บทที่ 5

การพัฒนาและการทดสอบตัวควบคุม

<u>เครื่องมือที่ใช้ในการทำวิจัย</u>

1. ฮาร์ดแวร์

1.1) เครื่องคอมพิวเตอร์ IBM PC PC/AT หรือเครื่องที่ compatible ทั่วไป ที่มีหน่วยความจำอย่างน้อย 1 เมกะไบต์

1.2) ฮาร์ดดิสก์ ที่มีเนื้อที่ในฮาร์ดดิสก์ไม่น้อยกว่า 6 เมกะไบต์

1.3) เมาส์

2. ซอฟต์แวร์

2.1) ระบบ Microsoft Window Thai version 3.1

2.2) Microsoft Window Software Development Kit (SDK) สำหรับ Microsoft Window Thai version 3.1 ขึ้นไป

2.3) Microsoft Visual Basic Control Development Kit (CDK) สำหรับ Microsoft Window Thai version 3.1 ขึ้นไป

2.4) Microsoft Visual C/C++ Compiler version 1.0

2.5) Visual Basic for Window version 3.0

<u>แฟ้มข้อมูลที่เป็นส่วนประกอบในการพัฒนาเป็นตัวควบคุมแผ่นตารางทำการ</u>

1. แฟ้มสกุล .H

ในที่นี้คือ แฟ้ม SHEET.H ซึ่งจะกำหนด

1.1) ค่าคงที่และรูปแบบที่ใช้ในการติดต่อกับฟังก์ชันต่าง ๆ ภายในตัวควบคุม

1.2) ตัวแบบของตัวควบคุม (Control Model)

1.3) ตารางข้อมูลคุณสมบัติ (Property Information Table)

1.4) ตารางข้อมูลเหตุการณ์ (Event Informatiion Table)

2. แฟ้มสกุล .C

ในที่นี้หมายถึงแฟ้ม SHEET.C ซึ่งจะกำหนดกระบวนงานของตัวควบคุม (Control Procedure) และชุดคำสั่งของฟังก์ชันภายในตัวควบคุม

3. แฟ้มสกุล .BMP

เป็นแฟ้มรูปภาพบิตแมพที่เป็นภาพสัญรูปของตัวควบคุมในทูลบอกซ์ โดยใช้โปรแกรมสร้างภาพที่ชื่อ App Studio ของ Microsoft Visual C/C++ ในการพัฒนา ซึ่งแฟ้มรูปภาพบิตแมพจะมีทั้งหมด 4 แฟ้มคือ

3.1) SHEETVD.BMP

สำหรับจอภาพวีจีเอเมื่อมีการคลิกเมาส์ที่สัญรูปของตัวควบคุมแผ่นตารางทำการในทูลบอกซ์

3.2) SHEETVU.BMP

สำหรับจอภาพวีจีเอเมื่อไม่มีการคลิกเมาส์ที่สัญรูปของตัวควบคุมแผ่นตารางทำการในทูลบอกซ์

3.3) SHEETEU.BMP

สำหรับจอภาพอีจีเอ

3.4) SHEETMU.BMP

สำหรับจอภาพสีเดียว

4. แฟ้มสกุล .RC

ในที่นี้คือแฟ้ม SHEET.RC โดยเป็นแฟ้มที่เก็บรูปแบบของทรัพยากร (Resource) ต่างๆ ที่ใช้ในการสร้างโปรแกรมตัวควบคุม ซึ่งก็คือ รูปภาพบิตแมพ

5. แฟ้มสกุล .DEF

ในที่นี้คือแฟ้ม SHEET.DEF ซึ่งแฟ้มนี้จะกำหนดลักษณะต่าง ๆ ของโปรแกรม เช่น ขนาดของฮีป และขนาดของกองซ้อน (Stack)

6. แฟ้มสกุล .MAK

ในที่นี้คือ แฟ้ม SHEET.MAK ซึ่งแฟ้มนี้จะช่วยโปรแกรม NMAKE ในการเรียกใช้งานตัวแปลโปรแกรม โปรแกรมเชื่อมโยงและตัวแปลโปรแกรมทรัพยากร ตลอดจนตรวจสอบและดำเนินต่าง ๆ เพื่อสร้างโปรแกรมประยุกต์

7. แฟ้ม VBAPI.H

เป็นแฟ้มที่กำหนดรูปแบบในการติดต่อเรียกใช้ฟังก์ชันทั้งหมด ตลอดจนโครงสร้างและค่าคงที่ต่าง ๆ ในแฟ้ม VBAPI.LIB

8. แฟ้ม VBAPI.LIB

เป็นแฟ้มของคลังชุดคำสั่งที่มีฟังก์ชันทั้งหมดที่เป็น API ของวิชวลเบสิก โปรแกรมเชื่อมโยงของภาษาโปรแกรมที่ใช้สร้างตัวควบคุม จะต้องเชื่อมโยงคลังชุดคำสั่งนี้เข้ากับแฟ้ม SHEET.VBX ด้วย

ขั้นตอนการพัฒนาตัวควบคุม

หลังจากที่ได้ออกแบบคุณสมบัติ เหตุการณ์ และรูปร่างของตัวควบคุมแล้ว จะต้องทำดังนี้

1. กำหนดตัวแบบของตัวควบคุมในแฟ้มข้อมูลสกุล .H
2. กำหนดคุณสมบัติของตัวควบคุมในตารางข้อมูลคุณสมบัติในแฟ้มข้อมูลสกุล .H
3. กำหนดเหตุการณ์ของตัวควบคุมในตารางข้อมูลเหตุการณ์ในแฟ้มข้อมูลสกุล .H
4. กำหนดกระบวนงานของตัวควบคุมและชุดคำสั่งของฟังก์ชันภายในตัวควบคุมในแฟ้มข้อมูลสกุล .C
5. สร้างสัญรูปของตัวควบคุมด้วยโปรแกรมสร้างภาพ

การกำหนดตัวแบบของตัวควบคุม

ก่อนที่จะทำการกำหนดตัวแบบของตัวควบคุมนั้น ควรจะทราบถึงโครงสร้างข้อมูลของตัวแบบ (SHEET.H บรรทัดที่ 167-184) ซึ่งโครงสร้างข้อมูลของตัวแบบประกอบด้วย

1. usVersion

เป็นเลขรุ่นของวิชวลเบสิกที่ใช้พัฒนาตัวควบคุม (SHEET.H บรรทัดที่ 169) ในที่นี้คือค่า VB_VERSION ตามที่วิชวลเบสิกกำหนดไว้ใน VBAPI.H

2. fl

เป็นตัวบ่งชี้ของตัวแบบ (Model flag) (SHEET.H บรรทัดที่ 170) ซึ่งมีให้เลือกใช้ดังตารางที่ 5.1

ตารางที่ 5.1 แสดงตัวบ่งชี้ของด้นแบบ

ตัวบ่งชี้	ความหมาย
MODEL_fArrows	ให้ส่งข้อความคีย์บอร์ดเมื่อมีการกดปุ่มลูกศรต่าง ๆ ถ้าไม่ได้ใช้ตัวบ่งชี้นี้ การให้ปุ่มลูกศรจะเป็นการเปลี่ยนโฟกัสระหว่างตัวควบคุม
MODEL_fChildrenOk	ให้ผู้พัฒนาโปรแกรมประยุกต์ด้วยวิชวลเบสิก สามารถสร้างตัวควบคุมอื่นขึ้นภายในตัวควบคุมที่ใช้ตัวบ่งชี้นี้ได้ เช่น รูปภาพต่าง ๆ
MODEL_fDesInteract	ส่งข้อความที่เป็นของปุ่มเมาส์ด้านขวาได้แก่ WM_RBUTTONDOWN WM_RBUTTONDBLCLK และ WM_RBUTTONUP ไปยังตัวควบคุมในช่วงเวลาการออกแบบ
MODEL_fFocusOk	สามารถกำหนดโฟกัสให้กับตัวควบคุมในช่วงเวลาดำเนินงานได้
MODEL_fGraphical	กำหนดให้เป็นตัวควบคุมทางด้านรูปภาพ
MODEL_fInitMsg	ให้วิชวลเบสิกส่งข้อความ VBM_INITIALIZE ไปยังตัวควบคุม
MODEL_fInvisAtRun	กำหนดให้ไม่แสดงตัวควบคุมในช่วงเวลาดำเนินงาน
MODEL_fLoadMsg	ให้ส่งข้อความ VBM_CREATED และ VBM_LOADED ไปยังตัว ควบคุม
MODEL_fMnemonic	ให้ตอบสนองต่อปุ่มสัญลักษณ์ช่วยจำ (Mnemonic Key) ของตัวควบคุม โดยกำหนดโฟกัสให้กับตัวควบคุม

3. pctlproc

เป็นตำแหน่งที่อยู่กระบวนงานของตัวควบคุมในหน่วยความจำ (SHEET.H บรรทัดที่ 171)

4. fsClassStyle

หมายถึง ลักษณะคลาสของวินโดวส์ (Window Class Style) ของตัวควบคุม โดยที่ลักษณะคลาสเหล่านี้จะนำหน้าด้วย CS ซึ่งอยู่ในแฟ้ม WINDOW.H การกำหนดลักษณะคลาสของวินโดวส์สามารถกำหนดได้ครั้งละมากกว่าหนึ่งลักษณะ (SHEET.H บรรทัดที่ 172) ลักษณะคลาสของวินโดวส์มีดังตารางที่ 5.2

ตาราง 5.2 แสดงลักษณะคลาสของวินโดวส์

ลักษณะคลาสของวินโดวส์	ความหมาย
CS_BYTEALIGNCLIENT	จัดวางพื้นที่ใช้งานตามไบต์ (แนวนอน)
CS_BYTEALIGNWINDOW	จัดวางวินโดวส์ตามไบต์ (แนวนอน)
CS_CLASSDC	จองคอนเท็กซ์ดิสเพลย์เพียงหนึ่งสำหรับทุก วินโดวส์ที่ใช้คลาสนี้
CS_DBCLKS	ส่งเมสเสจดับเบิ้ลคลิกไปยังฟังก์ชันประจำวินโดวส์
CS_GLOBALCLASS	กำหนดว่าคลาสนั้นสามารถถูกใช้ได้กับทุก โปรแกรมประยุกต์
CS_HREDRAW	กำหนดว่าจะต้องมีการวาดวินโดวส์ใหม่เมื่อมีการย้ายหรือเปลี่ยนแปลงขนาดที่มีผลต่อความกว้างของวินโดวส์
CS_NOCLOSE	ไม่ให้ใช้คำสั่ง Close ในเมนูของระบบ
CS_OWNDC	จองคอนเท็กซ์ดิสเพลย์สำหรับทุกวินโดวส์ที่ใช้ คลาสนั้น
CS_PARENTDC	ให้คลาสของวินโดวส์นั้นใช้คอนเท็กซ์ดิสเพลย์ของวินโดวส์แม่ของวินโดวส์นั้น ๆ
CS_SAVEBITS	กำหนดให้มีการเก็บภาพส่วนของวินโดวส์ที่ถูกบัง เพื่อใช้ในการวาดใหม่เมื่อวินโดวส์ถูกย้ายการทำเช่นนี้จะไม่มีการส่ง WM_PAINT ไปยังวินโดวส์ (ถ้าหน่วยความจำที่เก็บรูปภาพส่วนนั้นไม่ถูกทิ้ง)
CS_VREDRAW	กำหนดว่าจะต้องมีการวาดวินโดวส์ใหม่เมื่อมีการย้ายหรือเปลี่ยนแปลงขนาดที่มีผลต่อความสูงของวินโดวส์

5. fsWndStyle

หมายถึง ลักษณะของวินโดวส์โดยปริยายของตัวควบคุม (SHEET.H บรรทัดที่ 173) เช่น WS_HSCROLL จะหมายถึงวินโดวส์จะมีแถบเลื่อนแนวนอนเป็นลักษณะโดยปริยาย ซึ่งรายละเอียดสามารถศึกษาได้จาก คู่มือ Microsoft Windows Software Development Kit

6. cbCtlExtra

เป็นขนาดของโครงสร้างของผู้เขียนโปรแกรม (SHEET.H บรรทัดที่ 174) กรณีที่ไม่มีโครงสร้างนี้ให้กำหนดค่าเป็น 0

7. idBmpPalette

เป็นหมายเลขเริ่มต้นของรูปภาพบิตแมพที่ใช้เป็นสัญรูปแทนตัวควบคุมทูลบอกซ์ (SHEET.H บรรทัดที่ 175) ซึ่งมีรายละเอียดดังตารางที่ 5.3

ตารางที่ 5.3 แสดงความหมายของ idBmpPalette

หมายเลขของรูปภาพบิตแมพ	ความหมาย
idBmpPalette	สำหรับจอภาพวีจีเอเมื่อมีการคลิกเมาส์ที่สัญรูปในทูลบอกซ์
idBmpPalette +1	สำหรับจอภาพวีจีเอเมื่อไม่มีการคลิกเมาส์ที่สัญรูปในทูลบอกซ์
idBmpPalette +3	สำหรับจอภาพสีเดียว
idBmpPalette +6	สำหรับจอภาพอีจีเอ

8. npszDefCtlName

เป็นชื่อโดยปริยายของตัวควบคุม (SHEET.H บรรทัดที่ 176)

9. npszClassName

เป็นชื่อของคลาสที่จะใช้ในวิชวลเบสิก (SHEET.H บรรทัดที่ 177)

10. npszParentClassName

เป็นชื่อของคลาสที่ให้กำเนิด (Window Parent Class) ของตัวควบคุม (SHEET.H บรรทัดที่ 178)

11. npproplist

เป็นตัวชี้ขนาด 16 บิตไปยังตารางข้อมูลคุณสมบัติ (SHEET.H บรรทัดที่ 179)

12. npeventlist

เป็นตัวชี้ขนาด 16 บิต ไปยังตารางข้อมูลเหตุการณ์ (SHEET.H บรรทัดที่ 180)

13. nDefProp

เป็นดัชนีของคุณสมบัติที่จะใช้เป็นคุณสมบัติโดยปริยายในวินโดวส์คุณสมบัติ (SHEET.H บรรทัดที่ 181)

