

## BIBLIOGRAPHY

1. Donald E. Knuth. The Art of Computer Programming : Sorting and Searching. vol.2 Philippines : Addison-Wesley Publishing Co., 1973.
2. IBM, DOS/VS Sort/Merge Logic. Form LY 33-8038, Bangkok : IBM Co.,Ltd.  
IBM Co., Ltd. (Thailand).
3. IBM, DOS/VS Sort/Merge Programmer's Guide. Form SC 33-4028-1,  
Bangkok : IBM Co., Ltd. (Thailand).
4. IBM, DOS/VS System Control Statement. Form GC 33-5376, Bangkok :  
IBM Co., Ltd. (Thailand).
5. IBM, DOS/VS System Management Guide. Form GC 33-5371-6, Bangkok :  
IBM Co., Ltd. (Thailand).
6. IBM, DOS/VS Sort/Merge Installation Reference Manual. Form SC  
SC 33-4026, Bangkok : IBM Co., Ltd. (Thailand).
7. IBM, DOS Full American National Standard COBOL. Form GC 28-6394-6,  
Bangkok : IBM Co.,Ltd. (Thailand).

ภาคผนวก ก

หน้าที่ของแต่ละโมดูลสำหรับการจัดรวมข้อมูลอย่างมีลำดับ

1. เฟส 0

โมดูล	ถูกเรียก จากโมดูล	โมดูล ที่เรียก	จุดออก	หน้าที่
RTMG	Supervisor	RCA RCM RCB RCI RCJ RCK RCØ RCQ RCP RCS RCT RCD RCC RCG RCE, RCF, RCH	จะทำงานเฟส 1 หรือ เฟส 3 ต่อไป ถ้าเกิด ผิดพลาดขึ้นจะส่งการควบคุมกลับไปยังโปรแกรม เรียก	ทำการควบคุมเฟส 0 ทดสอบ ว่าโปรแกรมสำเร็จรูปถูกโหลด อย่างไร (โดยทางคำสั่ง LOAD และ BALR หรือคำสั่ง EXEC) รักษาค่ารีจิสเตอร์

โมดูล	ถูกเรียก จากโมดูล	โมดูล ที่เรียก	จุดออก	หน้าที่
RCA	RTMG	RCM	กลับไปยังโมดูล RTMG	คำสั่งควบคุมเครื่องอ่านข้อมูล - อ่านคำสั่งควบคุมจาก SYSIPT - รวบรวมคำสั่งและเก็บยังพจนานุกรมคำสั่งควบคุม - เขียนคำสั่งบน SYSLST ถ้าต้องการ
RCL	ถูกเรียกจากคำสั่งที่ต้องการประมวลผลทั้งหมด	-	ขึ้นอยู่กับตัวเรียกโมดูล	ตัวพิจารณาคำสั่งควบคุม - ทำการพิจารณารูปคำสั่งในพจนานุกรมคำสั่งควบคุม, ทำการแปลเขตข้อมูล, เปลี่ยนข้อมูลให้เป็นเลขฐาน 2 และคำนวณหาชนิดของคำสั่งควบคุม
RCD	RTMG	-	คืนกลับโมดูล RTMG	การตัดแยกข้อมูลและคำนวณขนาดของบิน (BIN) - คำนวณหาเขตควบคุมข้อมูลที่จะถูกทำการตัดแยก - คำนวณขนาดที่ต้องการของชุดคำสั่งที่ทำการตัดแยก - คำนวณขนาดของบินที่เหมาะสมสำหรับระเบียบข้อมูลนำเข้า

โมดูล	ถูกเรียก จากโมดูล	โมดูล ที่เรียก	จุดออก	หน้าที่
RCC	RTMG	RCN	คืนกลับโมดูล RTMG	<p>การค้นหาของระบบ</p> <ul style="list-style-type: none"> <li>- รับข่าวสารที่ได้มาจากส่วน ความจำหลัก</li> <li>- กำหนดสื่อที่ใช้ในการรับข้อมูล เข้าและออก</li> <li>- ตรวจสอบค่าที่เป็นไปได้และ เก็บเอาไว้ในส่วนของพีพีไอ</li> <li>- ทำการเปิดแฟ้มข้อมูล SORT x WK บนจานแม่เหล็ก</li> </ul>
RCE <sup>1</sup> , REF <sup>2</sup> , RCH <sup>3</sup>	RTMG	-	คืนกลับโมดูล RTMG	<p>คำนวณค่า B/G คำนวณค่าคงที่</p> <ul style="list-style-type: none"> <li>- บล็อกกิ่งแพคเตอร์สำหรับการ จัดเรียงลำดับ (B)</li> <li>- ขนาดของส่วนความจำในเฟลล 1 (G)</li> <li>- ลำดับของการจัดรวมข้อมูล อย่างมีลำดับ</li> <li>- จำนวนของหน่วยความจำ ชั่วคราวของแต่ละเฟลล</li> <li>- จำนวนระเบียบข้อมูลที่จะ สามารถถูกจัดเรียงได้ (N<sub>max</sub>)</li> </ul>

โมเดล	ถูกเรียก จากโมเดล	โมเดล ที่เรียก	จุดออก	หน้าที่
RCG	RTMG	-	คืนกลับโมเดล RTMG	<p>ต้องสร้างชุดคำสั่งคัดแยกข้อมูลที่ต้องการ</p> <p>- สร้างรหัสสำหรับชุดคำสั่งในการคัดแยกข้อมูล</p>

2. เฟส 3

โมดูล	โมดูลที่เรียก	เรียกโมดูล	จุดออก	หน้าที่
RSM	RTMG (MERGE)	assignment	เมื่อ assignment function	ดึงโมดูลต่าง ๆ ของเฟส 3
	RSJ (SORT)	module	เรียบร้อยแล้วจะส่งให้ ASP	- ดึงโมดูลต่าง ๆ - ดึงชุดคำสั่ง โปรแกรมที่ต้องการใช้
ASK, ASL	RSM	-	คืนกลับโมดูล RSM	การจัดวางตำแหน่งในส่วนความจำหลักและสร้างตัวควบคุมบล็อก - กำหนดหน่วยความจำชั่วคราวสำหรับการนำข้อมูลเข้าและออก - กำหนดบล็อก เริ่มต้นของข้อมูลนำเข้าและออก - เปิดแฟ้มข้อมูล
ASP	RSM	-	ไปยัง โมดูลที่ทำกาารจัดรวมข้อมูล	ชุดคำสั่ง เริ่มต้นการ เรียงลำดับ - เริ่มต้นทำการจัดลำดับสำหรับโมดูลที่ทำกาารจัดรวมข้อมูล
RØC <sup>4</sup> , RØE <sup>5</sup> , RØG <sup>6</sup>	ASP	- Extract - ชุดคำสั่งอ่าน - ชุดคำสั่งบล็อก	คืนกลับโมดูล RBG, RBI, RBJ	ทำการจัดรวมข้อมูลอย่างมีลำดับ - ดึงระเบียบข้อมูลจากแต่ละแฟ้มข้อมูลนำเข้ามาทำการจัดรวมข้อมูลอย่างมีลำดับ

โมดูล	โมดูลที่เรียก	เรียกโมดูล	จุดออก	หน้าที่
RBG <sup>7</sup> , RBI <sup>8</sup> , RBJ <sup>9</sup>	RØC, RØE, RØG	- ชุดคำสั่งอ่าน - ชุดคำสั่ง บันทึก - E32, E35	คืนกลับโมดูล RØC, RØE, RØG	ชุดคำสั่งทำดีบล็อกและบล็อก - ทำการดีบล็อกข้อมูลนำเข้า และ ส่งให้อยู่ในรูปที่ใช้งานได้
RGF <sup>1</sup> , RGG <sup>2</sup> RGH <sup>10</sup> , RGJ <sup>11</sup>	RBG, RBI, RBJ	E31, E38	คืนกลับโมดูล RBG, RBI, RBJ	ชุดคำสั่งการอ่าน - อ่านข้อมูลนำเข้าจาก SORTIN - ตรวจสอบข้อผิดพลาดที่จำเป็น - สำหรับระเบียบข้อมูลที่ความ ยาวไม่คงที่จะเพิ่ม RDW เข้า ไปในแต่ละระเบียบข้อมูล
RPE <sup>7</sup> , RPF <sup>12</sup> , RPG <sup>13</sup>	RBG, RBI RBJ	E39	ถ้าทำงานเรียบร้อยจะ กลับไปยังโมดูล RBG, RBI, RBJ หรือถ้าจบ การทำงานจะไปยังโมดูล RSN	ชุดคำสั่งบันทึก - บันทึกข้อมูลที่ถูกจัดรวมลำดับ ครั้งสุดท้ายไปยังแฟ้มข้อมูล นำออก - ตรวจสอบข้อผิดพลาดที่จำเป็น - สำหรับระเบียบข้อมูลที่ความ ยาวไม่คงที่จะเพิ่ม RDW เข้า ไปในแต่ละระเบียบข้อมูล
RSN	เฟลล์ 1, 2, 3 (ถ้า เกิดผิด)	- RMC, RMD - RSR - E37	ส่งกลับให้ supervisor	จุดจบของการทำงาน - คำนวณจำนวนของระเบียบ ข้อมูลที่จะถูกประมวลผล

