การพัฒนาโปรแกรมช่วยการเขียนย่อหน้าแสดงความสัมพันธ์แบบเหตุและผล

รองศาสตราจารย์ผ่าน บาลโพธิ์

ความนำ

ปัญหาที่สำคัญประการหนึ่งของการสอนเขียนข้อความระดับย่อหน้า (paragraph) คือการตรวจงานเขียน เพราะนอกจากผู้สอนส่วนใหญ่จะมีภาระงานสอน และงานอื่นๆ มากอยู่แล้ว ผู้เรียนแต่ละชั้นหรือแต่ละตอนเรียนมักมีจำนวนมากด้วยเช่นกัน จึงทำให้ผู้สอนต้องใช้เวลามากในการตรวจงาน และไม่สามารถมอบหมายงานให้ผู้เรียนได้ฝึกเขียนมากเท่าที่ต้องการ นอกจากนี้ ดังเป็นที่ทราบกันดี งานเขียนจำนวนมากมีทั้งข้อผิดพลาดด้านไวยากรณ์ การใช้รูปประโยคและคำที่เหมาะสมกับสัมพันธสาร (discourse) เช่น เมื่อเขียนย่อหน้าแสดงความสัมพันธ์แบบเหตุและผล (cause-effect) ก็ต้องใช้รูปประโยคและคำที่ใช้แสดงความสัมพันธ์แบบเหตุและผล (cause-effect relationship) รวมทั้งการใช้คำบ่งชี้ (marker) ที่ถูกต้องเหมาะสมด้วย และที่สำคัญคือข้อผิดพลาดเหล่านี้มักเกิดซ้ำๆ ซึ่งอาจเกิดจากหลายสาเหตุ แต่ไม่ว่าจะเกิดจากสาเหตุใดก็ตาม ผู้สอนมักต้องตรวจข้อผิดพลาดเดิมๆ ของผู้เรียนอยู่เสมอ ผลก็คือผู้เรียนจะได้รับคืนงานเขียนที่เต็มไปด้วยการแก้ข้อผิดพลาด ซึ่งอาจทำให้ผู้เรียนหมดกำลังใจที่จะพัฒนาการเขียน สำหรับผู้สอนเองการตรวจข้อผิดพลาดเดิมๆ ของผู้เรียนอยู่เสมอนั้น ก็ไม่ใช่กิจกรรมที่น่าสนใจและท้าทายความสามารถแต่อย่างใด

รองศาสตราจารย์ผ่าน บาลโพธิ์

เป็นผู้ประสานงานและเป็นผู้สอนรายวิชา Technical English for Medical Science Graduates ซึ่งเน้นการอ่านและการเขียนบทความวิจัย อาจารย์ผ่านสนใจทั้งด้านการอ่าน การเขียน การวิเคราะห์ภาษา และการเขียนโปรแกรมคอมพิวเตอร์ช่วยการเรียนภาษา (CALL) และเป็นวิทยากรรับเชิญบรรยายการเขียนบทความวิจัยทางการแพทย์จากหลายสถาบัน

ดังนั้น หากมีโปรแกรมคอมพิวเตอร์ที่สามารถตรวจสอบด้านไวยากรณ์ การใช้รูปประโยคและคำที่เหมาะสมกับการเสนอเนื้อหาที่แสดงความสัมพันธ์แบบเหตุและผล การใช้คำบ่งชี้ ตลอดจนความซ้ำซ้อนของการใช้คำ และรูปประโยคในย่อหน้า และผู้เรียนสามารถคลิกเมาส์ขอความช่วยเหลือได้ตลอดเวลาในขณะฝึกเขียนกับโปรแกรม โปรแกรมดังกล่าวนี้น่าจะช่วยให้ผู้เรียนรู้สึกมั่นใจว่า การฝึกเขียนจะประสบผลสำเร็จ และท้ายที่สุด เมื่อผู้เรียนเขียนครบทุกส่วนของย่อหน้า และโปรแกรมตรวจรับว่าถูกต้องแล้ว ผู้เรียนสามารถพิมพ์ผลงานการเขียนย่อหน้าออกทางเครื่องพิมพ์ การได้เห็นผลงานของตนเองอาจช่วยให้ผู้เรียนรู้สึกว่าตนเองประสบความสำเร็จในการเขียนและอยากพัฒนาการเขียนของตนให้ดียิ่งขึ้น

อย่างไรก็ตาม หลายท่านอาจสงสัยว่าโปรแกรมคอมพิวเตอร์จะสามารถตรวจงานเขียนได้อย่างไร เพราะคอมพิวเตอร์ไม่สามารถคิดได้เหมือนผู้สอนที่เป็นมนุษย์ ดังนั้น บทความนี้จึงเสนอแนวคิด วิธีการนำแนวคิดมาสู่การปฏิบัติ ตลอดจนปัญหาในการปฏิบัติ การแก้ไขปัญหา จากประสบการณ์การพัฒนาโปรแกรมช่วยการเขียนย่อหน้าเรื่อง Writing a Cause-Effect Paragraph ที่ผู้เขียนพัฒนาขึ้น ดังมีเนื้อหาโดยสังเขปต่อไปนี้