14. nDefEvent

เป็นดัชนีของเหตุการณ์ที่จะใช้เหตุการณ์โดยปริยายในวินโดวส์สำหรับเขียนโปรแกรม (SHEET.H บรรทัดที่ 182)

15. nValueProp

เป็นดัชนีของคุณสมบัติที่จะเป็นคุณสมบัติโดยปริยายของตัวควบคุม (SHEET.H บรรทัดที่ 183)

จากส่วนประกอบโครงสร้างของตัวแบบข้างต้น จึงได้ทำการกำหนดตัวแบบของตัวควบคุมแผ่นตารางทำการดังต่อไปนี้

```
MODEL  modelSheet =
    {
    VB_VERSION,                          // VB version being used
    MODEL_fFocusOk | MODEL_fLoadMsg,     // MODEL flag
    (PCTLPROC) SheetCtlProc,             // Control procedure
    CS_VREDRAW | CS_HREDRAW,             // Class style
    WS_BORDER   |   WS_HSCROLL   |       // Default Windows style
    WS-VSCROLL,
    sizeof (SheetStru) ,                 // Size of STD structure
    IDBMP_SHEET,                         // Palette bitmap ID
    "Sheet",                             // Default control name
    "SHEET",                             // Visual Basic class name
    NULL,                                // Parent class name
    Sheet_Properties,                    // Property information table
    Sheet_Events,                        // Event information table
    IPROPINFO_STD_CLICK,                 // Default property
    IEVENTINFO_STD_BACKCOLOR,            // Default event
    NULL                                 // Property representingvalue of ctl
    };
```

<u>การกำหนดคุณสมบัติให้กับตัวควบคุม</u>

ก่อนที่จะทำการกำหนดคุณสมบัติให้กับตัวควบคุมนั้น ควรจะทราบถึงโครงสร้างของตารางข้อมูลคุณสมบัติ ซึ่งมีส่วนประกอบดังนี้

1. npszName

เป็นชื่อของคุณสมบัติที่จะใช้วินโดวส์คุณสมบัติและในการเขียนโปรแกรม (SHEET.H บรรทัดที่ 71)

2. fl

เป็นตัวบ่งชี้ของคุณสมบัติ โดยตัวแรกใช้กำหนดชนิดของคุณสมบัติ (SHEET.H บรรทัดที่ 72) ซึ่งมีรายละเอียดดังตารางที่ 5.4

ตารางที่ 5.4 แสดงตัวบ่งชี้กำหนดชนิดของคุณสมบัติ

ตัวบ่งชี้	ความหมาย
DT_BOOL	กำหนดชนิดเป็นตรรกศาสตร์ (BOOLEAN) ค่าที่ไม่เท่ากับ 0 (จริง) จะถูกเปลี่ยนเป็น -1 เสมอสำหรับคุณสมบัติชนิดนี้
DT_COLOR	เป็นจำนวนเต็มชนิดยาว (Long Integer) ที่แทนสีแต่ละสีที่ใช้ในวิชวลเบสิก โดยในวินโดวส์คุณสมบัติจะแสดงค่าเป็นเลขฐาน 16
DT_ENUM	เป็นจำนวนเต็มชนิดสั้น (Short Integer) แบบไม่คิดเครื่องหมาย ซึ่งมีค่าได้ตั้งแต่ 0 - 255
DT_HSZ	เป็นแฮนเดิลของอักขระที่ปิดท้ายด้วย " \ 0 " ซึ่งคล้ายกับอักขระ (String) ในภาษาซี
DT_HLSTR	เป็นแฮนเดิลของอักขระในภาษาวิชวลเบสิก ซึ่งสามารถมี " \ 0 " อยู่ในข้อความนี้ได้ (ไม่เหมือนกับ DT_HSZ) และ DT_HLSTR จะใช้เป็นอาร์กิวเมนต์ในกระบวนงานเหตุการณ์ได้
DT_LONG	จำนวนเต็มขนาด 32 บิต แบบคิดเครื่องหมาย
DT_PICTURE	เป็นโครงสร้างแบบรูปภาพ (PICTURE STRUCTURE) ที่ใช้แฮนเดิลในการอ้างถึงรูปภาพแต่ละรูป (HPIC)
DT_REAL	เป็นจำนวนจริงขนาด 4 ไบต์
DT_SHORT	จำนวนเต็ม 16 บิต แบบคิดเครื่องหมาย

ตัวบ่งชี้ที่ถัดจากตัวบ่งชี้ชนิดของคุณสมบัติ (SHEET.H บรรทัดที่ 72) ใช้กำหนดว่า จะให้วิชวลเบสิกติดต่อกับคุณสมบัติในโครงสร้างของผู้เขียนโปรแกรมอย่างไร ได้แก่ PF_fGetMsg PF_fGetData PF_fSetMsg PF_fSetData PF_fSaveMsg PF_fSaveData

สมมติกำหนดตัวบ่งชี้ดังนี้

PF_fGetMsg | PF_SetMsg | PF_fSaveMsg

แสดงว่าจะเป็นการกำหนดให้วิชวลเบสิกส่งข้อความ VBM_GETPROPERTY เมื่อต้องการอ่านค่าคุณสมบัติ ส่งข้อความ VBM_SETPROPERTY เมื่อมีการกำหนดค่าคุณสมบัติ ส่งข้อความ VBM_SAVEPROPERTY และ VBM_SETPROPERTY เมื่อต้องการบันทึกค่าของคุณสมบัติลงจานบันทึก หรือนำค่าของคุณสมบัติเข้าสู่หน่วยความจำ ข้อความทั้ง 4 อาจเกิดจากการที่ผู้พัฒนาโปรแกรมด้วย วิชวลเบสิกเขียนชุดคำสั่งที่ต้องการอ่านค่าคุณสมบัติ เปลี่ยนแปลงค่าคุณสมบัติ นำฟอร์มเข้าสู่หน่วยความจำ (ด้วยชุดคำสั่ง Load) หรือเปลี่ยนแปลงค่าคุณสมบัติจากในวินโดวส์คุณสมบัติ

ถ้ากำหนดตัวบ่งชี้เป็น

PF_fGetData | PF_SetData | PF_fSaveData

แสดงว่าจะกำหนดให้วิชวลเบสิกทำการโอนถ่ายข้อมูลกับโครงสร้างของผู้เขียนโปรแกรมโดยตรง ไม่ต้องส่งข้อความเมื่อต้องการอ่านค่าคุณสมบัติ (GetData) กำหนดค่าคุณสมบัติ (SetData) และบันทึก (SaveData) หรือนำคุณสมบัติเข้าสู่หน่วยความจำ

ถ้ากำหนดตัวบ่งชี้เป็น

PF_fGetData | PF_SetMsg | PF_fSaveData

แสดงว่าจะกำหนดให้วิชวลเบสิกส่งข้อความ VBM_SETPROPERTY เมื่อมีการกำหนดค่าคุณสมบัติ แต่ให้โอนถ่ายข้อมูลกับโครงสร้างของผู้เขียนโปรแกรมโดยตรงเมื่อต้องการอ่านค่าคุณสมบัติ (GetData) กับ บันทึก (SaveData) หรือนำคุณสมบัติเข้าสู่หน่วยความจำ

นอกจากนี้ยังมีตัวบ่งชี้อื่น ๆ อีกดังตารางที่ 5.5

3. offsetData

หมายถึงตำแหน่งที่ใช้เก็บค่าของคุณสมบัติในโครงสร้างของผู้เขียนโปรแกรม (SHEET.H บรรทัดที่ 86)

4. infodata

กรณีที่ต้องการเก็บค่าของคุณสมบัติในลักษณะของบิต เพื่อประหยัดเนื้อที่ในโครงสร้างของผู้เขียนโปรแกรม จะต้องกำหนดค่า infodata โดยต้องใช้ควบคู่กับ DT_BOOL DT_SHORT และ DT_ENUM

ตาราง 5.5 แสดงตัวบ่งชี้อื่น ๆ ของคุณสมบัติ

ตัวบ่งชี้	ความหมาย
PF_fDefval	ให้วิชวลเบสิกไม่ต้องบันทึกค่าของคุณสมบัติลงสู่จานบันทึก ถ้าค่าของคุณสมบัตินั้นเท่ากับฟิลด์ (Field) dataDefault ของโครงสร้าง PROPINFO ดังนั้นเมื่อมีการนำตัวควบคุมเข้าสู่หน่วยความจำ ถ้าค่าของคุณสมบัติใดไม่ได้ถูกบันทึกไว้ วิชวลเบสิกก็จะกำหนดค่าคุณสมบัตินั้นตามฟิลด์ dataDefault
PF_fEditable	ให้ผู้พัฒนาโปรแกรมด้วยวิชวลเบสิก สามารถแก้ไขค่าของคุณสมบัติในกล่องกำหนดค่าได้โดยตรง ถึงแม้ว่าจะมีการใช้รายการเลือก หรือวินโดว์แบบผุดขึ้น (POP-UP WINDOW) ก็ตาม
PF_fGetHszMsg	ให้วิชวลเบสิกส่งข้อความ VBM_GETPROPERTYHSZ ไปยังตัวควบคุมเมื่อต้องการจะแสดงค่าของคุณสมบัติในวินโดว์คุณสมบัติ
PF_fLoadMsgOnly	ให้วิชวลเบสิกส่งข้อความ VBM_LOADPROPERTY เมื่อมีการนำฟอร์มจากแฟ้มเข้าสู่หน่วยความจำ ซึ่งข้อความนี้จะมีแฮนเดิลของตำแหน่งของแฟ้มนั้นด้วย ซึ่งกระบวนงานของตัวควบคุมสามารถใช้แฮนเดิลนี้ ในการนำค่าของคุณสมบัติเข้าสู่หน่วยความจำ โดยใช้ API ของวิชวลเบสิก คือ ฟังก์ชั่น VBReadFormFile
PF_fLoadDataOnly	ให้วิชวลเบสิกนำค่าของคุณสมบัติจากฟอร์มที่บันทึกไว้เข้าสู่หน่วยความจำ และไม่ต้องบันทึกค่าของคุณสมบัติลงจานบันทึกอีก ตัวบ่งชี้นี้ช่วยให้ตัวควบคุมสามารถเข้ากันได้กับวิชวลเบสิกรุ่น 1.0
PF_fNoInitDef	ไม่ให้วิชวลเบสิกกำหนดค่าของคุณสมบัติเท่ากับค่าของฟิลด์ dataDefault ในระหว่างที่มีการนำตัวควบคุมเข้าสู่หน่วยความจำ แต่ถ้าตัวบ่งชี้ PF_fDefVal ไม่ได้ถูกกำหนดขึ้น ตัวบ่งชี้ PF_fNoInitDef จะไม่มีผลใด ๆ
PF_fNoShow	จะไม่แสดงคุณสมบัตินั้นในวินโดวส์คุณสมบัติ
PF_fPropArray	กำหนดให้เป็นคุณสมบัติแบบช่องลำดับ และควรใช้ตัวบ่งชี้ PF_fNoShow ร่วมด้วย
PF_fNoRuntimeW	กำหนดให้คุณสมบัตินั้น ถูกอ่านค่าได้อย่างเดียว แก้ไขเปลี่ยนแปลงค่าไม่ได้ในช่วงเวลาดำเนินงาน

ตาราง 5.5 แสดงตัวบ่งชี้อื่น ๆ ของคุณสมบัติ (ต่อ)

ตัวบ่งชี้	ความหมาย
PF_fNoRuntimeR	กำหนดให้ผู้พัฒนาโปรแกรมด้วยวิชวลเบสิกไม่สามารถเขียนโปรแกรม เพื่ออ่านค่าคุณสมบัตินี้ได้ในช่วงเวลาดำเนินงาน แต่สามารถเปลี่ยนแปลงค่าได้ หากยังพยายามอ่านค่าของคุณสมบัตินี้ ก็จะเกิดข้อผิดพลาดในช่วงเวลาดำเนินงาน (Run Time Error) ขึ้น และถ้าใช้ตัวบ่งชี้นี้ร่วมกับตัวบ่งชี้ PF_tNoRuntimeW จะทำให้ไม่สามารถกล่าวถึงคุณสมบัตินั้นได้เลยในช่วงเวลาดำเนินงาน (เขียนโปรแกรมโดยใช้คุณสมบัตินั้นไม่ได้เลย) ซึ่งหากยังพยายามกล่าวถึงคุณสมบัตินั้นอยู่ เมื่อสั่งดำเนินงานโปรแกรม ก็ไม่สามารถทำงานได้ จะเกิดข้อผิดพลาดในการแปล (Compilation Error) ขึ้นเสียก่อน
PF_fPreHwnd	ให้คุณสมบัติถูกนำเข้าสู่หน่วยความจำก่อนที่โครงสร้างของวินโดวส์ (Window Structure) ของตัวควบคุมจะถูกสร้างขึ้น ซึ่งจะทำให้คุณสมบัตินั้นมีผลต่อลักษณะของวินโดวส์ (Window Style) ของตัวควบคุมที่จะสร้าง
PF_fSetCheck	ให้วิชวลเบสิกส่งข้อความ VBM_CHECKPROPERTY ก่อนที่จะเปลี่ยนแปลงค่าคุณสมบัติ ซึ่งทำให้กระบวนงานของตัวควบคุมสามารถตรวจสอบความวินโดวส์ถูกต้องของค่าดังกล่าวก่อนได้ หากค่าดังกล่าวไม่ถูกต้องกระบวนงานของตัวควบคุมต้องคืนค่าที่ไม่เท่ากับ 0 ให้กับวิชวลเบสิก ตัววิชวลเบสิกจึงจะไม่เปลี่ยนแปลงค่านั้น
PF_fUpdateOnEdit	ในการใช้กล่องกำหนดค่าคุณสมบัตินั้น ค่าของคุณสมบัติจะไม่เปลี่ยนเป็นค่าใหม่จนกว่าจะกด [Enter] แล้วเท่านั้น แต่ถ้ามีการใช้ตัวบ่งชี้นี้ ค่าของคุณสมบัติจะเปลี่ยนแปลงทันทีทุกครั้งที่มีการกดคีย์บอร์ด ควรใช้ตัวบ่งชี้นี้กับคุณสมบัติแบบอักขระ