โมดูล	โมดูลที่เรียก	เรียกโมดูล	จุดออก	หน้าที่
RMC, RMD	เพลส 1,2,3 (ถ้าต้องการ พิมพ์ข้อความ)	-	ส่งกลับไปยังชุดคำสั่งที่ เรียก	<ul style="list-style-type: none"> <li>- พิมพ์ข้อความที่เป็นสาเหตุของ การตัดการติดต่อ</li> <li>- ปิดแฟ้มข้อมูล</li> <li>- ออกคำสั่งให้หยุดการทำงาน หรือตัดการติดต่อกับคำสั่ง macro (EXEC)</li> </ul> <p>ชุดคำสั่งพิมพ์พร้อมและข่าวลสารของ โมดูล</p> <ul style="list-style-type: none"> <li>- โมดูล RMC จะเริ่มต้นเก็บข่าว ลสารที่ต้องการในโมดูลที่ใช้เก็บ ข่าวลสารโมดูล RMD และบันทึก ข่าวลสารเหล่านั้น</li> </ul>
RSR	RSN (เมื่อ ERASE กำหนด)	-	กลับคืนโมดูล RSN	การลบกลุ่มข้อมูลสำหรับกวระ่ลำดับ เรียงลำดับ



1. ใช้สำหรับเทปแม่เหล็ก (SORT)
2. ใช้สำหรับสื่อที่เข้าถึงข้อมูลโดยตรง (SORT)
3. ใช้สำหรับการจัดรวมข้อมูล
4. ใช้สำหรับเทปแม่เหล็กเมื่อ  $M > 5$  (SORT/MERGE)
5. ใช้สำหรับเทปแม่เหล็กเมื่อ  $M \leq 5$  (SORT/MERGE)
6. ใช้สำหรับสื่อที่เข้าถึงข้อมูลโดยตรง (MERGE)
7. ใช้สำหรับเทปแม่เหล็ก (SORT/MERGE)
8. ใช้สำหรับจานแม่เหล็กที่มีความยาวระเบียบข้อมูลไม่คงที่
9. ใช้สำหรับจานแม่เหล็กที่มีความยาวระเบียบข้อมูลคงที่
10. ใช้สำหรับ SAM (MERGE)
11. ใช้สำหรับ VSAM (MERGE)
12. ใช้สำหรับจานแม่เหล็กที่จัดแบบ SAM (SORT/MERGE)
13. ใช้สำหรับ VSAM (SORT/MERGE)

ภาคผนวก ข

คำสั่งควบคุม

โปรแกรมสำเร็จรูปสำหรับการคัดรวมข้อมูลอย่างมีลำดับจะทำการประมวลผลข้อมูลได้จะต้องได้รับคำสั่งควบคุมก่อน ซึ่งคำสั่งควบคุมจะทำหน้าที่ในการบอกโปรแกรมสำเร็จรูปให้รู้ถึง

- ประเภทของการทำงานที่ต้องการ
- พารามิเตอร์เขตควบคุม
- การแก้ไขเปลี่ยนแปลงโดยชุดคำสั่งของผู้ใช้
- หน้าที่ในการทำงาน
- แฟ้มข้อมูลนำเข้าและออก
- การเลือกใช้คำสั่งที่เหมาะสมกับงานแต่ละชนิด

รูปแบบคำสั่งควบคุม

คำสั่งควบคุมสำหรับการคัดรวมข้อมูลอย่างมีลำดับจะมีลักษณะดังนี้

ลำดับที่ 1 เป็นช่องว่าง ↓	ลำดับที่ 72 การต่อบัตร ↓
↑ ลำดับที่ 2 - 71 คำสั่งควบคุม	↑ ลำดับที่ 73 - 80 ลำดับบัตร

ความหมายของสัญลักษณ์สำหรับคำสั่งควบคุม

- {----} เครื่องหมายวงเล็บปีกกา จะกำหนดส่วนที่จะต้องเขียนในคำสั่งควบคุม แต่สามารถจะเลือกรูปแบบใดรูปแบบหนึ่งภายในเครื่องหมายวงเล็บปีกกา
- (----) เครื่องหมายวงเล็บกำมปู จะหมายถึงพารามิเตอร์ภายในวงเล็บกำมปูจะเลือกใช้หรือไม่ก็ได้
- (----) เครื่องหมายวงเล็บ จะหมายถึงกลุ่มของตัวแปร

- \_\_\_\_\_ ชัดเส้นใต้ภายในคำสั่ง จะหมายถึง ถ้าพารามิเตอร์ตัวนั้นไม่ถูกเลือกใช้ มันจะกำหนดค่าที่ชัดเจนไว้ให้โดยอัตโนมัติ

คำสั่งควบคุม

ส่วนนี้จะกล่าวถึง ตัวพารามิเตอร์และการเลือกใช้คำสั่งควบคุมทั้ง 11 ตัว ที่ยอมรับโดยโปรแกรมสำเร็จรูป ดังต่อไปนี้

1. คำสั่ง MERGE

คำสั่ง MERGE จะต้องมีเสมอ เมื่อต้องการเรียกใช้โปรแกรมการคัดรวมข้อมูลอย่างมีลำดับ

$$\text{รูปแบบ : MERGE } \left\{ \begin{array}{l} \text{FIELDS} = (p_1, m_1, f_1, s_1, \dots, p_{12}, m_{12}, f_{12}, p_{12}) \\ \text{FIELDS} = (p_1, m_1, s_1, \dots, p_{12}, m_{12}, s_{12}, \text{FORMAT} = \text{xx/xxx}) \end{array} \right\} \left\{ \begin{array}{l} \text{FILES} = n \\ \text{ORDER} = n \end{array} \right\}$$

- FIELDS จะบอกรายละเอียดของเขตควบคุม มีตัวแปรดังนี้
  - p จะเป็นตัวกำหนดจุดเริ่มต้นของเขตควบคุม
  - m จะเป็นตัวกำหนดความยาวของเขตควบคุม
  - f จะเป็นตัวกำหนดรูปแบบของข้อมูลในเขตควบคุม
  - s จะเป็นตัวกำหนดว่าเขตควบคุมจะถูกคัดลำดับอย่างไร
- FORMAT จะเป็นตัวกำหนดรูปแบบข้อมูลเช่นเดียวกับ f แต่จะใช้ในกรณีที่เขตควบคุมมีรูปแบบเดียวกันหมด
- FILES บอกจำนวนแฟ้มข้อมูลนำเข้า n มีค่าตั้งแต่ 1 - 8 หรือจะใช้ ORDER ก็ได้

2. คำสั่ง RECORD

จะต้องมีเสมอ ใช้กำหนดลักษณะของระเบียบข้อมูลที่จะถูกคัดรวมอย่างมีลำดับ

$$\text{รูปแบบ : RECORD TYPE} = \left\{ \begin{array}{l} \text{F} \\ \text{V} \\ \text{D} \end{array} \right\}, \text{LENGTH} = (l_1, l_2, l_3, l_4, l_5) (, \text{DELBLANK} = ((p_p, q)))$$

- TYPE บอกให้รู้ลักษณะของระเบียบข้อมูลนำเข้ามี
  - F สำหรับระเบียบข้อมูลที่มีความยาวคงที่
  - V สำหรับระเบียบข้อมูลที่มีความยาวไม่คงที่, ยกเว้น ASCII
  - D สำหรับระเบียบข้อมูลที่มีความยาวไม่คงที่, เฉพาะ ASCII
- LENGTH บอกให้รู้ความยาวของระเบียบข้อมูลนำเข้า
  - $1_1$  เป็นความยาวของแต่ละระเบียบในแฟ้มข้อมูลนำเข้า แต่ถ้า TYPE เป็น V หรือ D จะเป็นความยาวของระเบียบข้อมูลอันที่ยาวที่สุด
  - $1_2$  เป็นความยาวของแต่ละระเบียบที่ถูกจัดเรียงลำดับแล้ว แต่ถ้า TYPE เป็น V หรือ D จะเป็นความยาวของระเบียบข้อมูลที่ยาวที่สุดที่จะถูกจัดเรียงลำดับ
  - $1_3$  เป็นความยาวของแต่ละระเบียบในแฟ้มข้อมูลนำออก แต่ถ้า TYPE เป็น V หรือ D จะเป็นความยาวของระเบียบที่ยาวที่สุด
  - $1_4$  จะใช้สำหรับ TYPE เป็น V หรือ D ใช้ตรวจสอบความยาวของระเบียบข้อมูล
  - $1_5$  จะใช้สำหรับ TYPE เป็น V หรือ D จะเป็นความยาวของระเบียบข้อมูลที่มีความยาวเท่ากันเป็นจำนวนมากที่สุดของแฟ้มข้อมูลนำเข้า
- DELBLANK จะใช้เลือกระเบียบข้อมูลที่ประกอบด้วยตัวอักษร q ที่ตำแหน่ง p หรือ ถ้า q ไม่กำหนดจะเป็นช่องว่าง ซึ่งข้อมูลนั้นจะไม่ถูกอ่าน

### 3. คำสั่ง MØDS

ใช้กำหนดว่าโปรแกรมสำเร็จรูปจะถูกแก้ไขอย่างไร โดยชุดคำสั่งของผู้ใช้ มันจะกล่าวถึงจุดออกของโปรแกรมในแต่ละเฟลล์

รูปแบบ : MØDS PH<sub>n</sub> = (name, loading information, exit1, exit2, ---, exitn), ---

- PH<sub>n</sub> จะเป็นตัวบอกถึงเฟลล์ต่าง ๆ โดย
  - PH<sub>1</sub> จะหมายถึง internal sort phase
  - PH<sub>2</sub> จะหมายถึง external sort phase
  - PH<sub>3</sub> จะหมายถึง final merge phase

