บทที่ Ä

กรณีศึกษาระบบงานทะเบียน

จุดประสงค์เชิงพฤติกรรม

1. เข้าใจระบบงานของสำนักทะเบียน ในมุมมองของ Programmer มากขึ้น
2. เข้าใจการสร้างระบบงานคอมพิวเตอร์ โดยพิจารณาจากความต้องการของผู้ใช้
3. มีประสบการณ์ และมองเห็นแนวการออกแบบเมนูทั้งระบบงาน
4. เข้าใจแนวการเขียน และการแปลโปรแกรม โดยใช้วิธีการอย่างง่าย
5. เข้าใจการทำรายงาน และการปรับปรุงข้อมูลในรูปแบบต่าง ๆ
6. เข้าใจการเขียนเมนู และเรียกใช้งานอย่างง่าย

หัวข้อในบทเรียน

5.1 ศึกษาความต้องการ

5.2 โครงสร้างข้อมูล

5.3 วิเคราะห์ระบบงานเบื้องต้น

5.4 โปรแกรมในระบบงานทะเบียน

- 5.4.1 โปรแกรมต่าง ๆ ใน MAIN.PRG : สารบัญของโปรแกรม
- 5.4.2 โปรแกรมต่าง ๆ ใน SUB01.PRG : รวมโปรแกรมเตรียมข้อมูล และการลงทะเบียน
- 5.4.3 โปรแกรมต่าง ๆ ใน SUB02.PRG : รวมโปรแกรมรายงานหลังลงทะเบียน
- 5.4.4 โปรแกรมต่าง ๆ ใน SUB03.PRG : รวมรายงานตามคำร้องของผู้ใช้
- 5.4.5 โปรแกรมต่าง ๆ ใน SUB04.PRG : รวมรายงานประจำภาคเรียน
- 5.4.6 โปรแกรมต่าง ๆ ใน SUB05.PRG : รวมโปรแกรมปรับปรุงข้อมูล

ตัวอย่างโปรแกรมเกี่ยวกับระบบงานทะเบียน 38 ตัวอย่าง

หมายเหตุ

ตัวอย่างในบทนี้ประยุกต์จากประสบการณ์ที่ผู้เขียนทำงานเขียนโปรแกรมให้สำนักทะเบียนโยนก ซึ่งได้มีการประสานงานกับคุณลัดดาวรรณ ปัญญา ซึ่งเป็นผู้ปฏิบัติงาน แล้วเรียบเรียงใหม่ ให้เหมาะสมกับเนื้อหาวิชา และเวลาที่จะใช้สอน ให้ดูเหมือนเป็นระบบที่อาจนำไปใช้ได้ แต่ในความเป็นจริงแล้ว เป็นเพียงกรณีศึกษา เพื่อชี้ให้เห็นถึงจุดที่ต้องปรับปรุงเพิ่มเติมอีกมาก .. ประกอบการบรรยายในชั้นเรียน และโปรแกรมทั้งหมดในหนังสือ นักศึกษาสามารถ Download ไปศึกษา ประกอบการเรียนภาคปฏิบัติ

บทที่ 5 กรณีศึกษาระบบงานทะเบียน

เนื่องจากระบบทะเบียนทางการศึกษา ซึ่งเก็บประวัติการเรียนของนักเรียน เป็นระบบที่นำมาใช้เก็บประวัติการศึกษาในทุกสถาบัน โดยแต่ละสถาบันมีรายละเอียดบางประการที่ไม่เหมือนกัน แต่แก่นแท้ของงานทะเบียนแล้วใกล้เคียงกัน คืองานบริการข้อมูลการศึกษาสำหรับนักศึกษา อาจารย์และผู้บริหารของสถานศึกษานั้น ๆ

ด้วยเหตุที่ผู้สนใจศึกษาการเขียนโปรแกรมทุกคน ต้องผ่านการศึกษามาอย่างน้อยหนึ่งสถาบัน และได้เข้าไปเกี่ยวข้องกับงานทะเบียน หรือมีรายชื่อของตนในสถาบันนั้น และได้รับรายงานเช่น ใบแจ้งผลการเรียน ทำให้ทุกคนได้คุ้นเคย และมองภาพรวมของงานทะเบียน ว่าระบบนี้ควรมีงานอะไรที่ต้องทำบ้าง

ดังนั้นผู้เขียนจึงของนำระบบทะเบียนมาเป็นกรณีศึกษา เพื่อให้ผู้อ่านได้เข้าใจในเนื้อหาของระบบงาน เพื่อนำไปสู่หลักการพัฒนาโปรแกรมในระบบอื่น ๆ ต่อไป ซึ่งผู้เขียนขอเริ่มตั้งแต่การศึกษาความต้องการของผู้ใช้ แล้วนำมาวิเคราะห์หาความต้องการ ที่จะนำมาเป็นแนวทางในการพัฒนาเป็นระบบย่อยต่าง ๆ จนถึงการเขียนโปรแกรมให้ได้ตามความต้องการ[12]

& 5.1 ศึกษาความต้องการ

แผนกทะเบียนที่จะศึกษาต่อไปนี้ ไม่ได้เน้นที่จะศึกษาจนครอบคลุมทุกอย่างที่แผนกทะเบียนทำ แต่จะศึกษาเฉพาะส่วนงานที่เกี่ยวกับการลงทะเบียน จัดเก็บเกรด และรายงานเกรดให้นักศึกษาทราบ

5.1.1 บุคคลที่เกี่ยวข้องกับงาน

ลำดับ	ผู้เกี่ยวข้อง	ความสัมพันธ์ต่อข้อมูล
1	นักศึกษา	มาลงทะเบียนเรียน และรอรับผลการเรียนเมื่อสิ้นภาคการศึกษา
2	อาจารย์ผู้สอน	เป็นผู้ระบุวิชาที่ต้องการสอน รอรับรายชื่อ และส่งเกรดท้ายภาค
3	ผู้บริหาร	เป็นผู้ให้นโยบาย และรับรายงานสรุปผลต่าง ๆ
4	แผนกทะเบียน	ทำหน้าที่ติดต่อกับฝ่ายต่าง ๆ เพื่อนำข้อมูลเข้าระบบ และพิมพ์รายงาน

5.1.2 ระบบงานที่เป็นอยู่ และสิ่งที่ผู้ใช้ต้องการ

ความต้องการของผู้ใช้มีมากมาย โดยเฉพาะการพูดคุยกับผู้ใช้ที่ไม่รู้จักคอมพิวเตอร์ จำเป็นต้องตีความหมายระบบงานที่เป็นอยู่ และเรียบเรียงใหม่เป็นข้อ ๆ แล้วให้ผู้ใช้ตรวจสอบอีกครั้งหนึ่งว่าสิ่งที่ผู้ใช้ต้องการ กับทีมพัฒนาระบบเข้าใจตรงกันหรือไม่

บ่อยครั้งที่การพูดคุยระหว่างทีมพัฒนาระบบ กับกลุ่มผู้ใช้ ถึงปัญหาในระบบงานเดิม ทำให้ต้องปรับปรุงการทำงานแบบเดิมในหลาย ๆ จุด เพราะในการนำระบบคอมพิวเตอร์เข้ามาใช้นั้น การทำงานต้องมีขั้นตอนที่ชัดเจน และตรวจสอบความถูกต้องได้ การทะเลาะและไม่เข้าใจกันจึงเกิดขึ้นจนเป็นเรื่องปกติในระหว่างการพัฒนาระบบงานใหม่ แต่ทั้งสองฝ่ายควรทำความเข้าใจซึ่งกันและกัน เพื่อความสมบูรณ์ของระบบใหม่ที่จะนำมาใช้ หลังจากการพัฒนาเสร็จสิ้นลง

---

[12] ระบบทะเบียนนี้อ้างอิงจากประสบการณ์ของผู้เขียน ที่ทำงานในสำนักทะเบียนวิทยาลัยโยนก แล้วประยุกต์มาเป็นรูปแบบของ Clipper ซึ่งเดิมพัฒนาด้วย COBOL แล้วเปลี่ยนมาเป็น Microsoft Access

บางครั้งจำเป็นต้องเปลี่ยนแปลงระบบงานที่เป็นอยู่ให้มีความเป็นระเบียบ เพื่อจะได้จัดเก็บข้อมูลและประมวลผลในคอมพิวเตอร์ได้อย่างถูกต้อง รายการต่อไปนี้เป็นความต้องการของผู้ใช้ที่ได้ปรับปรุงให้เข้าใจได้ง่าย เป็นข้อ ๆ โดยศึกษามาจากระบบงานที่มีอยู่

ความต้องการของผู้ใช้

1. อาจารย์ผู้สอนให้ข้อมูลการเปิดวิชา ก่อนเปิดภาคการศึกษาใหม่ 1 เดือน
2. นำข้อมูลนักศึกษาใหม่เข้าฐานข้อมูล ก่อนเปิดเรียน
3. การลงทะเบียน มีการตรวจสอบจำนวนไม่ให้เกินที่กำหนด
4. การขอแก้ไข บอกเพิ่ม-ลดวิชา 1 สัปดาห์หลังวันลงทะเบียน
5. พิมพ์ข้อมูลการลงทะเบียนติดประกาศ ให้นักศึกษาตรวจสอบอีกครั้งหนึ่ง
6. พิมพ์รายชื่อนักเรียนให้อาจารย์ผู้สอน
7. พิมพ์สรุปการลงทะเบียนแต่ละวิชาให้ผู้บริหาร
8. การขอเพิกถอนวิชา ของนักเรียน
9. พิมพ์รายชื่อแต่ละวิชาติดหน้าห้องสอบ และเลขที่โต๊ะของนักศึกษา
10. การส่งเกรดของอาจารย์ เมื่อสิ้นภาคเรียน
11. พิมพ์ใบรายงานผลการเรียนประจำภาค
12. ปรับปรุงข้อมูลอาจารย์ เมื่อรับอาจารย์ใหม่ ลาออก หรือแต่งงาน
13. พิมพ์รายชื่อวิชาที่อาจารย์ผู้สอนแต่ละท่านรับผิดชอบ
14. พิมพ์รายชื่อวิชาที่ห้องเรียนแต่ละห้องถูกใช้
15. แสดงตารางเรียนของนักศึกษา เมื่อผู้ปกครองมาขอพบ
16. พิมพ์สรุปจำนวนนักศึกษาแต่ละวิชา
17. พิมพ์รายชื่อแยกตามสถานภาพ
18. พิมพ์รายชื่อแยกตามสาขา
19. พิมพ์ใบรายงานผลการเรียนตลอดหลักสูตร เมื่อนักศึกษาร้องขอ
20. การปรับปรุงและพิมพ์ข้อมูลจากแฟ้มมาตรวจสอบ

& 5.2 โครงสร้างข้อมูล

เมื่อทราบถึงสิ่งที่ต้องการ ทีมพัฒนาระบบจะออกแบบแฟ้มข้อมูลสำหรับเก็บข้อมูลหลาย ๆ แฟ้มซึ่งรวมกันแล้วเรียกว่าฐานข้อมูล ซึ่งจะเก็บข้อมูลเพื่อรองรับกับสิ่งที่ผู้ใช้ต้องการ จึงต้องมีการลองผิดลองถูก หากกลุ่มแฟ้มที่จัดขึ้นยังไม่สามารถเก็บข้อมูลเพื่อสนองสิ่งที่ผู้ใช้ต้องการได้ จะต้องแก้ไขใหม่จนกว่าจะได้กลุ่มแฟ้มที่รองรับงานของผู้ใช้ได้หมด

แฟ้มที่เกี่ยวข้องกับระบบงานทะเบียน

1. แฟ้มนักศึกษา IDSTD.DBF

1.1 รหัส	*IDSTD	9(7)	
(2หลักแรก:ปีที่เข้า 2หลักต่อมา:คณะที่เรียน 3หลักสุดท้าย:ลำดับที่)			
1.2 ชื่อ	INAME	X(20)	
1.3 สกุล	ISURN	X(20)	
1.4 สาขา	IMAJOR	99	
(11:บัญชี 12:การจัดการ 13:การตลาด 14:คอมพิวเตอร์ธุรกิจ)			
1.5 สถานภาพ	ISTAT	X	
(A:กำลังศึกษาอยู่ D:ลาออก)			

ตัวอย่างข้อมูล

3901001 นาย สมคิด สมควร 11 A
3901002 น.ส.ถนัด ประดิษฐ์ 14 A
3901003 นาย สมบัติ ภูเบศร์ 12 D

2. แฟ้มวิชา SUBJECT.DBF

2.1 ชื่อวิชา	*SUBJ	X(8)	
(4หลักแรก:ชื่อหมวด 3หลักหลัง:ชื่อลำดับของวิชาในหมวด)			
2.2 รหัสผู้สอน	TEACHID	9(7)	
(2หลักแรก:ปีที่เข้า ต่อมา:หน่วยงานที่ประจำ 3หลักสุดท้าย:ลำดับ)			
2.3 เลขที่ห้อง	ROOM	X(4)	
(เบอร์ห้องที่ปรากฏหน้าห้องตามตึกต่าง ๆ)			
2.4 เวลาเรียน	STIME	X(11)	
(เช่น 09.00-11.50 เริ่มเรียน จนถึงเลิกเรียน)			
2.5 จำนวนลงทะเบียน	SREGIST	999	
2.6 จำนวนรับได้	SLIMIT	999	

ตัวอย่างข้อมูล

SOCI 101 3930005 4201 09.00-11.50 28 30
BCOM101 3914007 1203 08.00-10.50 6 6
ENGL202 3930002 1211 14.00-16.50 25 25

3. แฟ้มอาจารย์ผู้สอน TEACHER.DBF

3.1 รหัสผู้สอน	*TEACHID	9(7)	
(2หลักแรก:ปีที่เข้า ต่อมา:หน่วยงาน 3หลักสุดท้าย:ลำดับที่)			
3.2 ชื่อผู้สอน	TNAME	X(20)	
3.3 สกุลผู้สอน	TSURN	X(20)	
3.4 อายุ	TAGE	999	
3.5 เพศ	TSEX	X	
(M:ผู้ชาย F:ผู้หญิง)			

ตัวอย่างข้อมูล

3910001 นาย อติชาติ ปรีชา 40 M
3914007 น.ส.ศิริลักษณ์ เกศรินทร์ 45 F
3930005 นาย จักรพันธ์ สุวรรณ 34 M

4. แฟ้มลงทะเบียน REGIST.DBF

4.1 ปีการศึกษา	*RYEAR	9999
4.2 ภาคเรียน	*RSEM	X
4.3 รหัสนักศึกษา	*IDSTD	9(7)
4.4 ชื่อวิชา	*SUBJ	X(8)
4.5 เกรดที่ได้รับ	RGRADE	9V9

(0:F 1:D 1.5:D+ 2:C 2.5:C+ 3:B 3.5:B+ 4:A)

ตัวอย่างข้อมูล

```
2002 1 3901001 BCOM 101 4.0
2002 1 3901001 SOCI 101 2.5
2002 1 3901001 ENGL 101 2.0
2002 1 3901002 BCOM 101 1.5
2002 1 3901002 ENGL 201 1.0
2002 1 3901003 BCOM 301 3.5
2002 1 3901003 MATH 303 3.0
2002 2 3901001 GOVT 101 3.0
2002 2 3901001 ENGL 102 3.5
2002 2 3901003 ARTS 201 1.0
2002 2 3901003 COMM 401 1.5
2002 2 3901003 BCOM 305 2.0
```

5. แฟ้มห้อง ROOM.DBF

5.1 เลขที่ห้อง	*ROOM	X(4)
5.2 ลำดับโต๊ะ	RCSEQ	999
5.3 เลขที่โต๊ะ	RCHAIR	XX

(แต่ละห้องเลขโต๊ะเรียงไม่เหมือนกัน เช่น A1,A2,A3,...)