5. dataDefault

   เป็นค่าโดยปริยายของคุณสมบัติ

6. npszEnumList

   จะกำหนดค่านี้เมื่อเป็นคุณสมบัติชนิดจำนวนเต็ม 8 บิตแบบไม่คิดเครื่องหมายเท่านั้น

(DT_ENUM) โดยเป็นอักขระที่ใช้เป็นรายการเลือกของคุณสมบัติให้วินโดวส์คุณสมบัติ และคั่นแต่ละรายการด้วย "\ 0 " และ กำหนดค่านี้เป็น NULL ในกรณีที่ไม่มีรายการเลือกใด ๆ

7. enumMax

เป็นค่าสูงสุดของรายการเลือกในกรณีที่เป็นคุณสมบัติชนิด DT_ENUM

จากส่วนประกอบของโครงสร้างตารางข้อมูลคุณสมบัติ จึงสามารถทำการกำหนดคุณสมบัติต่าง ๆ ตามที่ต้องการได้โดยมีขั้นตอนดังนี้คือ

1. กำหนดแต่ละคุณสมบัติในตารางคุณสมบัติ

- กรณีที่เป็นคุณสมบัติแบบคัสทอม

จะต้องมีการกำหนดตามโครงสร้างตารางข้อมูลคุณสมบัติ เช่น

```
PROPINFO    Property_Alignment =
            {
            "Alignment",
            DT_ENUM | PF_fGetData | PF_fSetMsg | PF_fSaveData,
            0, 0, 2,
            "0-Left\0"  "1-Center\0"  "2-Right\0" , 2
            };
```

จากนั้นเพิ่มคุณสมบัตินี้เข้าไปในตารางข้อมูลคุณสมบัติของตัวควบคุม โดยเป็นการให้ที่อยู่ของ Property_Alignment ดังนี้

```
PPROPINFO     Sheet_Properties[ ] =
            {
            PPROPINFO_STD_CTLNAME,
            PPROPINFO_STD_INDEX,
            ....
            & Property_Alignment,
            ....
            NULL
            };
```

- กรณีที่เป็นคุณสมบัติแบบมาตรฐาน

สามารถเพิ่มคุณสมบัตินั้นเข้าไปในตารางข้อมูลคุณสมบัติของตัวควบคุมนั้นได้เลย

2. กำหนดค่าคงที่ เพื่อใช้เป็นดัชนีของแต่ละคุณสมบัติตามลำดับในตารางคุณสมบัติ เพื่อให้การเขียนโปรแกรมจัดการกับคุณสมบัติทำได้สะดวกขึ้น โดยที่ไม่จำเป็นต้องเรียงคุณสมบัติทั้งหมดตามลำดับอักษร ตัววิชวลเบสิกจะจัดการให้ในวินโดวส์คุณสมบัติ เช่น

```
enum    Prop_Index
  {
  IPROPINFO_STD_CTLNAME,
  IPROPINFO_STD_INDEX,
  IPROPINFO_BACK_COLOR,
  ....
  ....
  IPROPINFO_ALIGNMENT,
  ....
  ....
  IPROPINFO_END
  };
```

3. กรณีที่เป็นคุณสมบัติแบบคัสทอม อาจเขียนชุดคำสั่งเพื่อจัดการกับข้อความที่มีผลต่อคุณสมบัติของตัวควบคุม

<u>การกำหนดเหตุการณ์ให้ตัวควบคุม</u>

ตารางข้อมูลเหตุการณ์ มีโครงสร้างซึ่งประกอบไปด้วย

1. npzName

เป็นชื่อของเหตุการณ์ที่ใช้ในวินโดวส์สำหรับเขียนโปรแกรม (SHEET.Hบรรทัดที่ 124)

2. cParms

เป็นจำนวนอาร์กิวเมนต์ทั้งหมด โดยไม่รวมอาร์กิวเมนต์ Index (SHEET.H บรรทัดที่ 125)

3. cwParms

เป็นจำนวนคำ (WORD) ของรายชื่ออาร์กิวเมนต์ โดยจะมีตัวชี้ขนาด 2 ไบต์ไปยังแต่ละอาร์กิวเมนต์ ดังนั้นโดยส่วนใหญ่ cwParms มักมีค่าเป็น 2 เท่า (SHEET.H บรรทัดที่ 126) ของ cParms

4. npParmTypes

เป็นตัวชี้ไปยังข้อมูลแถวลำดับแบบไบต์ที่บอกถึงชนิดของแต่ละอาร์กิวเมนต์ตามลำดับในกระบวนงานเหตุการณ์ (SHEET.H บรรทัดที่ 127)

5. npszParmProf

เป็นข้อความที่ปิดด้วย "\0" ที่จะใช้แสดงอาร์กิวเมนต์ของเหตุการณ์ในวินโดวส์สำหรับเขียนโปรแกรม (SHEET.H บรรทัดที่ 128) โดยใช้เครื่องหมายจุลภาค (,) คั่นแต่ละอาร์กิวเมนต์ และไม่ต้องมีวงเล็บเหมือนในวินโดวส์สำหรับเขียนโปรแกรม และไม่ต้องเขียนอาร์กิวเมนต์ Index ด้วย เพราะวิชวลเบสิกจะจัดการอาร์กิวเมนต์นี้ให้ตัวควบคุมเอง

6. fl

เป็นตัวบ่งชี้ของเหตุการณ์ สามารถกำหนดค่าเป็น PF_fNoUnload หรือ 0 ถ้ากำหนดค่าเป็น PF_fNoUnload วิชวลเบสิกจะไม่นำฟอร์มหรือตัวควบคุมภายในฟอร์มออกจากหน่วยความจำในขณะที่เหตุการณ์นี้เกิดขึ้น

จากส่วนประกอบของโครงสร้างตารางข้อมูลเหตุการณ์ข้างต้น จึงทำให้สามารถกำหนดเหตุการณ์สำหรับตัวควบคุมตามลำดับดังนี้

1. กำหนดเหตุการณ์ต่าง ๆ ไว้ในตารางข้อมูลเหตุการณ์ กรณีที่เหตุการณ์นั้นต้องมีอาร์กิวเมนต์ด้วย จะต้องประกาศชนิดของอาร์กิวเมนต์ด้วยตัวบ่งชี้ชนิดของอาร์กิวเมนต์ก่อนโครงสร้างตารางข้อมูลเหตุการณ์โดยสามารถเลือกใช้ตัวบ่งชี้ได้ดังตารางที่ 5.6

หลังจากกำหนดเหตุการณ์แบบคัสทอมของตัวควบคุมแล้ว ให้เพิ่มเหตุการณ์นั้นเข้าไปในตารางข้อมูลเหตุการณ์ โดยระบุเป็นตัวชี้ไปยังโครงสร้างของเหตุการณ์นั้น ส่วนกรณีที่เป็นเหตุการณ์แบบมาตรฐานนั้นก็สามารถเพิ่มเหตุการณ์นั้นเข้าไปในตารางได้เลย เช่น

```
WORD  Paramtypes_Col_Width_Change [ ] = {ET_I2, ET_I2};
EVENTINFO     Event_Col_Width_Change =
                {
                " ColWidthChange"
                2,
```

```
        4,
        Paramtypes_Col_Width_Change,
        "SelStartRow As Integer, SelEndRow As Integer"
);
```

ตารางที่5.6 แสดงตัวบ่งชี้ชนิดของอาร์กิวเมนต์

ตัวบ่งชี้	ความหมาย
ET_I2	เลขจำนวนเต็ม 16 บิต แบบคิดเครื่องหมาย
ET_I4	เลขจำนวนเต็ม 32 บิต แบบคิดเครื่องหมาย
ET_R4	เลขจำนวนจริงขนาด 4 ไบต์
ET_R8	เลขจำนวนจริงขนาด 8 ไบต์
ET_CY	เลขจำนวนเงินขนาด 8 ไบต์
ET_HLSTR	อักขระ

```
PEVENTINFO    Sheet_Events [ ] =
        {
        PEVENTINFO_STD_CLICK,
        PEVENTINFO_STD_DBLCLICK,
        .....
        &Event_Col_Width_Change,
        .....
       NULL
        };
```

2. กำหนดค่าคงที่ เพื่อใช้เป็นดัชนีของแต่ละเหตุการณ์ตามลำดับ เช่น

```
enum    Event_Index
  {
  IEVENTINFO_STD_CLICK,
  IEVENTINFO_STD_DBLCLICK,
```

```
.....
IEVENTINFO_Col_Width_Change,
.....
IEVENTINFO_END
};
```

3. กรณีที่เป็นเหตุการณ์แบบคัสทอม อาจจะต้องเขียนชุดคำสั่งเพื่อกำหนดว่าเหตุการณ์ใดจะเกิดขึ้นเมื่อใด โดยใช้ฟังก์ชัน VBFireEvent โดยฟังก์ชันนี้จะไม่คืนกลับจนกว่ากระบวนงานของวิชวลเบสิกซึ่งกำหนดขึ้นโดยผู้พัฒนาโปรแกรมด้วยวิชวลเบสิกจะทำงานเสร็จ

การจัดการกับข้อความ WM_PAINT

ตัวกระบวนงานของตัวควบคุมจะจัดการกับข้อความที่มาจากวินโดวส์และวิชวลเบสิกการทำงานที่มีผลต่อลักษณะของหน้าต่างต่าง ๆ และตัวควบคุม เช่น การวาดตัวควบคุม การกดคีย์บอร์ดและเมาส์จะมีข้อความมาจากวินโดวส์ ส่วนข้อความจากวิชวลเบสิกจะเป็นการทำงานกับคุณสมบัติและเหตุการณ์ของตัวควบคุม

การแสดงลักษณะของตัวควบคุม โดยการจัดการในข้อความ WM_PAINT เหมือนกับโปรแกรมประยุกต์ทั่วไปสำหรับวินโดวส์ (SHEET.C บรรทัดที่ 82-94) เมื่อวิชวลเบสิกพิมพ์ฟอร์มออกทางเครื่องพิมพ์ (ด้วย คำสั่ง Printform ของวิชวลเบสิก) ในกรณีที่เป็นตัวควบคุมที่เกี่ยวกับรูปภาพ (Graphical Control) จะมีการสร้าง Printer Compatible Display Context ซึ่งเป็นแฮนเดิลใน พารามิเตอร์ wParam ของข้อความ WM_PAINT ดังนั้นเมื่อค่าของ wParam ไม่เป็นศูนย์ ก็จะใช้ค่าของ hDC ใน wParam แต่ถ้าค่าของ wParam เป็นศูนย์ก็จะใช้ฟังก์ชัน BeginPaint แทน เพื่อหาค่าของ hDC

ส่วนฟังก์ชัน OnPaint นั้นจะทำหน้าที่วาดตัวควบคุมแผ่นตารางทำการ โดยกำหนดการทำงานของฟังก์ชันนี้ไว้ใน SHEET.C (บรรทัดที่ 309-320)

การจัดการกรอบของตัวควบคุม

การแสดงกรอบของตัวควบคุมสามารถกำหนดรูปแบบได้ว่า ต้องการกรอบล้อมรอบหรือไม่ ด้วยการเปลี่ยนแปลงฟิลด์ nWndStyle ในตัวแบบของตัวควบคุม เพราะฟิลด์นี้กำหนดลักษณะ

กรอบของตัวควบคุม

WS_BORDER, //Default Window Style

หากไม่ต้องการให้มีกรอบรอบตัวควบคุมก็ให้ใช้ค่าศูนย์แทน WS_BORDER

0, //Default Window Style

หรือไม่ต้องใช้ค่าศูนย์แทน WS_BORDER แต่ให้เพิ่มคุณสมบัติ BorderStyle เข้าไปในตารางข้อมูลคุณสมบัติด้วยค่าใดค่าหนึ่งต่อไปนี้

PPROPINFO_STD_BORDERSTYLEON หรือ

PPROPINFO_STD_BORDERSTYLEOFF

แต่ละค่าจะแทนคุณสมบัติ BorderStyle เหมือนกัน แต่จะให้ค่าเริ่มต้นของคุณสมบัติต่างกัน ซึ่งต้องเลือกมาใช้เพียงค่าใดค่าหนึ่งเท่านั้น ไม่สามารถใช้พร้อมกันทั้งสองค่าไม่ได้

การขยายขนาดโดยปริยายของตัวควบคุม

เมื่อทำการดับเบิ้ลคลิกที่สัญรูปของตัวควบคุมใด ๆ ในทูลบอกซ์ ก็จะได้ตัวควบคุมหนึ่งตัวปรากฏขึ้นที่กลางฟอร์ม ซึ่งขนาดของตัวควบคุมขณะนี้ เรียกว่า ขนาดโดยปริยายของตัวควบคุม ซึ่งสามารถกำหนดขนาดโดยปริยายของตัวควบคุมได้โดยการจัดการกับข้อความ VBM_GETDEFSIZE จากวิชวลเบสิก

```
case VBM_GETDEFSIZE :
    {
    return (MAKELONG 200,200);
    }
```

ซึ่งเป็นการคืนค่าของความกว้างและความสูงในหน่วยของ Pixel ให้กับวิชวลเบสิก โดย Low Word ของค่าที่คืนเป็นความกว้าง High Word เป็นความสูง

การจัดการคุณสมบัติแบบช่องลำดับ (Array Properties)

คุณสมบัติที่มีการจัดเก็บแบบช่องลำดับ สำหรับตัวควบคุมนี้มีเพียงคุณสมบัติ ColWidth และ RowHeight ซึ่งมีคุณสมบัติดังกล่าว จะมีลักษณะดังนี้คือ

1) เมื่อจะอ่านค่าหรือกำหนดค่าคุณสมบัติจะต้องใช้ตัวชี้ไปยังโครงสร้าง DATASTRUCT ที่มีทั้งข้อมูลและดัชนีของแต่ละช่องที่ใช้เก็บข้อมูล ซึ่งในการอ่านหรือกำหนดค่านั้นจะทำได้ทีละช่องข้อมูลเท่านั้น