- name จะอ้างอิงถึงชื่อที่ถูกสร้างไว้ใน core image library ถ้าไม่กำหนดชุดคำสั่งของผู้ใช้จะทำงานในส่วนของความจำหลัก
- loading information จะอธิบายถึงชุดคำสั่งที่จะออกจากโปรแกรมที่อาจจะถูกดึงเข้าไปในส่วนของความจำหลักโดยโปรแกรมสำเร็จรูป
- exit จะกำหนดจุดที่โปรแกรมของผู้ใช้ถูกเรียกใช้

#### 4. คำสั่ง INPFIL

ใช้กำหนดแฟ้มข้อมูลนำเข้าสำหรับการจัดรวมข้อมูลอย่างมีลำดับ

รูปแบบ : INPFIL (BLKSIZE = n) (, BYPASS) (, EXIT) (, PRESEQ) (, VSAM) (, TØL)

$$\left( \begin{array}{l} ,VØLUME=n \\ ,VØLUME=(n, ---) \end{array} \right) \left( \begin{array}{l} ,OPEN=RWD \\ ,ØPEN=NØRWD \end{array} \right) \left( \begin{array}{l} ,CLØSE=RWD \\ ,CLØSE=UNLD \\ ;CLØSE=NØRWD \end{array} \right) \left( \begin{array}{l} ,DATE=E \\ ,DATE=A, BUFOFF=n \end{array} \right)$$

- BLKSIZE จะกำหนดขนาดบล็อกของระเบียบข้อมูลนำเข้า (หน่วย : ไบท์)
- BYPASS จะกำหนดให้โปรแกรมอ่านข้ามระเบียบข้อมูลที่เกิดการผิดพลาดแล้วทำงานต่อไป แต่ถ้าไม่กำหนดจะหยุดการทำงานทันทีถ้าข้อมูลเกิดการผิดพลาด
- VØLUME กำหนดจำนวนลือที่ใช้เก็บข้อมูลของแต่ละแฟ้มข้อมูลที่ไม่มี lable
- ØPEN กำหนดให้เทปแม่เหล็กม้วนแรกก่อนใช้ให้กรอเทปกส์หรือไม่ก่อนจะทำการอ่าน
- CLØSE กำหนดให้เทปแม่เหล็กเมื่ออ่านจบแล้วจะกรอเทปกส์หรือทั้งกรอเทปกส์และเปิดตู้เทปหรือไม่กรอ

- DATA กำหนดชนิดของข้อมูลนำเข้าว่าเป็น EBCDIC หรือ ASCII
- BUFOFF กำหนดขนาดของบล็อกแรกที่จะพบตอนต้นของแต่ละระเบียบข้อมูลบนแฟ้มข้อมูลนำเข้า

- VSAM จะชี้ว่าแฟ้มข้อมูลนำเข้ามีลักษณะการจัดองค์กรแบบ VSAM
- TØL ใช้สำหรับแฟ้มข้อมูลนำเข้าที่เป็น VSAM เท่านั้น
- PRESEQ จะชี้ว่าแฟ้มข้อมูลนำเข้าเกือบทั้งหมดถูกจัดลำดับเรียบร้อยแล้ว

5. คำสั่ง ØUTFIL

ใช้กำหนดแฟ้มข้อมูลนำออกและกำหนดการทำงานเมื่อแฟ้มข้อมูลที่เป็นเทปแม่เหล็กถูก เปิด หรือปิด

รูปแบบ :  $\text{ØUTFIL}(\text{BLKSIZE}=\text{n}) (\text{,EXIT}) (\text{,NØTPMK}) \left\{ \begin{array}{l} \text{,ØPEN=RWD} \\ \text{,ØPEN=NØRWD} \end{array} \right\} \left\{ \begin{array}{l} \text{,CLØSE=RWD} \\ \text{,CLØSE=UNLD} \\ \text{,CLØSE =NØRWD} \end{array} \right\}$

$\left\{ \begin{array}{l} \text{,KSDS} \\ \text{,RRDS} \\ \text{,ESDS} \end{array} \right\} (\text{,TØL}) (\text{,BUFØFF}=\text{n}) (\text{,REUSE})$

- BLKSIZE กำหนดขนาดของบล็อกที่ใหญ่ที่สุดของแฟ้มข้อมูลนำออก ถ้าไม่กำหนดจะเป็นแบบไม่มีบล็อก
- EXIT กำหนดชุดคำสั่งที่จะรับแฟ้มข้อมูลนำออกแทนโปรแกรมสำเร็จรูป
- NØTPMK บอกให้รู้ว่าแฟ้มข้อมูลนำออกแต่ละตัวจะไม่มีกาเขียน tape mark
- ØPEN บอกให้รู้ว่าเทปข้อมูลนำออกจะอยู่ในตำแหน่งใดก่อนที่ข้อมูลจะถูกบันทึก
- CLØSE บอกให้รู้ว่าเทปข้อมูลนำออกจะอยู่ในตำแหน่งใดหลังจากบันทึกข้อมูลเสร็จ
- BUFØFF กำหนดเขียนบล็อกหน้าอยู่ตอนหน้าของผลลัพธ์แต่ละบล็อก
- KSDS, RRDS, ESDS ชี้ว่าแฟ้มข้อมูลนำออกจะถูกเก็บอยู่ใน VSAM
- REUSE ใช้เมื่อต้องการบันทึกข้อมูลมากกว่าที่ให้บันทึกใน VSAM
- TØL สำหรับเฉพาะ VSAM มันจะชี้ว่าโปรแกรมสำเร็จรูปจะเปิดบริเวณพื้นที่เก็บข้อมูลนำออกและ VSAM จะสร้างรหัสเตือนกลับมาและทำงานต่อไป

6. คำสั่ง INCLUDE/ØMIT

ใช้กำหนดเงื่อนไขตรวจสอบระเบียบข้อมูลนำเข้าว่าข้อมูลตัวใดที่จะมีสิทธิ์ผ่านโปรแกรมสำเร็จรูปหรือว่าตัดทิ้งไป ถ้าคำสั่งนี้ไม่ใช่ทุก ๆ ระเบียบข้อมูลจะถูกจัดรวมอย่างมีลำดับหมด

รูปแบบ :  $\left( \begin{array}{l} \text{INCLUDE} \\ \text{ØMIT} \end{array} \right) \text{COND} = (\text{logical expression}) (, \text{FØRMAT} = \text{xx/xxx})$

- COND กำหนดการเปรียบเทียบทางตรรก

$\text{COND} = (\text{relational condition}_1 \left( \left\{ \begin{array}{l} \text{AND} \\ \text{OR} \end{array} \right\} \text{relational condition}_2 \text{---} \right))$

รูปแบบของ relational condition

$p_1, m_1(, f_1) \left\{ \begin{array}{l} \text{EQ} \\ \text{NE} \\ \text{GT} \\ \text{GE} \\ \text{LT} \\ \text{LE} \end{array} \right\} \left\{ \begin{array}{l} p_2, m_2(, f_2) \\ \text{self-defining term} \end{array} \right\}$

7. คำสั่ง ALTSEQ

ใช้กำหนดให้ลิสต์กลุ่มลำดับที่จะใช้โดยโปรแกรมสำเร็จรูปในการเปรียบเทียบเขตควบคุมหรือสำหรับการเปรียบเทียบเขตที่กำหนดในคำสั่ง INCLUDE/ØMIT ถ้าคำสั่งนี้ไม่ใช่กลุ่มลำดับแบบ EBCDIC จะถูกเรียกใช้

รูปแบบ :  $\text{ALTSEQ COND} = (\text{fftt}(, \text{fftt} \text{---}))$

- ff แทนตัวเลขฐาน 16 จำนวน 2 ตัว ใช้กำหนดตัวอักษรเก่าที่จะถูกแทนที่

- tt แทนตัวเลขฐาน 16 จำนวน 2 ตัว ใช้กำหนดตัวอักษรใหม่ที่จะมาแทนที่ตัวอักษรเก่า

8. คำสั่ง ØUTRBC

ใช้จัดรูปแบบใหม่ของระเบียบข้อมูลนำเข้า กำหนดส่วนของระเบียบข้อมูลนำเข้าซึ่งจะถูกรวมอยู่ในระเบียบข้อมูลนำออก จะถูกจัดอย่างไร

รูปแบบ :  $\text{ØUTREC FIELDS} = (p_1, m_1(, a_1) \text{---}, p_n(, m_n(, a_n)))$

- p เป็นจุดเริ่มต้นของเขตในระเบียนข้อมูลนำเข้า ซึ่งจะกลายเป็นส่วนของระเบียนข้อมูลนำออก

- m เป็นความยาวของเขตรวมเครื่องหมายด้วย

- a กำหนดการตัดแถวของข้อมูลในระเบียนข้อมูลนำออก สัมพันธ์กับจุดเริ่มต้นของระเบียนข้อมูลนำออก

### 9. คำสั่ง SUM

ใช้ขั้วเขตที่เป็นตัวเลขในระเบียนข้อมูลนำเข้า ให้เป็นเขตละลุ่มและกำหนดว่าเมื่อไรก็ตามที่ระเบียนข้อมูล 2 ระเบียนมีเขตควบคุมเท่ากัน ข้อความของเขตละลุ่มจะถูกเพิ่มขึ้นผลรวมจะใส่ไว้ในระเบียนข้อมูลหนึ่ง และระเบียนข้อมูลอื่นจะถูกลบทิ้ง

รูปแบบ : 
$$\text{SUM} \left\{ \begin{array}{l} \text{FIELDS} = (p_1, m_1, f_1, \dots, p_n, m_n, f_n) \\ \text{FIELDS} = (p_1, m_1, \dots, p_n, m_n), \text{FORMAT} = \text{xx} \end{array} \right\}$$

