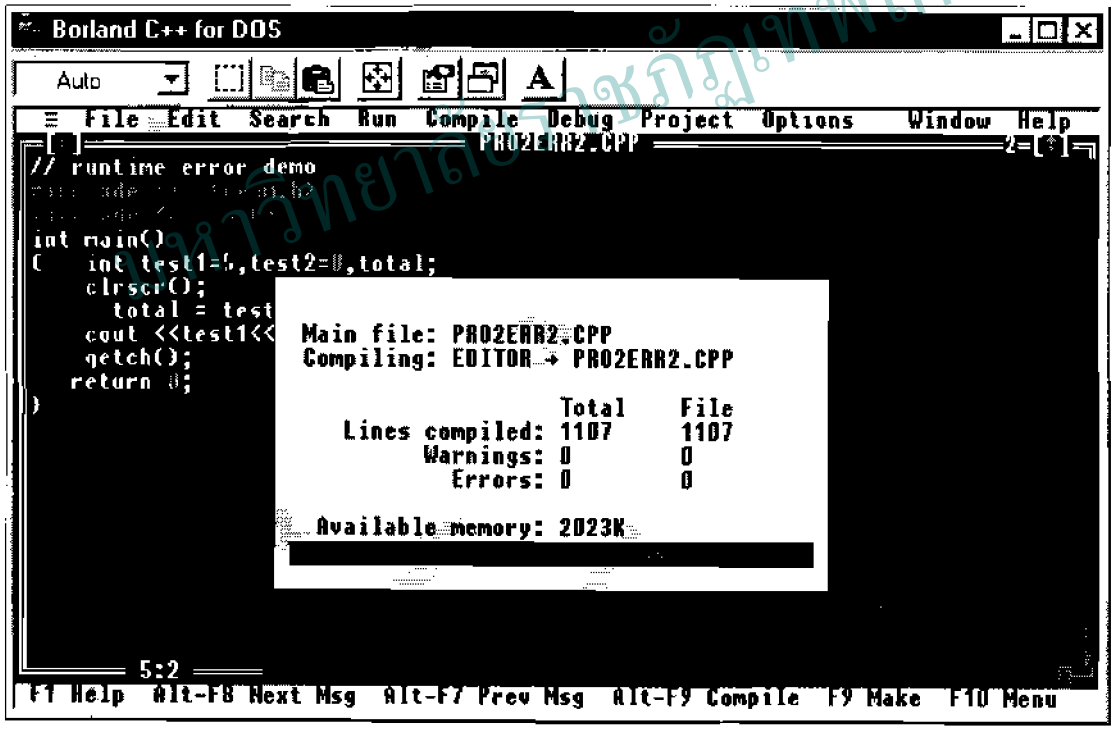
ภาพที่ 1.4 ผลลัพธ์ของการแปลโปรแกรมที่ไม่มีข้อผิดพลาด

2. ความผิดพลาดที่เกิดระหว่างโปรแกรมทำงาน (run-time error) ข้อผิดพลาดนี้เป็นข้อผิดพลาดที่เกิดจากการที่โปรแกรมทำงานแล้ว มีบางอย่างเกิดการผิดพลาดขึ้นมาระหว่างที่โปรแกรมทำงาน เช่น การคำนวณตัวเลขที่มีการหารด้วย ค่าศูนย์ทำให้เกิดค่า อนันต์ (infinity) โปรแกรมนั้นจะเกิดการผิดพลาดขณะที่โปรแกรมทำงาน และหยุดการทำงานเพื่อให้ผู้เขียน นำโปรแกรมไปแก้ไข ดังเช่นโปรแกรมต่อไปนี้

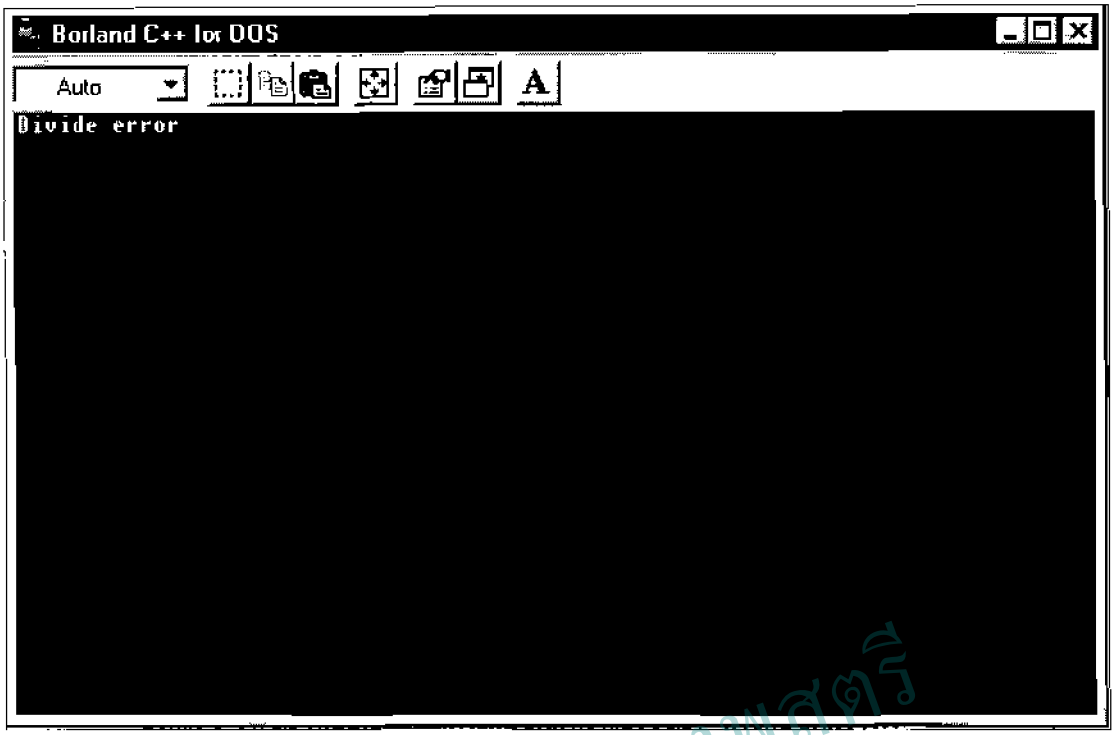
```
1   // runtime error demo
2   #include <iostream.h>
3   #include <conio.h>
4   int main()
```

```
5       {   int test1=5,test2=0,total;
6           clrscr();
7           total = test1 / test2;
8           cout <<test1<<'\t'<<test2<<'\t'<<total;
9           getch();
10          return 0;
11      }
```

เมื่อทำการแปลโปรแกรม ด้วยโปรแกรมแปลภาษา จะไม่พบข้อผิดพลาด ทั้งนี้เพราะว่าการเขียนคำสั่งถูกต้องตามกฎเกณฑ์ของภาษา โปรแกรมแปลภาษาจึงแจ้งข่าวสาร ว่าโปรแกรมไม่มีที่ผิดพลาดดังภาพที่ 1.5 แต่เมื่อสั่งให้โปรแกรมทำงาน โปรแกรมจะหยุดการทำงาน และแสดงข้อผิดพลาดดังภาพที่ 1.6



ภาพที่ 1.5 ผลลัพธ์การแปลโปรแกรมที่ไม่มีข้อผิดพลาด



ภาพที่ 1.6 ผลลัพธ์ของโปรแกรมที่ผิดพลาดขณะโปรแกรมทำงาน

3. ความผิดพลาดจากการกำหนดเงื่อนไข หรือ การประมวลผลไม่ถูกต้อง (logic error) หมายถึง การที่เราทดสอบด้วยข้อมูลสมมุติแล้ว ได้คำตอบไม่ตรงกับที่เป็นจริง ซึ่งอาจเกิดจากการกำหนดเงื่อนไขในการตัดสินใจที่ไม่ถูกต้อง หรือ อาจเกิดจาก การเขียนนิพจน์ทางคณิตศาสตร์ที่ไม่ถูกต้อง ความผิดพลาดแบบนี้ โปรแกรมแปลภาษาไม่สามารถตรวจสอบได้ ผู้เขียนโปรแกรม ต้องทำการทดลองด้วยข้อมูลสมมุติ หรือ ข้อมูลเทียม เพื่อตรวจคำตอบว่าได้ตามที่ต้องการหรือไม่ ตัวอย่างความผิดพลาดชนิดนี้เช่น สมมุติว่าการคำนวณค่าน้ำประปาแบบขั้นทวีคูณตาม เกณฑ์ ดังตารางที่ 1.2

ตารางที่ 1.2 เกณฑ์การคิดค่าน้ำประปา

หน่วยการใช้น้ำ	อัตราค่าน้ำ(บาท)
1-10	0.80
11-20	1.20
21-30	1.40
31ขึ้นไป	1.60

กำหนดให้ U หมายถึง หน่วยการใช้น้ำ

P หมายถึง ค่าน้ำ

ผู้ที่ใช้น้ำ 9 หน่วย อยู่ในช่วงการใช้น้ำ 1-10 หน่วย ค่าน้ำ จะเท่ากับ 9 หน่วยคูณกับอัตราค่าน้ำ คือ 0.80 บาท มีค่าเท่ากับ 7.20 บาท ทำเป็นสูตรการคำนวณด้วยคอมพิวเตอร์ได้ดังนี้

P = U * 0.8;

ผู้ที่ใช้น้ำ 12 หน่วย อยู่ในช่วงการใช้น้ำ 1-10 เท่ากับ 10 หน่วย และ 11-20 เท่ากับ 2 หน่วย ดังนั้นการคำนวณจึงทำได้ดังนี้ คือ 10 หน่วย คูณ 0.80 บาทเท่ากับ 8 บาท รวมกับ 2 คูณ 1.20 เท่ากับ 2.40 บาท ได้ค่าน้ำรวม 10.40บาท ทำเป็นสูตรการคำนวณด้วยคอมพิวเตอร์ได้ดังนี้

P – 10 * 0.8 + U – 10 * 1.20;

สูตรที่เขียนลงไป จะทำให้เครื่องคำนวณ เครื่องหมายคูณก่อน มีการคำนวณดังนี้คือ

ลำดับที่ 1. 10 * 0.8 เท่ากับ 8

ลำดับที่ 2. 10 * 1.20 เท่ากับ 12

ลำดับที่ 3. 8 + U หรือ 8 + 12 เท่ากับ 20

ลำดับที่ 4. 20 12 เท่ากับ 8

จากผลลัพธ์ของการเขียนสูตรดังกล่าวทำให้โปรแกรม ได้ผลลัพธ์เท่ากับ 8 เป็นผลลัพธ์ที่ไม่ถูกต้อง ลักษณะข้อผิดพลาดนี้เราเรียกว่า ความผิดพลาดเนื่องจากเงื่อนไข (logic error) สูตรคำนวณที่ถูกต้องในการคำนวณครั้งนี้คือ

P = 10 * 0.08 + (U – 10) *1.20;

จากสูตรดังกล่าว จะเกิดการคำนวณดังนี้คือ

ลำดับที่ 1. U – 10 เท่ากับ 2

ลำดับที่ 2. 10 * 0.8 เท่ากับ 8

ลำดับที่ 3. 2 * 1.20 เท่ากับ 2.40

ลำดับที่ 4. 8 + 2.40 เท่ากับ 10.40

5. จัดทำเอกสารเกี่ยวกับโปรแกรม (Program Documentation) คือ เอกสารที่ผู้พัฒนาโปรแกรมจัดทำขึ้น เพื่อให้เป็นคู่มือสำหรับผู้ใช้โปรแกรม เรียกว่า User's manual หรือ User's guide ส่วนประกอบที่สำคัญในการเขียน มีดังนี้

5.1 ชื่อโปรแกรม และชื่อผู้เขียนโปรแกรม

5.2 คุณสมบัติของโปรแกรม

5.3 คุณสมบัติขั้นต่ำของเครื่องที่ใช้ได้กับโปรแกรม

5.4 วิธีการติดตั้งลงสู่เครื่องคอมพิวเตอร์

5.5 ขั้นตอนการใช้งาน

การพัฒนาโปรแกรม คือการนำเอาเครื่องคอมพิวเตอร์ไปทำการแก้ปัญหาให้กับมนุษย์ เพื่อให้เกิดความรวดเร็วและ ถูกต้อง โปรแกรมที่มีการใช้ข้อมูลมาก ย่อมมีความสลับซับซ้อน การออกแบบและพัฒนาโปรแกรม ด้วยหลักการที่ถูกต้อง จึงเป็นเรื่องจำเป็น ในการที่จะได้โปรแกรมที่มีประสิทธิภาพ

หลักเกณฑ์ทั่ว ๆ ไป ในการเขียนผังงาน

ดังกล่าวแล้วว่า ผังงานเป็นเครื่องมือในการออกแบบโปรแกรมที่มีการใช้ สัญลักษณ์ แทนการทำงานของเครื่องคอมพิวเตอร์ บอกลำดับการทำงานของเครื่องคอมพิวเตอร์ จากขั้นตอนหนึ่งไปขั้นตอนหนึ่ง โดยใช้เส้นแสดงการเดินทาง จากลำดับหนึ่ง ไปอีกลำดับหนึ่ง ของการทำงาน ในการออกแบบโปรแกรมคอมพิวเตอร์ ที่มีความสลับซับซ้อน เครื่องมือที่ดีที่สุดอีกตัวหนึ่งก็คือ ผังงาน ลำดับขั้นตอนของผังงาน จึงเหมือนกับ ลำดับขั้นตอนของโปรแกรม มีขั้นตอนใหญ่ในการดำเนินการของโปรแกรมดังนี้

1. กำหนดค่าเริ่มต้น (initialization)เป็นการกำหนดค่าเริ่มต้นให้กับตัวแปรบางตัว เช่น ตัวแปรที่ทำหน้าที่เป็นหน่วยนับ ตัวแปรที่ค่าเป็นผลการคำนวณสะสม เช่น การบวกสะสม หรือการคูณสะสม เป็นต้น

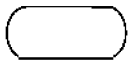
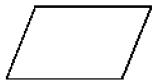
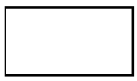
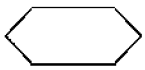
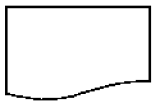
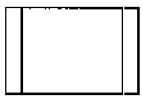
2. การรับข้อมูล (input) เป็นการรับค่าตัวแปรที่ระบุไว้ในขั้นตอนการนำเข้าข้อมูลของการวิเคราะห์งาน การรับข้อมูลจะต้องกระทำก่อนจะมีการนำข้อมูลนั้นไปใช้

3. การประมวลผล (process) เป็นการแสดงวิธีประมวลผล หรือการคำนวณซึ่งจะต้องกระทำทีละขั้นตอนตามลำดับ ถ้าผลการคำนวณต้องนำไปใช้ในขั้นตอนต่อไป จะต้องแยกสัญลักษณ์ให้เห็นชัด

4. การแสดงผลลัพธ์ (output) เป็นการแสดงผลลัพธ์หรือค่าของตัวแปรที่ระบุไว้ในหัวข้อผลลัพธ์ที่ต้องแสดงในการวิเคราะห์ การแสดงค่าของตัวแปรหรือผลลัพธ์ต้องกระทำหลังจากการประมวลผล หรือ มีข้อมูลนั้นในหน่วยความจำแล้ว

ผังงานเป็นการบอก ลำดับการทำงานของเครื่องคอมพิวเตอร์ ด้วยสัญลักษณ์มี สัญลักษณ์ต่าง ๆ ที่ใช้เป็นผังงาน บอกลักษณะ การทำงานต่าง ๆ ของเครื่องคอมพิวเตอร์ ที่นิยมใช้แสดงในตารางที่ 1.3

ตารางที่ 1.3 สัญลักษณ์ที่นิยมใช้ในผังงาน

	จุดเริ่ม/จุดจบ (terminal, interrupt)แสดงการเริ่มต้นหรือการสิ้นสุดของการเขียนผังงาน
	การรับข้อมูล (input) แสดงการรับข้อมูลเข้า
	การประมวลผล (process) แสดงการประมวลผล
	เปรียบเทียบ (decision) แสดงการเปรียบเทียบหรือการตัดสินใจเลือกข้อมูล
	กำหนดค่าล่วงหน้า (preparation) แสดงการกำหนดค่าล่วงหน้าซึ่งเป็นการทำงานในช่วงใดช่วงหนึ่งที่ซ้ำ ๆ กัน
	การแสดงผลในรูปแบบของเอกสาร (document)
	การแสดงผลในทางจอภาพ (display)
	ตัวเชื่อมต่อ (link) ใช้สำหรับเชื่อมต่อจากจุดหนึ่งไปยังอีกจุดหนึ่ง
	กระบวนการที่กำหนดไว้ล่วงหน้าหรือโปรแกรมย่อย (predefined process) แสดงถึงกระบวนที่กำหนดไว้ล่วงหน้าหรือโปรแกรมย่อยที่อยู่ภายนอกผังงานนี้

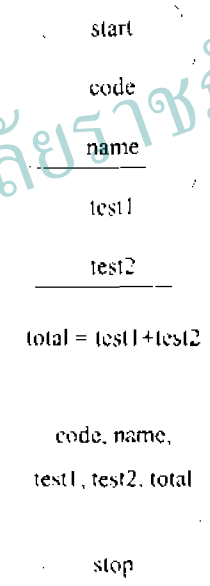
ประโยชน์ของผังงานสรุปได้ดังนี้

1. ช่วยลำดับขั้นตอนการทำงานของโปรแกรมได้โดยไม่สับสน
2. ช่วยให้สามารถตรวจสอบ แก้ไขข้อผิดพลาดของโปรแกรมได้ง่าย
3. ช่วยให้การดัดแปลง แก้ไขโปรแกรมทำได้สะดวกรวดเร็ว
4. ช่วยให้ผู้อื่นสามารถศึกษาการทำงานของโปรแกรมได้ง่ายและรวดเร็ว

วิธีการเขียนผังงานให้เกิดขั้นตอนการทำงานที่ดีและถูกต้อง

1. ใช้สัญลักษณ์ตามที่กำหนดไว้
2. ใช้ลูกศรแสดงทิศทางการไหลของข้อมูลจากบนลงล่างหรือซ้ายไปขวา
3. คำอธิบายในภาพควรสั้นกะทัดรัด และเข้าใจง่าย
4. ทุกแผนภาพต้องมีลูกศรแสดงทิศทางเข้า-ออก
5. ไม่ควรโยงเส้นเชื่อมผังงานที่อยู่ไกลมากๆ ควรใช้สัญลักษณ์จุดเชื่อมต่อแทน
6. ผังงานควรมีการทดสอบความถูกต้องของการทำงานก่อนนำไปเขียนโปรแกรม

ตัวอย่างการเขียนผังงานเพื่อการเขียนโปรแกรมหาคะแนนรวมของการสอบ ของนักศึกษาที่มีการสอบ 2 ครั้งใน 1 ภาคเรียน เขียนเป็นผังงานได้ดังภาพที่ 1.7



ภาพที่ 1.7 ผังงานแสดงขั้นตอนการทำงานของโปรแกรมหาคะแนนรวม

สรุป

คอมพิวเตอร์ หมายถึงอุปกรณ์ทางอิเลคทรอนิกส์ที่ ทำงานด้วยระบบอัตโนมัติ มี 3 ขนาด คือ ขนาดใหญ่ ขนาดกลาง และ ขนาดเล็ก มีส่วนประกอบ 3 ส่วนด้วยกัน ได้แก่ หน่วยรับข้อมูล หน่วยประมวลผลกลาง และ หน่วยแสดงผลลัพธ์ องค์ประกอบพื้นฐานของระบบ

งานคอมพิวเตอร์ ประกอบด้วย อุปกรณ์ทางคอมพิวเตอร์ โปรแกรมสั่งงาน บุคลากรที่เกี่ยวข้อง กระบวนวิธีการ และ ข้อมูล คอมพิวเตอร์ทำงานได้ด้วยการสั่งงานเป็นภาษาคอมพิวเตอร์ แบ่งเป็น 3 ประเภทคือ ภาษาเครื่อง ภาษาสัญลักษณ์ และ ภาษาระดับสูง ภาษาที่ใช้กันอยู่ในปัจจุบันเป็นภาษาระดับสูง ที่ต้องนำไปแปลเป็นภาษาเครื่อง การแปลมี 2 ลักษณะคือ การแปลแบบอินเตอร์พรีตเตอร์ และการแปลแบบคอมไพเลอร์ การพัฒนาโปรแกรม แบ่งเป็น 5 ขั้นตอน ได้แก่ การวิเคราะห์ปัญหา การวางแผนในการสั่งงาน การลงรหัส การทดสอบโปรแกรม และ การทำเอกสารประกอบ เครื่องมือที่ใช้ในการพัฒนาโปรแกรม ที่นิยมใช้ได้แก่ ผังงาน หมายถึงการแสดงลำดับขั้นตอนโดยการใช้สัญลักษณ์ และ อัลกอริทึม ที่แสดงขั้นตอนโดยการเขียนอยู่ในลักษณะของภาษามนุษย์ที่สามารถเข้าใจได้ง่าย

คำถามทบทวน

จงตอบคำถามต่อไปนี้โดยสังเขป

1. ให้นักเรียนบอกความหมายและคุณสมบัติของเครื่องคอมพิวเตอร์มาพอสังเขป
2. ให้นักเรียนยกตัวอย่างอุปกรณ์ที่ทำหน้าที่รับข้อมูลและแสดงผลข้อมูลมาอย่างละ 3 ชื่อ
3. ให้นักเรียนบอกความแตกต่างของ floppy disk และ hard disk
4. ให้นักเรียนบอกองค์ประกอบพื้นฐานของระบบคอมพิวเตอร์ว่ามีอะไรบ้าง
5. ให้นักเรียนบอกอธิบายความหมายของภาษาคอมพิวเตอร์ ทั้ง 3 ภาษา คือ
   5.1 ภาษาเครื่อง
   5.2 ภาษาสัญลักษณ์
   5.3 ภาษาระดับสูง
6. ให้นักเรียนอธิบายถึงขั้นตอนในการพัฒนาโปรแกรม

เอกสารอ้างอิง

ครรชิต มาลัยวงศ์. (2538). **พจนานุกรมคอมพิวเตอร์.** กรุงเทพฯ: ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ.

ทักษิณา สวนานนท์. (2544). **พจนานุกรมศัพท์คอมพิวเตอร์ ฉบับนิสิตนักศึกษา.** กรุงเทพฯ: ไอบริด พริ้นติ้ง.

ธงชัย สิทธิกรณ์. (2540). **ทฤษฎีระบบคอมพิวเตอร์.** กรุงเทพฯ: สยามสปอร์ตซินดิเคท.

วรรณวิภา จำเริญดารารัศมี. (2535). **วิทยาการคอมพิวเตอร์เบื้องต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

วาสนา สุขกระสานติ. (2540). **โลกของคอมพิวเตอร์และสารสนเทศ**. กรุงเทพฯ: จุฬาลงกรณ์มหาวิทยาลัย.

ศิริรัตน์ ชำนาญรบ, เกื้อกูล ตาเย็น, และวัชระ โพธิสรณ์. (2539). **ความรู้เบื้องต้นเกี่ยวกับคอมพิวเตอร์.** กรุงเทพฯ: ภูมิบัณฑิต.

แผนบริหารการสอนประจำบทที่ 2

เนื้อหาประจำบท

แนะนำภาษาซี

โครงสร้างของโปรแกรมภาษาซี

ข้อมูลและชนิดของข้อมูล

การใช้ข้อมูลในภาษาซี

การใช้ตัวแปรในภาษาซี

ตัวดำเนินการในภาษาซี

การแปลโค้ดของภาษาซี

สรุป

คำถามทบทวน

จุดประสงค์เชิงพฤติกรรม

1. เพื่อให้ผู้เรียนสามารถอธิบายถึงลักษณะของโปรแกรมในภาษาซีได้
2. เพื่อให้ผู้เรียนสามารถอธิบายถึง ข้อมูล และ ชนิดของข้อมูลที่ใช้ในภาษาซีได้
3. เพื่อให้ผู้เรียนสามารถบอกชนิดของตัวดำเนินการ ที่ใช้ในภาษาซีได้
4. เพื่อให้ผู้เรียนสามารถบอกความหมายของตัวดำเนินการ ที่ใช้ในภาษาซีได้
5. เพื่อให้ผู้เรียนสามารถอธิบายถึงการเขียนนิพจน์ชนิดต่างๆได้
6. เพื่อให้ผู้เรียนสามารถอธิบายถึงการแปลงโค้ดในภาษาซีได้

วิธีสอนและกิจกรรมการเรียนการสอน

1. สอนแบบบรรยาย นำเข้าสู่บทเรียนด้วยการกล่าวถึงการสั่งให้คอมพิวเตอร์ทำงานในปัจจุบัน

2. กิจกรรมการเรียนการสอน

   2.1 แสดงตัวอย่างการใช้สั่งงานคอมพิวเตอร์โดยโปรแกรมภาษาซี

   2.2 ฟังบรรยาย

   2.3 ทำแบบฝึกหัดท้ายบท

สื่อการเรียนการสอน

1. เอกสารประกอบการเรียนบทที่ 2
2. เพาเวอร์พอยท์ พรีเซนเตชั่น (power point presentation) ประกอบการบรรยาย

การวัดและการประเมินผล

1. สังเกตจากการตอบคำถาม และการตั้งคำถาม ในชั้นเรียน
2. วัดเจตคติจากการสังเกตพฤติกรรม ความกระตือรือร้นในการทำกิจกรรม
3. ความถูกต้องและคุณภาพของการทำแบบฝึกหัดท้ายบทเรียน

บทที่ 2
การเขียนโปรแกรมด้วยภาษาซี

การสั่งให้คอมพิวเตอร์ทำงานตามที่ต้องการนั้น จะต้องมีปัจจัยต่าง ๆ เข้ามาเกี่ยวข้อง เช่น เครื่องคอมพิวเตอร์ บุคลากร ซอฟท์แวร์ ข้อมูล และ ขั้นตอนกระบวนการ การสั่งงานด้วยคอมพิวเตอร์นั้นมี สองแนวทางคือ การใช้โปรแกรมสำเร็จรูปซึ่งถือว่าเป็นเรื่องที่ง่าย เพราะเพียงแต่ศึกษาเรื่องการใช้งาน ก็จะสามารถนำมาใช้ประยุกต์กับงานในชีวิตประจำวันได้ อีกแนวทางหนึ่งคือ การเขียนโปรแกรมด้วยภาษาคอมพิวเตอร์ แนวทางนี้จำเป็นต้องศึกษาความรู้ทางคอมพิวเตอร์ มากพอสมควรถึงจะสั่งงานด้วยโปรแกรมได้ การถึงแม้ว่าจะมีโปรแกรมสำเร็จรูปที่ผลิตออกมาใช้งานกันอย่างมากมายและง่ายแก่การใช้งาน แต่โปรแกรมสำเร็จรูปก็ไม่ได้สนองความต้องการของงานทุกงานได้อย่างถูกใจผู้ใช้ในทุกกรณีไป การเขียนโปรแกรมก็ยังเป็นสิ่งจำเป็นที่จะต้องมีผู้ศึกษาเพื่อที่จะเขียนโปรแกรมออกมาใช้งานและผลิตเป็นโปรแกรมสำเร็จรูปในที่สุด อย่างไรก็ตาม การเขียนโปรแกรมให้กับคอมพิวเตอร์ทำงานนั้น จะต้องเรียนรู้ถึง พื้นฐานของภาษานั้น ๆ ก่อน เพื่อจะได้เข้าใจ ความเป็นมาของภาษา ตลอดจนโครงสร้างโดยรวมของภาษา เช่น การใช้อักขระ การใช้คำ เพื่อมาประกอบกันเป็นข้อมูลเพื่อใช้งาน ดังนั้นในบทนี้จะกล่าวถึงความเป็นมาของภาษาซี โครงสร้างของโปรแกรมในภาษาซี ข้อมูลและชนิดของข้อมูลที่สามารถ ใช้ในภาษาซีได้ การดำเนินการต่างในภาษาซี และ การแปลงโค้ดในภาษาซี เพื่อให้ผู้ที่ศึกษาสามารถเข้าใจถึงโครงสร้างและสามารถเขียนโปรแกรมในภาษาซีอย่างง่ายได้

แนะนำภาษาซี

ภาษาซี เป็นภาษาคอมพิวเตอร์ระดับสูง ที่มีลักษณะของโปรแกรมเป็นโปรแกรมโครงสร้าง มีลักษณะรูปแบบของคำสั่งที่เข้าใจง่าย และ มีฟังก์ชันให้ประยุกต์ใช้งานอย่างมากมาย ลักษณะการแบ่งส่วนการทำงาน เป็นลักษณะโปรแกรมย่อย ๆ หรือที่เรียกว่า โมดูล (module) ส่งค่าผ่านไปมาระหว่าง โปรแกรมย่อยได้อย่างสะดวก แบ่งส่วนการทำงานได้ ตามจุดประสงค์ของผู้เขียน โปรแกรม ทำให้เกิดประสิทธิภาพในการสั่งงานได้ในระดับสูง เหมาะกับงานด้าน ตัวเลข และ ธุรกิจ มีคำจำกัดความเกี่ยวกับภาษาซี จากนักวิชาการต่างๆ ดังนี้

มนตรี พจนาถลาวัลย์ (2535, หน้า 28) กล่าวว่า ภาษาซีได้ถูกคิดค้นขึ้นโดย Denis Ritchie ในปี ค.ศ. 1970 โดยใช้ระบบปฏิบัติการยูนิกซ์ (UNIX) เป็นภาษาคอมพิวเตอร์ที่มีการแปลโดยวิธี คอมไพล์ (compile) และเป็นโปรแกรมชนิดโครงสร้าง ที่ง่ายแก่การควบคุมการทำงาน

ใช้งานกับคอมพิวเตอร์ตั้งแต่ระดับไมโครคอมพิวเตอร์จนถึงระดับเมนเฟรม ทั้งนี้เนื่องจากความอ่อนตัวของของภาษาในลักษณะตัวภาษาที่สามารถปรับเข้าได้กับทุกระบบ

ธันวา ศรีประโมง (2539, หน้า 20) กล่าวว่า มาตรฐานของภาษาซีที่เราใช้ในปัจจุบันคือ ANSI C ยึดตามคอมไพล์เลอร์ ของบอร์แลนด์ซี ซึ่งเป็น ภาษาซีพลัสพลัส (C++) ภาษาซี ทำการควบคุมการทำงานทั้งหมดของคอมพิวเตอร์ได้ อย่างที่ภาษาระดับสูงอื่น ๆ ก่อนหน้านั้นไม่มีความสามารถดังกล่าว ซึ่งแต่เดิมจะมีเฉพาะในภาษาแอสเซมบลีเท่านั้น ข้อดีของภาษาซีสรุปได้ดังนี้

1. คุณสมบัติการเข้ากันได้กับฮาร์ดแวร์ (portable) ซึ่งไม่จำเป็นต้องทำการเปลี่ยนซ้อสโค้ด (source code) เมื่อเปลี่ยนฮาร์ดแวร์ หรือเปลี่ยนก็เพียงเล็กน้อยเท่านั้น

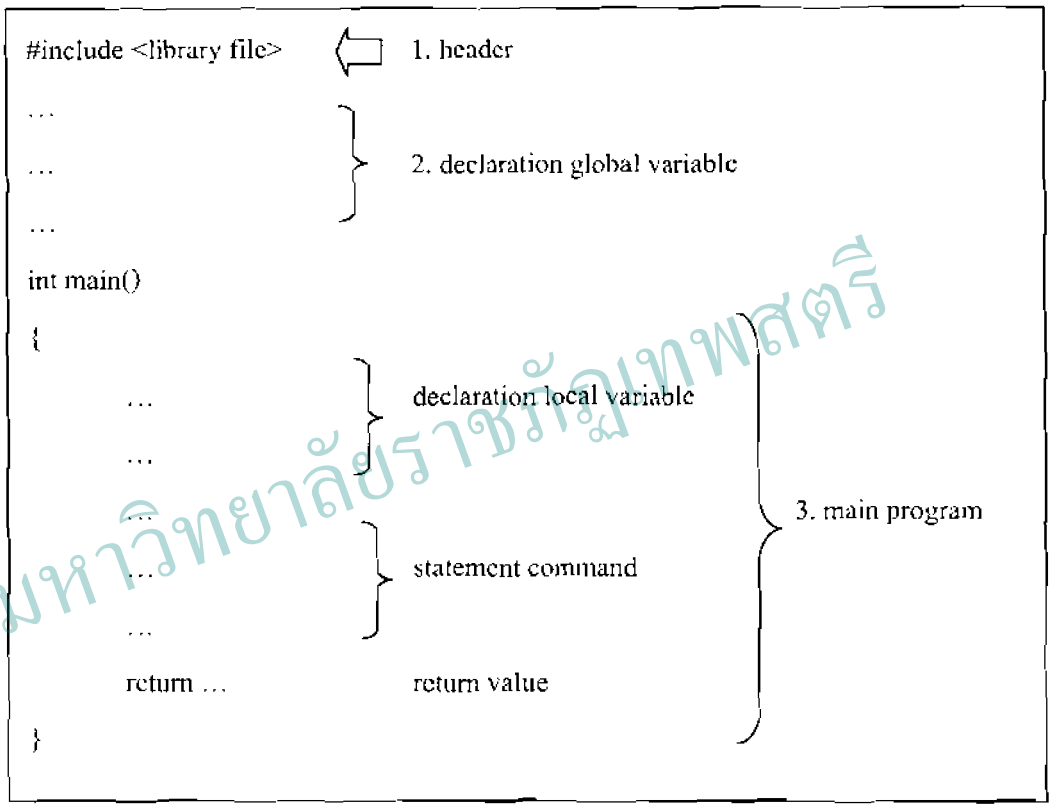
2. การอ่อนตัว (flexible) ความอ่อนตัวนี้จะเห็นจากภาษาซี สามารถที่จะทำงานในระดับหน่วยย่อยสุดของข้อมูลคือ ระดับบิต ได้เช่นเดียวกับภาษาระดับต่ำแต่มีรูปแบบการเขียนโปรแกรมเป็นเหมือนภาษาระดับสูงอย่าง FORTRAN, PASCAL นี่เป็นสาเหตุให้ โปรแกรมคอมพิวเตอร์และซอฟต์แวร์ที่สำคัญ ๆ จำนานมากสร้างขึ้นจากภาษาซี แม้กระทั่ง DOS หรือ WINDOWS ที่ท่านคุ้นเคยก็เขียนด้วยภาษาซี

3. ประสิทธิภาพสูง (powerful) หลังจากคอมไพล์โปรแกรมแล้ว ภาษาซีจะให้ออบเจคโค้ด (object code) ที่สั้นทำให้การทำงานของโปรแกรมที่ถูกเขียนด้วยภาษาซี มีความเร็วสูงแต่ประสิทธิภาพนี้อาจถูกจำกัดที่ตัวคอมไพล์เลอร์ ด้วยคอมไพล์เลอร์ ที่นิยมใช้กันมากในกลุ่มผู้พัฒนาระบบเป็นของ บริษัท Borland C เพราะมีเครื่องมือช่วยพัฒนาระบบมากมาย การตรวจจับข้อผิดพลาด ทำได้ง่าย

จากคุณสมบัติดังกล่าว ภาษาซี จึงเป็นภาษาที่เหมาะสมกับสภาพการณ์ในปัจจุบัน เพื่อการศึกษา หาความรู้ด้านการเขียนโปรแกรม เนื่องจากเป็นภาษาที่ง่ายแก่การศึกษา และมีประสิทธิภาพสูง สามารถนำเอาความรู้ที่ได้จากการเขียนโปรแกรมนี้ ไปประยุกต์ใช้งาน ได้ทั้งทางงานด้านวิทยาศาสตร์ และ ทางด้านธุรกิจ อีกทั้งยังสามารถนำความรู้ที่ได้ไปเป็นพื้นฐานในการใช้งานโปรแกรมภาษาอื่นได้ง่าย เพราะโปรแกรมภาษาซี มีส่วนคล้ายกันกับ ภาษาปาสคาล ที่สามารถนำไปเขียนในภาษาเดลไฟ ได้ และยังมีส่วนคล้ายกับ ภาษาจาวา ที่เป็นภาษาที่นิยมใช้งานในปัจจุบันโปรแกรมภาษาซี ที่จะกล่าวถึงในเอกสารฉบับนี้ จะกล่าวถึง โปรแกรมภาษาซีพลัสพลัส ซึ่งพัฒนาต่อยอดมาจากภาษาซี โดยได้เพิ่ม คำสั่งและฟังก์ชันที่จำเป็นในการใช้งานเข้าไป ในที่นี้จะเรียกสั้น ๆ ว่า ภาษาซี

โครงสร้างของโปรแกรมภาษาซี

ภาษาซี เป็นภาษาที่มีการเขียนโปรแกรมในลักษณะของโครงสร้างที่ประกอบด้วยโปรแกรมย่อย ๆ หลายโปรแกรม ภาษาซีมองโปรแกรมย่อยเป็นลักษณะของฟังก์ชัน ดังนั้นหลังจากทำงานเสร็จสิ้นจากฟังก์ชันแล้ว จะต้องมีการส่งค่าคืนไปยังโปรแกรมที่เรียกใช้ฟังก์ชันนั้น ฟังก์ชันหลักของโปรแกรมในภาษาซี เรียกว่า main เพื่อให้เข้าใจได้ยิ่งขึ้น จึงนำเสนอรูปแบบโครงสร้างโปรแกรมในภาษาซีดัง ภาพที่ 2.1



ภาพที่ 2.1 โครงสร้างโปรแกรมภาษาซี

จากภาพที่ 2.1 โครงสร้างของโปรแกรมในภาษาซี ประกอบด้วย 3 ส่วนประกอบใหญ่ด้วยกันคือ

1. ส่วนหัว (header)
2. ส่วนกำหนดค่า (declaration)
3. ส่วนตัวโปรแกรมหลัก (main program)

1. ส่วนหัว (header) เป็นส่วนแรกที่ต้องมีการกำหนด มีความหมายถึง การนำเอาคำสั่ง หรือ ฟังก์ชันที่เขียนสำเร็จรูปแล้ว เข้ามาร่วมแปลเพื่อใช้ในโปรแกรม โดยไม่ต้องเสียเวลาในการเขียนอีก คำสั่ง หรือฟังก์ชันที่เขียนไว้ต่างๆ เหล่านี้ เรียกว่า ไลบรารี (library)ในภาษาซี จะหมายถึงไฟล์ที่มีสกุลเป็น h เช่น stdio.h conio.h iostream.h เป็นต้น หรือบางครั้งส่วนหัวนี้ จะมีการกำหนดชื่อหรือค่าต่างๆ เพื่อเป็นค่าที่จะนำไปใช้ในโปรแกรม อาจเป็นค่าคงที่ หรือนิพจน์ต่าง ๆ สิ่งสำคัญในส่วนนี้มี 2 ส่วนได้แก่

1.1 # (preprocessor) หมายถึงเครื่องหมายที่แจ้งให้คอมไพเลอร์ทำการใดๆ ก่อนที่จะทำการแปลโปรแกรมต่อไป เช่น การนำไฟล์อื่นมาแปลร่วมด้วย การกำหนดชื่อต่าง ๆ เพื่อเป็นตัวแทนค่าหรือนิพจน์ต่าง ๆ ที่จะนำมาใช้ในโปรแกรม

1.2 ชื่อของ ไลบรารีไฟล์ หรือ ชื่อของค่าต่าง ๆ

ส่วนหัวที่นิยมใช้โดยทั่วไปในภาษาซีพลัสพลัส ได้แก่

include <iostream.h> หมายถึงให้นำไฟล์ iostream.h เข้ามาร่วมแปลด้วย ไฟล์นี้จะรวมคำสั่งและฟังก์ชัน ที่เกี่ยวข้องกับ การรับ และ การแสดงผลข้อมูล ไว้

#define expression_name expression หมายถึงกำหนด ชื่อ หรือ นิพจน์ (expression _name) ที่จะมีการนำไปใช้ในโปรแกรม ให้มีค่าตามที่ต้องการ เช่น # define vat 7/100 หมายถึง กำหนดให้ vat มีค่าเท่ากับ 7/100 เพื่อที่จะนำไปใช้ในโปรแกรม

2. ส่วนกำหนดค่า (declaration) หมายถึง ส่วนที่จะใช้ในการประกาศชื่อข้อมูล (variable) หรือชื่อฟังก์ชัน ที่จะนำไปใช้ในโปรแกรม ชื่อที่มีการประกาศในส่วนนี้ จะเรียกว่าประกาศให้เป็นประเภทโกลบอล (global) มีขอบเขตที่จะสามารถนำไปใช้ได้ตลอดโปรแกรม เช่น int a; จะหมายถึง กำหนดให้ a เป็นชื่อของข้อมูลที่เป็นเลขจำนวนเต็ม และสามารถนำชื่อ a นี้ไปใช้ได้ตลอดโปรแกรม

3. ส่วนตัวโปรแกรม (main program) หมายถึง ส่วนของฟังก์ชันหลักซึ่งหมายถึง ทุกโปรแกรมจะต้องมีฟังก์ชันนี้ มีการทำงานของคำสั่ง ตามลำดับจาก ด้านบน ลง ด้านล่าง ตัวโปรแกรมนี้ ยังแบ่งเป็นส่วนย่อยได้อีก 3 ส่วนคือ

3.1 ส่วนกำหนดค่า หมายถึง การประกาศชื่อข้อมูลต่าง ๆ ขึ้นมาใช้งานเฉพาะ ในส่วนนี้หรือที่เรียกว่า โลคอล (local) ชื่อข้อมูลที่มีการประกาศตรงส่วนนี้จะใช้ได้เฉพาะส่วนนี้เท่านั้น เมื่อออกจากส่วนนี้ไป เช่นออกไปยังฟังก์ชันหรือโปรแกรมย่อยอื่น ๆ จะไม่สามารถใช้ชื่อเหล่านี้ได้

3.2 ส่วนประโยคคำสั่ง (statement) หมายถึง ส่วนที่ใช้ในการออกคำสั่งให้เครื่องทำงานตามลำดับขั้นตอน ที่กำหนดไว้ เพื่อให้ได้ผลลัพธ์ตามที่ต้องการ รายละเอียดของคำสั่งต่าง ๆ จะกล่าวถึงในส่วนต่อไป

3.3 ส่วนคืนค่า (return value) หมายถึงส่วนที่เป็นการคืนค่าจากฟังก์ชัน เมื่อทำงานสิ้นสุดแล้ว การคืนค่านี้ จะต้องสอดคล้องกับชนิดของฟังก์ชันนั้นด้วย เช่น ถ้าเป็นฟังก์ชันชนิดตัวเลข ต้องคืนค่าแบบ ตัวเลข ถ้าเป็นฟังก์ชันชนิด ตัวอักษรก็จะคืนค่าแบบ ตัวอักษร เป็นต้น

ข้อมูลและชนิดของข้อมูล

ข้อมูลที่ใช้ในโปรแกรมคอมพิวเตอร์ เป็นอีกปัจจัยหนึ่งของระบบงานคอมพิวเตอร์ ที่จะทำงานให้ได้ นั้น จำเป็นจะต้องมีข้อมูล เครื่องคอมพิวเตอร์ จะสามารถทำงาน หรือ ให้คำตอบกับเราได้ คอมพิวเตอร์เครื่องนั้นจะต้องมีข้อมูล ถ้าไม่มีข้อมูลแล้ว คอมพิวเตอร์ ก็ไม่สามารถให้คำตอบกับเราได้ ข้อมูลที่เราใช้ในชีวิตประจำวัน จะถูกถ่ายทอดลงในเครื่องคอมพิวเตอร์ เพื่อให้คอมพิวเตอร์ นำเอาข้อมูลเหล่านั้นไปทำการประมวลผล แล้วให้ผลลัพธ์ที่ต้องการออกมา ดังนั้น ข้อมูลที่นำเข้าไปใช้ในคอมพิวเตอร์ ก็จะถูกกำหนดมาจากความจำเป็นของการใช้ข้อมูลในชีวิตประจำวันของคนเรา ที่มีการสื่อสารแลกเปลี่ยนข้อมูลซึ่งกันและกัน ข้อมูลที่เราใช้มีอยู่ด้วยกัน 3 ประเภทใหญ่ ๆ ได้แก่

1. ข้อมูลประเภทตัวอักษร (text) หมายถึง ข้อมูลที่เราใช้ในการเขียนข้อความ หรือประโยค ข้อมูลประเภทนี้ ไม่มีผลในทางการคำนวณ ใช้เพื่อเป็นการเรียกชื่อบางสิ่งบางอย่าง เช่น ชื่อคน ชื่อสิ่งของ ชื่อสถานที่ หรือ ใช้อธิบายการแสดงออกของสถานการณ์ เช่น ให้เติมข้อมูล ให้แสดงข้อมูล เป็นต้น

2. ข้อมูลประเภทตัวเลข (numeric) หมายถึง ข้อมูลที่เราใช้สำหรับการกำหนดมูลค่าให้กับสิ่งของที่เราใช้ ให้มีมูลค่าทางตัวเลข เพื่อจะให้มีผลในการคำนวณ และการจัดลำดับ เช่น ภาษีมูลค่าเพิ่ม ราคาสินค้า ค่าจ้างรายวัน เป็นต้น

3. ข้อมูลประเภทตรรกะ (logic) หมายถึง ข้อมูลที่ใช้ประกอบในการตัดสินใจ ของมนุษย์เรา เช่น ถ้าฝนตกจะปลูกต้นไม้ ถ้าฟังเสียงโทรทัศน์แล้วได้ยินไม่ถนัดให้เร่งความดังของเสียงขึ้น เหล่านี้เป็นต้น

ข้อมูลในภาษาซี ก็จะยึดหลักความต้องการใช้ข้อมูลของมนุษย์ดังกล่าว แต่มีการแบ่งชนิดของข้อมูลให้ย่อยลงไปเพื่อความสะดวกในการทำงาน จึงได้แบ่งข้อมูลที่ใช้ในโปรแกรมออกเป็น 5 ประเภทด้วยกันคือ

1. ข้อมูลชนิดตัวเลขจำนวนเต็ม (integer type) ได้แก่ ตัวเลขที่เป็นตัวเลขจำนวนเต็มไม่มีเศษ หรือ ไม่มีทศนิยม เช่น 25, 30, -75 เป็นต้น

2. ข้อมูลชนิดตัวเลขทศนิยม (floating point type) ได้แก่ ตัวเลขที่มีค่าเป็นเศษได้หรือเรียกว่า มีทศนิยม เช่น 10.5, 20.8, -28.75 เป็นต้น

3. ข้อมูลชนิดตัวอักษร (character type) ได้แก่ ตัวอักษร A-Z, a-z เลข 0-9 และสัญลักษณ์ต่าง ๆ มีความยาวเพียง 1 ตัวอักษร เช่น อักษร A อักษร ก หรือเครื่องหมาย % เป็นต้น

4. ข้อมูลชนิดข้อความ (string type) ได้แก่ การที่เรานำเอาตัวอักษรมาประกอบกันเป็นข้อความ เช่น ประกอบกันเป็นคำนามเพื่อเรียกชื่อต่าง ๆ หรือ ประกอบกันเพื่อแสดงอาการต่าง ๆ เป็นต้น

5. ข้อมูลชนิดตรรก (logic type) ได้แก่ ข้อมูลประเภทที่ได้จากการ เอาข้อมูลที่กล่าวไว้แล้วมาทำการเปรียบเทียบกัน ตามลักษณะที่ต้องการเช่น มากไปหรือไม่ น้อยไปหรือไม่ และเท่ากันหรือไม่ เป็นต้น

การใช้ข้อมูลในภาษาซี

ในการใช้ข้อมูลเราสามารถนำข้อมูลที่กล่าวมาทั้งหมด มาใช้โดยทางตรง และ โดยทางอ้อม ได้ ซึ่งจะมีประโยชน์และความจำเป็นแตกต่างกันไป การใช้ข้อมูลแบบทางตรง จะเรียกว่าการใช้ข้อมูลในลักษณะค่าคงที่ ส่วนการใช้ข้อมูลโดยทางอ้อม จะเรียกว่าการใช้ข้อมูลผ่านทางตัวแปร หรือ การใส่ชื่อเพื่อเรียกข้อมูลนั้น ๆ ในโปรแกรมคอมพิวเตอร์ เราก็จะใช้ข้อมูลโดยวิธีการทั้ง 2 อย่างควบคู่กันไป ดังนั้น จึงต้องทราบถึงค่าของข้อมูลต่าง ๆ ที่จะถูกนำมาใช้ดังนี้

1. การใช้ข้อมูลแบบค่าคงที่ (constant) หมายถึงการนำข้อมูลมาใช้โดยทางตรง ค่าของข้อมูลที่เกิดขึ้นจึงเป็นค่าคงที่ ซึ่งประกอบไปด้วย

1.1 ค่าคงที่ประเภทตัวเลข หมายถึงค่าคงที่ที่ประกอบไปด้วยค่า 0-9 แล้วแต่จะนำมาประกอบกันเป็นค่าเท่าใดก็ได้ รวมไปถึงค่าเศษที่เรียกว่า ทศนิยมด้วย

1.2 ค่าคงที่ประเภทตัวอักษร หมายถึง การที่เรานำเอาอักษร และ เครื่องหมายต่าง ๆ มาประกอบกันเป็น คำ ข้อความ หรือ ประโยค เพื่อการสื่อสารให้เข้าใจตรงกัน