ตัวอย่างข้อมูล

```
1201 1 A1
1201 2 A2
1201 3 A3
1201 4 B1
1201 5 B2
1202 1 A1
1202 2 A2
1202 3 B1
```

& 5.3 วิเคราะห์ระบบงานเบื้องต้น

การวิเคราะห์ระบบงานเบื้องต้น ในครั้งนี้จะสนใจในเรื่องของสิ่งที่ผู้ใช้ต้องการ แล้วนำมาแยกเป็นระบบเล็ก เพื่อที่จะจับกลุ่มให้กระชับ และนำไปออกแบบเมนู แล้วนำไปเป็นแนวทางในการจัดทำโปรแกรม แต่หากนำไปใช้งานจริงจะต้องมีการศึกษาความเป็นไปได้ ผังงาน ความสัมพันธ์ระหว่างข้อมูล และต้นทุนให้ชัดเจนกว่านี้

5.3.1 จัดระบบย่อยจากความต้องการของผู้ใช้

ระบบเตรียมงานและงานลงทะเบียน

1. อาจารย์ผู้สอนให้ข้อมูลการเปิดวิชา ก่อนเปิดภาคการศึกษาใหม่ 1 เดือน
2. ปรับปรุงข้อมูลนักศึกษาเดิม และนำข้อมูลนักศึกษาใหม่เข้าฐานข้อมูล ก่อนเปิดเรียน
3. การลงทะเบียน มีการตรวจสอบจำนวนไม่ให้เกินที่กำหนด
4. การขอแก้ไข บอกเพิ่ม-ลดวิชา 1 สัปดาห์หลังวันลงทะเบียน

ระบบงานหลังลงทะเบียน

1. พิมพ์ข้อมูลการลงทะเบียนติดประกาศ ให้นักศึกษาตรวจสอบอีกครั้งหนึ่ง
2. พิมพ์รายชื่อนักเรียนให้อาจารย์ผู้สอน
3. พิมพ์สรุปการลงทะเบียนแต่ละวิชาให้ผู้บริหาร
4. พิมพ์รายชื่อวิชาที่ห้องเรียนแต่ละห้องถูกใช้
5. พิมพ์รายชื่อวิชาที่อาจารย์ผู้สอนแต่ละท่านรับผิดชอบ
6. พิมพ์สรุปจำนวนนักศึกษาแต่ละวิชา

ระบบงานเมื่อถูกร้องขอ

1. การขอเพิกถอนวิชา ของนักเรียน
2. แสดงตารางเรียนของนักศึกษา เมื่อผู้ปกครองมาขอพบ
3. พิมพ์ใบรายงานผลการเรียนตลอดหลักสูตร เมื่อนักศึกษาร้องขอ
4. ปรับปรุงข้อมูลอาจารย์ เมื่อรับอาจารย์ใหม่ ลาออก หรือแต่งงาน

ระบบงานประจำภาคเรียน

1. พิมพ์รายชื่อแต่ละวิชาติดหน้าห้องสอบ และเลขที่โต๊ะของนักศึกษา
2. พิมพ์รายชื่อแยกตามสถานภาพ
3. พิมพ์รายชื่อแยกตามสาขา
4. การส่งเกรดของอาจารย์ เมื่อสิ้นภาคเรียน
5. พิมพ์ใบรายงานผลการเรียนประจำภาค

ระบบปรับปรุงและพิมพ์ข้อมูลจากแฟ้ม

1. แฟ้มนักศึกษา
2. แฟ้มวิชา
3. แฟ้มผู้สอน
4. แฟ้มลงทะเบียน
5. แฟ้มห้อง

5.3.2 ออกแบบเมนูตามระบบย่อย

เมนูหลัก

1. ระบบเตรียมงานและงานลงทะเบียน
2. ระบบงานหลังลงทะเบียน
3. ระบบงานเมื่อถูกร้องขอ
4. ระบบงานประจำภาคเรียน
5. ระบบปรับปรุงและพิมพ์ข้อมูลจากแฟ้ม
6. ข้อมูลระบบ และทีมพัฒนาโปรแกรม
7. ออกไประบบปฏิบัติการชั่วคราว
8. เลิกการทำงาน

1. เมนูระบบเตรียมงานและงานลงทะเบียน

   1. รับข้อมูลการเปิดวิชาใหม่
   2. รับข้อมูลข้อมูลนักศึกษาใหม่
   3. รับลงทะเบียนจากนักศึกษา
   4. รับบอกเพิ่ม-ลดวิชา
   5. กลับเมนูหลัก

2. เมนูระบบงานหลังลงทะเบียน

   1. พิมพ์ข้อมูลการลงทะเบียนติดประกาศ
   2. พิมพ์รายชื่อนักเรียนให้อาจารย์ผู้สอน
   3. พิมพ์สรุปการลงทะเบียนแต่ละวิชาให้ผู้บริหาร
   4. พิมพ์รายชื่อวิชาที่ห้องเรียนแต่ละห้องถูกใช้
   5. พิมพ์รายชื่อวิชาที่อาจารย์ผู้สอนแต่ละท่าน
   6. พิมพ์สรุปจำนวนนักศึกษาแต่ละวิชา
   7. กลับเมนูหลัก

3. เมนูระบบงานเมื่อถูกร้องขอ

   1. การขอเพิกถอนวิชา
   2. ตรวจสอบตารางเรียนของนักศึกษาเฉพาะคน
   3. พิมพ์รายงานผลการเรียนตลอดหลักสูตร
   4. ปรับปรุงข้อมูลอาจารย์
   5. กลับเมนูหลัก

4. เมนูระบบงานประจำภาคเรียน

1. พิมพ์รายชื่อติดหน้าห้องสอบ
2. พิมพ์รายชื่อแยกตามสถานภาพ
3. พิมพ์รายชื่อแยกตามสาขา
4. ใส่เกรด
5. พิมพ์รายงานผลการเรียนประจำภาค
6. กลับเมนูหลัก

5. เมนูระบบปรับปรุงและพิมพ์ข้อมูลจากแฟ้ม

1. แฟ้มนักศึกษา
2. แฟ้มวิชา
3. แฟ้มผู้สอน
4. แฟ้มลงทะเบียน
5. แฟ้มห้อง
6. กลับเมนูหลัก

5.3.3 หลักการตั้งชื่อโปรแกรม

การตั้งชื่อโปรแกรมมีหลายวิธี และแต่ละวิธีมีจุดเด่น และจุดด้อยที่แตกต่างกัน ขึ้นกับลักษณะหน่วยงาน ลักษณะบุคลากร และข้อตกลงของทีมพัฒนาโปรแกรม

หลักในการตั้งชื่อมีหลายวิธี

1. ตั้งชื่อตามลักษณะงานที่ใช้
2. ตั้งชื่อตามหน่วยงานที่เกี่ยวข้อง และข้อมูลที่เกี่ยวข้อง
3. ตั้งชื่อตามลำดับในเมนู
4. ตั้งชื่อตามผู้รับผิดชอบโปรแกรม
5. ตั้งชื่อตามวันที่ที่เริ่มเขียนโปรแกรม
6. ตั้งชื่อตามลำดับของโปรแกรม
7. อื่น ๆ อีกมาก

สำหรับระบบงานทะเบียน ขอตั้งชื่อตามลำดับของตัวเลือกในเมนู เพื่อสื่อให้เข้าใจได้โดยง่าย สำหรับชื่อของโปรแกรมกำหนดไว้ 5 หลัก โดยแต่ละหลักมีความหมายดังนี้

หลักที่ 1 ให้เป็นตัว R ซึ่งย่อมาจาก REGISTRA

หลักที่ 2 และ 3 เป็นลำดับในเมนูหลัก

หลักที่ 4 และ 5 เป็นลำดับในเมนูย่อย

โปรแกรมต่าง ๆ ในระบบทะเบียน

MAIN.PRG : เมนูหลักสำหรับระบบงานทะเบียน เพื่อทำตัวเลือกหลักทั้งหมด ประกอบด้วย 21 โปรแกรมย่อย

MAINALL : แสดงเมนูหลักบนจอภาพ

SUB1 : แสดงเมนูย่อย ระบบเตรียมงาน และงานลงทะเบียน

SUB2 : แสดงเมนูย่อย ระบบงานหลังลงทะเบียน

SUB3 : แสดงเมนูย่อย ระบบงานเมื่อถูกร้องขอ

SUB4 : แสดงเมนูย่อย ระบบงานประจำภาคเรียน

SUB5 : แสดงเมนูย่อย ระบบปรับปรุง และพิมพ์ข้อมูลจากแฟ้ม

WORKSYSTEM : แสดงข้อมูลของระบบทางจอภาพ

ASKPRT : สอบถามการปลายทางของผลลัพธ์

ASKUPD : ถามย้ำการปรับปรุงข้อมูล

SORTREG1 : จัดเรียงแฟ้ม REGIST ตามฟิลด์ RYEAR,RSEM,IDSTD,SUBJ

SORTREG2 : จัดเรียงแฟ้ม REGIST ตามฟิลด์ SUBJ,RYEAR,RSEM,IDSTD

SORTSUB1 : จัดเรียงแฟ้ม SUBJECT ตามฟิลด์ ROOM,SUBJ

SORTSUB2 : จัดเรียงแฟ้ม SUBJECT ตามฟิลด์ SUBJ

SORTID1 : จัดเรียงแฟ้ม IDSTD ตาม IDSTD

SORTID2 : จัดเรียงแฟ้ม IDSTD ตาม ISTAT,IDSTD

SORTID3 : จัดเรียงแฟ้ม IDSTD ตาม IMAJOR,INAME,ISURN

GRADECHR : แปลงเกรดจากตัวเลขเป็นตัวอักษร

GRADENUM : แปลงเกรดจากตัวอักษรเป็นตัวเลข

HELPFORM : แสดงข้อความช่วยเหลืออย่างง่าย ๆ เมื่อกดปุ่มฟังก์ชัน F1

HELPREADVAR : แสดงค่าที่เป็นไปได้ของฟิลด์ที่กำลังรอรับค่า

QUITPROC : สั่งเลิกทำงานของโปรแกรมทั้งหมด

SUB01.PRG : ระบบเตรียมงานและลงทะเบียน ประกอบด้วย 4 โปรแกรมย่อย

R0101 : ปรับปรุงแฟ้มวิชา ด้วยข้อมูลการเปิดวิชา จากอาจารย์ผู้สอน

R0102 : เพิ่มรายชื่อนักศึกษาเข้าใหม่ในแฟ้มนักศึกษา

R0103 : เพิ่มข้อมูลการลงทะเบียนในแฟ้มลงทะเบียน

R0104 : ปรับปรุงข้อมูลการลงทะเบียนในแฟ้มลงทะเบียน

SUB02.PRG : ระบบงานหลังลงทะเบียน ประกอบด้วย 6 โปรแกรมย่อย

R0201 : พิมพ์ข้อมูลการลงทะเบียนติดประกาศ

R0202 : พิมพ์รายชื่อนักเรียนให้อาจารย์ผู้สอน

R0203 : พิมพ์สรุปการลงทะเบียนแต่ละวิชาให้ผู้บริหาร

R0204 : พิมพ์รายชื่อวิชาที่ห้องเรียนแต่ละห้องถูกใช้

R0205 : พิมพ์รายชื่อวิชาที่อาจารย์ผู้สอนแต่ละท่านรับผิดชอบ

R0206 : พิมพ์สรุปจำนวนนักศึกษาแต่ละวิชา

SUB03.PRG : ระบบงานเมื่อถูกร้องขอ ประกอบด้วย 4 โปรแกรมย่อย

R0301 : ปรับปรุงเกรดนักศึกษาที่ขอเพิกถอนวิชา

R0302 : พิมพ์ตารางเรียนของนักศึกษาแต่ละวิชา

R0303 : พิมพ์ใบรายงานผลการเรียนตลอดหลักสูตร

R0304 : ปรับปรุงแฟ้มอาจารย์

SUB04.PRG : ระบบงานประจำภาคเรียน ประกอบด้วย 5 โปรแกรมย่อย

R0401 : พิมพ์รายชื่อนักศึกษาแต่ละวิชาติดหน้าห้องสอบ

R0402 : พิมพ์รายชื่อแยกตามสถานภาพ

R0403 : พิมพ์รายชื่อแยกตามสาขา และจัดเรียงตามชื่อ

R0404 : ปรับปรุงเกรดนักศึกษา เมื่ออาจารย์ส่งเกรด

R0405 : พิมพ์ใบรายงานผลการเรียนประจำภาค

SUB05.PRG : ระบบปรับปรุงและพิมพ์ข้อมูลจากแฟ้ม ประกอบด้วย 5 โปรแกรมย่อย

R0501 : เพิ่ม ลบ แก้ไขหรือพิมพ์ข้อมูลจาก แฟ้มนักศึกษา

R0502 : เพิ่ม ลบ แก้ไขหรือพิมพ์ข้อมูลจาก แฟ้มวิชา

R0503 : เพิ่ม ลบ แก้ไขหรือพิมพ์ข้อมูลจาก แฟ้มผู้สอน

R0504 : เพิ่ม ลบ แก้ไขหรือพิมพ์ข้อมูลจาก แฟ้มลงทะเบียน

R0505 : เพิ่ม ลบ แก้ไขหรือพิมพ์ข้อมูลจาก แฟ้มห้อง

5.3.4 วิธีการแปลโปรแกรม

การแปลโปรแกรม ในกรณีศึกษานี้ ใช้คำสั่ง RMAKE ในการแปลโปรแกรม โดยระบุชื่อโปรแกรมที่จะแปลไว้ในแฟ้ม MAIN.RMK เพราะการแปลแบบนี้จะทำให้การพัฒนาเร็วขึ้น หากมีการเรียกโปรแกรมย่อยจากภายนอกหลาย ๆ โปรแกรม เมื่อแก้โปรแกรมย่อยบางโปรแกรมจะต้องแปลโปรแกรมทั้งหมดใหม่ด้วยการแปลวิธีเดิม แต่การแปลด้วย RMAKE จะเลือกเฉพาะโปรแกรมย่อยที่ถูกแก้ไขมารับการแปลใหม่เท่านั้น

จุดเด่นของการใช้ RMAKE.EXE ช่วยคือ การแปลโปรแกรมแต่ละครั้งจะเลือกเฉพาะโปรแกรมย่อยที่ถูกปรับปรุงมาแปลใหม่ ส่วนที่ไม่ได้แก้ไขจะใช้ของแฟ้มเดิม ทำให้เร็วขึ้นอย่างมากในการพัฒนาโปรแกรมที่มีการเรียกใช้โปรแกรมย่อยจากภายนอกหลาย ๆ โปรแกรม

การสร้างแฟ้ม MAIN.RMK

```
// MAIN.RMK
// ========
.PRG.OBJ:
    CLIPPER $< /M /N
.OBJ.EXE:
    SET RTLINKCMD=/POSI
    RTLINK $**, $@ ;
MAIN.OBJ:    MAIN.PRG
SUB01.OBJ:   SUB01.PRG
SUB02.OBJ:   SUB02.PRG
SUB03.OBJ:   SUB03.PRG
SUB04.OBJ:   SUB04.PRG
SUB05.OBJ:   SUB05.PRG
MAIN.EXE: MAIN.OBJ  SUB01.OBJ SUB02.OBJ SUB03.OBJ SUB04.OBJ SUB05.OBJ
```

การแปลโปรแกรมจะนำ MAIN.RMK มารับการแปลด้วย RMAKE โดยเขียนที่ DOS PROMPT ดังนี้ C:\CLIPPER5\RMAKE MAIN แต่ยังคงใช้แฟ้ม CLIPPER.EXE และ RTLINK.EXE เหมือนการแปลด้วย C:\CLIPPER5\CL MAIN

สิ่งที่แตกต่างของ RMAKE และ CL คือ RMAKE.EXE จะช่วยเลือกโปรแกรมย่อยที่มีการแก้ไขเท่านั้นมาผ่านการแปลด้วยโปรแกรมชุดเดิม แต่ไม่ได้นำโปรแกรมย่อยทั้งหมดมาแปลใหม่