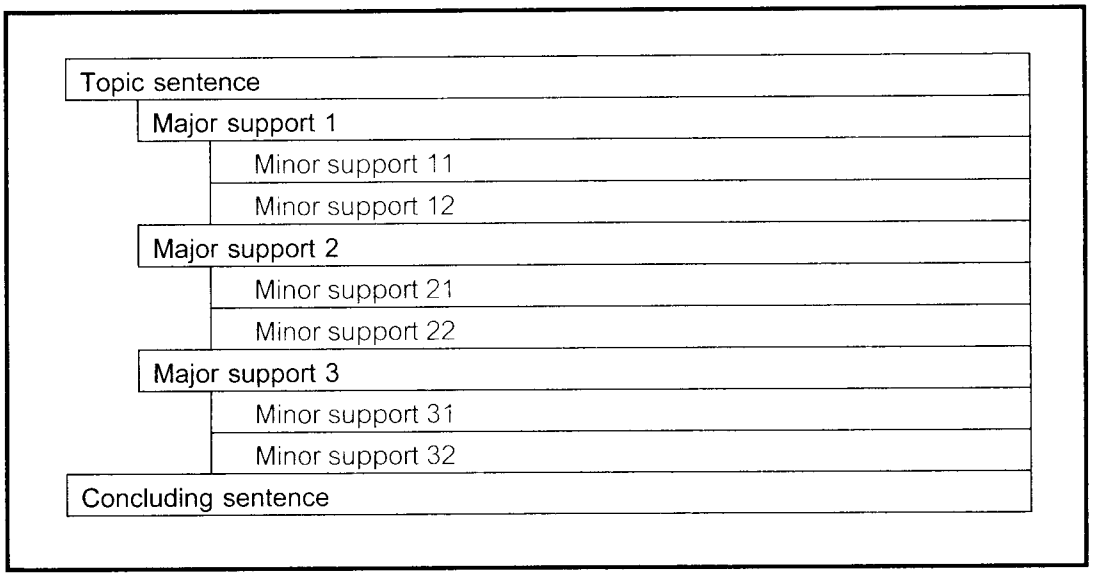
แนวคิดและความเป็นไปได้

เนื่องจากย่อหน้าประกอบด้วยประโยคหลายประโยค ดังนั้น แม้เราสามารถเขียนโปรแกรมให้ผู้เรียนเขียนย่อหน้าได้ แต่การตรวจ และการให้ข้อมูลป้อนกลับจะเป็นปัญหาสำคัญ เพราะการตรวจย่อหน้านั้นตรวจได้ยาก เนื่องจากจะมีข้อผิดพลาดทั้งด้านเนื้อหาและไวยากรณ์ รวมทั้งการใช้รูปประโยค คำบ่งชี้ และความเหมาะสมของการใช้ถ้อยคำ นอกจากนี้ ในกรณีที่มีข้อผิดพลาดจำนวนมาก การให้คำแนะนำสำหรับการแก้ไขข้อผิดพลาดในแต่ละประเด็นจะทำได้ยาก คำแนะนำส่วนใหญ่อาจเป็นเพียงคำแนะนำในภาพรวม จะไม่สามารถแนะนำการแก้ไขได้ตรงประเด็นและเป็นรูปธรรม ดังที่พบเห็นในโปรแกรมการเขียนทั่วไป เช่น อาจแนะนำว่าประโยคยาวเกินไป ใช้ประโยค passive voice บ่อยเกินไป ใช้คำยากหรือง่ายเกินไป เป็นต้น

แนวคิดเบื้องต้นจึงเป็นแนวคิดที่มุ่งแก้ไขปัญหาการตรวจและการแนะนำว่าทำอย่างไรจึงจะทำให้โปรแกรมที่พัฒนาขึ้นสามารถตรวจและให้คำแนะนำแต่ละส่วนของย่อหน้าได้อย่างชัดเจนและตรงประเด็น นอกจากนี้ เนื่องจากโปรแกรมที่พัฒนาขึ้นเป็นโปรแกรมที่ให้ผู้เรียนฝึกเขียนย่อหน้าแบบเหตุและผล (cause-effect) ด้วยตนเอง โปรแกรมจึงต้องสามารถช่วยให้ผู้เรียนเขียนย่อหน้าได้สำเร็จ มีความน่าสนใจ และมีลักษณะที่ดีในหลายประการ

เมื่อพิจารณาข้อมูลเบื้องต้นเกี่ยวกับโปรแกรมที่จะพัฒนา จะเห็นได้ว่าผู้เรียนจะเขียนย่อหน้าจากข้อมูลที่กำหนดสำหรับทุกส่วนของย่อหน้า ดังนั้น การพัฒนาโปรแกรมนี้จึงง่ายขึ้น เพราะเราสามารถแบ่งการเขียนออกเป็นส่วนๆ และตรวจเป็นส่วนๆ ด้วยเช่นกัน แนวคิดนี้เป็นไปตามแนวคิดหลักที่ว่า เมื่อมีงานใหญ่ (main task) ควรแบ่งงานใหญ่นั้นออกเป็นงานย่อย (sub-task) ที่สามารถบริหารจัดการได้ ซึ่งงานใหญ่ในที่นี้คือการเขียนย่อหน้าทั้งย่อหน้า ดังนั้น ในขั้นต้นจึงควรแบ่งย่อหน้าออกเป็นส่วนๆ (parts/moves) เพื่อที่ผู้สอนจะสามารถตรวจ และให้คำแนะนำเพื่อแก้ไขข้อผิดพลาดแก่ผู้เรียนได้อย่างถูกต้องและตรงประเด็น งานย่อยในที่นี้คือการเขียนส่วนต่างๆ ของย่อหน้า ซึ่งได้แก่การเขียน topic sentence การเขียน major support และ concluding sentence สำหรับการเขียน major support นั้น โดยทั่วไปจะมีหลาย major supports ซึ่งอาจแบ่งออกเป็น major support 1, 2 และ 3 และในแต่ละ major support อาจมี minor supports สนับสนุน ดังตัวอย่างในรูปต่อไปนี้

รูปที่ 1: ส่วนประกอบของย่อหน้า



แนวคิดที่สำคัญอีกประการหนึ่งคือ โปรแกรมที่จะพัฒนาขึ้นควรมีคุณลักษณะทั่วไปของโปรแกรมสำหรับการเรียนการสอนภาษา โดยเฉพาะคุณลักษณะต่อไปนี้

1. โปรแกรมมีจุดมุ่งหมายที่ชัดเจน
2. หน้าจอและส่วนประกอบของหน้าจอควรเหมาะสมกับการใช้โปรแกรม
3. โปรแกรมควรให้ความรู้ทั่วไป และความรู้เฉพาะสำหรับแบบฝึกหัดที่ผู้เรียนต้องเขียน
4. โปรแกรมสามารถรับประโยคคำตอบที่ถูกต้อง และเป็นไปได้ (possible sentences) ได้จำนวนมาก
5. โปรแกรมสามารถอธิบายและแนะนำวิธีแก้ไขข้อผิดพลาดด้านไวยากรณ์ได้ถูกต้องและตรงประเด็น และผู้เรียนสามารถนำข้อแนะนำนั้นไปแก้ไขข้อผิดพลาดได้จริง
6. โปรแกรมเปิดโอกาสให้ผู้เรียนเขียนประโยคสำหรับแต่ละส่วนของย่อหน้าโดยใช้ศัพท์และรูปประโยคได้หลากหลาย
7. ผู้เรียนสามารถเขียนส่วนใดของย่อหน้าก่อนก็ได้โดยไม่ต้องเขียนเรียงแต่ละส่วนตามลำดับ
8. ผู้เรียนสามารถควบคุมโปรแกรมได้ในระดับที่เหมาะสม
9. โปรแกรมสามารถตรวจความซ้ำซ้อนของการใช้รูปประโยค และคำ
10. โปรแกรมสามารถประเมินผลการฝึกเขียนย่อหน้าในภาพรวม
11. ความยากง่ายของแบบฝึกหัดควรเหมาะสมกับความสามารถของผู้เรียน
12. โปรแกรมช่วยให้ผู้เรียนประสบผลสำเร็จในการเขียน

จากที่กล่าวมานี้ จะเห็นได้ว่าเราสามารถพัฒนาโปรแกรมช่วยการเขียนย่อหน้าได้ โดยการให้ผู้เรียนพิมพ์ประโยคสำหรับแต่ละส่วนของย่อหน้าลงไปในกรอบรับข้อความ (textbox) และเมื่อพิมพ์เสร็จแล้วให้ผู้เรียนคลิกปุ่มคำสั่งให้โปรแกรมตรวจ และเสนอแนะการแก้ไข การทำเช่นนี้โปรแกรมจะสามารถตรวจรายละเอียดต่างๆ ของประโยคได้ตามที่ผู้สอน

