บทที่ 4

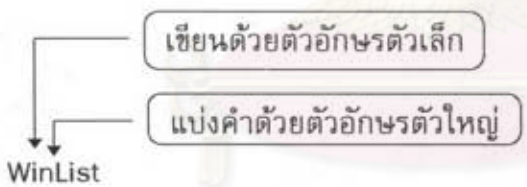
การพัฒนาโปรแกรม

การพัฒนาโปรแกรมได้ใช้ภาษา C++ ในการพัฒนา เพื่อให้สามารถสืบทอดคุณสมบัติของโมดูลหลักและสามารถเพิ่มคุณสมบัติของโมดูลรองลงมาได้โดยสะดวก

การกำหนดชื่อ

1. **การตั้งชื่อโมดูล** ได้กำหนดการตั้งชื่อของโมดูลเป็นดังนี้
   1) ใช้ชื่อเต็มหรือชื่อย่อที่สามารถสื่อความหมายได้
   2) ใช้ตัวอักษรเล็ก และใช้ตัวอักษรใหญ่ในการแบ่งคำ

ตัวอย่าง



อธิบายได้ดังนี้

โมดูลชื่อ WinList ประกอบด้วยคำ 2 คำคือ Win และ List

2. **การตั้งชื่อตัวแปร** ในการออกแบบได้กำหนดการตั้งชื่อของตัวแปรเป็นดังนี้
   1) ใช้ชื่อเต็มหรือชื่อย่อที่สามารถสื่อความหมายได้
   2) เขียนด้วยตัวอักษรเล็ก และใช้ตัวอักษรใหญ่ในการแบ่งคำ

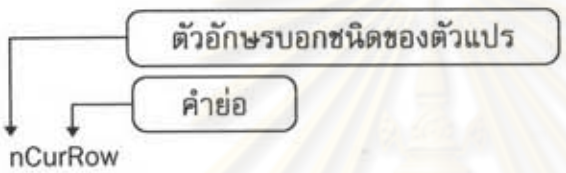
ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

3) ประกอบด้วยตัวอักษรบอกชนิดของตัวแปร มีรายละเอียดดังนี้

ตารางที่ 4.1 แสดงตัวอักษรที่ใช้แสดงชนิดของตัวแปร

ตัวอักษร	ชนิดของตัวแปร
n	integer
c	char
p	pointer

ตัวอย่าง



การกำหนดโมดูล

ในการพัฒนาโปรแกรมได้พัฒนาโดยแบ่งเป็นโมดูลใหญ่ๆ 3 ประเภทคือ

1) โมดูลโครงสร้าง เป็นโมดูลที่ใช้สร้างโครงสร้างที่กำหนดขึ้นเพื่อใช้ในการสร้างโปรแกรมจอภาพ การจัดเก็บข้อมูลในแต่ละโครงสร้างใช้รายการโยงคู่ในการเก็บ ทำให้สามารถเข้าถึงข้อมูลตัวถัดไปและตัวก่อนได้ง่าย รวมทั้งสดวกในการแสดงผล

2) โมดูลข้อมูล เป็นโมดูลที่ใช้เก็บข้อมูลที่จัดเก็บในโครงสร้างแต่ละโครงสร้างตามความเหมาะสม

3) โมดูลหลักและโมดูลช่วยเหลืออื่นๆ เป็นโมดูลหลักที่ใช้เชื่อมโยงโมดูลอื่นๆ และโมดูลช่วยในการทำงานต่างๆ

โมดูลโครงสร้าง

ได้แบ่งโมดูลโครงสร้างโดยคำนึงถึงการจัดเก็บข้อมูลต่างๆที่จำเป็นในการสร้างโปรแกรม ทำให้โมดูลโครงสร้างประกอบด้วยโมดูลย่อยดังนี้

1) โมดูล WinList เป็นโมดูลที่ใช้ในการแสดงผลข้อมูลแบบลำดับบนหน้าต่าง

2) โมดูล VarWin เป็นโมดูลที่ใช้เก็บข้อมูลของตัวแปรหรือเซตข้อมูลทั้งหมด เพื่อใช้อ้างอิงในการสร้างภาษาอาร์พีจี

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

3) โมดูล Rpg เป็นโมดูลหลักใช้ในการติดต่อของโมดูลโครงสร้างอื่นในการเก็บสำรองข้อมูลลงสู่แฟ้มข้อมูล ตลอดจนเป็นแกนในการสร้างภาษาอาร์พีจีและข้อกำหนดของจอภาพ

4) โมดูล TaskWin เป็นโครงสร้างควบคุม ใช้เก็บที่อยู่ของโครงสร้างอื่นๆ และโปรแกรมหลักในการติดต่อกับโมดูลโครงสร้างอื่น

5) โมดูล DsWin เป็นโครงสร้างข้อมูล ใช้เก็บที่อยู่ของตัวแปร

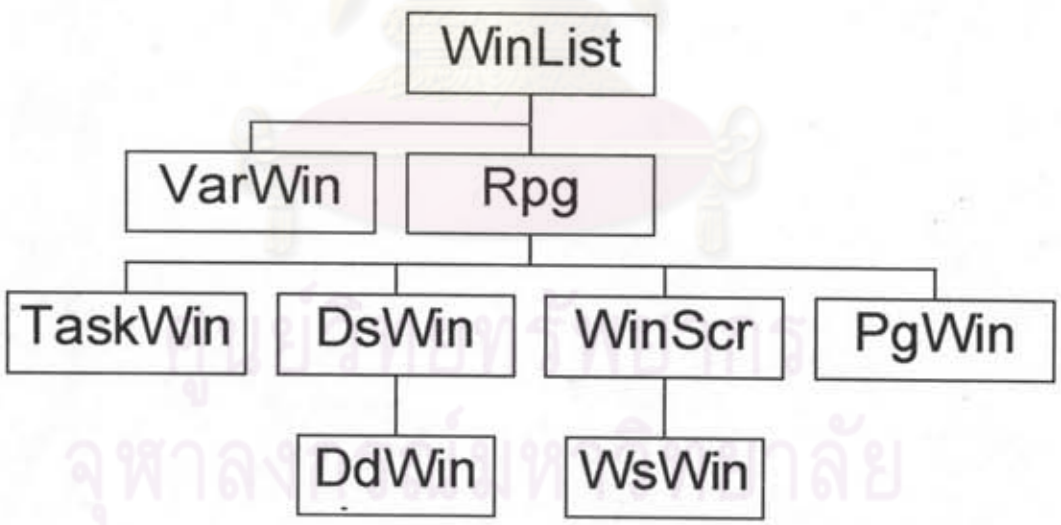
6) โมดูล DdWin เป็นโครงสร้างของสารบัญแฟ้มข้อมูล ใช้เก็บที่อยู่ของเขตข้อมูลและดัชนีของแฟ้มข้อมูล

7) โมดูล WinScr ใช้ในการปรับวิธีการแสดงผลจากเรียงแถวเป็นแบบเรียงลำดับตำแหน่งบนจอภาพ

8) โมดูล WsWin เป็นโครงสร้างจอภาพ ใช้เก็บส่วนประกอบต่างๆของจอภาพ เช่น ข้อความ ตัวแปรที่ใช้ การกำหนดแป้นพิมพ์พิเศษ

9) โมดูล PgWin เป็นโครงสร้างโปรแกรมย่อย ใช้เก็บโปรแกรมย่อย

ความสัมพันธ์ของโมดูลโครงสร้างแสดงได้ดังรูป



รูปที่ 4.1 แสดงความสัมพันธ์ของโมดูลโครงสร้างที่ใช้เก็บข้อมูล

โมดูลข้อมูล

เป็นโมดูลซึ่งการใช้งานจะขึ้นอยู่กับความต้องการของโมดูลโครงสร้าง และได้ออกแบบให้ประกอบด้วยโมดูลย่อยดังนี้

1) โมดูล ObjElem เป็นโมดูลแม่ของโมดูลอื่นๆ และยังใช้เป็นหลักในการเรียกใช้งานโมดูลรอง โดยการสืบทอดคุณสมบัติ

2) โมดูล TaskElem ใช้เก็บข้อมูลตำแหน่งที่อยู่ของโครงสร้างอื่นๆ

3) โมดูล DsElem ใช้เก็บข้อมูลตำแหน่งที่อยู่ของตัวแปร

4) โมดูล WsElem ใช้เก็บข้อความรวมถึงลักษณะในการแสดงผลบนจอภาพ เก็บที่อยู่ของโปรแกรมย่อยที่ระบุการทำงานหน้าที่ก่อนและหลังการแสดงผล และเป็นโมดูลหลักของโมดูลข้อมูลของโมดูลจอภาพ

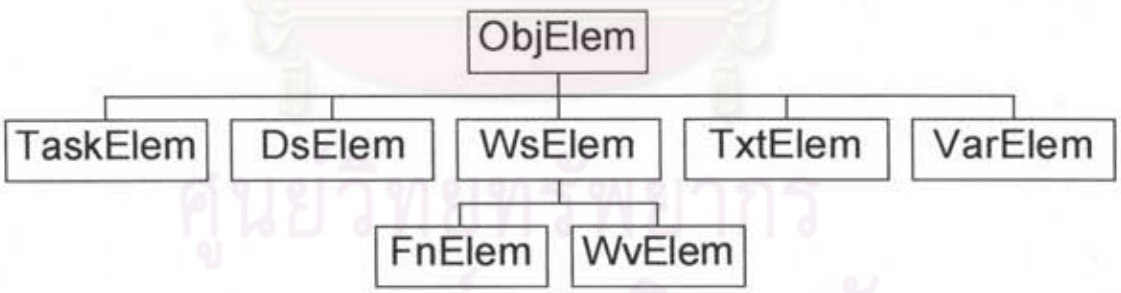
5) โมดูล WvElem ใช้เก็บข้อมูลที่อยู่ของตัวแปรสำหรับโครงสร้างจอภาพ

6) โมดูล FnElem ใช้เก็บข้อมูลการทำงานของแป้นพิมพ์พิเศษ โครงสร้างจอภาพ

7) โมดูล TxtElem ใช้เก็บข้อความต่างๆ รวมถึงใช้เก็บ ตัวเลข และตำแหน่งที่อยู่เพื่อใช้อ้างอิงภายในโปรแกรม

8) โมดูล VarElem ใช้เก็บข้อมูลของตัวแปรและเขตข้อมูลทั้งหมดของระบบ เนื่องจากข้อมูลมีการซ้ำซ้อนกัน