การเขียนโปรแกรมให้ RMAKE แปล ต้องเขียนทุกโปรแกรมให้อยู่ใน PROCEDURE หรือ FUNCTION ถึงเป็นโปรแกรมหลักก็ต้องมีโปรแกรมแรกอยู่ใน PROCEDURE แรกของโปรแกรม

& 5.4 โปรแกรมในระบบทะเบียน

โปรแกรมของระบบนี้ ได้จัดทำเป็นโปรแกรมย่อยไว้ภายนอก โดยมีโปรแกรม MAIN.PRG คอยเรียกโปรแกรมเล็ก ๆ ที่อยู่ในโปรแกรมย่อยอีกครั้งหนึ่ง สำหรับโปรแกรมย่อยภายนอกนั้น แต่ละโปรแกรมย่อยคือระบบย่อยระบบหนึ่ง เพราะทำให้ง่ายต่อการปรับปรุงแก้ไขและค้นหา หากประมวลผลตามตัวเลือกแล้วผิดพลาด

โปรแกรมย่อยจากภายนอกมี 5 โปรแกรมคือ SUB01.PRG จนถึง SUB05.PRG รวมโปรแกรมทั้งหมดในระบบนี้เป็น 6 โปรแกรม โดยให้โปรแกรมย่อยทั้ง 5 นี้ควบคุมระบบย่อย 5 ระบบ

1. ระบบเตรียมงาน และงานลงทะเบียน
2. ระบบงานหลังลงทะเบียน
3. ระบบงานเมื่อถูกร้องขอ
4. ระบบงานประจำภาคเรียน
5. ระบบปรับปรุงและพิมพ์ข้อมูลจากแฟ้ม

5.4.1 โปรแกรมต่าง ๆ ใน MAIN.PRG

โปรแกรมใน MAIN.PRG นี้มีโปรแกรมย่อยทั้งหมด 21 โปรแกรม โดยรวมโปรแกรมที่เกี่ยวกับเมนู และโปรแกรมสนับสนุน เช่น โปรแกรมสอบถามปลายทางของผลลัพธ์ โปรแกรมถามย้ำการปรับปรุงข้อมูล โปรแกรมจัดเรียงข้อมูล โปรแกรมแปลงเกรด โปรแกรมสั่งเลิกการทำงาน เป็นต้น

โปรแกรมเมนูในโปรแกรม MAIN.PRG นี้มีหลักการง่าย ๆ โดยสามารถเลือกตัวเลือกด้วยคำสั่ง PROMPT และส่งการทำงานไปเมนูย่อย ในขณะที่เมนูย่อยแต่ละเมนูมีระบบการทำงานที่สมบูรณ์ในตนเอง จึงทำให้เมนูแบบนี้ทำความเข้าใจได้ง่าย และนำไปใช้ในทางปฏิบัติได้ทันที

ในหัวข้อนี้จะแสดงตัวอย่างโปรแกรมต่าง ๆ ที่มีในโปรแกรม MAIN.PRG ทั้งหมด 21 โปรแกรม โดยเริ่มจากตัวอย่างที่ 5.1 ถึง ตัวอย่างที่ 5.14 (บางโปรแกรมสั้นและมีการทำงานคล้ายกัน จึงนำมารวมในตัวอย่างเดียวกัน)

: ตัวอย่างที่ 5.1

& ผลลัพธ์

```
// โปรแกรมหลัก ของระบบงานทะเบียน เพื่อจัดทำเมนูหลัก แล้วเลือกตัวเลือก
// เพื่อเข้าไปสู่เมนูย่อย แต่มีการระบุโปรแกรมย่อยให้กับปุ่มฟังก์ชัน โดยระบุให้
// ปุ่มฟังก์ชัน F1 เรียกโปรแกรมย่อย HELPFORM เพื่อแสดงข้อมูลผู้เขียน
// ปุ่มฟังก์ชัน F2 เรียกโปรแกรมย่อย HELPREADVAR เพื่อช่วยผู้ใช้ป้อนค่าได้ง่าย
// ปุ่มฟังก์ชัน F10 เรียกโปรแกรมย่อย QUITPROC เพื่อเลิกการทำงาน
PROCEDURE MAINALL
  SET KEY 28 TO HELPFORM          // ควบคุมปุ่ม F1 เพื่อแสดงข้อมูลเกี่ยวกับผู้เขียนโปรแกรม
  SET KEY -1 TO HELPREADVAR       // ควบคุมปุ่ม F2 เพื่อเลือกข้อมูลได้อย่างง่าย ๆ
  SET KEY -9 TO QUITPROC          // ควบคุมปุ่ม F10 เพื่อถามการเลิกทำงาน
  SET WRAP ON
  SET DELIMITERS TO '[ ]'
```

```
  SET DELIMITERS ON
  SET EPOCH TO 1960
  DO WHILE .T.
    CLS ; OPT = 1
    @ 8,5 SAY "เมนูหลัก ระบบงานทะเบียน"
    @ 9,5 SAY "===================="
    @ 10,5 PROMPT "1. ระบบเตรียมงานและงานลงทะเบียน"
    @ 11,5 PROMPT "2. ระบบงานหลังลงทะเบียน"
    @ 12,5 PROMPT "3. ระบบงานเมื่อถูกร้องขอ"
    @ 13,5 PROMPT "4. ระบบงานประจำภาคเรียน"
    @ 14,5 PROMPT "5. ระบบปรับปรุงและพิมพ์ข้อมูลจากแฟ้ม"
    @ 15,5 PROMPT "6. ข้อมูลระบบ และทีมพัฒนาโปรแกรม"
    @ 16,5 PROMPT "7. ออกไปสู่ระบบปฏิบัติการชั่วคราว"
    @ 17,5 PROMPT "8. เลิกการทำงาน"
    MENU TO OPT
    CLS
    DO CASE
      CASE OPT >= 1 .AND. OPT <= 5 ;  DO &("SUB"+LTRIM(STR(OPT)))
      CASE OPT = 6 ;  DO WORKSYSTEM
      CASE OPT = 7
         IF MEMORY(2) > 400  ;  RUN COMMAND
         ELSE ; WAIT "หน่วยความจำไม่พอ จึงออกไประบบปฏิบัติการชั่วคราวไม่ได้"
         ENDIF
      CASE OPT = 8 .OR. OPT = 0 ;  EXIT
    ENDCASE
  ENDDO
RETURN
```

: ตัวอย่างที่ 5.2

& ผลลัพธ์

```
// สร้างเมนูย่อยสำหรับระบบเตรียมงานและงานลงทะเบียน เพื่อเลือกโปรแกรมต่าง ๆ มาทำงาน
PROCEDURE SUB1
  DO WHILE LASTKEY() != 27
    CLS
    @ 8,5 SAY "1. เมนูระบบเตรียมงานและงานลงทะเบียน"
```

```
    @ 9,5 SAY "====================="
    @ 10,5 PROMPT "1. รับข้อมูลการเปิดวิชาใหม่"
    @ 11,5 PROMPT "2. รับข้อมูลข้อมูลนักศึกษาใหม่"
    @ 12,5 PROMPT "3. รับลงทะเบียนจากนักศึกษา"
    @ 13,5 PROMPT "4. รับบอกเพิ่ม-ลดวิชา"
    @ 14,5 PROMPT "5. กลับเมนูหลัก"
    MENU TO OPT
    CLS
    DO CASE
      CASE OPT >= 1 .AND. OPT <=4 ;  DO &("R010"+LTRIM(STR(OPT)))
      CASE OPT = 5 ;  EXIT
    ENDCASE
  ENDDO
RETURN
```

: ตัวอย่างที่ 5.3

& ผลลัพธ์

```
// สร้างเมนูย่อยสำหรับระบบงานหลังลงทะเบียน เพื่อเลือกโปรแกรมต่าง ๆ มาทำงาน
PROCEDURE SUB2
  DO WHILE LASTKEY() != 27
    CLS
    @ 8,5 SAY "2. เมนูระบบงานหลังลงทะเบียน"
    @ 9,5 SAY "====================="
    @ 10,5 PROMPT "1. พิมพ์ข้อมูลการลงทะเบียนติดประกาศ"
    @ 11,5 PROMPT "2. พิมพ์รายชื่อนักเรียนให้อาจารย์ผู้สอน"
    @ 12,5 PROMPT "3. พิมพ์สรุปการลงทะเบียนแต่ละวิชาให้ผู้บริหาร"
    @ 13,5 PROMPT "4. พิมพ์รายชื่อวิชาที่ห้องเรียนแต่ละห้องถูกใช้"
    @ 14,5 PROMPT "5. พิมพ์รายชื่อวิชาที่อาจารย์ผู้สอนรับผิดชอบ"
    @ 15,5 PROMPT "6. พิมพ์สรุปจำนวนนักศึกษาแต่ละวิชา"
    @ 16,5 PROMPT "7. กลับเมนูหลัก"
    MENU TO OPT
    CLS
    DO CASE
      CASE OPT >= 1 .AND. OPT <=6 ;  DO &("R020"+LTRIM(STR(OPT)))
      CASE OPT = 7 ;  EXIT
```

```
    ENDCASE
  ENDDO
RETURN
```

: ตัวอย่างที่ 5.4

& ผลลัพธ์

```
// สร้างเมนูย่อยสำหรับระบบเมื่อถูกร้องขอ เพื่อเลือกโปรแกรมต่าง ๆ มาทำงาน
PROCEDURE SUB3
  DO WHILE LASTKEY() != 27
    CLS
    @ 8,5 SAY "3. เมนูระบบเมื่อถูกร้องขอ"
    @ 9,5 SAY "====================="
    @ 10,5 PROMPT "1. การขอเพิกถอนวิชา"
    @ 11,5 PROMPT "2. ตรวจสอบตารางเรียนของนักศึกษาเฉพาะคน"
    @ 12,5 PROMPT "3. พิมพ์รายงานผลการเรียนตลอดหลักสูตร"
    @ 13,5 PROMPT "4. ปรับปรุงข้อมูลอาจารย์"
    @ 14,5 PROMPT "5. กลับเมนูหลัก"
    MENU TO OPT
    CLS
    DO CASE
      CASE OPT >= 1 .AND. OPT <=4 ;  DO &("R030"+LTRIM(STR(OPT)))
      CASE OPT = 5 ;  EXIT
    ENDCASE
  ENDDO
RETURN
```

: ตัวอย่างที่ 5.5

& ผลลัพธ์

```
// สร้างเมนูย่อยสำหรับระบบงานประจำภาคเรียน เพื่อเลือกโปรแกรมต่าง ๆ มาทำงาน
PROCEDURE SUB4
  DO WHILE LASTKEY() != 27
    CLS
    @ 8,5 SAY "4. เมนูระบบงานประจำภาคเรียน"
    @ 9,5 SAY "====================="
    @ 10,5 PROMPT "1. พิมพ์รายชื่อติดหน้าห้องสอบ"
    @ 11,5 PROMPT "2. พิมพ์รายชื่อแยกตามสถานภาพ"
```

```
    @ 12,5 PROMPT "3. พิมพ์รายชื่อแยกตามสาขา"
    @ 13,5 PROMPT "4. ใส่เกรด"
    @ 14,5 PROMPT "5. พิมพ์รายงานผลการเรียนประจำภาค"
    @ 15,5 PROMPT "6. กลับเมนูหลัก"
    MENU TO OPT
    CLS
    DO CASE
      CASE OPT >= 1 .AND. OPT <=5 ;  DO &("R040"+LTRIM(STR(OPT)))
      CASE OPT = 6 ;  EXIT
    ENDCASE
  ENDDO
RETURN
```

: ตัวอย่างที่ 5.6

& ผลลัพธ์

```
// สร้างเมนูย่อยสำหรับระบบปรับปรุง เพื่อเลือกโปรแกรมต่าง ๆ มาทำงาน
PROCEDURE SUB5
  DO WHILE LASTKEY() != 27
    CLS
    @ 8,5 SAY "5. เมนูระบบปรับปรุงและพิมพ์ข้อมูลจากแฟ้ม"
    @ 9,5 SAY "=================="
    @ 10,5 PROMPT "1. แฟ้มนักศึกษา"
    @ 11,5 PROMPT "2. แฟ้มวิชา"
    @ 12,5 PROMPT "3. แฟ้มผู้สอน"
    @ 13,5 PROMPT "4. แฟ้มลงทะเบียน"
    @ 14,5 PROMPT "5. แฟ้มห้อง"
    @ 15,5 PROMPT "6. กลับเมนูหลัก"
    MENU TO OPT
    CLS
    DO CASE
      CASE OPT >= 1 .AND. OPT <=5 ;  DO &("R050"+LTRIM(STR(OPT)))
      CASE OPT = 6 ;  EXIT
    ENDCASE
  ENDDO
RETURN
```

: ตัวอย่างที่ 5.7

& ผลลัพธ์

```
// พิมพ์ข้อมูลของระบบทางจอภาพ เมื่อเลือกตัวเลือกที่ 6 จากเมนูหลัก
// โดยแสดงวันที่ เวลา ชื่อระบบปฏิบัติการ หน่วยความจำที่เหลือ
// รุ่นที่ใช้พัฒนาโปรแกรม สถานะของเครื่องพิมพ์ ตลอดจนข้อมูลของ
// ผู้พัฒนาโปรแกรม และสถานที่ติดต่อ
PROCEDURE WORKSYSTEM
  CLS
  ? "              วันที่ :",DTOC(DATE())
  ? "              เวลา :",TIME()
  ? "              วินาที :",SECOND()
  ? "     ระบบปฏิบัติการที่ใช้ :",OS()
  ? "ขนาดหน่วยความจำที่เหลือ :",MEMORY(0)
  ? "         รุ่นของภาษา :",VERSION()
  ? " เครื่องพิมพ์เปิดหรือปิดอยู่ :",ISPRINTER()
  ? "====================================="
  ? "โปรแกรมนี้ถูกพัฒนาโดย. อ.บุรินทร์ รุจจนพันธุ์"
  ? "เริ่มพัฒนาเมื่อ 8 กุมภาพันธ์ 2545"
  ? "โปรแกรมนี้เป็นรุ่นที่ 2.200202"
  ? "หากสนใจ หรือต้องการตัวโปรแกรม ติดต่อขอรับฟรีได้ที่"
  ? "วิทยาลัยโยนก ลำปาง 52000"
  ? "TEL.(054)265170 EXT.210   FAX. (054)265184"
  INKEY(2)
RETURN
```

: ตัวอย่างที่ 5.8

& ผลลัพธ์

```
// ตัวอย่างนี้มี 2 ฟังก์ชันคือ ฟังก์ชันเพื่อเลือกปลายทางของผลลัพธ์
// และฟังก์ชันถามย้ำการปรับปรุงข้อมูล เมื่อมีการแก้ไขข้อมูล
// ใช้ถามว่าจะพิมพ์ผลลัพธ์ออกทางใด
FUNCTION ASKPRT()
  ASKPRT = ALERT("ผลลัพธ์ของรายงาน",{'จอภาพ','เครื่องพิมพ์'})
RETURN (ASKPRT)
// ใช้ถามว่าจะยอมรับการปรับปรุงข้อมูลหรือไม่
FUNCTION ASKUPD()
  ASKUPD = ALERT("การปรับปรุงข้อมูล",{'ยอมรับการปรับปรุง','ยกเลิกการปรับปรุง'})
RETURN (ASKUPD)
```