- p เป็นจุดเริ่มต้นของเขตละลุ่ม

- m เป็นความยาวของเขตละลุ่ม

- f เป็นรูปแบบข้อมูลในเขตละลุ่ม

### 10. คำสั่ง OPTION

ทำหน้าที่กำหนดค่าตัวแปรต่าง ๆ ให้เลือกใช้ในการทำงานของโปรแกรมสำเร็จรูป

รูปแบบ : 
$$\text{OPTION} \left( \begin{array}{l} \text{PRINT} \\ \text{PRINT=NONE} \\ \text{PRINT=ALL} \\ \text{PRINT=CRITICAL} \end{array} \right) \left( \begin{array}{l} \text{STORAGE} = \left( \begin{array}{l} n \\ nk \\ (n, \text{VIRT}) \\ (nk, \text{VIRT}) \end{array} \right) \end{array} \right)$$

(, LABEL=(output, input<sub>1</sub>, ---, input<sub>n</sub>, work)) (, VIRIFY) (, KEYLEN=n)



$$\left[ \begin{array}{l} ,ADDR\emptyset UT \\ ,ADDR\emptyset UT=A \\ ,ADDR\emptyset UT=D \end{array} \right] \left[ ,CALCAREA \right] \left[ ,ALTWK \right] \left[ ,ERASE \right] \left[ ,N\emptyset TPMK \right]$$

$$\left[ \begin{array}{l} ,R\emptyset UTE=LST \\ ,R\emptyset UTE=L\emptyset G \end{array} \right] \left[ \begin{array}{l} ,DUMP \\ ,N\emptyset DUMP \end{array} \right] \left[ ,TP \right] \left[ ,DIAG \right] \left[ \begin{array}{l} ,FILNM=output \\ ,FILNM=( output ,input_1,---,input_n ) \end{array} \right]$$

$$\left[ \begin{array}{l} ,SORTIN=input \\ ,SORTOUT=output \end{array} \right] \left[ \begin{array}{l} ,SORTIN=(input_1,---,input_n) \end{array} \right]$$

$$\left[ \begin{array}{l} ,SORTWK=work \\ ,SORTWK=(work_1,---,work_n) \end{array} \right]$$

- PRINT

PRINT=NONE บอกให้โปรแกรมไม่ต้องพิมพ์ข้อความใด ๆ ออกมา

PRINT,PRINT=ALL บอกให้โปรแกรมพิมพ์ข้อความออกมาทั้งหมดรวมทั้งข้อผิดพลาด

ด้วย

PRINT=CRITICAL จะพิมพ์เฉพาะข้อความที่สำคัญเช่นข้อผิดพลาด

- STORAGE บอกถึงขนาดความจำหลักที่ใช้และจะให้ fix page หรือไม่

n เป็นเลขฐาน 10 หน่วยเป็นไบต์ กำหนดส่วนความจำหลัก

VIRT บอกให้รู้ว่าโปรแกรมวิ่งใน VIRTUAL MODE ไม่ fix page การทำงาน

จะข้าลง

- LABEL กำหนดประเภท label ของ output, input และ work มี 3 ประเภท

N = non standard labels

S = standard labels ถ้าไม่กำหนดจะให้ค่าเป็น S

U = unlabels

- VERIFY ใช้ในกรณีที่แฟ้มข้อมูลนำออกถูกเก็บใน direct access device และต้องการให้แต่ละบล็อกถูกตรวจสอบเพื่อให้แน่ใจว่าการบันทึกถูกต้อง คำสั่งนี้จะไม่ใช่สำหรับแฟ้มข้อมูลนำออกที่เป็น
- KEYLEN ใช้กับระเบียบข้อมูลนำเข้าที่เก็บใน direct access device ที่ถูกนำด้วยเขตหลัก (key field) ยาว n ไบท์ ความยาวของเขตหลักจะถูกรวมอยู่ในความยาวของระเบียบข้อมูล, ขนาดบล็อก, เขตควบคุม คำสั่งนี้ห้ามใช้กับแฟ้มข้อมูล
- ADDR $\emptyset$ UT = n  
ถ้า n เป็น A จะใช้ผลลัพธ์จากการจัดเรียงข้อมูลจะเป็นเพียงของระเบียบข้อมูลนำเข้าเท่านั้น ถ้า n = D ก็จะมีลักษณะเดียวกันแต่จะตามด้วยเขตควบคุมของแต่ละระเบียบข้อมูล
- CALCAREA โปรแกรมจะใช้ทำงานเฉพาะเฟลล์ 0 และเมื่อแฟ้มข้อมูลชั่วคราวเป็นจานแม่เหล็กจะทำการคำนวณหาขนาดที่เหมาะสมของแฟ้มข้อมูลชั่วคราว
- ALTWK กำหนดให้มีการสลับตัวแม่เหล็กในงานการจัดเรียงข้อมูล ซึ่งจะช่วยให้ค่าสูงสุดของขนาดแฟ้มข้อมูลนำเข้าขยายขึ้นเป็น 2 เท่า ที่อยู่ของสื่อที่จะใช้สลับกัน จะต้องต่างกับที่อยู่ของสื่ออื่น ๆ ที่จะใช้ในงานนี้
- ERASE กำหนดให้แฟ้มข้อมูลชั่วคราวถูกลบทิ้งหลังจากการจัดเรียงข้อมูลเสร็จแล้ว แต่จะไม่ใช่กับแม่เหล็ก
- N $\emptyset$ TPMK กำหนดว่าจะไม่มีการ write tapemark ก่อนที่ระเบียบข้อมูลอันแรกของแฟ้มข้อมูลนำออกในทุก ๆ หน่วย พารามิเตอร์นี้จะใช้ได้กับแฟ้มข้อมูลนำออกที่ไม่มี label เท่านั้น
- R $\emptyset$ UTE = LST โปรแกรมสำเร็จรูปจะส่งข้อความไปให้แฟ้มข้อมูล SYSLST และข้อความเหล่านี้ก็จะออกทางจอภาพด้วย แต่ถ้ากำหนดเท่ากับ LOG ข้อความเหล่านี้จะถูกพิมพ์ทาง SYSLST
- DUMP/N $\emptyset$ DUMP ถ้าเกิดการผิดพลาดขึ้นสามารถกำหนดให้ดึงข้อมูลในส่วนความจำหลักออกทาง SYSLST ได้ หรือไม่ต้องการก็ได้
- TP เมื่อโปรแกรมทำงานคู่ขนานกับงานที่ทำผ่านทางสายสื่อสาร เป็นไปได้ที่เวลาที่ใช้ในการขานรับจะสูงขึ้น ดังนั้นพารามิเตอร์ TP จะช่วยในการกำหนดค่าระดับของการจัดรวมข้อมูล

- DIAG กำหนดให้พิมพ์ข้อความตรวจสอบออกมาด้วย

- FILNM กำหนดชื่อแฟ้มข้อมูลเป็นแบบไม่มาตรฐาน ซึ่งใช้ใน TLBL และ DLBL

คำสั่งควบคุมการทำงานสำหรับแฟ้มข้อมูลนำเข้าและแฟ้มข้อมูลนำออก พารามิเตอร์ที่อยู่ในวงเล็บต้องถูกแทนที่ด้วยชื่อแฟ้มข้อมูลหรือเครื่องหมาย (,)

- SØRTØUT กำหนด logical unit number ของแฟ้มข้อมูลนำออก พารามิเตอร์ต้องถูกแทนที่ด้วย symbolic unit number ซึ่งเป็นเลขสูงสุด 3 หลัก (1-221)

- SØRTIN กำหนดหน้าที่เช่นเดียวกับ SØRTØUT แต่เป็นของแฟ้มข้อมูลนำเข้า

- SØRIWK กำหนดหน้าที่เช่นเดียวกับ SØRTIN แต่เป็นของแฟ้มข้อมูลชั่วคราว

11. คำสั่ง END จะปรากฏเป็นตัวสุดท้ายของคำสั่งควบคุม ซึ่งจะสิ้นสุดของบัตรควบคุมสำหรับโปรแกรม

หมายเหตุ การจัดเรียงลำดับคำสั่งควบคุมจะเรียงลำดับก่อนหลังอย่างไรก็ได้ ยกเว้นคำสั่ง END จะต้องเป็นคำสั่งสุดท้าย และคำสั่ง ØPTION ควรจะเป็นคำสั่งแรก เพราะเหตุว่าโปรแกรมจะใช้คำสั่งที่กำหนดให้ก่อนจนกว่าจะอ่านพบคำสั่ง ØPTION

ภาคผนวก ค

จุดออกของโปรแกรมสำเร็จรูปสำหรับการจัดรวมข้อมูลอย่างมีลำดับ

รูปต่อไปนี้จะแสดงให้เห็นถึงการใช้จ่ายจุดออกของโปรแกรมสำเร็จรูปช่วยในการทำงาน

การใช้จ่ายจุดออก	เฟส 3					
	E31	E32	E35	E37	E38	E39
ตรวจสอบจุดต่าง ๆ	X	X <sup>2</sup>				
จัดเลเบล	X	X <sup>2</sup>		X		
เปิดแฟ้มข้อมูล	X	X <sup>2</sup>				
ปิดแฟ้มข้อมูล	X			X	X	X
ใส่ค่าของผ่านของ VSAM		X <sup>2</sup>				
นับจำนวนระเบียน		X <sup>2</sup>	X			
เพิ่มหรือลบระเบียน		X <sup>2</sup>	X			
ตัดหรือเพิ่มข้อมูลในระเบียน		X	X			
เปลี่ยนแปลงข้อมูลในระเบียน <sup>1</sup>		X <sup>2</sup>				
อ่านข้อมูลนำเข้า เพื่อรวมแฟ้มข้อมูล		X <sup>2</sup>				
รวมระเบียน		X <sup>2</sup>	X			
แทนระเบียน (รวมแฟ้มข้อมูล)		X				
ไม่ตรวจสอบลำดับ			X			
บันทึกผลลัพธ์			X			