และ/หรือ ผู้เขียนโปรแกรมต้องการ ผู้สอน และ/หรือ ผู้เขียนโปรแกรมสามารถกำหนดเงื่อนไขได้ว่า ประโยคใดถูกและประโยคใดผิด ผิดด้วยสาเหตุใด ควรแก้ไขข้อผิดพลาดนั้นอย่างไร นอกจากนี้ โปรแกรมควรสามารถแจ้งความถี่ของข้อผิดพลาดแต่ละประเด็น เพราะคอมพิวเตอร์สามารถจำและบันทึกข้อมูลต่างๆ ได้ เมื่อสามารถระบุข้อผิดพลาดได้ชัดเจนเช่นนี้ เราจึงสามารถคิดคะแนนการฝึกเขียนของผู้เรียนได้ และสามารถใช้คะแนนนั้นสำหรับการประเมินผลการฝึกปฏิบัติในภาพรวม

ด้วยแนวคิดที่กล่าวมาแล้ว ทำให้เห็นภาพของโปรแกรมนี้ว่าเป็นโปรแกรมสอนและฝึกเขียนย่อหน้า โดยผู้เรียนต้องเขียนส่วนต่างๆ (parts/moves) ของย่อหน้า และป้อนสิ่งที่เขียนเข้าไปในโปรแกรม ซึ่งประกอบด้วยส่วนสำคัญต่างๆ หลายส่วน คือส่วนสำหรับเขียนและตรวจ topic sentence ส่วน major support 1 และส่วน minor support(s) สำหรับ major support 1 จากนั้นเป็นส่วนสำหรับ major และ minor support(s) อื่นๆ และส่วนสุดท้ายเป็นส่วนสำหรับ concluding sentence

เมื่อมีแนวคิดที่ชัดเจน และมีแนวทางที่จะสามารถพัฒนาโปรแกรมนี้ได้ ขั้นต่อไปจึงเป็นขั้นตอนการดำเนินการ

การออกแบบหน้าจอ

โดยทั่วไปหน้าจอจะประกอบด้วยหลายส่วน แต่ละส่วนมีหน้าที่แตกต่างกัน และในการออกแบบและสร้างหน้าจอ เราต้องใช้เครื่องมือที่มีในกล่องเครื่องมือ (toolbox) ของโปรแกรมภาษาคอมพิวเตอร์ (computer language) ที่ใช้สำหรับเขียนโปรแกรม เช่นโปรแกรมภาษา Visual Basic และโปรแกรมภาษา Visual C $^{++}$

1. ส่วนประกอบสำคัญของหน้าจอ

หลักทั่วไปในการออกแบบหน้าจอโปรแกรมช่วยการเรียนภาษา (computer-assisted language learning) คือต้องออกแบบให้เหมาะกับการใช้งานโปรแกรมที่พัฒนาขึ้น และช่วยให้ผู้เรียนใช้โปรแกรมได้สะดวก ซึ่งส่วนใหญ่แล้วหน้าจอจะมีส่วนแสดงข้อความหรือข้อมูล ส่วนรับข้อมูล ส่วนแจ้งผลการตรวจ และให้คำแนะนำ ดังตัวอย่างในรูปที่ 2 และรูปที่ 5 ซึ่งเป็นหน้าจอของโปรแกรม Writing a Cause-Effect Paragraph ที่ผู้เขียนพัฒนาขึ้น

ส่วนประกอบสำคัญของหน้าจอโปรแกรม Writing a Cause-Effect Paragraph มีดังนี้

Part 1: Instructions เป็นส่วนแสดงคำสั่งการเขียนย่อหน้า

Part 2: Advisor/Help เป็นส่วนประกอบที่เป็น "ตัวช่วย" สำหรับให้ข้อมูลความรู้ที่จะช่วยให้ผู้เรียนสามารถเขียนย่อหน้าได้สำเร็จ

Part 3: Information/Data เป็นส่วนแสดงข้อมูลสำหรับใช้เขียนย่อหน้า

Part 4: Your draft/Your complete sentence เป็น ส่วนรับคำตอบ

Part 5: Results/Feedback เป็นส่วนแสดงข้อมูลป้อนกลับ และคำแนะนำ

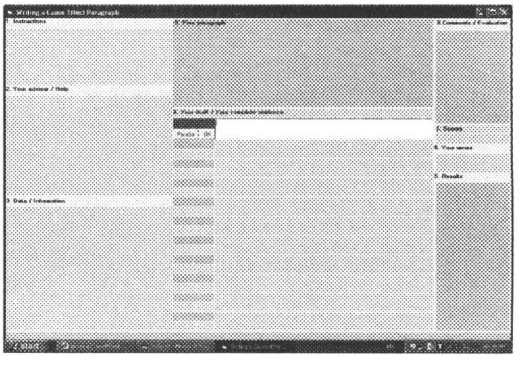
Part 6: Your errors เป็นส่วนที่แสดงความถี่ของข้อผิดพลาดด้านไวยากรณ์

Part 7: Scores เป็นส่วนแจ้งคะแนน

Part 8: Comments/Evaluation เป็นส่วนประเมิน ผลการฝึกเขียนในภาพรวม

Part 9: Your paragraph เป็นส่วนแสดงย่อหน้าที่สมบูรณ์แล้ว

รูปที่ 2: **ภาพร่างของหน้าจอ**

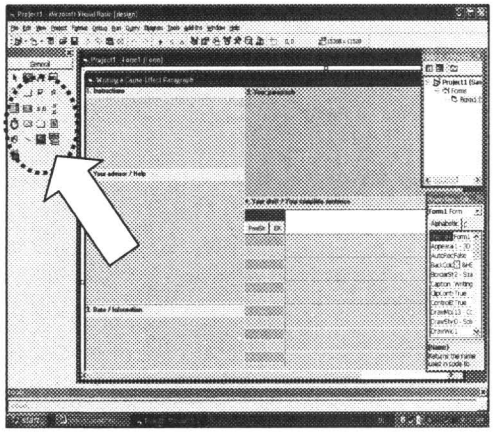


นอกจากส่วนประกอบหลักดังกล่าวแล้ว ยังต้องมีส่วนประกอบย่อยที่ให้คำแนะนำเฉพาะเพื่อใช้เขียนส่วนต่างๆ ของย่อหน้าที่เป็นแบบฝึกหัด ส่วนที่ให้ผู้เรียนคลิกเมาส์เพื่อให้โปรแกรมตรวจคำตอบ (เช่น ปุ่มคำสั่ง OK) ส่วนสำหรับขยายขนาดของข้อความในย่อหน้าให้มีขนาดใหญ่ขึ้น และส่วนสำหรับสั่งพิมพ์ย่อหน้าที่เขียนเรียบร้อยแล้ว

2. เครื่องมือสำหรับสร้างหน้าจอ

หลังจากได้พิจารณาส่วนประกอบต่างๆ ของหน้าจอแล้ว ขั้นต่อไปเป็นการเลือกเครื่องมือที่จะใช้สร้างหน้าจอ เครื่องมือดังกล่าวนี้เรียกว่าวัตถุ (Control object/Object) ซึ่งแท้จริงก็คือโปรแกรมขนาดเล็กที่โปรแกรมภาษาคอมพิวเตอร์ (เช่น Visual Basic) เตรียมไว้ให้ใช้ วัตถุที่ใช้อยู่เสมอได้แก่ กรอบข้อความ (Label) กรอบรับข้อความ (Textbox) กรอบรูปภาพ (Picture) และที่สำหรับลากเส้น (Line) และปุ่มคำสั่ง (Command button) เครื่องมือเหล่านี้อยู่ในกล่องเครื่องมือ (toolbox) ซึ่งอยู่ด้านซ้ายของโปรแกรมภาษาคอมพิวเตอร์ที่นำมาใช้เขียนโปรแกรม (ดังตัวอย่างในรูปที่ 3)