: ตัวอย่างที่ 5.9

& ผลลัพธ์

```
// เนื่องจากระบบนี้ไม่มีการใช้แฟ้มดรรชนี แต่ต้องการการเรียงข้อมูลให้ถูกต้อง
// จึงจัดทำโปรแกรมย่อย เพื่อเรียงข้อมูลตามฟิลด์ที่ต้องการ โดยใช้คำสั่งจัดเรียง
// ทำให้ผลการจัดเรียงถูกเก็บในแฟ้มใหม่ แล้วจึงคัดลอกกลับมาทับแฟ้มเดิม
// ใช้จัดเรียงแฟ้ม REGIST ตามปี ภาค รหัส วิชา
PROCEDURE SORTREG1
  USE REGIST
  SORT ON RYEAR,RSEM,IDSTD,SUBJ TO TEMPF
  USE TEMPF ; COPY TO REGIST ; CLOSE
RETURN
// ใช้จัดเรียงแฟ้ม REGIST ตามวิชา ปี ภาค และรหัส
PROCEDURE SORTREG2
  USE REGIST
  SORT ON SUBJ,RYEAR,RSEM,IDSTD TO TEMPF
  USE TEMPF ; COPY TO REGIST ; CLOSE
RETURN
// ใช้จัดเรียงแฟ้ม SUBJECT ตามห้องเรียน และชื่อวิชา
PROCEDURE SORTSUB1
  USE SUBJECT
  SORT ON ROOM,SUBJ TO  TEMPF
  USE TEMPF ; COPY TO SUBJECT ; CLOSE
RETURN
```

```
// ใช้จัดเรียงแฟ้ม SUBJECT ตามชื่อวิชา
PROCEDURE SORTSUB2
  USE SUBJECT
  SORT ON SUBJ TO  TEMPF
  USE TEMPF ; COPY TO SUBJECT ; CLOSE
RETURN
// ใช้จัดเรียงแฟ้ม IDSTD ตามชื่อรหัส
PROCEDURE SORTID1
  USE IDSTD
  SORT ON IDSTD TO TEMPF
  USE TEMPF ; COPY TO IDSTD ; CLOSE
RETURN
// ใช้จัดเรียงแฟ้ม IDSTD ตามชื่อสถานภาพ และรหัส
PROCEDURE SORTID2
  USE IDSTD
  SORT ON ISTAT,IDSTD TO  TEMPF
  USE TEMPF ; COPY TO IDSTD ; CLOSE
RETURN
// ใช้จัดเรียงแฟ้ม IDSTD ตามชื่อสาขา และชื่อ
PROCEDURE SORTID3
  USE IDSTD
  SORT ON IMAJOR,INAME,ISURN TO  TEMPF
  USE TEMPF ; COPY TO IDSTD ; CLOSE
RETURN
```

: ตัวอย่างที่ 5.10

& ผลลัพธ์

```
// ฟังก์ชันเปลี่ยนเกรด จากตัวเลข เป็นตัวอักษร นำไปแสดงผลได้ชัดเจนขึ้น
FUNCTION  GRADECHR(RGRADE)
  GRADECHR = '--'
  DO CASE
    CASE RGRADE = 4   ;  GRADECHR = 'A '
    CASE RGRADE = 3.5 ;  GRADECHR = 'B+'
    CASE RGRADE = 3   ;  GRADECHR = 'B '
    CASE RGRADE = 2.5 ;  GRADECHR = 'C+'
```

```
    CASE RGRADE = 2   ;  GRADECHR = 'C '
    CASE RGRADE = 1.5 ;  GRADECHR = 'D+'
    CASE RGRADE = 1   ;  GRADECHR = 'D '
    CASE RGRADE = 0   ;  GRADECHR = 'F '
    CASE RGRADE = 4.2 ;  GRADECHR = 'W '
    CASE RGRADE = 4.6 ;  GRADECHR = 'P '
  ENDCASE
RETURN (GRADECHR)
```

: ตัวอย่างที่ 5.11

& ผลลัพธ์

```
// ฟังก์ชันปลี่ยนเกรด จากตัวอักษรเป็นตัวเลข ทำให้จัดเก็บลงแฟ้มได้ง่ายขึ้น
FUNCTION  GRADENUM(_GRADE)
  GRADENUM = 0
  DO CASE
    CASE UPPER(_GRADE) = 'A'  ; GRADENUM = 4
    CASE UPPER(_GRADE) = 'B+' ; GRADENUM = 3.5
    CASE UPPER(_GRADE) = 'B'  ; GRADENUM = 3
    CASE UPPER(_GRADE) = 'C+' ; GRADENUM = 2.5
    CASE UPPER(_GRADE) = 'C'  ; GRADENUM = 2
    CASE UPPER(_GRADE) = 'D+' ; GRADENUM = 1.5
    CASE UPPER(_GRADE) = 'D'  ; GRADENUM = 1
    CASE UPPER(_GRADE) = 'F'  ; GRADENUM = 0
    CASE UPPER(_GRADE) = 'W'  ; GRADENUM = 4.2
    CASE UPPER(_GRADE) = 'P'  ; GRADENUM = 4.6
  ENDCASE
RETURN (GRADENUM)
```

: ตัวอย่างที่ 5.12

& ผลลัพธ์

```
// แสดงข้อความง่าย ๆ ทันที เมื่อกดปุ่มฟังก์ชัน F1 โดยไม่ทำให้ภาพเดิมบนจอเปลี่ยน
// เพราะมีการใช้คำสั่งเก็บสถานะของจอ และคืนสถานะเดิมของจอ และที่ใช้ F1 ได้
// เพราะ ระบุในโปรแกรมหลัก MAINALL ว่า SET KEY 28 TO HELPFORM
// แสดงข้อมูลของผู้พัฒนาโปรแกรมให้ทราบ
```

```
PROCEDURE HELPFORM
  SAVE SCREEN TO SCR
  @ 5,5 CLEAR TO 20,75                    // ลบข้อความบนพื้นที่กำหนด
  @ 9,8 TO 15,71                          // ตีกรอบรอบข้อความ
  @ 10,10 SAY "CLIPPER PROGRAMING"
  @ 11,10 SAY "LEARING BY CASE STUDY"
  @ 12,10 SAY "GOOD LUCK"
  @ 13,10 SAY DATE()
  @ 14,10 SAY TIME()
  INKEY(5)
  RESTORE SCREEN FROM SCR
RETURN
```

: ตัวอย่างที่ 5.13

& ผลลัพธ์

```
// ฟังก์ชันนี้ช่วยให้การใส่ค่าในตัวแปรบางตัวแปรถูกต้อง เพราะจะแสดง
// ค่าที่เหมาะสมกับฟิลด์นั้นมาให้เลือก โดยการอ่านค่าจากแฟ้มมาแสดงให้
// ดู แต่จะต้องระบุให้ชัดเจนว่าจะนำฟิลด์ใดมาใช้บ้าง
// ฟังก์ชันนี้จึงแสดงข้อมูลที่เป็นไปได้สำหรับตัวแปรที่กำลังรอรับค่า
FUNCTION HELPREADVAR
  OLDALIAS = ALIAS()                      // เก็บพื้นที่ทำงานเดิม
  SHOWAR = {}
  DO CASE
    CASE READVAR() = "_SUBJ"
       IF OLDALIAS != "SUBJECT"
         USE SUBJECT NEW                  // ถ้าเปิดแล้วต้องไม่เปิดอีก
       ENDIF
       GO TOP
       WHILE !EOF()
         AADD(SHOWAR,SUBJ)
         SKIP
       END
       IND := ACHOICE(2,71,5,78,SHOWAR)
       IF IND > 0                         // ป้องกันการใช้ปุ่ม ESC
         _SUBJ = SHOWAR[IND]
```

```
      ENDIF
      IF OLDALIAS != "SUBJECT"          // เพราะอนุญาติให้เรียก F2 2 ครั้ง
        CLOSE                           // ในจุดที่ไม่ได้เปิดแฟ้มนี้
      ENDIF
    CASE READVAR() = "_IDSTD"
      IF OLDALIAS != "IDSTD"
        USE IDSTD NEW                   // ถ้าเปิดแล้วต้องไม่เปิดอีก
      ENDIF
      GO TOP
      FOR I = 1 TO  RECCOUNT()
        AADD(SHOWAR,STR(I,2)+". "+STR(IDSTD)+" : "+INAME)
        SKIP
      NEXT
      IND := ACHOICE(2,50,10,78,SHOWAR)
      IF IND > 0                        // ป้องกันการใช้ปุ่ม ESC
        _IDSTD = VAL(SUBSTR(SHOWAR[IND],5,7))
      ENDIF
      IF OLDALIAS != "IDSTD"            // เพราะอนุญาติให้เรียก F2 2 ครั้ง
        CLOSE                           // ในจุดที่ไม่ได้เปิดแฟ้มนี้
      ENDIF
    OTHERWISE
      X = ALERT("ขอโทษ ไม่ได้เตรียมส่วนสนับสนุน สำหรับจุดนี้",{"OK."})
  ENDCASE
  IF LEN(OLDALIAS) > 0
    SELE &OLDALIAS                      // กลับสู่พื้นที่ทำงานเดิม
  ENDIF
RETURN
```

: ตัวอย่างที่ 5.14

& ผลลัพธ์

```
// สอบถามการเลิกทำงานด้วยฟังก์ชัน ALERT
PROCEDURE QUITPROC
  Q =  ALERT("สอบถามการเลิกการทำงาน",{"เลิกการทำงาน","ยังไม่เลิกงาน"})
  IF Q = 1
    QUIT
  ENDIF
RETURN
```

5.4.2 โปรแกรมต่าง ๆ ใน SUB01.PRG

โปรแกรมใน SUB01.PRG นี้มีโปรแกรมย่อยทั้งหมด 4 โปรแกรม โดยรวมโปรแกรมที่เกี่ยวกับระบบเตรียมงาน และงานลงทะเบียน เช่น โปรแกรมรับข้อมูลการลงทะเบียน โปรแกรมขอบอกเพิ่มลดวิชา โปรแกรมป้อนข้อมูลนักศึกษา หรือโปรแกรมปรับปรุงรายวิชาเป็นต้น

ระบบเตรียมงาน และงานลงทะเบียนเป็นระบบที่สำคัญยิ่ง เพราะระบบนี้ถูกพัฒนาขึ้นเพื่อเก็บประวัติการเรียน และโปรแกรมในระบบนี้จะบันทึกข้อมูลการลงทะเบียนของนักศึกษาแต่ละคน ทำให้เกิดข้อมูลขึ้น เสมือนว่าจุดนี้เป็นจุดเริ่มต้นที่ทำให้เกิดข้อมูล นำไปปรับปรุง นำไปใช้ และประมวลผลในงานของระบบอื่นได้อีกมาก

ในหัวข้อนี้จะแสดงตัวอย่างโปรแกรมต่าง ๆ ที่มีในโปรแกรม SUB01.PRG ทั้งหมด 4 โปรแกรม โดยเริ่มจากตัวอย่างที่ 5.15 ถึง ตัวอย่างที่ 5.18 (R0101 - R0104)

: ตัวอย่างที่ 5.15

& ผลลัพธ์

```
// ปรับปรุงข้อมูลในแฟ้มวิชาอย่างง่าย ๆ ด้วยฟังก์ชัน BROWSE
// แล้วสอบถามว่าจะปรับปรุงหรือไม่ ถ้าไม่จะยกเลิกการปรับปรุงทั้งหมด
PROCEDURE R0101
  USE SUBJECT
  COPY TO TEMPF
  BROWSE(5,5,22,75)
  IF ASKUPD() = 1
    PACK
  ELSE
    USE TEMPF
    COPY TO SUBJECT
  ENDIF
  CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.16

& ผลลัพธ์

```
// เพิ่มรายชื่อนักศึกษาใหม่ เพียงคนเดียว โดยระบุรหัสใหม่ให้อัตโนมัติ
PROCEDURE R0102
  USE IDSTD
  GOTO BOTTOM
  _IDSTD = IDSTD + 1
  _INAME = SPACE(20)
  _ISURN = SPACE(20)
  _IMAJOR = 0
  _ISTAT = 'A'
  @ 10,5 SAY "เพิ่มรายชื่อนักศึกษาใหม่"
  @ 11,5 SAY "รหัส"+STR(_IDSTD)
  @ 12,5 SAY "ชื่อ" GET _INAME
  @ 13,5 SAY "สกุล" GET _ISURN
  @ 14,5 SAY "สาขา" GET _IMAJOR
  @ 15,5 SAY "สถานภาพ" GET _ISTAT
  READ
  IF ASKUPD() = 1                          // สอบถามการปรับปรุง
    APPEND BLANK
    REPLACE IDSTD WITH _IDSTD ,INAME WITH _INAME ,ISURN WITH _ISURN,;
          IMAJOR WITH _IMAJOR,ISTAT WITH _ISTAT
  ENDIF
RETURN
```

: ตัวอย่างที่ 5.17

& ผลลัพธ์

```
// เพิ่มข้อมูลการลงทะเบียนของนักศึกษาแต่ละคน
// วิธีนี้จะลงทะเบียนและตัดยอดให้ทันที สำหรับแต่ละวิชา
// หากลง 5 วิชาแต่ผิดพลาด 2 วิชา จะเพิ่มข้อมูลเข้าไป 3 วิชาทันที
// การปรับปรุงจะเพิ่มจำนวนอีก 1 เข้าไปในฟิลด์จำนวนที่ลง ในแฟ้มวิชาแต่ละวิชา
// และเพิ่มเรคอร์ดเท่าจำนวนวิชาที่ลงทะเบียนได้ในแฟ้มลงทะเบียน
// พิมพ์ใบเสร็จโดยคิดค่าลงทะเบียนเรียนวิชาละ 1000 บาท
PROCEDURE R0103
  SET CENTURY ON
```

```
_RYEAR = IIF(MONTH(DATE())<5,YEAR(DATE())-1,YEAR(DATE()))
_RSEM = '1'
_IDSTD = 0
_SUBJ := ARRAY(8)
_RGRADE = 4.6
@ 4,5 SAY "โปรแกรม ลงทะเบียนเรียน"
@ 5,5 SAY "ประจำปีการศึกษา : " GET _RYEAR PICT '9999'
@ 6,5 SAY "   ภาคการศึกษา : " GET _RSEM
SAVE SCREEN TO SCR
DO WHILE LASTKEY() != 27
  AFILL(_SUBJ,SPACE(8))                    // กำหนดค่าเริ่มต้นให้ตัวแปรอาเรย์ _SUBJ ทุกสมาชิก
  _IDSTD = 0
  DO WHILE .T.                             // รับข้อมูลนักศึกษาที่มีในแฟ้ม IDSTD เท่านั้น
    USE IDSTD
    @ 7,5 SAY "    รหัสนักศึกษา : " GET _IDSTD PICT '9999999'
    READ
    LOCATE FOR IDSTD = _IDSTD
    IF FOUND()
      _INAME = INAME
      _ISURN = ISURN
      @ 8,5 SAY INAME+' '+ISURN+' '+STR(IMAJOR)+' '+ISTAT
      EXIT
    ELSE ; @ 8,5 SAY "ไม่มีรหัสนักศึกษานี้ในแฟ้ม IDSTD'
    ENDIF
  ENDDO
  @ 10,10 SAY "SUBJ1 : " GET _SUBJ[1] PICT '@!'
  @ 11,10 SAY "SUBJ2 : " GET _SUBJ[2] PICT '@!'
  @ 12,10 SAY "SUBJ3 : " GET _SUBJ[3] PICT '@!'
  @ 13,10 SAY "SUBJ4 : " GET _SUBJ[4] PICT '@!'
  @ 14,10 SAY "SUBJ5 : " GET _SUBJ[5] PICT '@!'
  @ 15,10 SAY "SUBJ6 : " GET _SUBJ[6] PICT '@!'
  @ 16,10 SAY "SUBJ7 : " GET _SUBJ[7] PICT '@!'
  @ 17,10 SAY "SUBJ8 : " GET _SUBJ[8] PICT '@!'
```