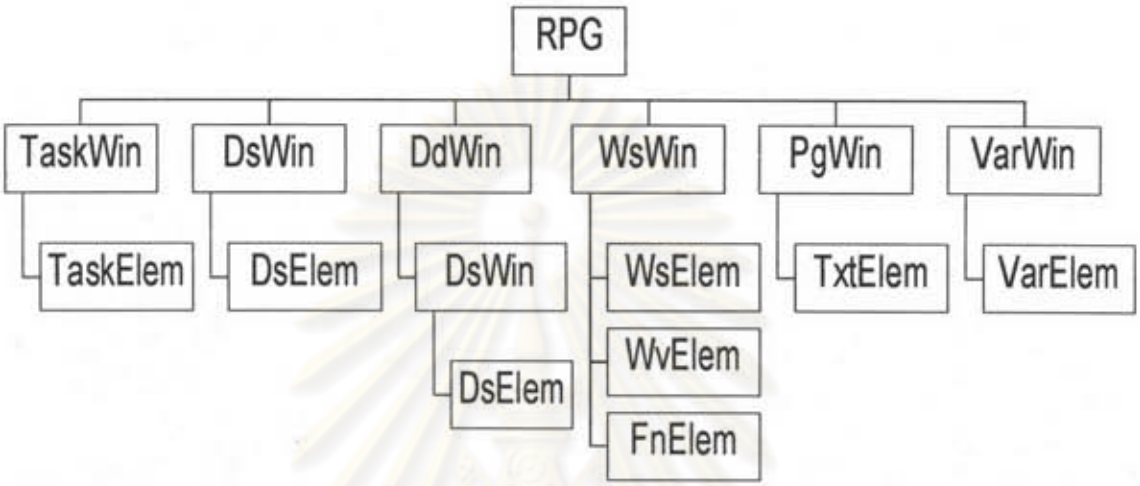
ความสัมพันธ์ของโมดูลข้อมูลแสดงได้ดังรูป



รูปที่ 4.2 แสดงความสัมพันธ์ของโมดูลข้อมูล

ความสัมพันธ์ระหว่างโมดูลโครงสร้างกับโมดูลข้อมูล

ในการจัดข้อมูลจำเป็นต้องแยกการจัดเก็บตามโครงสร้างชนิดต่าง ๆที่ได้ออกแบบไว้ มีดังนี้



รูปที่ 4.3 แสดงความความสัมพันธ์ของโมดูลตามโครงสร้างกับโมดูลข้อมูล

จากรูปสามารถอธิบายความสัมพันธ์ระหว่างโมดูลโครงสร้างกับโมดูลข้อมูลได้ดังนี้

1) โมดูล Rpg ใช้เป็นศูนย์กลางของโมดูลโครงสร้าง
2) โมดูล TaskWin ใช้ควบคุมดูแลโมดูลอื่น ๆโดยใช้โมดูล TaskElem
3) โมดูล DsWin เก็บข้อมูลโดยใช้โมดูล DsElem
4) โมดูล DdWin การเก็บข้อมูลจะดำเนินการผ่านโมดูล DsWin โดยอาศัยคุณสมบัติการสืบทอด
5) โมดูล WsWin มีการเก็บข้อมูลอยู่หลายชนิดดังนี้
   ก) ข้อความคงที่ต่าง ๆ เก็บข้อมูลโดยใช้โมดูล WsElem
   ข) ตัวแปร เก็บข้อมูลใช้โมดูล WvElem
   ค) แป้นพิมพ์พิเศษ เก็บข้อมูลใช้โมดูล FnElem
6) โมดูล PgWin เก็บโปรแกรมที่เขียนขึ้นแต่ละบรรทัดโดยใช้โมดูล TxtElem
7) โมดูล VarWin เก็บตัวแปรและเขตข้อมูลที่สร้างขึ้นโดยใช้โมดูล VarElem

โมดูลหลักและโมดูลช่วยอื่น ๆ

ในการพัฒนาโมดูลต่าง ๆที่เป็นโมดูลหลักและโมดูลช่วยอื่นๆ มีรายละเอียดดังนี้

1) โมดูล GenRpg เป็นโมดูลหลักใช้ในการเริ่มต้นการทำงานของโปรแกรม
2) โมดูล List เป็นโมดูลที่ใช้ในการดำเนินการเชื่อมโยงข้อมูลต่าง ๆเข้าด้วยกัน
3) โมดูล Window เป็นโมดูลใช้ในการแสดงผลแบบหน้าต่าง
4) โมดูล Editor เป็นโมดูลที่ใช้ในการแก้ไขข้อความ
5) โมดูล Dialog เป็นโมดูลที่ใช้ในการโต้ตอบคำถาม
6) โมดูล Analysis เป็นโมดูลใช้ในการวิเคราะห์ภาษาอาร์พีจี
7) โมดูล Box เป็นโมดูลในการปรับการแสดงผลแบบแนวนอน
8) โมดูล Cursor เป็นโมดูลใช้ในการควบคุมขนาดของตัวชี้ตำแหน่ง
9) โมดูล Keyboard เป็นโมดูลใช้ในการตรวจสอบการกดแป้นพิมพ์
10) โมดูล Util เป็นโมดูลใช้งานทั่ว ๆไป

รายละเอียดของโมดูลในแต่ละโมดูลที่ได้ออกแบบ

โมดูลแต่ละโมดูลได้ออกแบบลักษณะโปรแกรมดังนี้

1. **โมดูล GenRpg** เป็นโมดูลหลักใช้ในการเริ่มต้นของระบบ มีรายละเอียดดังนี้

ตารางที่ 4.2 แสดงส่วนประกอบของโมดูล GenRpg

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม	void main();	เป็นโปรแกรมเริ่มต้นของระบบ

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

2. **โมดูล List** เป็นโมดูลภายนอกที่ได้นำมาปรับปรุงให้เหมาะสมในการใช้งาน ประกอบด้วย

2.1 **Class ListElem** เป็นศูนย์กลางของโมดูลข้อมูล มีรายละเอียดดังนี้

ตารางที่ 4.3 แสดงส่วนประกอบของ class ListElem

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม		
	int isEqual(ListElem *other)	ใช้เปรียบเทียบเท่ากัน
	int isMore(ListElem *)	ใช้เปรียบเทียบมากกว่า
	int isLess(ListElem *other)	ใช้เปรียบเทียบน้อยกว่า
	~ListElem()	ใช้ยกเลิก

2.2 **โครงสร้าง Node** เป็นโครงสร้างที่ใช้ในการเชื่อมโยงข้อมูลของโมดูลข้อมูล มีรายละเอียดดังนี้

```
struct Node {
        public:
           ListElem *data;
           Node     *next;
           Node     *prev;
        };
```

2.3 **Class List** ทำหน้าที่ในการจัดการเชื่อมโยงโมดูลข้อมูลต่างๆเข้าด้วยกัน การเชื่อมโยงข้อมูลกระทำผ่านโครงสร้างข้อมูล Node มีรายละเอียดดังนี้

ตารางที่ 4.4 แสดงส่วนประกอบของ class List

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม		
	List()	ใช้ตั้งค่าเริ่มต้น
	ListElem *getHead()	ให้ค่าข้อมูลตัวแรก
	ListElem *getTail()	ให้ค่าข้อมูลตัวสุดท้าย
	void addHead(ListElem *newItem)	เพิ่มข้อมูลส่วนหัว

ตารางที่ 4.4 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	void addTail(ListElem *newItem)	เพิ่มข้อมูลส่วนท้าย
	ListElem *removeHead()	ลบข้อมูลตัวแรก
	ListElem *removeTail()	ลบข้อมูลตัวสุดท้าย
	void initIterator()	ย้ายตำแหน่งไปยังข้อมูลตัวแรก
	ListElem *getNext()	ย้ายไปตำแหน่งถัดไป
	ListElem *getPrev()	ย้ายไปตำแหน่งก่อนหน้า
	int find(ListElem *searchItem)	ค้นหาข้อมูล
	int insertBefore(ListElem *newItem)	แทรกข้อมูลก่อนตำแหน่งบัจจุบัน
	int insertAfter(ListElem *newItem)	แทรกข้อมูลหลังตำแหน่งบัจจุบัน
	ListElem *remove()	ใช้ลบข้อมูล
	int isEmpty()	ใช้ตรวจสอบว่ามีข้อมูลหรือไม่
	~List()	ใช้ยกเลิก

3. **โมดูล window** เป็นโมดูลภายนอก ที่ได้นำมาปรับปรุงให้เหมาะสมในการใช้งาน ประกอบด้วย

3.1 **โปรแกรมการแสดงผลบนจอภาพ** ใช้ในการกำหนดการแสดงผลบนจอภาพ มีรายละเอียดดังนี้

ตารางที่ 4.5 แสดงโปรแกรมการแสดงผลบนจอภาพของโมดูล window

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	static char far *vid_mem = NULL	ใช้ชี้ตำแหน่งของหน่วยความจำของจอภาพ
โปรแกรม	void set_v_ptr()	ใช้ในการกำหนดตำแหน่งเริ่มต้นของหน่วยความจำของจอภาพ
	int video_mode(void)	ใช้ในการทดสอบชนิดของจอภาพ
	void goto_xy(int x, int y)	ใช้ในการเคลื่อนย้ายตำแหน่งของตัวชี้ตำแหน่ง บนจอภาพ

ตารางที่ 4.5 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	void write_char(int x, int y, char ch, int attrib)	ใช้แสดงตัวอักษรบนจอภาพ
	void write_string(int x, int y, char *p, int attrib)	ใช้แสดงข้อความบนจอภาพ

3.2 **Class wintype** ใช้ในการแสดงผลแบบหน้าต่าง โดยทำการสืบทอดคุณสมบัติมาจากโมดูล KeyBoard มีรายละเอียดดังนี้

ตารางที่ 4.6 แสดงส่วนประกอบของ class wintype

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	int leftx	ใช้ระบุตำแหน่งด้านซ้าย
	int upy	ใช้ระบุตำแหน่งด้านบน
	int rightx	ใช้ระบุตำแหน่งด้านขวา
	int downy	ใช้ระบุตำแหน่งด้านล่าง
	int border	ชนิดของหน้าต่าง 1 มีกรอบ 0 ไม่มีกรอบ
	int active	สถานะการเปิดใช้งาน
	char *title	ชื่อของหน้าต่าง
	int curx, cur	ใช้เก็บตำแหน่งของตัวชี้ตำแหน่ง
	char *buf	ระบุที่เก็บข้อมูลของจอภาพก่อนการแสดงหน้าต่าง
	char color	ใช้เก็บกำหนดสีที่ใช้ในหน้าต่าง
โปรแกรม		
	void save_screen()	ใช้เก็บจอภาพเพื่อใช้แสดงหลังการปิดหน้าต่าง
	void restore_screen()	ใช้ในการแสดงจอภาพที่เก็บไว้หลังการปิดหน้าต่าง
	void display_title()	ใช้แสดงชื่อของหน้าต่าง