รูปที่ 3: **กล่องเครื่องมือ**



3. ขั้นตอนการสร้างหน้าจอ

ขั้นตอนสำคัญในการสร้างหรือตกแต่งหน้าจอมีดังนี้

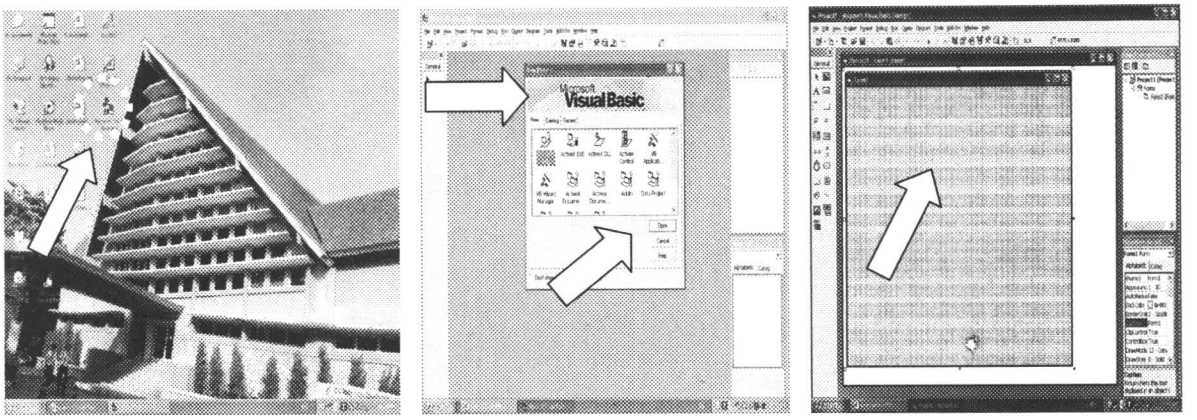
1. เปิดเข้าสู่โปรแกรมภาษาคอมพิวเตอร์ที่ท่านใช้สำหรับพัฒนาโปรแกรม โดยการคลิกเมาส์ที่ icon ของโปรแกรมซึ่งอยู่ที่ Desktop (รูปแรกของรูปที่ 4)

2. เลือกสร้างโปรแกรมใหม่ (New) (รูปที่สองของรูปที่ 4)

3. เมื่อคลิกปุ่มคำสั่งเปิดโปรแกรมแล้วจะมี Form ซึ่งเป็นที่ว่างสำหรับให้ผู้เขียนโปรแกรมนำสิ่งต่างๆ ที่เรียกว่าวัตถุ (Object) มาวาดลงบน Form และจัดวางวัตถุต่างๆ ที่นำมาใช้บนหน้าจอให้เหมาะสมกับการใช้งาน และสอดคล้องกับหลักการจัดองค์ประกอบ (composition) ทางศิลปะ (รูปที่สามของรูปที่ 4)

การวาดวัตถุต่างๆ ลงบน Form คล้ายกับวิธีการที่ใช้กับโปรแกรม PowerPoint เช่น เมื่อต้องการพิมพ์ข้อความออกหน้าจอจะต้องใช้วัตถุ Label ถ้าจะนำรูปภาพมาวางบนหน้าจอก็ต้องใช้กรอบ Picture และถ้าต้องการลากเส้นก็ต้องใช้วัตถุ Line แต่ความแตกต่างที่สำคัญคือ สำหรับโปรแกรมภาษาคอมพิวเตอร์ (เช่น Visual Basic) นั้น เราสามารถกำหนดรายละเอียดต่างๆ ของวัตถุได้ในกรอบ Properties ไม่ว่าจะเป็นรายละเอียดทางกายภาพ เช่น ขนาด สี หรือตำแหน่งที่ต้องการให้ปรากฏบนหน้าจอ ไปจนถึงรายละเอียดที่สำคัญอื่นๆ เช่น จะให้ปรากฏเมื่อใด จะให้ใช้งานได้หรือจะเพียงปรากฏให้เห็นเท่านั้น ตัวอย่างเช่น ในกรณีของกรอบรับข้อมูล ผู้เขียนโปรแกรมสามารถกำหนดได้ว่าจะให้กรอบนี้ปรากฏบนหน้าจอแต่รับข้อมูลไม่ได้ หรือจะให้รับข้อมูลได้

รูปที่ 4: **การเข้าสู่โปรแกรมภาษา** Visual Basic



4. เมื่อนำวัตถุใดมาใช้ ควรตั้งชื่อวัตถุนั้นใหม่ให้ตรงกับหน้าที่ของวัตถุนั้น ทั้งนี้เพราะเมื่อเราวาดวัตถุ หรือ Object ลงบน Form โปรแกรมภาษาคอมพิวเตอร์ที่เราใช้เขียนโปรแกรมจะตั้งชื่อวัตถุนั้นทันที เช่น เมื่อวาดกรอบรับข้อความ (Textbox) ลงบน Form กรอบรับข้อความแรกจะถูกตั้งชื่อว่า Text1 และเมื่อนำกรอบรับข้อความมาวาดลงบน Form อีก โปรแกรมแม่จะตั้งชื่อให้กรอบรับข้อความใหม่เป็น Text2 Text3 และ Text4 ตามลำดับ ซึ่งไม่สะดวกต่อการเขียนคำสั่งเรียกใช้กรอบรับข้อความเหล่านี้ เพราะเลข 1 2 3 และ 4 ไม่บอกว่าเป็นปุ่มคำสั่งใด ดังนั้น ท่านอาจตั้งชื่อกรอบรับข้อความเหล่านี้ใหม่ เช่น txtTopicSent ซึ่งมีสองส่วน คือ txt ใช้แทน Textbox และ TopicSent บอกให้ทราบว่าเป็นกรอบรับประโยค topic sentence

ตัวย่อในตอนต้นของชื่อ Object นั้น จะเป็นตัวย่อใดก็ได้ หรือจะไม่มีก็ได้ แต่ถ้าไม่มีอาจไม่สะดวกต่อการเรียกใช้ เพราะจะไม่ทราบว่าเป็น Object ใด ตัวย่อที่ใช้โดยทั่วไปมีดังนี้

lbl แทน Label เช่น lblFirstMajor (เนื้อหาสำหรับเขียน Major support 1)

txt แทน Textbox เช่น txtMajorSupp1 (กรอบรับคำตอบ Major support 1)

cmd แทน Command button เช่น cmdMajor1OK (สำหรับสั่งตรวจคำตอบ)

lin แทน Line เช่น linLinkNewForm (สำหรับเรียกใช้หน้าจอใหม่)