```
READ
YOURSUBJ = 8                        // จำเป็นต้อง 8 เพราะตรวจช่องว่างของวิชา
OKSUBJ = 0
USE SUBJECT
FOR I = 1 TO 8                      // นำวิชามาตรวจสอบในแฟ้มวิชา แล้วเพิ่มจำนวนเข้าไป
  IF _SUBJ[I] != SPACE(8)
    LOCATE FOR _SUBJ[I] = SUBJ
    IF FOUND()
      IF SREGIST >= SLIMIT
        @ 9+I,40 SAY 'วิชานี้เต็มแล้ว เพราะรับเพียง'+STR(SLIMIT)
      ELSE
        @ 9+I,40 SAY 'OK.'          // วิชาลงเพิ่ม จึงเพิ่มค่าในฟิลด์จำนวนที่ลง
        REPLACE SREGIST WITH SREGIST + 1
        USE REGIST                  // เพิ่มเรคอร์ดเข้าไปในแฟ้ม REGIST
          APPEND BLANK              // 4.6 คือเกรด P หมายถึงกำลังศึกษาอยู่
          REPLACE RYEAR WITH _RYEAR, RSEM WITH _RSEM ,;
            IDSTD WITH _IDSTD, SUBJ WITH _SUBJ[I],RGRADE WITH 4.6
        USE SUBJECT
        OKSUBJ = OKSUBJ + 1
      ENDIF
    ELSE
      @ 9+I,40 SAY 'ไม่มีวิชานี้ในแฟ้ม SUBJECT'
    ENDIF
  ELSE
    YOURSUBJ = I - 1
    I = 8
  ENDIF
NEXT
@ 18,10 SAY 'จำนวนวิชาที่ท่านลงทะเบียนเรียนคือ '+STR(YOURSUBJ)
@ 19,10 SAY 'จำนวนวิชาที่ท่านลงได้คือ '+STR(OKSUBJ)
INKEY(2)
// ส่วนนี้ใช้พิมพ์ใบเสร็จ โดยคิดวิชาละ 1000 บาท
IF OKSUBJ >= 1
```

```
      GASKPRT = ASKPRT()
      IF GASKPRT = 2 .AND. ISPRINTER()          // เลข 2 หมายถึงเครื่องพิมพ์
        SET DEVICE TO PRINTER
      ELSE ; CLS
      ENDIF
      @ 1,5  SAY "รหัส :"+STR(_IDSTD)
      @ 2,5  SAY "ชื่อ-สกุล :"+_INAME+" "+_ISURN
      @ 3,5  SAY "=========================="
      FOR I = 1 TO YOURSUBJ
        @ 3+I,5 SAY _SUBJ[I]
      NEXT
      @ 4+YOURSUBJ,5 SAY "จำนวนเงินทั้งหมดที่ท่านต้องชำระ"+STR(OKSUBJ*1000)
      IF GASKPRT = 2 .AND. ISPRINTER() ; EJECT ; ENDIF
      SET DEVICE TO SCREEN
    ENDIF
    INKEY(0)                                  // ขณะรอกดปุ่ม ถ้ากด ESC จะเลิกงานลงทะเบียน ถ้าไม่จะรับคนใหม่
    RESTORE SCREEN FROM SCR
  ENDDO
  SET CENTURY OFF
  CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.18

& ผลลัพธ์

```
// บอกเพิ่ม-ลดวิชา เพื่อปรับปรุงข้อมูลการลงทะเบียน
// การปรับปรุงสำหรับการบอกเพิ่มคือเพิ่มจำนวนในฟิลด์จำนวนที่ลง แต่บอกเลิกจะลดลง
// และจะเพิ่มหรือลดเรคอร์ดเท่าจำนวนที่ระบุในแฟ้มลงทะเบียน
// สุดท้ายพิมพ์ใบเสร็จ โดยคิดวิชาละ 1000 บาท และค่าธรรมเนียมวิชาละ 20 บาท
PROCEDURE R0104
  SET CENTURY ON
  _RYEAR = IIF(MONTH(DATE())<5,YEAR(DATE())-1,YEAR(DATE()))
  _RSEM = '1'
  _IDSTD = 0
  _SUBJ := ARRAY(8)
  @ 4,5 SAY "โปรแกรม บอกเพิ่ม-ลดวิชา"
```

```
@ 5,5 SAY "ประจำปีการศึกษา : " GET _RYEAR
@ 6,5 SAY "   ภาคการศึกษา : " GET _RSEM
SAVE SCREEN TO SCR
DO WHILE LASTKEY() != 27
  AFILL(_SUBJ,SPACE(8))                    // กำหนดค่าเริ่มต้นให้ตัวแปรอาเรย์ _SUBJ ทุกสมาชิก
  _IDSTD = 0
  @ 7,5 SAY "    รหัสนักศึกษา : " GET _IDSTD PICT '9999999'
  READ
  @ 8,10 SAY "เพิ่มวิชา"
  @ 9,10 SAY "SUBJ1 : " GET _SUBJ[1] PICT '@!'
  @ 10,10 SAY "SUBJ2 : " GET _SUBJ[2] PICT '@!'
  @ 11,10 SAY "SUBJ3 : " GET _SUBJ[3] PICT '@!'
  @ 12,10 SAY "SUBJ4 : " GET _SUBJ[4] PICT '@!'
  @ 13,10 SAY "ลดวิชา"
  @ 14,10 SAY "SUBJ5 : " GET _SUBJ[5] PICT '@!'
  @ 15,10 SAY "SUBJ6 : " GET _SUBJ[6] PICT '@!'
  @ 16,10 SAY "SUBJ7 : " GET _SUBJ[7] PICT '@!'
  @ 17,10 SAY "SUBJ8 : " GET _SUBJ[8] PICT '@!'
  READ
  YOURADD := YOURDROP := 4
  OKSUBJADD := OKSUBJDROP := 0
  USE SUBJECT
  FOR I = 1 TO 4
    IF _SUBJ[I] != SPACE(8)
      LOCATE FOR _SUBJ[I] = SUBJ
      IF FOUND()
        IF SREGIST >= SLIMIT
          @ 8+I,40 SAY 'วิชานี้เต็มแล้ว เพราะรับเพียง'+STR(SLIMIT)
        ELSE
          @ 8+I,40 SAY 'OK.'               // วิชาลงเพิ่ม จึงเพิ่มค่าในฟิลด์จำนวนที่ลง
          REPLACE SREGIST WITH SREGIST + 1
          USE REGIST
            APPEND BLANK                   // 4.6 คือเกรด P หมายถึงกำลังศึกษาอยู่
```

```
          REPLACE RYEAR WITH _RYEAR, RSEM WITH _RSEM ,;
            IDSTD WITH _IDSTD, SUBJ WITH _SUBJ[I],RGRADE WITH 4.6
        USE SUBJECT
        OKSUBJADD = OKSUBJADD + 1
      ENDIF
    ENDIF
  ELSE
    YOURADD = I - 1   ; I = 4
  ENDIF
NEXT
USE REGIST                              // การบอกเลิกจะต้องมีเรคอร์ดที่นั้นก่อนจึงจะบอกเลิกได้
FOR I = 5 TO 8
  IF _SUBJ[I] != SPACE(8)
    LOCATE FOR _RYEAR = RYEAR .AND. _RSEM = RSEM .AND. ;
           _IDSTD = IDSTD .AND. _SUBJ[I] = SUBJ
    IF FOUND()
      DELETE
      USE SUBJECT
      LOCATE FOR _SUBJ[I] = SUBJ
      IF FOUND()
        @ 9+I,40 SAY 'OK.'              // วิชานี้มีคนบอกเลิก จึงลดค่าใน SREGIST
        REPLACE SREGIST WITH SREGIST - 1
        OKSUBJDROP = OKSUBJDROP + 1
      ENDIF
      USE REGIST
    ENDIF
  ELSE
    YOURDROP = I - 5   ; I = 8
  ENDIF
NEXT
PACK                                    // ลบเรคอร์ดที่ทำเครื่องหมายลบไว้ ออกจากแฟ้มอย่างแท้จริง
@ 18,10 SAY 'จำนวนวิชาที่ท่านบอกเพิ่ม '+STR(YOURADD)
@ 19,10 SAY 'จำนวนวิชาที่ท่านบอกเลิก '+STR(YOURDROP)
```

```
    @ 20,10 SAY 'จำนวนวิชาที่บอกเพิ่มได้ '+STR(OKSUBJADD)
    @ 21,10 SAY 'จำนวนวิชาที่บอกเลิกได้ '+STR(OKSUBJDROP)
    INKEY(2)
    // พิมพ์ใบเสร็จ คิดวิชาละ 1000 บาท และค่าธรรมเนียมเพิ่มลด วิชาละ 20 บาท
    IF (OKSUBJADD + OKSUBJDROP) >= 1
      GASKPRT = ASKPRT()
      IF GASKPRT = 2 .AND. ISPRINTER()  // เลข 2 หมายถึงเครื่องพิมพ์
        SET DEVICE TO PRINTER
      ELSE ; CLS
      ENDIF
      @ 1,5  SAY "รหัส :"+STR(_IDSTD)
      @ 2,5  SAY "=========================="
      FOR I = 1 TO YOURADD                // พิมพ์วิชาที่บอกเพิ่ม
        @ 3+I,5 SAY 'A : '+ _SUBJ[I]
      NEXT
      FOR I = 1 TO YOURDROP               // พิมพ์วิชาที่บอกเลิก
        @ 3+I+YOURADD,5 SAY 'D : '+ _SUBJ[I+4]
      NEXT
      @ 4+YOURADD+YOURDROP,5 SAY "จำนวนเงินทั้งหมดที่ท่านต้องชำระ หรือรับคืน"+ ;
          STR(((OKSUBJADD-OKSUBJDROP)*1000)+(OKSUBJADD+OKSUBJDROP)*20)
      IF GASKPRT = 2 .AND. ISPRINTER() ; EJECT ; ENDIF
      SET DEVICE TO SCREEN
    ENDIF
    INKEY(0)                          // ขณะรอกดปุ่ม ถ้ากด ESC จะเลิกงานลงทะเบียน ถ้าไม่จะรับคนใหม่
    RESTORE SCREEN FROM SCR
  ENDDO
  SET CENTURY OFF
RETURN
```

5.4.3 โปรแกรมต่าง ๆ ใน SUB02.PRG

โปรแกรมใน SUB02.PRG นี้มีโปรแกรมย่อยทั้งหมด 6 โปรแกรม โดยรวมโปรแกรมที่เกี่ยวกับระบบงาน หลังลงทะเบียน เช่น โปรแกรมพิมพ์รายชื่อนักศึกษาที่ลงทะเบียนติดประกาศ พิมพ์รายชื่อให้อาจารย์ผู้สอน พิมพ์รายชื่อวิชาติดหน้าห้องเรียน พิมพ์สรุปรายงานให้ผู้บริหาร เป็นต้น

ระบบงานหลังลงทะเบียนเป็นระบบที่ทำให้เกิดประสิทธิภาพในการเรียนการสอน เพราะอาจารย์ผู้สอนจะได้รับรายชื่อ เพื่อตรวจสอบเวลาเรียนและลงคะแนน ผู้บริหารทราบจำนวนนักศึกษาที่ลงทะเบียน ทราบสถิติการใช้ห้องเรียนว่ามีประสิทธิภาพเพียงใด ดังนั้นระบบนี้จึงมุ่งที่จะทำรายงานให้หน่วยงานต่าง ๆ หลังลงทะเบียนทันที

ในหัวข้อนี้จะแสดงตัวอย่างโปรแกรมต่าง ๆ ที่มีในโปรแกรม SUB02.PRG ทั้งหมด 6 โปรแกรม โดยเริ่มจากตัวอย่างที่ 5.19 ถึง ตัวอย่างที่ 5.24 (R0201 - R0206)

: ตัวอย่างที่ 5.19

& ผลลัพธ์

```
// พิมพ์ข้อมูลการลงทะเบียนเรียนติดประกาศ ให้นักศึกษาดู
// รายงานแสดงลำดับ รหัสนักศึกษา และรหัสวิชา
// เมื่อเริ่มโปรแกรมจะทำการจัดเรียงข้อมูลใหม่หมด แล้วแยกกลุ่มข้อมูล โดยใช้รหัส
// จะนำข้อมูลเฉพาะปี 1996 ภาคเรียนที่ 1 เท่านั้นมาแสดง
// ในขณะพิมพ์ข้อมูลสามารถนับจำนวนของนักศึกษา และวิชาของนักศึกษาแต่ละคน
PROCEDURE R0201
  DO SORTREG1                          // ใช้จัดเรียงข้อมูลตามปี ภาค รหัส และวิชา อยู่ใน MAIN.PRG
  USE REGIST
  SET FILTER TO RYEAR = 1996 .AND. RSEM = '1'
  GO TOP
  _IDSTDOLD = 0
  IIDSTD := ISUBJ := 0
  DO WHILE !EOF()
    IF IDSTD <> _IDSTDOLD
      IIDSTD++
      ISUBJ = 0
      INKEY(0.5)
      ? '* --> ',IIDSTD,IDSTD
      _IDSTDOLD = IDSTD
    ENDIF
    ISUBJ++
    ? ISUBJ,SUBJ
    SKIP
  ENDDO
  INKEY(0)
  CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.20

& ผลลัพธ์

```
// พิมพ์รายชื่อนักศึกษาให้อาจารย์ผู้สอน
// รายงานแสดงลำดับ รหัสวิชา รหัสนักศึกษา ชื่อ สกุล รหัสสาขา และสถานภาพ
// เมื่อเริ่มโปรแกรมต้องจัดเรียงข้อมูลให้เรียงตามรหัสวิชาก่อน
PROCEDURE R0202
  DO SORTREG2                          // ใช้จัดเรียงข้อมูลตามวิชา ปี ภาค และรหัส อยู่ใน MAIN.PRG
  SELE 1
  USE REGIST
  SET FILTER TO RYEAR = 1996 .AND. RSEM = '1'
  GO TOP
  SELE 2
  USE IDSTD
  SELE 1
  _SUBJOLD = SPACE(8)
  IIDSTD := ISUBJ := 0
  DO WHILE !EOF()
    IF SUBJ <> _SUBJOLD
      ISUBJ++
      IIDSTD = 0
      INKEY(0.5)
      ? '* --> ',ISUBJ,SUBJ
      _SUBJOLD = SUBJ
    ENDIF
    IIDSTD++
    _IDSTD = IDSTD
    SELE 2
    LOCATE FOR IDSTD = _IDSTD
    IF FOUND()
      ? IIDSTD,IDSTD,INAME,ISURN,IMAJOR,ISTAT
    ELSE ; ? IIDSTD,_IDSTD          // เพราะรหัสนี้ไม่มีใน IDSTD จึงต้องใช้ตัวแปรแทน
    ENDIF
    SELE 1                           // ต้องมาที่พื้นที่ 1 จึงจะ SKIP เพราะต้องการเลื่อนเรคอร์ดใน REGIST
    SKIP
```

```
  ENDDO
  INKEY(0)
  CLOSE ALL                          // ต้อง CLOSE ALL เพราะเปิดแฟ้มไว้หลายพื้นที่
RETURN
```

: ตัวอย่างที่ 5.21

& ผลลัพธ์

```
// พิมพ์สรุปการลงทะเบียนให้ผู้บริหาร โดยพิมพ์จำนวนที่ลงเรียนในแต่ละวิชา
// โปรแกรมนี้คุมกลุ่ม 2 แบบคือ กลุ่มของวิชา และกลุ่มของภาคการศึกษา
// รายงานแสดงปีการศึกษา ภาคการศึกษา วิชา จำนวนที่ลงทะเบียนเรียน
// สามารเลือกปลายทางของผลลัพธ์ได้
// ควบคุมจำนวนบรรทัดต่อหน้าของจอภาพเป็น 20 และของเครื่องพิมพ์เป็น 30
PROCEDURE R0203
  DO SORTREG2                        // ใช้จัดเรียงข้อมูลตามวิชา ปี ภาค และรหัส อยู่ใน MAIN.PRG
  USE REGIST
  _SUBJO = SUBJ
  _RYEARO = RYEAR
  _RSEMO = RSEM
  IIDSTD := ISUBJ := 0
  GASKPRT = ASKPRT()
  IF GASKPRT = 2                     // เลข 2 หมายถึงพิมพ์รายงานทางเครื่องพิมพ์
    SET DEVICE TO PRINTER
    CNTPAGE = 30
  ELSE
    CNTPAGE = 20
  ENDIF
  CNTLINE = 1
  DO WHILE !EOF()
    IIDSTD++
    IF SUBJ <> _SUBJO .OR. ISUBJ = 0     // ตรวจสอบความต่างของวิชา
      INKEY(0.5)
      ISUBJ++
      @ CNTLINE,5 SAY STR(ISUBJ)+SUBJ
      _SUBJO = SUBJ
      CNTLINE++
```