การใช้จุดออก	เฟส 3					E39
	E31	E32	E35	E37	E38	
ประมวลผลเพื่อการอ่านผิดพลาด การประมวลผลของอุปกรณ์เข้าถึง ข้อมูลโดยตรงเมื่อการบันทึกผิดพลาด		X <sup>2</sup>			X	X
1. ไม่รวมการตัดหรือเพิ่มเขตข้อมูลที่ใช้ในการเรียงลำดับ 2. เมื่อมีการกำหนด INPFIL EXIT เท่านั้น						

### จุดออกของเฟส 3

- E31 ใช้ในการเปิดแฟ้มข้อมูลที่มีการคัดเลเบลแบบไม่มาตรฐานหรือเลเบลของผู้ใช้เอง และเริ่มโปรแกรมของผู้ใช้งาน คำพารามิเตอร์ที่โปรแกรมคัดให้แก่โปรแกรมของผู้ใช้คือ ตำแหน่งที่อยู่ของ

1. รหัสกำหนดการทำงานเพื่อการตรวจสอบซึ่งในเฟส นี้ไม่มีใช้
2. รายการอุปกรณ์ที่จะตรวจสอบ
3. หน่วยของอุปกรณ์ (logical) ที่จะทำ trailer label
4. หน่วยของอุปกรณ์ (logical) ที่จะทำ header label
5. ตั้งนับค่าบล็อกของ volume ที่เพิ่งจะประมวลผลเสร็จ

- E32 ใช้สำหรับการคัดรวมข้อมูลอย่างมีลำดับเท่านั้น ในการประมวลผลระเบียนข้อมูลนำเข้า ถ้าไม่กำหนด INPFIL EXIT แล้ว คำพารามิเตอร์ที่โปรแกรมสำเร็จรูปจะส่งให้กับโปรแกรมของผู้ใช้งานก็คือ ค่าตำแหน่งของระเบียนต่อไป ซึ่งจะสามารถเปลี่ยนแปลงข้อมูลในระเบียนนี้ได้ แต่ไม่สามารถเปลี่ยนความยาวของระเบียนได้ ถ้ามีการกำหนด INPFIL EXIT ผู้ใช้งานจะต้องรับผิดชอบทั้งหมดเกี่ยวกับแฟ้มข้อมูลนำเข้าคือ ตั้งแต่การเปิดแฟ้มข้อมูลการคัดเลเบล การคัดการเกี่ยวกับการดึงระเบียนออกมาเพื่อส่งให้กับโปรแกรมสำเร็จรูป พารามิเตอร์ที่ใช้ติดต่อกันระหว่าง

โปรแกรมของผู้ใช้งานและโปรแกรมสำเร็จรูปมี 3 ตัว คือ

1. ตำแหน่งของระเบียบที่โปรแกรมของผู้ใช้งานส่งให้กับโปรแกรมสำเร็จรูป
2. รหัสซึ่งโปรแกรมสำเร็จรูปสร้างให้เพื่อบ่งว่าระเบียบต่อไปจะมาจากแฟ้มข้อมูลใด
3. รหัสกำหนดการทำงานซึ่งผู้ใช้งานเป็นผู้กำหนด

- E35 ใช้ในการเพิ่ม ลบ หรือเปลี่ยนแปลงระเบียบซึ่งจะถูกบันทึกเป็นผลลัพธ์ ซึ่งอาจจะไม่มีการตรวจสอบลำดับ ณ จุดออกนี้ ถ้ามีการใช้ EXIT ในคำสั่งควบคุม ØUTFIL แล้ว แฟ้มข้อมูลผลลัพธ์ทั้งหมดอาจถูกลบทิ้งและนำไปเขียนไว้ในอุปกรณ์ที่ผู้ใช้งานเลือกพารามิเตอร์ ซึ่งโปรแกรมสำเร็จรูปส่งให้กับโปรแกรมของผู้ใช้งาน คือ

1. ตำแหน่งของระเบียบที่เป็น winner
2. ตำแหน่งของ winner ตัวที่แล้ว
3. ผู้ใช้งานต้องกำหนดรหัสที่จะต้องทำงาน
4. ผู้ใช้งานต้องกำหนดรหัสที่ใช้ในการตรวจสอบลำดับ

- E37 ใช้ในการปิดแฟ้มข้อมูล บันทึกหรือตรวจสอบ trailer label ที่มีเลเบลเป็นแบบไม่มาตรฐาน คำพารามิเตอร์ที่โปรแกรมสำเร็จรูปส่งให้กับโปรแกรมของผู้ใช้งานคือตำแหน่งของบริเวณตัวนับค่าบล็อก (สำหรับการคัดเรียงลำดับ)

- E38 ใช้ในการประมวลผลสำหรับความผิดพลาดในการทำงานเกี่ยวกับ I/O ของแฟ้มข้อมูลที่จัดแบบ VSAM และใช้สำหรับการรวมแฟ้มข้อมูลเท่านั้น พารามิเตอร์ที่โปรแกรมสำเร็จรูปส่งให้กับโปรแกรมของผู้ใช้งาน คือ

1. ตำแหน่งหน่วยของ CCB ที่เกิดความผิดพลาด
2. ตำแหน่งของโปรแกรมที่ทำงานเกี่ยวกับ channel
3. โปรแกรมของผู้ใช้งานจะต้องกำหนดรหัสเพื่อระบุการทำงาน

ผู้ใช้งานสามารถตรวจสอบความผิดพลาดได้ว่าเกิดจากความยาวของระเบียบไม่ถูกต้องหรือเกิดจากการอ่านข้อมูลโดยการตรวจสอบสถานะของระเบียบในส่วนของ CCB ซึ่งจะมีการตรวจสอบสถานะถ้าความยาวของระเบียบไม่ถูกต้อง

สำหรับแฟ้มข้อมูลที่จัดแบบ VSAM ผู้ใช้งานจะสามารถจัดการเกี่ยวกับความผิดพลาดทางด้าน I/O หรือการใส่คำขอผ่านหรือรายการตรวจสอบ VSAM พารามิเตอร์ที่โปรแกรมสำเร็จรูปจะส่งให้กับโปรแกรมของผู้ใช้งาน คือ flag ซึ่งระบุว่าต้องการคำขอผ่านหรือส่วนโปรแกรมที่จัดการเกี่ยวกับความผิดพลาดของ I/O หรือรายการการออกจากโปรแกรมพร้อมทั้งตำแหน่งที่อยู่ของ VSAM RPL (ทำงานเมื่อเกิดความผิดพลาดเกี่ยวกับ I/O) ผู้ใช้งานจะต้องระบุรหัสเพื่อกำหนดการปฏิบัติงานให้แก่โปรแกรมสำเร็จรูป

- E39 จะใช้เมื่อเกิดความผิดพลาดเกี่ยวกับการเขียน ผู้ใช้งานอาจจะนำข้อมูลของบล็อกออกมา เมื่อโปรแกรมของผู้ใช้งานสั่งการควบคุมกลับไปยังโปรแกรมสำเร็จรูปการทำงานจะเป็นอันสิ้นสุดลง ค่าพารามิเตอร์ที่โปรแกรมสำเร็จรูปส่งให้แก่โปรแกรมของผู้ใช้งาน คือ ตำแหน่งที่อยู่ของหน่วย CCB ที่เกิดความผิดพลาดและตำแหน่งโปรแกรมของ channel

## โมดูลที่กำหนดหน้าที่เกี่ยวกับจุดออกของโปรแกรม

จุดออก	โมดูล
E31	ASL, RGH
E32	RBG
E35    การเรียงลำดับในเทปแม่เหล็ก การเรียงลำดับในจานแม่เหล็ก (ความยาวคงที่) การเรียงลำดับในจานแม่เหล็ก (ความยาวไม่คงที่) การคัดรวมข้อมูลอย่างมีลำดับ	RBG RBJ RBI RBG
E37	RSN
E38    SAM VSAM    คำขอผ่าน, ความผิดพลาดของ I/Ø VSAM    ความผิดพลาดของ I/Ø	RGH AGJ RGJ
E39    SAM (เทปแม่เหล็ก) SAM (จานแม่เหล็ก) VSAM (คำขอผ่าน) VSAM (ความผิดพลาดของ I/Ø)	RPE RPF APG RPG





## ภาคผนวก ง

### วิธีที่ใช้ในการคัดรวมข้อมูลอย่างมีลำดับของ โปรแกรมสำเร็จรูป

โปรแกรมสำเร็จรูปจะใช้วิธีการเปรียบเทียบแบบไบนารี ในการคัดรวมข้อมูลอย่างมีลำดับ

2 ลักษณะด้วยกัน

1. ใช้วิธีไบนารี ในการคัดรวมข้อมูลในเฟลล์ 2 ของการคัดเรียงลำดับข้อมูล
2. ใช้วิธีไบนารี ในการคัดรวมข้อมูลในเฟลล์ 3 ทั้งของการคัดเรียงลำดับข้อมูลและการคัดรวมข้อมูลอย่างมีลำดับ