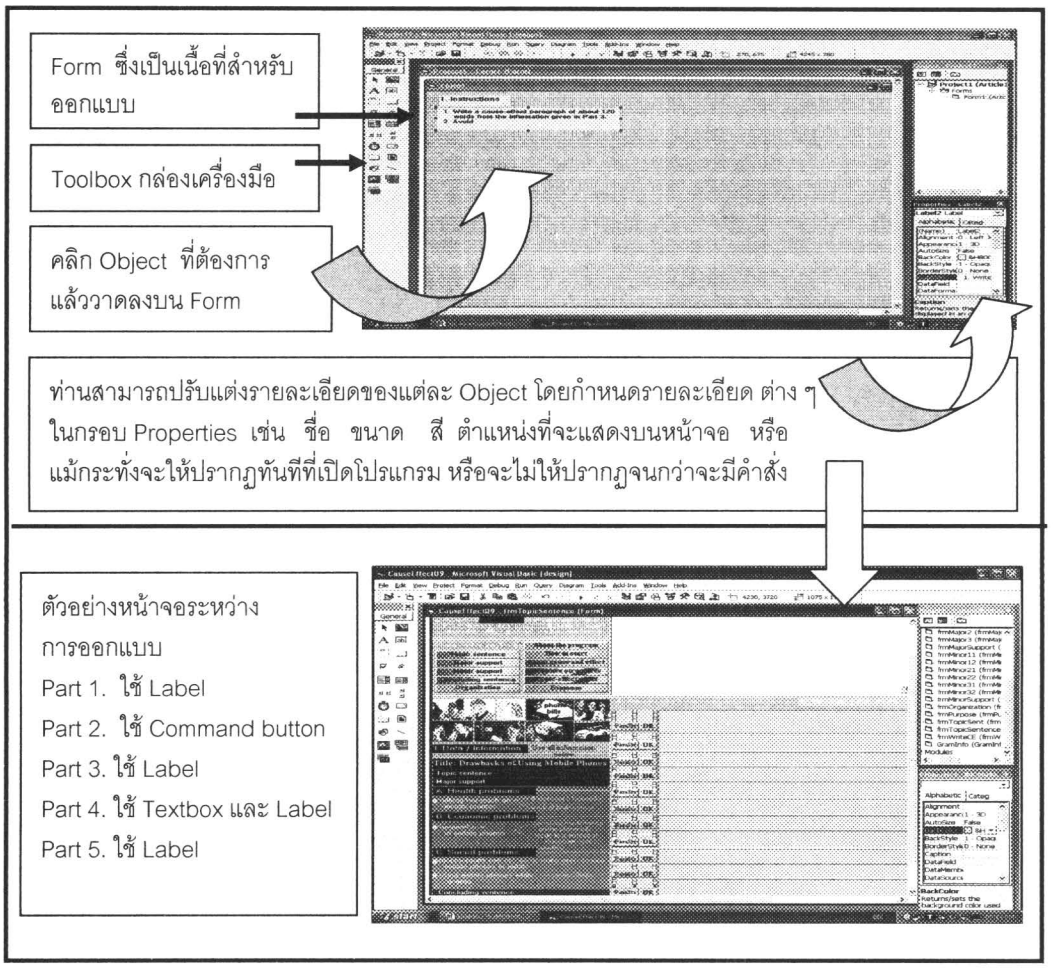
img แทน Image เช่น imgLogo (สำหรับแสดงภาพโลโก้)

วัตถุ	ชื่อเดิม	ตัวอย่างชื่อที่ตั้งขึ้นใหม่	หน้าที่ / งาน
Label	Label10	lblFirstMajor	แสดงข้อมูลที่จะใช้เขียน Major support 1
Textbox	Text9	txtMajorSupp1	กรอบรับประโยคที่ผู้เรียนเขียน
Command button	Command7	cmdSentStructures	แสดงรูปประโยคสำหรับการเขียนแสดงความสัมพันธ์แบบเหตุและผล
Command button	Command8	cmdMajor1OK	สำหรับสั่งให้โปรแกรมตรวจประโยคที่เขียนเสร็จแล้ว
Line	Line5	linLink	สำหรับเรียกใช้หน้าจอใหม่

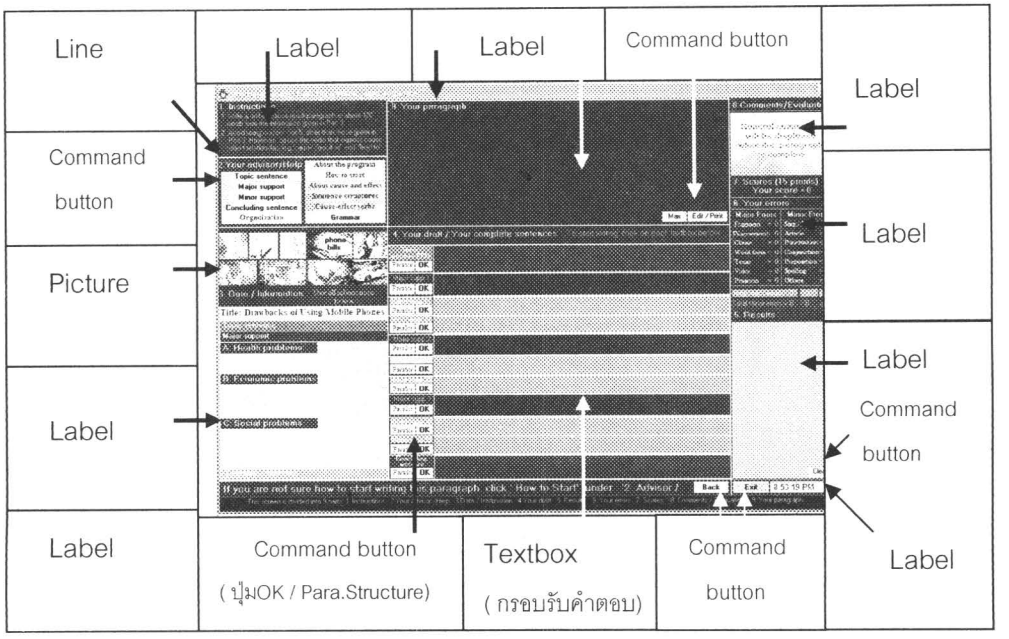
4. ตัวอย่างการสร้างหน้าจอใน Visual Basic

การสร้างหน้าจอของโปรแกรม Writing a Cause-Effect Paragraph ทำได้โดยการนำวัตถุหรือ Object หลายชนิดมาวาดลงบน Form ดังรายละเอียด ในรูปที่ 5, 6 และ 7