```
    ENDIF
    SKIP                                    // ตรวจสอบภาคการศึกษา
    IF SUBJ <> _SUBJO .OR. _RYEARO <> RYEAR .OR. _RSEMO <> RSEM .OR. EOF()
      @ CNTLINE,10 SAY STR(_RYEARO) + _RSEMO + STR(IIDSTD)
      CNTLINE++
      _RYEARO = RYEAR
      _RSEMO = RSEM
      IIDSTD = 0
    ENDIF
    IF CNTLINE >= CNTPAGE .OR. EOF()   // ตรวจสอบการแบ่งหน้า
      CNTLINE = 1
      IF GASKPRT = 2 ; EJECT
      ELSE ; INKEY(3) ; CLS ; ENDIF
    ENDIF
  ENDDO
  SET DEVICE TO SCREEN
  INKEY(3)
  CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.22

& ผลลัพธ์

```
// พิมพ์รายชื่อวิชาของแต่ละห้อง โดยเลือกปลายทางของผลลัพธ์ได้
// รายงานแสดงลำดับและรหัสห้อง รหัสวิชา เวลาเรียน และจำนวนที่ลงทะเบียน
// โปรแกรมนี้จะแบ่งห้องเรียน แล้วรายงานชื่อวิชาในห้องนั้น
// โดยรายงานจะแสดงห้องละ 1 หน้าเท่านั้น
PROCEDURE R0204
  DO SORTSUB1                           // ใช้จัดเรียงแฟ้มวิชาตามห้องเรียนและชื่อวิชา
  USE SUBJECT
  _ROOMO = SPACE(4)
  IROOM := ISUBJ := 0
  GASKPRT = ASKPRT()
  IF GASKPRT = 2
    SET DEVICE  TO PRINTER
  ENDIF
```

```
  DO WHILE !EOF()
    IF ROOM <> _ROOMO
      IF GASKPRT = 2 .AND. IROOM != 0
        EJECT
      ELSE
        INKEY(3)
        CLS
      ENDIF
      IROOM++
      @ 1,1 SAY "รายวิชาที่มีการใช้ในแต่ละห้อง"
      @ 2,1 SAY "ห้องที่ "+STR(IROOM)+" เบอร์ "+ROOM
      @ 3,1 SAY "============================"
      _ROOMO = ROOM
      ISUBJ = 0
    ENDIF
    ISUBJ++
    @ ISUBJ+3,1 SAY STR(ISUBJ) + SUBJ + STIME + STR(SREGIST)
    SKIP
  ENDDO
  SET DEVICE TO SCREEN
  INKEY(3)
  CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.23

& ผลลัพธ์

```
// พิมพ์รายชื่อวิชาที่อาจารย์ผู้สอนแต่ละท่านรับผิดชอบ
// รายงานแสดงรหัสอาจารย์ ชื่ออาจารย์ รหัสวิชา ห้องเรียน และเวลาเรียน
// สามารถเลือกปลายทางของผลลัพธ์ได้
PROCEDURE R0205
  DO SORTSUB2
  SELE 1
  USE TEACHER
  SELE 2
  USE SUBJECT
```

```
IF ASKPRT() = 2                          // ได้ผลลัพธ์ออกเครื่องพิมพ์
  SET DEVICE TO PRINTER
  FOR I= 1 TO RECCOUNT()
    @ I,10 SAY STR(I) + SUBJ + ' | ' + ROOM + ' | ' + STIME
    SELE 1
      LOCATE FOR SUBJECT->TEACHID = TEACHID
      IF FOUND()
        @ I,50 SAY ' | ' + TNAME + ' | ' + TSURN
      ENDIF
      IF MOD(I,15) = 0
        EJECT
      ENDIF
    SELE 2
    SKIP
  NEXT
  EJECT
  SET DEVICE TO SCREEN
ELSE
  I = 0                                  // ได้ผลลัพธ์ออกทางจอภาพ
  DO WHILE !EOF()
    I++
    ? I,SUBJ,ROOM,STIME
    SELE 1
      LOCATE FOR SUBJECT->TEACHID = TEACHID
      IF FOUND() ; ?? TNAME,TSURN ; ENDIF
      IF I = 15
        INKEY(3)
        CLS
        I = 0
      ENDIF
    SELE 2
    SKIP
  ENDDO
```

```
  ENDIF
  INKEY(3)
  CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.24

& ผลลัพธ์

```
// พิมพ์สรุปจำนวนนักศึกษาแต่ละวิชา โดยระบุปี และภาค
// รายงานแสดงลำดับ รหัสวิชา จำนวนที่ลงทะเบียน
// สามารถเลือกรูปแบบของผลลัพธ์ได้
PROCEDURE R0206
  _RYEAR = 0
  _RSEM = '0'
  @ 4,5 SAY 'สรุปจำนวนนักศึกษาในปีและภาคที่ระบุ'
  @ 5,5 SAY 'ปีการศึกษา : ' GET _RYEAR
  @ 6,5 SAY 'ภาคการศึกษา : ' GET _RSEM
  READ
  CLS
  DO SORTREG2                          // ใช้จัดเรียงข้อมูลตามวิชา ปี ภาค และรหัส อยู่ใน MAIN.PRG
  USE REGIST
  SET FILTER TO RYEAR = _RYEAR .AND. RSEM = _RSEM
  GO TOP
  _SUBJO = SUBJ
  IIDSTD := ISUBJ := 0
  GASKPRT = ASKPRT()
  CNTLINE = 1
  CNTPAGE = 20                         // เริ่มต้นสำหรับจอภาพให้หน้าละ 20 บรรทัด
  IF GASKPRT = 2                       // เลข 2 หมายถึงพิมพ์รายงานทางเครื่องพิมพ์
    SET DEVICE TO PRINTER
    CNTPAGE = 30
  ENDIF
  @ CNTLINE,5 SAY "สรุปจำนวนนักศึกษาประจำปี " + STR(_RYEAR) + "/" + _RSEM
  CNTLINE++
  DO WHILE !EOF()
    IIDSTD++
```

```
    SKIP
    IF SUBJ <> _SUBJO .OR. EOF()          // ตรวจสอบความต่างกลุ่มของวิชา
      ISUBJ++
      @ CNTLINE,5 SAY STR(ISUBJ)+_SUBJO+STR(IIDSTD)
      _SUBJO = SUBJ
      CNTLINE++
      IIDSTD = 0
    ENDIF
    IF CNTLINE >= CNTPAGE .OR. EOF()   // ตรวจสอบการแบ่งหน้า
      CNTLINE = 1
      IF GASKPRT = 2 ; EJECT
      ELSE ; INKEY(3) ; CLS ; ENDIF
    ENDIF
  ENDDO
  SET DEVICE TO SCREEN
  INKEY(3)
  CLOSE ALL
RETURN
```

5.4.4 โปรแกรมต่าง ๆ ใน SUB03.PRG

โปรแกรมใน SUB03.PRG นี้มีโปรแกรมย่อยทั้งหมด 4 โปรแกรม โดยรวมโปรแกรมที่เกี่ยวกับระบบงาน เมื่อถูกร้องขอ เช่น โปรแกรมขอถอนวิชาเรียน โปรแกรมพิมพ์ตารางเรียน โปรแกรมพิมพ์ใบรายงานผลการเรียนโปรแกรมปรับปรุงข้อมูลอาจารย์ เป็นต้น

ระบบงานเมื่อถูกร้องขอ เป็นระบบเกิดขึ้นเมื่อมีผู้ร้องขอปรับปรุง หรือรายงานจากระบบ โดยไม่ขึ้นกับเวลา ว่าจะเป็นต้นภาคเรียน ท้ายภาค หรือช่วงปิดภาคเรียน แต่ในระบบอื่น ๆ ผู้จัดระบบอาจแยกโปรแกรมต่าง ๆ ในระบบนี้เข้าไปรวมกับระบบอื่นได้ แต่ผู้เขียนแยกออกมาเพื่อให้เห็นการแยกระบบย่อย ในอีกมุมมองหนึ่ง

ในหัวข้อนี้จะแสดงตัวอย่างโปรแกรมต่าง ๆ ที่มีในโปรแกรม SUB03.PRG ทั้งหมด 4 โปรแกรม โดยเริ่มจากตัวอย่างที่ 5.25 ถึง ตัวอย่างที่ 5.28 (R0301 - R0304)

: ตัวอย่างที่ 5.25

& ผลลัพธ์

```
// ป้อนข้อมูลการขอถอนวิชาของนักศึกษา (W) โดยระบุรหัสนักศึกษา และวิชา
// สำหรับปีการศึกษาและภาคเรียนจะระบุไว้แน่นอนในโปรแกรมเป็นปี 1996 ภาคเรียนที่ 1
// โปรแกรมจะค้นหาข้อมูลแล้วเปลี่ยนเกรด จากนั้นจะรอรับรหัสนักศึกษาและวิชาใหม่
// จนกว่าจะกดปุ่ม ESC เพื่อเลิกการทำงาน
```

```
PROCEDURE R0301
  USE REGIST
  SET FILTER TO RYEAR = 1996 .AND. RSEM= '1'
  DO WHILE LASTKEY() != 27
    GO TOP
    _IDSTD = 0
    _SUBJ = SPACE(8)
    @ 6,5 CLEAR
    @ 4,5 SAY "รหัสนักศึกษาที่ขอเพิกถอนวิชา : " GET _IDSTD
    @ 5,5 SAY "รหัสวิชาที่ขอเพิกถอนวิชา : " GET _SUBJ
    READ
    LOCATE FOR IDSTD = _IDSTD .AND. SUBJ = _SUBJ
    IF FOUND()
      @ 6,5 SAY 'เกรดเดิม ' + STR(RGRADE)
      @ 7,5 SAY 'การขอเพิกถอนของ ' + STR(_IDSTD) + ' สมบูรณ์'
      REPLACE RGRADE WITH 4.2
    ELSE
      @ 6,5 SAY "ไม่พบรหัสนักศึกษา ' + STR(_IDSTD)
    ENDIF
    INKEY(0)
  ENDDO
RETURN
```

: ตัวอย่างที่ 5.26

& ผลลัพธ์

```
// พิมพ์ตารางเรียนของนักศึกษาแต่ละวิชา
// รายงานแสดงให้เห็นว่านักศึกษาแต่ละคน เรียนวิชาอะไร ห้อง เวลาและใครคือผู้สอน
// ตัวอย่างนี้แสดงการใช้ DEVPOS และ DEVOUT เพื่อนำข้อมูลมาแสดง
// เป็นอีกวิธีหนึ่งที่จะนำข้อมูลออกทางเครื่องพิมพ์ได้
// แสดงให้เห็นการเชื่อมแฟ้มจาก 3 แฟ้มคือ TEACHER,SUBJECT และ REGIST
// สามารถเลือกปลายทางของผลลัพธ์ได้
PROCEDURE R0302
  DO SORTREG1                          // จัดเรียงตามปี ภาค และรหัส
  SELE 1 ; USE TEACHER
  SELE 2 ; USE SUBJECT
```

```
SELE 3 ; USE REGIST
SET FILTER TO RYEAR = 1996 .AND. RSEM = '1'
GO TOP
_IDSTDO = 0
GASKPRT = ASKPRT()
IF GASKPRT = 2
  SET DEVICE TO PRINTER
ENDIF
DO WHILE !EOF()
  IF IDSTD <> _IDSTDO
    IF GASKPRT = 2 ;  DEVPOS(PROW()+1,1)
      ELSE ; DEVPOS(ROW()+1,1) ; ENDIF
    DEVOUT(IDSTD)
    _IDSTDO = IDSTD
  ENDIF
  SELE 2                                  // แฟ้ม SUBJECT.DBF
  LOCATE FOR SUBJ = REGIST->SUBJ
  IF FOUND()
    TXT = SUBJ + ROOM + STIME + STR(TEACHID)
    SELE 1                                // แฟ้ม TEACHER.DBF
    LOCATE FOR TEACHID = SUBJECT->TEACHID
    IF FOUND()
      TXT = TXT + TNAME + TSURN
    ENDIF
    IF GASKPRT = 2 ;  DEVPOS(PROW()+1,1)
      ELSE ; DEVPOS(ROW()+1,1) ; ENDIF
    DEVOUT(TXT)                           // นำข้อความทั้งหมดมารวมกันก่อนพิมพ์
  ENDIF
  SELE 3
  SKIP
ENDDO
IF GASKPRT = 2 ; EJECT ; ENDIF
SET DEVICE TO SCREEN
```

```
  INKEY(0)
  CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.27

& ผลลัพธ์

```
// พิมพ์รายงานผลการเรียนตลอดหลักสูตร โดยระบุรหัสนักศึกษา
// รายงานแสดงชื่อวิชา และเกรดที่นักศึกษาได้รับ
// บรรทัดสุดท้ายแสดงการนับเกรดที่นักศึกษาได้ A และ F
// แสดงให้เห็นว่า สามารถใช้ SET FILTER แทนคำสั่ง LOCATE ได้
// และการพิมพ์ข้อมูลทางเครื่องพิมพ์ด้วย SET PRINT ON และ ?
// สามารถเลือกปลายทางของผลลัพธ์ได้
PROCEDURE R0303
  DO SORTREG1                          // จัดเรียงตามปี ภาค และรหัส
  SELE 1 ; USE IDSTD
  SELE 2 ; USE REGIST
    _IDSTD = 0
    @ 0,5 SAY "รหัสนักศึกษาที่ต้องการรายงาน : " GET _IDSTD  ; READ
    GASKPRT = ASKPRT()
    IF  GASKPRT = 2 ; SET PRINT ON ;  ENDIF
    CLS ; _RYEAR = 0  ;  _RSEM = '0'
    SET FILTER TO IDSTD = _IDSTD
    COUNT TO CNTSUBJ                   // นำไปพิมพ์เป็นผลลัพธ์ตอนท้ายโปรแกรม
    COUNT TO CNTA   FOR RGRADE = 4   // นับเฉพาะเกรด A
    COUNT TO CNTF   FOR RGRADE = 0   // นับเฉพาะเกรด F
    GO TOP                             // คำสั่ง COUNT ทำให้ตัวชี้เรคอร์ดเปลี่ยน จึงต้องเซตตัวชี้ที่นี่
  SELE 1
    SET FILTER TO IDSTD = _IDSTD  ;    GO TOP
    ? IDSTD,INAME,ISURN,IMAJOR,ISTAT
  SELE 2
  DO WHILE !EOF() .AND. CNTSUBJ > 0
    IF RYEAR <> _RYEAR .OR. RSEM <> _RSEM
      ? RSEM , "/" , STR(RYEAR)
      _RYEAR = RYEAR
      _RSEM = RSEM
```

```
    ENDIF
    _GRADE = GRADECHR(RGRADE)
    ? "|" , SUBJ , "|" , _GRADE , "|"
    SKIP
  ENDDO
  ? "จำนวนวิชาที่นักศึกษาลงทะเบียน : " , STR(CNTSUBJ)
  ? "จำนวนวิชาที่ได้เกรด A : " , STR(CNTA)
  ? "จำนวนวิชาที่ได้เกรด F : " , STR(CNTF)
  IF GASKPRT = 2 ; EJECT ; ENDIF
  SET PRINT OFF
  INKEY(0)
RETURN
```

: ตัวอย่างที่ 5.28

& ผลลัพธ์