#### ขั้นตอนการทำงานวิธีไบนารี

ขั้นแรก จะทำการเพิ่มระเบียบข้อมูลตัวใหม่ลงในกลุ่มข้อมูลที่เรียงตามลำดับ โดยทำการเปรียบเทียบระเบียบข้อมูลตัวใหม่กับระเบียบข้อมูลตัวกึ่งกลางของกลุ่มข้อมูลที่เรียงตามลำดับ ถ้าระเบียบข้อมูลตัวใหม่มีค่ามากกว่าแล้ว ระเบียบข้อมูลที่มีค่าน้อยกว่าจะถูกแยกออก หรือถ้าระเบียบข้อมูลใหม่มีค่าน้อยกว่า ระเบียบข้อมูลที่มีค่ามากกว่าจะถูกแยกออก

ขั้นที่สอง จะทำการเปรียบเทียบระเบียบข้อมูลใหม่กับระเบียบข้อมูลตัวตรงกลางของกลุ่มข้อมูลที่เหลือ ทำเช่นนี้ไปเรื่อย ๆ จนกระทั่งเหลือเพียงระเบียบข้อมูลตัวเดียว

ขั้นที่สาม การเพิ่มระเบียบข้อมูลใหม่จะเพิ่มด้านซ้ายหรือด้านขวาของระเบียบข้อมูลที่เหลือขึ้นอยู่กับว่า ถ้าระเบียบข้อมูลใหม่มีค่ามากกว่าก็จะเพิ่มทางด้านขวา หรือถ้ามีค่าน้อยกว่าจะเพิ่มทางด้านซ้าย และทำการเคลื่อนระเบียบข้อมูลตัวต่ำสุดออกไปเก็บยังที่เก็บผลสัทธิชั่วคราว

#### การคัดรวมข้อมูลในเฟลล์ 2

เริ่มแรกจะดึงข้อมูลหนึ่งบล็อกของระเบียบข้อมูลแต่ละกลุ่มที่จะถูกคัดรวมเข้าไปไว้ในส่วนความจำหลัก หาความสัมพันธ์ของแต่ละระเบียบข้อมูล และเก็บที่อยู่ของแต่ละระเบียบไว้ในตาราง โดยจัดเรียงลำดับตามเขตควบคุม หาระเบียบข้อมูลที่มีค่าต่ำสุดตัวแรกและทำการบันทึกออกไปก่อน การเพิ่มระเบียบข้อมูลใหม่เข้าไปในบล็อกนั้นจะใช้วิธีไบนารี

การทำงานจะทำเรื่อย ๆ จนกว่าจะพบจุดจบของกลุ่ม ค่าสำหรับการจัดรวมข้อมูลจะถูกลดลงหนึ่ง และทำการจัดรวมข้อมูลต่อไป เมื่อทำการจัดรวมข้อมูลเรียบร้อยแล้วจะทำการบันทึกข้อมูล จากนั้นการประมวลผลจะกลับมา เริ่มต้นใหม่ตั้งแต่จุดแรกเข้าไปไว้ในส่วนความจำหลัก กำหนดค่าเริ่มต้นใหม่ของค่าสำหรับการจัดรวมข้อมูลและทำการจัดรวมข้อมูลต่อไป จนกว่าการทำงานจะ เสร็จสิ้นทั้งแฟ้มข้อมูล

### การจัดรวมข้อมูลในเฟส 3

การจัดรวมข้อมูลในเฟส นี้ก็คล้ายกับเฟส 2 แต่กลุ่มข้อมูลนำเข้าแต่ละกลุ่มจะประกอบด้วยเพียงกลุ่มเดียวในข้อมูลนำเข้า 1 ตัว

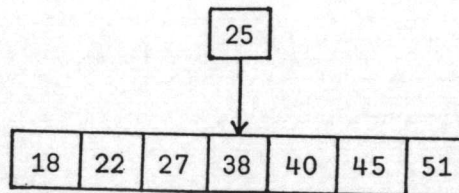
ระเบียนข้อมูลใหม่

25
----

กลุ่มข้อมูลเรียงลำดับ

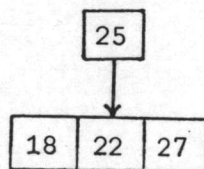
18	22	27	38	40	45	51
----	----	----	----	----	----	----

1 เปรียบเทียบ 25 กับ 38



กลุ่มข้อมูลที่อยู่ในข่ายการพิจารณา

2 เปรียบเทียบ 25 กับ 22



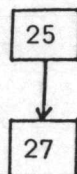
กลุ่มข้อมูลที่อยู่ในข่ายการพิจารณา

38	40	45	51
----	----	----	----

กลุ่มข้อมูลที่ไม่อยู่ในข่ายการพิจารณา

3 เปรียบเทียบ 25 กับ 27

18	22
----	----



38	40	45	51
----	----	----	----

กลุ่มข้อมูลที่ไม่อยู่ในข่ายการพิจารณา    กลุ่มข้อมูลที่อยู่ในข่ายการพิจารณา    กลุ่มข้อมูลที่ไม่อยู่ในข่ายการพิจารณา

4 เพิ่มค่า 25 เข้าไปในกลุ่มข้อมูล และส่งค่า 18 ออกจากกลุ่มข้อมูล

กลุ่มข้อมูลเรียงลำดับใหม่

22	25	27	38	40	45	51
----	----	----	----	----	----	----

ที่เก็บระเบียนข้อมูลผลลัพธ์ชั่วคราว

18
----

ที่เก็บระเบียนข้อมูลนำเข้าตัวใหม่

X
---

แสดงตัวอย่างขั้นตอนการทำงานของวิธีไบนารี

ภาคผนวก จ

โปรแกรม PDAID

โปรแกรม PDAID เป็นเครื่องมือที่ใช้ในการบอกข่าวสารต่าง ๆ ที่ได้รับจากระบบ ขณะที่โปรแกรมที่มีปัญหา กำลังทำงานอยู่ การเริ่มต้นใช้โปรแกรม PDAID จะใช้เมื่อทำการ rerun โปรแกรมที่มีปัญหาหลังจากการวิเคราะห์ปัญหาครั้งแรกแล้ว ชนิดของการคัดลอก (trace) ที่จะใช้กับโปรแกรมที่มีปัญหาขึ้นอยู่กับผลของการวิเคราะห์ครั้งแรกและจำนวนข่าวสารที่ต้องการใช้ช่วยในการแยกข้อผิดพลาด

ชนิดของการคัดลอกโดยโปรแกรม PDAID มี 7 ชนิด

- บันทึกการทำงานของ I/Ø (I/Ø trace)
- บันทึกลำดับของแต่ละเฟลล์และที่เรียกชั่วคราว (Fetch/Load trace)
- บันทึกลำดับซึ่ง supervisor call (SVC) ถูก execute (Generalized SVC trace)
- บันทึกลำดับซึ่งตั้งแต่ SVC0 ถึง SVC31 และการเกิด I/Ø interrupts (QTAM trace)
- บันทึกลำดับซึ่งตั้งแต่ SVC49 ถึง SVC53 และการเกิด I/Ø interrupts (VTAM trace)
- บันทึกจำนวนสูงสุดของ buffer ที่ใช้ pool หรือ queue (VTAM Buffer Pool trace)
- Dump PTA และ LTA เมื่อเกิดโปรแกรม check (Transient Dump)

การคัดลอกสามารถที่จะเลือกให้บันทึกเหตุการณ์ลงบนเทปแม่เหล็ก, พิมพ์บนกระดาษต่อเนื่อง หรือเก็บไว้ใน PD area

การบันทึกเหตุการณ์ลงบนเทปแม่เหล็กทำได้ 2 วิธีคือ

1. บันทึกเหตุการณ์ลงบนเทปแม่เหล็กในรูปแบบของ core image
2. โดยใช้ PDLIST utility จะต้อง assign SYS005 ให้กับเทปแม่เหล็กและ

SYSLST ให้กลับเครื่องพิมพ์

ขณะที่ทำการบันทึกข้อมูลถ้าข้อมูลที่ใช้ยังไม่ส่งภาพที่ยังไม่พร้อมจะเกิดข้อความ

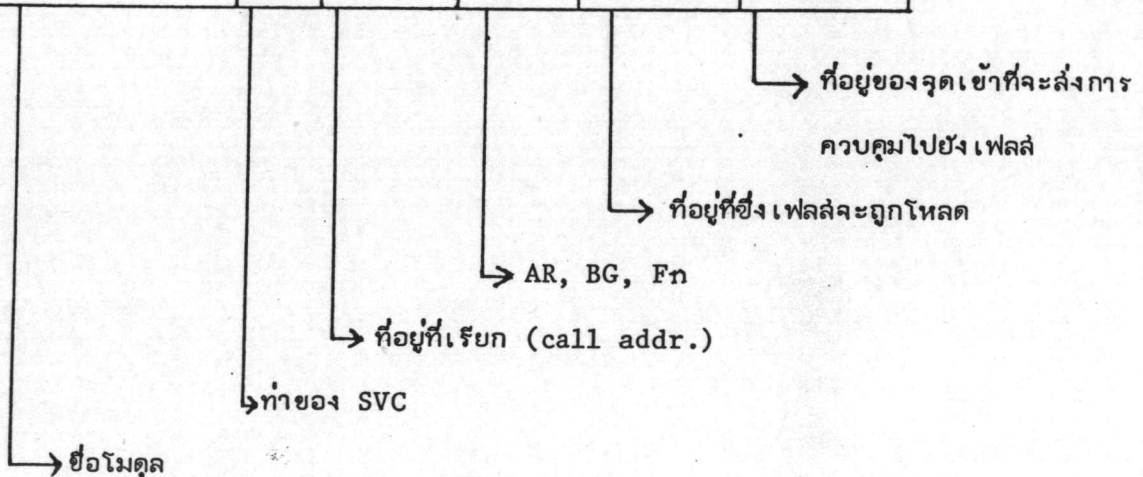
4C24A NØ I/Ø TØ ØD

ซึ่งจะทำให้ระบบเกิดว่าง การแก้ไขให้ทำการคัดลอกข้อมูลให้อยู่ในสล็อตที่พร้อมจะใช้งานแล้วกด END/ENTER การทำงานจะทำได้ไป