2. การใช้ข้อมูลแบบตัวแปร (variable) หมายถึง การใช้ข้อมูลโดยวิธีการทางอ้อม ทำได้โดยการตั้งชื่อให้ข้อมูลนั้น ๆ ซึ่งปกติแล้ว การเก็บข้อมูลลงในหน่วยความจำ จะเก็บในตำแหน่งต่าง ๆ ของหน่วยความจำ โดยมีหมายเลขกำกับ ซึ่งเรียกว่า แอดเดรส (address) แต่การอ้างถึงโดยใช้หมายเลข แอดเดรส ไม่สะดวกในการเรียกใช้ เพราะหมายเลขแอดเดรสมีจำนวนหลักของตัวเลขมาก เพื่อแก้ปัญหาส่วนนี้ จึงตั้งชื่อขึ้นมาเพื่ออ้างถึง แอดเดรสดังกล่าวอีกชั้นหนึ่ง ซึ่งเรียกชื่อ

เหล่านั้นว่า ชื่อตัวแปร ในการเขียนโปรแกรมที่มีการใช้ตัวแปร ต้องทำการกำหนด ตัวแปรเพื่อการใช้งานก่อน ว่าตัวแปรที่กำหนดขึ้นมานั้น จะให้เก็บข้อมูลชนิดใด การใช้ข้อมูลในลักษณะนี้ จะมีประโยชน์มาก เพราะตัวแปรสามารถเปลี่ยนแปลงค่าได้ตามที่เราสั่งการ หรือ ตามสถานการณ์ที่เกิดขึ้น ดังนั้นจึงต้องทราบถึงการกำหนดชื่อ ตัวแปรก่อนว่ามีกฎเกณฑ์ อย่างไร กฎเกณฑ์ในการตั้งชื่อตัวแปรในภาษาซี มีดังนี้

กฎเกณฑ์ในการตั้งชื่อตัวแปร

1. อักขระที่จะนำมาเป็นชื่อได้ประกอบด้วยอักษร A-Z ตัวเลข 0-9 และเครื่องหมาย ขีดล่าง (_)

2. ต้องขึ้นต้นด้วย ตัวอักษร หรือ เครื่องหมายขีดล่าง (_) เท่านั้น

3. ความยาวของชื่อไม่จำกัด แต่เฉพาะจำนวน 32 ตัวแรกเท่านั้นที่จะมีความหมายในการกำหนดชื่อ

4. ชื่อที่เขียนด้วยตัวอักษรตัวใหญ่ และ ตัวอักษรตัวเล็ก จะถือว่าเป็นคนละตัวกัน เช่น TOTAL_SALE Total_sale และ total_sale จะถือเป็นชื่อคนละตัวกัน

5. ชื่อที่กำหนดขึ้นจะต้องไม่ซ้ำกับคำสงวนในภาษาซี (reserve words)

ตัวอย่างการตั้งชื่อตัวแปร

code	เป็นชื่อที่ ถูกต้องตามกฎการตั้งชื่อ
student name	เป็นชื่อที่ ผิด เพราะมีช่องว่าง
student_name	เป็นชื่อที่ ถูกต้องตามกฎการตั้งชื่อ
subject#1	เป็นชื่อที่ ผิด เพราะใช้เครื่องหมาย #
SbJcOdE	เป็นชื่อที่ ถูกต้องตามกฎการตั้งชื่อ
short	เป็นชื่อที่ ผิด เพราะเป็นคำสงวน

การตั้งชื่อตัวแปร จะต้องไม่ไปซ้ำกับคำสงวน ผู้เขียนโปรแกรม จำเป็นที่จะต้องทราบว่าคำใดเป็น คำสงวนในภาษาซี หรือ จะกล่าวอีกนัยหนึ่ง คำสงวนก็คือ คำที่ภาษาซีใช้เป็นคำสั่ง หรือใช้ในการอื่น ที่เกี่ยวข้องกับภาษา เพื่อให้ผู้เรียนได้ทราบถึงคำสงวน จึงแสดงในตารางที่ 2.1

ตารางที่ 2.1 คำสงวนในภาษาซี

Asm	_asm	__asm	auto	break	case
cdecl	_cdecl	__cdecl	char	class	const
continue	_cs	__cs	eefault	delete	do
double	_ds	__ds	else	enum	_cs
__cs	_export	__export	exter	far	_far
__far	_fastcall	__fastcall	float	for	friend
goto	huge	_huge	__huge	if	inline
int	interrupt	_interrupt	__interrupt	_loadds	__loadds
long	near	_near	__near	new	operator
pascal	_pascal	__pascal	private	protected	public
register	return	_saveregs	__saveregs	_seg	__seg
short	signed	sizeof	_ss	__ss	static
struct	switch	template	this	typedef	union
unsigned	Virtual	void	volatile	while	

ที่มา (Borland International, 1992, p. 19)

การใช้ตัวแปรในภาษาซี

ดังกล่าวแล้วว่าการใช้ข้อมูลในโปรแกรมภาษาซีนั้น เราเลือกที่จะใช้ข้อมูลได้ 2 วิธี คือ การใช้ข้อมูลโดยนำข้อมูลนั้นมาใช้โดยตรง เรียกว่าการใช้ ค่าคงที่ ส่วนอีกวิธีหนึ่ง เป็นการใช้ข้อมูลแบบการใช้ตัวแปร มีวิธีการใช้คือ การที่เราตั้งชื่อให้ข้อมูลเหล่านั้นซึ่งเรียกว่าการตั้งชื่อตัวแปร เมื่อต้องการใช้ข้อมูลก็เพียงแต่ เรียกชื่อตัวแปรเหล่านั้น การใช้ข้อมูลโดยตัวแปร มีขั้นตอนในการใช้ดังนี้

1. กำหนดชื่อตัวแปร ในการกำหนดชื่อตัวแปร นอกจากการมีกฎเกณฑ์ในการตั้งชื่อดังกล่าวมาแล้ว ยังต้องมีการกำหนด ประเภทของตัวแปรว่า ต้องการให้ตัวแปรนั้นนำไปใช้กับข้อมูลชนิดใด ซึ่งมีรายละเอียดดังตารางที่ 2.2

ตารางที่ 2.2 ชนิดของตัวแปร

ชนิด	ขนาด (บิต)	ขอบเขต	ตัวอย่างการประยุกต์ใช้
unsigned char	8	0 ถึง 255	ตัวอักษร ASCII ทั้งหมด
char	8	-128 ถึง 127	เลขจำนวนเต็มขนาดเล็กรวมทั้งตัวอักษร ASCII
enum	16	-32,768 ถึง 32,767	กลุ่มของค่าจำนวนเต็มที่ถูกจัดลำดับ
unsigned int	16	0 ถึง 65,535	จำนวนเต็มขนาดกลาง
short int	16	-32,768 ถึง 32,767	จำนวนเต็มที่มีค่าบวกและลบ
int	16	-32,768 ถึง 32,767	จำนวนเต็มที่มีค่าบวกและลบ
unsigned long	32	-0 ถึง 4,9294,967,295	จำนวนเต็มบวกขนาดใหญ่
long	32	-2,147,483,648 ถึง 2,147,483,647	จำนวนเต็มบวกและลบขนาดใหญ่
float	32	3.4×10^{-38} ถึง 3.4×10^{38}	ใช้ในการคำนวณเลขทศนิยม 7 หลัก
double	64	1.7×10^{-308} ถึง 1.7×10^{308}	ใช้ในการคำนวณเลขทศนิยม 15 หลัก
long double	80	3.4×10^{-4932} ถึง 1.1×10^{4932}	ใช้ในการคำนวณเลขทศนิยม 19 หลัก
near pointer	16	Not applicable	การอ้างถึงเลขตำแหน่งในหน่วยความจำ
far pointer	32	Not applicable	การอ้างถึงเลขตำแหน่งที่อยู่นอกเซกเมนต์

จากตารางที่ 2.2 ตัวแปรที่นิยมนำมาใช้งานในโปรแกรมโดยทั่วไป ได้แก่ char ใช้กับข้อมูลชนิดตัวอักษร int ใช้กับตัวเลขจำนวนเต็ม และ float ใช้กับตัวเลขที่มีทศนิยม

2. นำตัวแปรไปใช้งาน หมายถึง การที่เราจะอ้างถึง ข้อมูลที่อยู่ในตัวแปรนั้น ๆ โดยการเรียกชื่อตัวแปร การใช้งานตัวแปรมีอยู่ 2 ลักษณะคือ

2.1 การกำหนดค่าให้กับตัวแปร หมายถึง การที่เรากำหนดลงไปว่า จะให้ตัวแปรตัวนั้นมีค่าเป็นเท่าใด กำหนดได้ 2 ลักษณะคือ กำหนดโดยใช้การกำหนดค่าจากโปรแกรม และกำหนดค่าโดยให้ผู้ใช้โปรแกรม ใส่ข้อมูลทางแป้นพิมพ์ รายละเอียดจะกล่าวถึงในบทต่อไป

2.2 การแสดงผลค่าของตัวแปร หมายถึงการสั่งให้เครื่องนำค่าที่เก็บอยู่ในตัวแปรนั้นออกมาทำการแสดงผล รายละเอียดจะกล่าวถึงในบทต่อไป

ตัวดำเนินการในภาษาซี

การดำเนินการเกี่ยวกับข้อมูลที่มีอยู่นั้น หมายถึง การที่เราสั่งให้เครื่องคอมพิวเตอร์ทำงานให้เรา เพื่อให้ได้ผลลัพธ์ที่ต้องการ หรือเรียกอีกอย่างหนึ่งว่า การประมวลผลในรูปแบบต่าง ๆ โดยใช้ ตัวดำเนินการทางด้านต่าง ๆ การดำเนินการในภาษาซี แบ่งประเภทตามการใช้งานได้ดังนี้

1. การดำเนินการทางคณิตศาสตร์
2. การดำเนินการทางตรรกศาสตร์
3. การดำเนินการเพื่อกำหนดค่า
4. การดำเนินการเกี่ยวกับขนาดของข้อมูล
5. การดำเนินการเกี่ยวกับบิต
6. การดำเนินการด้านอื่น ๆ

การดำเนินการทางคณิตศาสตร์ เป็นการดำเนินการในลักษณะของการคำนวณ ทางตัวเลข ในลักษณะต่าง ๆ ตามเครื่องหมายที่สั่งให้คำนวณ เช่น การบวก การลบ การคูณ การหาร เป็นต้น ตัวดำเนินการทางคณิตศาสตร์ แสดงไว้ในตารางที่ 2.3

ตารางที่ 2.3 ตัวดำเนินการทางคณิตศาสตร์

ตัวดำเนินการ	ชื่อ	ตัวอย่าง	คำอธิบาย
*	เครื่องหมายคูณ	X * Y	คูณค่าของ X ด้วยค่า Y
/	เครื่องหมายหาร	X / Y	หารค่า X ด้วยค่า Y
%	เครื่องหมายหารเอาเศษ	X % Y	เศษที่เหลือจากการนำค่า X หาร Y
+	เครื่องหมาย +	X + Y	บวกค่า X กับค่า Y
-	เครื่องหมาย –	X Y	ลบค่า X ด้วย Y
++	Increment	X++,	เพิ่มค่า X ขึ้นอีก 1 หลังจากการใช้
		++X	เพิ่มค่า X ขึ้นอีก 1 ก่อนการใช้
--	Decrement	Y--,	ลดค่า Y ลงอีก 1 หลักจากการใช้
		--Y	ลดค่า X ลงอีก 1 ก่อนจากการใช้
-	Negation	-X	ให้ค่าลบของค่าX
+	Unary Plus	+Y	ให้ค่าบวกของค่า Y

ที่มา (เกษมสันต์ พานิชการ, 2537, หน้า 40)

การประมวลผลของคอมพิวเตอร์ อีกลักษณะหนึ่ง คือการนำเอาข้อมูลมาเปรียบเทียบกัน เพื่อให้ได้ คำตอบที่เป็นตรรกศาสตร์ ที่มีคำตอบเป็น จริง หรือ เท็จ เพื่อประโยชน์ในการตัดสินใจในการทำงานของเครื่อง ว่าจะเลือกดำเนินการตามคำสั่งใด ตัวดำเนินการทางตรรกศาสตร์ แสดงในตารางที่ 2.4

ตารางที่ 2.4 ตัวดำเนินการทางตรรกศาสตร์

ตัวดำเนินการ	ชื่อ	ตัวอย่าง	คำอธิบาย
>	มากกว่า	x > y	ให้ค่า 1 เมื่อเป็นจริง นอกนั้น 0
<	น้อยกว่า	x < y	ให้ค่า 1 เมื่อเป็นจริง นอกนั้น 0
>=	มากกว่าหรือเท่ากับ	x >= y	ให้ค่า 1 เมื่อเป็นจริง นอกนั้น 0
<=	น้อยกว่าหรือเท่ากับ	x <= y	ให้ค่า 1 เมื่อเป็นจริง นอกนั้น 0
==	เท่ากันกับ	x == y	ให้ค่า 1 เมื่อเป็นจริง นอกนั้น 0
!=	ไม่เท่ากันกับ	x!= y	ให้ค่า 1 เมื่อเป็นจริง นอกนั้น 0
&&	AND	x && y	นำค่า x และค่า y มา and กันทางตรรกศาสตร์
||	OR	x || y	นำค่า x และค่า y มา or กันทางตรรกศาสตร์
!	NOT	! x	ให้ค่า 1 เมื่อ x เป็น 0 นอกนั้น 0

ที่มา (เกษมสันต์ พานิชการ, 2537, หน้า 41)

ตัวดำเนินการกำหนดค่า เป็นการกำหนดค่าให้กับตัวแปร ให้มีค่าตามที่ผู้เขียนโปรแกรมต้องการ การกำหนดค่านี้ นอกจากที่เราจะสามารถกำหนดค่าเป็นค่าคงที่ต่าง ๆ แล้ว เรายังสามารถกำหนดค่าที่เป็น นิพจน์ เพื่อให้เกิดการประมวลผลก่อนที่จะนำมากำหนดค่า ตัวดำเนินการกำหนดค่าแสดงในตารางที่ 2.5

ตารางที่ 2.5 ตัวดำเนินการกำหนดค่า

ตัวดำเนินการ	ชื่อ	ตัวอย่าง	คำอธิบาย
=	เครื่องหมายกำหนดค่า	x = y	เก็บค่า y ลงไปบนตัวแปร x
#=	เครื่องหมายกำหนดค่าผสม	x #==y;	มีผลการทำงานเหมือนกันกับ x = x # y; โดยตัว # สามารถแทนด้วยตัวดำเนินการต่อไปนี้ + - x / ~ % << >>& | ^

ที่มา (เกษมสันต์ พานิชการ, 2537, หน้า 41)

ตัวดำเนินการเกี่ยวกับขนาดและข้อมูล เป็นตัวดำเนินการเกี่ยวกับการนับจำนวนข้อมูล การหาตำแหน่งของข้อมูล และการหาขนาดของข้อมูล ว่ามีขนาด หรือ จำนวนเท่าใด ใช้กับข้อมูลประเภทโครงสร้างซึ่งจะกล่าวในบทท้าย ๆ ตัวดำเนินการเกี่ยวกับขนาดและข้อมูล แสดงในตารางที่ 2.6

ตารางที่ 2.6 ตัวดำเนินการเกี่ยวกับขนาดและข้อมูล

ตัวดำเนินการ	ชื่อ	ตัวอย่าง	คำอธิบาย
[ ]	สมาชิกของอะเรย์	X[0]	สมาชิกตัวที่ 1 ของอะเรย์ x
.	สมาชิกข้อมูลโครงสร้าง	Str.x	สมาชิกชื่อ x ของคลาส หรือ struct str
->	สมาชิกข้อมูลโครงสร้าง	Ptr->x	สมาชิกชื่อ x ของคลาสหรือ struct ที่ ptr ชี้ไป
*	ตัวแปรพอยน์เตอร์	*ptr	ptr จะเก็บข้อมูลเป็นตำแหน่งในหน่วยความจำ
&	Address of	&x	แสดงตำแหน่งในหน่วยความจำของ x
sizeof	ตัวดำเนินการบอกขนาด	sizeof(x)	จะแสดงขนาดของ x เป็นไบต์

ที่มา (เกษมสันต์ พานิชการ, 2537, หน้า 42)

ภาษาซี เป็นภาษาที่สามารถ เข้าไปสั่งงานได้ ถึงระดับ หน่วยประมวลผลภายใน หน่วยคำนวณและเปรียบเทียบ หรือ เรียกว่า รีจิสเตอร์ เป็นคุณสมบัติอีกข้อหนึ่งของภาษาซี การจัดการประมวลผลภายใน รีจีสเตอร์ หรือเรียกว่า บิต มีตัวดำเนินการดังตารางที่ 2.7

ตารางที่ 2.7 ตัวดำเนินการเกี่ยวกับบิต (bitwise)

ตัวดำเนินการ	ชื่อ	ตัวอย่าง	คำอธิบาย
~	ตัวดำเนินการ กอมพลีเมนต์	~x	ทำการกลับบิต 1 เป็น 0 และกลับบิต 0 เป็น 1
&	ตัวดำเนินการ AND	a & b	ทำการ AND a และ b ในระดับบิต
|	ตัวดำเนินการ OR	a | b	ทำการ OR a และ b ในระดับบิต
^	ตัวดำเนินการ XOR	a ^ b	ทำการ XOR a และ b ในระดับบิต
<<	ตัวดำเนินการเลื่อนซ้าย	x << y	ค่า x จะถูกเลื่อนไปทางซ้าย y บิต
>>	ตัวดำเนินการเลื่อนขวา	x >> y	ค่า x จะถูกเลื่อนไปทางขวา y บิต

ตัวดำเนินการอื่น ๆ ในภาษาซี เป็นการดำเนินการ ในเรื่องต่าง ๆ เช่น การใส่ค่าอาร์กิวเมนต์ ให้กับฟังก์ชัน การแปลงค่าชนิดข้อมูล การเลือกทำตามเงื่อนไข และ อื่น ๆ แสดงในตารางที่ 2.8

ตารางที่ 2.8 ตัวดำเนินการอื่น ๆ

ตัวดำเนินการ	ชื่อ	ตัวอย่าง	คำอธิบาย
( )	ฟังก์ชันคอลล์	pow(10,3)	เรียกฟังก์ชัน pow() โดยมีค่าอาร์กิวเมนต์ 2 ตัวคือ 10, 3
type( )	ประกาศชนิดของข้อมูล	double(i)	เปลี่ยนค่า i เป็นชนิด double
(type)	ประกาศชนิดของข้อมูล	(double)i	เปลี่ยนค่า i เป็นชนิด double
?:	เลือกตาม conditions	x1 ? X2 : x3	เมื่อ x1 เป็น 1 แล้วทำข้อคำสั่ง x2 นอกนั้นจะทำข้อคำสั่ง x3
,	ลำดับการทำงาน	i++,j++	จะทำการเพิ่มค่า i ก่อนจึงเพิ่ม j

ที่มา (เกษมสันต์ พานิชการ, 2537, หน้า 43)

ในการดำเนินการ ตามเครื่องหมายของการดำเนินการต่าง ๆ จะมีลำดับความสำคัญก่อนหลัง ว่าเมื่อเขียนเครื่องหมายในการดำเนินการ หลายตัวในนิพจน์ เดียวกัน จะมีลำดับของการดำเนินการอย่างไร วิธีการจัดลำดับการดำเนินการจะจัดลำดับ จากซ้ายไปขวา หรือขวาไปซ้าย ขึ้นอยู่กับตัวดำเนินการแต่ละตัว แสดงไว้ในตารางที่ 2.9

ตารางที่ 2.9 ลำดับการทำงานของตัวดำเนินการ

ชนิดของตัวดำเนินการ	ตัวดำเนินการ	ทิศทางการดำเนินการ
Expression	( ) [ ] ->	ซ้ายไปขวา
Unary	- + ~ ! * & ++ --	ขวาไปซ้าย
	sizeof (type)	
	*(deraference)	
	typecast	
Member selection	.,->	ขวาไปซ้าย
Pointer to member	.*,->*	ขวาไปซ้าย
การคำนวณทางด้านการคูณ, หาร	* / %	ซ้ายไปขวา
การบวก	+ -	ซ้ายไปขวา
การเลื่อนระดับบิต	<< >>	ซ้ายไปขวา
Relational (inequality)	<<= >>=	ซ้ายไปขวา
Relational (equality)	== !=	ซ้ายไปขวา
ระดับบิต	&	ซ้ายไปขวา
ระดับบิต	^	ซ้ายไปขวา
ระดับบิต	|	ซ้ายไปขวา
ทางตรรกะ	&&	ซ้ายไปขวา
ทางตรรกะ	||	ซ้ายไปขวา
การตัดสินใจ	?:	ขวาไปซ้าย
การกำหนดค่า	=*=/=%=	ขวาไปซ้าย
	+= -= <<= =>>=&=|=^=	
Sequential Evaluation		ซ้ายไปขวา

ที่มา (เกษมสันต์ พานิชการ, 2537, หน้า 41)

ในการแสดงข้อมูล เราสามารถ แสดงข้อมูลในรูปแบบต่าง ๆ ได้ หลายรูปแบบ นอกจากการแสดงข้อมูลในแบบปกติ เช่น การแสดงในรูปแบบของเลขฐาน การแสดงเลขยกกำลัง เป็นต้น การดำเนินการเกี่ยวกับการแสดงผลข้อมูล แสดงไว้ในตารางที่ 2.10

ตารางที่ 2.10 ตัวดำเนินการเกี่ยวกับการแสดงข้อมูล

ตัวกำหนดชนิดข้อมูล	ความหมาย
%c	แทนตัวอักษร (Character)
%d	แทนเลขจำนวนเต็ม (Decimal)
%e	แทนเลขยกกำลัง (Exponential form)
%f	แทนเลขทศนิยม (Floating point)
%h	แทนเลขจำนวนเต็มชนิดสั้น (short) ไม่ค่อยนิยมใช้
%p	แทนข้อมูลแบบพอยน์เตอร์ (Pointer)
%s	แทนข้อความ (String)
%u	แทนเลขจำนวนเต็มไม่คิดเครื่องหมาย (Unsigned)
%o	แทนเลขฐานแปด (Octal)
%x	แทนเลขฐานสิบหก (ไม่มีเครื่องหมาย) ตัว a – f
%X	แทนเลขฐานสิบหก (ไม่มีเครื่องหมาย) ตัว A – F
%%	แทนเครื่องหมาย %

ที่มา (Horsington, 1991, p. 238)

การดำเนินการในการแสดงแบบพิเศษ มีลักษณะคล้ายกับ ตัวดำเนินการแสดงข้อมูล ตัวดำเนินการนี้ จะแสดงผลในแบบพิเศษ เช่น การขึ้นบรรทัดใหม่ การเว้นระยะในการแสดงผล หรือ การส่งเสียงออกมาจากลำโพงเป็นต้น รายละเอียดของตัวดำเนินการแบบพิเศษแสดงไว้ในตารางที่ 2.11

ตารางที่ 2.11 ตัวดำเนินการในการแสดงแบบพิเศษ

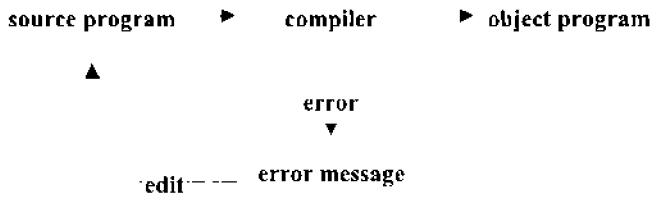
ตัวอักษรพิเศษ	ค่าของเลขฐาน 16	ตัวอักษร	ความหมาย
\a	0 × 07	BEL	Audible bell
\b	0 × 08	BS	Backspace
\f	0 × 0c	FF	Formfeed
\n	0 × 0A	LF	Newline (linefeed)
\r	0 × 0D	CR	Carriage return
\t	0 × 09	HT	Tab (horizontal)
\v	0 × 0B	VT	Vertical tab
\\	0 × 5C	\	Backslash
\'	0 × 27	'	Single quote (apostrophe)
\"	0 × 22	"	Double Quote
\?	0 × 3F	?	Question mark
\0		any	0 = a string of up to three octal digits
\xH		any	H = a string of hex digits

ที่มา (Borland International , 1992, p. 15)

การแปลโค้ดของภาษาซี

การแปลโค้ดของภาษาซี หมายถึง ลำดับของการแปลภาษาเพื่อให้ได้ ความหมายไปสั่งเครื่องคอมพิวเตอร์ให้ทำงานตามที่เราต้องการ ซึ่งภาษาซี ใช้วิธีการแปลแบบ คอมไพเลอร์ คือแปลให้หมดโปรแกรม ก่อนจึงทำงานตามโค้ดที่ได้จากการแปลนั้น ๆ โปรแกรมที่เขียนด้วยภาษาระดับสูงอย่างเช่น ภาษาซีนี้ เราเรียกว่า ซอร์สโปรแกรม (source program) หลังจากกระบวนการแปลภาษาแล้ว เราจะได้ โปรแกรมที่เป็นภาษาระดับต่ำหรือภาษาเครื่องซึ่งเราเรียกว่า ออบเจคโปรแกรม (object program) ขณะที่ทำการแปลภาษาด้วยคอมไพเลอร์นั้น ถ้าหากพบข้อผิดพลาด

คอมไพเลอร์จะไม่สามารถทำการแปลให้เป็น ออบเจคโปรแกรมได้ แต่จะมีการส่งข่าวสารการผิดพลาด (error message) ออกมา ให้ผู้เขียนโปรแกรมได้นำเอาไปทำการแก้ไขก่อน แล้วจึงทำการแปลอีกครั้ง จนกว่าจะไม่พบข้อผิดพลาดในโปรแกรม จึงจะสามารถสร้าง ออบเจคโปรแกรมได้ กระบวนการแปลภาษาด้วยคอมไพเลอร์โดยทั่วไป แสดงได้ดังภาพที่ 2.2



ภาพที่ 2.2 กระบวนการแปลภาษาด้วยคอมไพเลอร์

ลำดับการแปลโค้ดของโปรแกรมภาษาซี จะเริ่มจากการแปล ส่วนหัวของโปรแกรมซึ่งเรียกว่า header หรือ preprocessor เป็นส่วนที่ใช้สำหรับประกาศค่าต่าง ๆ ที่จะนำมาใช้ในโปรแกรม รวมไปถึง การนำไลบรารีไฟล์ บางอย่างมาใช้ การประกาศค่าในส่วนนี้ ถือว่าเป็นการประกาศให้เป็นแบบ โกลบอล ในส่วนต่อไป จะทำการแปล คำสั่งที่อยู่ใน ฟังก์ชันเมน หรือ ที่เรียกว่าตัวโปรแกรมในภาษาซี และสุดท้ายจึงทำการแปล ฟังก์ชันอื่น ๆ ที่ผู้เขียนโปรแกรมเขียนไว้ ซึ่งอาจจะมี หรือไม่มีก็ได้ ลำดับขั้นตอนการแปลโค้ดของภาษาซี เขียนเป็นไดอะแกรมได้ดังภาพที่ 2.3

ภาพที่ 2.3 แผนผังลำดับขั้นตอนการแปลของภาษาซี

ลำดับการแปลในภาษาซี ดังที่กล่าว อาจจะมีข้อแตกต่าง ไปจากที่กล่าวไว้บ้าง ในกรณีที่ผู้เขียนโปรแกรม ได้เขียนฟังก์ชันอื่น ๆ ไว้ ก่อนที่จะถึงฟังก์ชันเมน เพราะภาษาซีสามารถเขียนฟังก์ชันอื่น ๆ ไว้ได้ 2 ตำแหน่ง ถ้าเขียน ฟังก์ชันอื่นก่อน เครื่องก็จะแปลฟังก์ชันอื่นก่อน ตามลำดับ เพื่อให้ง่ายแก่การทำความเข้าใจการแปลภาษา ในโปรแกรมภาษาซี จะมีการแปลภาษา จากบนลงล่าง ดังนั้น ถ้ามีการเรียกใช้ฟังก์ชันใดก็ตาม ฟังก์ชันนั้น จะต้องเขียนก่อน ที่จะถูกเรียกใช้เสมอ

สรุป

ภาษาซีที่เราจะใช้งานในเอกสารนี้ เป็น ภาษาซีพลัสพลัส ของบริษัท บอร์แลนด์ จำกัด ภาษาซี มีความอ่อนตัวในการนำไปใช้งาน เพราะสามารถใช้ได้กับคอมพิวเตอร์ทุกขนาด และให้ประสิทธิภาพในการใช้งานสูง โครงสร้างโปรแกรมในภาษาซีประกอบด้วย ส่วนหัว ซึ่งแบ่งเป็นส่วนกำหนดค่าต่าง ๆ และ ส่วนที่ใช้เรียก ไลบรารีไฟล์ต่าง ๆ เข้ามาใช้งานร่วมกับโปรแกรมส่วนตัวโปรแกรมจะเรียกว่า ฟังก์ชันเมน ซึ่งลักษณะของภาษาซีนั้น จะสามารถแบ่งโปรแกรมออกเป็นส่วนย่อย ๆ ได้ในรูปแบบของ ฟังก์ชัน ดังนั้นฟังก์ชันเมน จึงเป็นฟังก์ชันหลัก

ภาษาซี ได้กำหนดให้มีการนำข้อมูลข้อมูลมาใช้เพื่อให้สอดคล้องกับการใช้ข้อมูลในชีวิตประจำวันของมนุษย์ ซึ่งประกอบด้วย ข้อมูลประเภทข้อความ ข้อมูลประเภทตัวเลข และข้อมูลประเภทตรรกศาสตร์ ในการนำข้อมูลไปใช้งาน สามารถใช้โดยทางตรง เรียกว่าใช้แบบค่าคงที่ และใช้แบบทางอ้อม เรียกว่าใช้แบบตัวแปร นอกจากนั้นยังสามารถดำเนินการต่าง ๆ กับข้อมูลโดยใช้เครื่องหมายในการดำเนินการ ซึ่งมีหลายประเภท ตามชนิดของข้อมูลที่จะดำเนินการ การดำเนินการต่าง ๆ มีความสามารถพิเศษนอกเหนือไปจากการสั่งงานตามปกติ ซึ่งมีการแสดงในแบบพิเศษอยู่หลายแบบ

ภาษาซี ใช้การแปลภาษาแบบคอมไพเลอร์ การดำเนินการแปลโปรแกรม จะเริ่มจากการแปลจากส่วนหัวของโปรแกรม ต่อจากนั้นจึงทำการแปลส่วนของฟังก์ชันเมน และถ้าหากมีการกำหนดฟังก์ชันอื่นขึ้นมาใช้ ก็จะทำการแปลตามลำดับก่อนหลัง

คำถามทบทวน

จงตอบคำถามต่อไปนี้

1. header ในโครงสร้างโปรแกรมของภาษาซี มีหน้าที่ และประโยชน์อย่างไร
2. declaration ในโครงสร้างโปรแกรมภาษาซี มีด้วยกัน 2 ส่วน คือ global และ local จงบอกถึงประโยชน์และข้อแตกต่างของแต่ละส่วน
3. return value ในโครงสร้างโปรแกรมภาษาซี หมายถึงอะไร ทำไมต้องมีส่วนนี้ จงอธิบาย
4. ข้อมูลชนิด integer กับ floating point แตกต่างกันอย่างไร จงอธิบาย
5. ข้อมูลชนิด character กับ string แตกต่างกันอย่างไร จงอธิบาย
6. การใช้ข้อมูล แบบค่าคงที่ และ การใช้ข้อมูลแบบตัวแปร มีข้อแตกต่างกันอย่างไร จงอธิบาย
7. ตัวดำเนินการทางคณิตศาสตร์ต่อไปนี้มีความหมายอย่างไร จงอธิบายพร้อมยกตัวอย่าง

   a. + b. -+ c. – d. –

   e. * f. / g. %
8. ตัวดำเนินการเกี่ยวกับการแสดงข้อมูล ต่อไปนี้มีความหมายอย่างไร จงอธิบาย พร้อมยกตัวอย่าง

   a. %c b. %s c. %d d. %f

   e. \n f. \t g. \f
9. จงอธิบายถึงการแปลภาษาแบบคอมไพเลอร์
10. การแปลโค้ด ของภาษาซี มีลักษณะเป็นอย่างไรจงอธิบาย

เอกสารอ้างอิง

เกษมสันต์ พานิชการ. (2537). **C++ และหลักการของ OOP ฉบับเริ่มต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

มนตรี พจนารถลาวัณย์. (2535). **การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี.** กรุงเทพฯ: เอช-เอน การพิมพ์.

ธันวา ศรีประโมง. (2539). **การเขียนโปรแกรมภาษาซีสำหรับวิศวกรรม.** กรุงเทพฯ: มหาวิทยาลัยเทคโนโลยีมหานคร.

Horsington Gordon. (1991). **Programming in ANSI Standard C.** Singapore: John Wiley & Sons (SEA) Ptc.

Borland International. (1992). **Programmer's Guide.** Scotts Valley, California: Borland International.

แผนบริหารการสอนประจำบทที่ 3

เนื้อหาประจำบท

ไลบรารีที่สำคัญในการเขียนโปรแกรม

คำสั่งแสดงผล

คำสั่งรับข้อมูล

คำสั่งกำหนดค่า

ฟังก์ชันมาตรฐาน

ฟังก์ชันทางคณิตศาสตร์

ฟังก์ชันเกี่ยวกับตัวอักษร

ฟังก์ชันเกี่ยวกับสตริง

ฟังก์ชันทั่ว ๆ ไป

สรุป

คำถามทบทวน

จุดประสงค์เชิงพฤติกรรม

1. เพื่อให้ผู้เรียนสามารถอธิบายถึงไลบรารีที่สำคัญของภาษาซีได้
2. เพื่อให้ผู้เรียนสามารถเขียนและอธิบายถึงคำสั่งรับข้อมูลได้อย่างถูกต้อง
3. เพื่อให้ผู้เรียนสามารถเขียนและอธิบายถึงคำสั่งแสดงผลได้อย่างถูกต้อง
4. เพื่อให้ผู้เรียนสามารถเขียนและอธิบายถึงคำสั่งกำหนดค่าได้อย่างถูกต้อง
5. เพื่อให้ผู้เรียนสามารถเขียนโปรแกรมที่มีการรับข้อมูล การประมวลผล และการแสดงผลได้อย่างถูกต้อง

วิธีสอนและกิจกรรมการเรียนการสอน

1. สอนแบบบรรยาย นำเข้าสู่บทเรียนด้วยการกล่าวถึงการสั่งให้คอมพิวเตอร์ทำงานด้วยการเขียนโปรแกรม
2. กิจกรรมการเรียนการสอน
   2.1 แสดงตัวอย่างการใช้สั่งงานคอมพิวเตอร์โดยโปรแกรมภาษาซี
   2.2 ฟังบรรยาย
   2.3 ทำแบบฝึกหัดท้ายบท

สื่อการเรียนการสอน

1. เอกสารประกอบการเรียนบทที่ 3
2. เพาเวอร์พอยท์ พรีเซนเตชั่น (power point presentation) ประกอบการบรรยาย

การวัดและการประเมินผล

1. สังเกตจากการตอบคำถาม และการตั้งคำถาม ในชั้นเรียน
2. วัดเจตคติจากการสังเกตพฤติกรรม ความกระตือรือร้นในการทำกิจกรรม
3. ความถูกต้องและคุณภาพของการทำแบบฝึกหัดท้ายบทเรียน

บทที่ 3
คำสั่งพื้นฐานในภาษาซี

ในการเขียนโปรแกรมคอมพิวเตอร์ ไม่ว่าภาษาใดก็ตาม การเรียนรู้ภาษาคอมพิวเตอร์จะต้องเริ่มเรียนรู้ไปจากเรื่องพื้นฐาน เช่น อักขระที่ใช้ในภาษาต่างๆ จนถึงสามารถเขียนโปรแกรมได้ ในบทนี้จะกล่าวถึงการใช้คำสั่งพื้นฐานที่จำเป็นที่จะต้องใช้ในภาษาซี ซึ่งได้แก่ คำสั่งที่ใช้ในการรับข้อมูล คำสั่งที่ใช้ในด้านการประมวลผล และ คำสั่งที่ใช้ในด้านการแสดงผล คำสั่งในภาษาซีนั้นเราสามารถ สร้างคำสั่งขึ้นมาใช้งานเองได้ หรือเรียกใช้คำสั่งที่มีอยู่แล้ว คำสั่งที่มีอยู่แล้ว จะถูกเก็บอยู่ในไฟล์ ที่เรียกว่า ไลบรารี ในไฟล์ประเภทนี้จะรวบรวมเอาคำสั่ง และ ฟังก์ชันต่าง ๆ แล้วแบ่งประเภทออกตามการใช้งาน เช่น ใช้งานเกี่ยวกับจอภาพ ใช้งานเกี่ยวกับแฟ้มข้อมูล หรือ ใช้งานเกี่ยวกับการรับและการแสดงผลข้อมูลเป็นต้น ในบทนี้ จะกล่าวถึง ไลบรารี ที่สำคัญในการใช้งานทั่วไป การรับข้อมูล การแสดงผลข้อมูล และ การกำหนดค่าให้กับ ข้อมูล ซึ่งคำสั่งกำหนดค่านี้ จะรวมไปถึงการประมวลผลต่าง ๆ ด้วย

ไลบรารี่ที่สำคัญในการเขียนโปรแกรม

สิ่งที่ขาดไม่ได้ในการเขียนโปรแกรมแต่ละครั้ง คือการเรียกใช้คำสั่ง หรือ ฟังก์ชัน ที่อยู่ในไลบรารีไฟล์ เนื่องจากในไลบรารีไฟล์เหล่านี้ ได้รวบรวมเอาคำสั่ง สำเร็จที่มีการใช้งานบ่อย ๆ เอาไว้ให้ใช้งานอย่างมากมาย ถ้าไม่มีความจำเป็นผู้เขียนโปรแกรม จะไม่มีการสร้างคำสั่งขึ้นมาใช้งานเอง แต่จะใช้วิธีการเรียกใช้คำสั่ง หรือ ไลบรารีไฟล์ ที่มีอยู่แล้ว เหล่านั้น ในส่วนหัวของโปรแกรม เว้นเสียแต่ว่า ฟังก์ชันที่จะเรียกใช้นั้น เป็นฟังก์ชันเฉพาะงาน ผู้เขียนโปรแกรมก็สามารถเขียนฟังก์ชันเหล่านั้น ขึ้นมาเพื่อใช้งานได้ตามจุดประสงค์ ไลบรารีจึงเป็นส่วนหนึ่งของ การเขียนโปรแกรมด้วยภาษาซี มีผู้ให้คำจำกัดความของไลบรารีในภาษาซี ดังนี้

เบญจพร ศักดิ์ศิริ (2540, หน้า 20) กล่าวว่า ส่วนหัวของโปรแกรมในภาษาซี เป็นส่วนที่โปรแกรมจะทำการ คอมไพล์ ก่อนส่วนอื่น ในส่วนนี้ เราสามารถเรียกใช้คำสั่ง สำเร็จที่สำคัญ ๆ ซึ่งมีการเขียนคำสั่งไว้แล้ว เรียกว่า ไลบรารีไฟล์ หรือ ในภาษาซี หมายถึง ไฟล์ที่มีสกุลเป็น h เช่น stdio.h conio.h iostream.h เป็นต้น ไลบรารีไฟล์เหล่านี้ จะถูกเรียกมาใช้ตามแต่จะเห็นสมควร ว่าภายในโปรแกรม ต้องการใช้ไลบรารีที่เกี่ยวข้องกับเรื่องใด ตัวอย่างเช่น ต้องการให้มีการรับข้อมูลจากแป้นพิมพ์ หรือ ต้องการแสดงผลลัพธ์ทางจอภาพ ก็จะต้อง เรียกใช้ไลบรารีไฟล์ที่เกี่ยวข้องกับเรื่องนั้น ๆ เพื่อให้เครื่องได้รู้จักคำสั่งที่จะเรียกใช้ต่อไป ไลบรารีที่สำคัญที่นิยมใช้กันมีดังนี้

stdio.h เป็นไลบรารีที่มีคำสั่งที่เกี่ยวข้องกับ รับเข้า และ ส่งออก คำสั่งที่เราจะใช้ ในไลบรารี นี้คือ คำสั่งรับข้อมูลด้วย scanf และ แสดงผลลัพธ์ด้วย printf

iostream.h เป็นไลบรารีที่รวบรวมคำสั่งที่เกี่ยวข้องกับ การรับเข้า และการส่งออกข้อมูล ในลักษณะต่อเนื่อง (stream) หรือ เปรียบเสมือน ท่อส่งน้ำ คำสั่งที่เกี่ยวข้องที่เราจะใช้มี 3 คำสั่ง คือ clrscr() เป็นฟังก์ชันที่ใช้ลบจอภาพ cin หมายถึงการนำเข้าข้อมูล และ cout หมายถึง การแสดงผลลัพธ์

conio.h เป็นไลบรารีที่เกี่ยวข้องกับ การรับเข้า และ การส่งออกข้อมูล ในอีกลักษณะหนึ่ง คำสั่งที่เราจะนำมาใช้คือ getch() หมายถึงการรับข้อมูลเข้า 1 อักษร โดยไม่มีการแสดงผล และ gotoxy() หมายถึงการกำหนดตำแหน่งรับหรือแสดงข้อมูลในตำแหน่งที่ต้องการบนจอภาพ

math.h เป็นไลบรารีที่เกี่ยวข้องการหาค่าทางคณิตศาสตร์ เช่น การหาค่าตัวเลขยกกำลังด้วย pow() การหาค่ารากที่สอง ด้วย sqrt(x) เป็นต้น

ctype.h เป็นไลบรารีที่เกี่ยวข้องกับการจัดการตัวอักษร เช่น การเปลี่ยนให้เป็นตัวพิมพ์ใหญ่ด้วย tolower() การเปลี่ยนให้เป็นตัวพิมพ์เล็กด้วย toupper() เป็นต้น

string.h เป็นไลบรารีที่เกี่ยวข้องกับการจัดการข้อความ เช่น การเปรียบเทียบข้อความด้วย strcmp() หาความยาวของข้อความด้วย strlen() เป็นต้น

จากจุดประสงค์ และความหมายของไลบรารีไฟล์ในภาษาซี สรุปได้ว่า ถ้ามองอีกมุมหนึ่ง ก็พูดได้ว่า ไลบรารีไฟล์ ก็คือ คลังของฟังก์ชันต่าง ๆ ที่สามารถเรียกขึ้นมาใช้งานได้ จากส่วนหัวของโปรแกรม ไลบรารีไฟล์จะแตกต่างกันที่ จุดประสงค์ ของการสั่งงาน เช่น ไลบรารีเกี่ยวกับอุปกรณ์รับและแสดงผลข้อมูล หรือ ไลบรารีเกี่ยวกับ แฟ้มข้อมูล เป็นต้น ไลบรารีไฟล์รายละเอียดของคำสั่ง หรือฟังก์ชัน ในไลบรารีต่าง ๆ จะกล่าวในหัวข้อต่อไป

คำสั่งแสดงผล

คำสั่งแสดงผล หมายถึง คำสั่งที่สั่งให้เครื่องทำการแสดงผล ข้อมูลที่ต้องการ ออกมาทางจอภาพ ในการแสดงผล สามารถแสดงผลได้ทั้ง ตัวเลข และ ตัวอักษร คำสั่งที่ใช้สำหรับการแสดงผลในภาษาซี ที่จะกล่าวในที่นี้ มี 2 คำสั่ง คือ printf และ cout ซึ่งมีรายละเอียดดังต่อไปนี้

คำสั่ง printf เป็นคำสั่ง ที่อยู่ใน ไลบรารี stdio.h มีรูปแบบดังนี้

```
printf("control string",variable list);
```

เมื่อคำสั่งนี้ทำงาน จะทำการแสดงข้อมูลออกไปทางจอภาพ โดยที่ข้อมูลนั้น อาจเป็นค่าคงที่ หรือ ตัวแปรก็ได้ ในการแสดงผลนั้นข้อมูลจะมีการแสดงผลออกไปตามสัญลักษณ์ใน control string ซึ่งมีรายละเอียดดังแสดงในตารางที่ 3.1

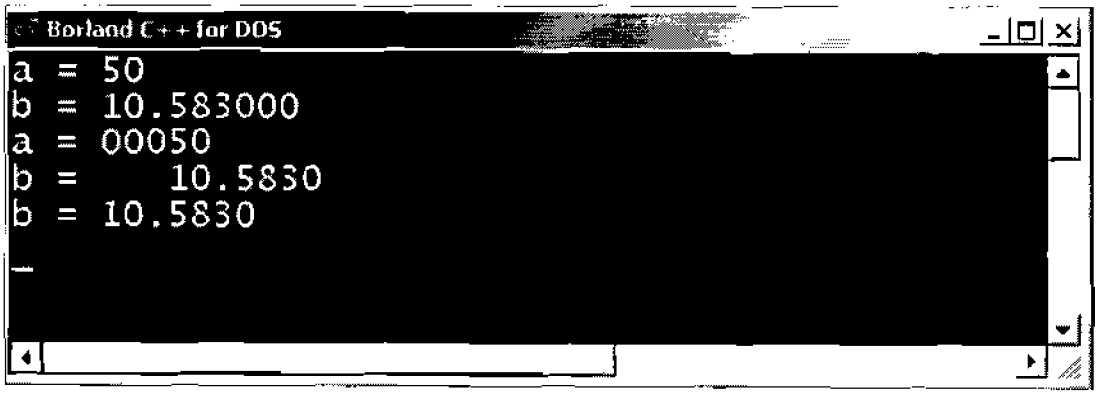
ตารางที่ 3.1 สัญลักษณ์ control string ที่ใช้ในการแสดงผล

Control string	ความหมาย
%c	ข้อมูลชนิดตัวอักษร
%d	ข้อมูลชนิดตัวเลขจำนวนเต็ม
%e	ข้อมูลชนิดตัวเลขแบบวิทยาศาสตร์
%f	ข้อมูลชนิดตัวเลขทศนิยม
%o	ข้อมูลชนิดตัวเลขฐานแปด
%s	ข้อมูลชนิดข้อความ
%u	ข้อมูลชนิดตัวเลขจำนวนเต็ม ไม่คิดเครื่องหมาย
%x	ข้อมูลชนิดตัวเลขฐานสิบหก
%p	ข้อมูลชนิดพอยน์เตอร์
%%	ใช้เครื่องหมาย %

ที่มา (Horisington, 1991, p.238)

โปรแกรมที่ 3.1 ตัวอย่างในการแสดงผลข้อมูล ในแบบของตัวเลขจำนวนเต็ม โดยใช้สัญลักษณ์ %d และ จำนวนจริง ด้วยสัญลักษณ์ %f หลังจากแสดงข้อมูลในแต่ละบรรทัด ให้ขึ้นบรรทัดใหม่ด้วย สัญลักษณ์ \n

```
1   #include <stdio.h>
2   int main ()
3   { int a;
4     float b;
5     a = 50;
6     b = 10.583;
7     printf("a – %d\n",a);
8     printf("b = %f\n",b);
9     printf("a – %05d\n",a);
10    printf("b = %10.4f\n",b);
11    printf("b = %-10.4f\n",b);
12    return 0;
13  }
```



ภาพที่ 3.1 ผลลัพธ์ของโปรแกรมที่ใช้สัญลักษณ์ %d และ %f

จากผลลัพธ์ของโปรแกรม การแสดงผลในบรรทัดที่ 8 และ บรรทัดที่ 9 ไม่มีการกำหนดขนาดของข้อมูลที่จะแสดง จะเห็นได้ว่า %d จะแสดงตัวเลขจำนวนเต็ม เท่าที่มีอยู่ ส่วน %f จะแสดงออกมาในขนาดความกว้างเท่ากับ 9 จำนวนเลขหลังจุดทศนิยม จะเพิ่มขึ้นโดยเติมเลข ศูนย์เข้าไปให้ครบจำนวน 9 ตัว บรรทัดที่ 10 ระบุจำนวนว่า ให้แสดงตัวเลขออกมาในจำนวน 5 หลัก และถ้าค่าของตัวเลขไม่ถึง 5 หลัก ให้เติม 0 ลงไปข้างหน้า บรรทัดที่ 11 ระบุขนาดทั้งหมด 10 หลัก นับรวมจุดทศนิยม ผลลัพธ์คือ แสดงหน้าจุด 5 หลัก และ จำนวนเลขหลังจุดทศนิยมเท่ากับ 4 หลัก ส่วนบรรทัดที่ 12 ระบุเหมือนบรรทัดที่ 11 แต่เติมเครื่องหมาย ลบ เข้าไป หมายถึงให้ตัดช่องว่างข้างหน้าออกในกรณีที่จำนวนหลักของตัวเลขหน้าจุด มีน้อยกว่าค่าที่กำหนด ในที่นี้คือ กำหนดให้เลขหน้าจุดทศนิยมจำนวน 5 หลัก แต่จำนวนที่จะแสดงมีแค่ 2 หลัก ดังนั้นช่องว่างที่อยู่ด้านหน้าจึงถูกตัดออก

นอกจากสัญลักษณ์ ที่เรียกว่า control string แล้ว ในตัวอย่าง ยังพบ สัญลักษณ์ที่ขึ้นต้นด้วยเครื่องหมาย backslash จากตัวอย่าง \n มีความหมายว่า ให้ขึ้นบรรทัดใหม่ นอกจากนี้แล้ว ยังมีสัญลักษณ์อื่น ๆ อีก ดังรายละเอียดในตารางที่ 3.2