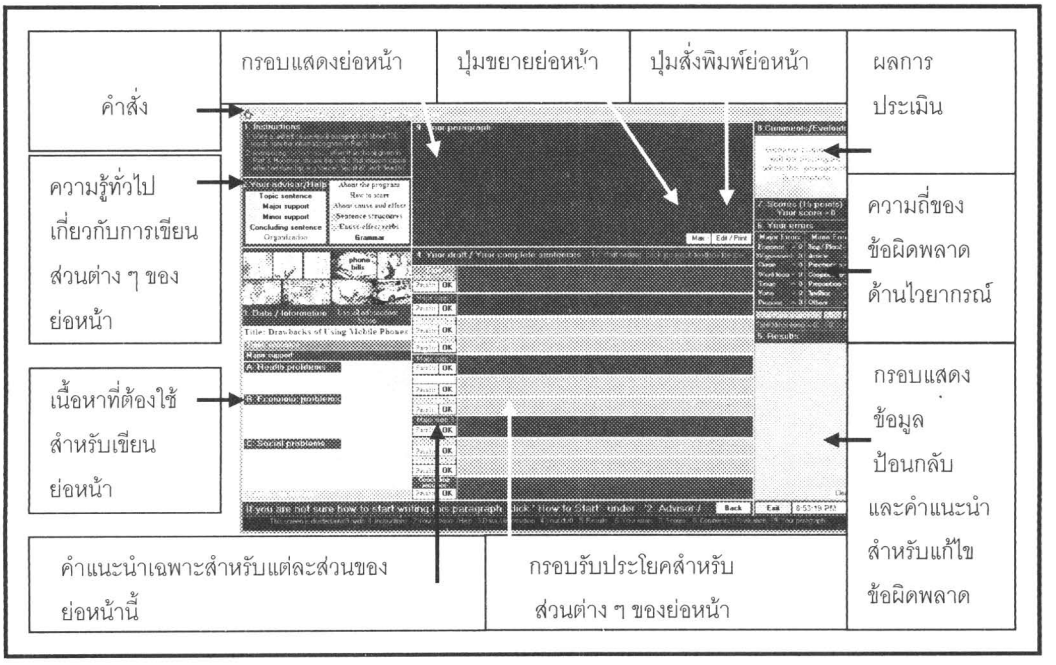
รูปที่ 5: **ตัวอย่างการนำวัตถุ** (Object) **มาใช้ในการออกแบบหน้าจอ**



รูปที่ 6: วัตถุ (Object) ที่ใช้สร้างหน้าจอโปรแกรม Writing a Cause-Effect Paragraph



รูปที่ 7: ส่วนประกอบและหน้าที่ของส่วนประกอบของหน้าจอจริง



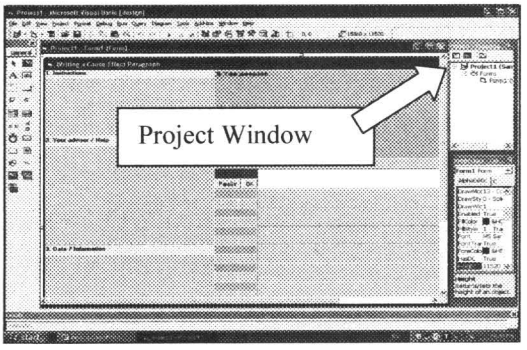
การเขียนคำสั่ง

แม้จะได้หน้าจอที่ต้องการดังรูปที่ 7 แต่หน้าจอนี้เป็นเพียงข้อความและรูปภาพเท่านั้น จึงยังไม่สามารถตอบสนองเหตุการณ์ (event) ที่ผู้เรียนกระทำต่อวัตถุต่างๆได้ เป็นต้นว่าไม่สามารถตอบสนองต่อการที่ผู้เรียนคลิกเมาส์ที่วัตถุนั้น เช่น เมื่อผู้เรียนคลิกเมาส์ที่ปุ่มคำสั่ง Exit ก็จะไม่สามารถออกจากโปรแกรมได้ หรือเมื่อผู้เรียนพิมพ์ประโยคและคลิกเมาส์ที่ปุ่ม OK ก็จะไม่มีการตรวจประโยคนั้น เนื่องจากหน้าจอที่ได้มายังไม่ใช่โปรแกรม และยังไม่มีการเขียนคำสั่งใดๆ ขั้นต่อไปจึงเป็นการเขียนคำสั่งให้กับวัตถุต่างๆ เพื่อให้วัตถุที่ต้องการสามารถตอบสนองผู้เรียนได้

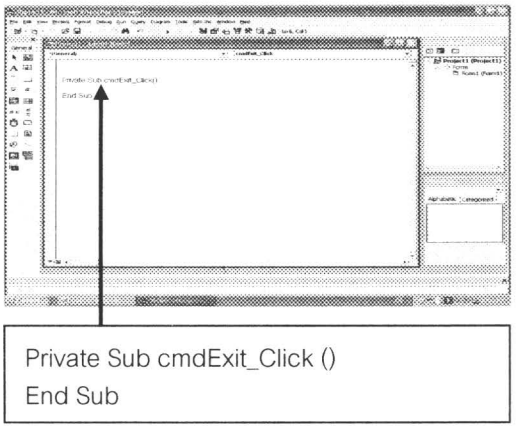
1. การเข้าสู่เนื้อที่เขียนคำสั่ง (Code editor)

การเขียนคำสั่ง (codes) ต้องเขียนในเนื้อที่ส่วนที่เรียกว่า Code editor ซึ่งเป็นที่ว่างสำหรับเขียนคำสั่งโดยเฉพาะ (รูปที่ 9) การเข้าสู่ Code editor ทำได้โดยคลิกเมาส์ที่กรอบ View Code ที่ Project Window (รูปที่ 8) หรือ คลิกเมาส์ที่วัตถุ (Object) ที่วาดไว้บน Form ซ้อนกัน 2 ครั้ง (double click) จากนั้นจะเข้าสู่ Code editor และจะมีข้อความว่า Private Sub แล้วจบด้วย End Sub เรียกว่าโปรแกรมย่อย (Procedure) ซึ่งใช้เขียนคำสั่ง

รูปที่ 8: Project 1 – Form1 (Form)



รูปที่ 9: Project 1 – Form1 (Code)



2. การแบ่งส่วนคำสั่งใน Code editor

เนื่องจาก Code editor เป็นเนื้อที่ว่างคล้ายกับเมื่อท่านเปิดเข้าสู่โปรแกรม Microsoft Word ดังนั้นจึงควรแบ่งเนื้อที่ดังกล่าวนี้ออกเป็นส่วนๆ จึงจะสะดวกต่อการเขียนคำสั่ง และการค้นหา โดยทั่วไปส่วนแรกของ Code editor จะใช้เพื่อประกาศตัวแปรสำหรับเก็บข้อมูล หรือ ข้อความ เช่น ใช้เก็บข้อผิดพลาด คะแนน หรือคำตอบที่ผู้เรียนป้อนเข้ามาในโปรแกรม การประกาศตัวแปรจะใช้คำสั่ง Dim ตามด้วยชื่อตัวแปรและชนิดของตัวแปร เช่น Dim Preposition as Integer Dim TopicSentence as String หรือ Dim TopicSentence$

จากตัวอย่างข้างต้น Dim เป็นคำสั่งให้คอมพิวเตอร์จองเนื้อที่หน่วยความจำไว้สำหรับแต่ละตัวแปร Integer คือตัวแปรจำนวนหรือตัวเลข String หรือ $ (อ่านว่า string) คือข้อความหรืออักขระ ดังนั้น Topic-Sentence$ จึงหมายถึงตัวแปรที่ใช้เก็บคำตอบที่ผู้เรียน ป้อนเข้ามาในโปรแกรม และต่อไปเมื่อจะกล่าวถึง หรือ จะนำคำตอบสำหรับ topic sentence มาตรวจ ก็จะเรียกคำตอบนั้นว่า TopicSentence$ (อ่านว่า topic sentence string) เช่น