2) จำนวนช่องข้อมูลที่กำหนดขึ้นในโครงสร้างของผู้เขียนโปรแกรม จะต้องมากพอที่จะเก็บค่าของคุณสมบัติที่กำหนดขึ้น

3) เมื่อมีการอ่านหรือเปลี่ยนแปลงค่าคุณสมบัตินี้ โดยผู้พัฒนาโปรแกรมประยุกต์ด้วยวิชวลเบสิก กระบวนงานของตัวควบคุมจะต้องตรวจสอบดัชนีของช่องลำดับด้วยว่า มีค่าอยู่ในช่วงที่กำหนดไว้หรือไม่

4) ไม่ควรให้มีการกำหนดค่าของคุณสมบัติชนิดนี้ในช่วงเวลาการออกแบบ ดังนั้นจึงต้องใช้ตัวบ่งชี้ PF_fNoShow ร่วมด้วย ซึ่งก็ทำให้ไม่ต้องมีการบันทึกหรือนำคุณสมบัติเข้าสู่หน่วยความจำแต่อย่างใด (ไม่ต้องใช้ตัวบ่งชี้ PF_fSaveData)

```
#define D_MAXCOLUMN 100
#define D_MAXROW    100
typedef struct tagSHEET
{
....
int ColWidth[D_MAXCOLUMN];
int RowHeight[D_MAXROW];
....
} SheetStru;
```

เป็นการกำหนดคุณสมบัติแบบช่วงลำดับของจำนวนเต็มในโครงสร้างของผู้เขียนโปรแกรม ส่วนโครงสร้าง PROPINFO กำหนดดังนี้

```
PROPINFO Property_Col_Width =
{
"ColWidth",
DT_SHORT | PF_fPropArray | PF_fGetMsg | PF_fSetMsg | PF_fNoShow,
OFFSETIN (SheetStru, ColWidth),
0, 0, NULL, 0
};
```

คุณสมบัติแบบช่องลำดับต้องใช้ตัวบ่งชี้ PF_fPropArray เสมอ และใช้ PF_fGetMsg PF_fSetMsg และ PF_fNoShow ร่วมด้วย ส่วนตัวบ่งชี้ PF_fSaveData นั้นไม่ต้องใช้ และให้เพิ่มดัชนีและที่อยู่ของคุณสมบัติในตารางข้อมูลคุณสมบัติด้วย

IPROPINFO_Col_Width

&Property_Col_Width

ตัวกระบวนงานของตัวควบคุมต้องจัดการกับข้อความ VBM_GETPROPERTY จากวิชวลเบสิกโดยการตรวจสอบดัชนีและคืนค่าที่ถูกต้องแก่วิชวลเบสิก

```
case VBM_GETPROPERTY:
    switch (wParam)
    {
   case IPROPINFO_Col_Width:
       {
    LONG i;
    LPDATASTRUCT LpDs = (LPDATASTRUCT)lParam;
    i = LpDs->index[0].data;
        if (i < 0 || i >= D_MAXCOLUMN)
      return ERR_BADINDEX;
    LpDs->data = (LONG)LpSheet->ColWidth[i];
       }
        return 0;
   ....
   default :
       return VBDefControlProc(hctl, hWnd, msg, wParam, lParam);
   }
```

พารามิเตอร์ lParam ในที่นี้เป็นตัวชี้ไปยังโครงสร้าง DATASTRUCT โครงสร้างนี้มีฟิลดที่สำคัญคือ index[0].data คือดัชนีของช่องข้อมูลที่ต้องการ และ data คือข้อมูลในช่องข้อมูล หลังจากประกาศตัวแปร LpDs แล้ว ก็จะหาค่าดัชนีโดย

```
i   =   LpDs->index[0].data;                    // Get Index
```

ดังนั้น หากชุดคำสั่งของวิชวลเบสิกต้องการอ่านค่า ColWidth (5) ค่าของ i ก็จะเป็น 5 จึงนำค่าของ i มาตรวจสอบว่า อยู่ในช่วงที่กำหนดไว้หรือไม่ ถ้าไม่อยู่ในช่วงที่กำหนด จะคืนค่าที่ไม่เท่ากับศูนย์ให้วิชวลเบสิก

```
if (i = 0 || i >= D_MAXCOLUMN)          // Is index out of range?
  return ERR_BADINDEX;
```

จากนั้นจึงทำการโอนถ่ายข้อมูลของช่องที่ต้องการให้กับฟิลด์ LpDs->data คือ

```
LpDs->data = (LONG)LpSheet->ColWidth[i];      //Transfer Data
```

แล้วคืนค่าศูนย์ให้กับวิชวลเบสิก แสดงว่าวิชวลเบสิกสามารถอ่านค่าได้โดยสมบูรณ์ โดยค่าที่อ่านได้จะอยู่ในรูป LpDs->data เรียบร้อยแล้ว

ส่วนการจัดการกับข้อความ VBM_SETPROPERTY ก็คล้ายกับ VBM_GETPROPERTY ต่างกันที่ทิศทางของการโอนถ่ายข้อมูลเท่านั้น

```
case VBM_SETPROPERTY:
    switch (wParam)
    {
  case IPROPINFO_Col_Width:
      {
    LONG i;
    LPDATASTRUCT LpDs = (LPDATASTRUCT)lParam;
    i = LpDs->index[0].data;
        if (i < 0 || i >= D_MAXCOLUMN)
      return ERR_BADINDEX;
         LpSheet->ColWidth[i] = (SHORT)LpDs->data;
      }
       return 0;
    ....
  default :
      return VBDefControlProc(hctl, hWnd, msg, wParam, lParam);
  }
```

ฟังก์ชันที่สำคัญของ DLL สำหรับวิชวลเบสิก

1. ฟังก์ชันภายในแฟ้ม LIBENTRY.OBI

เนื่องจากแฟ้มตัวควบคุมที่สร้างขึ้น (แฟ้มที่มีสกุล SHEET.VBX) จะมีลักษณะเป็น DLL จึงต้องมีกระบวนงานการกำหนดค่าเริ่มต้นที่แตกต่างไปจากโปรแกรมประยุกต์ทั่ว ๆ ไป ซึ่งโปรแกรมประยุกต์จะติดต่อกับแฟ้มตัวควบคุมสกุล .VBX ได้ด้วยฟังก์ชัน Library Initialization ที่อยู่ภายในแฟ้ม LIBENTRY.OBI ซึ่งถ้าใช้ Microsoft C/C++ Compiler ตั้งแต่รุ่น 7.0 ขึ้นไป หรือ Microsoft Visual C/C++ ฟังก์ชันนี้จะถูกเชื่อมโยงเข้ากับแฟ้มตัวควบคุมสกุล .VBX โดยอัตโนมัติ แต่ถ้าใช้ Microsoft C รุ่น 6.0 จะต้องทำการเชื่อมโยงฟังก์ชัน Library Initialization เข้ากับแฟ้มตัวควบคุมสกุล .VBX เอง ซึ่งแฟ้ม LIBENTRY.OBI นี้ วิชวลเบสิกได้เตรียมให้ไว้ในสารบบย่อย CDK

2. ฟังก์ชันภายในแฟ้มตัวควบคุมสกุล .C

แฟ้มตัวควบคุมแบบคัสทอม จะต้องมีฟังก์ชันที่ใช้ในการทำงาน 2 ฟังก์ชัน และจบการทำงาน 2 ฟังก์ชัน ได้แก่

2.1) ฟังก์ชัน LibMain

ฟังก์ชันนี้เป็นจุดเข้า (Entry Point) ของ DLL ซึ่งจะถูกเรียกใช้เมื่อแฟ้มตัวควบคุมสกุล .VBX ถูกนำมาบรรจุในหน่วยความจำเป็นครั้งแรก (SHEET.C บรรทัดที่ 146-160) ฟังก์ชันนี้จะกำหนดค่า hmodDLL เป็นตัวแปรแบบโกลบอล ซึ่งฟังก์ชัน VBINITCC จะเรียกใช้ตัวแปรนี้ด้วย โดยที่ตัวแปร hmodDLL จะเก็บค่าแฮนเดิลของแฟ้มตัวควบคุมไว้

2.2) ฟังก์ชัน VBINTCC

ฟังก์ชันนี้เป็นจุดเข้าของวิชวลเบสิก (SHEET.C บรรทัดที่ 124-132) ซึ่งจะทำหน้าที่ลงทะเบียนให้กับแต่ละตัวควบคุมในแฟ้มตัวควบคุม .VBX ฟังก์ชัน LibMain และ VBINITCC จะมีข้อแตกต่างกัน คือ ฟังก์ชัน LibMain จะถูกเรียกใช้เพียงครั้งเดียวในครั้งแรกที่มีการนำแฟ้มตัวควบคุมสกุล .VBX เข้าสู่หน่วยความจำ แต่ฟังก์ชัน VBINITCC จะถูกเรียกใช้ทุกครั้งที่โปรแกรมประยุกต์นำแฟ้มตัวควบคุมเข้าหน่วยความจำ เช่น กรณีที่โปรแกรมประยุกต์ 2โปรแกรมมีการเรียกใช้แฟ้มตัวควบคุมสกุล .VBX ในขณะเดียวกันจะมีผลให้มีการเรียกใช้ฟังก์ชัน LibMain เพียงครั้งแรกครั้งเดียว ส่วนฟังก์ชัน VBINITCC จะมีการเรียกใช้ถึง 2 ครั้ง

2.3) ฟังก์ชัน VBTERMCC

ฟังก์ชันนี้ใช้ในการยกเลิกการจองทรัพยากรและยกเลิกค่าต่าง ๆ (SHEET.C บรรทัดที่

137-141) มีลักษณะการทำงานคล้ายกับ ฟังก์ชัน VBINITCC โดยจะถูกเรียกทุกครั้งที่โปรแกรมประยุกต์ยกเลิกการใช้งานตัว ควบคุมแบบดัสทอม โดยที่ฟังก์ชันนี้อาจจะมีหรือไม่มีในแฟ้มตัวควบคุมสกุล .C ก็ได้

2.4) ฟังก์ชัน WEP

ฟังก์ชันนี้ (SHEET.C บรรทัดที่ 166-195) จะถูกเรียกใช้เมื่อแฟ้มตัวควบคุมสกุล .VBX จะถูกนำออกจากหน่วยความจำ ซึ่งฟังก์ชันนี้จะทำการคืนค่า 1 เท่านั้น เพื่อแสดงว่า แฟ้มตัวควบคุมสกุล .VBX นั้นจบการทำงานโดยสมบูรณ์ ฟังก์ชัน WEP จะถูกแปลและเชื่อมโยงเข้ากับตัวควบคุม ในกรณีที่ใช้ Microsoft C Compiler รุ่น 6.0 แต่ถ้าใช้รุ่น 7.0 ขึ้นไป หรือ Microsoft Visual C/C++ ฟังก์ชันนี้จะถูกเชื่อมโยงเข้ากับแฟ้ม ตัวควบคุมสกุล .VBX ไว้โดยอัตโนมัติ

<u>การสร้างสัญรูปแทนตัวควบคุมในทูลบอกซ์</u>

ในการสร้างสัญรูปเพื่อใช้แทนตัวควบคุมในทูลบอกซ์นั้น จะต้องสร้างขึ้นมาเพื่อใช้กับจอภาพแบบ วีจีเอ แบบอีจีเอ และแบบสีเดียว ซึ่งจะต้องใช้แฟ้มรูปภาพบิตแมพ 4 แฟ้ม คือ 2 แฟ้มสำหรับจอภาพวีจีเอ เพราะต้องใช้สำหรับแสดงตอนที่ไม่ได้มีการคลิกเมาส์ในสัญรูป 1 แฟ้ม และมีการคลิกเมาส์อีก 1 แฟ้ม และสำหรับจอภาพอีจีเอกับจอภาพสีเดียวอีกอย่างละ 1 แฟ้ม เพราะเมื่อมีการคลิกเมาส์ที่สัญรูป ก็จะใช้การสลับสีของแต่ละบิตในภาพแทน ส่วนจอภาพขนาดใหญ่จะใช้แฟ้ม 2 แฟ้มเดิมที่ใช้กับจอวีจีเอ แฟ้มรูปภาพบิตแมพทั้ง 4 แฟ้มเป็นทรัพยากรของตัวควบคุม จึงต้องมีหมายเลขแทนแต่ละแฟ้มดังตารางที่ 5.7

สำหรับค่า idBmpPalette นั้นสามารถกำหนดเป็นเลขจำนวนเต็มใดก็ได้ให้กับจอภาพวีจีเอเมื่อไม่ได้ถูกคลิกด้วยเมาส์ ส่วนจอภาพอื่น ๆ สามารถกำหนดโดยการเพิ่มค่าขึ้นอีก 1 3 และ 6 ตามลำดับ ซึ่งการกำหนดค่าซ้ำกับตัวควบคุมอื่น ๆ จะไม่มีผลกระทบกับตัวควบคุมนี้

ส่วนแฟ้มรูปภาพนั้นสามารถสร้างได้จากโปรแกรมสำหรับวาดภาพ เช่น Window PaintBrush App Studio ของ Microsoft Visual C/C++ เป็นต้น โดยสร้างแฟ้มรูปภาพบิตแมพสำหรับจอภาพวีจีเอ และจอภาพสีเดียวให้มีขนาด 28 x 28 ทุกภาพ ส่วนจอภาพอีจีเอจะมีขนาด 28 x 22 กรณีที่ใช้ภาพที่มีขนาดใหญ่กว่า เช่น ขนาด 30 x 30 วิชวลเบสิกจะตัดส่วนที่เกินทิ้ง แล้วแสดงเฉพาะส่วนที่อยู่ในขนาด 28 x 28 หรือ 28 x 22 เท่านั้น

ตารางที่ 5.7 แสดงความสัมพันธ์ชนิดของจอภาพกับidBmpPalette

ชนิดของจอภาพ	หมายเลข
จอภาพวีจีเอ เมื่อไม่มีการคลิกด้วยเมาส์	idBmpPalette เป็นเลขจำนวนเต็มที่กำหนดไว้ในตัวแบบของตัวควบคุม
จอภาพวีจีเอ เมื่อถูกคลิกด้วยเมาส์	idBmpPalette +1
จอภาพสีเดียว	idBmpPalette +3
จอภาพอีจีเอ	idBmpPalette +6