ตารางที่ 3.2 สัญลักษณ์พิเศษที่ขึ้นต้นด้วยเครื่องหมาย backslash

สัญลักษณ์	ความหมาย
\b	พิมพ์เครื่องหมาย backspace
\f	ขึ้นหน้ากระดาษใหม่
\n	ขึ้นบรรทัดใหม่
\r	พิมพ์เครื่องหมาย return
\t	เว้นช่องว่างทางแนวนอน
\"	พิมพ์เครื่องหมาย "
\'	พิมพ์เครื่องหมาย '
\0	พิมพ์เครื่องหมาย null
\\	พิมพ์เครื่องหมาย backslash
\v	เว้นช่องว่างทางแนวตั้ง
\a	เสียงบี๊บ
\N	แสดงเป็นเลขฐาน 8 (N=จำนวนที่ต้องการแสดง)
\XN	แสดงเป็นเลขฐาน 16 (N=จำนวนที่ต้องการแสดง)

ที่มา (มนตรี พจนาถลาวัณย์, หน้า 91)

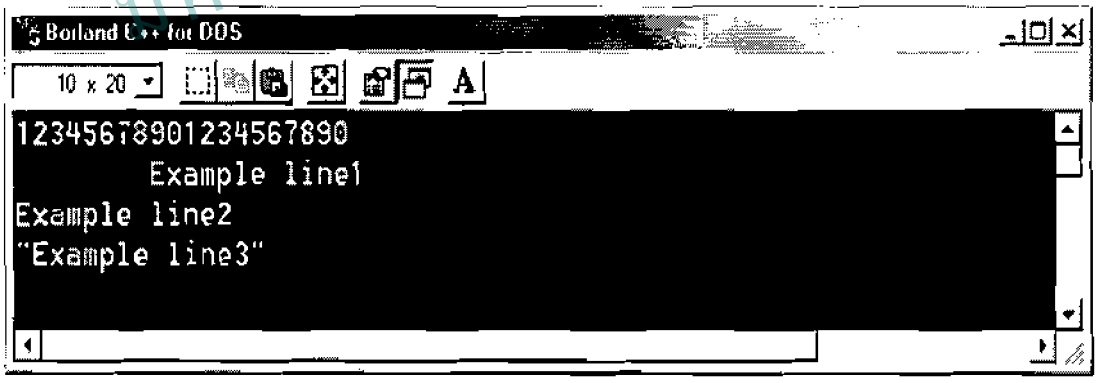
โปรแกรมที่ 3.2 ตัวอย่างการพิมพ์ด้วยสัญลักษณ์พิเศษ ที่ขึ้นต้นด้วย เครื่องหมาย back slash ได้แก่

\n เพื่อขึ้นบรรทัดใหม่

\t เพื่อเว้นระยะในการแสดงผล และ

\a ส่งเสียงบี๊บออกมา

```
1   #include <stdio.h>
2   int main()
3   { printf("123456789012345678901 \n");
4     printf("\t Example line 1 \n");
5     printf("Example line2 \n");
6     printf("\"Example line3\"\n\a");
7     getchar();
8     return 0;
9   }
```



ภาพที่ 3.2 ผลลัพธ์ของโปรแกรมที่มีการใช้สัญลักษณ์พิเศษ

ผลลัพธ์ของโปรแกรมตัวอย่าง มีการใช้ \n หมายถึงขึ้นบรรทัดใหม่ \t หมายถึง เว้นระยะการพิมพ์แนวนอน จะเห็นว่า ข้อความ Example line1 เว้นระยะการพิมพ์มาจากทางด้านซ้าย \" หมายถึง การพิมพ์ เครื่องหมาย " และ \a หมายถึง การส่งเสียง บี๊บ

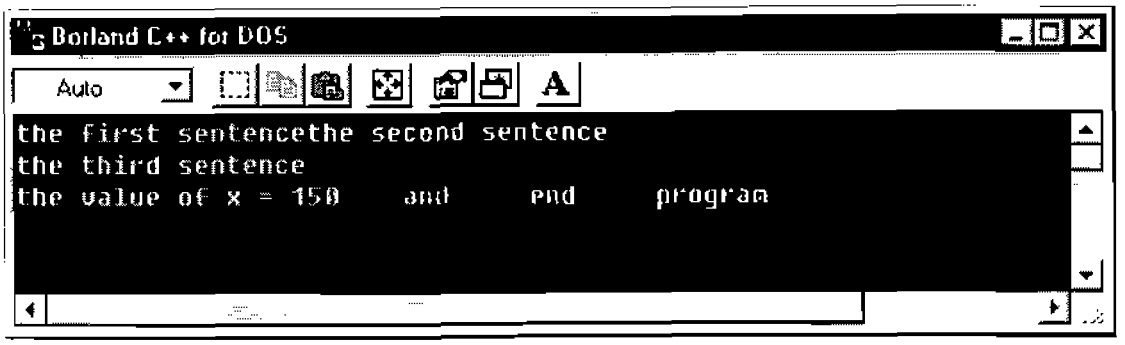
คำสั่ง cout เป็นคำสั่งที่อยู่ในไลบรารี iostream.h มีรูปแบบดังนี้

```
cout << data1[,<<data2,...];
```

คำสั่ง cout เป็นคำสั่ง แสดงผลอีกคำสั่งหนึ่ง มีการทำงานในลักษณะ การส่งออกข้อมูลโดยวิธีการส่งแบบใช้กระแสการไหลของข้อมูล หรือเรียกว่า สตรีม (stream) ทำการลำเลียงข้อมูลออกไปที่อุปกรณ์แสดงผลลัพธ์ทีละตัวจนหมด กล่าวโดยสรุปก็คือทำการแสดงผลข้อมูล ที่อยู่หลังเครื่องหมาย << ดังตัวอย่างโปรแกรมที่ 3.3

โปรแกรมที่ 3.3 ตัวอย่างการแสดงข้อมูลด้วยคำสั่ง cout ร่วมกับสัญลักษณ์ \n เพื่อขึ้นบรรทัดใหม่ และ \t เพื่อเว้นระยะในการพิมพ์

```
1   #include <iostream.h>
2   #include <conio.h>
3   int x= 150;
4   int main()
5   {
6     clrscr();
7     cout << "the first sentence";
8     cout << "the second sentence \n";
9     cout << "the third sentence \n";
10    cout << "the value of x = " << x << "\tand\tend\tprogram"  ;
11    getch();
12    return 0;
13  }
```



ภาพที่ 3.3 ผลลัพธ์ของโปรแกรมที่ใช้คำสั่ง cout

จากโปรแกรมที่ 3.3 จะเห็นว่า เรายังสามารถใช้ สัญลักษณ์พิเศษในการควบคุมการแสดงผล เหมือนกันกับ คำสั่ง printf เช่น \n ที่หมายถึงการขึ้นบรรทัดใหม่ \t ที่หมายถึงเว้นระยะการพิมพ์ในทางแนวนอน และสัญลักษณ์อื่น ๆ

คำสั่งรับข้อมูล

คำสั่งรับข้อมูล หมายถึง คำสั่งที่สั่งให้เครื่อง รับข้อมูล เข้ามาเพื่อดำเนินการอย่างใดอย่างหนึ่งต่อไป การเขียนโปรแกรมแต่ละครั้ง โปรแกรมโดยทั่วไปจะมีการรับข้อมูล ประมวลผล และ การแสดงผล การรับข้อมูลจึงเป็นคำสั่งที่สำคัญอีกคำสั่งหนึ่ง ที่สามารถทำให้ผู้ใช้ ติดต่อกับเครื่องคอมพิวเตอร์ ได้ การรับข้อมูล สามารถรับข้อมูลได้ทั้งแบบที่เป็น ตัวเลข และ ตัวอักษร คำสั่งที่เราใช้รับข้อมูลในภาษาซีมีดังนี้

คำสั่ง scanf อยู่ในไลบรารี stdio.h มีรูปแบบดังนี้

```
scanf ("format code",&var);
```

เมื่อคำสั่งนี้ถูกสั่งให้ทำงาน เครื่องจะหยุดเพื่อรอรับการป้อนข้อมูลจากผู้ใช้ โดยข้อมูลที่ป้อนจะปรากฏบนจอภาพ เมื่อป้อนข้อมูลเสร็จกด Enter ข้อมูลจะถูกส่งไปเก็บที่ตัวแปร var และข้อมูลที่ป้อนลงไป จะถูกกำหนดรูปแบบโดย format code ซึ่งมีรายละเอียดดังตารางที่ 3.3 ดังนี้

ตารางที่ 3.3 สัญลักษณ์รูปแบบของข้อมูล

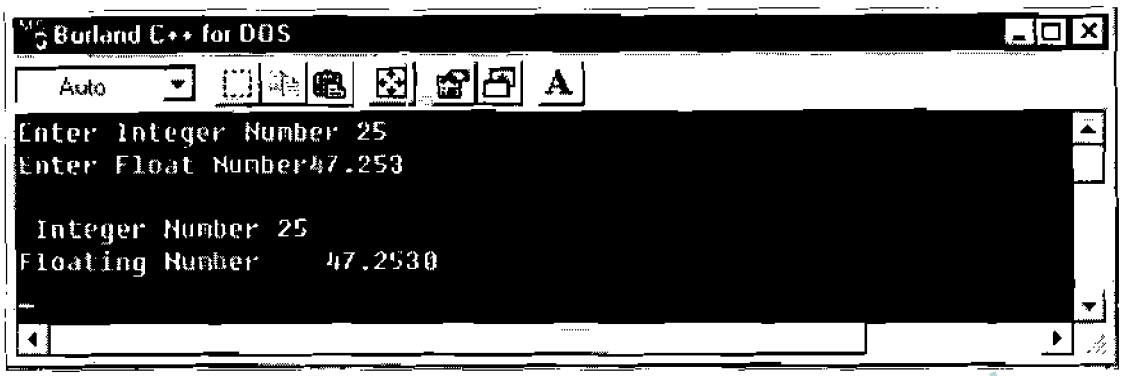
format code	ความหมาย
%c	ข้อมูลชนิดตัวอักษร
%d	ข้อมูลชนิดตัวเลขจำนวนเต็ม
%e	ข้อมูลชนิดตัวเลขแบบวิทยาศาสตร์
%f	ข้อมูลชนิดตัวเลขทศนิยม
%o	ข้อมูลชนิดตัวเลขฐานแปด
%s	ข้อมูลชนิดข้อความ
%u	ข้อมูลชนิดตัวเลขจำนวนเต็ม ไม่คิดเครื่องหมาย
%x	ข้อมูลชนิดตัวเลขฐานสิบหก
%p	ข้อมูลชนิดพอยน์เตอร์
%%	ใช้เครื่องหมาย %

ที่มา (Horisington, 1991, p.238)

โปรแกรมที่ 3.4 ตัวอย่างการใช้คำสั่ง scanf เพื่อรับข้อมูลประเภทตัวเลข ด้วยสัญลักษณ์ %d เพื่อรับข้อมูลตัวเลขจำนวนเต็ม %f เพื่อรับข้อมูลประเภทตัวเลขจำนวนจริง และมีการแสดงผลลัพธ์ด้วยคำสั่ง printf

```
1   #include <stdio.h>
2   #include <conio.h>
3   main()
4   {  clrscr();
5      int n;
6      float x;
7      printf("Enter Integer Number ");
8      scanf("%d",&n);
9      printf("Enter Float Number");
10     scanf("%f",&x);
11     printf("\n Integer Number %d\n",n);
```

```
12      printf("Floating Number %10.4f\n",x);
13      getch();
14  }
```



ภาพที่ 3.4 ผลลัพธ์ของโปรแกรมที่มีการใช้คำสั่ง scanf

จากโปรแกรมตัวอย่าง ใช้คำสั่ง scanf ในการรับข้อมูลจากแป้นพิมพ์ ในบรรทัดที่ 7 ใช้สัญลักษณ์ พิเศษในการรับข้อมูล ได้แก่ %d หมายถึง การรับข้อมูลประเภทตัวเลขจำนวนเต็ม และรับข้อมูลประเภทตัวเลขจำนวนจริง ด้วยสัญลักษณ์ %f ที่บรรทัดที่ 9 ส่วนบรรทัดที่ 10 และ 11 เป็นการแสดงข้อมูลที่อยู่ในตัวแปรออกมา ยังมีสัญลักษณ์ พิเศษในการรับข้อมูลอีก หลายสัญลักษณ์ ซึ่งมีลักษณะการรับข้อมูลที่แตกต่าง กันไป

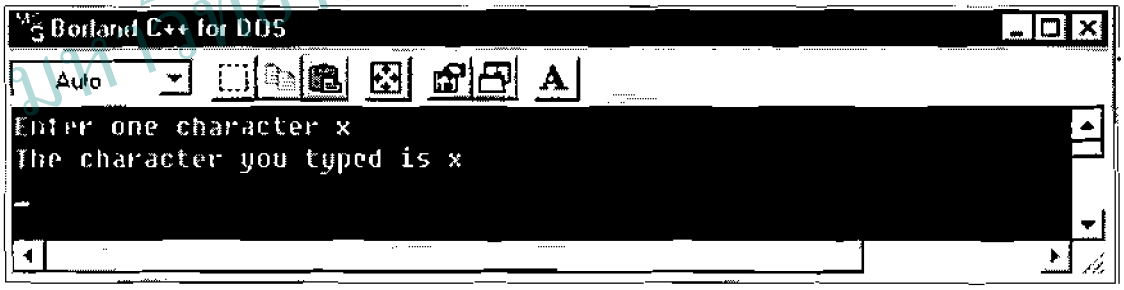
คำสั่ง getchar() อยู่ในไลบรารี stdio.h มีรูปแบบดังนี้

```
ch = getchar();
```

คำสั่ง getchar() จะทำให้เครื่องหยุดรับข้อมูลแบบตัวอักษร 1 ตัว หลังจากกด enter เครื่องจะนำค่าของตัวอักษรที่ ผู้ใช้ป้อนลงไปมาเก็บไว้ที่ ตัวแปร ch ซึ่งเป็นตัวแปรชนิดตัวอักษร คำสั่งนี้ เหมาะสำหรับการรับข้อมูลที่ เป็นตัวอักษร เพียง 1 ตัว เช่น ตัวอักษรย่อของเพศ ได้แก่ m ย่อมาจาก male หรือ f ย่อมาจาก female การสอบถามผู้ใช้ว่าต้องการดำเนินการต่อไปหรือไม่ ได้แก่ y ย่อมาจาก yes หรือ n ย่อมาจาก no เป็นต้น

โปรแกรมที่ 3.5 ตัวอย่างการใช้คำสั่ง getchar() เพื่อรับข้อมูล 1 ตัวอักษร แล้ว แสดงข้อมูลที่ป้อนเข้าไปออกมา

```
1   #include <stdio.h>
2   #include <conio.h>
3   main()
4   { clrscr();
5     char c;
6     printf("Enter one character ");
7     c = getchar();
8     printf("The character you typed is %c\n",c);
9     getch();
10  }
```



ภาพที่ 3.5 ผลลัพธ์ของโปรแกรมที่ใช้คำสั่ง getchar()

จากผลลัพธ์ของโปรแกรม เมื่อโปรแกรมทำงาน เราได้ทดลองป้อนข้อมูล ได้แก่ตัว x ลงไป แล้วกด enter เครื่องก็จะนำเอาค่าที่ป้อนลงไปมาแสดงผล ดังกล่าว ถึงแม้ว่า เราจะป้อนข้อมูลที่มีความยาวมากกว่า 1 ตัวก็ตาม เครื่องก็จะเก็บข้อมูลให้เพียง 1 ตัวเท่านั้น

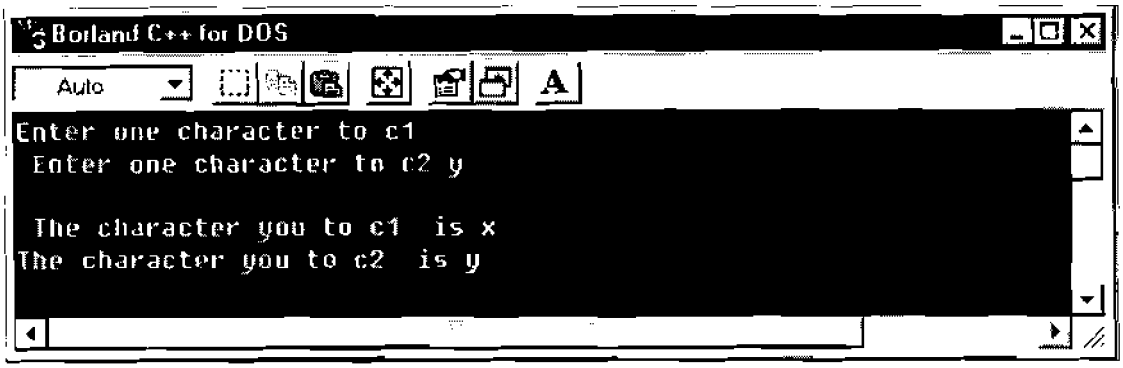
คำสั่ง getch() และ **คำสั่ง getche()** เป็นคำสั่งที่อยู่ใน ไลบรารี conio.h มีรูปแบบดังนี้

```
ch = getch();
ch = getche();
```

ทั้ง 2 คำสั่ง เป็นคำสั่งที่ใช้รับค่าข้อมูลที่เป็นตัวอักษร เมื่อป้อนข้อมูลเสร็จ ไม่ต้องกด enter ซึ่งแตกต่างจากคำสั่งรับข้อมูลที่ผ่านมา ที่ต้องกดแป้น enter เมื่อเครื่องทำงานตามคำสั่งนี้ ก็จะมีการรอรับข้อมูลจากแป้นพิมพ์ ผู้ใช้เพียงแค่กดแป้นพิมพ์ 1 ครั้ง เครื่องก็จะเก็บข้อมูลที่ป้อนลงในตัวแปร ch ในรูปแบบของข้อมูลประเภทตัวอักษร ข้อแตกต่างระหว่าง คำสั่ง getch() และ getche() ตรงที่ getch() ไม่มีการแสดงผลข้อมูลที่ป้อนลงไป ส่วน getche() มีการแสดงผล

โปรแกรมที่ 3.6 ตัวอย่างการใช้ คำสั่ง getch() และ getche() โปรแกรมนี้จะทำการรับข้อมูลเข้ามาเก็บที่ตัวแปร c1 และ c2 ตามลำดับ ขณะที่โปรแกรมทำงาน ให้สังเกตการรับข้อมูลระหว่างคำสั่ง getch() กับ getche() ว่ามีข้อแตกต่างกันอย่างไร


```
1   #include <stdio.h>
2   #include <conio.h>
3   main()
4   { clrscr();
5     char c1,c2;
6     printf("Enter one character to c1 ");
7     c1 = getch();
8     printf("\n Enter one character to c2 ");
9     c2 = getche();
10    printf("\n\n The character you to c1  is %c\n",c1);
11    printf("The character you to c2  is %c\n",c2);
12    getch();
13  }
```



ภาพที่ 3.6 ผลลัพธ์ของโปรแกรมที่ใช้คำสั่ง getch() และ getche()

จากโปรแกรมตัวอย่างที่ 3.6 เราได้ทดลองป้อนข้อมูล x ให้กับ ตัวแปร c1 และ y ให้กับตัวแปร c2 ขณะที่ป้อนข้อมูลให้กับ c1 นั้นไม่มีการแสดงผล ส่วนการป้อนข้อมูลให้กับ c2 นั้นจะมีการแสดงผล ส่วนคำสั่งที่น่าสนใจอีกคำสั่งหนึ่งก็คือ คำสั่ง getch() ในบรรทัดที่ 12 ไม่มีการนำเอาตัวแปรไปรองรับ เครื่องก็จะหยุดรอรับการกดแป้นพิมพ์เหมือนกันมีไว้เพื่อการหยุดเพื่ออ่านผลลัพธ์ เมื่อมีการกดแป้นพิมพ์จึงดำเนินการต่อไป

คำสั่ง cin เป็นคำสั่งที่อยู่ในไลบรารี iostream.h มีรูปแบบดังนี้

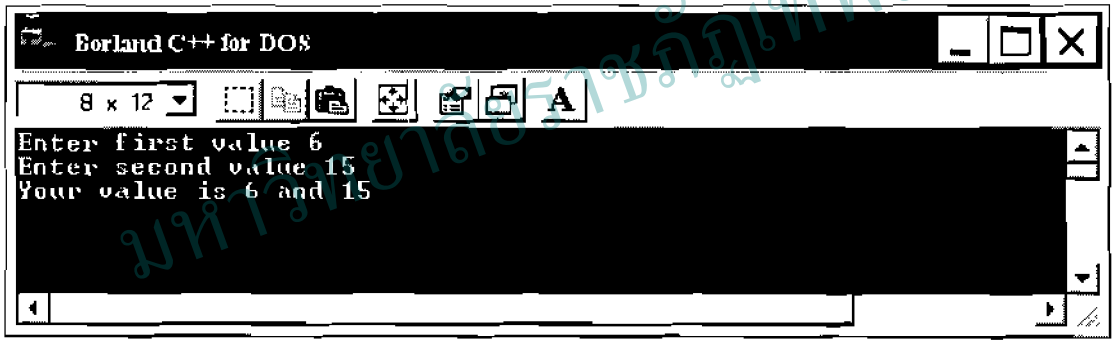
```
cin >> var;
```

คำสั่ง cin เป็นคำสั่ง รับข้อมูลจากแป้นพิมพ์เข้าไปเก็บไว้ในตัวแปร ที่อยู่หลังเครื่องหมาย >> การรับข้อมูลจะเสร็จสิ้นเมื่อผู้ใช้กดแป้น enter แสดงการใช้ดังตัวอย่างโปรแกรมที่ 3.7

```
1   #include <iostream.h>
2   #include <conio.h>
3   int x,y;
4   int main()
5   {
6     clrscr();
7     cout << "Enter first value ";
```

```
8     cin >> x;
9     cout << "Enter second value ";
10    cin >> y;
11    cout << "Your value is " << x <<" and " << y;
12    return 0;
13  }
```

จากตัวอย่าง กำหนดให้มีตัวแปร x และ y เป็นตัวแปรแบบตัวเลขจำนวนเต็ม เมื่อโปรแกรมทำงาน จะมีการแสดงข้อความเพื่อเป็นการบอกให้ผู้ใช้ทราบว่า ให้ใส่ข้อมูล และมีการรับข้อมูลเข้ามาเก็บที่ตัวแปร x และ y ตามลำดับ จากนั้นจึงทำการแสดงผล ข้อมูลที่ได้รับเข้าไปออกมาทางจอภาพ ดังภาพที่ 3.7 ข้อมูลที่ผู้ใช้ใส่เข้าไปได้แก่ 6 และ 15



ภาพที่ 3.7 ผลลัพธ์ของโปรแกรมที่ใช้คำสั่ง cin

คำสั่งกำหนดค่า

คำสั่งกำหนดค่า หมายถึง คำสั่งกำหนดค่าให้กับตัวแปรต่าง ๆ ที่เราต้องการ ค่าที่จะกำหนดนั้น อาจจะเป็นค่าคงที่ หรือ นิพจน์ก็ได้ การกำหนดค่าให้กับตัวแปร โดยนิพจน์ทางคณิตศาสตร์ เครื่องจะทำการหาคำตอบจากนิพจน์นั้น ๆ เสียก่อน แล้วจึงนำคำตอบที่ได้มาเก็บไว้เป็นค่าของตัวแปร ดังนั้นคำสั่งที่ใช้สำหรับการคำนวณหาค่าตัวเลข ในภาษาซี ก็จะเกิดจากคำสั่งกำหนดค่านี้ คำสั่งกำหนดค่ามีรูปแบบดังนี้

variable = constant or expression;

จากรูปแบบคำสั่ง variable ทางด้านซ้ายมือของเครื่องหมาย = จะถูกกำหนดให้มีค่าเป็นค่าคงที่ (constant) หรือ นิพจน์ (expression) ที่อยู่ทางด้านขวามือ เช่น

score – 40; หมายความว่า กำหนดให้ตัวแปร score มีค่าเป็น 50

rate – 2+4; หมายความว่า กำหนดให้ตัวแปร rate มีค่าเป็น 2+4

name – "SUCHADA"; หมายความว่า กำหนดให้ตัวแปร name มีค่าเป็นค่าคงที่แบบตัวอักษรมีค่าเท่ากับ SUCHADA

ดังกล่าวแล้วว่า การกำหนดค่าด้วยนิพจน์ทางคณิตศาสตร์นั้น เป็นคำสั่งที่ใช้ในเรื่องของการคำนวณ คำสั่งนี้จึงมีใช้มากในโปรแกรม ดังนั้นจึงควรรู้เกี่ยวกับ นิพจน์ก่อนว่า มีลักษณะการเขียนอย่างไร

นิพจน์ทางคณิตศาสตร์ของภาษาซี มีรูปแบบดังนี้

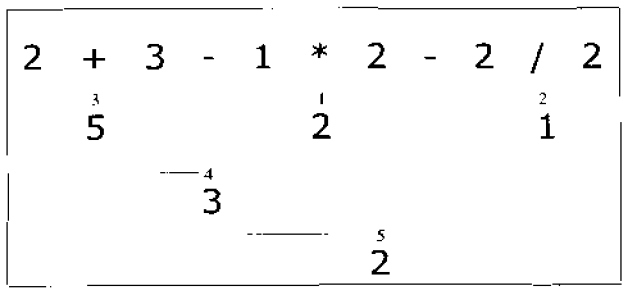
v1 operator v2 [operator v3...]

จากรูปแบบของนิพจน์ v1 หมายถึง ค่าตัวเลขที่ 1 v2 หมายถึง ค่าตัวเลขที่ 2 operator เครื่องหมายทาง คณิตศาสตร์ ซึ่งประกอบด้วย เครื่องหมาย + - * / % +- และ – ดังกล่าวถึงรายละเอียดแล้วในบทที่ 2 และยังสามารถเขียนนิพจน์ให้มีมากกว่า 1 เครื่องหมายได้อีก จากรูปแบบได้แก่ v3 และยังมีอื่นๆ ต่อไปได้อีก ดังนั้นจึงต้องทราบถึง ลำดับความสำคัญของเครื่องหมายคำนวณ ซึ่งมีลำดับความสำคัญดังต่อไปนี้

ลำดับความสำคัญอันดับ 1 ได้แก่ คูณ (*) หาร (/) หารแบบไม่เอาเศษ (%)

ลำดับความสำคัญอันดับ 2 ได้แก่ บวก (+) ลบ (-)

จากลำดับความสำคัญดังกล่าว การคำนวณ ของเครื่องคอมพิวเตอร์ เมื่อเขียนนิพจน์ 2 + 3 – 1 * 2 – 2 / 2 จะมีขั้นตอนในการประมวลผลดังภาพที่ 3.8

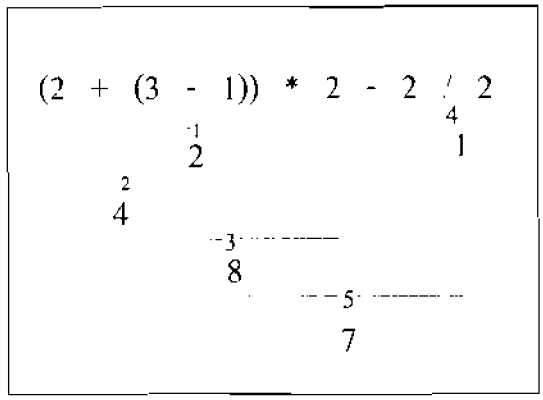


ภาพที่ 3.8 ขั้นตอนการประมวลผลของนิพจน์

จากภาพที่ 3.8 คำตอบที่ได้คือ 2 การประมวลผลจะทำตามลำดับความสำคัญของเครื่องหมาย ถ้าเครื่องหมายอยู่ในระดับเดียวกัน จะทำจากซ้ายไปขวา ในที่นี้ลำดับขั้นตอนในการคำนวณ มีขั้นตอนดังนี้

1. 2 / 2 ได้ผลลัพธ์เท่ากับ 1
2. 1 * 2 ได้ผลลัพธ์เท่ากับ 2
3. 2 + 3 ได้ผลลัพธ์เท่ากับ 5
4. 5 – 3 ได้ผลลัพธ์เท่ากับ 2
5. 3 - 1 ได้ผลลัพธ์เท่ากับ 2

อย่างไรก็ตาม ถ้ามีความจำเป็นที่จะต้องคำนวณนอกเหนือจากกฎเกณฑ์นี้ สามารถใช้วงเล็บ เพื่อให้ทำในวงเล็บก่อนดังภาพที่ 3.9



ภาพที่ 3.9 ขั้นตอนการประมวลผลโดยใส่วงเล็บ

จากภาพที่ 3.9 เมื่อใส่วงเล็บลงไป เครื่องจะทำการประมวลผลในวงเล็บก่อน ถ้าวงเล็บซ้อนวงเล็บ ก็จะทำจากวงเล็บใน ออกไปหาวงเล็บนอก จากตัวอย่างจึงได้คำเท่ากับ 7 ขั้นตอนในการประมวลผล เป็นดังนี้

1. 3 – 1 ได้ผลลัพธ์เท่ากับ 2
2. 2 + 2 ได้ผลลัพธ์เท่ากับ 4
3. 4 * 2 ได้ผลลัพธ์เท่ากับ 8
4. 2 / 2 ได้ผลลัพธ์เท่ากับ 1
5. 8 – 1 ได้ผลลัพธ์เท่ากับ 7

ฟังก์ชันมาตรฐาน

ภาษาซี มีฟังก์ชันในการใช้งานอยู่เป็นจำนวนมาก ฟังก์ชันที่ติดตั้งมากับ เอดิเตอร์ ของภาษาซี เรียกว่า ฟังก์ชันมาตรฐาน เก็บอยู่ในไฟล์ประเภท ไลบรารี สามารถเรียกใช้ได้โดยกำหนดที่ส่วนหัวของโปรแกรม ทำให้สะดวกต่อผู้เขียนโปรแกรม ฟังก์ชันมาตรฐาน แบ่งกลุ่มตามการใช้งาน เช่นใช้สำหรับการรับและแสดงผลข้อมูล การคำนวณทางคณิตศาสตร์ และ วิทยาศาสตร์ เป็นต้น ไลบรารีไฟล์ บางครั้งก็จะเรียกว่า ไฟล์อินคลูด (includc) ซึ่งเราจะต้องระบุไฟล์อินคลูด ก่อนที่จะใช้ฟังก์ชันมาตรฐานนั้น ๆ สำหรับในบทนี้จะได้กล่าวถึงฟังก์ชันมาตรฐานที่ควรจะต้องทราบดังนี้

ฟังก์ชันทางคณิตศาสตร์

ฟังก์ชันทางคณิตศาสตร์เก็บอยู่ในไลบรารีที่ชื่อ math.h สำหรับตัวแปรที่ใช้กับฟังก์ชันประเภทนี้ จะ ต้องเป็นชนิดเลขจำนวนจริงละเอียด 2 เท่า ผลลัพธ์จากฟังก์ชันจะเป็นข้อมูลชนิดเลขจำนานจริงละเอียด 2 เท่า เนื่องจากเป็นฟังก์ชันทางตัวเลข จึงต้องมีความละเอียดในการเก็บค่าตัวเลขต่าง ๆ ค่อนข้างสูง ฟังก์ชันทางคณิตศาสตร์ที่ควรทราบมีดังนี้

acos(x)

ฟังก์ชันนี้จะให้ค่ามุมในหน่วยเรเดียนของค่า arc cosine ของ x

asin(x)

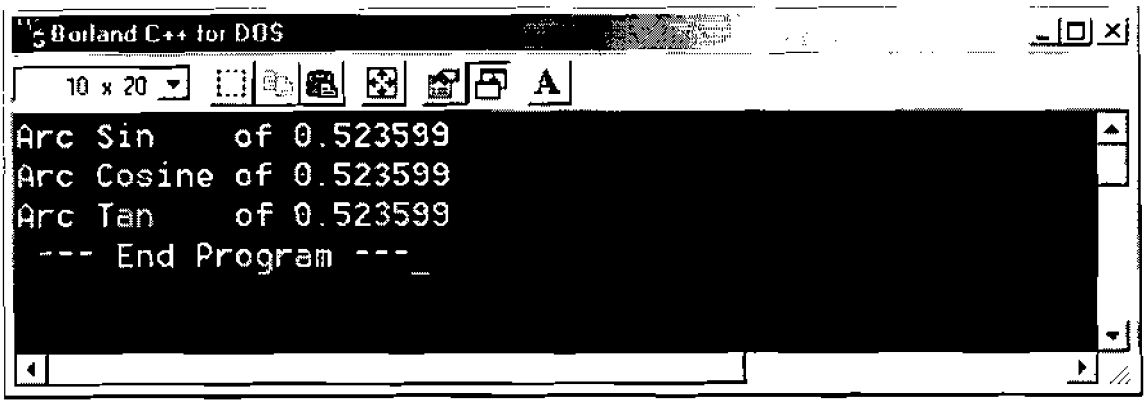
ฟังก์ชันนี้จะให้ค่ามุมในหน่วยเรเดียนของค่า arc sine ของ x

atan(x)

ฟังก์ชันนี้จะให้ค่ามุมในหน่วยเรเดียนของค่า arc tangent ของ x

โปรแกรมที่ 3.8 แสดงตัวอย่างการใช้ฟังก์ชัน เพื่อหาค่า arc sin, arc cosine และ arc tangent ด้วยฟังก์ชัน asin(x) acos(x) และ atan(x) และกำหนดค่าคงที่ให้กับ ฟังก์ชัน ให้สังเกตผลลัพธ์ของแต่ละฟังก์ชัน


```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <math.h>
4   main()
5   { clrscr();
6     cout<<"Arc Sin    of "<<asin(0.5)<<"\n";
7     cout<<"Arc Cosine of "<<acos(0.8660254)<<"\n";
8     cout<<"Arc Tan    of "<<atan(0.57735)<<"\n";
9     cout<<" --- End Program ---":getch();
10    return 0;
11  }
```



ภาพที่ 3.10 ผลลัพธ์ของใช้ฟังก์ชัน asin acos และ atan

จากโปรแกรมตัวอย่าง จะได้ผลลัพธ์จาก ฟังก์ชัน asin(0.5) มีค่าเท่ากับ 0.523599 ฟังก์ชัน acos(0.8660254) มีค่าเท่ากับ 0.523599 และ atan(0.57735) มีค่าเท่ากับ 0.523599

sin(x)

ฟังก์ชันนี้จะให้ค่า sine ของ x โดย x จะต้องเป็นมุมในหน่วยเรเดียน

cos(x)

ฟังก์ชันนี้จะให้ค่า cosine ของ x โดย x จะต้องเป็นมุมในหน่วยเรเดียน

tan(x)

ฟังก์ชันนี้จะให้ค่า tangent ของ x โดย x จะต้องเป็นมุมในหน่วยเรเดียน

sinh(x)

ฟังก์ชันนี้จะให้ค่า hyperbolic sine ของ x โดย x จะต้องเป็นมุมในหน่วยเรเดียน

cosh(x)

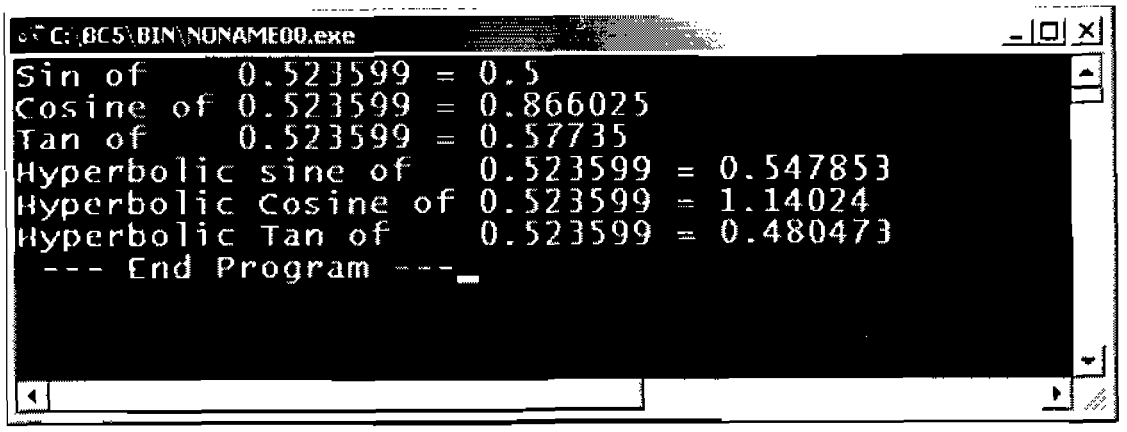
ฟังก์ชันนี้จะให้ค่า hyperbolic cosine ของ x โดย x จะต้องเป็นมุมในหน่วยเราเดียน

tanh(x)

ฟังก์ชันนี้จะให้ค่า hyperbolic tangent ของ x โดย x จะต้องเป็นมุมในหน่วยเรเดียน

โปรแกรมที่ 3.9 แสดงตัวอย่างการใช้ฟังก์ชัน เพื่อหาค่า arc sin, arc cosine และ arc tangent ด้วยฟังก์ชัน asin(x) acos(x) และ atan(x) และกำหนดค่าคงที่ให้กับ ฟังก์ชัน ให้สังเกตผลลัพธ์ของแต่ละฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <math.h>
4   main()
5   {  float v1=3.141592654/180.00;
6     clrscr();
7     cout<<"Sin of    "<<v1*30<<" = "<<sin(v1*30)<<"\n";
8     cout<<"Cosine of "<<v1*30<<" = "<<cos(v1*30)<<"\n";
9     cout<<"Tan of    "<<v1*30<<" = "<<tan(v1*30)<<"\n";
10    cout<<"Hyperbolic sine of  "<<v1*30<<" = "<<sinh(v1*30)<<"\n";
11    cout<<"Hyperbolic Cosine of "<<v1*30<<" = "<<cosh(v1*30)<<"\n";
12    cout<<"Hyperbolic Tan of    "<<v1*30<<" = "<<tanh(v1*30)<<"\n";
13    cout<<" --- End Program ---";getch();
14    return 0;
15  }
```



ภาพที่ 3.11 ผลลัพธ์ของการใช้ฟังก์ชัน sin cos tan sinh cosh และ tanh

จากโปรแกรมตัวอย่าง จะได้ผลลัพธ์จาก ฟังก์ชันต่าง ๆ ดังนี้

ฟังก์ชัน sin(0.523599) มีค่าเท่ากับ 0.5

ฟังก์ชัน cos(0.523599) มีค่าเท่ากับ 0.866025

ฟังก์ชัน tan(0.523599) มีค่าเท่ากับ 0.57735

ฟังก์ชัน sinh(0.523599) มีค่าเท่ากับ 0.547853

ฟังก์ชัน cosh(0.523599) มีค่าเท่ากับ 1.14024

ฟังก์ชัน tanh(0.523599) มีค่าเท่ากับ 0.480473

exp(x)

ฟังก์ชันนี้จะให้ค่าของ e ยกกำลัง x(e^x) โดย e มีค่า 2.718281828

log(x)

ฟังก์ชันนี้จะให้ค่า natural logaritlun ของ x x จะต้องมีค่ามากกว่าศูนย์เสมอ

log10(x)

ฟังก์ชันนี้จะให้ค่า log ฐาน 10 ของ x x จะต้องมีค่ามากกว่าศูนย์เสมอ

pow(x,a)

ฟังก์ชันนี้จะให้ค่าของ x ยกกำลัง a (x^a) x และ a จะต้องมีค่ามากกว่าศูนย์ ถ้า x มีค่าเป็นลบและ a ไม่ใช่เลขจำนวนเต็มจะเกิดข้อผิดพลาดขึ้น

sqrt(x)

ฟังก์ชันนี้จะให้ค่ารากที่สองของ x

ceil(x)

ฟังก์ชันนี้จะให้ค่าเลขจำนวนเต็มของ x ถ้า x มีค่าเป็นเทศนิยม และจะเท่ากับ x ถ้า x มีค่าเป็นเลขจำนวนเต็ม

floor(x)

ฟังก์ชันนี้จะให้ค่าเลขจำนวนเต็มของ x ซึ่งมีค่าน้อยกว่า x ถ้า x มีค่าเป็นทศนิยม และจะเท่ากับ x ถ้า x มีค่าเป็นจำนวนเต็ม

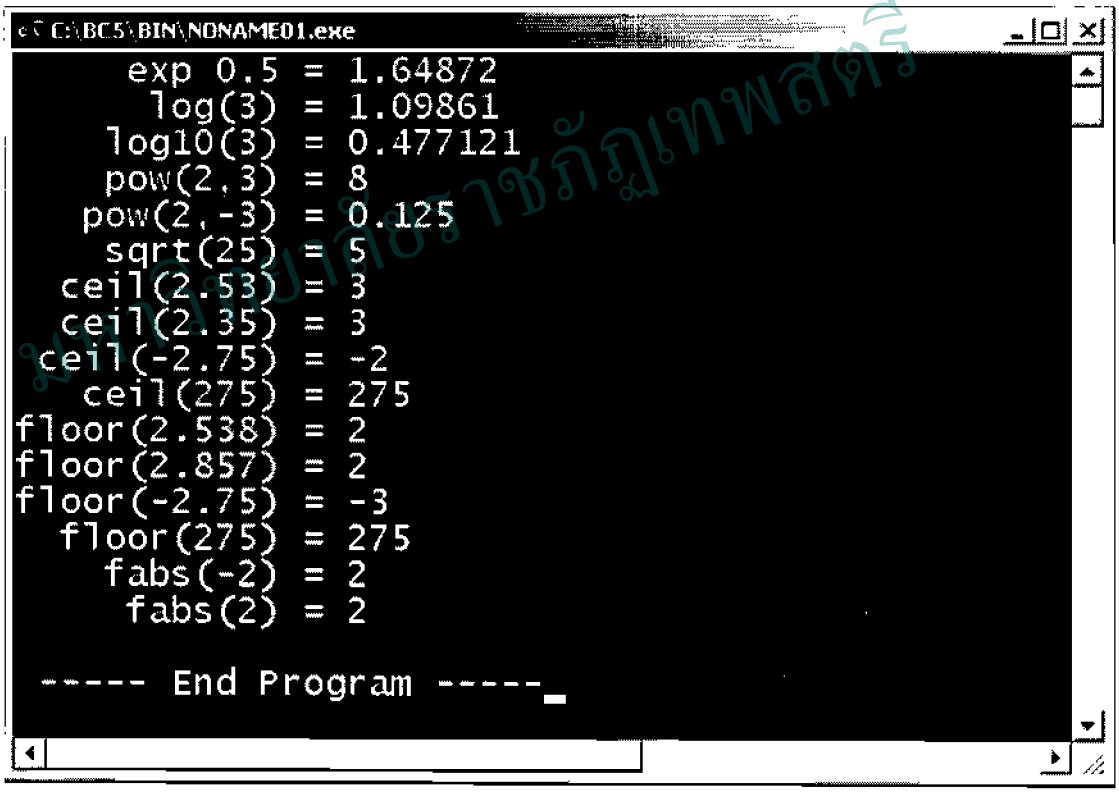
fabs(x)

ฟังก์ชันนี้จะให้ค่าสัมบูรณ์ของ x (absolute valve)

โปรแกรมที่ 3.10 แสดงตัวอย่างการใช้ฟังก์ชัน เพื่อหาค่า exp, log, log10, pow, sqrt, ceil, floor และ fabs โดยกำหนดค่าคงที่ให้กับ ฟังก์ชัน ให้สังเกตผลลัพธ์ของแต่ละฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <math.h>
4   main()
5   {
6     clrscr();
7       cout <<"    exp 0.5 = " <<exp(0.5)    <<"\n";
8       cout <<"     log(3) = " <<log(3)      <<"\n";
9       cout <<"   log10(3) = " <<log10(3)    <<"\n";
10      cout <<"   pow(2,3) = " <<pow(2,3)    <<"\n";
11      cout <<"  pow(2,-3) = " <<pow(2,-3)   <<"\n";
12      cout <<"   sqrt(25) = " <<sqrt(25)    <<"\n";
13      cout <<" ceil(2.53) = " <<ceil(2.53)  <<"\n";
14      cout <<" ceil(2.35) = " <<ceil(2.35)  <<"\n";
15      cout <<" ceil(-2.75) = " <<ceil(-2.75) <<"\n";
16      cout <<"   ceil(275) = " <<ceil(275)   <<"\n";
```

```
17    cout <<"floor(2.538) = " <<floor(2.538)<<"\n";
18    cout <<"floor(2.857) = " <<floor(2.857)<<"\n";
19    cout <<"floor(-2.75) = " <<floor(-2.75)<<"\n";
20    cout <<"  floor(275) = " <<floor(275)  <<"\n";
21    cout <<"    fabs(-2) = " <<fabs(-2)   <<"\n";
22    cout <<"     fabs(2) = " <<fabs(2)    <<"\n";
23    cout <<"\n ----- End Program -----";getch();
24    return 0;
25  }
```



ภาพที่ 3.12 ผลลัพธ์ของการใช้ฟังก์ชัน exp log pow sqrt ceil floor และ fabs

จากโปรแกรมตัวอย่าง จะได้ผลลัพธ์จาก ฟังก์ชันต่าง ๆ ดังนี้

ฟังก์ชัน exp(0.5) มีค่าเท่ากับ 1.64872

ฟังก์ชัน log(3) มีค่าเท่ากับ 1.09861

ฟังก์ชัน log10(3) มีค่าเท่ากับ 0.477121
ฟังก์ชัน pow(2,3) มีค่าเท่ากับ 8
ฟังก์ชัน pow(2,-3) มีค่าเท่ากับ 0.125
ฟังก์ชัน sqt(25) มีค่าเท่ากับ 5
ฟังก์ชัน ceil(2.53) มีค่าเท่ากับ 3
ฟังก์ชัน ceil(2.35) มีค่าเท่ากับ 3
ฟังก์ชัน ceil(-2.75) มีค่าเท่ากับ 2
ฟังก์ชัน ceil(2.53) มีค่าเท่ากับ 3
ฟังก์ชัน ceil(275) มีค่าเท่ากับ 275
ฟังก์ชัน floor(2.538) มีค่าเท่ากับ 2
ฟังก์ชัน floor(2.857) มีค่าเท่ากับ 2
ฟังก์ชัน floor(-2.75) มีค่าเท่ากับ -3
ฟังก์ชัน floor(275) มีค่าเท่ากับ 275
ฟังก์ชัน fabs(2) มีค่าเท่ากับ 2
ฟังก์ชัน fabs(-2) มีค่าเท่ากับ 2

ฟังก์ชันเกี่ยวกับตัวอักษร

การเรียกใช้ฟังก์ชันเกี่ยวกับตัวอักษรจะต้องระบุไฟล์ ctype.h ในโปรแกรมเสมอ ฟังก์ชันเกี่ยวกับตัวอักษรที่ควรทราบมีดังนี้

isalnum(ch)

ฟังก์ชันนี้จะให้ค่าตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ ถ้า ch มีค่าเป็นตัวอักษรหรือตัวเลข และจะให้ค่าเป็นศูนย์ถ้า ch ไม่ได้มีค่าเป็นตัวอักษรหรือตัวเลข

isalpha(ch)

ฟังก์ชันนี้จะให้ค่าตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ถ้า ch มีค่าเป็นตัวอักษร และจะให้ค่าเป็นศูนย์ถ้า ch ไม่ได้มีค่าเป็นตัวอักษรรวมถึงเครื่องหมายต่าง ๆ

isdigit(ch)

ฟังก์ชันนี้จะให้ค่าตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ถ้า ch มีค่าเป็นตัวเลข 0-9 และจะให้ค่าเป็นศูนย์ถ้า ch ไม่มีค่าเป็นตัวเลข

isgraph(ch)

ฟังก์ชันนี้จะให้ค่าตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ ถ้า ch มีค่าเป็นตัวอักษรที่สามารถพิมพ์ได้ (printable character) ยกเว้นช่องว่าง (space) ตัวอักษรที่สามารถพิมพ์ได้นี้จะมีค่ารหัส แอสกี (เลขฐานสิบหก) ตั้งแต่ 21-7E หรือตั้งแต่ 33-126 (เลขฐานสิบ) ในกรณีที่ ch ไม่ได้มีค่าของรหัสแอสกีดังกล่าว ฟังก์ชัน isgraph (ch) จะให้ค่าเป็นศูนย์

islower(ch)

ฟังก์ชันนี้จะให้ค่าตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ ถ้า ch มีค่าเป็นตัวอักษรซึ่งเป็นอักษรตัวเล็ก ในกรณีที่ ch มีค่าเป็นตัวอักษรตัวใหญ่ เครื่องหมายต่าง ๆ และตัวเลขฟังก์ชัน islower (ch) จะให้ค่าเป็นศูนย์

isupper(ch)

ฟังก์ชันนี้จะให้ค่าตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ถ้า ch มีค่าเป็นตัวอักษรตัวเล็ก ถ้า ch มีค่าเป็นตัวอักษรตัวใหญ่ เครื่องหมายต่าง ๆ และตัวเลขฟังก์ชัน isupper (ch) จะมีค่าเป็นศูนย์