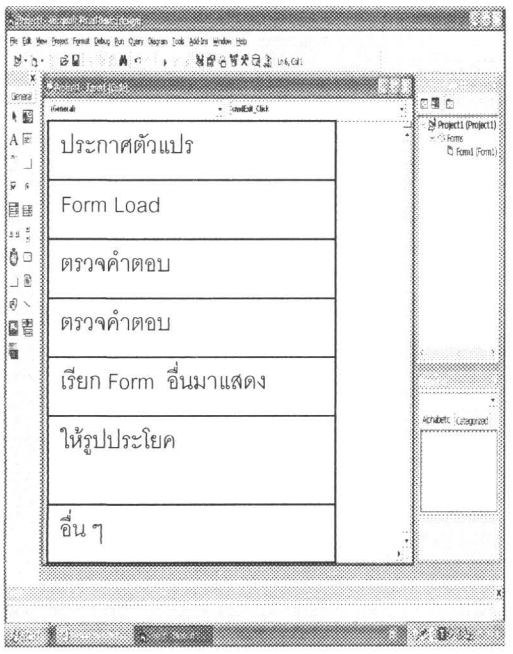
```
If TopicSentence$ = "There are many drawbacks of using mobile phones." Then
    MsgBox "  Very good.  "
```

อ่านว่า *"ถ้าคำตอบประโยค topic sentence ของผู้เรียนคือ (หรือ เท่ากับ) There are many drawbacks of using mobile phones. ให้โปรแกรมแสดงข้อความ Very good. กลางหน้าจอ"*

สำหรับ ตัวแปร Preposition เป็นตัวแปรที่จะใช้เก็บจำนวนครั้งของข้อผิดพลาดด้านการใช้ Preposition ของผู้เรียน

เนื้อที่ใน Code editor ส่วนที่ต่อจากการประกาศตัวแปร จะเป็นโปรแกรมย่อย (Procedure) สำหรับเขียนคำสั่งเฉพาะเรื่องหรือประเด็นตามความต้องการของผู้เขียนโปรแกรม (รูปที่ 10) เช่น Procedure สำหรับตรวจคำตอบ สำหรับเรียก Form อื่นมาแสดง สำหรับให้รูปประโยคที่แสดงความสัมพันธ์แบบเหตุและผล สำหรับเสนอคำกริยาที่ควรใช้เขียนประโยคแสดงความสัมพันธ์แบบเหตุและผล สำหรับสั่งให้จบโปรแกรม และ Procedure อื่นๆ ตามความจำเป็น

รูปที่ 10: **กลุ่มคำสั่งใน** Code editor



3. หลักเบื้องต้นในการเขียนคำสั่ง

สิ่งสำคัญที่ควรทราบก่อนการเขียนคำสั่งสำหรับแต่ละโปรแกรมย่อยคือ คอมพิวเตอร์จะทำงานตามคำสั่ง โดยจะทำงานตามคำสั่งจากบรรทัดบนลงไปบรรทัดล่าง และในแต่ละบรรทัด จะเริ่มทำงานจากคำสั่งด้านซ้ายไปด้านขวา ดังนั้น การวางคำสั่งให้ถูกตำแหน่งจะมีผลต่อการทำงานของโปรแกรม เช่น ถ้าคำสั่งในบรรทัดล่างเป็นเพียงส่วนย่อย (subset) หรือส่วนหนึ่งของคำสั่งในบรรทัดบน โปรแกรมจะทำงานไปไม่ถึงคำสั่งบรรทัดล่าง จะติดอยู่ที่บรรทัดบนเสมอ (เว้นแต่จะมีคำสั่งพิเศษว่าถ้าเคยเข้าสู่เงื่อนไขดังกล่าวนี้แล้ว ให้โปรแกรมไปทำงานที่คำสั่งใดต่อไป) ขอให้ศึกษาตัวอย่างต่อไปนี้ และพิจารณาว่าเพราะเหตุใดโปรแกรมจึงจะทำงานไปไม่ถึงเงื่อนไขที่ 2

```
If InStr(MajorSupp1$, "phone") <> 0 then
(เงื่อนไขที่ 1)
        Beep
        MsgBox " Check if the word 'phone' is singular or plural. "
        Exit Sub
ElseIf InStr(MajorSupp1$, "using mobile phones") = 0 then (เงื่อนไขที่ 2)
        Beep
        MsgBox " Your sentence must include 'using mobile phones'. "
        Exit Sub
```

หากเขียนคำสั่งดังตัวอย่างนี้ โปรแกรมจะทำงานไปไม่ถึงเงื่อนไขที่ 2 เพราะถ้าคำตอบของผู้เรียนมีคำว่า "phone" ก็จะตรงกับเงื่อนไขแรก หรือถ้าคำตอบคือ "phones" ก็ยังตรงกับเงื่อนไขแรกอีกเช่นกัน เนื่องจากในคำว่า "phones" มีคำว่า "phone" อยู่ด้วย ดังนั้น ท่านควรระมัดระวังลำดับของคำสั่ง ไม่เช่นนั้นโปรแกรมของท่านจะไม่สามารถทำงานได้ดังที่ท่านต้องการ การแก้ไขปัญหาในตัวอย่างนี้ อาจทำได้โดยการเคาะแป้น spacebar ให้มีช่องว่าง (blank space) หนึ่งช่องหลังอักษร "e" เป็น "phone " ซึ่งจะทำให้เงื่อนไขแรกเป็นเงื่อนไขที่ว่า เมื่อไม่มีอักษร "s" หรืออักษรใดๆ หลังอักษร "e" ของคำว่า

"phone" ดังนั้น หากคำตอบของผู้เรียนมีคำว่า "phone" ก็จะตรงกับเงื่อนไขแรก และโปรแกรมก็จะทำงานตามคำสั่งต่างๆ ภายใต้เงื่อนไขแรก ซึ่งเตือนให้ผู้เรียนตรวจสอบว่าต้องใช้คำว่า "phone" ในรูปเอกพจน์ หรือพหูพจน์ แต่ถ้าผู้เรียนใช้คำว่า "phones" ในคำตอบอยู่แล้ว โปรแกรมจะผ่านเงื่อนไขแรก เพราะไม่ตรงกับเงื่อนไขนี้ และโปรแกรมจะไปทำงานตามคำสั่งในเงื่อนไขที่ 2 ซึ่งมีคำว่า "phones"

นอกจากนี้ การเขียนคำสั่งจะต้องเขียนตามข้อกำหนด หรือไวยากรณ์ (syntax) ของภาษาคอมพิวเตอร์ที่นำมาใช้ เช่น ผู้เขียนพัฒนาโปรแกรม Writing a Cause-Effect Paragraph จากภาษา Visual Basic ก็ต้องใช้ระเบียบกฎเกณฑ์ของภาษา Visual Basic มิเช่นนั้นโปรแกรมภาษาคอมพิวเตอร์ที่นำมาใช้จะเตือนว่าเขียนคำสั่งผิดเสมอ และท่านจะไม่สามารถทำงานคืบหน้าต่อไปได้ อย่างไรก็ตาม เนื่องจากคำสั่งที่ใช้มีจำนวนมาก จึงไม่สามารถจะกล่าวถึงคำสั่งต่างๆ ได้ทั้งหมดในที่นี้ แต่จะให้แนวคิดในการเขียนคำสั่ง โดยเฉพาะควรสั่งให้โปรแกรมทำสิ่งใดบ้าง และให้ตัวอย่างคำสั่งภาษา Visual Basic พอเป็นสังเขป