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 4.6 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	wintype(int uy, int lx, int dy, int rx, int b= BORDER, char *pmess = "")	ใช้สร้างหน้าต่าง
	~wintype()	ใช้ยกเลิกหน้าต่าง
	void winput()	ใช้เปิดหน้าต่าง
	void winremove()	ใช้ปิดหน้าต่าง
	void wintitle(char *s,int pos = CENTER)	ใช้แสดงข้อความบริเวณส่วนหัวของหน้าต่าง
	void winfoot(char *s ,int pos = CENTER)	ใช้แสดงข้อความบริเวณส่วนหัวของหน้าต่าง
	int wincol()	ให้ค่าด้านซ้ายของหน้าต่าง
	int winrow()	ให้ค่าด้านบนของหน้าต่าง
	void winmove(int y, int x)	ใช้ย้ายหน้าต่างไปยังตำแหน่งใหม่
	void winup()	เคลื่อยย้ายหน้าต่างขึ้น
	void windown()	เคลื่อยย้ายหน้าต่างลง
	void winleft()	เคลื่อยย้ายหน้าต่างไปทางซ้าย
	void winright()	เคลื่อยย้ายหน้าต่างไปทางขวา
	void winborder()	ใช้วาดกรอบบนจอภาพ
	int winputs(char *s)	ใช้แสดงข้อความบนหน้าต่าง
	int winxy(int x, int y)	ย้ายตำแหน่งของตัวชี้ตำแหน่งไปยังแกน x และ y ที่กำหนด
	int winrc(int row, int col)	ย้ายตำแหน่งของตัวชี้ตำแหน่งไปยังแถวและคอลัมน์ที่กำหนด
	void wingets(char *s)	ใช้รับข้อความเข้าทางหน้าต่าง
	int wingetche()	รับข้อมูล 1 ตัวอักษรเข้าทางหน้าต่างและแสดงข้อมูลบนจอภาพ
	int wingetch()	รับข้อมูล 1 ตัวอักษรเข้าทางหน้าต่าง
	void wincls()	ใช้ลบข้อความในหน้าต่างทั้งหมด
	void wincleol()	ใช้ลบข้อความในบรรทัดจนหมด
	int winwidth()	ให้ค่าความกว้างของหน้าต่าง
	int winhight()	ให้ค่าความสูงของหน้าต่าง
	void setcolor(char c)	กำหนดสีที่ใช้แสดงผล

ตารางที่ 4.6 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	char getcolor() void setbkcolor(char c) char getbkcolor() friend wintype &operator<<(wintype &o, char *s) friend wintype &operator>>(wintype &o, char *s)	ให้ค่าสีที่ใช้แสดงผล กำหนดสีที่ใช้แสดงหลังฉาก ให้ค่าสีที่ใช้แสดงหลังฉาก ใช้แสดงข้อความบนหน้าต่าง ใช้รับข้อความบนจอหน้าต่าง

4. **โมดูล Editor** เป็นโมดูลที่ใช้ในการป้อนข้อมูลทางจอภาพ ประกอบด้วย

4.1 **โปรแกรมตรวจสอบความถูกต้อง** ใช้ในการตรวจสอบความถูกต้องของตัวแปรภาษาอาร์พีจี

ตารางที่ 4.7 แสดงโปรแกรมตรวจสอบความถูกต้องของโมดูล Editor

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อกำหนด	#define CHAR_FIELD " ABCDEFGHIJKLMNOPQRSTUVWXYXZ 0123456789@#$"	ใช้กำหนดตัวอักษรที่ใช้เป็นตัวแปรในภาษาอาร์พีจีได้
โปรแกรม	int IsValidName(char *cText)	ใช้ตรวจสอบความถูกต้องของชื่อตัวแปร

4.2 **class Editor** ใช้ป้อนข้อมูลทางจอภาพ มีรายละเอียดดังนี้

ตารางที่ 4.8 แสดงส่วนประกอบของ class Editor

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	wintype *pWin wintype *pWinMsg wintype *pEdStatus	ใช้ชี้หน้าต่างที่ใช้ในการป้อนข้อมูล ใช้ชี้หน้าต่างที่แสดงข่าวสาร ใช้ชี้หน้าต่างแสดงสถานะ

ตารางที่ 4.8 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	char cString[81]	ใช้เก็บข่าวสารที่จะแสดง
	int nCurPos	ใช้ระบุตำแหน่งของตัวชี้ตำแหน่ง
	int nStatus	ใช้ระบุการแสดงผลบนหน้าต่างสถานะ
	int nInsert	ใช้ระบุการแทรกข้อมูล
	int nConfirm	ใช้ระบุสถานะการยืนยันการสิ้นสุดการแก้ไข
	int nExit[MAX_KEY]	ใช้ระบุแป้นพิมพ์ที่ใช้ในการสิ้นสุดการป้อนข้อมูล
	int nEditColor	สีของหน้าต่างที่ใช้ป้อนข้อมูล
	int nNormalColor	สีเดิมของหน้าต่าง
	int nUpperCase	สถานะการเปลี่ยนเป็นตัวอักษรใหญ่
โปรแกรม		
	Editor(wintype *pWin)	ใช้กำหนดค่าเริ่มต้น
	~Editor()	ใช้ยกเลิก
	void Status(int nMode = OFF)	ใช้กำหนดการแสดงผลบนหน้าต่างสถานะ
	void Insert(int nMode = ON)	ใช้กำหนดสถานะการแทรกข้อมูล
	void MultiEdit(int nMode = ON)	ใช้กำหนดแป้นพิมพ์ลูกศรขึ้นลงในสิ้นสุดการป้อนข้อมูล
	void SetExit(int nKey)	ใช้กำหนดแป้นพิมพ์บอกการสิ้นสุดการป้อนข้อมูล
	void ResetExit(int nKey)	ใช้ยกเลิกแป้นพิมพ์บอกการสิ้นสุดการป้อนข้อมูล
	int Terminate(int nKey)	ใช้ตรวจสอบการสิ้นสุดการป้อนข้อมูล
	void Confirm(int nFull = ON)	ใช้กำหนดให้มีการยืนยันการสิ้นสุดการป้อนข้อมูล
	int AutoExit(int nFull)	ใช้ตรวจสอบการสิ้นสุดการป้อนข้อมูลแบบอัตโนมัติ
	void UpperCase(int nToUpper= OFF)	ใช้กำหนดให้เปลี่ยนตัวอักษรที่ป้อนเข้าเป็นตัวใหญ่
	int LastPos()	ให้ค่าตำแหน่งสุดท้ายของตัวชี้ตำแหน่ง

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 4.8 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	int Edit(int nRow, int nCol, char *s, char *legal,int maxlength,int pos=0)	ใช้ในการแก้ไขข้อความ
	int Edit (int nRow, int nCol, char * cString, int maxlength = 0,int nStart=0)	ใช้ในการป้อนข้อมูลชนิดข้อความ
	int EdField(int nRow, int nCol, char *cString, int maxlength = 0,int nStart=0)	ใช้ในการป้อนข้อมูลชนิดตัวแปรหรือเขตข้อมูล
	int Edit(int nRow, int nCol, int * pNumber, int maxlength = 0)	ใช้ในการป้อนข้อมูลชนิดตัวเลข
	int YesNo(int nRow, int nCol, int * pNumber = 0)	ใช้ในการป้อนข้อมูลลอจิก
	void Display(int nRow, int nCol, char *cString)	ใช้แสดงข้อความ
	void Display(int nRow, int nCol, int nNumber, int maxlength = 5)	ใช้แสดงตัวแลข
	char *SayYesNo(int nYesOrNo)	ใช้แสดงข้อความ YES หรือ NO
	void SetMessage(char *cString = "")	ใช้กำหนดข้อความบอกข่าวสาร
	void Message( )	ใช้แสดงข่าวสาร
	void ShowCol(int nPos)	ใช้ในการแสดงคอลัมที่แก้ไข

5. **โมดูล Dialog** เป็นโมดูลที่ใช้ในการแจ้งข่าวสารต่างๆบนจอภาพ ประกอบด้วย class Dialog มีรายละเอียดดังนี้

ตารางที่ 4.9 แสดงส่วนประกอบของ class Dialog

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม	int YesNoBox(char *cString1 = "", char *cString2 = "")	ใช้ในการเลือกตอบคำถาม

ตารางที่ 4.9 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	int MessageBox(char *cString1 = "", char *cString2 = "") int Center(char *cString, int nWidth)	ใช้แสดงข้อความบนหน้าต่าง ใช้จัดข้อความให้อยู่ตรงกลาง

6. **โมดูล Analysis** เป็นโมดูลที่ใช้ในการตรวจสอบความถูกต้อง สำหรับแบบฟอร์ม C ของภาษาอาร์พีจี ประกอบด้วย

6.1 **ข้อความแสดงข้อผิดพลาด** ใช้บอกข้อผิดพลาดที่เกิดขึ้นจากการทดสอบ มีรายละเอียดดังนี้

ตารางที่ 4.10 แสดงข้อความแสดงข้อผิดพลาดของโมดูล Analysis

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	static char *LineMsg[] = { "", "Must be blank", "Not alpha variable", "Not alpha", "Not numeric variable", "Not numeric", "Not alphanumeric variable", "Not alphanumeric", "Unknow routine", "Unknow file name", "Must be text"};	ใช้ระบุข้อผิดพลาดที่เกิดขึ้น

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

6.2 **รหัสแทนคำสั่งภาษาอาร์พีจี** ใช้ในการกำหนดรหัสของคำสั่ง มีรายละเอียดดังนี้

ตารางที่ 4.11 แสดงรหัสแทนคำสั่งภาษาอาร์พีจีของโมดูล Analysis

ลักษณะ	คำสั่ง		คำอธิบาย
ข้อมูล			
	#define RPG_ERROR	0	ใช้แทนคำสังค้นหาไม่พบ
	#define RPG_COMMENT	1	ใช้แทนคำอธิบาย
	#define RPG_OPCODE	2	ใช้แทนคำสั่งที่ไม่ต่อเนื่อง
	#define RPG_DIV	3	ใช้แทนคำสั่ง DIV
	#define RPG_MVR	4	ใช้แทนคำสั่ง MVR
	#define RPG_DO	5	ใช้แทนคำสั่ง DO
	#define RPG_DOUxx	6	ใช้แทนคำสั่ง DOU
	#define RPG_DOWxx	7	ใช้แทนคำสั่ง DOW
	#define RPG_CASxx	8	ใช้แทนคำสั่ง CASxx
	#define RPG_CAS	9	ใช้แทนคำสั่ง CAS
	#define RPG_IF	10	ใช้แทนคำสั่ง IF
	#define RPG_ELSE	11	ใช้แทนคำสั่ง ELSE
	#define RPG_END	12	ใช้แทนคำสั่ง END

6.3 **ตารางคำสั่ง** ใช้ในการตรวจสอบความถูกต้องของคำสั่ง ได้กำหนดโครงสร้างของตารางคำสั่งภาษาอาร์พีจี มีดังนี้

```
struct OpCodeSt {
        int Fac1;
        char *OpCode;
        int Cond;
        int Fac2;
        int Result;
        int Blank2;
        int Token;
};
```

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ได้กำหนดค่าของตัวแปรที่ใช้ในการทดสอบส่วนประกอบคำสั่ง ดังนี้

```
#define RPG_BLANK                0
#define RPG_ALPHA                1
#define RPG_NUMERIC              2
#define RPG_ALPHA_NUMERIC        3
#define RPG_ROUTINE              4
#define RPG_FILE                 5
#define RPG_TEXT                 6
#define RPG_VAR                 70+X
#define RPG_OPTION             128+X
#define RPG_ALPHA_VAR           71
#define RPG_NUMERIC_VAR         72
#define RPG_ALPHA_NUMERIC_VAR   73
```