ขั้นตอนวิธี (Algorithm) ที่สำคัญ

ตัวควบคุมจะมีขั้นตอนการดำเนินงานที่สำคัญดังนี้คือ

การคัดลอกข้อมูลในเซลล์

1) กำหนดตำแหน่งของแถวนอนและแถวสดมภ์ที่ต้องการคัดลอก โดยผ่านคุณสมบัติ CurrentRow และ CurrentCol ตามลำดับ

2) กำหนดจุดหมายปลายทางของแถวนอนและแถวสดมภ์ที่ต้องการ โดยผ่านคุณสมบัติ DestinationRow และ DestinationCol ตามลำดับ

3) ตรวจสอบเซลล์ต้นทาง และเซลล์ปลายทางว่ามีการจัดสรรหน่วยความจำมาก่อนหรือไม่

4) กรณีที่เซลล์ต้นทางยังไม่มีการจัดสรรหน่วยความจำมาก่อน ขณะที่เซลล์ปลายทางมีการจัดสรรหน่วยความจำแล้ว ก็จะต้องกำหนดประเภทของข้อมูลและข้อมูลของเซลล์ปลายทางเป็นอักขระว่าง

5) กรณีที่เซลล์ต้นทางมีการจัดสรรหน่วยความจำแล้ว ขณะที่เซลล์ปลายทางยังไม่มีการจัดสรรมาก่อน ก็จะต้องทำการจัดสรรหน่ายความจำให้กับเซลล์ปลายทางก่อน จากนั้นจึงทำการตรวจสอบประเภทของข้อมูลว่าเป็นข้อความ เลขจำนวนเต็ม จำนวนจริงที่มีทศนิยม หรือสูตร เพื่อที่จะได้ทำการคัดลอกข้อมูลในเซลล์ให้ถูกต้องตามรูปแบบ แล้วจึงทำการคักลอกข้อมูลในเซลล์พร้อมคุณสมบัติการจัดวางตำแหน่งของเซลล์ต้นทางไปยังเซลล์ปลายทางที่กำหนด

ดังนั้น เมื่อทำการคัดลอกข้อมูลแล้ว เซลล์ปลายทางจะมีข้อมูลและการจัดวางตำแหน่งของข้อมูลเหมือนกับเซลล์ต้นทาง

การแทรกแถวนอน

1) กำหนดตำแหน่งของแถวนอนที่ต้องการแทรก โดยผ่านคุณสมบัติ CurrentRow

2) เก็บตัวชี้ที่ไปยังแถวสดมภ์ ซึ่งอยู่ในช่องลำดับของแถวนอนสุดท้ายไว้

3) ลบตัวชี้ที่อยู่ในทุกช่องลำดับของแถวสดมภ์ ซึ่งแถวนอนสุดท้ายชี้อยู่ พร้อมทั้งคืนเนื้อที่ที่จัดสรรหน่วยความจำของแต่ละเซลล์ข้อมูล ดังนั้นช่องลำดับสุดท้ายของแถวนอน จะยังคงเก็บตัวชี้ไปยังแถวสดมภ์เดิมที่เคยชี้อยู่ เพียงแต่แถวสดมภ์นั้นจะเก็บตัวว่าง (Null) ในทุกช่องลำดับ เพื่อแสดงว่า ในแถวสดมภ์นั้นไม่มีเซลล์ข้อมูลใดอยู่เลย

4) ทำการย้ายตัวชี้ในช่องลำดับของแถวนอน ตั้งแต่แถวนอนที่ต้องการแทรกลงทีละช่องลำดับจนครบจำนวนแถวนอนทั้งหมด

5) กำหนดให้ช่องลำดับของแถวนอนที่ต้องการแทรกชี้ไปยังแถวสดมภ์ว่างที่เก็บไว้

ขั้นตอนการแทรกแถวนอนดังกล่าวข้างต้น จะเป็นการแทรกแถวนอนแบบก่อนหน้าแถวนอนที่ระบุ ดังนั้นกรณีที่มีข้อมูลครบทั้ง 100 แถวนอน (โดยที่แถวนอนแรกจะเก็บหัวเรื่องของแต่ละแถว) แล้วทำการแทรกแถวนอน 1 แถว จะมีผลทำให้ข้อมูลที่อยู่ในแถวนอนสุดท้ายถูกลบออกไป

การลบแถวนอน

1) กำหนดตำแหน่งแถวนอนที่ต้องการลบ โดยผ่านคุณสมบัติ CurrentRow

2) เก็บตัวชี้ที่ชี้ไปยังแถวสดมภ์ ซึ่งอยู่ในช่องลำดับของแถวนอนที่ต้องการลบ

3) ลบตัวชี้ที่อยู่ในทุกช่องลำดับของแถวสดมภ์ ซึ่งแถวนอนที่ต้องการลบชี้อยู่ พร้อมทั้งคืนเนื้อที่ที่จัดสรรให้หน่วยความจำของแต่ละเซลล์ข้อมูล ดังนั้นช่องลำดับของแถวนอนที่ต้องการลบ จะยังคงเก็บตัวชี้ไปยังแถวสดมภ์เดิมที่เคยชี้อยู่ เพียงแต่แถวสดมภ์นั้นจะเก็บตัวว่างในทุกช่องลำดับ เพื่อแสดงว่า ในแถวสดมภ์นั้นไม่มีเซลล์ข้อมูลใดอยู่เลย

4) ทำการย้ายตัวชี้ในช่องลำดับของแถวนอน ตั้งแต่แถวนอนที่ต้องการลบขึ้นทีละช่องลำดับจนครบจำนวนแถวนอนทั้งหมด

5) กำหนดให้ช่องลำดับสุดท้ายของแถวนอนเก็บค่าตัวชี้ของแถวสดมภ์ว่างที่เก็บไว้

การแทรกแถวสดมภ์

1) กำหนดตำแหน่งแถวสดมภ์ที่ต้องการแทรก โดยผ่านคุณสมบัติ CurrentCol

2) ทำการแทรกแถวสดมภ์ในทุก ๆ แถวนอน โดยการ

2.1) ตรวจสอบเซลล์ข้อมูลในช่องลำดับสุดท้ายของแถวสดมภ์นั้นว่ามีการจัดสรรหน่วยความจำไว้หรือไม่

2.2) กรณีที่เซลล์นั้นมีการจัดสรรหน่วยความจำไว้ จะต้องทำการลบตัวชี้ที่อยู่ในช่องลำดับสุดท้ายของแถวสดมภ์

2.3) ทำการย้ายตัวชี้ในช่องลำดับของแถวสดมภ์ตั้งแต่แถวสดมภ์ที่ต้องการแทรกลงทีละช่องลำดับจนครบจำนวนแถวสดมภ์ทั้งหมด

2.4) กำหนดให้ช่องลำดับของแถวสดมภ์ที่ต้องการแทรกเป็นตัวว่าง

การแทรกแถวสดมภ์จะมีลักษณะเดียวกับการแทรกแถวนอน กล่าวคือ จะเป็นการแทรกแถวสดมภ์แบบก่อนหน้าแถวสดมภ์ที่ต้องการแทรก ดังนั้น กรณีที่มีเซลล์ข้อมูลครบทั้ง 100 แถวสดมภ์ (โดยที่แถวสดมภ์แรกจะเก็บหัวเรื่องของแต่ละแถว) แล้วทำการแทรกแถวสดมภ์ 1 แถว จะมีผลทำให้ข้อมูลที่อยู่ในแถวสดมภ์สุดท้ายถูกลบออกไป

<u>การลบแถวสดมภ์</u>

1) กำหนดตำแหน่งแถวสดมภ์ที่ต้องการลบ โดยผ่านคุณสมบัติ CurrentCol

2) ทำการลบแถวสดมภ์ในทุก ๆ แถวนอน โดยการ

2.1) ตรวจสอบเซลล์ข้อมูลในช่องลำดับของแถวสดมภ์ที่ต้องการลบ ว่ามีการจัดสรรหน่วยความจำไว้หรือไม่

2.2) กรณีที่เซลล์นั้นมีการจัดสรรหน่วยความจำไว้ จะต้องทำการลบตัวชี้ที่อยู่ในช่องลำดับของแถวสดมภ์ที่ต้องการลบ

2.3) ทำการย้ายตัวชี้ในช่องลำดับของแถวสดมภ์ตั้งแต่แถวสดมภ์ที่ต้องการลบขึ้นทีละช่องลำดับจนครบจำนวนแถวสดมภ์ทั้งหมด

2.4) กำหนดให้ช่องลำดับสุดท้ายของแถวสดมภ์เป็นตัวว่าง

<u>การเรียงลำดับข้อมูลในแถวนอน</u>

1) กำหนดตำแหน่งแถวนอนที่ต้องการเรียงลำดับ และชนิดของการเรียงลำดับข้อมูลว่าต้องการเรียงลำดับข้อมูลแบบขึ้นหรือลง โดยผ่านคุณสมบัติ CurrentRow และ SortOrder ตามลำดับ

2) กำหนดตำแหน่งเริ่มต้นและตำแหน่งสิ้นสุดของแถวสดมภ์ที่ต้องการเรียงลำดับ โดยผ่านคุณสมบัติ SelStartCol และ SelEndCol ตามลำดับ

3) ทำการเปรี ยบเทียบค่าของข้อมูลในเซลล์ทุกเซลล์ทตั้งแต่แถวสดมภ์เริ่มต้น จนถึงแถวสดมภ์สิ้นสุด และเรียงลำดับข้อมูลแบบควิกซอร์ต (QuickSort)

การเรียงลำดับข้อมูลในแถวสดมภ์

1) กำหนดตำแหน่งแถวสดมภ์ที่ต้องการเรียงลำดับและชนิดของการเรียงลำดับข้อมูลว่าต้องการเรียงลำดับข้อมูลแบบขึ้นหรือลง โดยผ่านคุณสมบัติ CurrentCol และ SortOrder ตามลำดับ

2) กำหนดตำแหน่งเริ่มต้นและตำแหน่งสิ้นสุดของแถวนอนที่ต้องการเรียงลำดับ โดยผ่านคุณสมบัติ SelStartRow และ SelEndRow ตามลำดับ

3) ทำการเปรียบเทียบค่าของข้อมูลในเซลล์ทุกเซลล์ตั้งแต่แถวนอนเริมต้นจนถึงแถวนอนสิ้นสุด และเรียงลำดับข้อมูลแบบควิกซอร์ต

การคำนวณทางคณิตศาสตร์

1) ระบุตำแหน่งของเซลล์ผลลัพธ์ที่ต้องการทำการคำนวณโดยผ่านคุณสมบัติ CurrentRow และ CurrentCol

2) อ่านสูตรคำนวณที่เก็บไว้ในเซลล์ข้อมูลที่กำหนด และทำการแปลงตำแหน่งเซลล์ข้อมูลที่เกี่ยวข้องให้เป็นค่าที่เก็บอยู่ในเซลล์ข้อมูลนั้น ๆ

3) ทำการแปลงนิพจน์ทางคณิตศาสตร์ที่อยู่ในรูปสัญกรณ์เติมกลาง (Infix Notation) ให้อยู่ในรูปสัญกรณ์เติมหลัง (Postfix Notaion) โดยใช้แถวกองซ้อน (Stack) ซึ่งจะต้องคำนึงถึงลำดับการทำก่อน (Precedence) ของตัวดำเนินการ (Operator) ทางคณิตศาตร์ กล่าวคือ ตัวดำเนินการ คูณ (*) และหาร (/) จะมีลำดับการทำก่อนสูงกว่าตัวดำเนินการบวก (+) และ ลบ (-) นอกจากนั้นยังต้องพิจารณาอักขระในนิพจน์สัญกรณ์เติมกลางเรียงตามลำดับจากซ้ายไปขวา

4) คำนวณค่าทางคณิตศาสตร์ของนิพจน์ที่เป็นสัญกรณ์เติมหลังโดยใช้แถวกองซ้อน

5) นำค่าที่ได้จากการคำนวณเก็บที่เซลล์ผลลัพธ์ที่ระบุ

การคำนวณทางคณิตศาสตร์ จะทำการคำนวณกับตัวดำเนินการบวก ลบ คูณ และหารเท่านั้น โดยไม่มีทั้งวงเล็บเปิด และวงเล็บปิดอยู่ในสูตร นอกจากนั้น รูปแบบการอ้างอิงเซลล์ที่เกี่ยวข้องในสูตรจะต้องเป็นดังนี้คือ CR โดยที่ C แทน หัวเรื่องของแถวสดมภ์ที่เป็นตัวอักขระ และ R แทน หัวเรื่องของแถวนอนที่เป็นตัวเลข เช่น AA3 หมายถึง เซลล์ที่อยู่ในแถวสดมภ์ที่ AA และแถวนอนที่ 3 กรณีที่ชนิดของเซลล์ข้อมูลไม่ใช่ตัวเลขที่สามารถทำการคำนวณได้ เช่น เป็น

ข้อความ จะกำหนดค่าโดยปริยายให้เป็นศูนย์ เพื่อที่สามารถป้องกันความผิดพลาดที่อาจเกิดขึ้นได้

การทดสอบตัวควบคุม

ทำการทดสอบความสามารถของตัวควบคุมด้วยเครื่องคอมพิวเตอร์ที่เข้ากันได้กับเครื่องของ IBM ใช้หน่วยประมวลผลกลาง (CPU) 80486DX หน่วยความจำ 4 เมกะไบต์ ฮาร์ดดิสก์ที่มีความจุ 334 เมกะไบต์ และจอภาพวีจีเอ

1) ทดสอบความสามารถในการคัดลอกข้อมูลในเซลล์

กำหนดให้มีการทำงานร่วมกันของ 3 ตัวควบคุมได้แก่ TextBox CommandButton และ Sheet โดยคุณสมบัติของแต่ละตัวควบคุมเป็นดังนี้