โปรแกรมที่ 3.11 แสดงตัวอย่างการใช้ฟังก์ชัน เพื่อหาค่า isalnum, isalpha, isdigit, isgraph, islower และ isupper โดยกำหนดค่าคงที่ให้กับ ฟังก์ชัน ให้สังเกตผลลัพธ์ของแต่ละฟังก์ชัน


```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <ctype.h>
4   main()
5   {  clrscr();
6      cout <<"isalnum('a') = " <<isalnum('a')   <<"\n";
7      cout <<"isalnum('*') = " <<isalnum('*')   <<"\n";
8      cout <<"isalnum(';') = " <<isalnum(';')   <<"\n";
9      cout <<"isalnum('0') = " <<isalnum('0')   <<"\n";
10     cout <<"isalnum(';') = " <<isalnum(';')   <<"\n";
11     cout <<"------------------------- \n";
12     cout <<"isalpha('a') = " <<isalpha('a')   <<"\n";
```

```
13      cout <<"isalpha('*') = " <<isalpha('*')    <<"\n";
14      cout <<"isalpha(';') = " <<isalpha(';')    <<"\n";
15      cout <<"isalpha('2') = " <<isalpha('2')    <<"\n";
16      cout <<"---------------------------- \n";
17      cout <<"  isdigit(2) = " <<isdigit(2)     <<"\n";
18      cout <<"isdigit('2') = " <<isdigit('2')    <<"\n";
19      cout <<"---------------------------- \n";
20      cout <<"  isgraph(2) = " <<isgraph(2)     <<"\n";
21      cout <<"isgraph('2') = " <<isgraph('2')    <<"\n";
22      cout <<"---------------------------- \n";
23      cout <<"islower('a') = " <<islower('a')    <<"\n";
24      cout <<"islower('A') = " <<islower('A')    <<"\n";
25      cout <<"---------------------------- \n";
26      cout <<"isupper('a') = " <<isupper('a')    <<"\n";
27      cout <<"isupper('A') = " <<isupper('A')    <<"\n";
28      cout <<"\n ----- End Program -----";getch();
29      return 0;
30  }
```

ภาพที่ 3.13 ผลลัพธ์ของการใช้ฟังก์ชัน isalnum isalpha isdigit isgraph islower และ isupper

จากโปรแกรมตัวอย่าง จะได้ผลลัพธ์จาก ฟังก์ชันต่าง ๆ ดังนี้

ฟังก์ชัน isalnum('a') มีค่าเท่ากับ 8

ฟังก์ชัน isalnum('*') มีค่าเท่ากับ 0

ฟังก์ชัน isalnum(';') มีค่าเท่ากับ 0

ฟังก์ชัน isalnum('0') มีค่าเท่ากับ 2

ฟังก์ชัน isalpha('a') มีค่าเท่ากับ 8

ฟังก์ชัน isalpha('*') มีค่าเท่ากับ 0

ฟังก์ชัน isalpha(';') มีค่าเท่ากับ 0

ฟังก์ชัน isalpha('0') มีค่าเท่ากับ 0

ฟังก์ชัน isdigit(2) มีค่าเท่ากับ 0

ฟังก์ชัน isdigit('2') มีค่าเท่ากับ 2

ฟังก์ชัน isgraph(2) มีค่าเท่ากับ 0

ฟังก์ชัน isgraph('2') มีค่าเท่ากับ 18

ฟังก์ชัน islower('a') มีค่าเท่ากับ 8

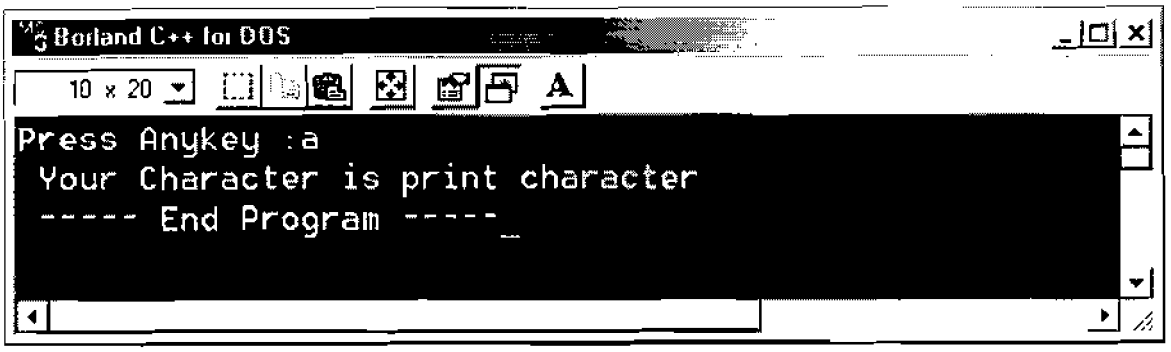
ฟังก์ชัน islower('A') มีค่าเท่ากับ 0
ฟังก์ชัน isupper('a') มีค่าเท่ากับ 0
ฟังก์ชัน isupper('A') มีค่าเท่ากับ 4

isprint(ch)

ฟังก์ชันนี้จะให้ค่าตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ ถ้า ch มีค่าเป็นตัวอักษรที่สามารถพิมพ์ได้รวมทั้งช่องว่าง ตัวอักษรที่สามารถพิมพ์ได้จะมีค่ารหัสแอสกี (เลขฐานสิบหก) ตั้งแต่ 20-7E ถ้า ch ไม่ได้มีค่ารหัสแอสกีดังกล่าว ฟังก์ชัน Isprint(ch) จะให้ค่าเป็นศูนย์

โปรแกรมที่ 3.12 แสดงตัวอย่างการใช้ฟังก์ชัน isprint โดยให้ผู้ใช้ป้อนข้อมูลลงไป โดยให้ป้อนทั้ง ข้อมูลที่สามารถพิมพ์ได้ คือ อักษรและ เครื่องหมาย ทั่ว ๆ ไป และ อักษรที่ไม่สามารถพิมพ์ได้ ให้สังเกตผลลัพธ์ของฟังก์ชัน


```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <ctype.h>
4   main()
5   { char ch;
6     clrscr();
7     cout << "Press Anykey :";ch = getche();
8     if (isprint(ch))
9       cout<<"\n Your Character is print character";
10    else
11      cout<<"\n Your Character is Not print character";
12    cout <<"\n ----- End Program -----";getch();
13    return 0;
14  }
```



ภาพที่ 3.14 ผลลัพธ์ของการใช้ฟังก์ชัน isprint

จากตัวอย่าง การใช้งาน ฟังก์ชัน isprint โปรแกรมนี้จะรอรับข้อมูลจากผู้ใช้โปรแกรม เมื่อมีการ ใส่ข้อมูลเข้าไป ฟังก์ชั่น isprint จะให้คำตอบออกมาเป็น 0 กรณีที่ ผู้ใช้ป้อนข้อมูล ที่ไม่ใช่ อักษรที่พิมพ์ได้ เช่น แป้น ctrl shift และ อื่น ๆ อีกคำตอบหนึ่งของ ฟังก์ชันนี้ คือ คำตอบเป็น 1 ในกรณีที่ ผู้ใช้ป้อนอักษรที่สามารถพิมพ์ได้เข้าไป เช่น a b c และ อักษร อื่น ๆ

ispunct(ch)

ฟังก์ชันนี้จะให้ค่าตัวเลขจำนวนซึ่งไม่เท่ากับศูนย์ ถ้า ch มีค่าเป็นเครื่องหมาย (punctuation character) ซึ่งในที่นี้ได้แก่เครื่องหมายต่าง ๆ ยกเว้นตัวเลข ตัวอักษร (A-Z) และ ช่องว่าง ถ้า ch ไม่มีค่าเป็นเครื่องหมายต่าง ๆ ฟังก์ชัน ispunct (ch) จะให้ค่าเป็นศูนย์

โปรแกรมที่ 3.13 แสดงตัวอย่างการใช้ฟังก์ชัน ispunct โดยให้ผู้ใช้ป้อนข้อมูลลงไป โดยให้ป้อนทั้ง ข้อมูลที่เป็น ตัวอักษร และ เครื่องหมาย ทั่ว ๆ ไป ให้สังเกตผลลัพธ์ของฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <ctype.h>
4   main()
5   { char ch;
6     clrscr();
7     cout << "Press Anykey :";ch = getche();
```

```
8       if (ispunct(ch))
9         cout<<"\n Your Character is punctation character";
10      else
11        cout<<"\n Your Character is Not punctation character";
12      cout <<"\n ----- End Program -----";getch();
13      return 0;
14    }
```

ภาพที่ 3.15 ผลลัพธ์ของการใช้ฟังก์ชัน ispunct

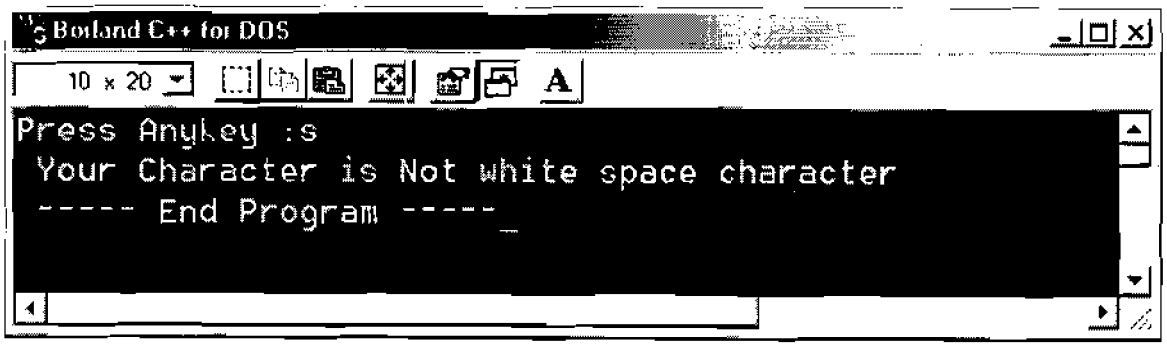
จากโปรแกรม เมื่อใช้ ฟังก์ชัน ispunct ซึ่งฟังก์ชันนี้จะให้ค่าเป็น จริง หรือ จำนวนที่ไม่เท่ากับ ศูนย์ เมื่อผู้ใช้ ป้อน เครื่องหมายลงไป และ ให้ค่าเป็น เท็จ หรือ ค่าศูนย์ เมื่อ ผู้ใช้ป้อน ตัวอักษรที่ไม่ใช่เครื่องหมาย ดังโปรแกรมตัวอย่าง ผู้ใช้ ป้อน ตัวอักษร s จะทำให้ฟังก์ชันมีค่าเป็น เท็จ และแสดงข้อความออกมาดังภาพที่ 3.15

isspace(ch)

ฟังก์ชันนี้ให้ค่าตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ หรือ ค่าเป็นจริง ถ้า ch มีค่าเป็น space, tab, vertical tab, form feed, carriage return หรือ newline character ค่าต่าง ๆ เหล่านี้เรียกว่า whitespace ถ้า ch ไม่มีค่าดังกล่าว ฟังก์ชัน isspace(ch) จะให้ค่าเป็นศูนย์ หรือ เท็จ

โปรแกรมที่ 3.14 แสดงตัวอย่างการใช้ฟังก์ชัน isspace โดยให้ผู้ใช้ป้อนข้อมูลลงไป โดยให้ป้อนทั้ง ข้อมูลที่เป็น ตัวอักษร และ เครื่องหมาย ทั่ว ๆ ไป ให้สังเกตผลลัพธ์ของฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <ctype.h>
4   main()
5   {  char ch;
6     clrscr();
7     cout << "Press Anykey :";ch = getche();
8     if (isspace(ch))
9       cout<<"\n Your Character is white space character";
10    else
11      cout<<"\n Your Character is Not white space character";
12    cout <<"\n ----- End Program -----";getch();
13    return 0;
14  }
```



ภาพที่ 3.16 ผลลัพธ์ของการใช้ฟังก์ชัน isspace

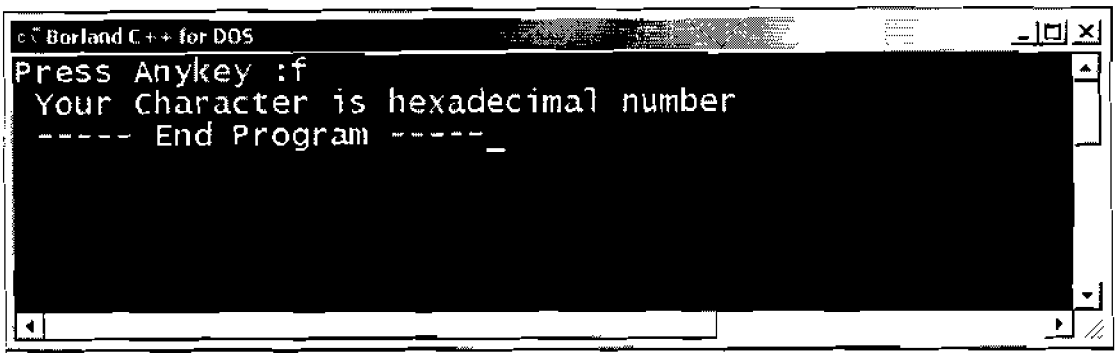
จากโปรแกรมตัวอย่าง เมื่อทำการป้อนตัวอักษร s ลงไป ฟังก์ชัน space จะให้ค่าเป็นเท็จ และพิมพ์ข้อความว่า Your Character is Not white space character เพราะข้อมูลที่ป้อนลงไปไม่ใช่ เครื่องหมาย ที่เรียกว่า white space ดังกล่าวแล้วในตอนต้น

isxdigit(ch)

ฟังก์ชันนี้จะให้ค่าเป็นตัวเลขจำนวนเต็มซึ่งไม่เท่ากับศูนย์ หรือให้ค่าเป็นจริง ถ้า ch มีค่าเป็นตัวเลขในระบบเลขฐานสิบหก ซึ่งได้แก่เลข 0-9 , A-F และ a-f ถ้า ch ไม่ได้มีค่าดังกล่าว ฟังก์ชันจะให้ค่าเป็นศูนย์ หรือเป็นเท็จ

โปรแกรมที่ 3.15 แสดงตัวอย่างการใช้ฟังก์ชัน isxdigit โดยให้ผู้ใช้ป้อนข้อมูลลงไป โดยให้ป้อนทั้ง ข้อมูลที่เป็น ตัวอักษร ตัวเลข และ เครื่องหมาย ทั่ว ๆ ไป ให้สังเกตผลลัพธ์ของฟังก์ชันเมื่อป้อนข้อมูลที่เป็น ตัวเลข ฐานสิบหก


```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <ctype.h>
4   main()
5   { char ch;
6     clrscr();
7     cout << "Press Anykey :";ch = getche();
8     if (isxdigit(ch))
9       cout<<"\n Your Character is hexadecimal number";
10    else
11      cout<<"\n Your Character is Not hexadecimal number";
12    cout <<"\n ----- End Program -----";getch();
13    return 0;
14  }
```



ภาพที่ 3.17 ผลลัพธ์ของการใช้ฟังก์ชัน isxdigit

จากโปรแกรม ตัวอย่าง เมื่อป้อนข้อมูลที่เป็นตัวเลขในเลขฐาน สิบหก ได้แก่ f ลงไป ฟังก์ชัน isxdigit จะให้ค่าเป็นจริง และพิมพ์ข้อความ Your Character is Hexadecimal Number

tolower(ch)

ฟังก์ชันจะเปลี่ยนตัวอักษรซึ่งเก็บอยู่ในตัวแปร ch เป็นอักษรตัวเล็ก

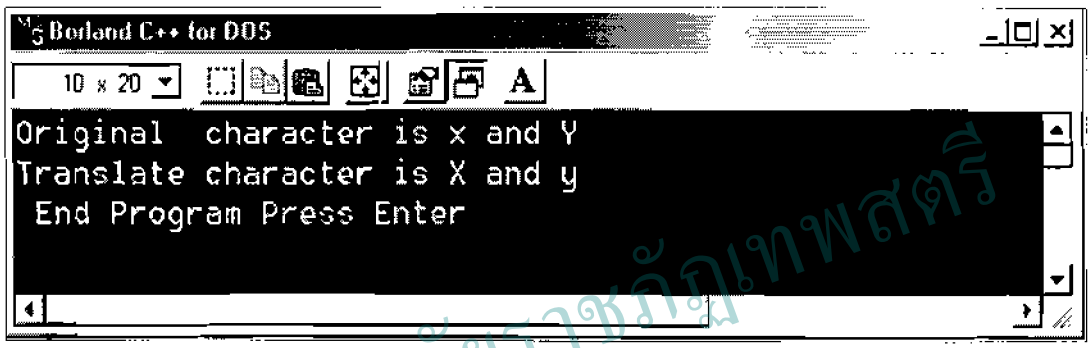
toupper(ch)

ฟังก์ชันนี้จะเปลี่ยนตัวอักษรซึ่งเก็บอยู่ในตัวแปร ch เป็นตัวอักษรตัวใหญ่

โปรแกรมที่ 3.16 แสดงตัวอย่างการใช้ฟังก์ชัน tolower และ toupper เมื่อกำหนดค่าคงที่ให้กับตัวแปร ให้เป็นตัวอักษรพิมพ์เล็ก และ ตัวอักษรพิมพ์ใหญ่ ให้สังเกตค่าที่ได้จากฟังก์ชัน

```
1   #include <iostream.h>
2   #include <stdio.h>
3   #include <conio.h>
4   #include <ctype.h>
5   main()
6   { char a,b;
7     a = 'x';
8     b = 'Y';
```

```
9     clrscr();
10    cout<<"Original  character is "<<a<<" and "<<b<<"\n";;
11    printf("Translate character is %c",toupper(a));
12    printf(" and %c",tolower(b));
13    cout<<"\n End Program Press Enter";getch();
14    return 0;
15  }
```



ภาพที่ 3.18 ผลลัพธ์ของการใช้ฟังก์ชัน tolower และ toupper

ฟังก์ชันเกี่ยวกับสตริง

การเรียกใช้ฟังก์ชันเกี่ยวกับสตริง ซึ่งหมายถึงฟังก์ชันประเภทตัวอักษร จะต้องระบุไฟล์ string.h ในโปรแกรมเสมอ ฟังก์ชันเกี่ยวกับสตริงที่ควรทราบมีดังนี้

strcmp(str1,str2)

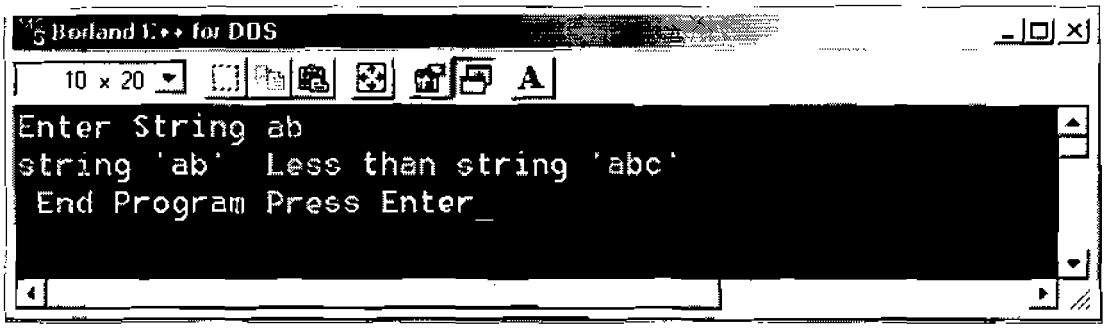
ฟังก์ชันนี้จะเปรียบเทียบอาร์เรย์สตริงก์ str1 และ str2 โดยใช้ค่ารหัสแอสกีเป็นหลัก ดังได้กล่าวมาแล้วในบทที่ 9 ผลลัพธ์ของฟังก์ชันจะได้ตัวเลขน้อยกว่าศูนย์ เท่ากับศูนย์ หรือมากกว่าศูนย์ ซึ่งมีความหมายดังนี้

น้อยกว่าศูนย์	str1	มีอันดับน้อยกว่า	str2
เท่ากับศูนย์	str1	มีอันดับเท่ากับ	str2
มากกว่าศูนย์	str1	มีอันดับมากกว่า	str2

Str1 และ str2 อาจจะเป็นสตริงซึ่งอยู่ในเครื่องหมาย “ ”

โปรแกรมที่ 3.17 แสดงตัวอย่างการใช้ฟังก์ชัน strcmp เพื่อทำการเปรียบเทียบ ข้อความที่ 1 กับข้อความที่ 2 โดยกำหนดให้ข้อความที่ 1 มาจากผู้ใช้ป้อนข้อมูลลงไป และเพื่อนำมาเปรียบเทียบข้อความที่ 2 ซึ่งเป็นค่าคงที่ ที่มีค่าเป็น abc เมื่อป้อนข้อมูลลงไป ให้สังเกตค่าที่ได้จากการเปรียบเทียบ จากฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <string.h>
4   main()
5   { char a[10],b[10]="abc";
6     clrscr();
7     cout<<"Enter String ";cin>>a;
8     if (strcmp(a,b) < 0)
9       cout<<"string '"<<a<<"'  Less than string '"<<b<<"'";
10    else if (strcmp(a,b) > 0)
11    cout<<"string '"<<a<<"'  Greate than string '"<<b<<"'";
12  else
13    cout<<"string '"<<a<<"'  Equal string '"<<b<<"'";
14    cout<<"\n End Program Press Enter";getch();
15    return 0;
16  }
```



ภาพที่ 3.19 ผลลัพธ์ของการใช้ฟังก์ชัน strcmp

จากตัวอย่าง เมื่อป้อนข้อมูล เป็นตัวอักษร ab ให้กับตัวแปร a แล้วนำไปเปรียบเทียบกับข้อความ abc ซึ่งเก็บในตัวแปร b ผลการเปรียบเทียบ สตริง a น้อยกว่า สตริง b จึงได้ผลลัพธ์ดังตัวอย่าง

strcpy(str1,str2)

ฟังก์ชันนี้จะนำค่าของอาร์เรย์สตริงก์ str2 ไปเก็บไว้ในอาร์เรย์สตริง str1 str2 อาจจะเป็นค่าคงที่สตริงซึ่งอยู่ในเครื่องหมาย " "

strlen(str)

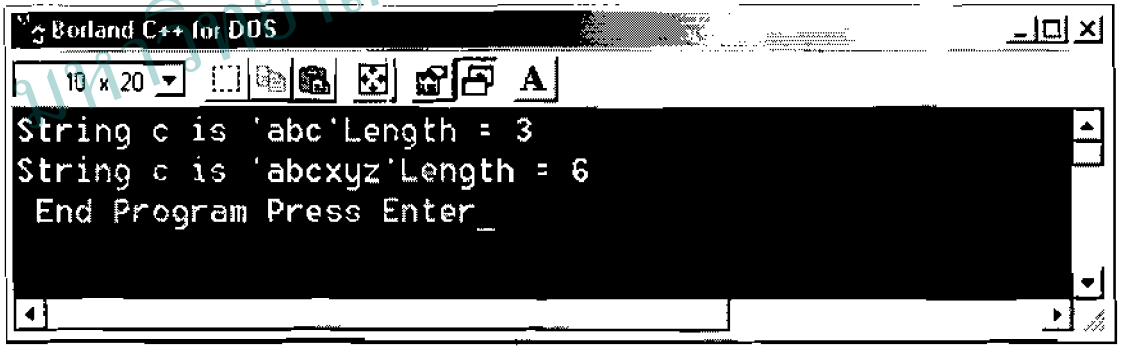
ฟังก์ชันนี้จะให้ตัวเลขจำนวนเต็มแสดงความยาวของสตริงซึ่งเก็บอยู่ในอาร์เรย์สตริง str str อาจจะเป็นค่าคงที่สตริงซึ่งอยู่ในเครื่องหมาย

strcat(str1, str2)

ฟังก์ชันนี้จะเชื่อมอาร์เรย์สตริงก์ str1 และ str2 ผลลัพธ์จะถูกเก็บในอาร์เรย์สตริงก์ str1 ทั้งนี้อาร์เรย์ str1 จะต้องมีขนาดใหญ่พอที่จะเก็บผลรวมของอาร์เรย์สตริงก์ str1 และ str2 str2 อาจจะเป็นค่าคงที่สตริงซึ่งอยู่ในเครื่องหมาย " "

โปรแกรมที่ 3.18 แสดงตัวอย่างการใช้ฟังก์ชัน strcpy strlen และ strcat เพื่อจัดการกับข้อมูลประเภทข้อความ สังเกตค่าที่ได้จากการจากฟังก์ชัน ซึ่งได้แก่ การคัดลอกข้อความ การแสดงความยาวของข้อความ และ การเชื่อมข้อความเข้าด้วยกัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <string.h>
4   main()
5   { char a[10]="abc",b[10]="xyz",c[10];
6     strcpy(c,a);
7     cout<<"String c is '"<<c<<"'"<<"Length = "<<strlen(c)<<"\n";
8     strcat(c,b);
9     cout<<"String c is '"<<c<<"'"<<"Length = "<<strlen(c);
10    cout<<"\n End Program Press Enter";getch();
11    return 0;
12  }
```



ภาพที่ 3.20 ผลลัพธ์ของการใช้ฟังก์ชัน strcpy strlen และ strcat

จากโปรแกรมตัวอย่าง กำหนดให้ a มีค่าเท่ากับ abc และ b มีค่าเท่ากับ xyz เมื่อใช้ฟังก์ชัน strcpy(c,a) หมายถึง การคัดลอกข้อความจาก สตริง a ไปเก็บที่ตัวแปร c เมื่อสั่งให้แสดงผล c จึงมีค่าเท่ากับ abc และใช้ฟังก์ชัน strlen ตรวจสอบความยาว มีค่าเท่ากับ 3 ต่อจากนั้น ใช้ฟังก์ชัน strcat เชื่อมต่อ ข้อความจาก c และ b เข้าด้วยกัน เมื่อแสดงออกมาจึงได้ผลลัพธ์ เป็น abcxyz มีความยาวเท่ากับ 6 โดยใช้ ฟังก์ชัน strlen ตรวจสอบ

ฟังก์ชันทั่ว ๆ ไป

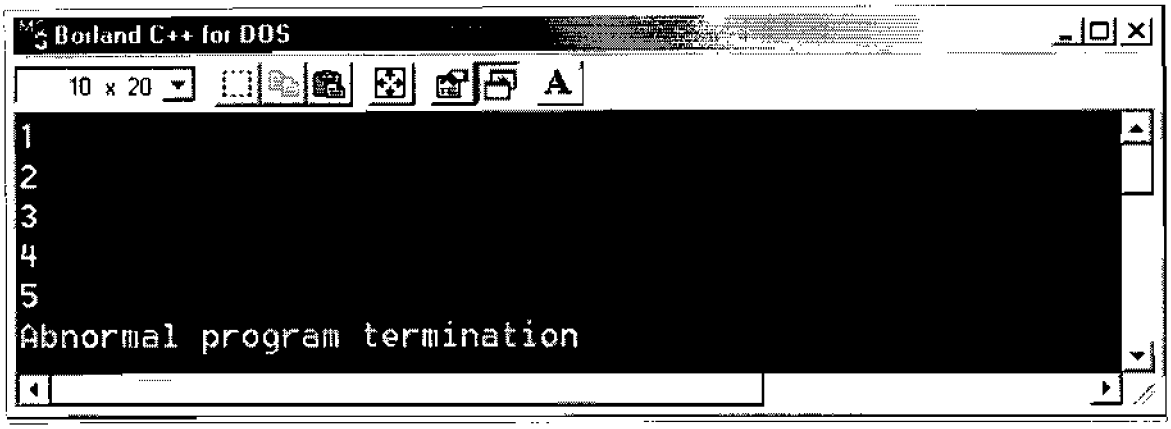
ในหัวข้อนี้จะกล่าวถึงฟังก์ชันที่เก็บอยู่ในไฟล์ stdlib.h ฟังก์ชันที่ควรทราบมีดังนี้

abort()

ฟังก์ชันจะยกเลิกการทำงานของโปรแกรมขณะนั้น และจะมีข้อความ "Abnormal program termination"

โปรแกรมที่ 3.19 แสดงตัวอย่างการใช้ฟังก์ชัน abort() เพื่อยกเลิกการทำงาน เมื่อโปรแกรมดำเนินการตามวงรอบ ได้เกิน 5 รอบ สังเกตค่าที่ได้จากการจากฟังก์ชัน จะแสดงข้อความ Abnormal program terminate

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <stdlib.h>
4   main()
5   { int i;
6   clrscr();
7     for (i=1;i<=10;i++)
8     {  if (i > 5)
9        abort();
10       else
11       cout<<i<<"\n";
12    }
13    cout<<"\n End Program Press Enter";getch();
14    return 0;
15  }
```



ภาพที่ 3.21 ผลลัพธ์ของการใช้ฟังก์ชัน abort

จากโปรแกรมตัวอย่าง ใช้วงรอบการทำงานโดยคำสั่ง for ให้ทำงานทั้งหมด 10 รอบ ในแต่ละรอบ จะพิมพ์ค่าตัวเลขประจำรอบ ได้แก่ค่าของ i พอถึงรอบ ที่มากกว่า 5 หรือ ในที่นี้คือรอบที่ 6 เครื่องจะหยุดการทำงานด้วย ฟังก์ชัน abort()

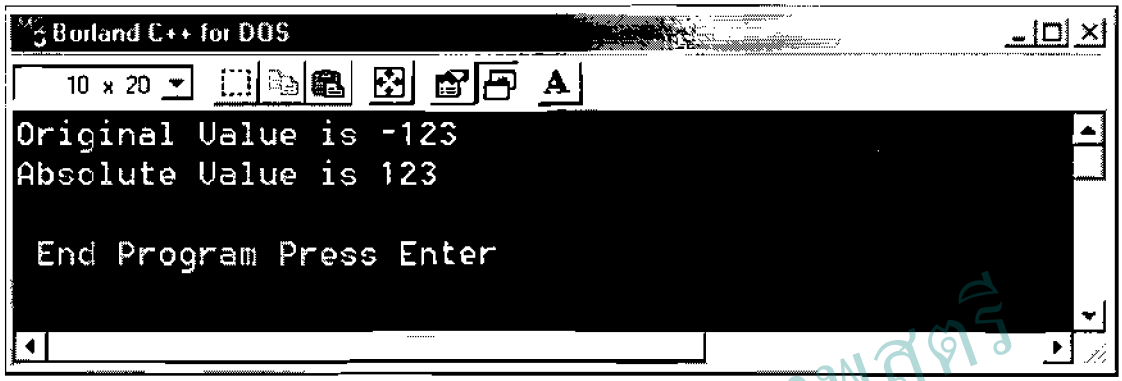
abs()

ฟังก์ชันนี้จะให้ค่า สัมบูรณ์ของ x ซึ่งเป็นชนิดเลขจำนวนเต็ม ไม่ว่า x จะเป็นค่าที่เป็นบวก หรือ ลบ ค่าสัมบูรณ์ ก็จะเป็นค่าที่เป็น บวกเสมอ

โปรแกรมที่ 3.20 แสดงตัวอย่างการใช้ฟังก์ชัน abs() เพื่อแสดงค่าสัมบูรณ์ของตัวเลข ซึ่งตัวเลขที่กำหนดไว้เป็นค่าลบ เมื่อใช้ฟังก์ชัน abs จะเปลี่ยนจากค่าที่ติดลบ มาเป็นค่าบวก สังเกตค่าที่ได้จากการจากฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <stdlib.h>
4   main()
5   { int i=-123;
6     clrscr();
7     cout<<"Original Value is "<<i<<"\n";
```

```
8       cout<<"Absolute Value is "<<abs(i)<<"\n";
9       cout<<"\n End Program Press Enter";getch();
10      return 0;
11  }
```



ภาพที่ 3.22 ผลลัพธ์ของการใช้ฟังก์ชัน abs

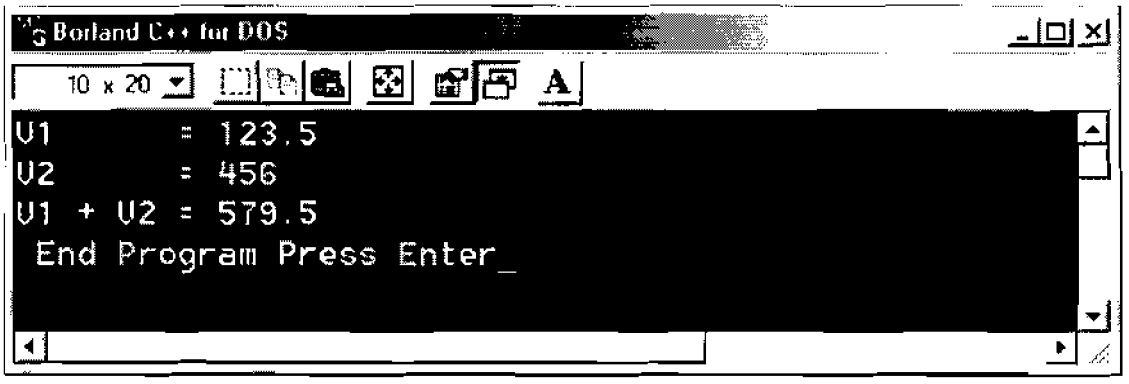
จากโปรแกรมตัวอย่าง กำหนดให้ตัวแปร i มีค่าเท่ากับ -123 เพื่อนำเข้าไปใน ฟังก์ชัน abs จะได้ค่าสัมบูรณ์ของตัวเลข ซึ่งมีค่าเท่ากับ 123

atof(str)

ฟังก์ชันนี้จะเปลี่ยนสตริงในอาร์เรย์สตริง str เป็นข้อมูลชนิดเลขจำนวนจริงละเอียดสองเท่าซึ่งสามารถนำไปคำนวณทางคณิตศาสตร์ได้ สตริงดังกล่าวนอกจากจะมีตัวเลขจำนวนเต็มหรือทศนิยม แล้วอาจจะตามด้วยตัวอักษร สัญลักษณ์ หรือเครื่องหมายต่าง ๆ ได้ ซึ่งส่วนนี้จะไม่มีผลต่อการเปลี่ยนชนิดข้อมูล

โปรแกรมที่ 3.21 แสดงตัวอย่างการใช้ฟังก์ชัน atof() เพื่อแปลงค่า สตริง ให้เป็นข้อมูลชนิดตัวเลขจำนวนจริง ถึงแม้ว่า ข้อความนั้น จะมีทั้งข้อมูลและตัวเลข ฟังก์ชันนี้ก็สามารถ แปลงค่าได้ โดยเลือกเฉพาะ ที่เป็นตัวเลข สังเกตผลลัพธ์ที่ได้จากการจากฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <stdlib.h>
4   main()
5   { float v1,v2,v3;
6     clrscr();
7     v1 = atof("123.5abc");
8     v2 = atof("456a4a6x");
9     v3 = v1 + v2;
10    cout<<"V1     = "<<v1<<"\n";
11    cout<<"V2     = "<<v2<<"\n";
12    cout<<"V1 + V2 = "<<v3;
13    cout<<"\n End Program Press Enter";getch();
14    return 0;
15  }
```

ภาพที่ 3.23 ผลลัพธ์ของการใช้ฟังก์ชัน atof

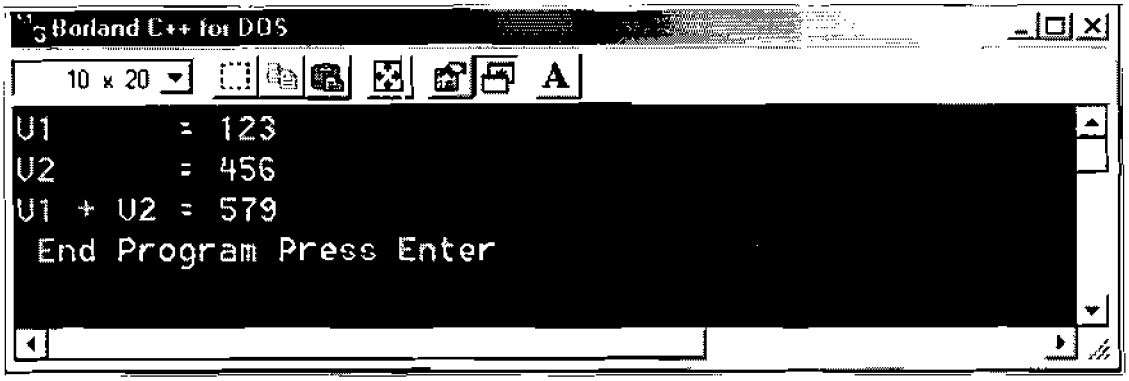
จากตัวอย่าง การแปลงค่าด้วย ฟังก์ชัน atof ซึ่งจะทำการแปลงค่าจาก ข้อความ เป็นตัวเลขจำนวนจริง จากข้อมูลตัวอย่าง 123.5abc จะถูกแปลงเป็น 123.5 และ 456a4a6x จะถูกแปลงเป็น 456 เมื่อนำมาคำนวณ โดยนำมารวมกัน จึงได้ผลลัพธ์ เป็น 579.5

atoi(str)

ฟังก์ชันนี้จะเปลี่ยนสตริงในอาร์เรย์สตริง str เป็นข้อมูลชนิดเลขจำนวนเต็มซึ่งสามารถนำไปคำนวณทางคณิตศาสตร์ได้ สตริงดังกล่าวนอกจากจะมีตัวเลขแล้วอาจจะตามด้วยอักษร สัญลักษณ์ หรือเครื่องหมายต่าง ๆ ซึ่งส่วนนี้จะมีผลในการเปลี่ยนชนิดข้อมูล

โปรแกรมที่ 3.22 แสดงตัวอย่างการใช้ฟังก์ชัน atoi() เพื่อแปลงค่า สตริง ให้เป็นข้อมูลชนิดตัวเลขจำนวนเต็ม ไม่ว่าข้อความนั้น จะมีทั้งข้อมูลและตัวเลข ฟังก์ชันนี้ก็สามารถ แปลงค่าได้โดยเลือกเฉพาะ ที่เป็นตัวเลข สังเกตผลลัพธ์ที่ได้จากการจากฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <stdlib.h>
4   main()
5   { float v1,v2,v3;
6     clrscr();
7     v1 = atoi("123.5abc");
8     v2 = atoi("456a4a6x");
9     v3 = v1 + v2;
10    cout<<"V1     = "<<v1<<"\n";
11    cout<<"V2     = "<<v2<<"\n";
12    cout<<"V1 + V2 = "<<v3;
13    cout<<"\n End Program Press Enter";getch();
14    return 0;
15  }
```



ภาพที่ 3.24 ผลลัพธ์ของการใช้ฟังก์ชัน atoi

จากตัวอย่าง การแปลงค่าด้วย ฟังก์ชัน atoi ซึ่งจะทำการแปลงค่าจาก ข้อความ เป็นตัวเลขจำนวนเต็ม จากข้อมูลตัวอย่าง 123.5abc จะถูกแปลงเป็น 123 และ 456a4a6x จะถูกแปลงเป็น 456 เมื่อนำมาคำนวณ โดยนำมารวมกัน จึงได้ผลลัพธ์ เป็น 579

atol(str)

ฟังก์ชันนี้จะเปลี่ยนสตริงในอาร์เรย์สตริง str เป็นข้อมูลชนิดเลขจำนวนเต็มยาว ซึ่งโดยปกติ เลขจำนวนเต็ม จะมีขนาด เท่ากับ 2 ไบท์ แต่ถ้าเป็นจำนวนเต็มยาว จะเพิ่มขนาดเป็น 4 ไบท์ ซึ่งสามารถนำไปคำนวณทางคณิตศาสตร์ได้ สตริงดังกล่าวนอกจากจะมีตัวเลขแล้วอาจจะตามด้วยตัวอักษร สัญลักษณ์ หรือ เครื่องหมายต่าง ๆ ซึ่งส่วนนี้จะไม่มีผลต่อการเปลี่ยนชนิดข้อมูล

โปรแกรมที่ 3.23 แสดงตัวอย่างการใช้ฟังก์ชัน atol() เพื่อแปลงค่า สตริง ให้เป็นข้อมูลชนิดตัวเลขจำนวนเต็ม ไม่ว่าข้อความนั้น จะมีทั้งข้อมูลและตัวเลข ฟังก์ชันนี้ก็สามารถ แปลงค่าได้โดยเลือกเฉพาะ ที่เป็นตัวเลข สังเกตผลลัพธ์ที่ได้จากการจากฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <stdlib.h>
4   main()
5   { int v1,v2,v3;
6     clrscr();
```

```
7     v1 = atol("1230.5abc");
8     v2 = atol("4560a4a6x");
9     v3 = v1 + v2;
10    cout<<"V1     = "<<v1<<"\n";
11    cout<<"V2     = "<<v2<<"\n";
12    cout<<"V1 + V2    "<<v3;
13    cout<<"\n End Program Press Enter";getch();
14    return 0;
15    }
```

ภาพที่ 3.25 ผลลัพธ์ของการใช้ฟังก์ชัน atol

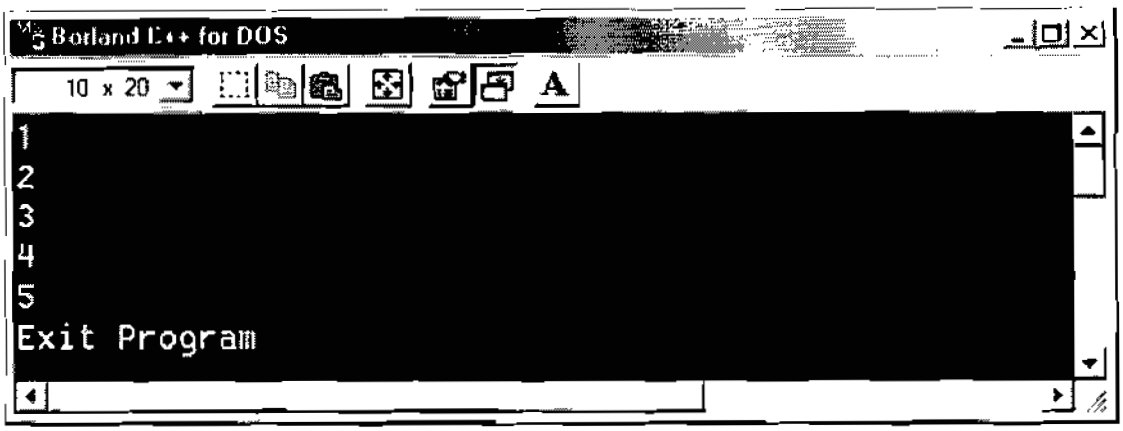
จากตัวอย่าง การแปลงค่าด้วย ฟังก์ชัน atol ซึ่งจะทำการแปลงค่าจาก ข้อความ เป็นตัวเลขจำนวนเต็ม จากข้อมูลตัวอย่าง 1230.5abc จะถูกแปลงเป็น 1230 และ 4560a4a6x จะถูกแปลงเป็น 456 เมื่อนำมาคำนวณ โดยนำมารวมกัน จึงได้ผลลัพธ์ เป็น 5790

exit(status)

ฟังก์ชันนี้จะยกเลิกการทำงานของโปรแกรมในขณะนั้น ซึ่งเหมือนกับการจบโปรแกรมตามปกติ สถานะ การจบโปรแกรม โดยปกติจะมีค่าเป็นศูนย์ถ้าเป็นจบโปรแกรมตามปกติ สำหรับค่า สถานะการจบอื่น ๆ จะใช้แทนข้อผิดพลาดต่าง ๆ

โปรแกรมที่ 3.24 แสดงตัวอย่างการจบโปรแกรมโดยใช้ฟังก์ชัน exit(0) ซึ่งเป็นค่าของสถานการณ์จบโปรแกรม แบบปกติ สังเกตผลลัพธ์ที่ได้จากการจากฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <stdlib.h>
4   main()
5   { int i;
6     clrscr();
7     for (i=1;i<=10;i++)
8     {
9      if (i > 5)
10       {cout << "Exit Program";exit(0);}
11     else
12       cout<<i<<"\n";
13    }
14    cout<<"\n End Program Press Enter";getch();
15    return 0;
16  }
```

ภาพที่ 3.26 ผลลัพธ์ของการใช้ฟังก์ชัน exit

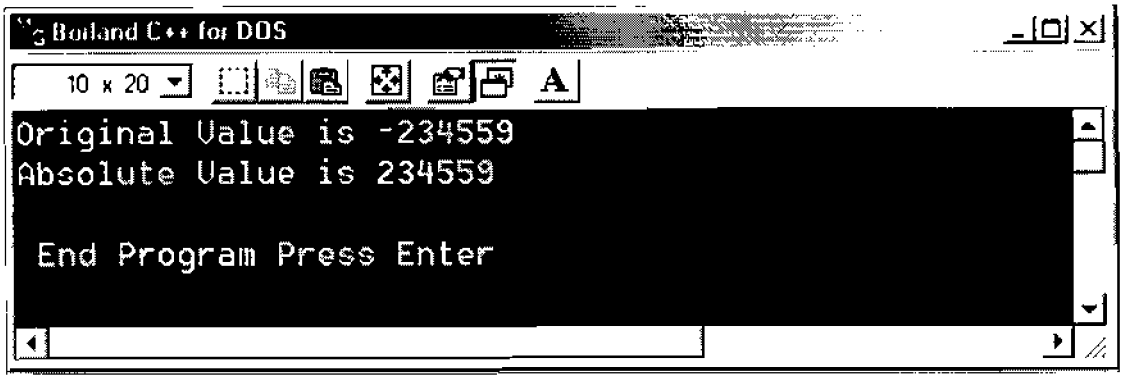
จากตัวอย่าง การใช้ฟังก์ชันการจบโปรแกรม ด้วยฟังก์ชัน exit(0) เครื่องจะถือว่า สถานะการจบโปรแกรมในแบบนี้ เป็นการจบโปรแกรมตามปกติ โดยก่อนจบ จะมีการแสดง ข้อความว่า Exit Program

labs(x)

ฟังก์ชันนี้จะให้ค่าสมบูรณ์ของ x ซึ่งเป็นชนิดเลขจำนวนเต็มยาว ซึ่งแตกต่างจากฟังก์ชัน abs(x) ที่เป็นจำนวนเต็มปกติ มีขนาด 2 ไบท์ แต่จำนวนเต็มยาวจะ มีขนาด 4 ไบท์

โปรแกรมที่ 3.25 แสดงตัวอย่างการหาค่าสัมบูรณ์ของตัวเลขจำนวนเต็มยาว ซึ่งจะมีขนาดการเก็บข้อมูล เป็นขนาด 4 ไบท์ สังเกตผลลัพธ์ที่ได้จากฟังก์ชัน