ได้กำหนดตารางตรวจสอบ มีรายละเอียดดังนี้

```
static OpCodeSt OpCode[] =
{{128+2, "ADD"   ,0 ,  2,   72,      0, RPG_OPCODE },
 {    0, "CAS"   ,0 ,  0,    4,      0, RPG_CAS    },
 {    3, "CAS"   ,1 ,  3,    4,      0, RPG_CASxx  },
 {    1, "CHAIN" ,0 ,  5,    0,      0, RPG_OPCODE },
 {    0, "DEL"   ,0 ,  5,    0,      0, RPG_OPCODE },
 {    2, "DIV"   ,0 ,  2,   72,      0, RPG_DIV    },
 {    2, "MVR"   ,0 ,  0,   72,      0, RPG_MVR    },
 {    0, "DO"    ,0 ,  0,    0,      0, RPG_DO     },
 {    3, "DOU"   ,1 ,  3,    0,      0, RPG_DOUxx  },
 {    3, "DOW"   ,1 ,  3,    0,      0, RPG_DOWxx  },
 {    0, "ELSE"  ,0 ,  0,    0,      0, RPG_ELSE   },
 {    0, "END"   ,0 ,  0,    0,      0, RPG_END    },
 {    0, "EXSR"  ,0 ,  4,    0,      0, RPG_OPCODE },
 {    3, "IF"    ,1 ,  3,    0,      0, RPG_IF     },
 {    0, "MOVE"  ,0 ,  3,   73,      0, RPG_OPCODE },
 {    0, "MOVEL" ,0 ,  3,   73,      0, RPG_OPCODE },
 {   71, "MSG"   ,0 ,  6,128+6,128+6, RPG_OPCODE },
 {128+2, "MULT"  ,0 ,  2,   72,      0, RPG_OPCODE },
 {    0, "READ"  ,0 ,  5,    0,      0, RPG_OPCODE },
 {128+2, "SUB"   ,0 ,  2,   72,      0, RPG_OPCODE },
 {    0, "UPDAT" ,0 ,  5,    0,      0, RPG_OPCODE },
 {    0, "WRITE" ,0 ,  5,    0,      0, RPG_OPCODE },
 {    0, "Z-ADD" ,0 ,  2,   72,      0, RPG_OPCODE },
 {    0, "Z-SUB" ,0 ,  2,   72,      0, RPG_OPCODE }};
```

6.4 **ตารางคำสงวน** ใช้สำหรับตรวจสอบคำสงวน แบ่งเป็น 2 ชนิดคือ

6.4.1 ชนิดตัวอักษร ใช้ในการตรวจสอบคำสงวนที่เป็นตัวอักษร มีรายละเอียดดังนี้

```
static char *RpgAlphaConst[MAX_ALPHA_CONST] =
        { "*ZERO",
        "*ZEROS",
        "*BLANK",
        "*BLANKS",
        "#FOUND",
        "#EOF"};
```

6.4.2 ชนิดตัวเลข ใช้ในการตรวจสอบคำสงวนที่เป็นตัวตัวเลข มีรายละเอียดดังนี้

```
static char *RpgNumericConst[MAX_NUMERIC_CONST] =
        { "*ZERO",
        "*ZEROS",
        "UDATE",
        "UMONTH",
        "UYEAR"};
```

6.5 **class Analysis** ใช้ในการตรวจสอบข้อผิดพลาดของคำสั่งภาษาอาร์พีจี มีรายละเอียดดังนี้

ตารางที่ 4.12 แสดงส่วนประกอบของ class Analysis

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	WinList *pSource	ใช้ชี้ตำแหน่งของโปรแกรมที่ทดสอบ
	int nLine	ใช้เก็บจำนวนบรรทัดของโปรแกรม
	int nError	ใช้นับจำนวนข้อผิดพลาด
	int nLastOpCode	ใช้เก็บคำสั่งล่าสุด
	char cBlank1[12]	ใช้ในการแบ่งคำสั่ง ส่วนที่ 1
	char cFac1[11]	ใช้ในการแบ่งคำสั่ง ส่วนที่ 2

ตารางที่ 4.12 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	char cOpCode[7]	ใช้ในการแบ่งคำสั่ง ส่วนที่ 3
	char cFac2[11]	ใช้ในการแบ่งคำสั่ง ส่วนที่ 4
	char cResult[7]	ใช้ในการแบ่งคำสั่ง ส่วนที่ 5
	char cBlank2[12]	ใช้ในการแบ่งคำสั่ง ส่วนที่ 6
	int nStack[STACK_SIZE]	ใช้ในการกำหนดระดับของการคำสั่งแบบโครงสร้าง
	int nTopStack	ใช้ชี้ตำแหน่งบนสุดของกองซ้อน
	int IsStackEmpty()	ใช้ตรวจสอบกองซ้อนว่าง
	int IsStackFull()	ใช้ทดสอบกองซ้อนเต็ม
	int StackFirst()	ให้ค่าบนสุดของกองซ้อน
	int Push(int nData)	ใส่ข้อมูลลงกองซ้อน
	int Pop( )	นำข้อมูลออกจากกองซ้อน
โปรแกรม		
	Analysis(WinList *pSource)	กำหนดค่าเริ่มต้นของโมดูลทดสอบ
	~Analysis()	ใช้ยกเลิก
	int Check()	เริ่มต้นทำการตรวจ
	void Error(char *cMsg1, char *	ใช้แสดงข้อผิดพลาด
	cMsg2 = "", char *cMsg3 = "")	
	void Extract(char *cText)	ใช้แบ่งคำสั่งออกเป็นส่วนๆ
	int CkOpCode()	ใช้ตรวจสอบคำสั่งกับตารางคำสั่ง
	int IsComment()	ใช้ตรวจสอบบรรทัดที่เป็นคำอธิบาย
	void CkLine(int i)	ใช้ตรวจสอบความถูกต้องของส่วนประกอบอื่นๆ
	int CkType(int nType, char *cText)	ใช้ตรวจสอบความถูกต้องของส่วนประกอบแต่ละส่วน
	int IsAlpha(char *cText)	ใช้ตรวจสอบตัวแปรชนิดตัวอักษร
	int IsAlphaConst(char *cText)	ใช้ตรวจสอบค่าคงที่ชนิดตัวอักษร
	int IsNumeric(char *cText)	ใช้ตรวจสอบตัวแปรชนิดตัวเลข
	int IsNumericConst(char *cText)	ใช้ตรวจสอบค่าคงที่ชนิดตัวตัวเลข
	int IsVar(char *cText)	ใช้ตรวจสอบตัวแปร
	int IsRoutine(char *cText)	ใช้ตรวจสอบโปรแกรมย่อย
	int IsFile(char *cText)	ใช้ตรวจสอบแฟ้มข้อมูล

7. **โมดูล Box** เป็นโมดูลใช้ในการปรับการแสดงผลเป็นแบบซ้ายขวา ประกอบด้วย class Box ซึ่งสืบทอดคุณสมบัติมาจาก class WinList มีรายละเอียดดังนี้

ตารางที่ 4.13 แสดงส่วนประกอบของโมดูล Box

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม		
	Box(int nURow, int nUCol, int nDRow, int nDCol, int nBorder=1, char *cTitle = "")	ใช้ในการกำหนดค่าเริ่มต้น
	~Box()	ใช้ยกเลิก
	virtual char *ObjectName()	ให้ชื่อ "Win Box List"
	virtual int ObjectNo()	ให้เลขประจำตัว
	virtual void Draw()	ใช้แสดงผลบนหน้าต่าง
	virtual void ShowOne(int nRow, int nCol, ObjElem *pCurElem)	ใช้แสดงข้อมูล 1 ตัว
	virtual int HandleEvent()	ใช้ตรวจสอบข้อมูลที่รับเข้า

8. **โมดูล Cursor** เป็นโมดูลที่ใช้ควบคุมขนาดของเคอเซอร์ มีรายละเอียดดังนี้

ตารางที่ 4.14 แสดงส่วนประกอบของโมดูลตัวชี้ตำแหน่ง

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	static int CursorShape;	ใช้เก็บขนาดของตัวชี้ตำแหน่ง
โปรแกรม		
	void CursorOn();	ใช้แสดงตัวชี้ตำแหน่ง
	void CursorOff();	ใช้ซ่อนตัวชี้ตำแหน่ง
	void CursorSize(int CursorType);	ใช้กำหนดขนาดของตัวชี้ตำแหน่ง

9. **โมดูล KeyBoard** เป็นโมดูลที่ใช้ในการควบคุมการใช้งานของแป้นพิมพ์ ประกอบด้วย class KeyBoard มีรายละเอียดดังนี้

ตารางที่ 4.15 แสดงส่วนประกอบของโมดูล Keyboard

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	int lastkey	ใช้เก็บข้อมูลที่รับเข้าทางแป้นพิมพ์
	int keystatus	ใช้เก็บสถานะของแป้นพิมพ์
โปรแกรม		
	KeyBoard()	กำหนดค่าเริ่มต้น
	int KeyEvent(void)	รอรับการกดแป้นพิมพ์ 1 ครั้ง
	int KeyPress(void)	รับการกดแป้นพิมพ์
	int LastKey(void)	ให้ค่าแป้นพิมพ์ครั้งสุดท้าย
	int KeyStatus(void)	ให้สถานะแป้นพิมพ์ครั้งสุดท้าย
	int IsShiftArrow(unsigned k)	ตรวจสอบการกดแป้นลูกศร

10. **โมดูล Util** เป็นโมดูลใช้งานทั่วๆไป มีรายละเอียดดังนี้

ตารางที่ 4.16 แสดงส่วนประกอบของโมดูล Utility

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม		
	#define BLANK(x)	ใช้เปลี่ยนข้อความเป็นช่องว่าง
	#define PAD(x,y)	ใช้เพิ่มลดขนาดของข้อความ
	char *FillChar(char *cString, int nLen, char cFill = ' ')	ใช้เติมข้อความด้วยตัวอักษรที่กำหนด
	char *Pad(char *cString, int nLen, char cFill= ' ')	ใช้เพิ่มลดข้อความตามความยาวที่กำหนด
	char *Trim(char *cString)	ใช้ตัดช่องว่างหลังข้อความ
	int Empty(char *cString)	ใช้ทดสอบข้อความที่ไม่มีข้อมูล
	int Match(char *cSource ,char * cMatch)	ใช้ทดสอบข้อความที่เหมือนกัน
	char *GenSeq(char *cSeqName, char *prefix, int nWsId)	ใช้ในการจัดข้อความให้รวมกับตัวเลข

11. **โมดูล WinList** เป็นโมดูลที่ทำหน้าที่ในการแสดงข้อมูลของโครงสร้างต่างๆ ประกอบด้วย class WinList ทำการสืบทอดคุณสมบัติในการจัดการข้อมูลมาจาก class List และการแสดงผลบนหน้าต่างมาจาก class wintype มีรายละเอียดดังนี้