การวิเคราะห์ผลของโปรแกรมทดสอบต้องการทราบว่าโมดูลใดใช้งานบ้าง จึงได้ใช้โปรแกรม PDAID ช่วย วิธีที่จะใช้ในการคัดลอกโมดูลจะใช้ Fetch/Load trace เท่านั้น ผลการวิเคราะห์ที่ได้จะมีรายละเอียดลักษณะดังนี้

8 ไบท์      1 ไบท์    3 ไบท์    2 ไบท์      4 ไบท์      4 ไบท์

MØDULE NAME	SVC NO.	SVC ADDR.	SYS LOG ID	LØAD ADDR.	ENTRY ADDR.
-------------	---------	-----------	------------	------------	-------------



ตัวอย่างโมดูล SØRTRCLX

SØRTRCLX 4 C-20DOAO F3 L-20D1D0 E-20D1E0

ตัวอย่างการเรียกใช้โปรแกรม PDAID ทางจอภาพ

เริ่มต้นจะต้องทำการโหลดเฟลลเริ่มต้นของโปรแกรม PDAID ด้วยคำสั่ง

// EXEC PDAID

หลังจากนั้นจะมีข้อความ

4C10D PDAID =

ให้ operator ทำการตอบข้อความอันหนึ่งอันใดดังต่อไปนี้

- IT สำหรับ I/Ø trace
- FT สำหรับ F/L trace
- GT สำหรับ GSVC trace
- QT สำหรับ QTAM trace
- VT สำหรับ VTAM trace
- VP สำหรับ VTAM Buffer Pool Trace
- TD สำหรับ transient Dump
- XX ใช้สำหรับตัดการติดต่อกับโปรแกรม PDAID

ต่อไปจะต้องเลือก output device จะมีข้อความ

4C11D ØUTPUT DEVICE

ตัวอย่างการใช้โปรแกรม PDAID ทำ Fetch/Load trace บน partition F3 ทางจอ

ภาพ

```

F3 // JOB SYSTEST
F3 // PAUSE SYSTEST
F3 // EXEC PDAID
F3 4C10D PDAID =
F3 FT
F3 4C11D OUTPUT DEVICE =
F3 00E
F3 4C12D TRACE PARTITION =
F3 F3
F3 4C12D TRACE PARTITION =
F3 GØ
F3 READLY FOR COMMUNICATIONS.
F3
F3 EØJ SYSTEST

```



ข้อควรระวังในการใช้โปรแกรม PDAID

1. จะต้อง run ในขณะที่ partition ที่ใช้ทดสอบว่าง คือ จะมีงานที่ทดสอบเพียงงานเดียวใน partition นั้น
2. กรณีที่กำหนด output device เป็น printer จะต้องทำการ stop printer เสียก่อน เพราะเหตุว่าขณะที่โปรแกรม PDAID ทำงานอยู่ข่าวสารที่ trace ออกมาจะถูกพิมพ์ทันที ทุกครั้งที่มีการ Fetch/Load
3. ทุกครั้งหลังจากทำการทดสอบเสร็จแล้ว จะต้องทำการ terminate โปรแกรม PDAID เสียก่อน เพราะเหตุว่าถ้าไม่ terminate โปรแกรม PDAID ทั้ง เมื่อมีงานอื่นเข้ามาทำงานใน partition นั้น มันก็จะทำการ trace โปรแกรมนั้นต่อไปด้วย

## ภาคผนวก ฉ

### การ catalogue โปรแกรมย่อย

การเรียกใช้โปรแกรมสำเร็จรูปโดยผ่านโปรแกรมภาษา FORTRAN IV โปรแกรมภาษา FORTRAN IV จะเป็นโปรแกรมหลักในการเรียกใช้โปรแกรมย่อยภาษา ASSEMBLY ให้ทำการโหลดโปรแกรมสำเร็จรูปขึ้นมาใช้งาน

#### โปรแกรมย่อยภาษา ASSEMBLY ประกอบด้วย

1. โปรแกรมย่อย MCARD หน้าที่อ่านข้อมูลจากบัตรเจาะรูบนเทปแม่เหล็ก
2. โปรแกรมย่อย MTAPE หน้าที่อ่านบัตรคำสั่งควบคุมและเก็บคำสั่งควบคุมไว้ในส่วนความจำหลักและสิ่งที่อยู่ของคำสั่งควบคุมไปยังโปรแกรมสำเร็จรูป
3. โปรแกรมย่อย MØDMERGE หน้าที่โหลดโปรแกรมสำเร็จรูปขึ้นมาทำงาน และส่งการควบคุมไปให้ยังโปรแกรมสำเร็จรูป

การคัดรวมข้อมูลอย่างมีลำดับด้วยโปรแกรมภาษา FORTRAN จะทำได้ต่อเมื่อโปรแกรมภาษา FORTRAN จะต้อง link กับโปรแกรมย่อยภาษา ASSEMBLY โปรแกรมภาษา FORTRAN จะมีการเปลี่ยนแปลงไปตามการใช้งาน แต่โปรแกรมย่อยจะไม่มี การเปลี่ยนแปลง ดังนั้นจึงได้มีการ catalogue โปรแกรมย่อยเอาไว้ใน Relocatable library เวลาใช้งานจริงไม่ต้อง compile โปรแกรมย่อยใหม่ เพียงแต่เรียกชื่อโมดูลขึ้นมาและ link โปรแกรมเข้าหากันเท่านั้น โปรแกรมก็สามารถที่จะ execute ได้ทันที





LOC OBJECT CODE ADDR1 ADDR2 STMT SOURCE STATEMENT DOS/V5 ASSEMBLER REL 34.0 17.01 82-04-10

PAGE 3

```

MOVA4E=MOTAPE,
TPYARK=YES,
TYPEFL=JUTPIT,
MOKA=YES
M140J RESFOR4=FIXUNB,
TYPEFL=OUTPIT,
MOKA=YES
399 MOTAPE

```

```

DS BUFIC 667
DS QUIT 668
DS CL132 669
DS CL30 670
DS SAVEA 671
DS CL132 672
DS CL132 673
DS CL30 674

```

LDR3

```

675 =C$$$3PEN
676 =C$$$3L3SE
677 =\PRINTO
678 =\PRINTA
679 =\C44)IN)
680 =A(T43E4)
681 =\T4PE)
682 =\T4PE3JF)
683 =\M14PE)
684 =C'DATA CARD
685 =C'RE3) DATA CARD-COMPLERE

```

```

0001E 000236
00029A 0000358 4C
000358 4040404040404040
000359 4040404040404040
0003D0 4040404040404040
000430 5858C2D6D7C5D540
000438 5F5RC2C3D3D6E2C5
000440 0C000160
000444 00000359
000448 00000C0D8
00044C 0000028A
000450 00000190
000454 000003D0
000458 00000000
00045C C4C1E3C140C3C1D9
000455 D9C5C1C440C4C1E3

```

LOC	OBJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT
000000					CSECT
000000	0580				LI,0
000002	003F				USING *LI
000003	90ED	03F6			STK 14,13,SAVED
000004	5850	84D6			OPEN CARD,PRINT
000005	5880	84D4			L 5,=(PARMETS)
000006	0048				L 5,=(JCS)
000007	0048				L 7,3
000008	4160	8344			703 READJCS
000009	4160	8344			709 SET
000010	0034				6,CARDA
000011	0508	8344	8522	0034	CLC CARDA(2),=C'1' ENDJCS'
000012	4780	8184			ENDJCS
000013	957C	8344	00316		CLC CARDA,2,a'
000014	4780	8094	0009C		BE SEARCH
000015	0216	8453	8528	00455	714 ERRJCS
000016	0283	843E	843E	00441	720 MVC PRINTA+20(23),=CL23'CONTROL STATEMENT ERROR'
000017	0283	843E	843E	00441	721 PRT
000018	024F	8453	8314	00455	727 MVC PRINTB,PRINTA-1
000019	024F	8453	8314	00455	728 MVC PRINTB,PRINTA,5,1
000020	47F0	8222	80224		741 H DEFRT
000021	0504	8317	8542	00349	742 CLC CARDA+1(3),=C'VERGE'
000022	4770	807C	0004E		743 BNE HREC
000023	4190	80C0	0000C		744 LA 9,1(0,1)
000024	47F0	8144	00148		745 B STCARDA
000025	0505	8317	844F	00349	746 CLC CARDA+1(6),=C'RECORD'
000026	4190	80B8	000C0		747 BNE MIPFIL
000027	4190	8004	00004		748 LA 9,4(0,1)
000028	47F0	8160	00142		749 B STCARJ
000029	0505	8317	8500	00349	750 CLC CARDA+1(5),=C'INPIL'
000030	4190	80D0	00002		751 BNE MOUTFIL
000031	4190	8038	00308		752 LA 9,8(0,1)
000032	47F0	8160	00162		753 B STCARJ
000033	0505	8317	8506	00349	754 CLC CARDA+3(6),=C'OUTFIL'
000034	4770	80E2	000E4		755 BNE MPT
000035	4190	800C	0000C		756 LA 9,12(1,3)
000036	4770	8160	00162		757 B STCARJ
000037	0505	8317	850C	00349	758 CLC CARDA+3(5),=C'OPTION'
000038	4770	80F4	000F6		759 BNE KMDS
000039	4190	8010	0001C		760 LA 9,16(1,3)
000040	47F0	8160	00162		761 B STCARJ
000041	0503	8347	84EE	00349	762 CLC CARDA+1(4),=C'MDOS'
000042	4770	8016	00108		763 BNE HALLS
000043	4190	8014	00014		764 LA 9,20(1,3)
000044	47F0	8160	00162		765 B STCARJ
000045	0504	8317	8512	00349	766 CLC CARDA+1(1),=C'ALTSE0'
000046	4190	8113	0011A		767 BNE MPTS
000047	4190	8028	00028		768 LA 9,3(1,3)
000048	47F0	8140	00142		769 B STCARJ
000049	0504	8317	8518	00349	770 CLC CARDA+1(1),=C'OTREC'