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <stdlib.h>
4   main()
5   { long int i= -234559;
6     clrscr();
7     cout<<"Original Value is "<<i<<"\n";
8     cout<<"Absolute Value is "<<labs(i)<<"\n";
9     cout<<"\n End Program Press Enter";getch();
10    return 0;
11  }
```



ภาพที่ 3.27 ผลลัพธ์ของการใช้ฟังก์ชัน labs

จากโปรแกรมตัวอย่าง กำหนดให้ตัวแปร i มีค่าเท่ากับ -234559 ซึ่งเป็นค่าที่เป็น จำนวนเต็มยาวแล้วนำค่าของ i ส่งเข้าไปใน ฟังก์ชัน labs จะได้ค่าสัมบูรณ์ของตัวเลข ซึ่งมีค่าเท่ากับ 234559

สรุป

คำสั่งพื้นฐานในภาษาซี ที่กล่าวในบทนี้ เป็นคำสั่งสำหรับการแสดงผล ใช้สำหรับการแสดงผลข้อมูลต่าง ๆ ตามที่ต้องการ คำสั่งรับข้อมูล เป็นคำสั่งสำหรับให้ผู้ใช้โปรแกรม ป้อนข้อมูลเข้ามาทางแป้นพิมพ์ เพื่อนำข้อมูลดังกล่าวไปใช้ในโปรแกรม และ คำสั่งกำหนดค่า เป็นคำสั่งที่ใช้สำหรับการกำหนดค่าต่าง ๆ ให้กับตัวแปร รวมไปถึงการคำนวณทางคณิตศาสตร์ สำหรับการคำนวณในเครื่องคอมพิวเตอร์ มีการจัดลำดับการคำนวณไว้ตามลำดับความสำคัญของเครื่องหมาย นอกจากการสั่งงานด้วยการเขียนคำสั่งตามที่ต้องการแล้ว ภาษาซียังได้เตรียม ฟังก์ชันมาตรฐานไว้เพื่อการใช้งานให้สะดวกยิ่งขึ้น แบ่งเป็นฟังก์ชันทางคณิตศาสตร์ ฟังก์ชันเกี่ยวกับตัวอักษร ฟังก์ชันเกี่ยวกับสตริง และฟังก์ชันทั่วไป สิ่งสำคัญอีกประการหนึ่งคือ ไลบรารี่ไฟล์ที่เก็บรวบรวมเอาคำสั่งสำเร็จที่สำคัญไว้ ดังกล่าวไว้แล้วในส่วนต้นของบทนี้ คำสั่งพื้นฐานของภาษาซี เป็นคำสั่งที่สามารถนำไปเขียน โปรแกรมได้ในระดับหนึ่ง ส่วนการเขียนโปรแกรมเพื่อตอบสนองความต้องการการใช้งานให้ได้มากขึ้น จะกล่าวไว้ในบทต่อไป

คำถามทบทวน

จงตอบคำถามต่อไปนี้โดยสังเขป

1. ไลบรารี่ในภาษาซี หมายถึง อะไร มีประโยชน์อย่างไร
2. จงยกตัวอย่างไลบรารี ที่ใช้ในการเขียนโปรแกรม 5 ตัวอย่าง
3. คำสั่งในการรับข้อมูลมีคำสั่งอะไรบ้าง จงยกตัวอย่างพร้อมอธิบาย
4. คำสั่งแสดงผล มีคำสั่งอะไรบ้าง จงยกตัวอย่างพร้อมอธิบาย
5. คำสั่งกำหนดค่ามีประโยชน์อย่างไรบ้าง จงยกตัวอย่างพร้อมอธิบาย
6. จงยกตัวอย่าง ฟังก์ชันมาตรฐานที่เกี่ยวข้องกับตัวอักษร
7. จงยกตัวอย่าง ฟังก์ชันมาตรฐานที่เกี่ยวข้องกับตัวเลข

เอกสารอ้างอิง

ธันวา ศรีประโมง. (2539). **การเขียนโปรแกรมภาษาซีสำหรับวิศวกรรม.** กรุงเทพฯ: มหาวิทยาลัยเทคโนโลยีมหานคร.

เบญจพร ศักดิ์ศิริ. (2540). **ทฤษฎีและตัวอย่างโจทย์ การเขียนโปรแกรมด้วยภาษา C++.** กรุงเทพฯ: แมคกรอ-ฮิล อินเตอร์เนชั่นแนล เอ็นเตอร์ไพร์ส, อิงค์.

มนตรี พจนารถลาวัณย์. (2535). **การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี.** กรุงเทพฯ: เอช-เอน การพิมพ์.

วรรณวิภา จำเริญคารารัศมี. (2535). **วิทยาการคอมพิวเตอร์เบื้องต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

แผนบริหารการสอนประจำบทที่ 4

เนื้อหาประจำบท

การทำงานแบบลำดับ

การทำงานแบบเลือกทำ

การทำงานแบบทำซ้ำ

สรุป

คำถามทบทวน

จุดประสงค์เชิงพฤติกรรม

1. เพื่อให้ผู้เรียนอธิบายและเขียนคำสั่งในการทำงานแบบลำดับได้
2. เพื่อให้ผู้เรียนอธิบายและเขียนคำสั่งในการทำงานแบบเลือกทำได้
3. เพื่อให้ผู้เรียนอธิบายและเขียนคำสั่งในการทำงานแบบวงรอบได้

วิธีสอนและกิจกรรมการเรียนการสอน

1. สอนแบบบรรยาย นำเข้าสู่บทเรียนด้วยการกล่าวถึงการสั่งให้คอมพิวเตอร์ทำงานในลักษณะ ที่มีใช้กันในชีวิตประจำวัน
2. กิจกรรมการเรียนการสอน
   2.1 บรรยาย และ แสดงรูปแบบของคำสั่ง พร้อมตัวอย่างคำสั่งในแบบต่างๆ
   2.2 ฝึกปฏิบัติ ด้วยการเขียนโปรแกรมตามกำหนด
   2.3 ทำแบบฝึกหัดท้ายบท

สื่อการเรียนการสอน

1. เอกสารประกอบการเรียนบทที่ 4
2. เพาเวอร์พอยท์ พรีเซนเตชั่น (power point presentation) ประกอบการบรรยาย
3. เครื่องคอมพิวเตอร์ที่สามารถใช้โปรแกรมภาษาซีได้

การวัดและการประเมินผล

1. สังเกตจากการตอบคำถาม และการตั้งคำถาม ในชั้นเรียน
2. วัดเจตคติจากการสังเกตพฤติกรรม ความกระตือรือร้นในการทำกิจกรรม
3. ความถูกต้องและคุณภาพของการทำแบบฝึกหัดท้ายบทเรียน

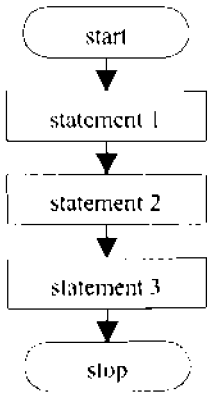
บทที่ 4
คำสั่งควบคุมการทำงาน

โดยปกติแล้ว โปรแกรมในภาษาซี มีการทำงานแบบ ทำจากซ้ายไปขวา และ จากบนลงล่าง การทำงานจะทำการดำเนินการตามคำสั่งทุกคำสั่ง ไม่มีการยกเว้น เป็นลักษณะพื้นฐานของโปรแกรม แต่บางครั้ง เราต้องการให้ โปรแกรมมีการตัดสินใจ เพื่อเลือกการทำงานตามการตัดสินใจเพื่อให้เกิดข้อแตกต่างของผลลัพธ์ เช่นการตัดสินใจในการ เลื่อนขั้นเงินเดือนให้กับพนักงาน การตัดสินใจลดราคาสินค้าบางรายการที่ขายไม่ค่อยดี เราจึงจำเป็นต้องเรียนรู้ถึงคำสั่งในลักษณะ ดังกล่าว นอกจากนั้นแล้ว การดำเนินการต่าง ๆ บางครั้งต้องดำเนินการซ้ำ หลายครั้งเพื่อให้ได้ปริมาณงานมากขึ้น เช่น การรวมคะแนนนักเรียนทั้งชั้นเรียน การเลื่อนขั้นเงินเดือนให้กับพนักงานทั้งหน่วยงาน การทำงานในลักษณะดังที่กล่าว จึงพอสรุปได้ว่า การทำงานของโปรแกรมโดยทั่วไป จะมีลักษณะเฉพาะที่สำคัญอยู่ 3 ลักษณะดังนี้

1. การทำงานแบบลำดับ (sequential)
2. การทำงานแบบเลือกทำ (decision)
3. การทำงานแบบทำซ้ำ (repetition)

การทำงานแบบลำดับ

การทำงานแบบลำดับเป็นลักษณะการทำงานในแบบพื้นฐานของโปรแกรมโดยทั่วไป รวมถึงโปรแกรมในภาษาซี มีลักษณะการทำงาน โดยเริ่มประมวลผลคำสั่งจากซ้ายไปขวา จากบนลงล่าง ทำทุกคำสั่งไม่มีการยกเว้น โปรแกรมในลักษณะนี้เรียกว่าการทำงานในแบบลำดับ เนื่องจากภาษาซี เป็นโปรแกรมที่ไม่มีหมายเลขบรรทัด การทำงานจึงมีลักษณะเป็นกลุ่มข้อความ หรือ บล็อก ภายในบล็อก จะมีลำดับการทำงาน จากบนลงล่าง เริ่มต้นแต่ละบล็อกด้วย เครื่องหมาย { และปิดด้วยเครื่องหมาย } ดังนั้น ถ้าไม่มีการกำหนด บล็อก เลย ก็จะต้องมี บล็อก หลักที่ชื่อ ฟังก์ชันเมน หรือ ฟังก์ชันหลัก แสดงเป็นผังงานได้ดังภาพที่ 4.1



ภาพที่ 4.1 ผังงานแสดงการทำงานในแบบลำดับ

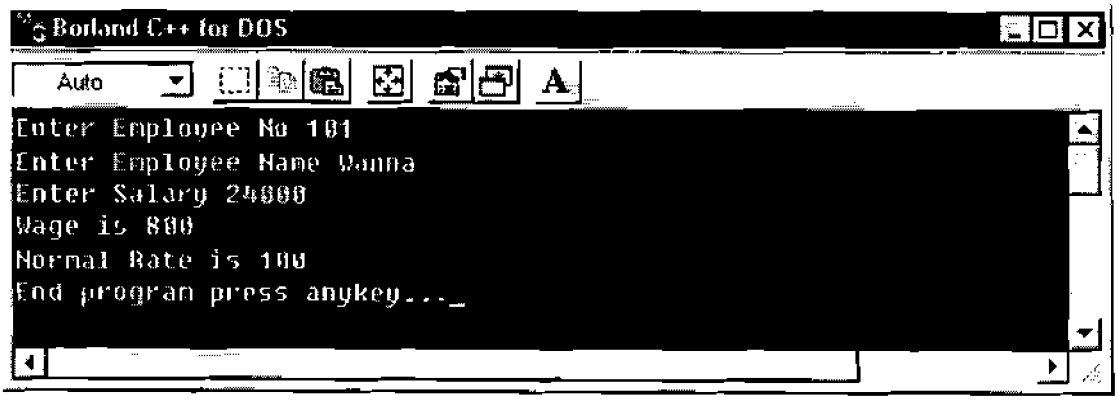
ตัวอย่างการทำงานแบบลำดับ โดยทั่วไปจะเป็นการประมวลผลที่ไม่มีเงื่อนไข หมายถึง เมื่อมีการประมวลผล ก็จะประมวลผลในลักษณะเดียวกันหมด เช่น การรวมคะแนนของนักเรียนในแต่ละชั้น การแสดงราคาสินค้าที่ลูกค้าซื้อ การพิมพ์รายชื่อของสมาชิกสหกรณ์ เป็นต้น เพื่อให้เห็นตัวอย่างของโปรแกรม ในลักษณะลำดับจึงนำเสนอตัวอย่างการประมวลผลในแบบลำดับต่อไปนี้

โปรแกรมที่ 4.1 ตัวอย่างการทำงานในแบบลำดับในการ คำนวณหาค่าแรงของพนักงานเป็นรายวัน รายชั่วโมง จาก เงินเดือนที่พนักงานได้รับ กำหนดให้มีรายละเอียดของข้อมูล และตัวแปรดังต่อไปนี้

ชื่อตัวแปร	ความหมาย	ประเภท
no	เลขที่	char
name	ชื่อ	char
salary	เงินเดือน	int
wage	ค่าแรงรายวัน	float
nor_rate	ค่าแรงชม.ปกติ	float

```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   { char no[3],name[20];
5     int salary;
6     float wage,nor_rate;
7     clrscr();
8     cout <<"Enter Employee No ";cin >>no;
9     cout <<"Enter Employee Name ";cin>>name;
10    cout <<"Enter Salary ";cin>>salary;
11     wage = salary / 30;
12     nor_rate = wage / 8;
13    cout <<"Wage is "<< wage<<"\n";
14    cout <<"Normal Rate is " <<nor_rate<<"\n";
15    cout <<"End program press anykey...";getch();
16    return 0;
17  }
```

เมื่อทำการป้อนข้อมูลให้กับโปรแกรม จะมีผลลัพธ์ออกมา ดังภาพที่ 4.2



ภาพที่ 4.2 ผลลัพธ์ของโปรแกรมที่มีการประมวลผลแบบลำดับ

การทำงานแบบเลือกทำ

การทำงานแบบเลือกทำ หมายถึง การทำงานในลักษณะที่มีการตัดสินใจ ในการทำงาน ว่าจะทำหรือ ไม่ทำ เพื่อให้ได้ผลลัพธ์ที่แตกต่างกัน ในการตัดสินใจเครื่องจะทำการเปรียบเทียบ ข้อมูลตั้งแต่ 2 ตัวขึ้นไป โดยทำการเปรียบเทียบตามความหมายของ ตัวดำเนินการทางตรรก หรือ ตัวดำเนินการทางการเปรียบเทียบ เพื่อให้ได้คำตอบทาง ตรรกะ คือ จริง หรือ เท็จ คำสั่งในภาษาซีที่ใช้สำหรับการเลือกทำมี 2 คำสั่งคือ

1. การเลือกทำด้วยคำสั่ง if
2. การเลือกทำด้วยคำสั่ง switch

การเลือกทำด้วยคำสั่ง if แบ่งเป็น 2 ลักษณะคือ การเลือกทำแบบ 1 ทางเลือก กับการเลือกทำแบบ 2 ทางเลือก มีรูปแบบของคำสั่งดังนี้

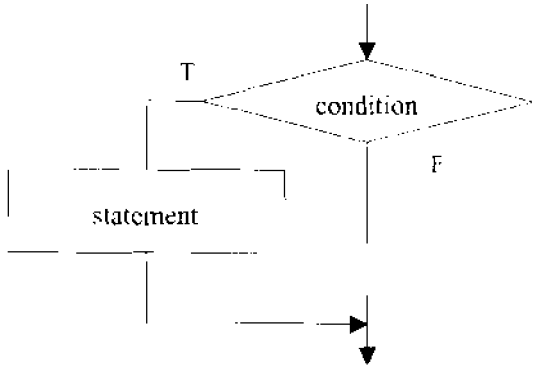
รูปแบบของการเลือกทำแบบ 1 ทางเลือก

```
if (condition) statement;
```

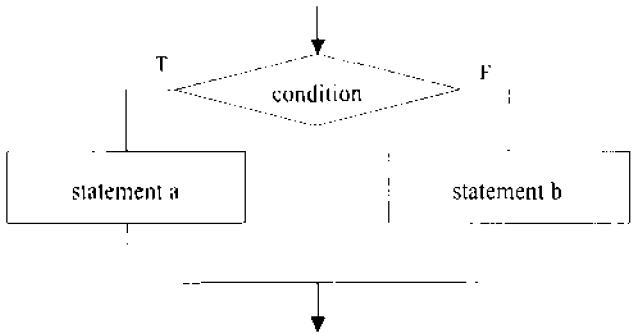
รูปแบบของการเลือกทำแบบ 2 ทางเลือก

```
if (condition) statement-1; else statement-2;
```

จากรูปแบบของคำสั่งจะเกิดการเปรียบเทียบ ข้อมูล เรียกว่า เงื่อนไข (condition) ผลลัพธ์จากการเปรียบเทียบ จะได้ค่า จริง (true :1) หรือ เท็จ (false : F) ถ้าเงื่อนไขเป็นจริง จะทำงานตาม ประโยคคำสั่ง (statement) ที่อยู่หลังเงื่อนไข ถ้าเงื่อนไขเป็น เท็จ จะไปทำคำสั่งอื่นต่อไป ในการเลือกทำแบบ 1 ทางเลือก ส่วนการเลือกทำแบบ 2 ทางเลือกนั้น ถ้าเงื่อนไขเป็นจริง ยังคงทำงานเหมือนแบบ 1 ทางเลือก แต่ถ้าเงื่อนไขเป็นเท็จ จะเลือกทำประโยคคำสั่งที่อยู่หลัง else เพื่อให้เห็นการทำงานของทั้ง 2 คำสั่ง แสดงเป็นผังงานได้ดังภาพที่ 4.3 และ ภาพที่ 4.4



ภาพที่ 4.3 ผังงานการเลือกทำแบบ 1 ทางเลือก



ภาพที่ 4.4 ผังงานการเลือกทำแบบ 2 ทางเลือก

การเขียนเงื่อนไข เพื่อให้เกิดการเปรียบเทียบข้อมูล เรียกอีกอย่างหนึ่งว่า การเขียนนิพจน์ทางตรรกศาสตร์ รูปแบบการเขียนเงื่อนไขมีดังนี้

data_1 operator data_2

data_1 หมายถึง ข้อมูลตัวที่ 1 ที่จะนำมาทำการเปรียบเทียบ
data_2 หมายถึง ข้อมูลตัวที่ 2 ที่จะนำมาทำการเปรียบเทียบ
operator หมายถึง เครื่องหมายในการเปรียบเทียบ แสดงในตารางที่ 4.2

ตารางที่ 4.1 เครื่องหมายที่ใช้ในการเปรียบเทียบ

ตัวดำเนินการ	ความหมาย
>	มากกว่า
<	น้อยกว่า
>=	มากกว่าหรือเท่ากับ
<=	น้อยกว่าหรือเท่ากับ
==	เท่ากันกับ
!=	ไม่เท่ากันกับ
&&	AND
||	OR
!	NOT

ที่มา (เกษมสันต์ พานิชการ, 2537, หน้า 41)

ตัวอย่างการเขียนเงื่อนไข เขียนได้ดังต่อไปนี้

1. 4 > 5	ผลลัพธ์มีค่าเท่ากับ	เท็จ
2. 4 < 5	ผลลัพธ์มีค่าเท่ากับ	เท็จ
3. 4 >= 5	ผลลัพธ์มีค่าเท่ากับ	เท็จ
4. 4 <= 5	ผลลัพธ์มีค่าเท่ากับ	จริง
5. 4 == 4	ผลลัพธ์มีค่าเท่ากับ	จริง
6. 4 != 5	ผลลัพธ์มีค่าเท่ากับ	จริง

จากตัวอย่าง เป็นการเปรียบเทียบโดยใช้ค่าคงที่ กับ ค่าคงที่ การเปรียบเทียบในลักษณะนี้ จะได้รับคำตอบที่เป็นค่าคงที่ไปด้วย ดังนั้น จึงไม่มีประโยชน์ที่จะช่วยให้เกิดการตัดสินใจแทนคนเรา ดังนั้นเงื่อนไขที่จะนำมาใช้ในโปรแกรมคอมพิวเตอร์จึงอยู่ในรูปที่เป็นการใช้ตัวแปร มี 2 ลักษณะดังนี้

1. การเปรียบเทียบโดยใช้ ตัวแปร กับ ค่าคงที่ เช่น a = 5 หรือ j < 75 เป็นต้น
2. การเปรียบเทียบโดยใช้ ตัวแปร กับ ตัวแปร เช่น a > n หรือ c <> 1 เป็นต้น

ตัวอย่าง การนำเงื่อนไขมาใช้งานเช่น ต้องการตัดเกรดนักเรียน โดยใช้ เกณฑ์ดังนี้

คะแนนรวม	80	เกรด	A
คะแนนรวม	70-79	เกรด	B
คะแนนรวม	60-69	เกรด	C
คะแนนรวม	50-59	เกรด	D
คะแนนรวม	0-49	เกรด	E

<u>กำหนดให้</u> คะแนนรวม เก็บในตัวแปร score และ เกรด เก็บในตัวแปร g

1. การตัดสินว่าได้เกรด A เขียนเป็นภาษาซีได้ดังนี้

   if (score>=80) g = "A";

2. การตัดสินว่าได้เกรด E เขียนเป็นภาษาซีได้ดังนี้

   if (score<50) g = "B";

3. การตัดสินว่าได้เกรด B ใช้เงื่อนไข เพียง 1 เงื่อนไข ไม่ได้ จึงจำเป็นต้องใช้เงื่อนไข 2 เงื่อนไข คือ score >= 70 และ score < 80 การใช้เงื่อนไข ในลักษณะนี้ จำเป็นต้องใช้ตัวเชื่อมความสัมพันธ์ ระหว่างเงื่อนไข ซึ่งได้แก่ and และ or ในที่นี้ ถ้าเงื่อนไข 2 เงื่อนไขดังกล่าว เป็นเท็จอันใดอันหนึ่ง

แสดงว่า เป็นเกรดอื่น ที่ไม่ใช่ เกรด B แต่ถ้าเป็นจริงทั้ง 2 เงื่อนไข แสดงว่าเป็นเกรด B ดังนั้นจึงต้องใช้ตัวเชื่อมเงื่อนไขที่เป็น and เขียนเป็นภาษาซีได้ดังนี้

```
if (score>=70 && score<80) g="B";
```

4. การตัดสินว่าได้เกรด C และ D ใช้หลักการเดียวกันกับ เกรด B เขียนเป็นภาษาซีได้ดังนี้

```
if(score>=60 && score<70) g = "C";
if(score>=50 && score<60) g = "D";
```

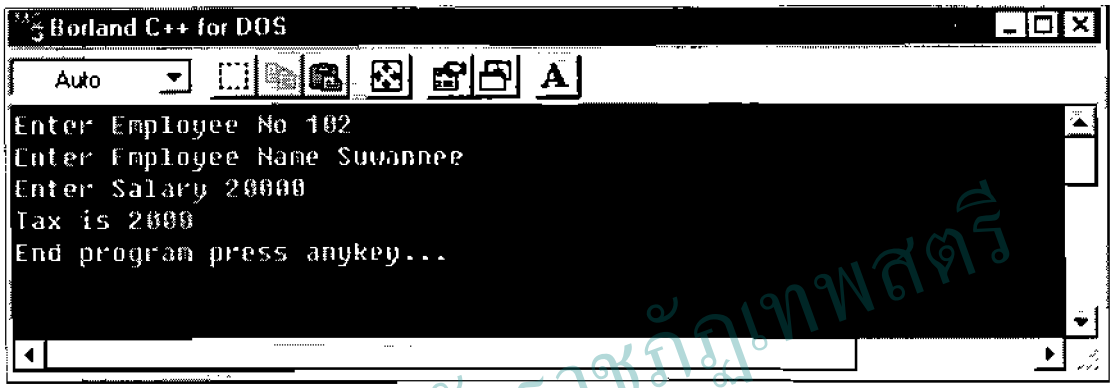
โปรแกรมที่ 4.2 ตัวอย่าง การคำนวณหาภาษีเงินได้ของพนักงาน ถ้ามีเงินเดือน เกิน 10,000 บาท จะต้องเสียภาษี 10% ของเงินเดือน กำหนดให้มีรายละเอียดของข้อมูล และตัวแปรดังต่อไปนี้

ชื่อตัวแปร	ความหมาย	ประเภท
no	เลขที่	char
name	ชื่อ	char
salary	เงินเดือน	int
tax	ภาษี	float


```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   {  char no[3],name[20];
5     int salary;
6     float tax=0;
7     clrscr();
8     cout <<"Enter Employee No ";cin >>no;
9     cout <<"Enter Employee Name ";cin>>name;
10    cout <<"Enter Salary ";cin>>salary;
```

```
11          if (salary > 10000) tax = salary * 0.1;
12        cout <<"Tax is " <<tax<<"\n";
13        cout <<"End program press anykey...";getch();
14        return 0;
15      }
```

ผลลัพธ์ของโปรแกรมเมื่อมีการป้อนข้อมูล แสดงดังภาพที่ 4.5



ภาพที่ 4.5 ผลลัพธ์ของโปรแกรมที่ใช้คำสั่งแบบ 1 ทางเลือก

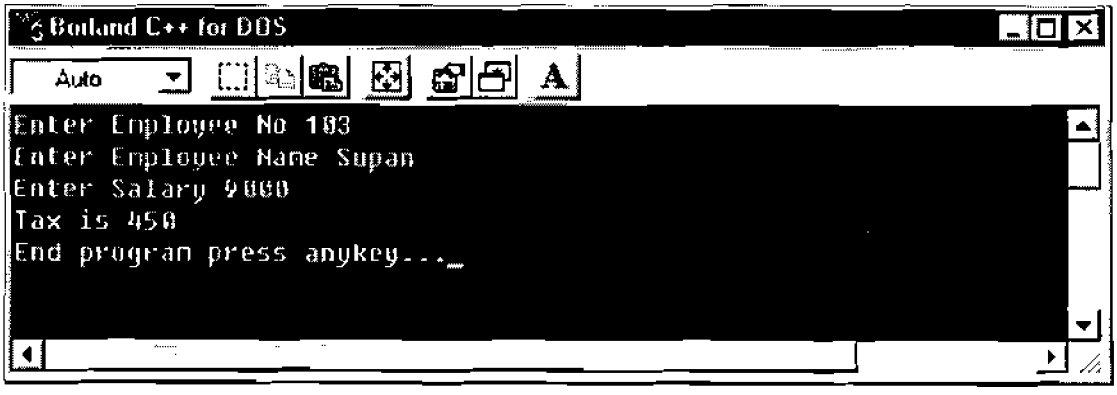
จากผลลัพธ์ของโปรแกรม การป้อนข้อมูลในบรรทัดที่ 10 ให้มี salary เกิน 10,000 โปรแกรมทำให้เงื่อนไขในบรรทัดที่ 11 เป็นจริง จึงคิดภาษีให้ 10% ของเงินเดือน ถ้าเงินเดือนไม่เกิน 10,000 ภาษีจะมีค่าเป็น 0

โปรแกรมที่ 4.3 ตัวอย่าง การคำนวณหาภาษีเงินได้ของพนักงาน ถ้ามีเงินเดือนไม่เกิน 10,000 บาท จะต้องเสียภาษี 5% ถ้าเกิน 10,000 บาท จะต้องเสียภาษี 10% ของเงินเดือนกำหนดให้มีรายละเอียดของข้อมูล และตัวแปรดังต่อไปนี้

ชื่อตัวแปร	ความหมาย	ประเภท
no	เลขที่	char
name	ชื่อ	char
salary	เงินเดือน	int
tax	ภาษี	float

```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   {  char no[3],name[20];
5      int salary;
6      float tax=0;
7      clrscr();
8      cout <<"Enter Employee No ";cin >>no;
9      cout <<"Enter Employee Name ";cin>>name;
10     cout <<"Enter Salary ";cin>>salary;
11        if (salary > 10000) tax = salary * 0.1;
12        else tax = salary * 0.05;
13     cout <<"Tax is " <<tax<<"\n";
14     cout <<"End program press anykey...";getch();
15     return 0;
16  }
```

ผลลัพธ์ของโปรแกรมเมื่อมีการป้อนข้อมูล แสดงดังภาพที่ 4.6



ภาพที่ 4.6 ผลลัพธ์ของโปรแกรมที่ใช้คำสั่งแบบ 2 ทางเลือก

จากผลลัพธ์ของโปรแกรม เป็นการป้อนข้อมูลโดยให้มีเงินเดือน 9,000 โปรแกรมจึงคิดภาษีให้ 5% ของเงินเดือนในตัวอย่างคือ 450 ถ้าเงินเดือนเกิน 10,000 ภาษีจะมีค่าเป็น 10%

โปรแกรมที่ 4.4 ตัวอย่าง การคำนวณหาภาษีเงินได้ของพนักงาน ถ้ามีเงินเดือนไม่เกิน 10,000 บาท จะต้องเสียภาษี 5% ถ้ามีเงินเดือนอยู่ในช่วง 10,000-15,000 จะต้องเสียภาษี 10% และถ้ามีเงินเดือนเกิน 15,000 บาท จะต้องเสียภาษี 15% ของเงินเดือนกำหนดให้มีรายละเอียดของข้อมูลและตัวแปรดังต่อไปนี้

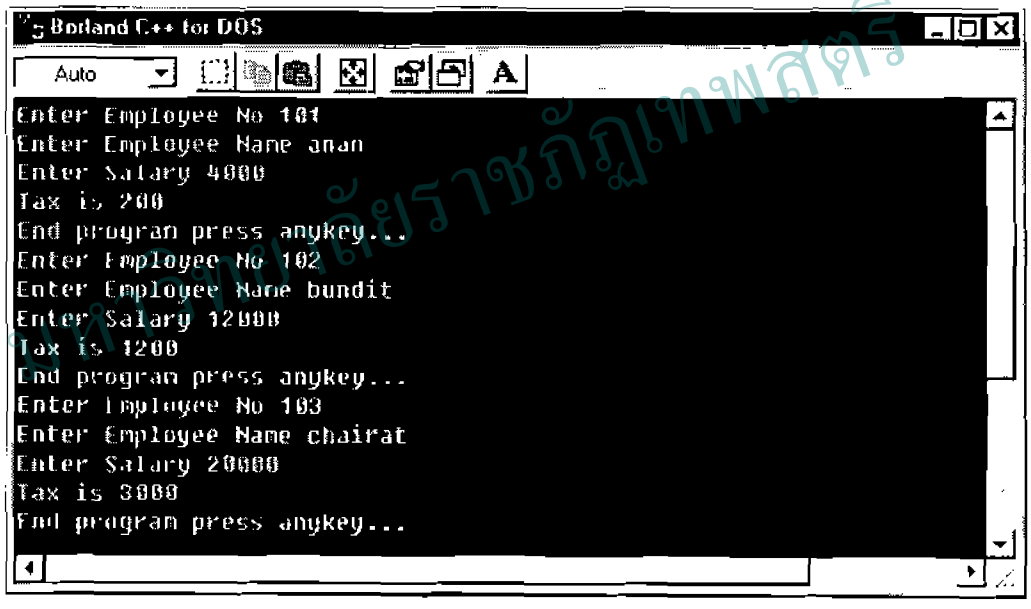
ชื่อตัวแปร	ความหมาย	ประเภท
no	เลขที่	char
name	ชื่อ	char
salary	เงินเดือน	int
tax	ภาษี	float

จากโจทย์ตัวอย่างนี้ จะเห็นว่ามีทางเลือกในการหักภาษี ถึง 3 ทางเลือก คือ 5% 10% และ15% จากคำสั่งที่ใช้ในการตัดสินใจเรามีทางเลือกแค่ 2 ทางเลือก ดังนั้น เราสามารถนำเอาคำสั่งในการตัดสินใจแบบ 2 ทางเลือก มาซ้อนกัน เพื่อให้เกิดทางเลือกที่ 3 ได้ ดังโปรแกรมต่อไปนี้


```
1   #include <conio.h>
2   main()
3   {  char no[3],name[20];
4      int salary;
5      float tax=0;
6      cout <<"Enter Employee No ";cin >>no;
7      cout <<"Enter Employee Name ";cin>>name;
8      cout <<"Enter Salary ";cin>>salary;
9        if (salary < 10000)
10                tax = salary * 0.05;
11       else
```

```
12              if (salary < 15000)
13                      tax = salary * 0.1;
14              else
15                      tax = salary * 0.15;
16      cout <<"Tax is " <<tax<<"\n";
17      cout <<"End program press anykey... \n";getch();
18      return 0;
19  }
```

เมื่อทำการป้อนข้อมูลดังตัวอย่างจะมีผลลัพธ์ดังภาพที่ 4.7



ภาพที่ 4.7 ผลลัพธ์ของโปรแกรมที่มีการนำคำสั่ง if มาซ้อนกัน

จากโปรแกรม กระบวนการตัดสินใจเริ่มต้นที่ บรรทัดที่ 9 ถึงบรรทัดที่ 15 ถ้าเงื่อนไขในบรรทัดที่ 9 เป็นจริง จะทำคำสั่งในบรรทัดที่ 10 แล้วกระโดดไปทำงานที่ บรรทัดที่ 16 แต่ถ้าเป็นเท็จ จะไปเช็คเงื่อนไขในบรรทัดที่ 12 ถ้าเป็นจริง จะทำคำสั่งในบรรทัดที่ 13 แล้วกระโดดไปทำงานที่บรรทัดที่ 16 แต่ถ้าเงื่อนไขเป็นเท็จ จะทำตามคำสั่งในบรรทัดที่ 15 กระบวนการตัดสินใจทั้งหมดจะทำให้เกิดการทำงานตามจุดประสงค์ของโปรแกรมได้

การเลือกทำด้วยคำสั่ง switch..case หมายถึง การเลือกทำโดยพิจารณาจากค่าของตัวแปรที่กำหนด เช่น ค่าตัวแปรเป็น 1 ทำทางเลือกที่ 1 ค่าตัวแปรเป็น 2 ทำทางเลือกที่ 2 เป็นต้น การเลือกทำด้วยคำสั่ง switch..case เหมาะสำหรับการเลือกทำแบบหลายทางทางเลือก และง่ายต่อการใช้งาน คำสั่ง switch..case จะต้องใช้คู่กับคำสั่ง break เสมอ ถ้าไม่มีคำสั่ง break หลังจากที่ทำงานตามทางเลือกแล้ว จะทำงานตามทางเลือกอื่นต่อ เช่น ถ้าเครื่องเลือกทำทางเลือกที่ 1 แล้วไม่พบคำสั่ง break เครื่องจะทำงานตามทางเลือกต่อไป จนหมดทางเลือก รูปแบบของคำสั่งมี ดังนี้

```
switch (variable or expression)
{     case value1  : statement1;
      case value2  : statement2;
      case value3  : statement3;
      [default     : statement4;]
}
```

ในการเลือกทำ ของคำสั่ง switch จะถูกกำหนดโดย ตัวแปร หรือ นิพจน์ ที่อยู่ใน วงเล็บ ว่ามีค่าเป็นเท่าใด แล้วเรื่องคอมพิวเตอร์ ก็จะทำการพิจารณาว่า ค่าของตัวแปร หรือ นิพจน์ มีค่าตรงกับ case ใด คำสั่ง ที่เขียนใน case นั้นๆ ก็จะถูกสั่งให้ทำงาน เพียง 1 คำสั่ง ส่วน default หมายถึง ถ้าค่าของตัวแปร หรือ นิพจน์ ไม่ตรงกับ case ใดเลย ให้เลือกทำคำสั่งที่อยู่หลัง default

โปรแกรมที่ 4.5 ตัวอย่าง การคำนวณหาเงินโบนัสของพนักงาน โดยพิจารณาว่าพนักงานคนนั้น อยู่ในระดับอะไร โดยมีเกณฑ์ดังต่อไปนี้

ระดับ 1 จ่ายเงินโบนัส 4 เท่า ของเงินเดือน

ระดับ 2 จ่ายเงินโบนัส 5 เท่า ของเงินเดือน

ระดับ 3 จ่ายเงินโบนัส 6 เท่า ของเงินเดือน

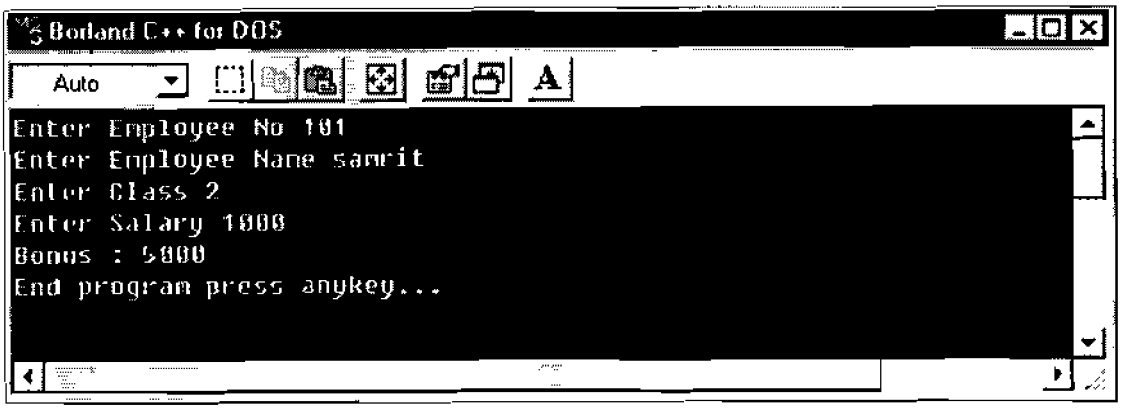
และนอกเหนือจาก 3 ระดับนี้ จ่ายเงินโบนัส 3 เท่าของเงินเดือน

กำหนดให้มีรายละเอียดของข้อมูล และตัวแปรต่อไปนี้

ชื่อตัวแปร	ความหมาย	ประเภท
no	เลขที่	char
name	ชื่อ	char
c	ระดับ	int
salary	เงินเดือน	int

```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   {  char no[3],name[20];
5      int c,salary,bonus;
6      clrscr();
7      cout <<"Enter Employee No ";cin >>no;
8      cout <<"Enter Employee Name ";cin>>name;
9      cout <<"Enter Class ";cin >> c;
10     cout <<"Enter Salary ";cin>>salary;
11     switch (c)
12     {   case 1: bonus = salary * 4;break;
13         case 2: bonus = salary * 5;break;
14         case 3: bonus = salary * 6;break;
15         default : bonus = salary * 3;
16     }
17     cout <<"Bonus : " <<bonus<<"\n";
18     cout <<"End program press anykey... \n";getch();
19     return 0;
20  }
```


จากโปรแกรมตัวอย่าง เมื่อสั่งให้โปรแกรมทำงานและป้อนข้อมูลดังตัวอย่าง จะมีผลลัพธ์ออกมาดังภาพที่ 4.8



ภาพที่ 4.8 ผลลัพธ์ของโปรแกรมที่ใช้คำสั่ง switch

จากตัวอย่าง สังเกตการป้อนข้อมูลให้กับตัวแปร c ให้มีค่า เท่ากับ 2 ดังนั้น โปรแกรมจึงสั่งให้เกิดการทำงาน ที่ case 2 แต่การทำงานของคำสั่ง case นั้นถ้าหากมีการเลือกทำงานในกรณีใดกรณีหนึ่งแล้ว ก็จะเลือก case อื่นต่อจากที่เลือกอีก ดังนั้น จึงต้องมีการใช้คำสั่ง break เพื่อหยุดการทำงาน ณ ตำแหน่งปัจจุบัน

การทำงานแบบทำซ้ำ

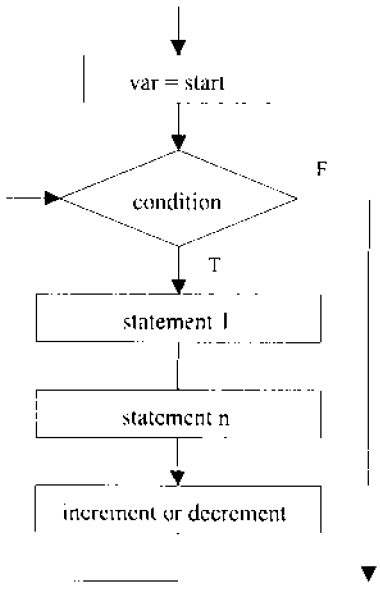
การทำงานแบบทำซ้ำหรือทำงานแบบวงรอบ (loop) เป็นการสั่งให้เครื่องทำงานแบบเดิมซึ่งก็จะหมายถึง การย้อนการทำงานกลับไปเริ่ม ต้นใหม่ ยังจุดที่เราระบุ การทำงานซ้ำทำให้เกิดประโยชน์ในด้านของปริมาณ นั่นหมายถึงว่า เราจะได้ปริมาณของสิ่งที่ทำซ้ำ ในจำนวนเท่าที่ต้องการ การทำซ้ำ การทำงานแบบทำซ้ำในภาษาซีมี 2 คำสั่งดังนี้

1. การทำซ้ำด้วยคำสั่ง for
2. การทำซ้ำด้วยคำสั่ง while

คำสั่ง for หมายถึง การสั่งให้เครื่องทำงานซ้ำเป็นแบบวงรอบ โดยใช้ค่าตัวแปร ชนิดตัวเลข เป็นตัวควบคุมการทำงาน โดยสร้างเงื่อนไขว่า จะทำซ้ำเมื่อค่าตัวแปรมีค่าเท่าใด และ จะหยุดการทำงานเมื่อตัวแปรมีค่าเท่าใด สำหรับค่าของตัวแปรที่ใช้สำหรับควบคุมการทำงานนั้น สามารถสั่งให้เพิ่มค่า หรือ ลดค่า ของตัวแปรได้ ตามความต้องการ มีรูปแบบดังนี้

```
for (var=start,condition test,increment or decrement)
        {       statement 1;
        statement 2;
        .
        .
        .
        statement n;
        }
```

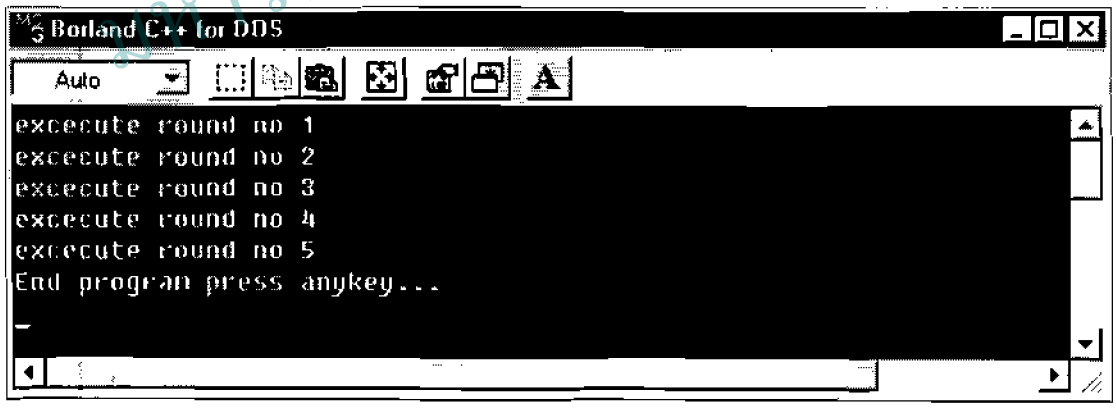
การทำงานของคำสั่ง for จะเริ่มต้นด้วยการกำหนดค่าให้กับ ตัวแปร ตามค่าที่เรากำหนด จากนั้น จะทำการทดสอบเงื่อนไขว่าเป็น จริง หรือ เท็จ ถ้าเงื่อนไขเป็นจริง จะทำคำสั่งในวงรอบ ในที่นี้หมายถึง statement 1 จนถึง statement n จากนั้นจะทำ การเพิ่ม (increment) หรือลดค่า (decrement) ตัวแปรที่กำหนด แล้วย้อนการทำงานกลับมาเริ่มต้นที่การทดสอบเงื่อนไขว่า จริง หรือ เท็จ ถ้าเงื่อนไขเป็นจริง ก็จะทำงานตามคำสั่งในวงรอบ แล้ว ทำการ เพิ่ม หรือ ลด ค่าตัวแปร จากนั้น ก็จะย้อนการทำงานกลับมาเริ่มที่การทดสอบเงื่อนไขอีก จนกระทั่งเงื่อนไขเป็นเท็จ การทำงานแบบทำซ้ำ จึงสิ้นสุดลง อธิบายเป็นผังงานได้ดัง ภาพที่ 4.9



ภาพที่ 4.9 ผังงานการทำงานของคำสั่ง for

โปรแกรมตัวอย่าง การใช้คำสั่งทำซ้ำแบบเพิ่มค่าตัวแปร

```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   { int i;
5     clrscr();
6     for(i=1;i<6;i++)
7     {
8       cout<<"excecute round no "<< i << "\n";
9     }
10    cout <<"End program press anykey... \n";getch();
11    return 0;
12  }
```



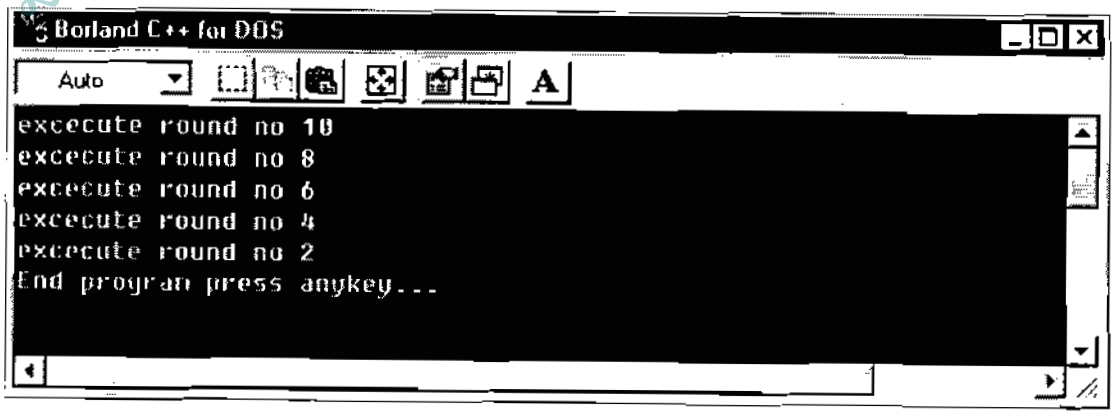
ภาพที่ 4.10 ผลลัพธ์ของการใช้คำสั่งทำซ้ำแบบเพิ่มค่าตัวแปร

จากโปรแกรมตัวอย่างจะเห็นว่ามีการใช้คำสั่งกำหนดค่าเริ่มต้นให้ตัวแปร i ให้มีค่าเท่ากับ 1 แล้วตรวจสอบด้วยเงื่อนไข i < 6 ถ้าเป็นจริงให้ทำคำสั่งในวงรอบคือ cout แล้วเพิ่มค่า i ขึ้นอีก 1 ด้วยคำสั่ง i++ แล้วย้อนกลับไปตรวจสอบเงื่อนไข จนกระทั่งเงื่อนไขเป็นเท็จจึงหยุดการทำงานแบบทำซ้ำ

โปรแกรมตัวอย่าง การใช้คำสั่งทำซ้ำแบบลดค่าตัวแปร

```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   { int i;
5     clrscr();
6     for(i=10;i>0;i=i-2)
7     {
8       cout<<"excecute round no "<< i << "\n";
9     }
10    cout <<"End program press anykey... \n";getch();
11    return 0;
12  }
```

จากโปรแกรมตัวอย่าง มีผลลัพธ์ออกมาดังภาพที่ 4.11



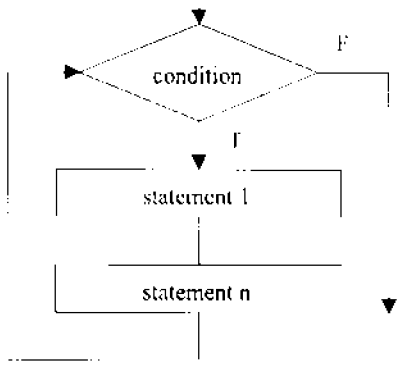
ภาพที่ 4.11 ผลลัพธ์ของการใช้คำสั่งทำซ้ำแบบลดค่าตัวแปร

จากโปรแกรมตัวอย่างจะเห็นว่ามีการใช้คำสั่งกำหนดค่าเริ่มต้นให้ตัวแปร i ให้มีค่าเท่ากับ 10 แล้วตรวจสอบด้วยเงื่อนไข i >0 ถ้าเป็นจริงให้ทำคำสั่งในวงรอบคือ cout แล้วลดค่า i ลง 2 ด้วยคำสั่ง i = i - 2 แล้วย้อนกลับไปตรวจสอบเงื่อนไข จนกระทั่งเงื่อนไขเป็นเท็จจึงหยุดการทำงานแบบทำซ้ำ

คำสั่ง while หมายถึง การสั่งให้เครื่องทำงานซ้ำเป็นแบบวงรอบโดยใช้เงื่อนไขในการควบคุมการทำงาน ซึ่งจะมีการทำงานตามคำสั่งที่อยู่ในวงรอบเมื่อเงื่อนไขเป็นจริงและหยุดเมื่อเงื่อนไขเป็นเท็จ เงื่อนไขของคำสั่ง while สามารถใช้ได้ทั้งตัวแปรแบบ ตัวเลข และ ตัวแปรประเภทตัวอักษร มีรูปแบบดังนี้

```
while (codition)
{     statement 1;
      statement 2;
      .
      .
      .
      statement n;
}
```

การทำงานจะเริ่มต้นที่คำสั่ง while แล้วทำการตรวจสอบเงื่อนไข ว่าเป็น จริง หรือ เท็จ ถ้าเงื่อนไขเป็นจริง จะทำงานตามคำสั่งที่อยู่ในวงรอบในที่นี้คือ statement 1 ถึง statement n แล้วย้อนกลับมาเริ่มต้นที่ while เพื่อตรวจสอบเงื่อนไขอีกครั้ง ถ้าเงื่อนไขเป็น จริง ก็จะทำตามคำสั่งในวงรอบอีก และ ถ้าเงื่อนไขเป็นเท็จ ก็จะหยุดการทำงานเพื่อไปทำคำสั่งอื่นต่อไป อธิบายเป็นผังงานได้ดังภาพที่ 4.12

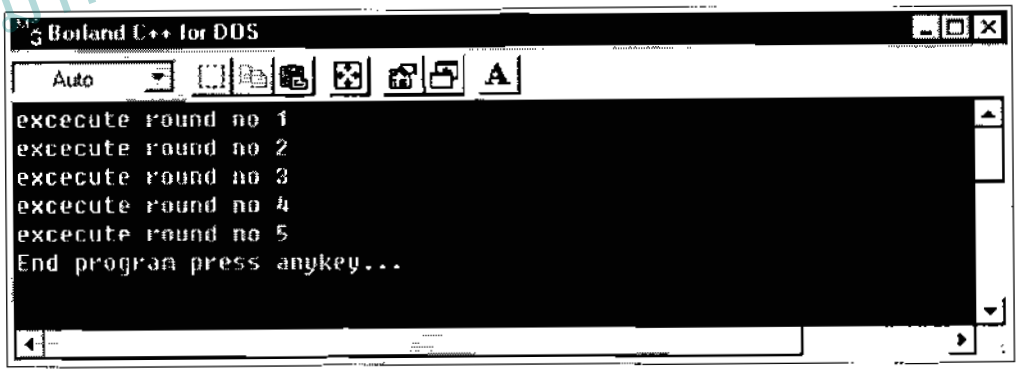


ภาพที่ 4.12 ผังงานการทำงานของคำสั่ง while

โปรแกรมตัวอย่าง การใช้วงรอบ while กับการเพิ่มค่าให้กับตัวแปร

```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   { int i=0;
5     clrscr();
6     while (i<5)
7     {
8       i++;
9       cout<<"excecute round no "<< i << "\n";
10    }
11    cout <<"End program press anykey... \n";getch();
12    return 0;
13  }
```

จากโปรแกรมตัวอย่าง มีผลลัพธ์ออกมาดังภาพที่ 4.13



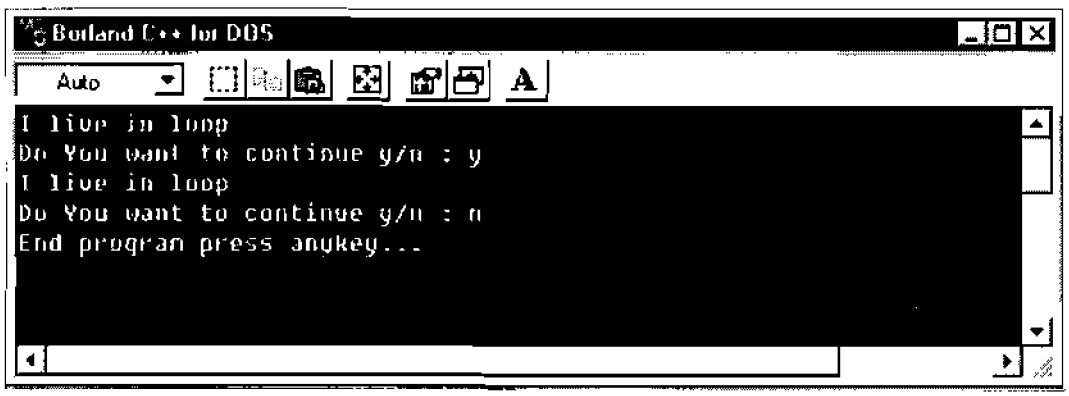
ภาพที่ 4.13 ผลลัพธ์ของโปรแกรมที่มีการใช้คำสั่ง while แบบเพิ่มค่าให้กับตัวแปร

จากโปรแกรมตัวอย่าง การทำงานของโปรแกรมจะเริ่มจาก การตรวจสอบเงื่อนไขหลัง while ว่า เป็นจริงหรือเท็จ ถ้าเป็นจริงจะดำเนินการตามคำสั่งที่อยู่ระหว่างวงเล็บปีกกาเปิด ({) จนถึงวงเล็บปีกกาปิด (}) เสร็จแล้วจะย้อนกลับมาเริ่มต้นที่ while เพื่อตรวจสอบเงื่อนไขในแบบเดิมอีกครั้ง จนกระทั่งเงื่อนไขเป็นเท็จจึงไปทำคำสั่งอื่นต่อไป

โปรแกรมตัวอย่าง การใช้วงรอบ while กับการใช้คำสั่ง cin