ตารางที่ 4.17 แสดงส่วนประกอบของโมดูล WinList

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	ObjElem *pObjElem	ใช้ระบุตำแหน่งของข้อมูล
	int nCount	จำนวนข้อมูลทั้งหมด
	int nOverList	ใช้กำหนดให้สามารถเลือกข้อมูลที่อยู่ถัดจากตัวสุดท้าย
	int nBeginRow	ใช้กำหนดแถวเริ่มต้นของการแสดงผล
	int nBeginCol	ใช้กำหนดคอลัมน์เริ่มต้นของการแสดงผล
	int nCurRow	ใช้กำหนดแถวปัจจุบัน
	int nCurCol	ใช้กำหนดคอลัมน์ปัจจุบัน
โปรแกรม		
	WinList(int nURow, int nUCol, int nDRow, int nDCol, int nBorder=1, char *cTitle = "")	ใช้กำหนดค่าของ WinList และหน้าต่างที่ใช้
	virtual ~WinList()	ใช้ยกเลิก
	virtual char *ObjectName()	ให้ชื่อ "Window List"
	virtual char *Alias()	ให้ชื่อย่อ "WL"
	virtual int ObjectNo()	ให้เลขประจำตัว
	virtual void Add(ObjElem*pCurElem)	ใช้เพิ่มข้อมูล
	virtual void Delete()	ใช้ลบข้อมูล
	virtual ObjElem *Remove()	ใช้ยกเลิกการเก็บข้อมูล
	virtual ObjElem *Search(ObjElem * pOther)	ใช้ค้นหาข้อมูล
	virtual ObjElem *Search(char * cName)	ใช้ค้นหาข้อความ

ตารางที่ 4.17 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	virtual ObjElem *Near(ObjElem * pOther)	ย้ายตำแหน่งไปใกล้ข้อมูล
	virtual ObjElem *Near(char *cName)	ย้ายตำแหน่งไปใกล้ข้อความ
	virtual ObjElem *Goto(int nPos)	ย้ายตำแหน่งไปยังตำแหน่งใหม่
	virtual int Goto(ObjElem *pOther)	ย้ายตำแหน่งไปยังข้อมูล
	virtual void Display()	ใช้แสดงข้อมูลทั้งหมด
	virtual void SetOver(int nOver)	ใช้กำหนดการเลือกข้อมูล
	virtual int Active(int nPos = 0)	ใช้กำหนดให้เริ่มทำงาน
	virtual void Prompt()	ใช้ในการเปิดหน้าต่างและแสดงข้อมูลอื่นๆ
	virtual void Draw()	ใช้แสดงผลบนจอภาพ
	virtual void ShowOne(int nRow, int nCol, ObjElem *pCurElem)	ใช้แสดงข้อมูล 1 ตัว
	virtual void Event()	ใช้ในการรับข้อมูล
	virtual int GetEvent()	ใช้รับแป้นพิมพ์
	virtual int HandleEvent()	ใช้ตรวจสอบข้อมูลที่รับเข้า
	int GetPosition()	ให้ค่าตำแหน่งปัจจุบัน
	int GetSize()	ให้ค่าจำนวนข้อมูลทั้งหมด
	int Row()	ให้ค่าแถวปัจจุบัน
	int Col()	ให้ค่าคอลัมน์ปัจจุบัน
	int ScreenRow()	ให้จำนวนแถว
	int ScreenCol()	ให้จำนวนคอลัม
	ObjElem *GetObject()	ให้ค่าตำแหน่งของข้อมูลปัจจุบัน
	virtual void Up()	เลื่อนตำหน่งปัจจุบันขึ้น 1 ตำแหน่ง
	virtual void Down()	เลื่อนตำหน่งปัจจุบันลง 1 ตำแหน่ง
	virtual void UnPrompt()	ใช้ปิดการแสดงผล
	virtual void SetPos(int nRow, int nCol)	ใช้กำหนดตำแหน่งของแถวและคอลัมน์ในการแสดงผล

12. **โมดูล VarWin** เป็นโมดูลที่ใช้เก็บโครงสร้างของตัวแปร ประกอบด้วย

12.1 **Class VarWin** ทำการสืบทอดคุณสมบัติมาจาก class WinList มีรายละเอียดดังนี้

ตารางที่ 4.18 แสดงส่วนประกอบของโมดูล VarWin

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม		
	VarWin()	ใช้กำหนดค่าเริ่มต้น
	~VarWin()	ใช้ยกเลิก
	char *ObjectName()	ให้ชื่อ "Variable table"
	char *Alias()	ให้ชื่อย่อ "VT"
	int ObjectNo()	ให้เลขประจำตัว
	void Prompt()	ใช้ในการเปิดหน้าต่างและแสดงข้อมูลอื่นๆ
	int HandleEvent()	ใช้ควบคุมการทำงานจากการกดแป้นพิมพ์
	ObjElem *Insert(ObjElem * NewElem)	ใช้เพิ่มตัวแปร
	void Delete(ObjElem *pNewElem)	ใช้ลบตัวแปร
	void Help()	ใช้แสดงหน้าต่างการใช้แป้นพิมพ์

12.2 **ตัวแปรภายนอก** ใช้ในการกำหนดค่าของตัวแปรต่างๆ มีรายละเอียดดังนี้

ตารางที่ 4.19 แสดงตัวแปรภายนอกของโมดูล VarWin

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	extern VarWin *pVarTable	ใช้ชี้ตำแหน่งของตารางตัวแปร

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

13. **โมดูล Rpg** เป็นโมดูลที่ใช้เป็นหลักในการสร้างภาษาอาร์พีจี มีส่วนประกอบคือ

13.1 **Class Rpg** ซึ่งสืบทอดคุณสมบัติมาจาก class WinList มีรายละเอียดดังนี้

ตารางที่ 4.20 แสดงส่วนประกอบของ class Rpg

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	char cFile[9]	ใช้เก็บชื่อแฟ้มข้อมูล
	char cName[7]	ใช้เก็บชื่อของโครงสร้าง
	char cInfo[15]	ใช้เก็บข้อความเกี่ยวกับโครงสร้าง
	int nId	ใช้เก็บลำดับของโครงสร้าง
โปรแกรม		
	Rpg(int nURow, int nUCol, int nDRow, int nDCol, int nBorder=1, char *cTitle = "")	ใช้กำหนดค่าเริ่มต้น
	~Rpg()	ใช้ยกเลิก
	virtual int GetFile(char *cTitle, char *cHead, char *cFoot)	ใช้รับข้อมูลสำหรับชื่อแฟ้มข้อมูล
	virtual int GetName()	ใช้รับข้อมูลสำหรับชื่อและรายละเอียดโครงสร้าง
	virtual int GetId() {return nId };	ให้เลขประจำตัวของโครงสร้าง
	virtual char *SetName(char *cText)	ใช้กำหนดชื่อของโครงสร้าง
	virtual char *File()	ให้ชื่อแฟ้มข้อมูล
	virtual char *Name()	ให้ชื่อโครงสร้าง
	virtual char *Info()	ให้รายละเอียดโครงสร้าง
	virtual void Save(ofstream& OutFile)	ใช้จัดเก็บโครงสร้าง
	virtual void SaveInfo(ofstream& OutFile)	ใช้จัดเก็บรายละเอียดของโครงสร้าง
	virtual void SaveData(ofstream& OutFile)	ใช้จัดเก็บข้อมูลของโครงสร้าง
	virtual int Load()	ใช้นำโครงสร้างเข้าสู่โปรแกรม

ตารางที่ 4.20 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	virtual void Load(ifstream &InFile)	ใช้นำข้อมูลโครงสร้างเข้าสู่โปรแกรม
	virtual int IsFile(char *cText)	ใช้ตรวจสอบชนิดของข้อความว่าเป็นชื่อแฟ้มข้อมูลหรือไม่
	virtual int IsName(char *cText)	ใช้ตรวจสอบชนิดของข้อความว่าเป็นชื่อโครงสร้างหรือไม่
	virtual int IsInfo(char *cText)	ใช้ตรวจสอบชนิดของข้อความว่าเป็นรายละเอียดหรือไม่
	virtual int IsTerminate(ifstream & InFile)	ใช้ตรวจสอบการสิ้นสุดของโครงสร้าง
	virtual void Create(ofstream &, char *cForm)	ใช้ในการสร้างภาษาอาร์พีจี
	virtual void CreateFile(ofstream &)	ใช้ในการสร้างแบบฟอร์ม F
	virtual void CreateDs(ofstream &)	ใช้ในการสร้างแบบฟอร์ม I
	virtual void CreateDd(ofstream &)	ใช้ในการสร้างแบบฟอร์ม I สำหรับแฟ้มข้อมูลและจอภาพ
	virtual void CreateCalc(ofstream &)	ใช้ในการสร้างแบบฟอร์ม C
	virtual void CreateWs(ofstream &)	ใช้ในการสร้างแบบฟอร์ม C สำหรับการควบคุมการแสดงผลจอภาพ
	virtual void CreateOutput(ofstream &)	ใช้ในการสร้างแบบฟอร์ม O สำหรับแฟ้มข้อมูล
	virtual void CreateFmt(ofstream &)	ใช้ในการสร้างข้อกำหนดของจอภาพ
	virtual int SyntaxCheck()	ใช้ในการตรวจสอบข้อผิดพลาดของโปรแกรม
	void C_FORM(ofstream& OutFile, char *id1, char *id2, char *id3, char *f1, char *op, char *f2, char *re)	ใช้เขียนแบบฟอร์ม C ลงแฟ้มข้อมูล

13.2 **ตัวแปรภายนอก** ใช้ในการกำหนดค่าตัวแปรที่ต่างๆ มีรายละเอียดดังนี้

ตารางที่ 4.21 แสดงตัวแปรภายนอกของโมดูล Rpg

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	*pFileTab	ใช้ชี้ตำแหน่งของตารางชื่อแฟ้มข้อมูล
	*pRtnTab	ใช้ชี้ตำแหน่งของตารางชื่อโปรแกรม
	*pMsgTab	ใช้ชี้ตำแหน่งของตารางข้อความแสดงข้อผิดพลาด
	*pErrWin	ใช้ชี้ตำแหน่งของตารางข้อผิดพลาดจากการตรวจสอบ
	nSeqId	ใช้กำหนดเลขประจำโครงสร้าง

14. **โมดูล TaskWin** เป็นโมดูลที่ใช้เก็บที่อยู่ของโครงสร้างประเภทต่างๆ ประกอบด้วย class TaskWin โดยสืบทอดคุณสมบัติมาจาก class Rpg มีรายละเอียดดังนี้

ตารางที่ 4.22 แสดงส่วนประกอบของโมดูล TaskWin

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	TaskElem *pWsMain	ใช้ระบุถึงจอภาพที่เป็นจุดเริ่มต้น
โปรแกรม	TaskWin()	ใช้ในการตั้งค่าเริ่มต้น
	~TaskWin()	ใช้ยกเลิก
	void Save( )	ใช้จัดเก็บโครงสร้าง
	void SaveData(ofstream& OutFile)	ใช้จัดเก็บข้อมูลของโครงสร้าง
	void LoadData(ifstream &InFile)	ใช้นำข้อมูลเข้าสู่โปรแกรม
	void IsMain(char *cMain, char * cText)	ใช้กำหนดโครงสร้างจอภาพที่เป็นจุดเริ่มต้น
	int Create( )	ใช้ในการสร้างภาษาอาร์พีจีและข้อกำหนดของจอภาพ
	void CreateTemp( )	ใช้ในการสร้างภาษาอาร์พีจี