Name = Command1

Name = Command2

Name = Text1

Name = Sheet1

และกำหนดกระบวนงานไว้ดังนี้

```
Sub   Command1_Click()
      Sheet1.DestinationRow = 6
      Sheet1.DestinationCol  = 2
      Sheet1.Action          = 5
End Sub

Sub Command2_Click()
      Dim i,j
      For i = 1 To 5
            Sheet1.CurrentCol = i
            For j = 1 To 5
                  Sheet1.CurrentCol = j
                  Sheet1.IntValue = i+j
            Next j
```

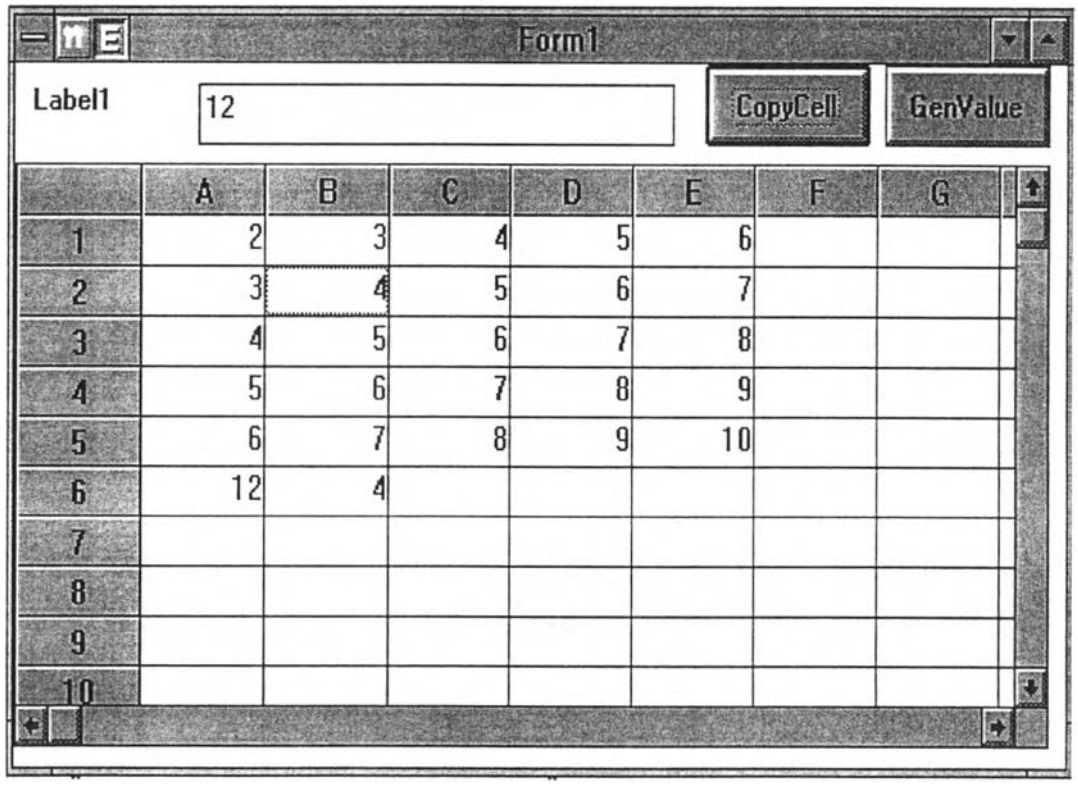
```
        Next i
End Sub

Sub Sheet1_KeyDown (KeyCode As Integer, Shift As Integer)
        Dim i%
        Select Case KeyCode
            Case KEY_LEFT
                i = Sheet1.CurrentCol
                i = i-1
                If i >= 0 Then Sheet1.CurrentCol = i
            Case KEY_RIGHT
                i = Sheet1.CurrentCol
                i = i+1
                If i <= 99 Then Sheet1.CurrentCol = i
            Case KEY_UP
                i = Sheet1.CurrentRow
                i = i-1
                If i >= 0 Then Sheet1.CurrentCol = i
            Case KEY_DOWN
                i = Sheet1.CurrentRow
                i = i+1
                If i >= 52 Then Sheet1.CurrentCol = i
        End Select
End Sub

Sub Text1_KeyPress (KeyAscii As Integer)
        If KeyAscii = 13 Then
            Sheet1.InValue = Text1.Text
            KeyAscii = 9
        End If
```

End Sub

เมื่อสั่งดำเนินงานแล้ว กดที่ปุ่ม Command2 โปรแกรมจะทำการกำหนดค่าลงในเซลล์ข้อมูลตั้งแต่เซลล์ที่ A1 ถึง E5 ขณะเดียวกัน ถ้าต้องการกำหนดค่าของเซลล์ที่ต้องการเอง สามารถทำได้โดยการคลิกเมาส์ลงในเซลล์ที่ต้องการ จากนั้นก็ทำการพิมพ์ค่าที่ต้องการลงใน TextBox ตัวควบคุมแผ่นตารางทำการจะทำการกำหนดค่าและประเภทของข้อมูลลงในเซลล์ที่ระบุให้

วิธีการทดสอบการคัดลอกข้อมูลสามารถทำได้โดยการคลิกเมาส์ที่เซลล์ข้อมูลต้นทางที่ต้องการคัดลอก แล้วกดปุ่ม Command1 ตัวควบคุมแผ่นตารางทำการจะทำการคัดลอกข้อมูลที่อยู่ในเซลล์ต้นทางพร้อมทั้งทำการจัดวางตำแหน่งข้อมูลให้เหมือนกับเซลล์ต้นทางลงในเซลล์ปลายทางที่ระบุให้เหตุการณ์คลิกของ Command1 ดังรูปที่ 5.1

สรุปได้ว่าตัวควบคุมแผ่นตารางทำการสามารถทำการคัดลอกข้อมูลในเซลล์ได้จริง และสามารถทำงานร่วมกับตัวควบคุมอื่นๆ ได้อย่างถูกต้องตามกระบวนงานและคุณสมบัติที่กำหนดขึ้น



รูปที่ 5.1 แสดงการคัดลอกข้อมูลในเซลล์

2) ทดสอบความสามารถในการจัดวางตำแหน่งข้อมูลในเซลล์

กำหนดให้ใช้ตัวควบคุมคุณสมบัติและเหตุการณ์จากการทดสอบการคัดลอกข้อมูลในเซลล์ นอกจากนั้นยังเพิ่มตัวควบคุม OptionButton ขึ้น 3 ตัว โดยมีคุณสมบัติของแต่ละตัวเป็นดังนี้

Name = Option1

Name = Option2

Name = Option3

Option1.Caption = Left

Option2.Caption = Center

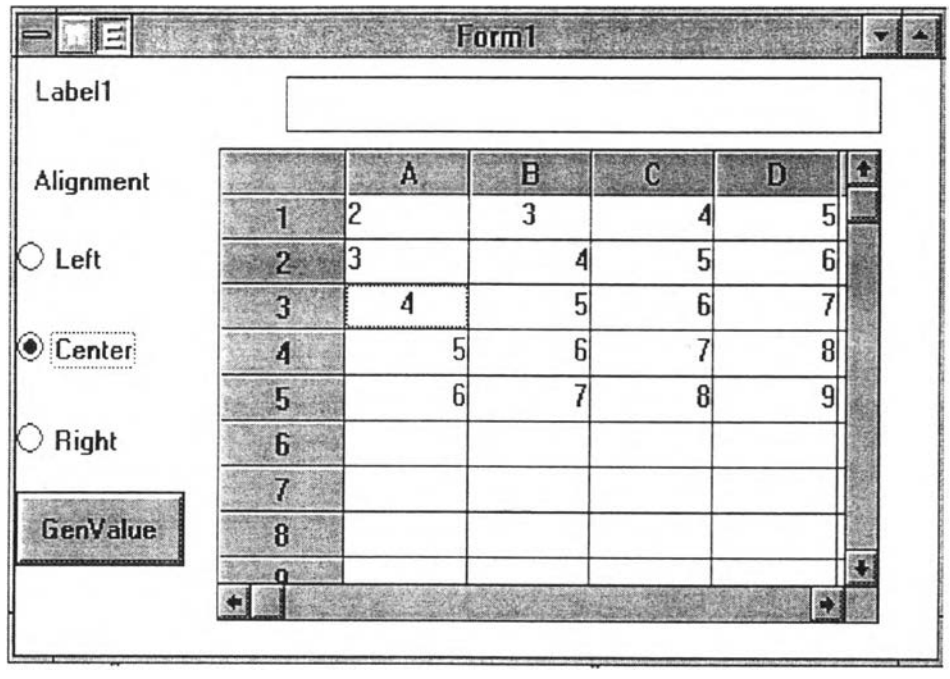
Option3.Caption = Right

และกำหนดกระบวนงานเพิ่มดังนี้

```
Sub   Option1_Click()
      Sheet1.Alignment = 0
End Sub

Sub Option2_Click()
      Sheet1.Alignment = 1
End Sub

Sub Option3_Click()
      Sheet1.Alignment = 2
End Sub
```

เมื่อสั่งดำเนินงานและกดปุ่ม Command2 แล้ว ตัวควบคุมแผ่นตารางทำการแสดงค่าข้อมูลที่อยู่ในเซลล์นั้นๆ โดยมีการจัดวางตำแหน่งชิดขวา ซึ่งถ้าต้องการให้ชิดซ้ายก็ทำได้โดยการคลิกเมาส์ในเซลล์ที่ต้องการ จากนั้นให้คลิกเมาส์ที่ OpionButton ของ Left ตัวควบคุมแผ่นตารางทำการก็จะทำการแสดงข้อมูลในเซลล์โดยมีการจัดวางตำแหน่งชิดซ้ายตามตาราง ส่วนการจัดวางตำแหน่งข้อมูลแบบกึ่งกลางก็จะสามารถทำได้ในลักษณะเดียวกันเพียงแต่ต้องคลิกเมาส์ที่ OptionButton ของ Center แทน ดังรูปที่ 5.2

สรุปได้ว่าตัวควบคุมแผ่นตารางทำการสามารถจัดวางตำแหน่งข้อมูลของเซลล์ที่กำหนดได้ทั้งการวางชิดซ้าย กึ่งกลางและชิดขวาได้



รูปที่ 5.2 แสดงการจัดวางตำแหน่งข้อมูลในเซลล์

3) ทดสอบความสามารถในการป้องกันการแก้ไขข้อมูลในเซลล์และการยกเลิกการป้องกันแก้ไขข้อมูลในเซลล์

กำหนดให้มีการทำงานร่วมกันของ 4 ตัวควบคุมได้แก่ TextBox OptionButton CommandButton และ Sheet โดยคุณสมบัติของแต่ละตัวควบคุมเป็นดังนี้

Name = Command1

Name = Text1

Name = Sheet1

Name = Option1

Name = Option2

Option1.Caption = Unlock

Option2.Caption = Lock

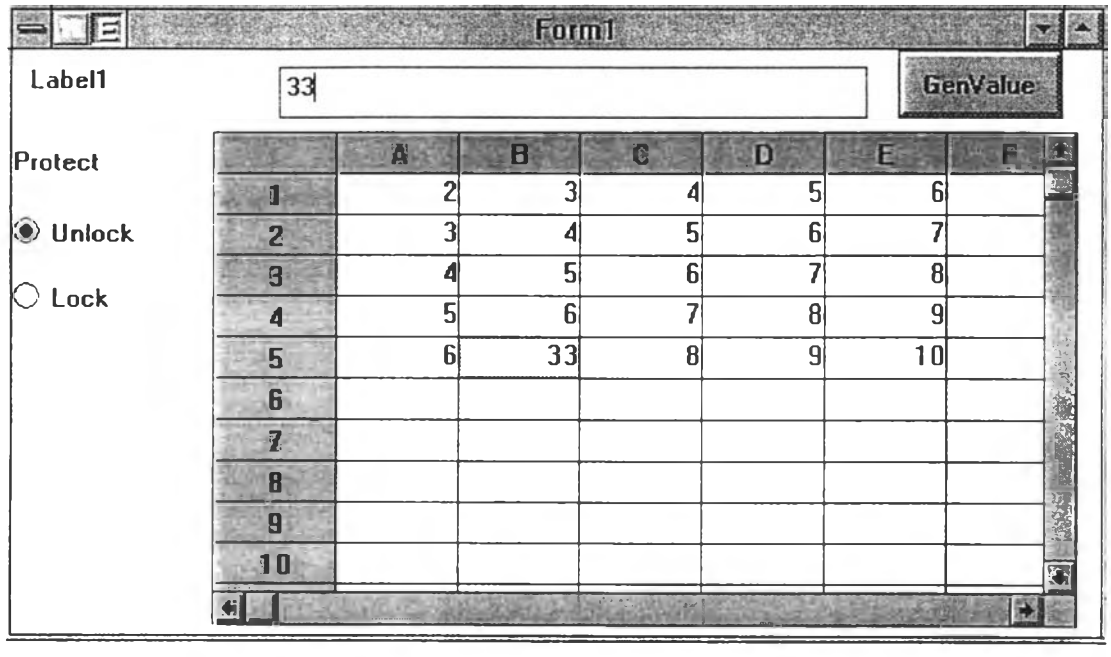
และกำหนดกระบวนงาน Sheet1_KeyDown และ Text1_KeyPress เหมือนกับ กระบวนงานที่ใช้ทดสอบความสามารถในการคัดลอกข้อมูลในเซลล์ นอกจากนั้นต้องเพิ่มกระบวนงานดังนี้คือ

```
Sub Option1_Click()
    Sheet1.Protect = 0
End Sub

Sub Option2_Click()
    Sheet1.Protect = 1
End Sub

Sub Command1.Click()
    Dim i,j
    For i = 1 to 5
        Sheet1.CurrentRow = i
        For j - 1 to 5
            Sheet1.CurrentCol = j
            Sheet1.IntValue = i + j
        Next J
    Next i
End Sub
```