```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   { char c = 'y';
5     clrscr();
6     while (c=='y')
7     {
8       cout<<"I live in loop \n";
9       cout<<"Do You want to continue y/n : ";
10      cin>>c;
11    }
12    cout <<"End program press anykey... \n";getch();
13    return 0;
14  }
```

จากโปรแกรมตัวอย่าง มีผลลัพธ์ออกมาดังภาพที่ 4.14



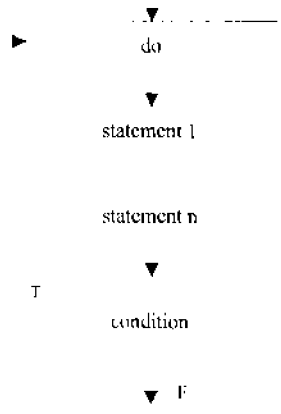
ภาพที่ 4.14 ผลลัพธ์ของโปรแกรมที่มีการใช้คำสั่ง while กับคำสั่ง cin

จากโปรแกรมตัวอย่าง การทำงานของโปรแกรมจะเริ่มจาก การตรวจสอบเงื่อนไขหลัง while ว่า เป็นจริงหรือเท็จ ถ้าเป็นจริงจะดำเนินการตามคำสั่งที่อยู่ระหว่างวงเล็บปีกกาเปิด ({) จนถึงวงเล็บปีกกาปิด (}) เสร็จแล้วจะย้อนกลับมาเริ่มต้นที่ while เพื่อตรวจสอบเงื่อนไขในแบบเดิมอีกครั้ง จนกระทั่งเงื่อนไขเป็นเท็จจึงไปทำคำสั่งอื่นต่อไป

คำสั่ง do...while หมายถึง คำสั่งสำหรับการทำงานแบบวงรอบอีกคำสั่งหนึ่ง มีการทำงานคล้ายกับ คำสั่ง while แตกต่างเพียงแค่ คำสั่ง do...while จะทำคำสั่งในวงรอบก่อนแล้วจึงตรวจสอบเงื่อนไขที่อยู่หลัง while ถ้าเงื่อนไขเป็นจริง จะย้อนการทำงานกลับมาเริ่มต้นที่ do แต่ถ้าเงื่อนไขเป็นเท็จจะหยุดการทำงานแบบวงรอบเพื่อไปทำคำสั่งอื่น ๆ ต่อไปมีรูปแบบดังนี้

```
do
{   statement_1;
    statement_2;
    .
    .
    .
    statement_n;
} while (condition)
```

จากรูปแบบคำสั่ง การทำงานจะเริ่มต้นที่คำสั่ง do แล้วจะทำงานตามคำสั่งที่อยู่ในวงรอบ ได้แก่ statement 1จนถึง statement n จึงทำการตรวจสอบเงื่อนไขหลังคำสั่ง while ถ้าเงื่อนไขเป็นจริง การทำงานจะกลับไปเริ่มต้นใหม่ที่คำสั่ง do แต่ถ้าเงื่อนไขเป็นเท็จจะหยุดการทำงานแบบวงรอบ อธิบายเป็นผังงานได้ดังภาพที่ 4.15

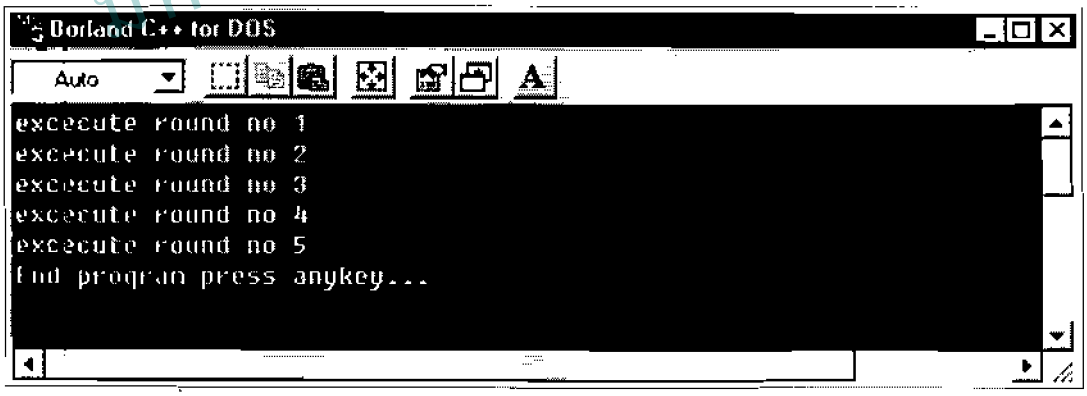


ภาพที่ 4.15 ผังงานแสดงการทำงานของคำสั่ง do..while

โปรแกรมตัวอย่าง การใช้วงรอบ do...while กับการเพิ่มค่าให้กับตัวแปร

```
1   #include <iostream.h>
2   #include <conio.h>
3   main()
4   {  int i=0;
5     clrscr();
6     do
7     {
8       i++;
9       cout<<"excecute round no "<< i << "\n";
10    } while (i<5);
11    cout <<"End program press anykey... \n";getch();
12    return 0;
13  }
```

จากโปรแกรมตัวอย่างมีผลลัพธ์ออกมาดังภาพที่ 4.16



ภาพที่ 4.16 ผลลัพธ์ของการใช้คำสั่ง do...while แบบเพิ่มค่าให้กับตัวแปร

จากโปรแกรมตัวอย่าง การทำงานของโปรแกรมจะเริ่มจากคำสั่ง do แล้วดำเนินการตามคำสั่งที่อยู่ระหว่างวงเล็บปีกกาเปิด ({)จนถึง วงเล็บปีกกาปิด (}) เสร็จแล้ว ทำการตรวจสอบเงื่อนไขหลังคำสั่ง while ถ้าเป็นจริง จะย้อนกลับมาเริ่มต้นที่ do เพื่อทำงานตามคำสั่งในวงรอบ เมื่อดำเนินการถึงคำสั่ง while จะทำการตรวจสอบเงื่อนไขในแบบเดิม จนกระทั่งเงื่อนไขเป็นเท็จจึงไปทำคำสั่งอื่นต่อไป

สรุป

ภาษาซีเป็นโปรแกรมประเภทโครงสร้าง ซึ่งเป็นที่นิยมใช้กันในปัจจุบัน เพราะโปรแกรมโครงสร้างทำให้ ง่ายแก่การควบคุมการทำงานของโปรแกรม ลักษณะการทำงานของโปรแกรมโครงสร้างที่สำคัญคือ การมีทางเข้าทางเดียว และ มีทางออกทางเดียว จึงสามารถตรวจสอบค่าต่าง ๆ ได้ง่าย โครงสร้างของโปรแกรมมีด้วยกัน 3 แบบคือ แบบทำงานตามลำดับ แบบทางเลือก และ แบบทำซ้ำ ในแบบลำดับจะมีการทำงาน ตามลำดับของคำสั่งที่สั่งลงไป คำสั่งใดอยู่ในอันดับต้น จะถูกทำก่อนอันดับหลังเสมอ แบบทางเลือก หมายถึง การสั่งให้คอมพิวเตอร์เกิดการตัดสินใจว่า จะเลือกทำคำสั่งที่มีใส่ไว้หรือไม่ ทั้งนี้ขึ้นอยู่กับเงื่อนไข เป็นตัวควบคุมคำสั่งแบบทางเลือก มี 3 แบบได้แก่ 1 ทางเลือก 2 ทางเลือก และ หลายทางเลือก โครงสร้างแบบทำซ้ำ หมายถึงการที่เราต้องการสั่งให้เครื่องทำงานในแบบเดิม หลาย ๆ ครั้ง การทำซ้ำแบ่งเป็น 2 ประเภทได้แก่ การทำซ้ำโดยใช้ ตัวนับ กับการทำซ้ำโดยใช้เงื่อนไข

คำถามทบทวน

จงตอบคำถามต่อไปนี้โดยสังเขป

1. จงอธิบายถึงโครงสร้างการทำงานแบบลำดับ แบบทางเลือก และ แบบทำซ้ำ
2. ต้องการให้มีการลดราคาสินค้าในแผนกเครื่องเขียนลง 10% จากยอดซื้อ สำหรับลูกค้าที่ซื้อสินค้าเกิน 500 บาท จะมีวิธีการทำอย่างไรจงเขียนโปรแกรมพร้อมอธิบาย
3. ต้องการให้มีการหักภาษี ณ ที่จ่ายสำหรับพนักงาน เป็นจำนวน 5% และ 10% โดยพิจารณาจากรายได้ ซึ่งเกิดจาก การทำงานในชั่วโมงล่วงเวลา และ เงินเดือน ถ้าพนักงานมีรายได้เกิน 30000 บาท ให้หักภาษี 10% นอกเหนือจากนั้นหักภาษี 5% จงเขียนโปรแกรมพร้อมอธิบาย
4. ต้องการหาผลบวกเลขคู่ จาก 1 ถึง 255 จงเขียนโปรแกรมพร้อมอธิบาย
5. ต้องการให้หาค่าเฉลี่ยของตัวเลขที่ผู้ใช้ป้อนเข้ามา โดยไม่จำกัดจำนวน การป้อนข้อมูลจะสิ้นสุดลงเมื่อ ผู้ใช้ป้อนตัวเลขที่มีค่า –1 จงเขียนโปรแกรมพร้อมอธิบาย

เอกสารอ้างอิง

เกษมสันต์ พานิชการ. (2537). **C++ และหลักการของ OOP ฉบับเริ่มต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

ธันวา ศรีประโมง. (2539). **การเขียนโปรแกรมภาษาซีสำหรับวิศวกรรม.** กรุงเทพฯ: มหาวิทยาลัยเทคโนโลยีมหานคร.

มนตรี พจนารถลาวัณย์. (2535). **การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี.** กรุงเทพฯ: เอช-เอน การพิมพ์.

วรรณวิภา จำเริญดารารัศมี. (2535). **วิทยาการคอมพิวเตอร์เบื้องต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

ศิริรัตน์ ชำนาญรบ, เกื้อกูล ตาเย็น, และวัชระ โพธิสรณ์. (2539). **ความรู้เบื้องต้นเกี่ยวกับคอมพิวเตอร์.** กรุงเทพฯ: ภูมิบัณฑิต.

แผนบริหารการสอนประจำบทที่ 5

เนื้อหาประจำบท

ความหมายของโปรแกรมย่อย

ประโยชน์ของโปรแกรมย่อย

โปรแกรมย่อยในภาษาซี

การส่งค่าผ่านพารามิเตอร์

ฟังก์ชันที่ไม่มีการส่งค่ากลับ

ฟังก์ชันที่มีการส่งค่ากลับ

ฟังก์ชันที่มีการส่งค่าไปกลับหลายค่า

สรุป

คำถามทบทวน

จุดประสงค์เชิงพฤติกรรม

1. เพื่อให้มีความรู้และความเข้าใจเกี่ยวกับโปรแกรมย่อยในภาษาซี
2. เพื่อให้อธิบายถึงประโยชน์ของโปรแกรมย่อยได้
3. เพื่อให้สามารถนำโปรแกรมย่อยไปใช้งานได้
4. เพื่อให้สามารถส่งผ่านค่าระหว่างโปรแกรมย่อยและโปรแกรมหลักได้

วิธีสอนและกิจกรรมการเรียนการสอน

1. สอนแบบบรรยาย นำเข้าสู่บทเรียนด้วยการกล่าวถึงการสั่งให้คอมพิวเตอร์ทำงานในลักษณะ ที่มีใช้กันในชีวิตประจำวัน โดยใช้และไม่ใช้โปรแกรมย่อย
2. กิจกรรมการเรียนการสอน
   2.1 บรรยาย และ แสดงรูปแบบในการใช้โปรแกรมย่อยในแบบต่างๆ
   2.2 ฝึกปฏิบัติ ด้วยการเขียนโปรแกรมตามกำหนด
   2.3 ทำแบบฝึกหัดท้ายบท

สื่อการเรียนการสอน

1. เอกสารประกอบการเรียนบทที่ 5
2. เพาเวอร์พอยท์ พรีเซนเตชั่น (power point presentation) ประกอบการบรรยาย
3. เครื่องคอมพิวเตอร์ที่สามารถใช้โปรแกรมภาษาซีได้

การวัดและการประเมินผล

1. สังเกตจากการตอบคำถาม และการตั้งคำถาม ในชั้นเรียน
2. วัดเจตคติจากการสังเกตพฤติกรรม ความกระตือรือร้นในการทำกิจกรรม
3. ความถูกต้องและคุณภาพของการทำแบบฝึกหัดท้ายบทเรียน

บทที่ 5
โปรแกรมย่อย

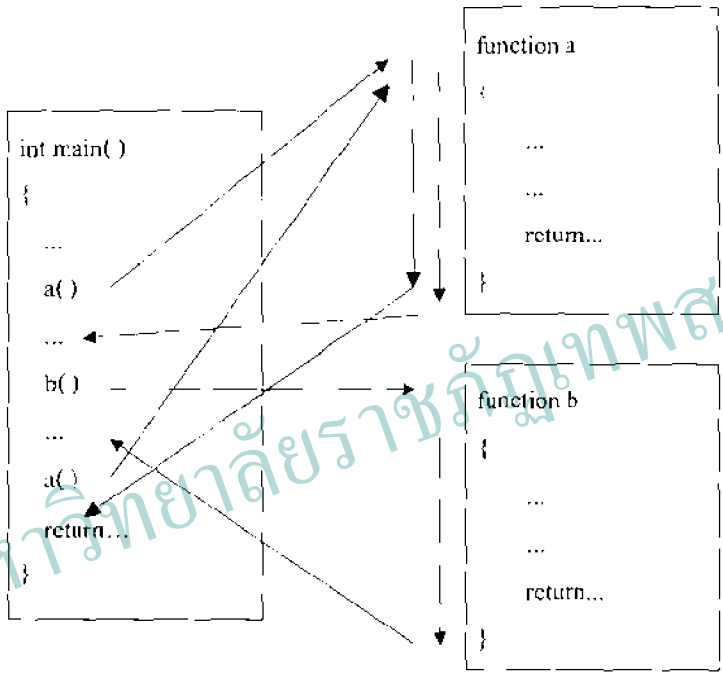
ในการเขียนโปรแกรมนั้นเราสามารถแบ่งส่วนของโปรแกรมออกเป็นส่วนย่อย ๆ ซึ่งโดยปกติการเขียนโปรแกรม ก็จะถูกแบ่งส่วนการทำงานออกเป็นส่วนได้แก่ ส่วนการรับข้อมูล ส่วนการประมวลผล และ ส่วนการแสดงผล ในแต่ละส่วนที่กล่าวถึง จะมีการทำงานตามจุดประสงค์ ของแต่ละส่วนนั้น ๆ ในบางครั้ง การทำงานของแต่ละส่วนยังแบ่งออกเป็นส่วนย่อย ๆ อีก เช่น ส่วนการแสดงผลข้อมูล อาจแบ่งเป็น การแสดงผลข้อมูลแบบทั้งหมด และ การแสดงผลแบบเฉพาะเจาะจงลงไป การเขียนโปรแกรม ที่มีขนาดใหญ่ และ มีส่วนปลีกย่อยจำนวนมาก ผู้เขียนโปรแกรม ควรแยกส่วนต่าง ๆ เหล่านั้น ออกให้ชัดเจน เพื่อประโยชน์ ในการติดตามแก้ไข และความเร็วในการทำงานของโปรแกรม การแบ่งโปรแกรมออกเป็น ส่วน ๆ นี้ เราเรียกว่า โปรแกรมย่อย ในแต่ละโปรแกรมย่อย ก็จะมีจุดประสงค์ไว้ชัดเจนว่า จะทำงานเพื่อให้เกิด อะไรขึ้นกับการประมวลผล ภาษาซี เป็นโปรแกรมที่มีลักษณะของโครงสร้าง เป็นแบบฟังก์ชัน จึงมีความสะดวกในการที่จะเขียนโปรแกรม ในแบบของ โปรแกรมย่อย เพื่อให้เกิดความสะดวกในการใช้งาน เพื่อประหยัดเวลาในการเขียนโปรแกรม ถ้าหากการประมวลผลนั้น ๆ จะเกิดขึ้นบ่อย ๆ ในบทนี้จะกล่าวถึง การใช้โปรแกรมย่อยในภาษาซี ประโยชน์ที่เกิดจากการใช้โปรแกรมย่อย และการนำโปรแกรมย่อยไปใช้งาน

ความหมายของโปรแกรมย่อย

โปรแกรมย่อย หมายถึง โปรแกรมที่ออกแบบมาเพื่อทำการประมวลผล อย่างใดอย่างหนึ่งตามความต้องการ ลักษณะของโปรแกรมจะเป็นโปรแกรมที่มีจุดจบแน่นอน โปรแกรมย่อยจะถูกเรียกให้ทำงานจากโปรแกรมหลัก (main program) ดังนั้นเมื่อการทำงานของโปรแกรมย่อยสิ้นสุดลง เครื่องจะย้ายการทำงานไปทำต่อจากจุดที่เรียก จึงเหมาะสำหรับการทำงาน หรือการประมวลผล ที่เกิดขึ้นซ้ำ ๆ บ่อย ๆ หรือลักษณะงานที่มีรูปแบบคงตัวเช่น การหาค่าเลขยกกำลัง หรือการหาค่าตัวเลขสูงสุด เป็นต้น การใช้โปรแกรมย่อยนั้น จะต้องมีการใช้ โปรแกรมเรียก (call program) และ โปรแกรมถูกเรียก หรือ เรียกว่า โปรแกรมย่อย (sub program) โปรแกรมย่อยในภาษาซี อยู่ในรูปแบบของ ฟังก์ชัน ดังนั้น ต่อไป จะเรียก โปรแกรมเรียก ว่า ฟังก์ชันหลัก ซึ่งโดยส่วนใหญ่ใน เอกสารเล่มนี้ จะหมายถึง ฟังก์ชัน main ส่วนโปรแกรมถูกเรียก จะเรียกว่า ฟังก์ชันย่อย

ธันวา ศรีประโมง (2539, หน้า 142) กล่าวว่า โปรแกรมในภาษาซี ถือได้ว่าภาษาประเภทโครงสร้างโมดูล เนื่องจากโปรแกรมในภาษาซี จะประกอบไปด้วยโมดูลต่าง ๆ หรือกล่าวอีกนัยหนึ่งก็คือ ประกอบด้วยโปรแกรมย่อยหลายตัวรวมอยู่ด้วยกัน มีโมดูลหลักชื่อ main

โปรแกรมย่อยในภาษาซี เรียกว่า ฟังก์ชัน (function) ฟังก์ชันย่อย ๆ เหล่านี้จะถูกเรียกใช้ภายในฟังก์ชันหลัก ที่ชื่อ main เพื่อให้มองเห็นได้ชัดยิ่งขึ้น ในเรื่องของ ฟังก์ชัน และ การเรียกใช้ฟังก์ชัน อธิบายได้ดังภาพที่ 5.1



ภาพที่ 5.1 การทำงานของโปรแกรมย่อยกับโปรแกรมหลัก

จากภาพที่ 5.1 การทำงานของโปรแกรมจะเริ่มจาก main ซึ่งเป็นโปรแกรมหลัก ภายในโปรแกรม โปรแกรมหลักนี้จะประกอบไปด้วย 2 โปรแกรมย่อย หรือ 2 ฟังก์ชัน ได้แก่ ฟังก์ชัน a และ ฟังก์ชัน b เมื่อการดำเนินการในโปรแกรมหลัก มาถึงคำสั่ง a() จะหมายถึง การเรียกใช้ฟังก์ชัน a การทำงานของโปรแกรมก็จะย้ายการทำงานมายังฟังก์ชัน a และดำเนินการจนเสร็จสิ้นแล้วย้ายการทำงานกลับไปที่ ฟังก์ชัน main เพื่อดำเนินการกับคำสั่งอื่น ๆ ต่อไป ข้อสังเกต ฟังก์ชัน a จะถูกเรียกใช้ 2 ครั้ง จากจุดนี้จะทำให้เห็นได้ว่า เราสามารถเขียนโปรแกรมได้สั้นลงกว่าเดิม

ประโยชน์ของโปรแกรมย่อย

1. ประหยัดเวลาในการทำงาน เพราะไม่ต้องมีการเขียนโปรแกรมสั่งงานซ้ำ
2. ทำให้โปรแกรมมีขนาดกะทัดรัด และ แก้ไขโปรแกรมได้ง่าย
3. สามารถนำไปใช้งานในโปรแกรมอื่นได้ในภายหลัง
4. สามารถเรียกใช้ฟังก์ชัน มาตรฐานที่มีการกำหนดไว้แล้ว
5. สามารถคอมไพล์ให้เป็น library ได้

โปรแกรมย่อยในภาษาซี

ภาษาซี มีโครงสร้างของโปรแกรม ที่อยู่ในลักษณะของฟังก์ชัน แม้กระทั่ง ฟังก์ชัน main ซึ่งเป็น ฟังก์ชันหลักของโปรแกรม ยังต้องมีการ ส่งคืนจากการใช้ ฟังก์ชัน สังเกตจาก จะต้องมีคำสั่ง return ปิดท้ายก่อนจบโปรแกรม ดังนั้นฟังก์ชันต่าง ๆ ที่ถูกกำหนดขึ้นมา นอกเหนือจากฟังก์ชัน main จะถือว่า เป็นโปรแกรมย่อย หรือ ฟังก์ชันย่อย ทั้งหมด โดยที่ฟังก์ชันที่ถูกกำหนดขึ้นมา จะมีจุดประสงค์ เพื่อทำงานตามที่ต้องการ เช่น ฟังก์ชันสำหรับการรับข้อมูล ฟังก์ชันสำหรับประมวลผลข้อมูล ในการใช้งานฟังก์ชันจะต้องมีการเรียกใช้ โดยใช้ชื่อ ฟังก์ชัน นั้น ๆ อาจจะมีการส่งค่าไปมาระหว่างฟังก์ชัน หรือ ไม่ ก็ได้ วิธีการส่งค่าไปมาระหว่าง ฟังก์ชัน จะกล่าวในหัวข้อต่อไป

ธันวา ศรีประโมง (2539, หน้า 144) กล่าวว่า โปรแกรมย่อยในภาษาซีนั้น เรียกกันว่า ฟังก์ชัน มีฟังก์ชันหลักชื่อ main ส่วนฟังก์ชันอื่นๆ ที่เกิดขึ้น จะแบ่งออกเป็น 2 ประเภทคือ

1. ฟังก์ชันมาตรฐาน (standard function) เป็นฟังก์ชันที่มีการกำหนดไว้แล้วในตัวคอมไพเลอร์ โดยกำหนดไว้ตาม ไลบรารีไฟล์ต่าง ๆ เช่น math.h string.h เมื่อผู้เขียนโปรแกรมต้องการใช้ฟังก์ชันใด ก็จะต้องทราบก่อนว่า ฟังก์ชันนั้น อยู่ในไลบรารี ใด แล้วใช้ include เรียกโมดูล เหล่านั้นเข้ามาร่วมแปลด้วย ฟังก์ชันมาตรฐานที่นิยมใช้กัน กล่าวไว้ตอนต้นแล้ว

2. ฟังก์ชันที่ผู้ใช้สร้างขึ้น (user-defined function) เป็นฟังก์ชัน ที่ผู้เขียนโปรแกรมกำหนดขึ้นมาใหม่ โดยอาจรวมไว้ในโปรแกรมเดียวกัน หรือแยกไว้ที่ไฟล์อื่น เพื่อใช้ include เรียกเข้ามาใช้งานก็ได้ ในบทนี้จะกล่าวถึงเฉพาะ การสร้างและการใช้งานฟังก์ชันที่ผู้ใช้สร้างขึ้น

ฟังก์ชันในภาษาซี มีตำแหน่งที่เขียน อยู่ 2 ตำแหน่งด้วยกันคือ ก่อนฟังก์ชัน main และหลังจากฟังก์ชัน main เพื่อให้เห็นภาพชัดขึ้น จะอธิบายตำแหน่งการเขียนฟังก์ชัน ในภาพที่ 5.2 และภาพที่ 5.3

```
#include <library file>
declaration part
function a
{
   ...
}
function b
{
   ...
}
.
.
.
main()
{
   ...
}
```


ภาพที่ 5.2 การเขียนฟังก์ชันแบบเขียนก่อนฟังก์ชันหลัก

การกำหนดฟังก์ชันในลักษณะนี้ จะกำหนด หลังจากส่วนหัวของโปรแกรม โครงสร้างของฟังก์ชัน จะมีลักษณะเหมือนโปรแกรมที่เคยเขียนทุกประการ ส่วนการเรียกใช้นั้นจะเรียกใช้จากฟังก์ชันหลัก หรือ ฟังก์ชันอื่น ๆ ก็ได้ ในกรณีที่มีการเรียกใช้จากฟังก์ชันอื่น ฟังก์ชันที่ถูกเรียกใช้จะต้องเขียนไว้ก่อนที่จะถูกเรียกใช้ เช่น ถ้า b เรียกใช้ a จะเขียนได้ตามตัวอย่าง แต่ถ้า a เรียกใช้ b จะต้องเปลี่ยนลำดับการเขียน โดยให้เขียนฟังก์ชัน b ก่อน ฟังก์ชัน a

```
#include <library file>
declaration part
function name a
function name b
main()
{
...
}
function a
{
...
}
function b
{
...
}
```

ภาพที่ 5.3 การเขียนฟังก์ชันแบบเขียนหลังฟังก์ชันหลัก

การเขียนฟังก์ชันในแบบนี้ เรียกว่าแบบ โปรโตไทป์ (prototype) หมายถึง การระบุชื่อและรายละเอียดในการใช้งาน ประกอบด้วย ชนิดของฟังก์ชัน ข้อมูลที่จะส่งไปมา โดยที่ยังไม่เขียนตัวของฟังก์ชัน แต่จะไปเขียนรายละเอียดของฟังก์ชันทั้งหมดไว้ที่ หลังจากจบ ฟังก์ชันหลัก การใช้งานก็จะเหมือนกัน คือเรียกใช้ได้ทั้งจากฟังก์ชันหลัก และฟังก์ชันอื่น ๆ และฟังก์ชันที่เรียกใช้ฟังก์ชันอื่น ต้องเขียนไว้หลัง ฟังก์ชันที่เรียกใช้

การส่งค่าผ่านพารามิเตอร์

การใช้งานฟังก์ชันนั้น โดยหลักการทั่วไปแล้ว ฟังก์ชันที่ถูกเรียกใช้นั้น จะต้องมีการส่งค่าย้อนกลับมาให้กลับโปรแกรมเรียก ค่าของข้อมูลที่มีการส่งค่าระหว่างโปรแกรมเรียก กับฟังก์ชันนั้น เรียกว่า ค่า พารามิเตอร์ (parameter) ซึ่งก็จะหมายถึง ค่าต่าง ๆ ที่มีการส่งไปมาระหว่าง การเรียกใช้ฟังก์ชัน การส่งค่าพารามิเตอร์ของฟังก์ชันในภาษาซีนั้น ส่งค่าได้ 3 แบบดังนี้

1. ฟังก์ชันที่ไม่มีการส่งค่ากลับ
2. ฟังก์ชันที่มีการส่งค่าค่ากลับ
3. ฟังก์ชันที่มีการส่งค่ากลับหลายค่า

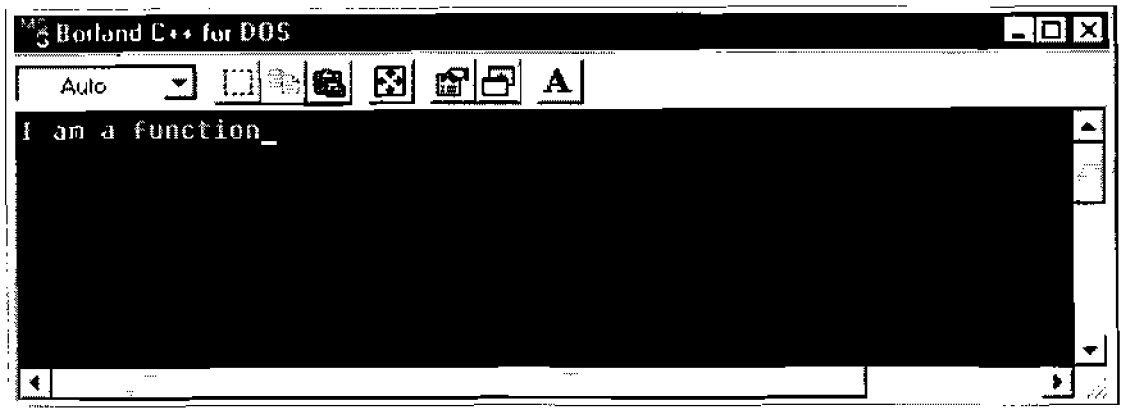
ฟังก์ชันที่ไม่มีการส่งค่ากลับ

ฟังก์ชันที่ไม่มีการส่งค่ากลับ หมายถึง การเรียกใช้ฟังก์ชัน โดยไม่มีการส่งค่าระหว่างโปรแกรมหลัก กับ โปรแกรมย่อย ดังนั้น ฟังก์ชันในลักษณะนี้ จึงไม่จำเป็นต้องระบุประเภทของข้อมูล ที่ใช้ ให้กับฟังก์ชัน ดังตัวอย่างต่อไปนี้

ตัวอย่างที่ 5.1 การเรียกใช้ฟังก์ชัน แบบไม่มีการส่งค่า

```
1   // void function example
2   #include <iostream.h>
3   #include <conio.h>
4   void dummyfunction(void)
5   {
6     cout << "I am a function";
7   }
8   int main()
9   {
10    dummyfunction();
11    getch();
12    return 0;
13  }
```

จากโปรแกรมตัวอย่าง มีผลลัพธ์ออกมาดังนี้



ภาพที่ 5.4 ผลลัพธ์ของโปรแกรมที่ไม่มีการส่งค่าพารามิเตอร์

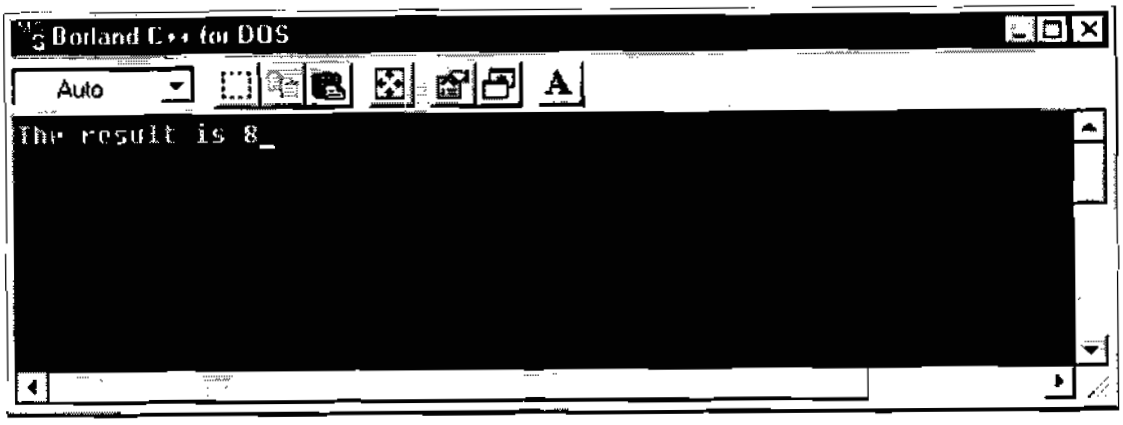
จากโปรแกรมตัวอย่าง กำหนดให้มีฟังก์ชันชื่อ dummyfunction ระบุประเภทของฟังก์ชัน เป็นข้อมูลชนิด void ซึ่งเป็นคำเฉพาะที่มีความหมายว่า ไม่มีการระบุชนิดของฟังก์ชัน เนื่องจาก ไม่มีการส่งค่าระหว่างฟังก์ชัน เมื่อโปรแกรมทำงาน จะเริ่มต้นที่ ฟังก์ชัน main แล้วเรียกใช้ dummyfunction ภายในฟังก์ชันที่ถูกเรียกใช้ จะแสดงข้อความว่า I am a function แล้วย้อนกลับมาที่ ฟังก์ชัน main โดยไม่มีการส่งค่ากลับมา ข้อสังเกตคือ ไม่มีคำสั่ง return ใน dummyfunction

ฟังก์ชันที่มีการส่งค่ากลับ

ฟังก์ชันที่มีการส่งค่ากลับ หมายถึง การเรียกใช้ฟังก์ชันที่มีการ ส่งค่ากลับ จาก ฟังก์ชัน ซึ่งเป็นเรื่องปกติ ของการเรียกใช้ฟังก์ชันโดยทั่วไป ซึ่งต้องมีการส่งค่า ผลลัพธ์ ที่ได้จากการทำงานของ ฟังก์ชัน ค่าผลลัพธ์ ที่ส่งกลับจากฟังก์ชันในลักษณะนี้ จะมีเพียง 1 ค่า ดังนั้นจึงต้องระบุชนิดของข้อมูลลงไปว่า ฟังก์ชันนั้น ๆ จะส่งผลลัพธ์ กลับคืน ในรูปแบบของข้อมูลชนิดใด ส่วนการส่งค่า ไปให้กับฟังก์ชัน นั้น อาจจะมีหรือไม่ก็ได้ อธิบายได้ด้วยตัวอย่างต่อไปนี้

ตัวอย่างที่ 5.2 การเรียกใช้ฟังก์ชัน แบบมีการส่งค่ากลับ แต่ไม่มีการส่งค่าไปจากโปรแกรมเรียก

```
1    #include <iostream.h>
2    #include <conio.h>
3    int func_a()
4    { int x=2,y=3,z=1,i;
5      for (i=1;i<=y;i++)
6        z = z * x;
7      return(z);
8    }
9    void main()
10   { int a;
11     clrscr();
12     a = func_a();
13     cout << "Result = " << a;
14     getch();
15   }
```

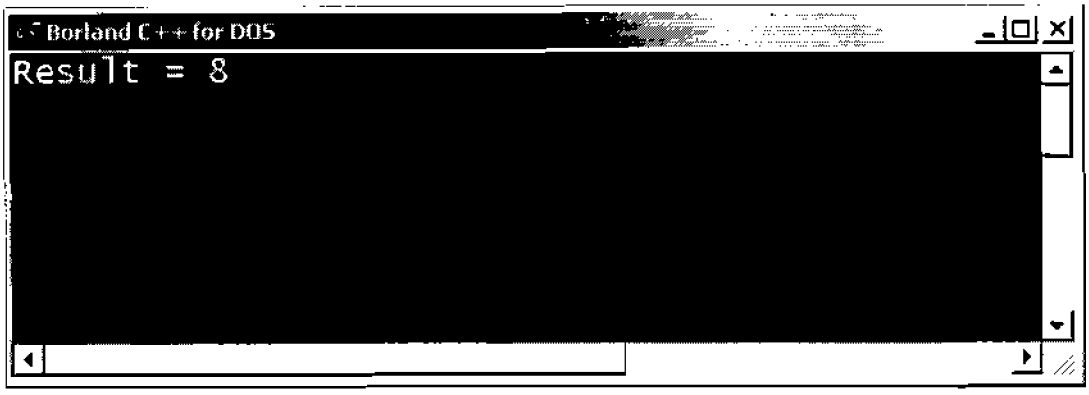


ภาพที่ 5.5 ผลลัพธ์การเรียกใช้ฟังก์ชันที่มีการส่งค่ากลับจากฟังก์ชัน

จากโปรแกรมตัวอย่าง มีฟังก์ชันชื่อ func a เป็นฟังก์ชัน ที่มีการส่งค่าผลลัพธ์คืนกลับจาก ฟังก์ชัน เป็นข้อมูลชนิด ตัวเลขจำนวนเต็ม ภายในตัวฟังก์ชัน มีการกำหนดตัวแปรขึ้นมาใช้งาน ได้แก่ตัวแปร x มีค่าเท่ากับ 2 ตัวแปร y มีค่าเท่ากับ 3 ตัวแปร z มีค่าเท่ากับ 1 และ ตัวแปร i เป็นตัวแปรที่ใช้ในการควบคุมวงรอบการทำงาน ของคำสั่ง for เมื่อถูกเรียกใช้ด้วยคำสั่ง a = func a; ฟังก์ชันก็จะถูกเรียกให้ทำงาน วงรอบของ i มีการทำงานทั้งหมด 3 รอบ จึงทำให้ค่า z มีค่าเท่ากับ 8 และส่งค่า z กลับไปยัง โปรแกรมเรียก จึงมีผลลัพธ์ เท่ากับ 8

ตัวอย่างที่ 5.3 การเรียกใช้ฟังก์ชัน แบบมีการส่งค่ากลับ และมีการส่งค่าไปจากโปรแกรมเรียก ในการส่งค่าไปจากโปรแกรมเรียกอาจจะมีการส่งค่า มากกว่า 1 ค่า แต่มีการคืนค่ากลับจากฟังก์ชัน เพียง 1 ค่า

```
1   #include <iostream.h>
2   #include <conio.h>
3   int func_b(int x,int y)
4   { int z=1,i;
5     for (i=1;i<=y;i++)
6       z = z * x;
7     return(z);
8   }
9   void main()
10  { int a,b=2,c=3;
11    clrscr();
12    a = func_b(b,c);
13    cout << "Result = " << a;
14    getch();
15  }
```



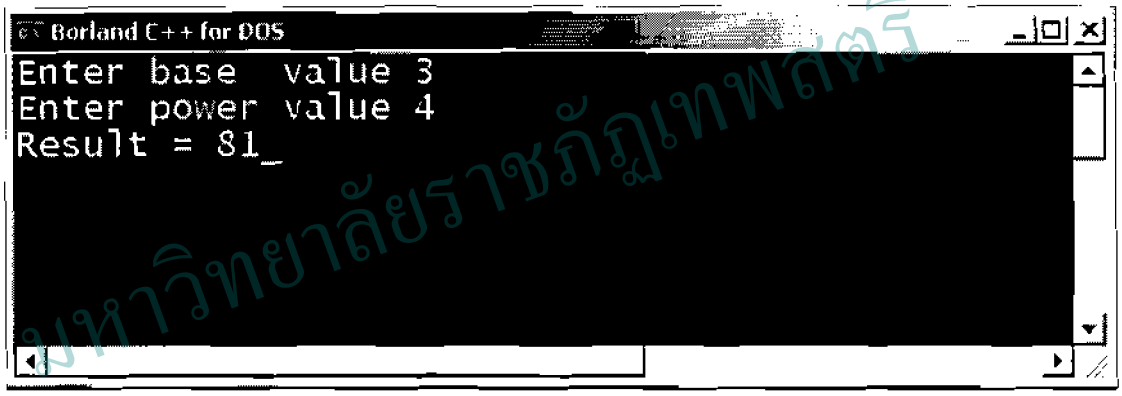
ภาพที่ 5.6 ผลลัพธ์การเรียกใช้ฟังก์ชันที่มีการส่งค่าไป และกลับจากฟังก์ชัน

จากตัวอย่างที่ 5.3 เป็นการเรียกใช้ฟังก์ชัน แบบที่มีการส่งค่าไปให้กับ ฟังก์ชัน หรือที่เรียกว่า มีการส่งค่า พารามิเตอร์ไปยังฟังก์ชัน นั้น ๆ ด้วย พารามิเตอร์ที่ส่งไป ได้แก่ค่า b และ c สังเกตจากคำสั่ง เรียกใช้ได้แก่ func_b(b,c) โดยที่ b มีค่าเท่ากับ 2 และ c มีค่าเท่ากับ 3 เมื่อมีการส่งค่าไปถึงฟังก์ชัน ก็จะมีการกำหนดให้มีการรับค่า ในที่นี้ func_b กำหนดให้ x และ y มารับค่าจาก b และ c ตามลำดับ ดังนั้น ค่า x จะมีค่าเท่ากับ 2 และค่า y จะมีค่าเท่ากับ 3 การทำงานของฟังก์ชันเหมือนกับตัวอย่าง ที่ 5.2 จึงทำให้ z มีค่าเท่ากับ 8 และสุดท้ายจึงมีผลลัพธ์ออกมาเท่ากับ 8

ตัวอย่างที่ 5.4 การเรียกใช้ฟังก์ชัน แบบมีการส่งค่ากลับ และมีการส่งค่าไปจากโปรแกรมเรียก ในการส่งค่าไปจากโปรแกรมเรียกอาจจะมีการส่งค่า มากกว่า 1 ค่า แต่มีการคืนค่ากลับจากฟังก์ชัน เพียง 1 ค่า ในการส่งค่าไปยังฟังก์ชัน ผู้ใช้สามารถกำหนดค่าลงไปได้ ว่าต้องการส่งค่าอะไรไปจากโปรแกรมเรียก โดยใช้คำสั่งในการรับค่าจากแป้นพิมพ์

```
1   #include <iostream.h>
2   #include <conio.h>
3   int func_b(int x,int y)
4   { int z=1,i;
5     for (i=1;i<=y;i++)
6        z = z * x;
7     return(z);
8   }
```

```
9   void main()
10  { int a,b,c;
11    clrscr();
12    cout << "Enter base  value ";cin >> b;
13    cout << "Enter power value ";cin >> c;
14    a = func_b(b,c);
15    cout << "Result = " << a;
16    getch();
17  }
```



ภาพที่ 5.7 ผลลัพธ์จากฟังก์ชันที่มีการส่งค่าไปและกลับจากฟังก์ชันโดยผู้ใช้กำหนด

จากตัวอย่างที่ 5.6 มีการเรียกใช้ฟังก์ชันโดยกำหนดค่าพารามิเตอร์ ที่ส่งไปโดยผู้ใช้โปรแกรม สามารถป้อนข้อมูล ให้กับฟังก์ชันได้ ฟังก์ชันที่ใช้เป็นตัวอย่าง ใช้สำหรับการหาค่าจำนวนยกกำลัง ของตัวเลข จากตัวอย่าง ผู้ใช้ป้อนข้อมูล 3 และ 4 ซึ่งหมายถึง ให้หาค่า 3 ยกกำลัง 4 ซึ่งได้ผลลัพธ์ เท่ากับ 81

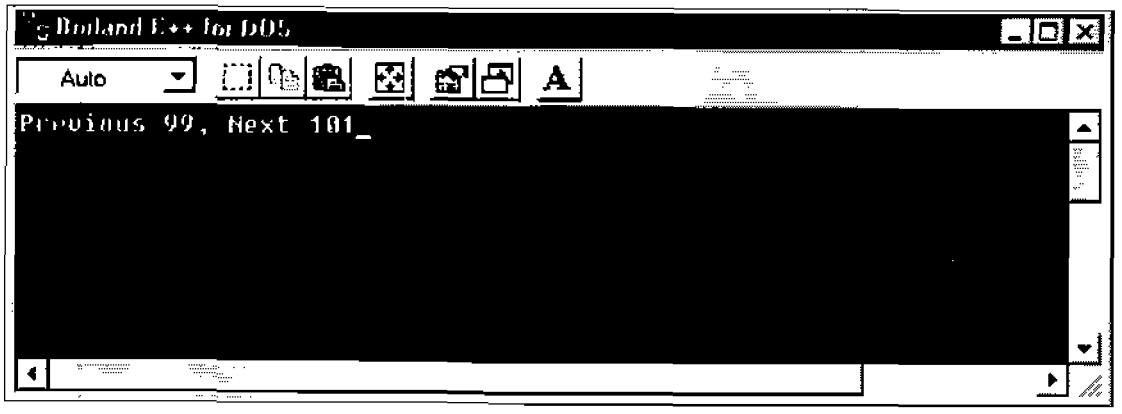
ฟังก์ชันที่มีการส่งค่าไปกลับหลายค่า

ฟังก์ชันที่มีการส่งค่าไปกลับหลายค่า หมายถึง การเรียกใช้โปรแกรมย่อยที่มีการส่งค่าไปหาโปรแกรมย่อย และ มีการส่งค่ากลับ มาหาโปรแกรมเรียกโดยค่าที่ส่งกลับมานั้น สามารถส่งค่าได้มากกว่า 1 ค่า หรือ หลายค่าได้ ซึ่งตามปกติแล้ว ค่าที่ส่งคืนจากฟังก์ชันมีได้เพียงค่าเดียว

สำหรับการเรียกใช้ฟังก์ชันในลักษณะนี้ค่าที่ส่งไปนั้นอาจมีการเปลี่ยนแปลงค่าไปจากเดิม และ การเลือกใช้การส่งค่าในลักษณะนี้ ส่วนใหญ่แล้วมีจุดประสงค์ เพื่อต้องการให้มีการเปลี่ยนแปลงค่าที่โปรแกรมย่อย แล้วส่งค่าใหม่คืนมาที่โปรแกรมหลักอธิบายได้ด้วยตัวอย่างต่อไปนี้

```
1   // more than one return value
2   #include <iostream.h>
3   #include <conio.h>
4   void prevnext(int x,int& prev,int& next)
5   {
6     prev = x - 1;
7     next = x + 1;
8   }
9   int main()
10  {
11    int x = 100,y,z;
12    prevnext(x,y,z);
13    cout << "Previous "<<y<<", Next "<<z;getch();
14    return 0;
15  }
```

จากโปรแกรมตัวอย่าง มีผลลัพธ์ออกมาดังภาพที่ 5.5



ภาพที่ 5.8 ผลลัพธ์ของโปรแกรมที่มีการส่งค่ากลับหลายค่า

จากโปรแกรมตัวอย่าง การเรียกใช้โปรแกรมย่อยในโปรแกรมนี้ เรียกใช้จากคำสั่ง prevnext (x,y,z) หมายถึงเรียกใช้โปรแกรมย่อยชื่อ prevnext และให้มี พารามิเตอร์ 3 ตัว ได้แก่ x y และ z ในการเรียกใช้นั้น x มีค่าอยู่ที่ 100 เมื่อไปถึง โปรแกรมย่อย ตัวแปร prev จะมีค่าเท่ากับ x – 1 และ ตัวแปร next จะมีค่าเท่ากับ x + 1 ในการส่งค่ากลับ จะมีการส่งค่ากลับ 2 ค่าได้แก่ prev และ next โปรดสังเกตรูปแบบการส่งค่ากลับที่ส่วนหัวของโปรแกรมย่อยคือ int& prev และ int& next หมายถึง ให้ส่งค่ากลับไปที่โปรแกรมหลัก ด้วย 2 ค่านี้

ตัวอย่างที่ 5.1 จงเขียนโปรแกรมเพื่อให้เกิดการทำงาน เหมือนกับการใช้เครื่องคิดเลข กำหนดให้ผู้ใช้ป้อนข้อมูล ดังนี้ ป้อนตัวเลขค่าที่ 1 ป้อนเครื่องหมาย และ ป้อนตัวเลขค่าที่ 2 แล้วให้เกิดการคำนวณ ตามเครื่องหมายที่ป้อน

วิธีทำ

กำหนดให้มีการแสดงผลลัพธ์ดังนี้

Result of … … … = …

กำหนดให้มีการรับข้อมูลดังนี้

ตัวเลขค่าที่ 1 ตัวเลขค่าที่ 2 และ เครื่องหมายที่จะคำนวณ

กำหนดให้มีการประมวลผลดังนี้

คำนวณหาผลลัพธ์ โดยพิจารณาจากเครื่องหมายที่ป้อน

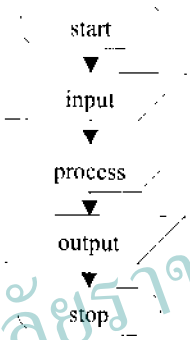
กำหนดตัวแปรที่จะใช้ในโปรแกรมหลัก

ชื่อตัวแปร	ความหมาย	ประเภท
d1	ค่าตัวเลขที่ 1	int
d2	ค่าตัวเลขที่ 2	int
sign	เครื่องหมาย	char
result	ผลลัพธ์	int

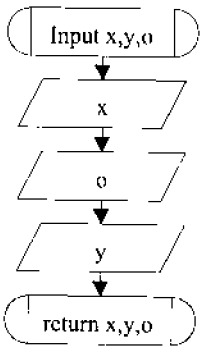
กำหนดตัวแปรที่จะใช้ในโปรแกรมย่อย

ชื่อตัวแปร	ความหมาย	ประเภท
x	ค่าตัวเลขที่ 1	int
y	ค่าตัวเลขที่ 2	int
o	เครื่องหมาย	char
r	ผลลัพธ์	int

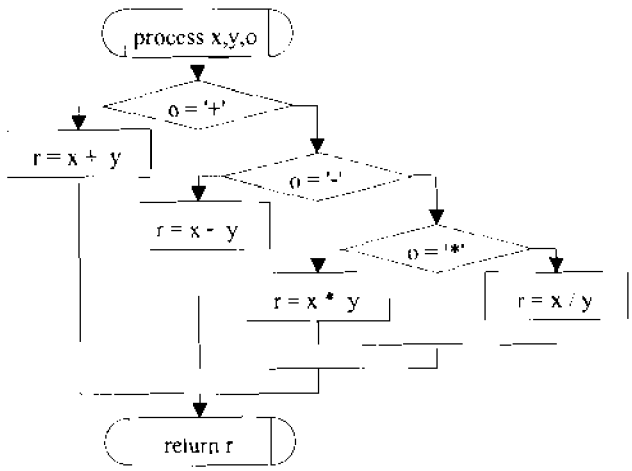
ผังงานการทำงานของแต่ละโปรแกรมมีดังนี้



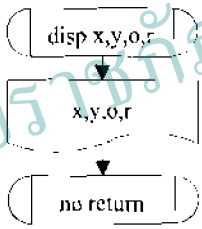
ภาพที่ 5.9 ผังงานของโปรแกรมหลัก



ภาพที่ 5.10 ผังงานของโปรแกรมย่อยสำหรับรับข้อมูล



ภาพที่ 5.11 ผังงานของโปรแกรมย่อยสำหรับประมวลผล



ภาพที่ 5.12 ผังงานของโปรแกรมย่อยสำหรับแสดงผล

จากผังงานที่นำเสนอ ลงรหัสเป็นโปรแกรมในภาษาซีได้ดังนี้