ตารางที่ 4.22 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	void MergeFile(ofstream &fOutfile, char *cFile)	ใช้ในการรวมส่วนต่างๆของภาษาที่สร้างขึ้นเข้าด้วยกัน
	void CreateMain(ofstream & fOutfile)	ใช้ในการสร้างภาษาอาร์พีจีเพื่อใช้เรียกจอภาพเริ่มต้น
	char *ObjectName()	ให้ชื่อ "Task List"
	char *Alias()	ให้ชื่อย่อ "TL"
	int ObjectNo()	ให้เลขประจำตัว
	void Prompt()	ใช้ในการเปิดหน้าต่างและแสดงข้อมูล
	void ShowOne(int nRow, int, ObjElem *pCurElem)	ใช้ในการแสดงข้อมูล 1 ตัว
	int HandleEvent()	ใช้ควบคุมการทำงานจากการกดแป้นพิมพ์
	void Modify()	ใช้แก้ไขโครงสร้าง
	void Insert()	ใช้เพิ่มโครงสร้าง
	int SelectTask()	ใช้ในการเลือกชนิดของโครงสร้าง
	void Delete()	ใช้ลบโครงสร้าง
	void SetMain()	ใช้กำหนดโครงสร้างจอภาพเริ่มต้น
	void Help()	ใช้แสดงหน้าต่างการใช้แป้นพิมพ์

15. **โมดูล DsWin** เป็นโมดูลที่ใช้ในการกำหนดโครงสร้างข้อมูล ประกอบด้วย class DsWin ซึ่งทำการสืบทอดคุณสมบัติมาจาก class Rpg มีรายละเอียดดังนี้

ตารางที่ 4.23 แสดงส่วนประกอบของโมดูล DsWin

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม		
	DsWin()	ใช้กำหนดค่าเริ่มต้น
	~DsWin()	ใช้ยกเลิก
	void LoadData(ifstream &InFile)	ใช้นำข้อมูลเข้าสู่โปรแกรม
	DsElem *LoadDs(char *cText)	ใช้ในการสร้างตัวแปรจากข้อความ
	void CreateDs(ofstream &OutFile)	ใช้ในการสร้างแบบฟอร์ม I

ตารางที่ 4.23 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	void CreateDsLine(ofstream & OutFile)	ใช้ในการสร้างแบบฟอร์ม I ในแต่ละบรรทัด
	char *ObjectName()	ให้ชื่อ "Data Structure"
	char *Alias()	ให้ชื่อย่อ "DS"
	int ObjectNo()	ให้เลขประจำตัว
	void Prompt()	ใช้ในการเปิดหน้าต่างและแสดงข้อมูลอื่นๆ
	int HandleEvent()	ใช้ควบคุมการทำงานจากการกดแป้นพิมพ์
	void Modify()	ใช้แก้ไขตัวแปร
	void Insert()	ใช้เพิ่มตัวแปร
	void Delete()	ใช้ลบตัวแปร
	void Help()	ใช้แสดงหน้าต่างการใช้แป้นพิมพ์
	int EditDs(VarElem *pEdElem)	ใช้แก้ไขตัวแปร
	ObjElem *CheckVar(VarElem * pVarElem)	ใช้ตรวจสอบตัวแปร
	ObjElem *CheckDs(VarElem * pVarElem)	ใช้ตรวจสอบการใช้ตัวแปรซ้ำ
	virtual void SetReDefine()	ใช้กำหนดการอ้างอิงถึงตำแหน่งตัวแปร
	ObjElem *GetReDefine()	ใช้เลือกตัวแปรที่จะใช้อ้างถึง

16. **โมดูล DdWin** เป็นโมดูลที่ใช้กำหนดโครงสร้างแฟ้มข้อมูล ประกอบด้วย class DdWin ทำการสืบทอดคุณสมบัติมาจาก class DsWin มีรายละเอียดดังนี้

ตารางที่ 4.24 แสดงส่วนประกอบของโมดูล DdWin

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	DsElem *pKey	ใช้ระบุตำแหน่งของดัชนีของแฟ้มข้อมูล
	int nLimit	ใช้เก็บสถานะของการใช้คำสั่ง SETLL
	int nChain	ใช้เก็บสถานะของการใช้คำสั่ง CHAIN

ตารางที่ 4.24 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	int nAdd	ใช้เก็บสถานะของการใช้คำสั่งเพิ่มข้อมูล
	int nUpdate	ใช้เก็บสถานะของการใช้คำสั่งปรับปรุงข้อมูล
	int nDelete	ใช้เก็บสถานะของการใช้คำสั่งลบข้อมูล
โปรแกรม		
	DdWin()	ใช้กำหนดค่าเริ่มต้น
	~DdWin()	ใช้ยกเลิก
	void LoadData(ifstream &InFile)	ใช้นำข้อมูลเข้าสู่โปรแกรม
	void SaveData(ofstream &OutFile)	ใช้จัดเก็บข้อมูลของโครงสร้าง
	void SetLimit()	ใช้กำหนดสถานะของตัวแปร nLimit
	void SetChain()	ใช้กำหนดสถานะของตัวแปร nChain
	void SetAdd()	ใช้กำหนดสถานะของตัวแปร nAdd
	void SetUpdate()	ใช้กำหนดสถานะของตัวแปร nUpdate
	void SetDelete()	ใช้กำหนดสถานะของตัวแปร nDelete
	void CreateDd(ofstream &OutFile)	ใช้ในการสร้างแบบฟอร์ม I สำหรับแฟ้มข้อมูล
	void CreateDs(ofstream &)	ใช้ในการสร้างแบบฟอร์ม I
	void CreateFile(ofstream &OutFile)	ใช้ในการสร้างแบบฟอร์ม F
	void CreateOutput(ofstream & OutFile)	ใช้ในการสร้างแบบฟอร์ม O สำหรับแฟ้มข้อมูล
	void CreateWrite(ofstream & OutFile)	ใช้ในการสร้างแบบฟอร์ม O สำหรับการเพิ่มข้อมูล
	void CreateUpdate(ofstream & OutFile)	ใช้ในการสร้างแบบฟอร์ม O สำหรับการปรับปรุงข้อมูล
	void CreateDelete(ofstream & OutFile)	ใช้ในการสร้างแบบฟอร์ม O สำหรับการลบข้อมูล
	char *ObjectName()	ให้ชื่อ "Data Dictionary"
	char *Alias()	ให้ชื่อย่อ "DD"
	int ObjectNo()	ให้เลขประจำตัว
	void Prompt()	ใช้ในการเปิดหน้าต่างและแสดงข้อมูลอื่นๆ

ตารางที่ 4.24 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	void ShowOne(int nRow, int, ObjElem *pCurElem)	ใช้ในการแสดงข้อมูล 1 ตัว
	int HandleEvent()	ใช้ควบคุมการทำงานจากการกดแป้นพิมพ์
	void Modify()	ใช้แก้ไขเขตข้อมูล
	void Delete()	ใช้ลบเขตข้อมูล
	void SetReDefine()	ใช้ในการป้องกันการอ้างอิงเขตข้อมูล
	void Help()	ใช้แสดงหน้าต่างการใช้แป้นพิมพ์
	void SetKey()	ใช้เลือกดัชนีของแฟ้มข้อมูล
	int KeyLength()	ให้ค่าความยาวของดัชนี

17. **โมดูล WinScr** เป็นโมดูลที่ใช้ปรับวิธีแสดงผลสำหรับการแก้ไขข้อมูลบนจอภาพ ประกอบด้วย class WinScr สืบทอดคุณสมบัติมาจาก class Rpg มีรายละเอียดดังนี้

ตารางที่ 4.25 แสดงส่วนประกอบของโมดูล WinScr

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม		
	WinScr(int nURow, int nUCol, int nDRow, int nDCol, int nBorder, char *cTitle)	ใช้กำหนดค่าเริ่มต้น
	virtual ~WinScr()	ใช้ยกเลิก
	virtual char *ObjectName()	ให้ชื่อ "Screen"
	virtual char *Alias()	ให้ชื่อย่อ "SCR"
	virtual int ObjectNo()	ให้เลขประจำตัว
	virtual int Active(int)	ใช้กำหนดให้เริ่มทำงาน
	virtual void Draw()	ใช้แสดงผลบนจอภาพ
	virtual void ShowOne(int nRow, int nCol, ObjElem *pCurElem)	ใช้แสดงข้อมูล 1 ตัว
	virtual int HandleEvent()	ใช้ตรวจสอบข้อมูลที่รับเข้า
	virtual void Event()	ใช้ในการรับข้อมูล

18. **โมดูล WsWin** ใช้กำหนดโครงสร้างจอภาพ ประกอบด้วย

18.1 **Class WsWin** ใช้ในการแก้ไขข้อมูลบนจอภาพ ทำการสืบทอดคุณสมบัติมาจาก class WinScr มีรายละเอียดดังนี้

ตารางที่ 4.26 แสดงส่วนประกอบของ class WsWin

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	wintype *pWinStatus	ใช้ชี้ตำแหน่งของหน้าต่างสำหรับแสดงสถานะต่างๆ
	PgWin *pBeforeWs	ใช้ชี้ตำแหน่งของโปรแกรมที่ทำงานก่อนเข้าสู่วงรอบการแสดงผล
	PgWin *pAfterWs	ใช้ชี้ตำแหน่งของโปรแกรมที่ทำงานหลังออกจากวงรอบการแสดงผล
โปรแกรม		
	WsWin()	ใช้กำหนดค่าเริ่มต้น
	virtual ~WsWin()	ใช้ยกเลิก
	void LoadData(ifstream &InFile)	ใช้นำข้อมูลเข้าสู่โปรแกรม
	void SaveData(ofstream& OutFile)	ใช้จัดเก็บข้อมูลของโครงสร้าง
	virtual char *ObjectName()	ให้ชื่อ "Screen"
	char *Alias()	ให้ชื่อย่อ "WS"
	virtual int ObjectNo()	ให้เลขประจำตัว
	void Prompt()	ใช้ในการเปิดหน้าต่างและแสดงข้อมูล
	void UnPrompt()	ใช้ปิดการแสดงผล
	void DropVar()	ใช้ยกเลิกข้อมูลจอภาพที่ไม่มีตัวแปรอ้างอิง
	int HandleEvent()	ใช้ควบคุมการทำงานจากการกดแป้นพิมพ์
	void ShowOne(int nRow, int nCol, ObjElem *pCurElem)	ใช้ในการแสดงข้อมูล 1 ตัว
	void Insert()	ใช้เพิ่มตัวแปรบนจอภาพ
	void FnKey()	ใช้เพิ่มการใช้แป้นพิมพ์บนจอภาพ
	void Text()	ใช้เพิ่มข้อความบนจอภาพ