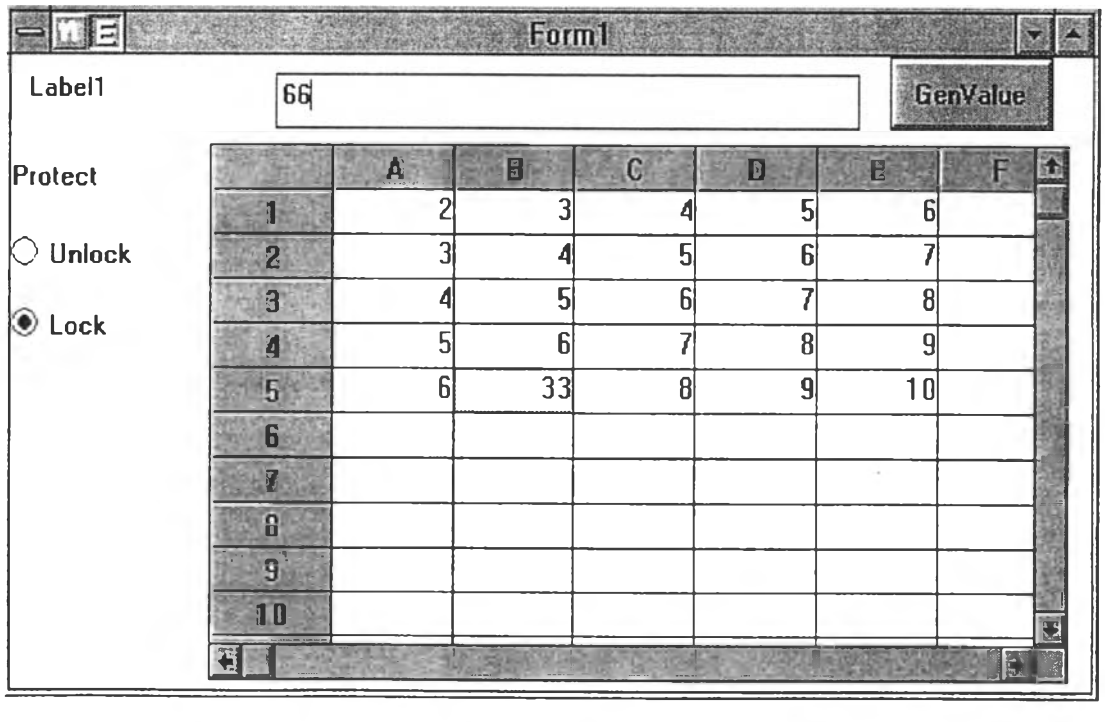
เมื่อทำการสั่งดำเนินงาน และกดปุ่ม Command1 แล้ว ตัวควบคุมแผ่นตารางทำการจะแสดงข้อมูลที่อยู่ในเซลล์ โดยที่สามารถป้องกันการแก้ไขข้อมูลได้ด้วยการคลิกเมาส์ไปที่เซลล์ต้องการป้องกันแล้วคลิกเมาส์ที่ Option ของ Lock วิธีการที่จะทดสอบว่าเซลล์นี้ได้ถูกป้องกันการแก้ไขสามารถทำได้โดยคลิกเมาส์ไปที่เซลล์นั้นแล้วพิมพ์ค่าที่กล่องข้อความ จะเห็นว่าเซลล์นั้นไม่มีการเปลี่ยนแปลงค่าเลย

ขณะเดียวกันถ้าต้องการยกเลิกการป้องกันการแก้ไขก็สามารถทำได้โดยการคลิกเมาส์ไปที่เซลล์นั้นพร้อมทั้งคลิกเมาส์ที่ Option ของ Unlock เซลล์นั้นก็จะถูกยกเลิกการป้องกันข้อมูล ซึ่งสามารถทดสอบได้ด้วยการคลิกเมาส์ไปที่เซลล์นั้น พร้อมกับพิมพ์ค่าที่ต้องการในกล่องข้อความ จะเห็นว่า ข้อมูลของเซลล์นั้นจะถูกเปลี่ยนเป็นค่าที่พิมพ์จากกล่องข้อความดังรูปที่ 5.3 และ 5.4

สรุปได้ว่า ตัวควบคุมแผ่นตารางทำการ สามารถทำการป้องกันการแก้ไขและยกเลิกการป้องกันการแก้ไขข้อมูลในเซลล์ได้



รูปที่ 5.3 แสดงการยกเลิกการป้องกันแก้ไขข้อมูลในเซลล์



รูปที่ 5.4 แสดงการป้องกันการแก้ไขข้อมูลในเซลล์

4. ทดสอบความสามารถในการแทรกแถวนอน และแถวสดมภ์

กำหนดให้มีการทำงานร่วมกันของ 3 ตัวควบคุมคือ CommandButton TextBar และ Sheet โดยกำหนดคุณสมบัติของแต่ละตัวควบคุมเป็นดังนี้คือ

Name = Command1

Name = Command2

Name = Command3

Name = Sheet1

Name = Text1

Command1.Caption = InsertCol

Command2.Caption = InsertRow

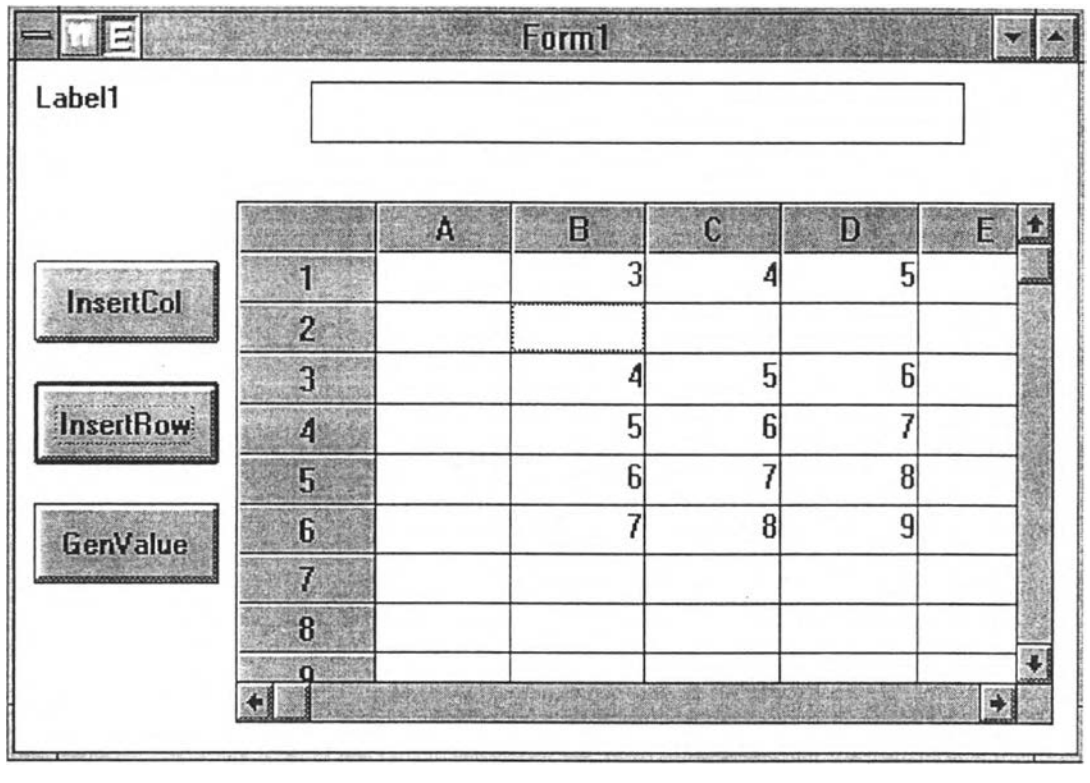
และกำหนดกระบวนงาน Sheet1_KeyDown และ Text1_KeyPress เหมือนกับ กระบวนงานที่ใช้ทดสอบความสามารถในการคัดลอกข้อมูลในเซลล์ นอกจากนั้นยังต้องเพิ่มกระบวนงานดังนี้

```
Sub Command1_Click()
    Sheet1.Action = 2
End Sub

Sub Command2_Click()
    Sheet1.Action = 1
End Sub

Sub Command3_Click()
    Dim i, j
    For i = 1 To 5
        Sheet1.CurrentRow = i
        For j = 1 To 5
            Sheet1.IntValue = i + j
            Sheet1.CurrentCol = j
        Next j
    Next i
```

End Sub

เมื่อทำการสั่งดำเนินงาน และกดปุ่ม Command3 แล้ว ตัวควบคุมแผ่นตารางทำการก็จะแสดงข้อมูลที่อยู่ในเซลล์ โดยที่สามารถทำการแทรกแถวนอนและแถวสดมภ์ได้ด้วยการคลิกเมาส์ที่แถวนอนที่ต้องการแทรกก่อนหน้า แล้วทำการกดปุ่ม InsertRow ตัวควบคุมก็จะทำการแทรกแถวนอนให้ก่อนหน้าแถวนอนที่ระบุ ขณะเดียวกันการแทรกแถวสดมภ์ก็สามารถทำได้ด้วยการคลิกเมาส์ที่แถวสดมภ์ที่ต้องการแทรกก่อนหน้า แล้วกดปุ่ม InsertCol ตัวควบคุมแผ่นตารางทำการก็จะทำการแทรกแถวสดมภ์ก่อนหน้าแถวสดมภ์ที่ระบุได้ ดังรูปที่ 5.5

สรุปได้ว่า ตัวควบคุมแผ่นตารางทำการ สามารถทำการแทรกทั้งในแถวนอนและ แถวสดมภ์ได้จริง และสามารถทำงานร่วมกับตัวควบคุมอื่นๆ ได้อย่างถูกต้องตามกระบวนงานและคุณสมบัติที่กำหนดขึ้น



รูปที่ 5.5 แสดงการแทรกแถวนอนและแถวสดมภ์

5) ทดสอบการลบแถวนอนและแถวสดมภ์

กำหนดให้มีการทำงานร่วมกันของ 3 ตัวควบคุมคือ Command Button Text Box

และ Sheet โดยกำหนดคุณสมบัติของแต่ละตัวควบคุมเป็นดังนี้คือ

Name = Command1

Name = Command2

Name = Command3

Name = Sheet1

Name = Text1

Command1.Caption = DeleteCol

Command2.Caption = DeleteRow

และกำหนดกระบวนงาน Sheet1_KeyDown และ Text1_KeyPress และ Command3_Click เหมือนกับกระบวนงานที่ใช้ทดสอบความสามารถของการแทรกแถวนอนและแถวสดมภ์ได้ และทำการเพิ่มกระบวนงานดังนี้คือ

```
Sub Command1_Click()
    Sheet1.Action = 4
End Sub

Sub Command2_Click()
    Sheet1.Action = 3
End Sub
```

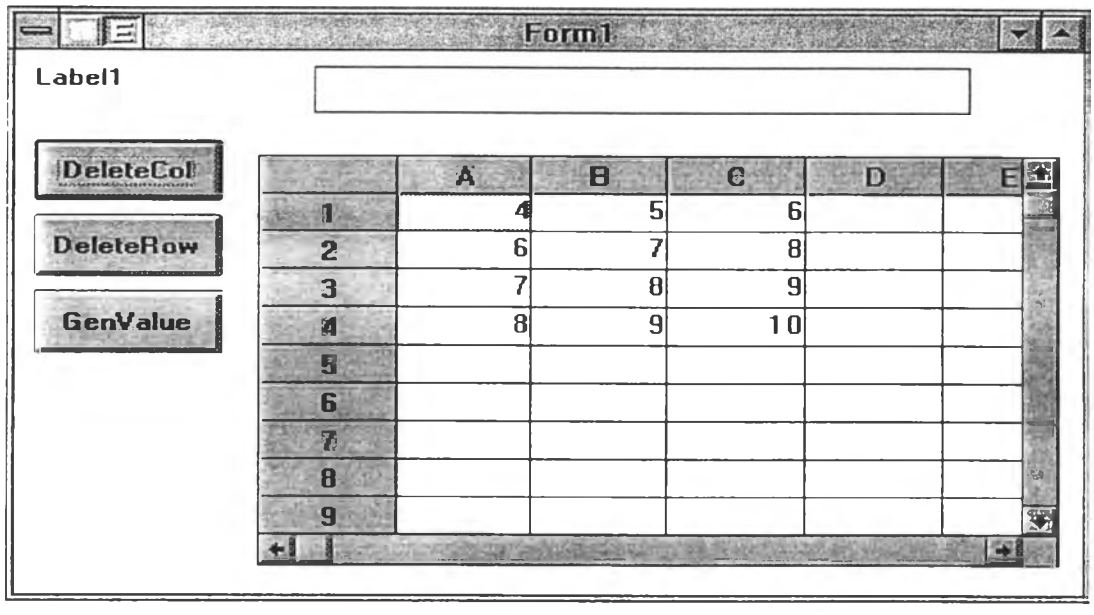
เมื่อสั่งดำเนินงานแล้วกดที่ปุ่ม Command3 แล้ว ตัวควบคุมแผ่นตารางทำการก็จะแสดงข้อมูลที่อยู่ในเซลล์นั้นๆ วิธีการทดสอบการลบแถวนอน สามารถทำได้ด้วยการคลิกเมาส์ที่แถวนอนที่ต้องการลบ และกดปุ่ม DeleteRow จะเห็นว่าแถวนอนที่ระบุจะถูกลบออกไป ขณะเดียวกัน สามารถทำการลบแถวสดมภ์ได้ด้วยการคลิกเมาส์ที่แถวสดมภ์ที่ตองการลบ และกดปุ่ม DeleteCol ดังรูปที่ 5.6

สรุปได้ว่า ตัวควบคุมแผ่นตารางทำการ สามารถทำการลบแถวนอนและแถวสดมภ์ได้จริง และสามารถทำงานร่วมกับตัวควบคุมอื่นๆ ได้อย่างถูกต้องตามกระบวนงานและคุณสมบัติที่กำหนดขึ้น