```
1   #include <iostream.h>
2   #include <conio.h>
3   void input(int& x,int& y,char& o)
4   {
5     clrscr();
6     cout<< "Enter First Value : ";cin >> x;
7     cout<< "Enter Operator    : ";cin >> o;
8     cout<< "Enter Second Value: ";cin >> y;
```

```
9   }
10  int process(int x,int y,char o)
11  { int r;
12    if (o=='+')
13      r = x + y;
14    else if (o=='-')
15            r = x - y;
16    else if (o=='*')
17            r = x * y;
18          else r = x / y;
19    return r;
20  }
21  void disp(int x,int y,char o ,int r)
22  {
23   cout<<"Result of "<<x<<" "<<o<<" "<<y<<" = "<<r<<"\n";
24  }
25  void main()
26  {
27    int d1,d2,result;
28    char sign;
29    input(d1,d2,sign);
30    result = process(d1,d2,sign);
31    disp(d1,d2,sign,result);
32    cout << "End Program";getch();
33  }
```

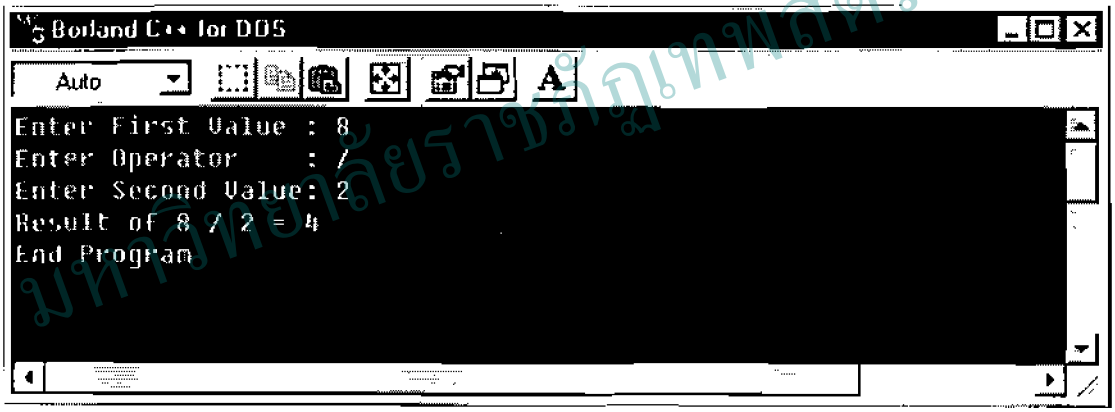

จากโปรแกรม จะเห็นได้ว่า โปรแกรมนี้ แบ่งส่วนการทำงานออกเป็นโปรแกรมย่อยไว้ 3 โปรแกรมด้วยกัน ได้แก่

1. input ทำหน้าที่ในการ รับข้อมูลเข้ามาจากการที่ผู้ใช้ป้อน อยู่ในช่วงบรรทัดที่ 3 ถึง บรรทัดที่ 9 เป็นลักษณะการใช้โปรแกรมย่อยแบบมีการส่งค่าไปมาระหว่างโปรแกรมเรียกกับโปรแกรมย่อยแบบ ส่งค่ากลับหลายค่า ในที่นี้ได้แก่ d1 d2 และ sign

2. process ทำหน้าที่ ประมวลผลตามเครื่องหมายที่ได้รับ อยู่ในช่วงบรรทัดที่ 10 ถึง บรรทัดที่ 20 เป็นลักษณะการใช้โปรแกรมย่อยแบบ ส่งค่ากลับเพียง 1 ค่า โดยค่าที่ส่งกลับมาได้แก่ค่า r ส่งกลับมาให้กับ result

3. disp ทำหน้าที่แสดงผลลัพธ์ อยู่ในช่วงบรรทัดที่ 21 ถึง บรรทัดที่ 24 เป็นลักษณะการใช้โปรแกรมย่อยแบบ ส่งค่าไปทางเดียว ไม่มีการส่งค่ากลับมายังโปรแกรมเรียก ในที่นี้ได้แก่ d1 d2 sign และ result

โปรแกรมหลัก จะเริ่มตั้งแต่ บรรทัดที่ 25 ถึงบรรทัดที่ 33 มีการทำงานตามลำดับดังนี้คือ รับข้อมูลด้วยฟังก์ชัน input ประมวลผลด้วยฟังก์ชัน process และ แสดงผลด้วยฟังก์ชัน disp เมื่อทำการสั่งให้โปรแกรมทำงาน และทำการป้อนข้อมูลจะมีผลลัพธ์ออกมาดังภาพที่ 5.10



ภาพที่ 5.13 ผลลัพธ์ของโปรแกรมที่มีการเรียกใช้โปรแกรมย่อย

ดังกล่าวมาแล้วว่า ในการเขียนโปรแกรมย่อยในภาษาซีนั้น สามารถเขียนได้ในลักษณะของโปรโตไทป์ จากโปรแกรมตัวอย่างเพื่อให้เห็นข้อแตกต่างของการเขียนโปรแกรมย่อยแบบโปรโตไทป์ จึงเขียนโปรแกรมในลักษณะของโปรโตไทป์ได้ดังนี้

```
1   #include <iostream.h>
2   #include <conio.h>
3   void input(int& x,int& y,char& o);
4   int process(int x,int y,char o);
```

```
5   void disp(int x,int y,char o ,int r);
6   void main()
7   {
8     int d1,d2,result;
9     char sign;
10    input(d1,d2,sign);
11    result = process(d1,d2,sign);
12    disp(d1,d2,sign,result);
13    cout << "End Program";getch();
14  }
15  void input(int& x,int& y,char& o)
16  {
17    clrscr();
18    cout<< "Enter First Value : ";cin >> x;
19    cout<< "Enter Operator    : ";cin >> o;
20    cout<< "Enter Second Value: ";cin >> y;
21  }
22  int process(int x,int y,char o)
23  { int r;
24    if (o=='+')
25      r = x + y;
26    else if (o=='-')
27          r = x - y;
28        else if (o=='*')
29              R = x * y;
30            else r = x / y;
31    return r;
```

```
32  }
33
34  void disp(int x,int y,char o ,int r)
35  {
36    cout<<"Result of "<<x<<" "<<o<<" "<<y<<" = "<<r<<"\n";
37  }
```

จากโปรแกรมส่วนที่แตกต่างกันคือ มีการกำหนดชื่อของโปรแกรมย่อยขึ้นมาก่อน และมีการระบุด้วยว่า โปรแกรมย่อยแต่ละโปรแกรม มีการส่งค่า พารามิเตอร์อย่างไรบ้าง ในโปรแกรมอยู่ระหว่าง บรรทัดที่ 3 ถึงบรรทัดที่ 5 จากนั้นเขียนโปรแกรมหลัก แล้วจึงมาเขียนโปรแกรมย่อยในตอนหลัง ลักษณะการทำงานและผลลัพธ์จะเหมือนกันกับโปรแกรมที่แล้ว

สรุป

การเขียนโปรแกรมกับกิจกรรมที่เกิดขึ้น ซ้ำ ๆ บ่อย ๆ เราสามารถ ออกแบบโปรแกรมขึ้นมาเป็นโปรแกรมย่อยได้ เพื่อให้มีการเรียกใช้โปรแกรมนั้นบ่อยๆ จะทำให้เกิดความสะดวกในการใช้โปรแกรม และการเขียนโปรแกรม โปรแกรมย่อยในภาษาซี เรียกว่า ฟังก์ชัน แบ่งออกเป็น 2 ประเภท ได้แก่ ฟังก์ชันมาตรฐาน หมายถึงฟังก์ชันที่เขียนไว้แล้วในตัวคอมไพลเลอร์ของภาษา โดยเขียนไว้ใน ไลบรารี่ต่าง ๆ ของภาษาซี ซึ่งหมายถึงไฟล์ที่มีสกุลเป็น .h ต่าง ๆ ฟังก์ชันอีกชนิดหนึ่งเป็นฟังก์ชันที่ผู้เขียนโปรแกรม เขียนขึ้นมาใช้งานเอง มีลักษณะการเขียนอยู่ 2 แบบคือ เขียนไว้ก่อนที่จะถึงตัวโปรแกรมหลัก หรือ ฟังก์ชันเมน กับอีกวิธีหนึ่ง เขียนไว้หลังจากฟังก์ชันเมน เรียกว่า การเขียนฟังก์ชันแบบ โปรโตไทป์ฟังก์ชันในภาษาซี แบ่งตามลักษณะของการใช้งาน แบ่งออกได้เป็น 3 ประเภท ได้แก่ ฟังก์ชันที่ไม่มีการส่งค่าระหว่างโปรแกรมหลักกับฟังก์ชัน ฟังก์ชันประเภทนี้จะไม่มีการส่งค่าไปมาขณะที่ถูกเรียกใช้งาน จึงกำหนดให้เป็นฟังก์ชันประเภท void ซึ่งเป็นคำเฉพาะของภาษาซี ซึ่ง หมายถึง ไม่ระบุชนิดของข้อมูล อีกประเภทหนึ่งเรียกว่า ฟังก์ชันที่มีการส่งค่ากลับเพียง 1 ค่า ฟังก์ชันประเภทนี้เราสามารส่งผ่านค่าไปมาระหว่างการเรียกใช้งานได้โดย ในการเรียกใช้งานอาจส่งค่ามายังโปรแกรมย่อยได้มากกว่า 1 จำนวน แต่หลังจากฟังก์ชันนั้นสิ้นสุดลงจะคืนค่ากลับไปยังโปรแกรมเรียกเพียง 1 ค่า และประเภทสุดท้ายของฟังก์ชันได้แก่ ฟังก์ชันที่มีการส่งค่าคืนกลับไปยังโปรแกรมหลักได้หลายค่าซึ่งก็จะหมายถึง เมื่อสิ้นสุดการทำงานของฟังก์ชันแล้วจะมีการส่งค่า ต่าง ๆ กลับไปยังโปรแกรมหลักได้มากกว่า 1 ค่านั่นเอง

คำถามทบทวน

จงตอบคำถามต่อไปนี้โดยสังเขป

1. โปรแกรมย่อยในภาษาซีเรียกว่าอะไร มีลักษณะการทำงานเป็นอย่างไร
2. จงบอกประโยชน์ของโปรแกรมย่อย
3. ฟังก์ชันในภาษาซีแบ่งเป็นกี่ประเภท อะไรบ้าง
4. ตำแหน่งการเขียนฟังก์ชันในภาษาซี มีอยู่ด้วยกันกี่แบบ อะไรบ้าง
5. การส่งค่าพารามิเตอร์ทำได้กี่แบบอะไรบ้าง

เอกสารอ้างอิง

เกษมสันต์ พานิชการ. (2537). **C++ และหลักการของ OOP ฉบับเริ่มต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

ทักษิณา สวนานนท์. (2544). **พจนานุกรมศัพท์คอมพิวเตอร์ ฉบับนิสิตนักศึกษา.** กรุงเทพฯ: ไอบริด พริ้นติ้ง.

ธันวา ศรีประโมง. (2539). **การเขียนโปรแกรมภาษาซีสำหรับวิศวกรรม.** กรุงเทพฯ: มหาวิทยาลัยเทคโนโลยีมหานคร.

เบญจพร ศักดิ์ศิริ. (2540). **ทฤษฎีและตัวอย่างโจทย์ การเขียนโปรแกรมด้วยภาษา C++.** กรุงเทพฯ: แมคกรอ-ฮิล อินเตอร์เนชั่นแนล เอ็นเตอร์ไพร์ส, อิงค์.

มนตรี พจนารถลาวัณย์. (2535). **การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี.** กรุงเทพฯ: เอช-เอน การพิมพ์.

วรรณวิภา จำเริญดารารัศมี. (2535). **วิทยาการคอมพิวเตอร์เบื้องต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

ศิริรัตน์ ชำนาญรบ, เกื้อกูล ตาเย็น, และวัชระ โพธิสรณ์. (2539). **ความรู้เบื้องต้นเกี่ยวกับคอมพิวเตอร์.** กรุงเทพฯ: ภูมิบัณฑิต.

แผนบริหารการสอนประจำบทที่ 6

เนื้อหาประจำบท

โครงสร้างข้อมูลชนิดแถวลำดับ

โครงสร้างข้อมูลชนิดเรคคอร์ด

โครงสร้างข้อมูลชนิดแฟ้มข้อมูล

สรุป

คำถามทบทวน

จุดประสงค์เชิงพฤติกรรม

1. เพื่อให้ผู้เรียนมีความรู้และความเข้าใจการใช้ข้อมูลชนิดอาร์เรย์
2. เพื่อให้ผู้เรียนอธิบายและเขียนคำสั่งในการใช้งานข้อมูลชนิดโครงสร้างได้
3. เพื่อให้ผู้เรียนอธิบายและเขียนคำสั่งในการใช้งานข้อมูลชนิดแฟ้มข้อมูลได้

วิธีสอนและกิจกรรมการเรียนการสอน

1. สอนแบบบรรยาย นำเข้าสู่บทเรียนด้วยการกล่าวถึงการสั่งให้คอมพิวเตอร์ทำงานในลักษณะ ที่มีใช้กันในชีวิตประจำวันที่มีการใช้เรียกใช้ข้อมูลเดิมที่เก็บไว้ในเครื่องคอมพิวเตอร์ เช่น ข้อมูลการฝากเงินไว้กับธนาคาร ข้อมูลราคาสินค้าในห้างสรรพสินค้า
2. กิจกรรมการเรียนการสอน
   2.1 บรรยาย และแสดงรูปแบบในการใช้คำสั่งเพื่อเก็บข้อมูลในลักษณะอาร์เรย์ ลักษณะโครงสร้าง และ แฟ้มข้อมูล
   2.2 ฝึกปฏิบัติ ด้วยการเขียนโปรแกรมตามกำหนด
   2.3 ทำแบบฝึกหัดท้ายบท

สื่อการเรียนการสอน

1. เอกสารประกอบการเรียนบทที่ 6
2. เพาเวอร์พอยท์ พรีเซนเตชั่น (power point presentation) ประกอบการบรรยาย
3. เครื่องคอมพิวเตอร์ที่สามารถใช้โปรแกรมภาษาซีได้

การวัดและการประเมินผล

1. สังเกตจากการตอบคำถาม และการตั้งคำถาม ในชั้นเรียน
2. วัดเจตคติจากการสังเกตพฤติกรรม ความกระตือรือร้นในการทำกิจกรรม
3. ความถูกต้องและคุณภาพของการทำแบบฝึกหัดท้ายบทเรียน

บทที่ 6
ข้อมูลแบบโครงสร้าง

ข้อมูลในภาษาซีที่กล่าวผ่านมาแล้วนั้น เป็นข้อมูลที่มีการจัดเก็บในหน่วยความจำแบบมีที่เก็บข้อมูลโดยใช้ 1 ชื่อ กับข้อมูล 1 ตัว เมื่อต้องการเก็บข้อมูล หลาย ๆ ตัวก็ต้องตั้งชื่อให้กับข้อมูลหลาย ๆ ชื่อ ตามที่ต้องการเก็บ ในการใช้งานจริง เราอาจไม่สะดวกในการ จัดการกับข้อมูล ที่มีมากมาย และ หลากหลายชื่อเหล่านั้น แต่งานที่เรามีความจำเป็น ต้องทำการประมวลผล เช่น การเรียงลำดับข้อมูล จำเป็นต้องนำเอาข้อมูลทุกตัวมาทำการเปรียบเทียบกัน จึงต้องเก็บค่าของข้อมูลที่จะทำการจัดเรียงนั้น ไว้ในหน่วยความจำในขณะนั้นทั้งหมด การตั้งชื่อ และ การจัดการกับข้อมูลดังกล่าว จึงเป็นปัญหา ที่ไม่สามารถแก้ได้ ถ้าหากไม่ใช้ การเก็บข้อมูลในลักษณะ โครงสร้างเข้ามาช่วย ข้อมูลในลักษณะโครงสร้างนี้ จะทำการจัดระเบียบของข้อมูล ให้สามารถกำหนด และ จัดการได้ง่ายขึ้น เพราะเราสามารถที่จะจัดการให้ข้อมูลเหล่านั้น อยู่ในรูปแบบของกลุ่มข้อมูล ที่ใช้ชื่อเดียวกัน แต่มีข้อแตกต่าง กันตรงที่ลำดับของข้อมูล จึง ทำให้ง่าย และ สะดวกต่อการจัดการ ดังนั้นในบทเรียนนี้ จึงจะกล่าวถึง ข้อมูลชนิดโครงสร้าง ที่มีใช้ในภาษาซี ซึ่งในที่นี้จะกล่าวถึง โครงสร้างข้อมูล 3 ประเภทดังนี้

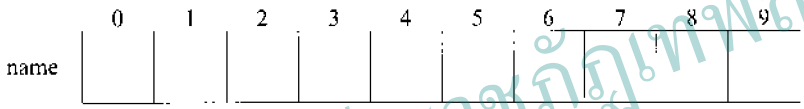
1. โครงสร้างข้อมูล ชนิดแถวลำดับ
2. โครงสร้างข้อมูล ชนิดเรคคอร์ด
3. โครงสร้างข้อมูล ชนิดแฟ้มข้อมูล

โครงสร้างข้อมูลชนิดแถวลำดับ

แถวลำดับ (array) คือ กลุ่มของข้อมูลที่เป็นข้อมูลชนิดเดียวกัน จัดเก็บในลักษณะการเรียงลำดับ ในหน่วยความจำหลักของ คอมพิวเตอร์ โดยใช้ชื่อตัวแปรเหมือนกัน แต่มีตัวชี้ (index) ซึ่งใช้ตัวเลขบอกลำดับ ที่แตกต่างกัน เราจึงสามารถกำหนดชื่อตัวแปรที่เหมือนกันได้ทั้งกลุ่ม หรือทั้งรายการ และ ถ้าต้องการใช้ข้อมูล ตัวไหน ก็ระบุได้ด้วย ตัวชี้ข้อมูล สำหรับตัวชี้ข้อมูล จะต้องเป็นข้อมูลชนิดตัวเลขจำนวนเต็ม และสามารถใช้ตัวแปร ในการอ้างอิงถึง ตัวชี้ข้อมูลได้ การเก็บข้อมูลด้วยโครงสร้างแถวลำดับ สามารถเก็บได้ หลายแบบ ซึ่งจะเรียกว่า แต่ละแบบว่า เป็นการเก็บในลักษณะ กี่มิติ ในที่นี้ จะกล่าวถึง การเก็บข้อมูลด้วยโครงสร้างแถวลำดับ 3 แบบดังนี้

1. โครงสร้างแถวลำดับ แบบ 1 มิติ
2. โครงสร้างแถวลำดับ แบบ 2 มิติ
3. โครงสร้างแถวลำดับ แบบ หลายมิติ

แถวลำดับ แบบ 1 มิติ หมายถึง การเก็บข้อมูลในลักษณะของแถวข้อมูล 1 แถว หรือ 1 มิติ ภายในแถวมีลำดับที่บอกว่า ข้อมูลเก็บอยู่ในช่อง หรือ คอมโพเนนท์ ใด ในการเก็บข้อมูลต้องทำการระบุชื่อตัวแปรและตำแหน่งที่จะนำไปเก็บ หมายเลขลำดับ จะเริ่มต้นที่ ลำดับที่ 0 ไปจนถึงลำดับสุดท้าย ที่กำหนด ดังนั้นการ กำหนดจำนวนช่องที่จะใช้สำหรับเก็บข้อมูลที่ต้องการ จะต้องกำหนดจาก จำนวนสูงสุดของข้อมูลที่จะจัดเก็บ เช่น ข้อมูลคะแนนนักเรียน 50 คน ก็จะต้องกำหนดให้ แถวลำดับมีขนาด 50 ช่อง การเก็บข้อมูลในแถวลำดับ 1 มิติ แสดงให้เห็นวิธีการเก็บได้ดังภาพที่ 6.1



ภาพที่ 6.1 โครงสร้างการเก็บข้อมูลชนิดแถวลำดับแบบ 1 มิติ

จากภาพที่ 6.1 เป็นโครงสร้างการเก็บข้อมูลชนิดแถวลำดับ 1 มิติ ขนาด 10 ช่องคือ ใช้ชื่อตัวแปร 1 ชื่อ และมีตัวชี้ข้อมูล ตั้งแต่ลำดับที่ 0 จนถึงลำดับที่ 9 การเก็บข้อมูล หรือ การนำเอาข้อมูลออกมาทำการแสดงผล จะต้องระบุชื่อตัวแปร และ หมายเลขลำดับของข้อมูล ที่ระบุด้วย ตัวชี้ของแถวลำดับ การกำหนดตัวแปรชนิดแถวลำดับ มีรูปแบบดังนี้

```
type variable_name[component];
```

type	หมายถึง การระบุชนิดของข้อมูลที่จะจัดเก็บ
variable_name	หมายถึง การกำหนดชื่อตัวแปร
component	หมายถึง จำนวนช่องที่ใช้เก็บข้อมูล

ตัวอย่าง การกำหนดตัวแปรแบบแถวลำดับ 1 มิติ

int a[20]; หมายถึง กำหนดให้ ตัวแปร a เป็นตัวแปรแบบแถวลำดับ 1 มิติ มีขนาด 20 ช่อง แต่ละช่องกำหนดให้เก็บข้อมูลแบบตัวเลขจำนวนเต็มเริ่มจากช่องที่ 0 จนถึงช่องที่ 19

float b[30]; หมายถึง กำหนดให้ตัวแปร b เป็นตัวแปรแบบแถวลำดับ 1 มิติ มีขนาด 30 ช่อง แต่ละช่องกำหนดให้เก็บข้อมูลแบบ ตัวเลขจำนวนจริงเริ่มจากช่องที่ 0 จนถึงช่องที่ 29

char c[50]; หมายถึง กำหนดให้ตัวแปร c เป็นตัวแปรแบบแถวลำดับ 1 มิติ มีขนาด 50 ช่อง แต่ละช่องกำหนดให้เก็บข้อมูลแบบ ตัวอักษร เริ่มจากช่องที่ 0 จนถึงช่องที่ 49

ตัวอย่างที่ 6.1 จงเขียนโปรแกรมเพื่อรับข้อมูลตัวเลขเข้ามาจำนวนหนึ่ง โดยให้ผู้ใช้โปรแกรมเป็นผู้ป้อนข้อมูลตัวเลขเหล่านั้น จนกว่าผู้ใช้โปรแกรม ป้อน –1 จึงหยุดการป้อนข้อมูล เสร็จแล้วทำการประมวลผลข้อมูล ให้ได้ ค่าสูงสุด ค่าต่ำสุด และค่าเฉลี่ยของข้อมูลที่ป้อนลงไป แล้วทำการแสดงผลลัพธ์ตามที่กำหนดดังนี้

Data Report

No. Item

---

1 ...

2 ...

3 ...

... ...

Maximum = ...

Minimum = ...

Average = ...

จากโจทย์ตัวอย่าง มีวิธีทำดังนี้

1. กำหนดให้มีการรับข้อมูลดังนี้

   ข้อมูลตัวเลข

2. กำหนดให้มีการประมวลผลดังนี้

   2.1 ผลรวมของตัวเลข

   2.2 จำนวนตัวเลขที่ป้อนเข้าไป

2.3 ค่าสูงสุดของตัวเลขที่ป้อน

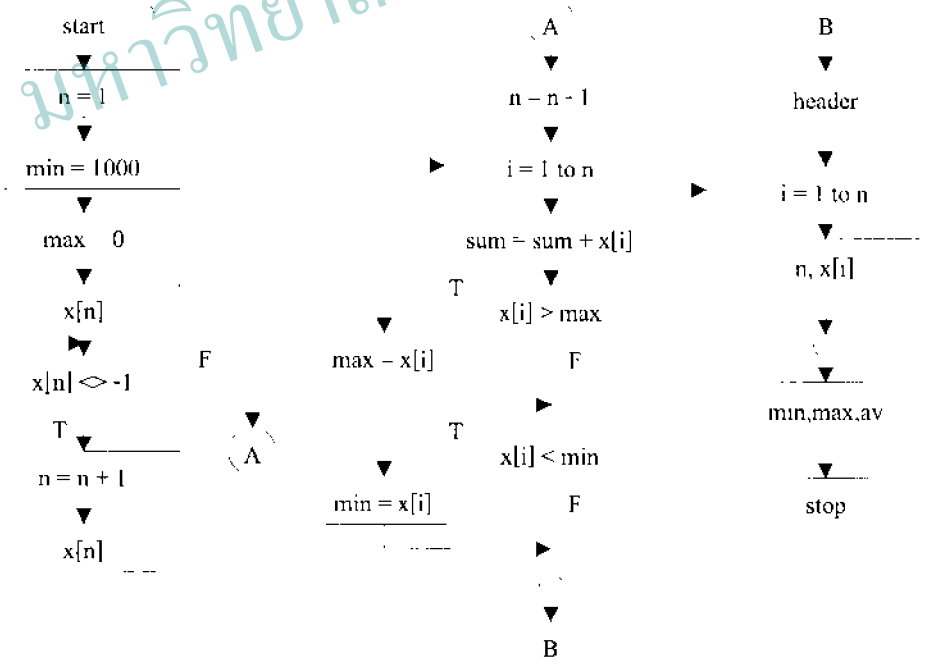
2.4 ค่าต่ำสุดของตัวเลขที่ป้อน

2.5 ค่าเฉลี่ยของตัวเลขที่ป้อน

3. กำหนดให้มีตัวแปรดังนี้

ชื่อตัวแปร	ประเภท	ความหมาย
x	array แบบ real	ค่าของตัวเลขที่ผู้ใช้ป้อนเข้ามา
n	integer	จำนวนตัวเลขที่ป้อนเข้ามา
sum	real	ผลรวมทั้งหมด
max	real	ค่าสูงสุด
min	real	ค่าต่ำสุด
av	real	ค่าเฉลี่ย

4. กำหนดขั้นตอนดำเนินการของโปรแกรมด้วยผังงานดังภาพที่ 6.3



ภาพที่ 6.2 ผังงานแสดงขั้นตอนการทำงานของโปรแกรม

5. ลงรหัสเป็นภาษาซีดังนี้

```
1   #include <iostream.h>
2   #include <conio.h>
3   int i,n;
4   float sum,av,min,max,x[100];
5   void main()
6   { n = 1;
7     min = 1000;
8     max = 0;
9     clrscr();
10    cout << "Enter Item No. " << n << " : ";
11    cin >> x[n];
12    while (x[n] != -1)
13    { n = n + 1;
14      cout << "Enter Item No. " << n << " : ";
15      cin >> x[n];
16    }
17    n = n - 1;
18    for (i=1;i<=n;i++)
19    { sum = sum + x[i];
20      if (x[i]>max) max = x[i];
21      if (x[i]<min) min = x[i];
22    }
23    av = sum / n * 1.00;
24    clrscr();
25    cout<<"   Data Report \n";
26    cout<<"------------------------ \n";
```

```
27    cout<<"\t No    Item \n";
28    cout<<"------------------------- \n";
29     for(i=1;i<=n;i++) cout <<"\t"<< i << "\t" << x[i] << "\n";
30    cout<<"------------------------- \n";
31    cout << "  Maximum = " << max << "\n";
32    cout << "  Minimum = " << min << "\n";
33    cout << "  Average = " << av  << "\n";
34    cout<<"------------------------- \n";
35    cout << "End Program";getch();
36  }
```

เมื่อสั่งให้โปรแกรมทำงานพร้อมป้อนข้อมูล จะมีผลลัพธ์ออกมาดังภาพที่ 6.3

ภาพที่ 6.3 ผลลัพธ์ของโปรแกรมการหาค่าต่าง ๆ ของตัวเลข

จากโปรแกรมตัวอย่าง ทำการกำหนดค่าตัวแปรชนิด แถวลำดับ แบบ 1 มิติ ไว้ที่ตัวแปร x จำนวน 100 ช่อง วงรอบที่บรรทัดที่ 9 จะทำเมื่อ x ในลำดับปัจจุบัน ไม่ใช่ – 1 และรับข้อมูลสำหรับ x ตัวต่อไป ในบรรทัดที่ 11 จนกระทั่ง ผู้ใช้ ใส่ค่า – 1 ลงไป หมายถึงสิ้นสุดการรับข้อมูล จากนั้นจะทำการประมวลผลข้อมูล เพื่อหาค่าผลรวม ค่าสูงสุด และค่าต่ำสุด ระหว่างบรรทัดที่ 13 – 16 ในบรรทัดที่ 17 เป็นการหาค่าเฉลี่ย จากนั้นจึงทำการแสดงผลตั้งแต่บรรทัดที่ 18 – 27

แถวลำดับแบบ 2 มิติ หมายถึง การเก็บข้อมูลในลักษณะตารางที่ แถว และ คอลัมน์มีความสัมพันธ์กัน การระบุตำแหน่งของข้อมูลในโครงสร้างชนิดนี้ จะต้องระบุด้วยตัวเลขทั้ง 2 แนว คือ แถว และ คอลัมน์ เช่น แถวที่ 0 คอลัมน์ที่ 0 เก็บข้อมูล อะไร หรือ แถวที่ 2 คอลัมน์ที่ 4 เก็บข้อมูลอะไร ตัวชี้ข้อมูลต้องจะเริ่มที่ตำแหน่งที่ 0 ทั้ง 2 แนว แสดงให้เห็นวิธีการเก็บได้ดัง ภาพที่ 6.4

	0	1	2	3	4	5
0						
1						
2						
3						

ภาพที่ 6.4 พื้นที่เก็บข้อมูลชนิดแถวลำดับแบบ 2 มิติ

จากภาพที่ 6.4 เป็นภาพโครงสร้างการเก็บข้อมูลแบบแถวลำดับ 2 มิติ ขนาด 4 x 6 หรือ 4 แถว 6 คอลัมน์ มีที่เก็บข้อมูลตั้งแต่ตำแหน่งที่ 0,0 หมายถึง แถวที่ 1 คอลัมน์ที่ 1 จนถึง ตำแหน่งที่ 3,5 หมายถึง แถวที่ 4 คอลัมน์ที่ 6 รวมแล้วมีที่สำหรับเก็บข้อมูลจำนวน 24 ตำแหน่ง หรือเท่ากับ 4 คูณ 6 ข้อมูลที่จะนำมาเก็บในโครงสร้างชนิดนี้ ควรเป็นข้อมูลที่สอดคล้องกับการเก็บในแบบตารางได้ เช่น คะแนนสอบ 5 วิชา ของนักเรียน 7 คน จะต้องออกแบบให้มีแถวในการเก็บข้อมูล 7 แถว เพื่อใช้เก็บจำนวนของนักเรียน และต้องกำหนดให้มีคอลัมน์ 5 คอลัมน์ สำหรับเก็บข้อมูลคะแนนสอบแต่ละวิชา แสดงการเก็บดังภาพที่ 6.5

	0	1	2	3	4
0					
1					
2					
3					
4					
5					
6					

ภาพที่ 6.5 การเก็บข้อมูลคะแนนของนักเรียนจำนวน 7 คน คนละ 5 วิชา

จากภาพที่ 6.5 มีพื้นที่ในการเก็บข้อมูลทั้งหมด เท่ากับ 7 คูณ 5 เท่ากับ 35 ช่อง ข้อมูลทางแถวหมายถึง ลำดับที่ของนักเรียน ส่วน ข้อมูลทางคอลัมน์ หมายถึง คะแนนในแต่ละวิชาโดย ช่องที่ 0,0 เก็บคะแนนนักเรียนคนที่ 1 ในรายวิชาที่ 1 ช่องที่ 0,1 เก็บคะแนนนักเรียนคนที่ 1 ในรายวิชาที่ 2 และ ช่องสุดท้าย คือช่องที่ 6,4 สำหรับเก็บคะแนน นักเรียนคนที่ 7 ในรายวิชาที่ 5

การกำหนดตัวแปรแบบ แถวลำดับ 2 มิติ มีรูปแบบดังนี้

type variable_name[x component][y component];

type	หมายถึง	การระบุชนิดของข้อมูลที่จะจัดเก็บ
variable_name	หมายถึง	การกำหนดชื่อตัวแปร
x component	หมายถึง	จำนวนช่องที่ใช้เก็บข้อมูลทางแกน x
y component	หมายถึง	จำนวนช่องที่ใช้เก็บข้อมูลทางแกน y

ตัวอย่างการกำหนดตัวแปรแบบแถวลำดับ 2 มิติ

int m[3][5]; หมายถึง กำหนดให้ m เป็นตัวแปรแบบแถวลำดับ 2 มิติที่เก็บข้อมูลทางแถว หรือ แกน x ได้ 3 แถว และเก็บข้อมูลทางคอลัมน์หรือ แกน y ได้ 5 คอลัมน์ สามารถเก็บข้อมูลได้ทั้งหมด 15 ตัว โดยที่แต่ละตัว เป็นข้อมูลชนิด ตัวเลขจำนวนเต็ม

ต้องการเก็บข้อมูลการเงิน ของพนักงานในบริษัทแห่งหนึ่ง โดยพนักงานคนหนึ่ง ๆ จะมีข้อมูลที่เป็น ค่าจ้างรายวัน ค่าจ้างในการทำงานล่วงเวลา และ ข้อมูลการจ่ายค่าประกันสังคม จัดเก็บโดยโครงสร้างแถวลำดับ 2 มิติ ดังนี้

```
float income [50][3];
```

การกำหนดตัวแปร income จะได้ที่สำหรับเก็บข้อมูลประเภทตัวเลขจำนวนจริงทั้งหมด 50 แถว แถวละ 3 คอลัมน์ รวมเก็บข้อมูลได้ทั้งหมด 150 ตัว โดยที่แต่ละแถวจะหมายถึงข้อมูลพนักงาน 1 คน ส่วนทางคอลัมน์ คอลัมน์ที่ 1 หมายถึงข้อมูลค่าจ้างรายวัน คอลัมน์ที่ 2 หมายถึงข้อมูลค่าจ้างในการทำงานล่วงเวลา และ คอลัมน์ที่ 3 หมายถึงข้อมูลในการจ่ายค่าประกันสังคม

ตัวอย่างที่ 6.3 จงเขียนโปรแกรมเพื่อรับข้อมูลคะแนนนักเรียนจำนวน 6 คน คนละ 5 วิชา โดยให้ผู้ใช้โปรแกรมเป็นผู้ป้อนข้อมูลค่าคะแนนเหล่านั้น จนครบ 30 รายการ แล้วทำการประมวลผลข้อมูล ให้ได้ค่าสูงสุดและ ค่าเฉลี่ย ของข้อมูลคะแนนที่ป้อนลงไป แล้วทำการแสดงผลลัพธ์ตามที่กำหนดดังนี้


```
                Student Report
-------------------------------------------------------------
  Student No.    Sbj1    Sbj2    Sbj3    Sbj4    Sbj5
-------------------------------------------------------------
      1          ...     ...     ...     ...     ...
      2          ...     ...     ...     ...     ...
      3          ...     ...     ...     ...     ...
      4          ...     ...     ...     ...     ...
      5          ...     ...     ...     ...     ...
      6          ...     ...     ...     ...     ...
-------------------------------------------------------------
Maximum =        ...              Average   =     ...
-------------------------------------------------------------
```

จากโจทย์ตัวอย่าง มีวิธีทำดังนี้

1. กำหนดให้มีการรับข้อมูลดังนี้

   ข้อมูลคะแนนนักเรียนจำนวน 6 คน คนละ 5 วิชา

2. กำหนดให้มีการประมวลผลดังนี้

   2.1 ผลรวมของคะแนนทั้งหมด

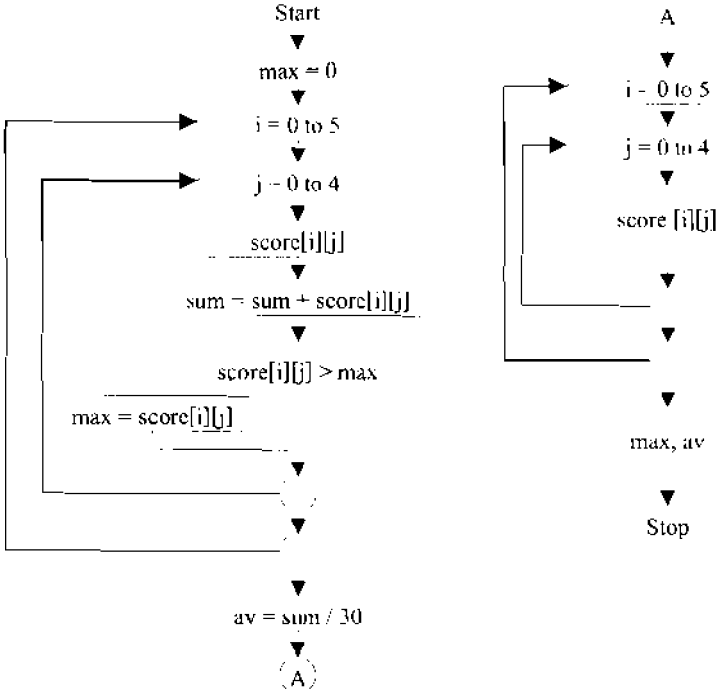
   2.2 ค่าสูงสุดของคะแนน

   2.4 ค่าเฉลี่ยของคะแนน

3. กำหนดให้มีตัวแปรดังนี้

ชื่อตัวแปร	ประเภท	ความหมาย
score	array 2 มิติขนาด 6 คูณ 5 แบบ integer	ค่าของคะแนน
sum	real	ผลรวมทั้งหมด
max	real	ค่าสูงสุด
av	real	ค่าเฉลี่ย
i	integer	สำหรับควบคุมจำนวนนักเรียน
j	integer	สำหรับควบคุมจำนวนวิชา

4. กำหนดขั้นตอนดำเนินการของโปรแกรมด้วยผังงานดังภาพที่ 6.6



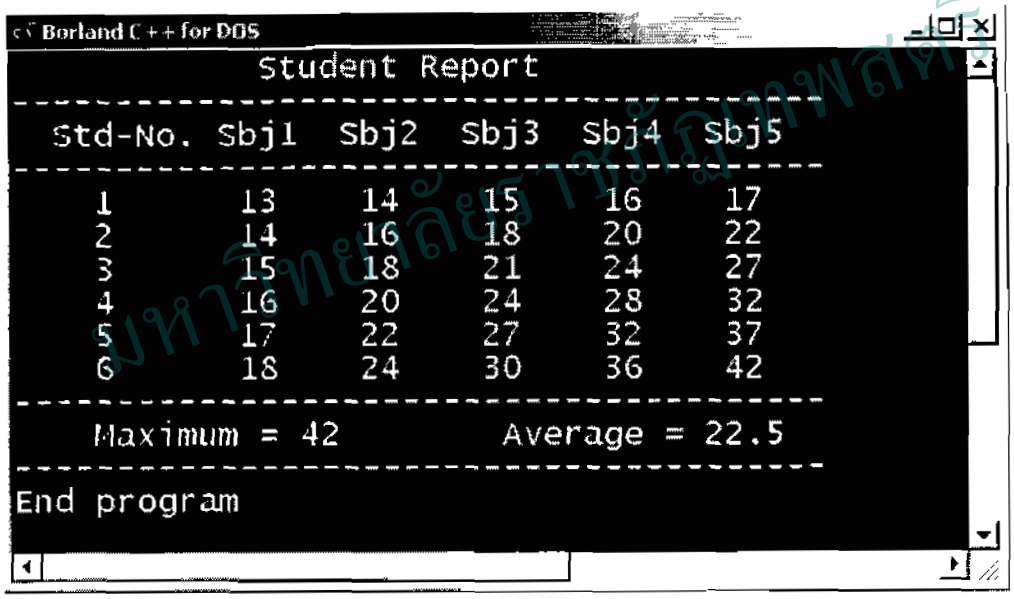
ภาพที่ 6.6 ผังงานแสดงขั้นตอนการทำงานของโปรแกรมเก็บข้อมูลคะแนน

5. ลงรหัสเป็นภาษาซีดังนี้

```
1   #include <iostream.h>
2   #include <conio.h>
3   int main()
4   { int score[6][5],i,j;
5     int max = 0,sum=0;
6     float av=0;
7       clrscr();
8       for(i=0;i<=5;i++)
9       { cout <<"Student No ." << i<<"\n";
10        for(j=0;j<=4;j++)
11        { cout <<"Subject No ." << j<<"\t";
12          cin >> score[i][j];
13          sum = sum + score[i][j];
14          if (score[i][j] > max) max = score[i][j];
15        }
16    }
17      av = sum / 30.00;
18      clrscr();
19      cout << "\t    Student Report \n";
20      cout << "------------------------------------ \n";
21      cout << "  Student No. Sj1  Sj2  Sj3  Sj4  Sj5 \n";
22      cout << "------------------------------------ \n";
23      for(i=0;i<=5;i++)
24      { cout << "\t" << i+1<<"\t";
25        for(j=0;j<=4;j++)
26        { cout << score[i][j]<<"    "; }
```

```
27          cout << "\n";
28        }
29        cout << "---------------------------------------- \n";
30        cout << "  Maximum = "<<max<<"\t"<<"Average = "<<av<<"\n";
31        cout << "---------------------------------------- \n";
32        cout <<"End program "; getch();
33        return 0;
34    }
```

เมื่อสั่งให้โปรแกรมทำงานพร้อมป้อนข้อมูล จะมีผลลัพธ์ออกมาดังภาพที่ 6.4



ภาพที่ 6.7 ผลลัพธ์ของโปรแกรมคะแนนนักเรียน

จากโปรแกรมตัวอย่าง กำหนดให้ตัวแปร score เป็นตัวแปรประเภท แถวลำดับ 2 มิติ ขนาด 6 คูณ 5 สำหรับเก็บข้อมูลคะแนนสอบของนักเรียน จำนวน 6 คน คนละ 5 วิชาโดย กำหนดให้ ข้อมูลทางด้านแถว เป็นลำดับที่ของนักเรียน ส่วนทางคอลัมน์ เป็นคะแนนในแต่ละ วิชา ทำการ รับข้อมูลจากผู้ใช้ที่ บรรทัดที่ 12 หาคะแนนรวมที่ บรรทัดที่ 13 และหาค่าคะแนน สูงสุดที่บรรทัดที่ 14 หลังจากรับข้อมูลจนครบ 6 คนแล้ว ทำการคำนวณหาค่าเฉลี่ยของคะแนน ที่ บรรทัดที่ 17 จากนั้นทำการแสดงผลตั้งแต่ บรรทัดที่ 19 จนถึง บรรทัดที่ 31

แถวลำดับ หลายมิติ หมายถึงการเก็บข้อมูลด้วยโครงสร้าง แถวลำดับ ที่มากกว่า 2 มิติ ในที่นี้ จะกล่าวถึง แถวลำดับ แบบ 3 มิติ ที่สามารเก็บข้อมูลได้ทั้ง แถว หรือ แกน x คอลัมน์ หรือ แกน y และ ทางด้านแนวลึก หรือ แกน z ข้อมูลที่จัดเก็บด้วยโครงสร้างชนิดนี้ เช่น การเก็บข้อมูลคะแนน นักเรียน 5 คน แต่ละคนเรียน 3 วิชา แต่ละวิชา มีคะแนนสอบ 2 ครั้ง

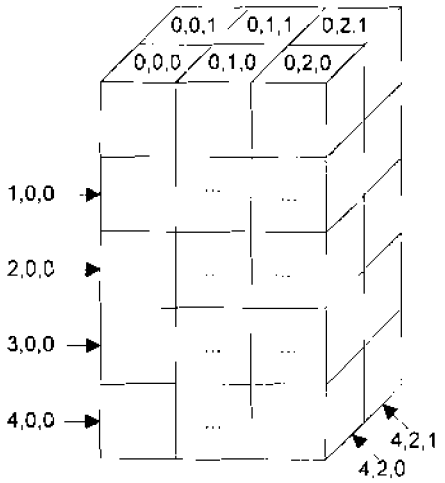
การกำหนดตัวแปรแบบแถวลำดับ 3 มิติ มีรูปแบบดังนี้

type variable_name[x component][y component][z component];

type	หมายถึง	การระบุชนิดของข้อมูลที่จะจัดเก็บ
variable_name	หมายถึง	การกำหนดชื่อตัวแปร
x component	หมายถึง	จำนวนช่องที่ใช้เก็บข้อมูลทางแกน x
y component	หมายถึง	จำนวนช่องที่ใช้เก็บข้อมูลทางแกน y
z component	หมายถึง	จำนวนช่องที่ใช้เก็บข้อมูลทางแกน z

ตัวอย่างการกำหนดตัวแปรแบบ แถวลำดับ 3 มิติ

int item [5][3][2]; หมายถึง การกำหนดให้ตัวแปร item เป็นตัวแปรที่มีความสามารถในการเก็บข้อมูล ได้แบบ 3 มิติ ได้แก่ 1 มิติทางด้านแถว 2 มิติทางด้านคอลัมน์ และ 3 มิติทางแนวลึก หรืออาจเรียกแต่ละมิติตามแนวแกนของกราฟ ได้ว่า มิติที่ 1 แกน x มิติที่ 2 แกน y มิติ ที่ 3 แกน z เช่น ข้อมูลสินค้าที่มีสินค้าจำนวน 5 ชนิด สินค้าแต่ละชนิด มี 3 ขนาด แต่ละขนาดมีราคาทุน และ ราคาขาย อธิบายได้ดังภาพที่ 6.8



ภาพที่ 6.8 การเก็บข้อมูลด้วยแถวลำดับ 3 มิติขนาด 5 คูณ 3 คูณ 2

จากภาพที่ 6.8 จะมีพื้นที่ในการเก็บข้อมูลทั้งหมด 30 ที่ ประกอบด้วย 5 แถวในแต่ละแถวจะมีสมาชิก 3 คอลัมน์ ในแต่ละคอลัมน์ จะมีสมาชิกอีก 2 ตัว โดยที่สมาชิกของแถวลำดับ ตัวที่ 0,0,0 หมายถึง สินค้าตัวที่ 1 ขนาดที่ 1 ราคาที่ 1 หมายถึงราคาทุน และ สมาชิกของแถวลำดับเตัวที่ 4,2,1 หมายถึง สินค้าตัวที่ 5 ขนาดที่ 2 ราคาที่ 2 หมายถึงราคาขาย

โครงสร้างข้อมูลชนิดเรคคอร์ด

ข้อมูลชนิดเรคคอร์ด (record) หมายถึง ข้อมูลที่ผู้ใช้กำหนดขึ้นมาใช้งานเอง ลักษณะเฉพาะของ ข้อมูลประเภทนี้คือ มีการรวบรวมเอาข้อมูลหลายประเภทเข้าไว้ด้วยกันได้อย่างมีระบบระเบียบ มีความสัมพันธ์ไปในทิศทางเดียวกัน เช่น ข้อมูลเกี่ยวกับสินค้า ซึ่งอาจได้แก่ รหัสสินค้า ชื่อสินค้า ขนาด และราคา หรือข้อมูลเกี่ยวกับพนักงาน อาจได้แก่ เลขประจำตัว ชื่อ สกุล แผนก และเงินเดือน เป็นต้น ข้อมูลชนิดเรคคอร์ดในภาษาซี มีรูปแบบในการกำหนดดังนี้

```
struct
{   type field-1;
    type field-2;
    type field-3;
    ...
} record-name;
```

field หมายถึง ตัวแปรต่างๆ ที่ประกอบกันขึ้นมาเป็น เรคคอร์ด

record-name หมายถึง ตัวแปรที่กำหนดให้เป็นข้อมูลประเภทเรคคอร์ด

ตัวอย่างการกำหนดข้อมูลประเภทเรคคอร์ดให้กับสินค้า ซึ่งมีรายละเอียดของข้อมูลในการจัดเก็บคือ รหัสสินค้า ชื่อ ขนาด และ ราคา กำหนดได้ดังนี้