ตารางที่ 4.26 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	void Attrib()	ใช้กำหนดคุณสมบัติการแสดงผล
	void Edit()	ใช้แก้ไขข้อมูลบนจอภาพ
	void EditBefore()	ใช้ป้อนโปรแกรมก่อนแสดงผล
	void EditAfter()	ใช้ป้อนโปรแกรมหลังแสดงผล
	WsElem *Found()	ใช้ตรวจสอบตำแหน่งข้อมูลจอภาพ
	int GetMaxSize()	ให้ค่าความยาวจากตำแหน่งปัจจุบันถึงสิ้นสุดจอภาพ
	void Delete()	ใช้ลบข้อมูลจอภาพ
	void Move()	ใช้เคลื่อนย้ายข้อมูลจอภาพ
	void Help()	ใช้แสดงหน้าต่างการใช้แป้นพิมพ์
	void Status(char *cText = "")	ใช้แสดงข้อความที่หน้าต่างแสดงสถานะ
	void CreateDd(ofstream &OutFile)	ใช้ในการสร้างแบบฟอร์ม I สำหรับแฟ้มจอภาพ
	void CreateOutput(ofstream& OutFile)	ใช้ในการสร้างแบบฟอร์ม O สำหรับแฟ้มจอภาพ
	void CreateWs(ofstream& OutFile)	ใช้ในการสร้างแบบฟอร์ม C สำหรับการควบคุมการแสดงผลจอภาพ
	void CreateFmt(ofstream& OutFile)	ใช้ในการสร้างข้อกำหนดของจอภาพ
	int SyntaxCheck()	ใช้ตรวจสอบข้อผิดพลาดของโปรแกรมภาษาอาร์พีจี

2. ตัวแปรภายนอก ใช้ในการกำหนดค่าให้กับตัวแปรต่างๆ มีรายละเอียดดังนี้

ตารางที่ 4.27 แสดงตัวแปรภายนอกของโมดูล WsWin

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	extern int nWsLen	ใช้เก็บความยาวสูงสุดของตัวแปรที่ใช้แสดงผล

19. **โมดูล PgWin** เป็นโมดูลที่ใช้ในการจัดการโปรแกรมย่อย ประกอบด้วย class PgWin ทำการสืบทอดคุณสมบัติมาจาก class Rpg มีรายละเอียดดังนี้

ตารางที่ 4.28 แสดงส่วนประกอบของโมดูล PgWin

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	TxtElem *pTextLine	ใช้ชี้ตำแหน่งบรรทัดปัจจุบัน
	wintype *pWinTitle	ใช้ชี้ตำแหน่งหน้าต่างส่วนหัวของจอภาพ
	wintype *pWinStatus	ใช้ชี้ตำแหน่งหน้าต่างแสดงสถานะ
โปรแกรม		
	PgWin()	ใช้กำหนดค่าเริ่มต้น
	~PgWin()	ใช้ยกเลิก
	char *ObjectName()	ให้ชื่อ "Routine"
	char *Alias()	ให้ชื่อย่อ "PG"
	int ObjectNo()	ให้เลขประจำตัว
	void LoadData(ifstream &InFile)	ใช้นำข้อมูลเข้าสู่โปรแกรม
	void CreateCalc(ofstream& OutFile)	ใช้ในการสร้างแบบฟอร์ม C
	void CreateLine(ofstream& OutFile)	ใช้ในการสร้างแบบฟอร์ม C 1 บรรทัด
	void Convert(ofstream &OutFile,	ใช้ในการปรับคำสั่งภาษาอาร์พีจี
	char *cText, int nCommand)	
	int SyntaxCheck( )	ใช้ในการตรวจสอบข้อผิดพลาด
	int SyntaxProg(char *cMsg1="",	ใช้ในการตรวจสอบโปรแกรมภาษา
	char *cMsg2="")	อาร์พีจี
	void Prompt()	ใช้ในการเปิดหน้าต่างและแสดงข้อมูล
	void UnPrompt()	ใช้ปิดการแสดงผล
	int GetEvent()	ใช้รับแป้นพิมพ์
	int HandleEvent()	ใช้ควบคุมการทำงานจากการกดแป้น
		พิมพ์
	void Insert()	ใช้เพิ่มบรรทัด
	void Copy()	ใช้เก็บสำเนาข้อความ
	void Paste()	ใช้วางสำเนาข้อความ
	void Delete()	ใช้ลบบรรทัด
	void Help()	ใช้แสดงหน้าต่างการใช้แป้นพิมพ์
	void Status()	ใช้แสดงสถานะต่างๆ

20. **โมดูล ObjElem** ใช้เป็นหลักอ้างถึงวัตถุตัวอื่น ประกอบด้วย class ObjElem สืบทอดคุณสมบัติมาจาก class ListElem มีรายละเอียดดังนี้

ตารางที่ 4.29 แสดงส่วนประกอบของโมดูล ObjElem

ลักษณะ	คำสั่ง	คำอธิบาย
โปรแกรม		
	ObjElem()	ใช้กำหนดค่าเริ่มต้น
	virtual ~ObjElem()	ใช้ยกเลิก
	virtual char *ObjectName()	ให้ชื่อ "OBJ ELEMENT."
	virtual int ObjectNo()	ให้เลขประจำตัว
	virtual int isEqual(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	virtual int isEqual(char *)	ใช้เปรียบเทียบข้อความ
	virtual int isMore(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	virtual int isMore(char *)	ใช้เปรียบเทียบข้อความ
	virtual int Length()	ให้ความยาวของข้อมูล
	virtual char *GetText(char *cText)	ให้ข้อความของข้อมูล
	virtual int GetPos(int *, int *)	ให้ตำแหน่งของข้อมูล
	virtual void SaveData(ofstream& OutFile)	ใช้ในการเก็บข้อมูลลงแฟ้มข้อมูล

21. **โมดูล TaskElem** ใช้เชื่อมต่อระหว่างโครงสร้างชนิดต่างๆ ประกอบด้วย class TaskElem ซึ่งสืบทอดคุณสมบัติมาจาก class ObjElem มีรายละเอียดดังนี้

ตารางที่ 4.30 แสดงส่วนประกอบของโมดูล TaskElem

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	Rpg *pTaskList	ใช้ชี้ตำแหน่งของโครงสร้าง
โปรแกรม		
	TaskElem()	ใช้กำหนดค่าเริ่มต้น
	~TaskElem()	ใช้ยกเลิก
	virtual char *ObjectName()	ให้ชื่อ "Program header"
	virtual int ObjectNo()	ให้เลขประจำตัว

ตารางที่ 4.30 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	int isEqual(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	int isMore(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	char *GetText(char *cText)	ให้ข้อความของข้อมูล
	void SaveData(ofstream& OutFile)	ใช้ในการเก็บข้อมูลลงแฟ้มข้อมูล

22. **โมดูล DsElem** เป็นโมดูลที่ใช้ระบุถึงเขตข้อมูลหรือตัวแปร ประกอบด้วย class DsElem ซึ่งสืบทอดคุณสมบัติมาจาก class ObjElem มีรายละเอียดดังนี้

ตารางที่ 4.31 แสดงส่วนประกอบของโมดูล DsElem

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	VarElem *pVarElem	ใช้ชี้ตำแหน่งของตัวแปร
	int nRefCounter	ใช้นับจำนวนการอ้างอิงกับตัวแปรอื่นๆ
	DsElem *pRefElem	ใช้ชี้ตำแหน่งของข้อมูลที่ถูกอ้างถึง
โปรแกรม		
	DsElem()	ใช้กำหนดค่าเริ่มต้น
	DsElem(VarElem *pVarElem, int nRefCounter = 0, DsElem * pRefElem = NULL)	ใช้สำเนา
	DsElem(DsElem *pDsElem)	ใช้สำเนา
	~DsElem()	ใช้ยกเลิก
	virtual char *ObjectName()	ให้ชื่อ "DS ELEMENT."
	virtual int ObjectNo()	ให้เลขประจำตัว
	int isEqual(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	int isMore(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	virtual int Length()	ให้ความยาวของข้อมูล
	char *GetText(char *cText)	ให้ข้อความของข้อมูล
	void SaveData(ofstream& OutFile)	ใช้ในการเก็บข้อมูลลงแฟ้มข้อมูล
	int Create(ofstream& OutFile, int nPos)	ใช้ในการสร้างแบบฟอร์ม I
	int CreateOutput(ofstream& OutFile, int nPos)	ใช้ในการสร้างแบบฟอร์ม O

23. **โมดูล WsElem** เป็นโมดูลที่ใช้เก็บข้อมูลที่ใช้ในการแสดงผล ประกอบด้วย class WsElem ซึ่งสืบทอดคุณสมบัติมาจาก class ObjElem มีรายละเอียดดังนี้

ตารางที่ 4.32 แสดงส่วนประกอบของโมดูล WsElem

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	int nRow	ใช้เก็บแถวที่ใช้แสดงผล
	int nCol	ใช้เก็บคอลัมที่ใช้แสดงผล
	char cText[80]	ใช้เก็บข้อความที่ใช้แสดงบนจอภาพ
	char cReverse	ใช้เก็บสถานะการแสดงผลแบบผันกลับ
	char cIntensity	ใช้เก็บสถานะการแสดงผลแบบความสว่างสูง
	char cUnderline	ใช้เก็บสถานะการแสดงผลแบบขีดเส้นใต้
	char cNonDisplay	ใช้เก็บสถานะการแสดงผลแบบไม่แสดงผล
	char cBlink	ใช้เก็บสถานะการแสดงผลแบบกระพริบ
	char cColumnSep	ใช้เก็บสถานะการแสดงผลแบบแบ่งคอลัมน์
	PgWin *pBeforeWs	ใช้ชี้ตำแหน่งของโปรแกรมที่ทำงานก่อนการแสดงผล
	PgWin *pAfterWs	ใช้ชี้ตำแหน่งของโปรแกรมที่ทำงานหลังการแสดงผล
โปรแกรม		
	WsElem(int nRow, int nCol, char * cText = "", char cReverse = 0, char cIntensity = 0,char cUnderline = 0, char cNonDisplay = 0,char cBlink = 0, char cColumnSep = 0)	ใช้กำหนดค่าเริ่มต้น
	WsElem(WsElem *pWsElem)	ใช้ทำสำเนา
	virtual ~WsElem()	ใช้ยกเลิก
	virtual char *ObjectName()	ให้ชื่อ "Ws Text."
	virtual int ObjectNo()	ให้เลขประจำตัว