```
// ปรับปรุงแฟ้มอาจารย์ อย่างง่าย ๆ ด้วยฟังก์ชัน BROWSE
// โดยเลือกกลุ่มข้อมูลตามฟิลด์ที่ต้องการแก้ไขได้
PROCEDURE R0304
  @ 1,5 SAY "เลือกกลุ่มข้อมูลที่ท่านต้องการปรับปรุง ตามฟิลด์ใด ?"
  @ 2,5 PROMPT "1. รหัส"
  @ 3,5 PROMPT "2. ชื่อ"
  @ 4,5 PROMPT "3. สกุล"
  @ 5,5 PROMPT "4. อายุ"
  @ 6,5 PROMPT "5. เพศ"
  MENU TO FTEACH
  IF FTEACH > 0
    _KEYIN = "?                    "
    @ FTEACH + 1,20 SAY ":>" GET _KEYIN ; READ // รับค่าที่ต้องการค้นหา
    IF FTEACH = 4 .OR. FTEACH = 1
      _KEYCHK = VAL(_KEYIN)
    ELSE
      _KEYCHK = RTRIM(_KEYIN)
    ENDIF
  ENDIF
  USE TEACHER
```

```
  IF FTEACH = 1 ; SET FILTER TO TEACHID = _KEYCHK ; ENDIF
  IF FTEACH = 2 ; SET FILTER TO TNAME = _KEYCHK ; ENDIF
  IF FTEACH = 3 ; SET FILTER TO TSURN = _KEYCHK ; ENDIF
  IF FTEACH = 4 ; SET FILTER TO TAGE = _KEYCHK ; ENDIF
  IF FTEACH = 5 ; SET FILTER TO TSEX = _KEYCHK ; ENDIF
  COUNT TO CNTREC
  IF CNTREC > 0
    GO TOP
    BROWSE(7,5,21,75)
    PACK
  ELSE
    @ 7,5 SAY "ไม่พบเรคอร์ด หรือกลุ่มเรคอร์ดที่ท่านต้องการปรับปรุง"
    INKEY(0)
  ENDIF
RETURN
```

5.4.5 โปรแกรมต่าง ๆ ใน SUB04.PRG

โปรแกรมใน SUB04.PRG นี้มีโปรแกรมย่อยทั้งหมด 5 โปรแกรม โดยรวมโปรแกรมที่เกี่ยวกับระบบงานประจำภาคเรียน เช่น โปรแกรมพิมพ์รายชื่อนักศึกษาติดหน้าห้องสอบในการสอบกลางภาค และปลายภาค โปรแกรมพิมพ์รายชื่อนักศึกษาแยกตามสถานภาพ หรือแยกตามสาขา โปรแกรมพิมพ์ใบรายงานผลการเรียนประจำภาคเรียน โปรแกรมปรับปรุงเกรดนักศึกษาเมื่ออาจารย์ส่งเกรด เป็นต้น

ระบบงานประจำภาคเรียน เป็นระบบที่มีงานที่ต้องทำในทุกภาคเรียน และส่วนใหญ่เป็นงานประจำ เช่น การจัดทำรายงานแยกตามสถานภาพ การพิมพ์รายชื่อติดหน้าห้องสอบ หรือการป้อนเกรดนักศึกษาเมื่อสิ้นภาคเรียน บางครั้งผู้ออกแบบระบบอาจนำโปรแกรมในระบบอื่น ๆ มารวมในระบบนี้ได้ เพราะระบบนี้เป็นระบบใหญ่มีงานมาก แต่ผู้ออกแบบอาจแยกงานบางอย่างไปสร้างระบบย่อยใหม่ เพื่อทำให้ลดความซับซ้อนได้

ในหัวข้อนี้จะแสดงตัวอย่างโปรแกรมต่าง ๆ ที่มีในโปรแกรม SUB04.PRG ทั้งหมด 5 โปรแกรม โดยเริ่มจากตัวอย่างที่ 5.29 ถึง ตัวอย่างที่ 5.33 (R0401 - R0405)

: ตัวอย่างที่ 5.29

& ผลลัพธ์

```
// พิมพ์รายชื่อนักศึกษาแต่ละวิชาติดหน้าห้องสอบ
// รายงานแสดงลำดับ รหัสวิชา รหัสนักศึกษา ชื่อ สกุล และเบอร์เก้าอี้ในห้องสอบ
// โดยสมมติให้ห้องสอบกับห้องเรียนใช้ห้องเดียวกัน
PROCEDURE R0401
  DO SORTREG2                    // ใช้จัดเรียงข้อมูลตามวิชา ปี ภาค และรหัส อยู่ใน MAIN.PRG
```

```
SELE 1 ; USE REGIST
  SET FILTER TO RYEAR = 1996 .AND. RSEM = '1'
  GO TOP
SELE 2 ; USE IDSTD
SELE 3 ; USE ROOM
SELE 4 ; USE SUBJECT
SELE 1
_SUBJOLD = SPACE(8)
IIDSTD := ISUBJ := 0
DO WHILE !EOF()
  _IDSTD = IDSTD
  IIDSTD++
  IF SUBJ <> _SUBJOLD                  // ใช้พิมพ์หัวของหน้าแต่ละหน้า
    ISUBJ++                            // ใช้นับวิชา
    IIDSTD = 1                         // ใช้นับนักศึกษาในวิชา
    INKEY(0.5)
    ? '*',ISUBJ,SUBJ
    _SUBJOLD = SUBJ
    SELE 4                             // แฟ้มวิชา ใช้ค้นหาห้องเรียน
      LOCATE FOR SUBJ = REGIST->SUBJ
      IF FOUND()
        ?? ROOM                        // ห้องสอบของวิชานั้น
      ENDIF
  ENDIF
  SELE 2                               // แฟ้มนักศึกษา ใช้ค้นหาชื่อ และสาขา
    LOCATE FOR IDSTD = _IDSTD
    IF FOUND()
      ? IIDSTD,IDSTD,INAME,ISURN
    ELSE
      ? IIDSTD,_IDSTD                  // เพราะรหัสนี้ไม่มีใน IDSTD จึงต้องใช้ตัวแปรแทน
    ENDIF
  SELE 3                               // แฟ้มห้องเรียน ใช้ค้นหาเลขที่โต๊ะ
    LOCATE FOR SUBJECT->ROOM = ROOM .AND. RCSEQ = IIDSTD
```

```
      IF FOUND()
        ?? "CHAIR : ",RCHAIR
      ENDIF
    SELE 1                          // แฟ้มลงทะเบียน เป็นแฟ้มหลักของรายงานนี้
    SKIP
  ENDDO
  INKEY(0)
  CLOSE ALL                         // ต้อง CLOSE ALL เพราะเปิดแฟ้มไว้หลายพื้นที่
RETURN
```

: ตัวอย่างที่ 5.30

& ผลลัพธ์

```
// พิมพ์รายชื่อแยกตามสถานภาพ โดยครั้งแรกแสดงผลทางจอภาพทันที
// ใช้คำสั่ง SET PRINT ON และ ? ในการจัดพิมพ์
// ถ้ากด P จะพิมพ์ข้อมูลที่พิมพ์ไปแล้วทางเครื่องพิมพ์ทันที
// หากกด P อีกครั้งจะพิมพ์ออกทางเครื่องพิมพ์อีก จนกว่าจะกดปุ่มอื่น
PROCEDURE R0402
  DO SORTID2                        // ใช้จัดเรียงตามสถานภาพ และรหัส
  _ISTATO = " "
  USE IDSTD
  CHKPRT = "P"
  YESPRINTON = "N"
  DO WHILE CHKPRT = "P"
    WHILE !EOF()
      IF ISTAT <> _ISTATO
        ? "======================"
        ? "CURRENT STATUS : ",ISTAT
        _ISTATO = ISTAT
      ENDIF
      ? IDSTD,INAME,ISURN,IMAJOR
      SKIP
    END
    WAIT "ถ้าอยากพิมพ์ให้กด P"     // บรรทัดนี้ให้ผลลัพธ์ทางเครื่องพิมพ์ และจอภาพ
    CHKPRT = UPPER(CHR(LASTKEY()))
    IF CHKPRT = "P"                 // ตรวจสอบความต้องการรายงานทางเครื่องพิมพ์
```

```
      YESPRINTON = "Y"
      SET PRINT ON
      GO TOP ; CLS
    ENDIF
  ENDDO
  IF  YESPRINTON = "Y" ; EJECT ; ENDIF
  SET PRINT OFF
RETURN
```

: ตัวอย่างที่ 5.31

& ผลลัพธ์

```
// พิมพ์รายชื่อแยกตามสาขา และจัดเรียงตามชื่อ
// จัดกลุ่มข้อมูลตามสาขา โดยรายงานแสดง รหัสนักศึกษา ชื่อ สกุล และสถานภาพ
// ผลลัพธ์แสดงทางเครื่องพิมพ์ด้วยการกด P เมื่อแสดงข้อมูลทางจอภาพไปแล้ว
// จำนวนบรรทัดต่อหน้าของจอภาพระบุไว้เป็น 10 และเครื่องพิมพ์เป็น 30
PROCEDURE R0403
  DO SORTID3                              // ใช้จัดเรียงตามสาขา และชื่อ
  _IMAJORO = 0
  USE IDSTD
  CHKPRT = "P"
  YESPRINTON = "N"
  LN = 1 ; LNSCR = 10 ; LNPRT = 30
  @ LN,1 SAY "ข้อมูลชุดนี้เรียงตามจัดกลุ่มตามสาขา และเรียงตามชื่อ"
  DO WHILE CHKPRT = "P"
    WHILE !EOF()
      IF IMAJOR <> _IMAJORO
        LN++ ; @ LN,1 SAY "======================"
        LN++ ; @ LN,1 SAY "CURRENT MAJOR : " + STR(IMAJOR)
        _IMAJORO = IMAJOR
      ENDIF
      LN++ ; @ LN,1 SAY STR(IDSTD) + INAME + ISURN + ISTAT
      IF (YESPRINTON = "N" .AND. LN >= LNSCR) .OR. ;
        (YESPRINTON = "Y" .AND. LN >= LNPRT)
        INKEY(1)
        CLS                               // เงื่อนไขนี้ ทำให้จำนวนบรรทัดต่อจอภาพ และเครื่องพิมพ์ต่างกัน
```

```
      LN = 0
    ENDIF
    SKIP
  END
  IF  YESPRINTON = "Y" ; EJECT ; ENDIF
  // คำสั่ง SET DEVICE TO PRINTER ไม่ส่งผลต่อคำสั่ง WAIT
  WAIT "ถ้าอยากพิมพ์ให้กด P"              // สอบถามการพิมพ์ออกเครื่องพิมพ์
  CHKPRT = UPPER(CHR(LASTKEY()))
  IF CHKPRT = "P"
    YESPRINTON = "Y"
    SET DEVICE  TO PRINTER
    GO TOP ; CLS
  ENDIF
 ENDDO
 SET DEVICE TO SCREEN
 CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.32

& ผลลัพธ์

```
// ปรับปรุงเกรดนักศึกษาเมื่ออาจารย์ส่งเกรด โดยระบุปีการศึกษา ภาค และวิชา
// รอรับเกรดใหม่ แล้วปรับปรุงทีละคน จนหมดข้อมูลที่ตรงกับเงื่อนไข
PROCEDURE R0404
 DO SORTREG1                          // ใช้จัดเรียงข้อมูลตามปี ภาค รหัส และวิชา
 _RYEAR = 1996
 _RSEM = '1'
 _SUBJ = SPACE(8)
 USE REGIST
 DO WHILE !FOUND()
  @ 5,5 SAY "ปรับปรุงเกรดนักศึกษา"
  @ 6,5 SAY "ปีการศึกษา " GET _RYEAR PICT "9999"
  @ 7,5 SAY "ภาคการศึกษา " GET _RSEM
  @ 8,5 SAY "วิชาที่ต้องการป้อนเกรด " GET _SUBJ PICT "@!"
  READ
  LOCATE FOR _RYEAR = RYEAR .AND. _RSEM = RSEM .AND. SUBJ = _SUBJ
```

```
  ENDDO
  SET FILTER TO RYEAR = _RYEAR .AND. RSEM = _RSEM .AND. SUBJ = _SUBJ
  COUNT TO CNTALL
  GO TOP
  CNT = 1
  @ 9,5 SAY "จำนวนนักศึกษาทั้งหมดในวิชานี้ : "+STR(CNTALL)
  DO WHILE !EOF() .AND. LASTKEY() != 27
    _GRADE = "  "
    WHILE _GRADE = "  "
      @ 10,5 SAY STR(CNT) +": "+ STR(IDSTD) +" : "+ STR(RGRADE) +": ";
        GET _GRADE
      READ
    END
    _RGRADE = GRADENUM(_GRADE)      // แปลงเกรดตัวอักษรเป็นตัวเลข
    REPLACE RGRADE  WITH _RGRADE
    SKIP
    CNT++
  ENDDO
  CLOSE ALL
RETURN
```

: ตัวอย่างที่ 5.33

& ผลลัพธ์

```
// พิมพ์ใบรายงานผลการเรียนประจำภาค โดยระบุปีการศึกษา และภาคการศึกษา
// จัดกลุ่มตามรหัสนักศึกษา เพื่อพิมพ์ใบรายงานผลการเรียนให้นักศึกษาแต่ละคน
// รายงานแสดงลำดับ รหัสวิชา และเกรด
// บรรทัดสุดท้ายแสดงเกรดเฉลี่ยประจำภาคเรียนของนักศึกษาคนนั้น
PROCEDURE R0405
  _RYEAR = 1996
  _RSEM = '1'
  @ 5,5 SAY "ปีการศึกษา" GET _RYEAR PICT "9999"
  @ 6,5 SAY "ภาคการศึกษา" GET _RSEM
  READ
  USE REGIST
  SET FILTER TO RYEAR = _RYEAR .AND. RSEM = _RSEM
```

```
GO TOP
_GPA = 0
_CNTSUBJ = 0
_SUMPOINT = 0
_IDSTDO = 0
CNTID = 0
CNT = 1
DO WHILE !EOF() .AND. LASTKEY() != 27
  IF IDSTD <> _IDSTDO
    IF CNT <> 1
      _NEXTIDSTD = IDSTD
      COUNT TO _CNTSUBJ FOR RYEAR = _RYEAR .AND. RSEM = _RSEM .AND.;
       IDSTD = _IDSTDO .AND. RGRADE <= 4
      SUM RGRADE TO _SUMPOINT FOR RYEAR = _RYEAR .AND. RSEM = _RSEM ;
       .AND. IDSTD = _IDSTDO .AND. RGRADE <= 4
      _GPA  = _SUMPOINT / _CNTSUBJ
      @ ROW()+1,5 SAY "เกรดเฉลี่ยประจำภาคเรียนนี้คือ "+STR(_GPA)
      INKEY(3)
      LOCATE FOR IDSTD = _NEXTIDSTD          // เพราะ COUNT จะเปลี่ยนตัวชี้ใหม่
    ENDIF                                    // จึงต้องใช้ LOCATE เลื่อนตัวชี้ให้ถูกต้องอีกครั้ง
    CNT = 1
    CNTID++
    @ 7,5 CLEAR TO 22,75
    SETPOS(7,1)
    @ ROW()+1,5 SAY STR(CNTID) + " : " + STR(IDSTD)
    @ ROW()+1,5 SAY "================================"
    _IDSTDO = IDSTD
  ENDIF
  _GRADE = GRADECHR(RGRADE)  //  แปลงเกรดตัวเลขเป็นตัวอักษร
  @ ROW()+1,10 SAY STR(CNT) + " : " + SUBJ + " : " + _GRADE
  SKIP
  CNT++
ENDDO
```

```
  CLOSE ALL
RETURN
```

5.4.6 โปรแกรมต่าง ๆ ใน SUB05.PRG

โปรแกรมใน SUB05.PRG นี้มีโปรแกรมย่อยทั้งหมด 5 โปรแกรม โดยรวมโปรแกรมที่เกี่ยวกับระบบปรับปรุงและพิมพ์ข้อมูลจากแฟ้ม เช่น โปรแกรมปรับปรุงแฟ้มและพิมพ์ ของแฟ้มในแบบต่าง ๆ ทำให้สามารถเลือกเปิดแฟ้มในระบบ โดยระบบประกอบด้วย แฟ้มนักศึกษา แฟ้มวิชา แฟ้มผู้สอน แฟ้มลงทะเบียน และแฟ้มห้อง