```
   struct
{  char id[4];
   char product[20];
   char size;
   int price;
}  prorec;
```

การกำหนดให้เป็นข้อมูลชนิดเรคคอร์ดดังตัวอย่าง มีความหมายว่า prorec สามารถเก็บข้อมูลได้ทั้งหมด 4 รายการ ได้แก่ id product size และ price โดยที่แต่ละฟิลด์จะมีความสัมพันธ์ไปในทางเดียวกันคือ เป็นเรื่องเกี่ยวกับสินค้าตัวหนึ่ง การนำเอาข้อมูลชนิดนี้ไปใช้ มีรูปแบบในการใช้ดังนี้

record-name.field-name

record-name	หมายถึง	ชื่อเรคคอร์ดที่กำหนด
field-name	หมายถึง	ชื่อฟิลด์ที่กำหนด

จากตัวอย่างเรื่องของสินค้า สามารถนำเอาฟิลด์ต่างๆ ไปใช้ได้ดังนี้

cin >> prorec.product; หมายถึง ให้ผู้ใช้ป้อนข้อมูลเข้ามาเก็บที่ ฟิลด์ที่ชื่อ product ของเรคคอร์ดชื่อ prorec

cout << prorec.id หมายถึง ให้แสดงค่าของ ฟิลด์ id ของจากเรคคอร์ดที่ชื่อ prorec

prorec.price = prorec.price * 120/100; หมายถึง กำหนดให้ ฟิลด์ price ของเรคคอร์ดที่ชื่อ prorec มีค่าที่เกิดจากการนำเอา prorec.price * 120/100

ตัวอย่างที่ 6.3 จงเขียนโปรแกรมเพื่อเรียงลำดับข้อมูลพนักงานตามเงินเดือนที่ได้รับจากมากไปหาน้อย ของพนักงานในบริษัทแห่งหนึ่ง ซึ่งมีข้อมูลดังนี้

เลขประจำตัว	ชื่อ	แผนก	เงินเดือน
101	Mayura	A	15000
102	Narinee	B	30000
103	Panawan	B	20000
104	Ratana	A	40000
105	Vatinee	A	35000

1. กำหนดให้มีการแสดงผลลัพธ์ดังนี้ (output)

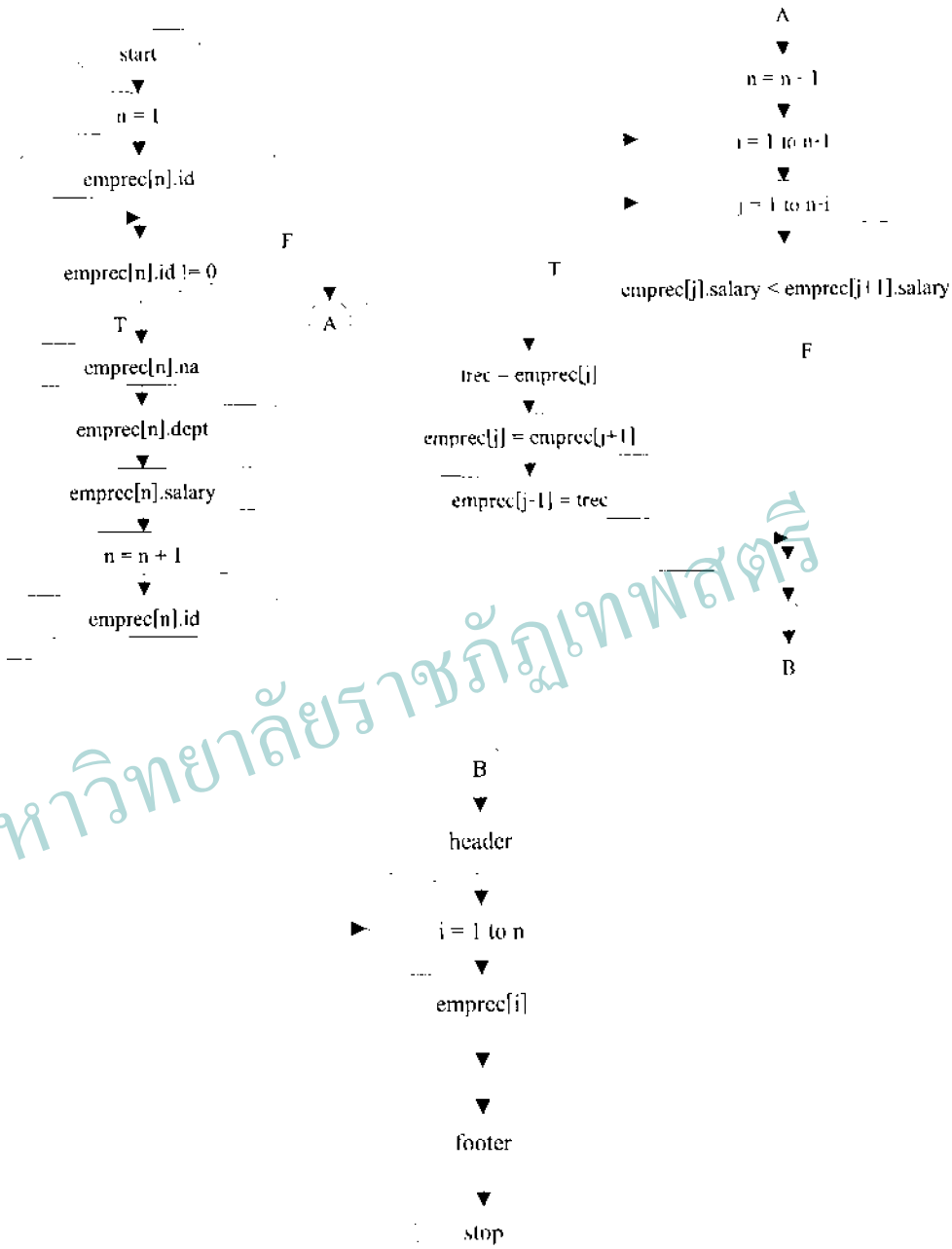
Employee Report

Code	Name	Dept.	Salary
999	xxxxxx	x	99999
999	xxxxxx	x	99999
...	...	...	...

2. กำหนดให้มีการรับข้อมูลดังนี้ (input)
   2.1 เลขประจำตัว
   2.2 ชื่อ
   2.3 แผนก
   2.4 เงินเดือน
3. กำหนดให้มีการประมวลผลดังนี้ (process)
   เรียงลำดับข้อมูล โดยใช้ เงินเดือนเป็นหลักในการจัดเรียงแบบมากไปหาน้อย
4. กำหนดให้มีตัวแปรดังนี้ (variable)

ชื่อตัวแปร	ประเภท	ความหมาย
id	char จำนวน 5 ตัว	ฟิลด์เลขประจำตัวพนักงาน
na	char จำนวน 20 ตัว	ฟิลด์ชื่อพนักงาน
dept	char	ฟิลด์แผนก
salary	long	ฟิลด์เงินเดือน
emprec	struct	เรคคอร์ดของพนักงาน
i และ j	int	ใช้ควบคุมวงรอบ
n	int	ใช้นับจำนวนข้อมูล

5. กำหนดขั้นตอนดำเนินการของโปรแกรมด้วยผังงานดังภาพที่ 6.9

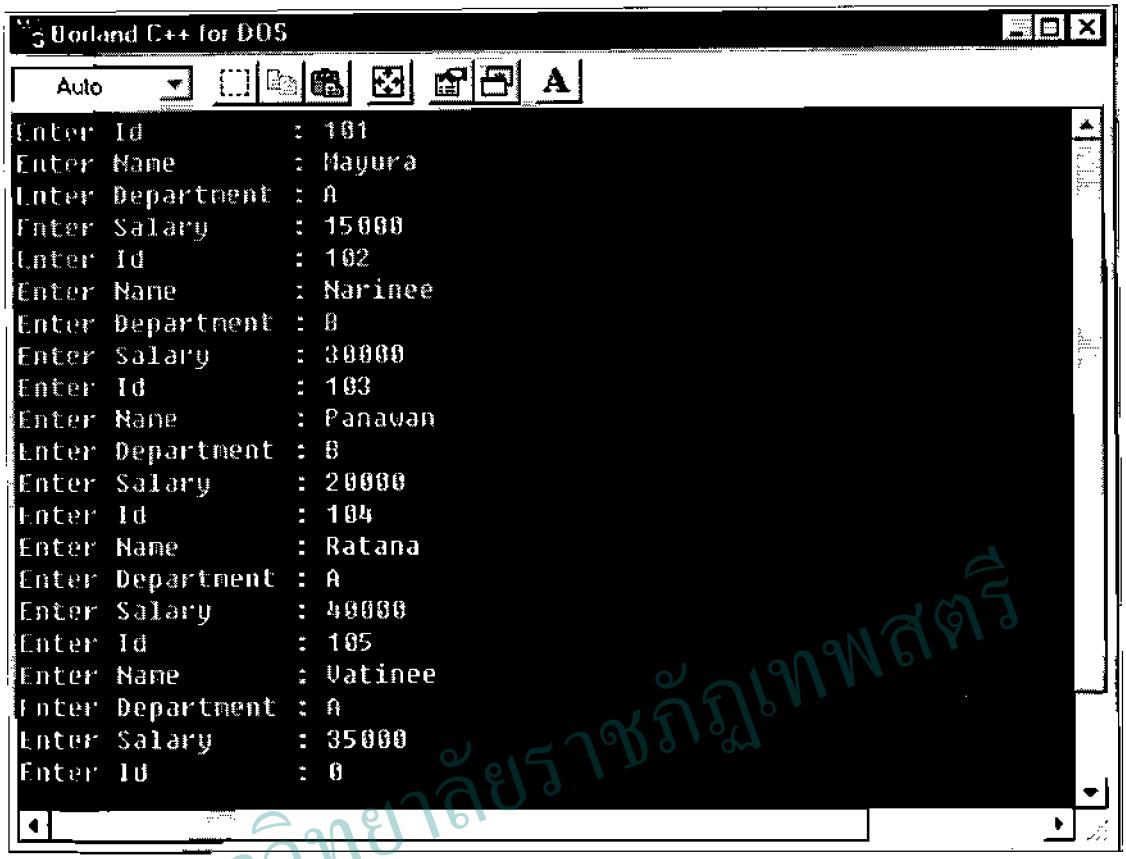


ภาพที่ 6.9 ผังงานแสดงขั้นตอนการทำงานของโปรแกรม

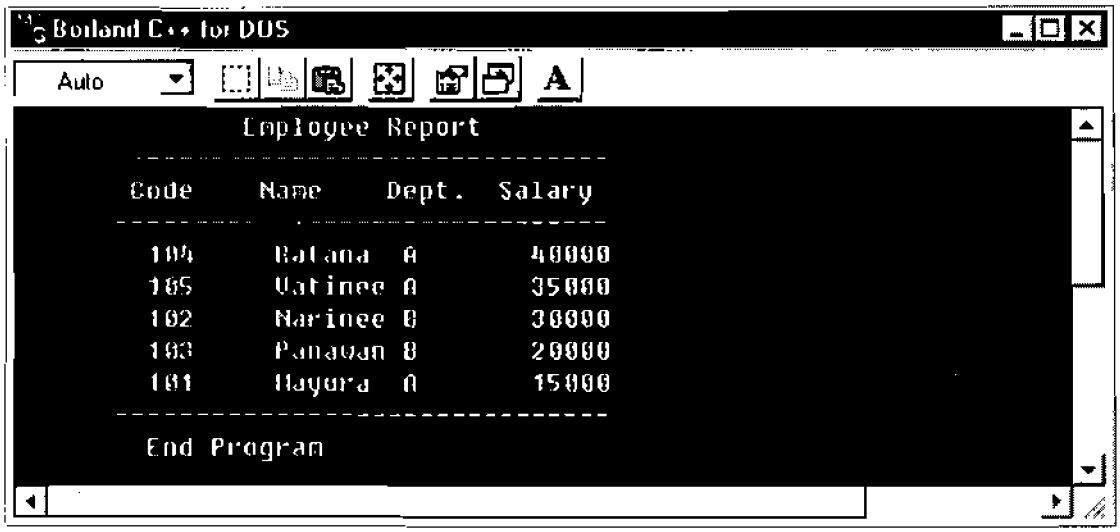
6. ลงรหัสเป็นภาษาซีได้ดังนี้

```
1   #include <iostream.h>
2   #include <conio.h>
3   #include <stdlib.h>
4   struct
5   { char id[5],na[20],dept;
6     long salary;
7   } emprec[100],trec;
8   int i,j,n;
9   void main()
10  {  n = 1;
11     clrscr();
12     cout << "Enter Id      : ";cin >> emprec[n].id;
13     while (atoi(emprec[n].id) != 0)
14     {
15        cout << "Enter Name       : ";cin >> emprec[n].na;
16        cout << "Enter Department : ";cin >> emprec[n].dept;
17        cout << "Enter Salary     : ";cin >> emprec[n].salary;
18         n++;
19        cout << "Enter Id         : ";cin >> emprec[n].id;
20     }
21     n = n - 1;
22     for(i=1;i<=n-1;i++)
23        for(j=1;j<=n-i;j++)
24           if(emprec[j].salary<  emprec[j+1].salary)
25           {       trec = emprec[j];
26                   emprec[j] = emprec[j+1];
```

```
27                  emprec[j+1] = trec;
28              }
29          clrscr();
30          cout<<"          Employee Report \n";
31          cout<<"     ------------------------------ \n";
32          cout<<"      Code   Name   Dept.  Salary  \n";
33          cout<<"     ------------------------------ \n";
34           for(i=1;i<=n;i++)
35           {
36            cout <<"\t"<<emprec[i].id<<"\t";
37            cout <<emprec[i].na<<"\t";
38            cout <<emprec[i].dept<<"\t";
39            cout <<emprec[i].salary<<"\n";
40          }
41          cout<<"     ------------------------------ \n";
42          cout <<"\t"<< "End Program";getch();
43      }
```



ภาพที่ 6.10 ข้อมูลที่ป้อนให้กับโปรแกรม



ภาพที่ 6.11 ผลลัพธ์ของโปรแกรม

โครงสร้างข้อมูลชนิดแฟ้มข้อมูล

แฟ้มข้อมูล (file) หมายถึง การจัดเก็บข้อมูลที่เป็นเรื่องเดียวกันอย่างเป็นระบบระเบียบลงในหน่วยความจำสำรอง เช่น แผ่นดิสก์ ฮาร์ดดิสก์ เทปแม่เหล็ก หรือ หน่วยความจำอื่น ๆ ในที่นี้จะขอกล่าวถึงเพียงแค่ หน่วยความจำสำรองที่เป็น แผ่นดิสก์ กับ ฮาร์ดดิสก์ การจัดเก็บข้อมูลด้วยวิธีการของแฟ้มข้อมูลนี้ จะช่วยให้เรา สามารถเก็บข้อมูลไว้ใช้ในคราวต่อไป หรือ ในอนาคตได้อย่างมีประสิทธิภาพ เนื่องจากคอมพิวเตอร์ มีความสามารถในการประมวลผลด้วยความเร็วสูง และมีความเที่ยงตรงในการหาคำตอบ

การจัดการกับแฟ้มข้อมูลโดยทั่วไปมีการจัดการในลักษณะต่างๆ ดังนี้

1. การจัดเก็บข้อมูล
2. การอ่านข้อมูล
3. การแก้ไขข้อมูล
4. การลบข้อมูล
5. การประมวลผลข้อมูล

ในการจัดการกับแฟ้มข้อมูลมีคำสั่งที่เกี่ยวข้องดังนี้

การกำหนดให้ตัวแปรเป็นชนิดแฟ้มข้อมูล มีรูปแบบดังนี้

FILE *file-var;

FILE หมายถึง ประเภทของข้อมูลในที่นี้ หมายถึง แฟ้มข้อมูล

*file-var หมายถึง ตัวแปรที่ใช้อ้างถึงแฟ้มข้อมูลที่จะใช้งาน

<u>ตัวอย่าง</u>

FILE *stdfi; หมายถึง การกำหนดให้ ตัวแปร stdfi เป็นตัวแปรประเภทแฟ้มข้อมูล

การเปิดแฟ้มข้อมูล มีรูปแบบดังนี้

file-var = fopen(disk-file,mode)

file-var หมายถึงตัวแปรที่ใช้อ้างถึงแฟ้มข้อมูล

fopen หมายถึง การสั่งให้เปิดแฟ้มข้อมูลที่อยู่ในวงเล็บ

disk-file หมายถึง การอ้างถึงไฟล์ที่เก็บในดิสก์

mode หมายถึง การเลือกกระทำกับแฟ้มข้อมูลในแบบต่าง ๆ แสดงในตารางที่ 6.1

ตารางที่ 6.1 ค่าของ mode และความหมายในการเปิดแฟ้มข้อมูล

Mode	ความหมาย
w	เปิดเพื่อสร้างไฟล์ใหม่ในแบบเท็กซ์
r	เปิดเพื่ออ่านข้อมูลในแบบเท็กซ์
a	เปิดเพื่อเขียนข้อมูลแบบต่อท้ายในแบบเท็กซ์
r+	เปิดเพื่อทำการอ่านและเขียนข้อมูลในแบบเท็กซ์
wb	เปิดเพื่อสร้างไฟล์ใหม่ในแบบไบนารี
rb	เปิดเพื่ออ่านข้อมูลในแบบไบนารี
ab	เปิดเพื่อเขียนข้อมูลแบบต่อท้ายในแบบไบนารี
r+b	เปิดเพื่อทำการอ่านและเขียนข้อมูลในแบบไบนารี

(ธันวา ศรีประโมง : หน้า 435)

ตัวอย่าง

stdfi = fopen("student.dat","w"); หมายถึง เปิดแฟ้มข้อมูลชื่อ student.dat เพื่อสร้างแฟ้มข้อมูลในแบบเท็กซ์

stdfi=fopen("student.dat","wb"); หมายถึง เปิดแฟ้มข้อมูลชื่อ student.dat เพื่อการสร้างแฟ้มข้อมูลในแบบไบนารี

stdfi = fopen("student.dat","r+"); หมายถึง เปิดแฟ้มข้อมูลชื่อ student.dat เพื่อทำการแก้ไขข้อมูลในแฟ้มข้อมูลแบบเท็กซ์

stdfi=fopen("student.dat","r+b"); หมายถึง เปิดแฟ้มข้อมูลชื่อ student.dat เพื่อการแก้ไขข้อมูลในแฟ้มข้อมูลแบบไบนารี

การปิดแฟ้มข้อมูล มีรูปแบบดังนี้

fclose(file-var);

file-var หมายถึง แฟ้มข้อมูลที่ต้องการปิด

ตัวอย่าง fclose(stdfi); หมายถึง ปิดแฟ้ม stdfi

การบันทึกข้อมูล มีรูปแบบดังนี้

fwritc(&data,size of data,n,file-var);

&data หมายถึง ข้อมูลที่จะบันทึก

size of data หมายถึง ขนาดข้อมูลที่จะบันทึก

n หมายถึง จำนวนข้อมูลที่จะบันทึก

file-var หมายถึง แฟ้มข้อมูลที่จะทำการบันทึกข้อมูล

ตัวอย่าง

fwritc(&stdrec,size of stdrec,1,stdfi); หมายถึง บันทึกข้อมูลลงในแฟ้มข้อมูล stdfi จำนวน 1 ครั้ง ด้วยข้อมูลที่อยู่ใน stdrec ขนาดที่จะบันทึกเท่ากับ ขนาดของ stdrec

การอ่านข้อมูล มีรูปแบบดังนี้

fread(&data,size of data,n,file-var);

&data หมายถึง ที่เก็บข้อมูลที่อ่านได้

size of data หมายถึง ขนาดข้อมูลที่จะอ่าน

n หมายถึง จำนวนข้อมูลที่จะอ่าน

file-var หมายถึง แฟ้มข้อมูลที่จะทำการอ่านข้อมูล

ตัวอย่าง

fread(&stdrec,size of stdrec,1,stdfi); หมายถึง อ่านข้อมูลจากแฟ้มข้อมูล stdfi จำนวน 1 ครั้ง ข้อมูลที่อ่านได้ส่งไปเก็บที่ stdrec ขนาดที่จะอ่านเท่ากับ ขนาดของ stdrec

การเลื่อนตัวชี้ข้อมูล มีรูปแบบดังนี้

fseek(file-var,(long)(n*size of data),start no);

file-var หมายถึง แฟ้มข้อมูลที่จะทำการเลื่อนตัวชี้

long หมายถึง ข้อมูลประเภทจำนวนเต็มแบบ long

n*size of data หมายถึง จำนวน คูณกับ ขนาดของข้อมูล

start no หมายถึง จุดเริ่มต้นของการเลื่อน

ตัวอย่าง

fseek(stdfi,(long)(5*size of stdrec),0); หมายถึง เลื่อนตัวชี้ของแฟ้มข้อมูล stdfi ไปที่ตำแหน่ง 5 * ขนาดของ stdrec โดยให้เริ่มต้นจาก การนับที่ 0 หรือให้เข้าใจง่ายขึ้น หมายถึง เลื่อนตัวชี้ไปยังเรคคอร์ดที่ 5

การเลื่อนตัวชี้ไปที่จุดเริ่มต้น มีรูปแบบดังนี้

rewind(file-var);

file-var หมายถึง แฟ้มข้อมูลที่ต้องการเลื่อนตัวชี้

ตัวอย่าง

rewind(stdfi); หมายถึง เลื่อนตัวชี้ข้อมูลของแฟ้มข้อมูล stdfi ไปยังจุดเริ่มต้น

การเปลี่ยนชื่อแฟ้มข้อมูล มีรูปแบบดังนี้

rename("old-name","new-name");

old-name หมายถึง ชื่อเก่าที่ต้องการเปลี่ยน

new-name หมายถึง ชื่อใหม่หลังจากเปลี่ยน

ตัวอย่าง

rename("temp.dat","student.dat"); หมายถึง เปลี่ยนชื่อแฟ้มข้อมูล จาก temp.dat เป็น student.dat

การลบแฟ้มข้อมูล มีรูปแบบดังนี้

remove("file-name");

file-name หมายถึง ชื่อแฟ้มข้อมูลที่ต้องการลบ

ตัวอย่าง

remove("student.dat"); หมายถึง ลบแฟ้มข้อมูลชื่อ student.dat

ตัวอย่างที่ 6.4 จงเขียนโปรแกรมเพื่อบันทึกข้อมูลเกี่ยวกับพนักงานลงในแฟ้มข้อมูลชื่อ employ.dat โดยให้มีข้อมูล รหัสพนักงาน ชื่อ แผนก และ เงินเดือน จัดเก็บในแฟ้มข้อมูลแบบไบนารี

วิธีทำ

1. กำหนดให้มีการรับข้อมูลดังนี้

1.1 รหัสพนักงาน

1.2 ชื่อ

1.3 แผนก

1.4 เงินเดือน

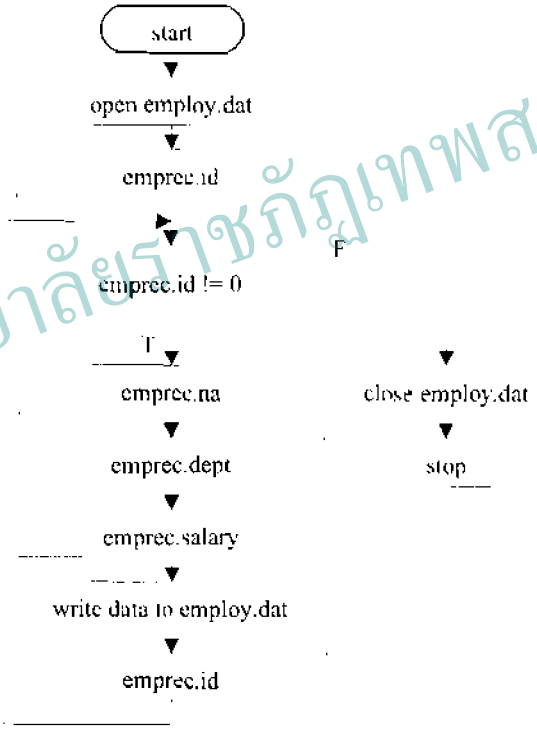
2. กำหนดให้มีการประมวลผลดังนี้

ประมวลผลโดยการบันทึกข้อมูลพนักงานลงในแฟ้มข้อมูล

3. กำหนดตัวแปรดังนี้

ชื่อตัวแปร	ประเภท	ความหมาย
id	char จำนวน 5 ตัว	ฟิลด์เลขประจำตัวพนักงาน
na	char จำนวน 20 ตัว	ฟิลด์ชื่อพนักงาน
dept	char	ฟิลด์แผนก
salary	long	ฟิลด์เงินเดือน
emprec	struct	เรคคอร์ดของพนักงาน

4. กำหนดขั้นตอนของโปรแกรมด้วยผังงานดังนี้

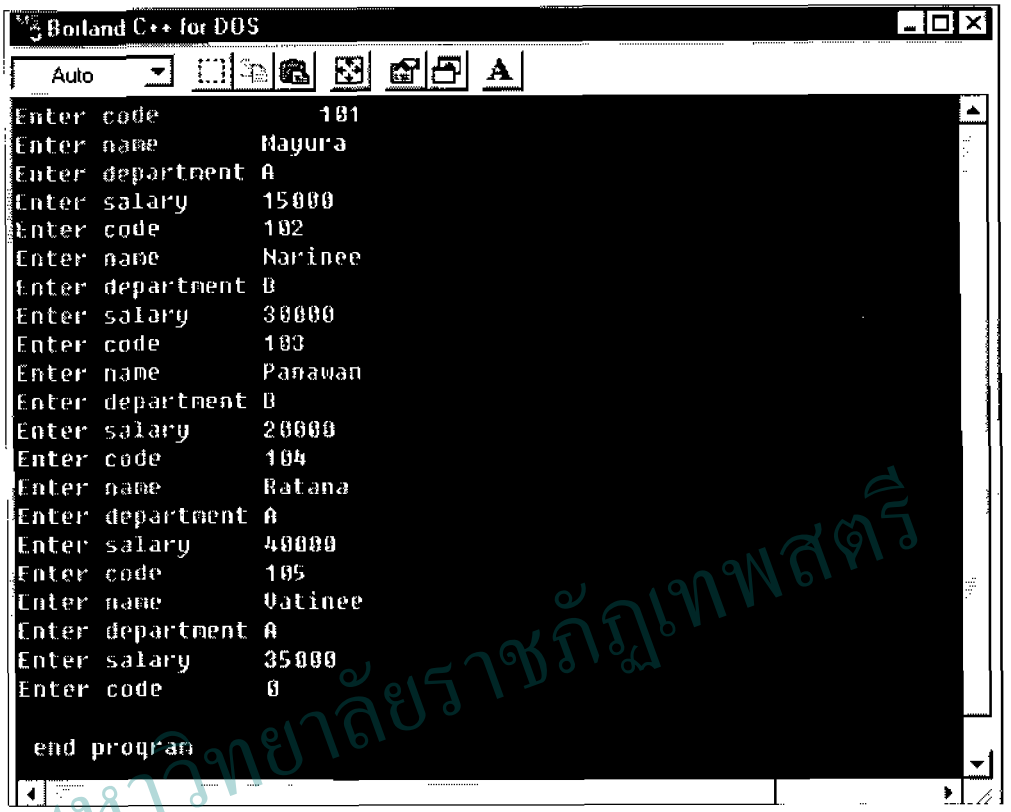


ภาพที่ 6.12 ผังงานแสดงขั้นตอนการทำงานของการบันทึกข้อมูลลงแฟ้มข้อมูล

5. ลงรหัสเป็นโปรแกรมดังนี้

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <conio.h>
4  #include <iostream.h>
5  struct {char id[15];
6          char name[20];
7          char dept;
8          long salary;
9       } erec;
10 FILE *empfi;
11 void main()
12 {
13 empfi = fopen("employ.dat","\wb");
14 clrscr();
15 cout << "Enter Code  ";cin>>erec.id;
16 while (atoi(erec.id)!=0)
17 {
18   cout << "Enter Name       ";cin>>erec.name;
19   cout << "Enter Department ";cin>>erec.dept;
20   cout << "Enter salary     ";cin>>erec.salary;
21   fwrite(&erec,sizeof erec,1,empfi);
22   cout << "Enter Code  ";cin>>erec.id;
23 }
24 fclose(empfi);
25 cout <<"\n end program";getch();
26 }
```

จากโปรแกรมตัวอย่าง เมื่อมีการป้อนข้อมูล ดังภาพที่ 6.13 ข้อมูลเกี่ยวกับพนักงานทั้งหมดที่ป้อน ก็จะถูกบันทึกลงในแฟ้มข้อมูล employ.dat



ภาพที่ 6.13 ข้อมูลที่ป้อนเข้าไปในแฟ้มข้อมูล employ.dat

ตัวอย่างที่ 6.5 จงเขียนโปรแกรมเพื่ออ่านข้อมูลจากแฟ้มข้อมูล employ.dat

<u>วิธีทำ</u>

1. กำหนดการแสดงผล

Employee Report

Code Name Dept. Salary Tax

---

999 xxxxxxx x 99999 99999

999 xxxxxxx x 99999 99999

---

2. กำหนดการรับข้อมูล

รับข้อมูลโดยการอ่านจากแฟ้มข้อมูล employ.dat

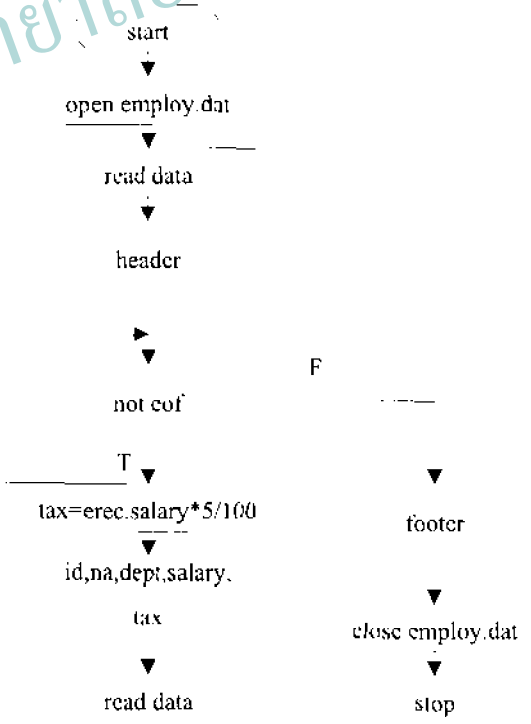
3. กำหนดการประมวลผล

คำนวณหาภาษี จาก 5% ของเงินเดือน

4. กำหนดตัวแปร

ชื่อตัวแปร	ประเภท	ความหมาย
id	char จำนวน 5 ตัว	ฟิลด์เลขประจำตัวพนักงาน
na	char จำนวน 20 ตัว	ฟิลด์ชื่อพนักงาน
dept	char	ฟิลด์แผนก
salary	long	ฟิลด์เงินเดือน
emprec	struct	เรคคอร์ดของพนักงาน
tax	float	ภาษี

5. กำหนดขั้นตอนการทำงานด้วยผังงาน

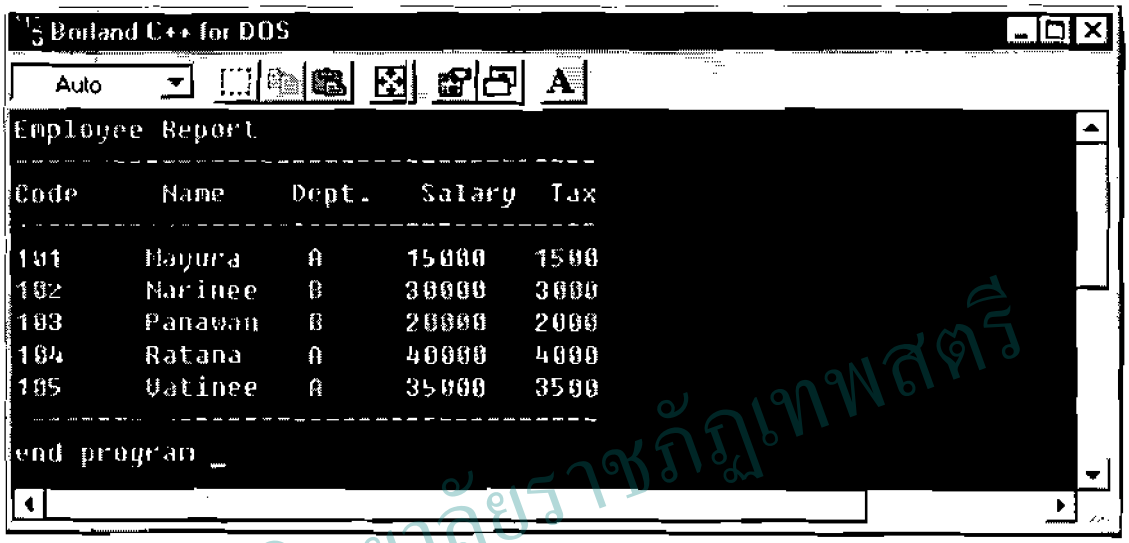


ภาพที่ 6.14 ผังงานแสดงขั้นตอนการทำงานของโปรแกรมอ่านข้อมูล

6. ลงรหัสเป็นโปรแกรมได้ดังนี้

```
1  #include <stdio.h>
2  #include <conio.h>
3  #include <iostream.h>
4  struct {char id[15];
5          char name[20];
6          char dept;
7          long salary;  } erec;
8  FILE *empfi;
9  void main()
10 {float tax;
11  empfi = fopen("employ.dat","rb");
12  fread(&erec,sizeof erec,1,empfi);
13  clrscr();
14  cout << "Employee Report \n";
15  cout << "-----------------------\n";
16  cout << "Code \t name \t Dept. \t salary Tax \n";
17  cout << "-----------------------\n";
18  while (!feof(empfi))
19  {   tax = erec.salary * 10 / 100;
20    cout << erec.id<<"\t"<<erec.name<<"\t"<<erec.dept;
21    cout << "\t" << erec.salary << "\t"<< tax<<"\n";
22    fread(&erec,sizeof erec,1,empfi); }
23  cout << "-----------------------\n";
24  fclose(empfi);
25  cout <<"\n end program";getch();
26  }
```

จากโปรแกรมคำสั่งในบรรทัดที่ 19 ฟังก์ชัน eof หมายถึง การตรวจสอบว่า ยังมีข้อมูลอยู่หรือไม่ โดยเครื่องจะตรวจสอบจาก ตัวชี้ข้อมูลในแฟ้มข้อมูล ถ้าตัวชี้ข้อมูลชี้อยู่ในบริเวณที่มีข้อมูลคำตอบจะเป็น เท็จ ถ้าชี้ในตำแหน่งที่หมดข้อมูลแล้ว คำตอบจะเป็นจริง ความหมายโดยรวมของคำสั่งในบรรทัดนี้คือ อ่านข้อมูลขณะที่ยังไม่หมดข้อมูล นั่นเอง ผลลัพธ์แสดงออกมาดังภาพที่ 6.15



ภาพที่ 6.15 ผลลัพธ์ของโปรแกรมอ่านข้อมูล

ตัวอย่างที่ 6.6 จงเขียนโปรแกรมเพื่อลบข้อมูลออกจาก แฟ้มข้อมูล employ.dat ในบางเรคคอร์ดที่ต้องการลบ โดยให้ผู้ใช้ป้อนข้อมูล รหัส ของพนักงานที่ต้องการลบ

วิธีทำ

1. กำหนดการแสดงผล

Enter Delete Id ...

Code...

Name...

Dept. ...

Salary ...

Press Any Key

2. กำหนดการรับข้อมูล

รับข้อมูลรหัสพนักงานที่ต้องการลบ

3. กำหนดการประมวลผล

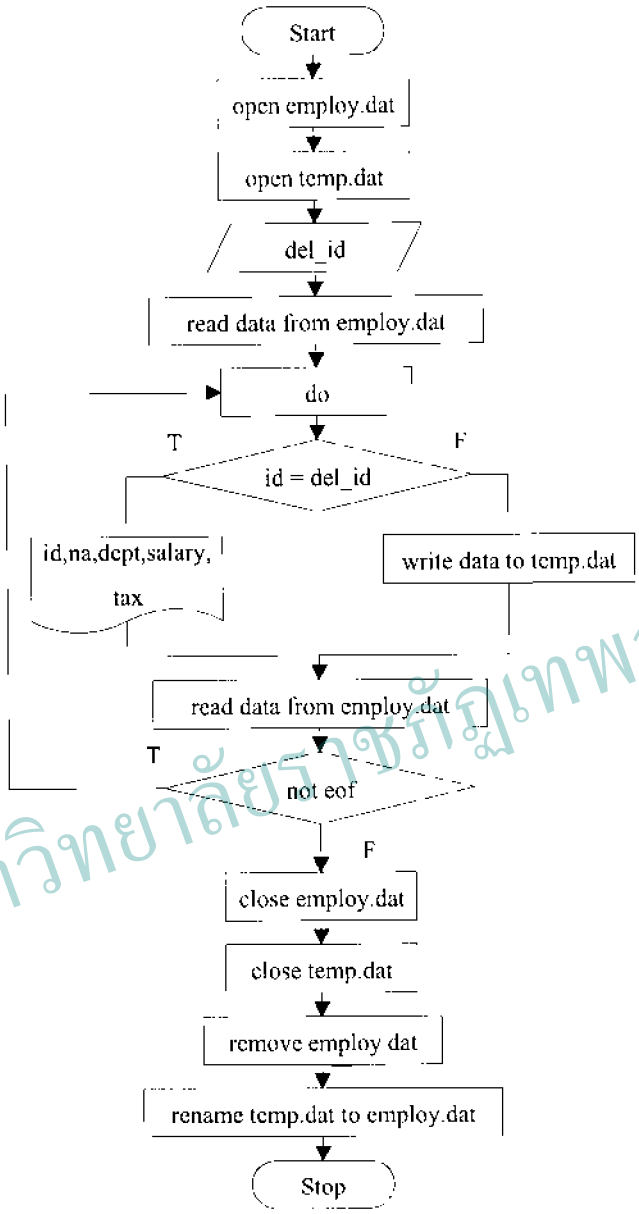
3.1 ค้นหาข้อมูลที่ต้องการลบโดยตรวจสอบว่ารหัสที่ป้อนเข้ามา เท่ากันกับรหัสที่อยู่ในแฟ้มหรือไม่

3.2 บันทึกข้อมูลลงในแฟ้มชั่วคราว (temp.dat) ถ้าไม่ใช่ข้อมูลที่จะลบ

4. กำหนดตัวแปร

ชื่อตัวแปร	ประเภท	ความหมาย
id	char จำนวน 5 ตัว	ฟิลด์เลขประจำตัวพนักงาน
na	char จำนวน 20 ตัว	ฟิลด์ชื่อพนักงาน
dept	char	ฟิลด์แผนก
salary	long	ฟิลด์เงินเดือน
emprec	struct	เรคคอร์ดของพนักงาน
del_id	char จำนวน 5 ตัว	เลขประจำตัวที่ต้องการลบ

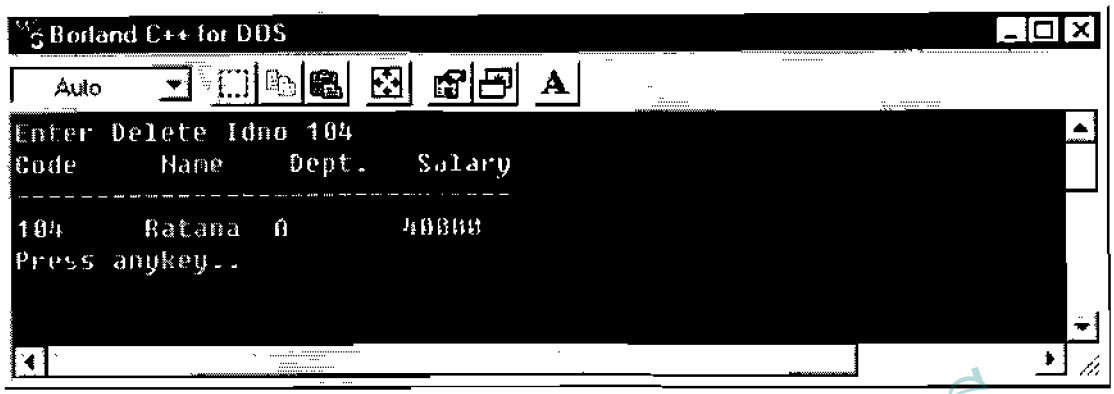
5. กำหนดขั้นตอนการทำงานด้วยผังงาน



ภาพที่ 6.16 ผังงานแสดงขั้นตอนการทำงานของโปรแกรมลบข้อมูล

6. ลงรหัสเป็นโปรแกรมได้ดังนี้

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <conio.h>
4  #include <iostream.h>
5  struct emp {char id[15],name[20],dept; long salary; } erec;
6  FILE *outfile ,*temp;
7  char del_id[5];
8  void main()
9  { outfile = fopen("employ.dat","r+b");
10    temp = fopen("temp.dat","wb");
11   cout << "Enter Delete Idno ";cin >> del_id;
12   fread(&erec,sizeof erec,1,outfile);
13  do  { if (atoi(erec.id)==atoi(del_id))
14       { cout << "Code \t name \t Dept. \t salary \n";
15         cout << "-----------------------\n";
16         cout << erec.id<<"\t"<<erec.name<<"\t"<<erec.dept:
17         cout << "\t" << erec.salary << "\n";
18         cout << "Press anykey.. \n";getch(); }
19       else
20         fwrite(&erec,sizeof erec,1,temp);
21   fread(&erec,sizeof erec,1,outfile);
22  } while (!feof(outfile));
23  fclose(outfile);fclose(temp);
24  remove("emply.dat");
25  rename("temp.dat","employ.dat");
26  cout <<"\n end program";getch();}
```



ภาพที่ 6.17 ผลลัพธ์ของโปรแกรมลบข้อมูล

จากผลลัพธ์ของโปรแกรมผู้ใช้ใส่รหัสพนักงานที่ต้องการลบได้แก่ รหัส 104 โปรแกรมก็จะทำการเปรียบเทียบรหัสประจำตัวพนักงานข้อมูลที่อ่านได้ (crec.id) กับรหัสพนักงานที่ต้องการลบ (del_id) ถ้าเท่ากัน จะไม่มีการบันทึกข้อมูลชุดนั้นลงใน temp.dat แต่ถ้าไม่เท่ากัน จะทำการบันทึกลงใน temp.dat สุดท้ายก็จะทำการเปลี่ยนชื่อ จาก temp.dat มาเป็น employ.dat ดังนั้นหลังจากจบโปรแกรมนี้ไปแล้ว จะไม่มีข้อมูลของพนักงานที่มีรหัส 104 อีกต่อไป

สรุป

การเก็บข้อมูลในภาษาซี เป็นการเก็บข้อมูลในหน่วยความจำหลัก และ หน่วยความจำสำรอง การเก็บข้อมูลโดยทั่วไปนั้น เราสามารถเก็บข้อมูลได้อย่างละ ไม่เกิน 1 ตัว แต่ถ้าเราใช้การเก็บข้อมูลในลักษณะโครงสร้างแล้ว เราจะสามารถ เก็บข้อมูล ที่มีลักษณะเดียวกัน ได้ครั้งละหลาย ๆ ตัว โครงสร้างในการเก็บข้อมูลที่กล่าวถึงในบทนี้ ประกอบไปด้วยโครงสร้าง แบบอาเรย์ ซึ่งหมายถึงการจัดเก็บข้อมูลในหน่วยความจำหลักโดยใช้ชื่อตัวแปร 1 ตัว แต่มีตัวชี้ ซึ่งเป็นตัวเลขลำดับของตัวแปรเป็นตัวกำหนดว่า ข้อมูลเหล่านั้นเก็บอยู่ในตำแหน่งใดในหน่วยความจำ โครงสร้างแบบเรคคอร์ด หรือในภาษาซีเรียกว่า ข้อมูลแบบโครงสร้าง เป็นการเก็บข้อมูลหลายประเภทไว้ในที่เดียวกัน ข้อมูลประเภทนี้ สามารถนำไปประยุกต์ใช้กับ หน่วยความจำหลัก และ

หน่วยความจำสำรอง อันได้แก่ ดิสก์เก็ต หรือ สื่อข้อมูลชนิดอื่นได้ โครงสร้างข้อมูลประเภทสุดท้ายในบทเรียนนี้ ได้แก่ โครงสร้างข้อมูลประเภท แฟ้มข้อมูล ซึ่งจะเป็นการจัดเก็บข้อมูลลงในสื่อข้อมูล ซึ่งอยู่ที่หน่วยความจำสำรอง ในที่นี้ หมายถึง ดิสก์เก็ต หรือ ฮาร์ดดิสก์ การจัดการกับข้อมูลภายในแฟ้มข้อมูล ประกอบด้วยการเก็บข้อมูล การอ่านข้อมูล การแก้ไขข้อมูล การลบข้อมูล และการประมวลผลข้อมูล ซึ่งแต่ละอย่างจะมีวิธีการซึ่งกล่าวไว้แล้วในรายละเอียด

คำถามทบทวน

จงตอบคำถามต่อไปนี้โดยสังเขป

1. กำหนดให้ตัวแปร x เก็บข้อมูลประเภทตัวเลขจำนวนจริง โดยใช้โครงสร้างแบบอาเรย์ ให้มีที่เก็บข้อมูล จำนวน 100 ที่ จงเขียนด้วยวิธีการของภาษาซี
2. จากข้อที่ 1 ต้องการให้ตัวแปรทุกตัวมีค่าเท่ากับ 0 จงเขียนด้วยวิธีการของภาษาซี
3. ข้อมูลแบบ struct หมายถึงการเก็บข้อมูลในแบบใด จงอธิบายพร้อมยกตัวอย่างประกอบ
4. ต้องการเก็บข้อมูลประเภท struct ไว้ในหน่วยความจำหลัก จำนวน 100 ชุด จะใช้วิธีใดในการเก็บ จงอธิบายพร้อมยกตัวอย่างประกอบ
5. การจัดการกับข้อมูล ในแฟ้มข้อมูล มีกี่ลักษณะ อะไรบ้าง จงอธิบายในแต่ละแบบ

เอกสารอ้างอิง

เกษมสันต์ พานิชการ. (2537). **C++ และหลักการของ OOP ฉบับเริ่มต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

ทักษิณา สวนานนท์. (2544). **พจนานุกรมศัพท์คอมพิวเตอร์ ฉบับนิสิตนักศึกษา.** กรุงเทพฯ: ไอบริด พริ้นติ้ง.

ธันวา ศรีประโมง. (2539). **การเขียนโปรแกรมภาษาซีสำหรับวิศวกรรม.** กรุงเทพฯ: มหาวิทยาลัยเทคโนโลยีมหานคร.

เบญจพร ศักดิ์ศิริ. (2540). **ทฤษฎีและตัวอย่างโจทย์ การเขียนโปรแกรมด้วยภาษา C++.** กรุงเทพฯ: แมคกรอ-ฮิล อินเตอร์เนชั่นแนล เอ็นเตอร์ไพร์ส, อิงค์.

มนตรี พจนารถลาวัณย์. (2535). **การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี.** กรุงเทพฯ: เอช-เอน การพิมพ์.

วรรณวิภา จำเริญดารารัศมี. (2535). **วิทยาการคอมพิวเตอร์เบื้องต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

ศิริรัตน์ ชำนาญรบ, เกื้อกูล ตาเย็น, และวัชระ โพธิสรณ์. (2539). **ความรู้เบื้องต้นเกี่ยวกับคอมพิวเตอร์.** กรุงเทพฯ: ภูมิบัณฑิต.

บรรณานุกรม

เกษมสันต์ พานิชการ. (2537). **C++ และหลักการของ OOP ฉบับเริ่มต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

ครรชิต มาลัยวงศ์. (2538). **พจนานุกรมคอมพิวเตอร์.** กรุงเทพ ฯ: ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ.

ทักษิณา สวนานนท์. (2544). **พจนานุกรมศัพท์คอมพิวเตอร์ ฉบับนิสิตนักศึกษา.** กรุงเทพฯ: ไอบริด พริ้นติ้ง.

ธงชัย สิทธิกรณ์. (2540). **ทฤษฎีระบบคอมพิวเตอร์.** กรุงเทพฯ: สยามสปอร์ต ซินดิเคท.

ธันวา ศรีประโมง. (2539). **การเขียนโปรแกรมภาษาซีสำหรับวิศวกรรม.** กรุงเทพฯ: มหาวิทยาลัยเทคโนโลยีมหานคร.

เบญจพร ศักดิ์ศิริ. (2540). **ทฤษฎีและตัวอย่างโจทย์ การเขียนโปรแกรมด้วยภาษา C++.** กรุงเทพฯ: แมคกรอ-ฮิล อินเตอร์เนชั่นแนล เอ็นเตอร์ไพร์ส, อิงค์.

มนตรี พจนารถลาวัณย์. (2535). **การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี.** กรุงเทพฯ: เอช-เอน การพิมพ์.

วรรณวิภา จำเริญดารารัศมี. (2535). **วิทยาการคอมพิวเตอร์เบื้องต้น.** กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

วาสนา สุขกระสานติ. (2540). โลกของคอมพิวเตอร์และสารสนเทศ. กรุงเทพฯ: จุฬาลงกรณ์มหาวิทยาลัย.

ศิริรัตน์ ชำนาญรบ, เกื้อกูล ตาเย็น, และวัชระ โพธิสรณ์. (2539). **ความรู้เบื้องต้นเกี่ยวกับคอมพิวเตอร์.** กรุงเทพฯ: ภูมิบัณฑิต.

Borland International. (1992). **Programmer's Guide.** Scotts Valley, California: Borland International.

Horsington Gordon. (1991). **Programming in ANSI Standard C.** Singapore: John Wiley & Sons (SEA) Pte.