DJS/V5 ASSEMBLER REL 34.0 17.01 82-04-10

OBJECT CODE	ADDR1 ADDR2	STMT	SOURCE STATEMENT	LCR3
0004C8	5858C2D0703D540	1144	=C'5858C2D0703D540'	1143
0004D0	5858C2C33D06E205	1145	=C'5858C2C33D06E205'	1146
0004D8	00000064	1146	=A(JCS)	1147
0004E0	00000240	1148	=A(CA1)	1149
0004E4	00000346	1149	=A(CA2)	1150
0004E8	000002C4	1150	=A(PRI1)	1151
0004EC	C0000441	1151	=A(PRI3)	1152
0004F0	D46C4F2	1152	=C'MOVS	1153
0004F4	D64C9E3	1153	=C'YHL'	1154
0004F8	00000000	1154	=A(ADJADJES)	1155
0004FC	D95C3D6D9C4	1155	=C'RECDK)	1156
000502	C95D7C669D3	1156	=C'INPFL'	1157
000504	D6E4E3C459D3	1157	=C'OUTFL'	1158
00050E	D6D7E3C946D5	1158	=C'PTIND)	1159
000514	C1D3E3E2C5D8	1159	=C'ALTSB)	1160
000518	D4E4F3D5C5C3	1160	=C'OUTREC'	1161
000520	7C7C	1161	=C'99'	1162
000522	0000	1162	=C'99'	1163
000524	7C4C0C339C4D173	1163	=C'99 ENJCS'	1164
000528	C30D5E339D6D340	1164	=CLZ3333TRCL STATEMENT ERROR'	1165
000544	D4C5D5C7C5	1165	=C'4E1SE'	1166
000548	E2E4D4	1166	=C'5D4'	1167
00054C	C9D5C3D54C4C5	1167	=C'10LJDE'	1168
000554	D4C5D5C7C54C1	1168	=C'ME1SE DATA COMPLETE'	1169
000564	D4C5D5C7C54C041	1169	=C'ME1SE DATA NOT COMPLETE'	

LCR3

LN#	OBJECT CODE	WORK ADRSZ	STMT	SOURCE STATEMENT
000000			1171	MCMBERGE CSECT
000000			1172	3ALR 9.0
000007	53E0 5025	0000Z	1173	USING *
000007		0002R	1174	ST 14,SAVEC
000007		0000R	1175	LOAD SORT,LOADLCC
000007		0000C	1181	LR 15,1
000007		0000C	1182	LA 1,PAR11
000007		0000R	1183	LA 14,SAVAKCA
000007		0000C	1184	3ALR 14,15
000007		0000C	1185	L 7,=A(4ETJRN)
000007		0000R	1186	L 14,SAVEC
000024	07FE	0002R	1187	BR 14
000028			1188	S:VEEC
000028			1189	PARAMET
000028			1190	PARAM
000030	00000000		1191	DC A(1)
000030	00000000		1192	DC A(1)
000030	00000000		1193	DC A(1)
000030	00000000		1194	DC A(1)
000030	00000000		1195	DC A(1)
000030	00000000		1196	DC A(1)
000030	00000000		1197	DC A(1)
000030	00000000		1198	DC A(1)
000030	00000000		1199	DC A(1)
000030	00000000		1200	DC A(1)
000030	00000000		1201	DC A(1)
000030	00000000		1202	DC A(1)
000030	00000000		1203	DC A(1)
000030	00000000		1204	JCS
000030	00000000		1205	WERGE
000030	00000000		1206	PCD
000030	00000000		1207	INFL
000030	00000000		1208	INFL
000030	00000000		1209	GPT
000030	00000000		1210	MOD
000030	00000000		1211	LTSEQ
000030	00000000		1212	CUTREC
000030	00000000		1213	SUM
000030	00000000		1214	INCLUDE
000030	00000000		1215	SAVEPA
000030	00000000		1216	RETRM
000030	00000000		1217	LTOR:
000030	00000000		1218	DC A(1)
000030	00000000		1219	DC A(1)
000030	00000000		1220	DC A(1)
000030	00000000		1221	DC A(1)
000030	00000000		1222	DC A(1)
000030	00000000		1223	DC A(1)

การคัดลำดับคำสั่งควบคุมการ catalogue โปรแกรมย่อย

```
// JOB CATALR
// PAUSE
// OPTION DECK
// ASSGN SYSPCH, X'280'
// EXEC ASSEMBLY
    PUNCH 'CATALR MODULE'
        } โปรแกรมย่อย MCARD
        }
        } โปรแกรมย่อย MTAPE
        }
        } โปรแกรมย่อย MØDMERGE
/*
// MTC WTM,SYSPCH,2
// MTC REW,SYSPCH
// RESET SYSPCH
// ASSGN SYSIPT,X'280'
// EXEC MAINT
/&
```

การจัดคำสั่งควบคุมการเรียกใช้โปรแกรมย่อยที่ catalogue

```

// JOB MERGERØR
// PAUSE
// ØPTION LINK
// EXEC FFØRTRAN
    }
    }      โปรแกรมภาษา FØRTRAN

/*
    INCLUDE MØDULE
/*
// EXEC LNKEDT
    }
    }      assign card

// EXEC
    }
    }      data card (ถ้ามี)

/*
    }
    }      control statement

@ ENDJCS
/*
/&

```



ภาคผนวก ข

การลำดับคำสั่งควบคุมการทำงานของโปรแกรมสำเร็จรูป

- โดยตรงด้วยคำสั่ง EXEC
- โดยผ่านโปรแกรมภาษา ASSEMBLY
- โดยผ่านโปรแกรมภาษา FORTRAN IV
- โดยผ่านโปรแกรมภาษา COBOL

การจัดลำดับคำสั่งควบคุมการทำงานของโปรแกรมสำเร็จรูปโดยตรงด้วยคำสั่ง

การจัดคำสั่งควบคุมการทำงานของโปรแกรมสำเร็จรูปจะประกอบด้วย

```
// JOB MEXEC
// PAUSE
// OPTION LOG
// ASSGN SYSxxx,X'cuu'
// EXEC SORT
/*
/ &
```

} กำหนดสื่อข้อมูล

} โปรแกรมคำสั่งควบคุม

การจัดลำดับคำสั่งควบคุมการทำงานของโปรแกรมสำเร็จรูปโดยผ่านโปรแกรมภาษา

### ASSEMBLY

การจัดลำดับคำสั่งควบคุมการทำงานของโปรแกรมสำเร็จรูปประกอบด้วย

```
// JOB MASSEM
// PAUSE
// OPTION LINK,LOG
// EXEC ASSEMBLY
```

} โปรแกรม ASSEMBLY

```
/*
```

```
// EXEC LNKEDT
// ASSGN SYSxxx,X'cuu'
```

} การกำหนดสื่อข้อมูล

```
// EXEC
```

```
/*
```

```
/&
```

การจัดลำดับคำสั่งควบคุมการทำงานของโปรแกรมสำเร็จรูปโดยผ่านโปรแกรม FØRTRAN

## IV

การจัดคำสั่งควบคุมการทำงานของโปรแกรมประกอบด้วย

```

// JOB MFØRTRAN
// PAUSE
// ØPTION LINK,LOG
// EXEC FFØRTRAN
                                     } โปรแกรม FØRTRAN
/*
// EXEC ASSEMBLY
                                     } โปรแกรม ASSEMBLY
/*
// EXEC LNKEDT
// ASSGN SYSxxx,X'cuu' } การกำหนดสื่อข้อมูล
// EXEC
                                     } ข้อมูลกรณีใช้ MCARD ถ้าใช้ MTAPE จะไม่มี
/*                                  ส่วนนี้
@ .....
                                     } บั้ตรคำสั่งควบคุม
@ ENDJCS
/*
/&

```

การติดตั้งคำสั่งควบคุมการทำงานของโปรแกรมสำเร็จรูปโดยผ่านโปรแกรม

การติดตั้งคำสั่งควบคุมการทำงานของโปรแกรมประกอบด้วย

```
// JOB MCØBØL
```

```
// PAUSE
```

```
// ØPTION LINK,LOG
```

```
// EXEC FCØBØL
```

} โปรแกรมภาษา CØBØL

```
/*
```

```
// EXEC LNKEDT
```

```
// ASSGN SYSxxx,X'cuu'
```

} การกำหนดสื่อข้อมูล

```
// EXEC
```

```
/*
```

```
/&
```

## ประวัติ

นายเลิศชาย ฤกษ์ดำรง เกิดวันที่ 25 มกราคม 2500 ที่จังหวัดกรุงเทพมหานคร  
สำเร็จการศึกษาระดับปริญญาตรี สาขาวิชาสถิติ จากคณะวิทยาศาสตร์และอักษรศาสตร์  
มหาวิทยาลัยเกษตรศาสตร์ เมื่อปีการศึกษา 2522 และเข้ารับการศึกษาระดับปริญญาโท  
ที่ภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2523