ระบบงานปรับปรุงและพิมพ์จากแฟ้มอย่างง่าย ๆ ช่วยทำให้การปรับปรุงข้อมูลทำได้ทันที แต่ไม่เหมาะกับการนำไปใช้งานจริง เพราะการปรับปรุงข้อมูลจะทำให้ควบคุมความปลอดภัยหรือความถูกต้องทำได้ยาก ดังนั้นการปรับปรุงจึงควรผ่านโปรแกรมที่อนุญาติให้แก้ไขข้อมูล โดยมีการตรวจสอบความถูกต้องของข้อมูลอย่างดี ก่อนจะจัดเก็บข้อมูลลงแฟ้ม และการพิมพ์ข้อมูลควรที่จะเชื่อมข้อมูลกับแฟ้มต่าง ๆ ทำให้เกิดรายงานที่อ่านเข้าใจได้โดยง่ายเพราะการนำแฟ้มเพียงแฟ้มเดียวมาพิมพ์จะไม่สมบูรณ์นัก เพราะมีแฟ้มหลาย ๆ แฟ้มที่เก็บข้อมูลเป็นรหัส ทำให้ต้องนำรหัสนั้นไปเชื่อมกับแฟ้มอื่น จึงจะได้ข้อมูลที่อ่านแล้วเข้าใจได้ทันที

ในหัวข้อนี้จะแสดงตัวอย่างโปรแกรมต่าง ๆ ที่มีในโปรแกรม SUB05.PRG ทั้งหมด 5 โปรแกรม โดยเริ่มจากตัวอย่างที่ 5.34 ถึง ตัวอย่างที่ 5.38 (R0501 - R0505)

: ตัวอย่างที่ 5.34

& ผลลัพธ์

```
// การปรับปรุงแฟ้มนักศึกษา
// โดยเลือกพิมพ์ เพิ่ม ลบ หรือแก้ไขได้
PROCEDURE R0501
  _IDSTD = 0
  X = ALERT("ปรับปรุงแฟ้มนักศึกษา",{'พิมพ์ข้อมูล','เพิ่ม','ลบ','แก้ไข'})
  DO CASE
    CASE X = 0 ; WAIT "ไม่ได้เลือกตัวเลือกใดเลย"
    CASE X = 1
    // พิมพ์ข้อมูลจากแฟ้มนักศึกษา
      @ 10,5 SAY "พิมพ์ข้อมูลจากแฟ้มนักศึกษา"
      GASKPRT = ASKPRT()
      IF GASKPRT = 0 ; RETURN ; ENDIF
      USE IDSTD
      IF GASKPRT = 1
        CLS
        LIST IDSTD,INAME,ISURN,IMAJOR,ISTAT
        INKEY(0)
```

```
  ENDIF
  IF GASKPRT = 2                                  // ผลออกทั้งจอภาพและเครื่องพิมพ์
    LIST IDSTD,INAME,ISURN,IMAJOR,ISTAT TO PRINTER
    EJECT
  ENDIF
CASE X = 2
// เพิ่มข้อมูลในแฟ้มนักศึกษา
  USE IDSTD
  APPEND BLANK
  @ 10,5 SAY "เพิ่มข้อมูลในแฟ้มนักศึกษาใหม่"
  @ 11,5 SAY "รหัส" GET IDSTD
  @ 12,5 SAY "ชื่อ" GET INAME
  @ 13,5 SAY "สกุล" GET ISURN
  @ 14,5 SAY "สาขา" GET IMAJOR
  @ 15,5 SAY "สถานภาพ" GET ISTAT
  READ
  REPLACE IDSTD WITH IDSTD ,INAME WITH INAME ,ISURN WITH ISURN,;
       IMAJOR WITH IMAJOR,ISTAT WITH ISTAT
CASE X = 3
// ลบข้อมูลจากแฟ้มนักศึกษา
  @ 10,5 SAY "ลบข้อมูลจากแฟ้มนักศึกษา"
  USE IDSTD
  @ 11,5 SAY "รหัสนักศึกษาที่ต้องการลบ : " GET _IDSTD ; READ
  LOCATE FOR IDSTD = _IDSTD
  IF FOUND()
    DELETE
    PACK
  ENDIF
CASE X = 4
// แก้ไขข้อมูลจากแฟ้มนักศึกษา
  @ 10,5 SAY "แก้ไขข้อมูลจากแฟ้มนักศึกษา"
  USE IDSTD
  @ 11,5 SAY "รหัสนักศึกษาที่ต้องการแก้ไข : " GET _IDSTD ; READ
```

```
      LOCATE FOR IDSTD = _IDSTD
      IF FOUND()
        @ 12,5 SAY "ชื่อ" GET INAME
        @ 13,5 SAY "สกุล" GET ISURN
        @ 14,5 SAY "สาขา" GET IMAJOR
        @ 15,5 SAY "สถานภาพ" GET ISTAT
        READ
        REPLACE IDSTD WITH _IDSTD ,INAME WITH INAME ,ISURN WITH ISURN,;
           IMAJOR WITH IMAJOR,ISTAT WITH ISTAT
      ENDIF
  ENDCASE
RETURN
```

: ตัวอย่างที่ 5.35

& ผลลัพธ์

```
// การปรับปรุงแฟ้มวิชา
// โดยเลือกพิมพ์ เพิ่ม ลบ หรือแก้ไขได้
PROCEDURE R0502
  _SUBJ = SPACE(8)
  _TEACHID = 0
  _ROOM = SPACE(4)
  _STIME = SPACE(11)
  _SREGIST := _SLIMIT := 0
  X = ALERT("ปรับปรุงแฟ้มวิชา",{'พิมพ์ข้อมูล','เพิ่ม','ลบ','แก้ไข'})
  DO CASE
    CASE X = 0 ; WAIT "ไม่ได้เลือกตัวเลือกใดเลย"
    CASE X = 1
    // พิมพ์ข้อมูลจากแฟ้มวิชา
      @ 4,5 SAY "พิมพ์ข้อมูลจากแฟ้มวิชา"
      GASKPRT = ASKPRT()
      IF GASKPRT = 0 ; RETURN ; ENDIF
      USE SUBJECT
      IF GASKPRT = 1
        AR = {"SUBJ","TEACHID","ROOM","STIME","SREGIST","SLIMIT"}
        DBEDIT(5,5,20,75,AR)
```

```
  ENDIF
  IF GASKPRT = 2                                  // ผลออกทั้งจอภาพและเครื่องพิมพ์
    LIST SUBJ,TEACHID,ROOM,STIME,SREGIST,SLIMIT
    EJECT
  ENDIF
CASE X = 2
// เพิ่มข้อมูลในแฟ้มวิชา
  USE SUBJECT
  @ 10,5 SAY "เพิ่มข้อมูลในแฟ้มวิชาใหม่"
  @ 11,5 SAY "รหัสวิชา" GET _SUBJ
  @ 12,5 SAY "รหัสอาจารย์ผู้สอน" GET _TEACHID
  @ 13,5 SAY "ห้องเรียน" GET _ROOM
  @ 14,5 SAY "เวลาเรียน" GET _STIME
  @ 15,5 SAY "จำนวนที่ลงทะเบียน" GET _SREGIST
  @ 16,5 SAY "จำนวนสูงสุดที่อนุญาติ" GET _SLIMIT
  READ
  IF ASKUPD() = 1
    APPEND BLANK
    REPLACE SUBJ WITH _SUBJ,TEACHID WITH _TEACHID,ROOM WITH _ROOM,;
          STIME WITH _STIME,SREGIST WITH _SREGIST,SLIMIT WITH _SLIMIT
  ENDIF
CASE X = 3
// ลบข้อมูลจากแฟ้มวิชา
  @ 10,5 SAY "ลบข้อมูลจากแฟ้มวิชา"
  USE SUBJECT
  @ 11,5 SAY "รหัสวิชาที่ต้องการลบ : " GET _SUBJ ; READ
  LOCATE FOR SUBJ = _SUBJ
  IF FOUND()
    @ 12,5 SAY SUBJ
    @ 13,5 SAY TEACHID
    @ 14,5 SAY ROOM
    @ 15,5 SAY STIME
    @ 16,5 SAY SREGIST
```

```
        @ 17,5 SAY SLIMIT
        IF ASKUPD() = 1
          DELETE ; PACK
        ENDIF
      ENDIF
    CASE X = 4
    // แก้ไขข้อมูลจากแฟ้มวิชา
      @ 10,5 SAY "แก้ไขข้อมูลจากแฟ้มวิชา"
      USE SUBJECT
      @ 11,5 SAY "รหัสวิชา : " GET _SUBJ ; READ
      LOCATE FOR SUBJ = _SUBJ
      IF FOUND()
        _SUBJ = SUBJ
        _TEACHID = TEACHID
        _ROOM = ROOM
        _STIME = STIME
        _SREGIST = SREGIST
        _SLIMIT = SLIMIT
        @ 11,5 SAY "รหัสวิชา : " GET _SUBJ
        @ 12,5 SAY "รหัสอาจารย์ผู้สอน" GET _TEACHID
        @ 13,5 SAY "ห้องเรียน" GET _ROOM
        @ 14,5 SAY "เวลาเรียน" GET _STIME
        @ 15,5 SAY "จำนวนที่ลงทะเบียน" GET _SREGIST
        @ 16,5 SAY "จำนวนสูงสุดที่อนุญาติ" GET _SLIMIT
        READ
        IF ASKUPD() = 1
          REPLACE SUBJ WITH _SUBJ,TEACHID WITH _TEACHID,;
                  ROOM WITH _ROOM,STIME WITH _STIME,;
                  SREGIST WITH _SREGIST,SLIMIT WITH _SLIMIT
        ENDIF
      ENDIF
  ENDCASE
RETURN
```

: ตัวอย่างที่ 5.36

& ผลลัพธ์

```
// การปรับปรุงแฟ้มผู้สอน
// โดยเลือกพิมพ์ เพิ่ม ลบ หรือแก้ไขได้
PROCEDURE R0503
  USE TEACHER
  BROWSE(5,5,22,75)
  PACK
RETURN
```

: ตัวอย่างที่ 5.37

& ผลลัพธ์

```
// การปรับปรุงแฟ้มลงทะเบียน
// โดยเลือกจัดเรียงข้อมูลด้วยคำสั่ง SORT
// ปรับปรุงข้อมูลโดยใช้คำสั่ง BROWSE
PROCEDURE R0504
  @ 4,5 SAY "การปรับปรุงแฟ้มลงทะเบียน"
  @ 9,5 SAY "สั่งจัดเรียงก่อนปรับปรุงข้อมูล"
  @ 10,5 PROMPT "1. จัดเรียงตามปี ภาค รหัสนักศึกษา และวิชา"
  @ 11,5 PROMPT "2. จัดเรียงตามวิชา ปี ภาค และรหัสนักศึกษา"
  @ 12,5 PROMPT "3. จัดเรียงตามรหัสนักศึกษา ปี ภาค และวิชา"
  @ 13,5 PROMPT "4. จัดเรียงตามเกรด ปี ภาค วิชา และรหัสนักศึกษา"
  @ 14,5 PROMPT "5. จัดเรียงตามรหัสนักศึกษา และวิชา จากมากไปน้อย"
  @ 15,5 PROMPT "6. ไม่จัดเรียงใหม่"
  MENU TO _SORTORDER
  IF _SORTORDER > 0 .AND. _SORTORDER < 6
    USE REGIST
    DO CASE
      CASE _SORTORDER = 1
        SORT ON RYEAR,RSEM,IDSTD,SUBJ TO TEMPF
      CASE _SORTORDER = 2
        SORT ON SUBJ,RYEAR,RSEM,IDSTD TO TEMPF
      CASE _SORTORDER = 3
        SORT ON IDSTD,RYEAR,RSEM,SUBJ TO TEMPF
      CASE _SORTORDER = 4
        SORT ON GRADE,RYEAR,RSEM,SUBJ,IDSTD TO TEMPF
```

```
      CASE _SORTORDER = 5
        SORT ON IDSTD/D ,SUBJ TO TEMPF
      ENDCASE
      USE TEMPF
      COPY TO REGIST
      CLOSE
  ENDIF
  USE REGIST
  BROWSE(5,5,22,75)
  PACK
RETURN
```

: ตัวอย่างที่ 5.38

& ผลลัพธ์

```
// การปรับปรุงแฟ้มห้อง
// โดยเลือกพิมพ์ เพิ่ม ลบ หรือแก้ไขได้
PROCEDURE R0505
  _ROOM = SPACE(4)
  _RCSEQ = 0
  _RCHAIR = SPACE(2)
  X = ALERT("ปรับปรุงแฟ้มห้อง",{'พิมพ์ข้อมูล','เพิ่ม','ลบ','แก้ไข'})
  DO CASE
    CASE X = 0 ; WAIT "ไม่ได้เลือกตัวเลือกใดเลย"
    CASE X = 1
    // พิมพ์ข้อมูลจากแฟ้มห้อง
      @ 4,5 SAY "พิมพ์ข้อมูลจากแฟ้มห้อง"
      GASKPRT = ASKPRT()
      IF GASKPRT = 0 ; RETURN ; ENDIF
      USE ROOM
      IF GASKPRT = 1
        DISPLAY ROOM,RCSEQ,RCHAIR ALL
        INKEY(5)
      ENDIF
      IF GASKPRT = 2                          // ผลออกทั้งจอภาพและเครื่องพิมพ์
        FOR I = 1 TO  RECCOUNT()
```

```
        DISPLAY I,ROOM,RCSEQ,RCHAIR TO PRINTER
        SKIP
      NEXT
      EJECT
    ENDIF
  CASE X = 2
  // เพิ่มข้อมูลในแฟ้มห้อง
    USE ROOM
    @ 10,5 SAY "เพิ่มข้อมูลในแฟ้มห้องใหม่"
    SET FORMAT TO R0505FORM
    READ
    SET FORMAT TO
    IF ASKUPD() = 1
      APPEND BLANK
      REPLACE ROOM WITH _ROOM,RCSEQ WITH _RCSEQ,RCHAIR WITH _RCHAIR
    ENDIF
  CASE X = 3
  // ลบข้อมูลจากแฟ้มห้อง
    @ 9,5 SAY "ลบข้อมูลจากแฟ้มห้อง"
    USE ROOM
    @ 10,5 SAY "รหัสห้องที่ต้องการลบ : " GET _ROOM
    @ 11,5 SAY "ลำดับเก้าอี้ : " GET _RCSEQ
    READ
    LOCATE FOR ROOM = _ROOM .AND. RCSEQ = _RCSEQ
    IF FOUND()
      @ 12,5 SAY RCHAIR
      IF ASKUPD() = 1
        DELETE ; PACK
      ENDIF
    ENDIF
  CASE X = 4
  // แก้ไขข้อมูลจากแฟ้มห้อง
    @ 7,5 SAY "แก้ไขข้อมูลจากแฟ้มห้อง"
```

```
      USE ROOM
      @ 8,5 SAY "รหัสห้องที่ต้องการแก้ไข : " GET _ROOM
      @ 9,5 SAY "ลำดับเก้าอี้ : " GET _RCSEQ
      READ
      LOCATE FOR ROOM = _ROOM .AND. RCSEQ = _RCSEQ
      IF FOUND()
        _ROOM = ROOM
        _RCSEQ = RCSEQ
        _RCHAIR = RCHAIR
        SET FORMAT TO R0505FORM
        READ
        SET FORMAT TO                    // ต้องยกเลิก FORMAT เพราะป้องกันการ READ แบบอื่น
        IF ASKUPD() = 1
          REPLACE ROOM WITH _ROOM,RCSEQ WITH _RCSEQ,RCHAIR WITH _RCHAIR
        ENDIF
      ENDIF
  ENDCASE
  PROCEDURE R0505FORM
  @ 10,5 SAY "ห้องเรียน" GET _ROOM
  @ 11,5 SAY "ลำดับของเก้าอี้" GET _RCSEQ
  @ 12,5 SAY "เลขที่เก้าอี้" GET _RCHAIR
  RETURN
RETURN
```