6) ทดสอบการเรียงลำดับข้อมูลในแถวนอนและแถวสดมภ์

กำหนดให้มีการทำงานร่วมกันของ 4 ตัวควบคุมคือ CommandButton OptionButton

TextBox และ Sheet โดยกำหนดคุณสมบัติของแต่ละตัวควบคุมเป็นดังนี้คือ



รูปที่ 5.6 แสดงการลบแถวนอนและแถวสดมภ์

Name = Command1

Name = Command2

Name = Command3

Name = Option1

Name = Option2

Name = Sheet1

Command1.Caption = SortCol

Command2.Caption = SortRow

Option1.Caption = Ascending

Option2.Caption = Descending

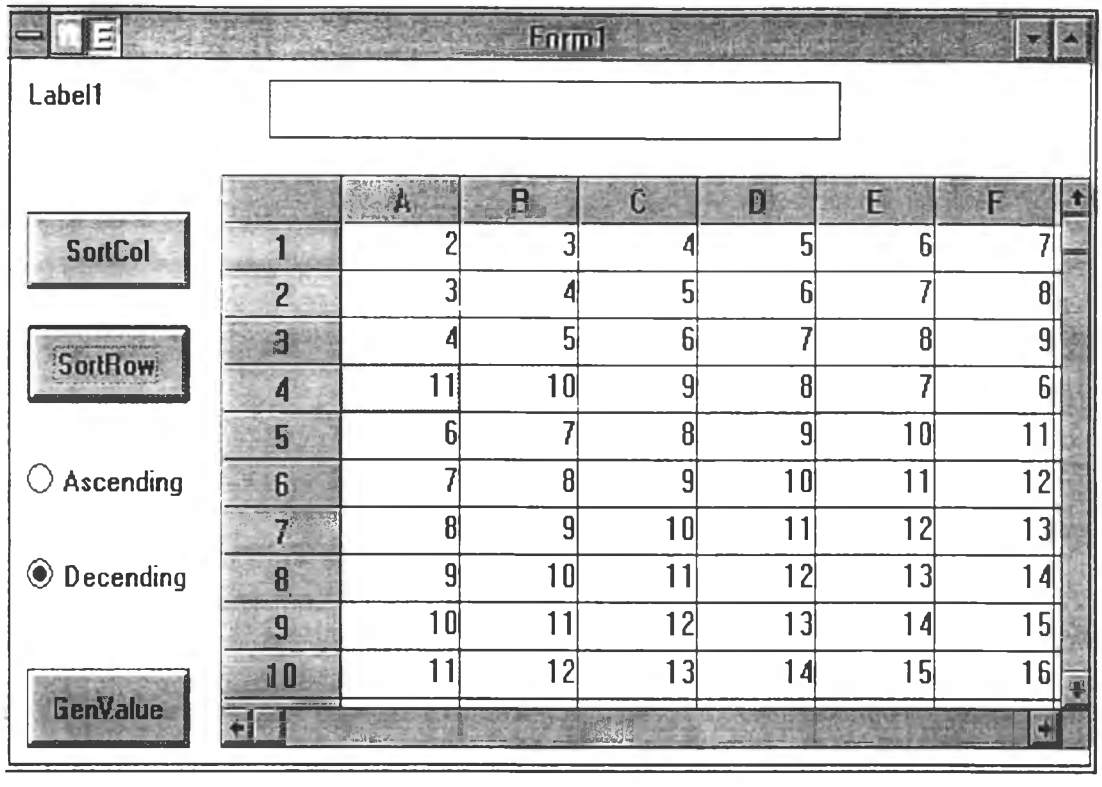
และกำหนดกระบวนงาน Sheet1_KeyDown และ Text1_KeyPress และ Command3_Click เหมือนกับกระบวนงานที่ใช้ทดสอบความสามารถของการแทรกแถวนอนและแถวสดมภ์ได้ และทำการเพิ่มกระบวนงานดังนี้คือ

```
Sub Command1_Click()
    Sheet1.SelStartCol = 1
```

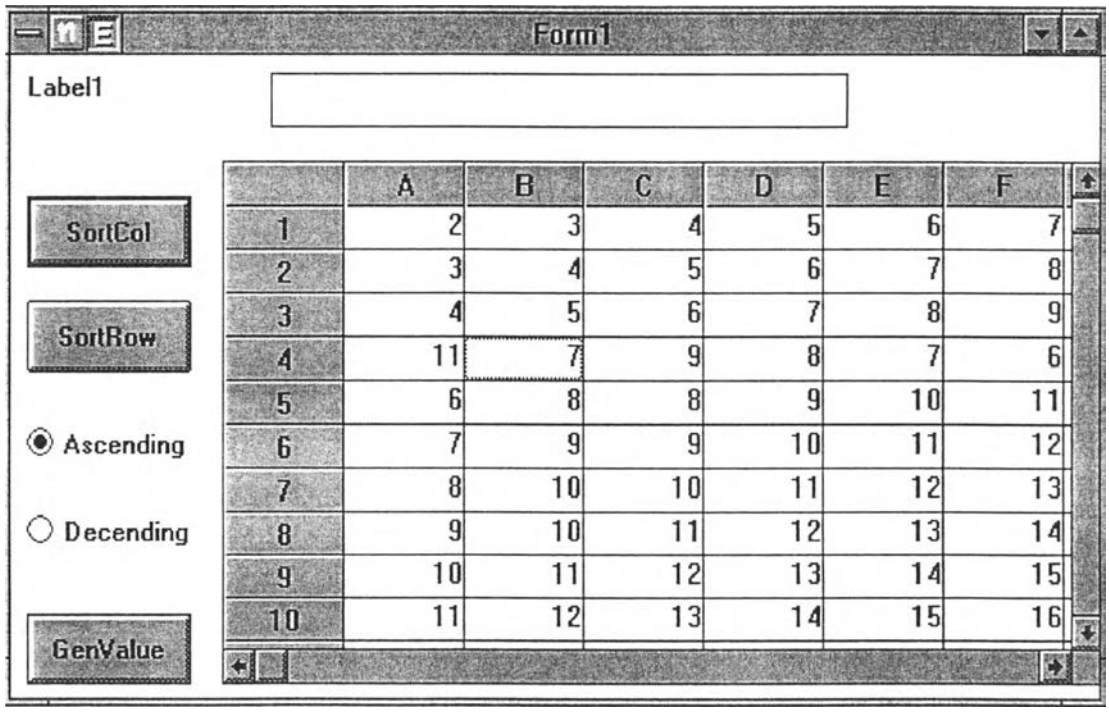
```
    Sheet1.SelEndCol   = 7
    Sheet1.Action      = 1
End Sub

Sub Command2_Click()
    Sheet1.SelStartRow = 1
    Sheet1.SelEndRow   = 10
    Sheet1.Action      = 7
End Sub

Sub Option1_Click()
    Sheet1.SortOrder = 0
End Sub

Sub Option2_Click()
    Sheet1.SortOrder = 1
End Sub
```

เมื่อสั่งดำเนินงานแล้วและกดปุ่ม Command3 แล้ว ตัวควบคุมแผ่นตารางทำการก็จะแสดงข้อมูลที่อยู่ในเซลล์นั้นๆ การทดสอบการเรียงลำดับข้อมูลแถวนอนและแถวสดมภ์นั้น สามารถทำได้ด้วยการคลิกเมาส์ที่แถวนอนหรือแถวนอนที่ต้องการเรียงลำดับ พร้อมทั้งกดปุ่ม SortCol ในกรณีที่ต้องการเรียงลำดับแถวสดมภ์หรือ SortRow กรณีที่ต้องการเรียงลำดับแถวนอน ซึ่งค่าโดยปริยายจะเป็นการเรียงลำดับขึ้น ตัวควบคุมแผ่นตารางทำการก็จะทำการจัดเรียงลำดับข้อมูลในแถวนอนหรือแถวสดมภ์ที่ต้องการ ขณะเดียวกันก็สามารถเปลี่ยนลำดับการเรียงให้เป็นแบบลงก็ได้ด้วยการกด Option ของ Descending ดังรูป 5.7 และ 5.8

สรุปได้ว่า ตัวควบคุมแผ่นตารางทำการสามารถทำการเรียงลำดับข้อมูลในแถวนอนและแถวสดมภ์ได้จริง และสามารถทำงานร่วมกับตัวควบคุมอื่นๆ ได้อย่างถูกต้องตามกระบวนงานและคุณสมบัติที่กำหนดขึ้น



รูปที่ 5.7 แสดงการเรียงลำดับแถวนอนแบบลง



รูปที่ 5.8 แสดงการเรียงลำดับแถวสดมภ์แบบขึ้น

7) ทดสอบความสามารถในการคำนวณทางคณิตศาสตร์

กำหนดให้มีการทำงานร่วมกันของ 3 ตัวควบคุมคือ CommandButton TextBox และ Sheet โดยกำหนดคุณสมบัติของแต่ละตัวควบคุมเป็นดังนี้คือ

Name = Command1

Name = Text1

Name = Text2

Name = Text3

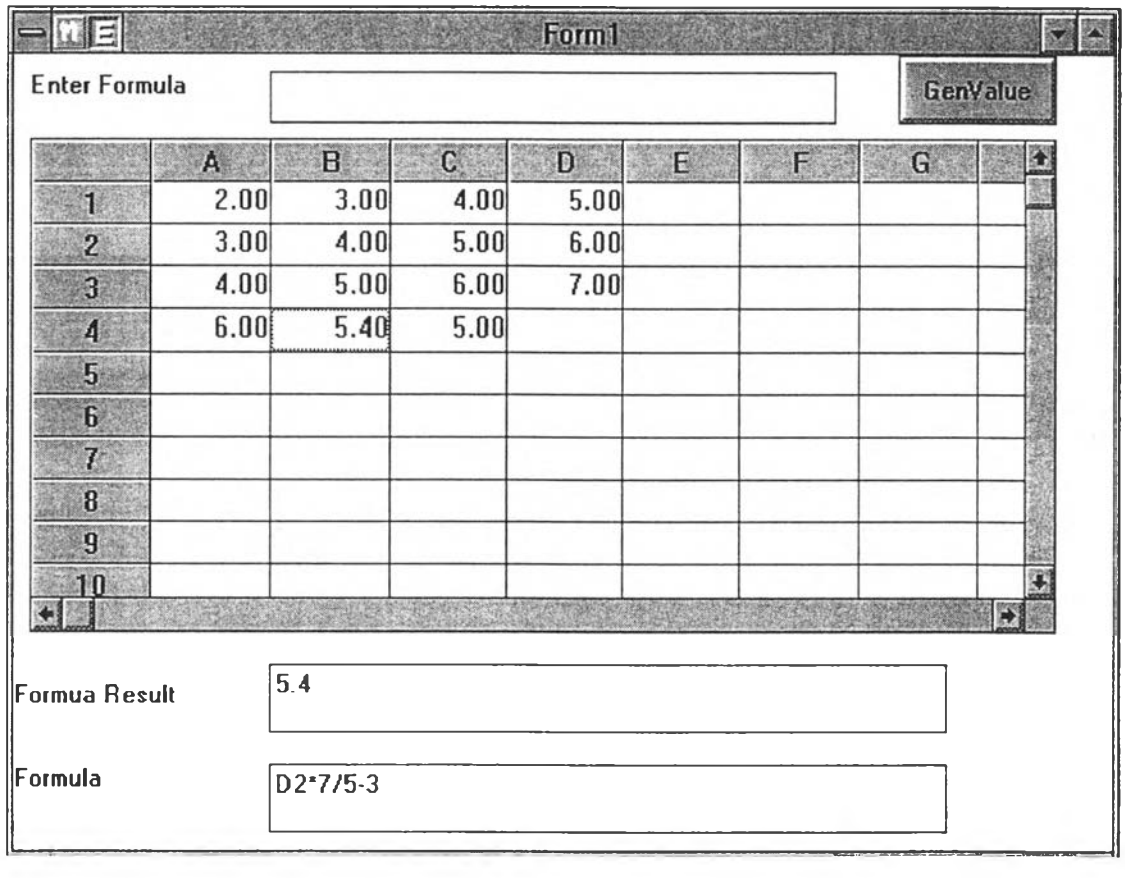
Name = Sheet1

กำหนดกระบวนงาน Sheet1_KeyDown เหมือนกับกระบวนงานที่ใช้ทดสอบความสามารถของการแทรกแถวนอนและแถวสดมภ์ได้ และกำหนดกระบวนงานเพิ่มดังนี้คือ

```
Sub Text1_KeyPress (KeyAscii As Integer)
    If KeyAscii = 13 Then
     Sheet1.Formula = Text1.text
     KeyAscii       = 0
    End If
End Sub

Sub Command1_Click ( )
    Dim i, j
    For i = 1 To 3
      Sheet1.CurrentRow = i
      For j = 1 To 4
        Sheet1.CurrentCol = j
        Sheet1.FloatValue = i+j
      Next j
    Next i
End Sub

Sub Text2_Click ( )
    Text2.text = Sheet1.Formula
```

```
End Sub

Sub Text3_Click ( )
    Text3.text = Sheet1.FResult
End Sub
```

เมื่อสั่งดำเนินงานแล้วและกดปุ่ม Command1 โปรแกรมจะทำการกำหนดค่าลงในเซลล์ข้อมูล ตั้งแต่เซลล์ A1 ถึงเซลล์ D3 ถ้าต้องการกำหนดสูตรการคำนวณของเซลล์สามารถทำได้โดยคลิกเมาส์ที่เซลล์ผลลัพธ์ จากนั้นให้พิมพ์สูตรลงในกล่องข้อความ ตัวควบคุมแผ่นตารางทำการจะทำการคำนวณค่าจากสูตรที่ระบุ พร้อมทั้งนำค่าผลลัพธ์ไปแสดงในเซลล์ผลลัพธ์ที่กำหนด กรณีที่ต้องการทราบว่าเซลล์ที่ระบุมาจากสูตรการคำนวณหรือไม่ สามารถทราบได้โดยการคลิกเมาส์ที่ Text2 ตัวควบคุมแผ่นตารางทำการก็จะแสดงสูตรการคำนวณของเซลล์ ดังรูปที่ 5.9



รูปที่ 5.9 แสดงการคำนวณทางคณิตศาสตร์

สรุปได้ว่า ตัวควบคุมแผ่นคารางทำการ สามารถทำการคำนวณสูตรทางคณิตศาสตร์ได้จริง และสามารถทำงานร่วมกับตัวควบคุมอื่น ๆ ได้อย่างถูกต้องตามกระบวนงานและคุณสมบัติที่กำหนด