ตารางที่ 4.32 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	int isEqual(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	int isMore(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	virtual int Length()	ให้ความยาวของข้อมูล
	char *GetText(char *cText)	ให้ข้อความของข้อมูล
	virtual char *GetName()	ให้ตำแหน่งของข้อความ
	int GetPos(int *pRow, int *pCol)	ให้ตำแหน่งของข้อมูล
	int Row() { return nRow}	ให้ค่าของแถว
	int Col() { return nCol }	ให้ค่าของคอลัม
	void Hide(wintype *pWin)	ซ่อนข้อมูล(ไม่เสดงบนจอภาพ)
	virtual int Edit(int nMaxSize = 0)	แก้ไขข้อมูล
	virtual void EditBefore()	แก้ไขโปรแกรมก่อนแสดงผล
	virtual void EditAfter()	แก้ไขโปรแกรมหลังแสดงผล
	void Move(int nRow, int nCol)	เคลื่อนย้ายข้อมูล
	void Attrib()	กำหนดลักษณะในการแสดงผล
	char GetAttrib()	ให้ค่าของลักษณะในการแสดงผล
	virtual int MatchVar()	ใช้ทดสอบข้อความกับตารางตัวแปร
	void SaveData(ofstream& OutFile)	เก็บข้อมูลลงแฟ้มข้อมูล
	virtual int Load(char *cText)	ใช้นำข้อความเข้า
	virtual void SaveBefore(ofstream & OutFile)	เก็บโปรแกรมก่อนแสดงผลลงแฟ้มข้อมูล
	virtual void SaveAfter(ofstream & OutFile)	เก็บโปรแกรมหลังแสดงผลลงแฟ้มข้อมูล
	virtual void LoadBefore(char *cText)	ใช้นำข้อความเข้าสู่โปรแกรมก่อนแสดงผล
	virtual void LoadAfter(char *cText)	ใช้นำข้อความเข้าสู่โปรแกรมหลังแสดงผล
	virtual int CreateDs(ofstream &, int nPos)	ใช้สร้างแบบฟอร์ม I
	virtual int CreateOutput(ofstream &, int nPos)	ใช้สร้างแบบฟอร์ม O

ตารางที่ 4.32 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	virtual void CreateBefore(ofstream &OutFile)	ใช้สร้างโปรแกรมก่อนแสดงผล
	virtual void CreateAfter(ofstream & OutFile)	ใช้สร้างโปรแกรมหลังแสดงผล
	virtual void CreateFunction(ofstream &)	ใช้สร้างข้อกำหนดของจอภาพส่วนการใช้แฟ้นพิมพ์พิเศษ
	virtual void CreateFmt(ofstream & OutFile)	ใช้สร้างข้อกำหนดของจอภาพ

24. **โมดูล WvElem** เป็นโมดูลที่ใช้เก็บข้อมูลของตัวแปรที่ใช้ในการแสดงผล ประกอบด้วย class WvElem สืบทอดคุณสมบัติมาจาก class ObjElem มีรายละเอียดดังนี้

ตารางที่ 4.33 แสดงส่วนประกอบของโมดูล WvElem

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	VarElem *pVarElem	ใช้ชี้ตัวแปรในตาราง
	char cName[7]	ใช้ระบุชื่อตัวแปร
	int nInput	ใช้ระบุให้เป็นข้อมูลรับเข้า
	int nMsg	ใช้ระบุให้เป็นข้อมูลเสดงข้อผิดพลาด
โปรแกรม		
	WvElem(int nRow,int nCol)	ใช้กำหนดค่าเริ่มต้น
	~WvElem() {} -	ใช้ยกเลิก
	char *ObjectName()	ให้ชื่อ "Ws Variable."
	int ObjectNo()	ให้เลขประจำตัว
	int Length()	ให้ความยาวของข้อมูล
	char *GetText(char *cText)	ให้ข้อความของข้อมูล
	virtual char *GetName()	ให้ตำแหน่งของชื่อตัวแปร
	int MatchVar()	ใช้ทดสอบข้อความกับตารางตัวแปร
	int Edit(int nMaxSize)	ใช้แก้ไขตัวแปร

ตารางที่ 4.33 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	int EditIO() void EditBefore() void EditAfter() void SaveData(ofstream &OutFile) int Load(char *cText) int CreateDs(ofstream& OutFile,int nPos)	ใช้ระบุการรับข้อมูลเข้า แก้ไขโปรแกรมก่อนแสดงผล แก้ไขโปรแกรมหลังแสดงผล เก็บข้อมูลลงแฟ้มข้อมูล ใช้นำข้อความเข้า ใช้สร้างแบบฟอร์ม I
	int CreateOutput(ofstream& OutFile,int nPos) void CreateFmt(ofstream &OutFile)	ใช้สร้างแบบฟอร์ม O ใช้สร้างข้อกำหนดของจอภาพ

25. **โมดูล FnElem** เป็นโมดูลที่ใช้เก็บข้อมูลของแป้นพิมพ์ ประกอบด้วย class FnElem สืบทอดคุณสมบัติมาจาก class ObjElem มีรายละเอียดดังนี้

ตารางที่ 4.34 แสดงส่วนประกอบของโมดูล FnElem

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	int nFnKey int nValidate int nExit char cRoutine[7]	ใช้เก็บค่าของแป้นพิมพ์ ใช้ระบุสถานะการตรวจสอบ ใช้ระบุสถานะสิ้นสุดการแสดงผล ใช้ระบุชื่อโปรแกรมย่อยหรือจอภาพที่จะเรียกใช้
โปรแกรม	FnElem(int nRow, int nCol) FnElem(int nRow, int nCol, int nFnKey, int nValidate = 0, int nExit = 0, char *cRoutine= "") ~FnElem() {} char *ObjectName()	ใช้กำหนดค่าเริ่มต้น ใช้สำเนา ใช้ยกเลิก ให้ชื่อ "FUNCTION KEY."

ตารางที่ 4.34 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	int ObjectNo()	ให้เลขประจำตัว
	int isEqual(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	int EditKey()	ใช้สร้างแป้นพิมพ์
	int Edit(int nMaxSize)	ใช้แก้ไขแป้นพิมพ์
	void EditBefore()	ใช้ยกเลิกการแก้ไขโปรแกรมก่อนแสดงผล
	void EditAfter()	ใช้ยกเลิกการแก้ไขโปรแกรมหลังแสดงผล
	void SaveData(ofstream &OutFile)	เก็บข้อมูลลงแฟ้มข้อมูล
	int Load(char *cText)	ใช้นำข้อความเข้า
	char *GetKey(char *cKey)	ให้ค่าแป้นพิมพ์ของเครื่องไอบีเอ็ม S/36
	int IsTerminate()	ให้ค่าสถานะสิ้นสุดการแสดงผล
	int IsValidate()	ให้ค่าสถานะที่ต้องการการตรวจสอบข้อผิดพลาด
	char *GetRoutine()	ให้ชื่อโปรแกรมย่อยหรือจอภาพ

26. **โมดูล TxtElem** เป็นโมดูลที่ใช้เก็บข้อความคงที่ ประกอบด้วย class TxtElem ซึ่งสืบทอดคุณสมบัติมาจาก class ObjElem มีรายละเอียดดังนี้

ตารางที่ 4.35 แสดงส่วนประกอบของโมดูล TxtElem

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล		
	int nRow	ใช้เก็บค่าแถว
	int nCol	ใช้เก็บค่าคอลัมน์
	char cString[81]	ใช้เก็บข้อความ
	ObjElem *pTextObj	ใช้เก็บตำแหน่งของข้อมูลที่ใช้ในการอ้างอิง
	int nInt	ใช้เก็บตัวเลขที่ใช้ในการอ้างอิง

ตารางที่ 4.35 (ต่อ)

ลักษณะโปรแกรม	คำสั่ง	คำอธิบาย
	TxtElem()	ใช้กำหนดค่าเริ่มต้น
	TxtElem(char *cString, ObjElem * pOther = NULL)	ใช้กำหนดค่าเริ่มต้น
	TxtElem(char *cString, int pOther)	ใช้กำหนดค่าเริ่มต้น
	TxtElem(int nRow, int nCol, char * cString, ObjElem *pOther= NULL)	ใช้กำหนดค่าเริ่มต้น
	~TxtElem() {}	ใช้ยกเลิก
	char *ObjectName()	ให้ชื่อ "TEXT ELEMENT."
	int ObjectNo()	ให้เลขประจำตัว
	int isEqual(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	int isEqual(char *)	ใช้เปรียบเทียบข้อความ
	int isMore(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	int isMore(char *)	ใช้เปรียบเทียบข้อความ
	int GetLength()	ให้ความยาวของข้อมูล
	char *NewText(char *cText)	ใช้กำหนดข้อความใหม่
	char *GetText(char *cText)	ให้ข้อความของข้อมูล
	ObjElem *GetPointer()	ให้ตำแหน่งข้อมูลที่ใช้อ้างอิง
	void SetInt(int nInt)	กำหนดค่าตัวเลขที่ใช้อ้างอิง
	int GetInt()	ให้ค่าตัวเลขที่ใช้อ้างอิง
	int GetPos(int *pRow, int *pCol)	ให้ตำแหน่งของข้อมูล
	void SaveData(ofstream& OutFile)	ใช้ในการเก็บข้อมูลลงแฟ้มข้อมูล

27. **โมดูล VarElem** เป็นโมดูลที่ใช้เก็บตัวแปร ประกอบด้วย class VarElem ซึ่งสืบทอดคุณสมบัติมาจาก class ObjElem มีรายละเอียดดังนี้

ตารางที่ 4.36 แสดงส่วนประกอบของโมดูล VarElem

ลักษณะ	คำสั่ง	คำอธิบาย
ข้อมูล	char cName[7]	ใช้เก็บชื่อตัวแปร

ตารางที่ 4.36 (ต่อ)

ลักษณะ	คำสั่ง	คำอธิบาย
	int nType	ใช้เก็บชนิด
	int nLength	ใช้เก็บความยาว
	int nDecimal	ใช้เก็บจำนวนทศนิยม
	char cInfo[16]	ใช้เก็บคำอธิบาย
	int nUsedTime	ใช้เก็บจำนวนครั้งที่ใช้อ้างอิง
โปรแกรม		
	VarElem()	ใช้หนดค่าเริ่มต้น
	VarElem(ObjElem *Other)	ใช้ทำสำเนา
	VarElem(char *cName, int nType, int nLenght, int nDecimal, char * cInfo)	ใช้ทำสำเนา
	void Load(char *cText)	ใช้แปลงข้อความเป็นตัวแปร
	char *ObjectName()	ให้ชื่อ "VAR ELEMENT."
	int ObjectNo()	ให้เลขประจำตัว
	int isEqual(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	int isEqual(char *)	ใช้เปรียบเทียบข้อความ
	int isMore(ObjElem *other)	ใช้เปรียบเทียบข้อมูล
	int isMore(char *)	ใช้เปรียบเทียบข้อความ
	char *GetText(char *cText)	ให้ข้อความของข้อมูล
	void Clear()	ใช้ลบข้อมูลตัวแปร
	void Copy(ObjElem *Other)	กำหนดข้อมูลของตัวแปรใหม่
	int Inc()	เพิ่มจำนวนครั้งที่ใช้อ้างอิง
	int Dec()	ลดจำนวนครั้งที่ใช้อ้างอิง
	char *GetType()	ให้ชื่อชนิดของตัวแปร
	char *Name()	ให้ชื่อของตัวแปร
	int Type()	ให้ค่าชนิดของตัวแปร
	int Length()	ให้ค่าความยาวของตัวแปร
	int Decimal()	ให้ค่าจำนวนทศนิยมของตัวแปร
	char *Info()	ให้คำอธิบายของตัวแปร
	int SelectType()	เลือกชนิดของตัวแปร