4. คำสั่งควบคุมหน้าจอ

โดยทั่วไป ไม่จำเป็นต้องเขียนคำสั่งสำหรับทุกวัตถุที่นำมาสร้างหน้าจอ แต่จะเขียนคำสั่งให้กับวัตถุที่จำเป็นสำหรับการเขียนย่อหน้าเท่านั้น เช่น แม้หน้าจอของโปรแกรม Writing a Cause-Effect Paragraph มีหลายส่วน แต่มีเพียง 4 ส่วนที่ต้องเขียนคำสั่งกำกับได้แก่

1. ส่วนที่ให้ข้อมูลความรู้ทั่วไปเกี่ยวกับการเขียนย่อหน้าแสดงความสัมพันธ์แบบเหตุและผล

2. ส่วนที่เป็นเนื้อเรื่อง หรือเนื้อหาที่จะใช้เขียนย่อหน้าแสดงความสัมพันธ์แบบเหตุและผลเกี่ยวกับการใช้โทรศัพท์เคลื่อนที่

3. ส่วนที่ให้ผู้เรียนพิมพ์คำตอบ

4. ส่วนที่ให้ข้อมูลเฉพาะสำหรับแบบฝึกหัดที่ผู้เรียนกำลังจะเขียน

ดังนั้น เพื่อช่วยให้ผู้เรียนเห็นความสัมพันธ์ระหว่างข้อมูลที่จะนำมาเขียน กับกรอบรับข้อความที่ผู้เรียนพิมพ์ประโยคได้อย่างชัดเจน จึงต้องเขียนคำสั่งว่าเมื่อผู้เรียนคลิกเมาส์บนกรอบรับข้อความส่วนใดของย่อหน้า ให้โปรแกรมเปลี่ยนสีกรอบข้อความเหนือปุ่ม OK เป็นอักษรเหลืองและสีพื้นเป็นสีแดงและให้เปลี่ยนสีอักษรของข้อมูลส่วนที่ต้องนำมาเขียนแตกต่างกันไป เช่น คลิกเมาส์ที่กรอบรับข้อความ Topic sentence ให้อักษรของทุก Major support เป็นสีขาวบนพื้นแดง สำหรับกรอบเนื้อหาอื่นไม่เปลี่ยนแปลง ดังตัวอย่างต่อไปนี้

คำสั่ง	คำอธิบาย
Private Sub txtTopicSent_Click() lblTopicSent.BackColor = vbRed lblTopicSent.ForeColor = vbYellow	- เมื่อมีการคลิกที่กรอบรับข้อความ txtTopicSent ให้อักษรในกรอบข้อความเหนือปุ่ม OK มีสีเหลืองบนพื้นสีแดง
lblTopic.BackColor = vbRed lblTopic.ForeColor = vbWhite	- ให้ข้อความ "Topic sentence" (Part 2) มีสีขาวบนพื้นแดง
lblFirstMajor.BackColor = vbRed lblFirstMajor.ForeColor = vbWhite lblSecondMajor.BackColor = vbRed lblSecondMajor.ForeColor = vbWhite lblThirdMajor.BackColor = vbRed lblThirdMajor.ForeColor = vbWhite	- ให้ข้อความสำหรับทุก Major support (Part 2) มีสีขาวบนพื้นแดง
.... End Sub	สีของส่วนอื่นๆ

ตัวอย่างโปรแกรมย่อยสำหรับนำกรอบข้อความ lblAnswers ที่มีข้อความ "Suggested Answers" มาแสดงทับข้อความ "Part 5: Results" ของหน้าจอโปรแกรม Wring a Cause-Effect Paragraph และแสดงตัวอย่างคำตอบ

```
Private Sub txtMajorSupp1_DblClick()
lblAnswers.Visible = True

    lblResults = _
    " 1. To begin with, using mobile phones causes /can cause
        health problems." & vbCrLf & _
    " " & vbCrLf & _
    " 2. First of all, it contributes /can contribute to health
        problems. " & vbCrLf & _
    " " & vbCrLf & _
    " 3. First, it leads / can lead to health problems." & vbCrLf & _
    " " & vbCrLf & _
    " 4. Firstly, it results / can result in health problems." &
        vbCrLf & _
    " " & vbCrLf & _
    " 5. The first drawback is health problems." & vbCrLf & _
    " 6. The first drawback of using mobile phones is health
        problems." &
        vbCrLf & _
End Sub
```

สั่งให้กรอบข้อความชื่อ lblAnswers ซึ่งทับอยู่บนข้อความ "Part 5: Results" และไม่ได้ปรากฏให้เห็น เพราะถูกบังคับด้วยคำสั่ง Visible = False ให้ปรากฏด้วยคำสั่ง (Visible = True) ซึ่งจะทำให้ข้อความ "Part 5: Results" หายไป

คำสั่งให้แสดงตัวอย่างคำตอบของ Major support 1

ตัวอย่างโปรแกรมย่อยสำหรับสั่งไม่ให้กรอบรับข้อความรับคำตอบจากผู้เรียนอีก เพราะผู้เรียนรวมคำตอบ Minor support12 ไว้กับ Minor support11 แล้ว

```
Private Sub Minor11_12Answer ()                    ← โปรแกรมย่อยสำหรับสั่งไม่รับคำตอบ
                                                     คำสั่งนี้อ่านว่า "ถ้าคำตอบสำหรับ
    If MinorSupp11$ = "For example, the user is likely to   ← Minor support11 คือ For example,
        get headache and a risk of brain cancer when        ... phones."
        talking too much on mobile phones." then

        txtMajorSupp2.SetFocus                     ← ให้ cursor ไปที่กรอบ Major support 2
        txtMinorSupp12.Enabled = False             ← ให้กรอบ Minor support12 ไม่ทำงาน
        txtMinorSupp12.FontBold = False            ← ให้อักษรในกรอบนี้เป็นตัวธรรมดา
        txtMinorSupp12.Text = "You have already    ← ให้แสดงข้อความว่าคำตอบส่วนนี้รวม
            included this minor support in the first minor   อยู่กับคำตอบ Minor support11 แล้ว
            support. Go to the second major support. "

        MsgBox = "Very good"                       ← ให้แสดงคำชม
Else
End if
End Sub
```

5. คำสั่งตรวจคำตอบ

การเขียนคำสั่ง (code) ต้องเขียนอย่างระมัดระวัง ต้องเขียนตามไวยากรณ์ (syntax) ของภาษาคอมพิวเตอร์ที่ใช้ ซึ่งรวมทั้งการเขียนตัวคำสั่ง และการใช้เครื่องหมายต่างๆ เช่น เครื่องหมายจุลภาค และเครื่องหมายคำพูด หากมีข้อผิดพลาด โปรแกรมจะหยุดการทำงาน และจะมีคำเตือน "Syntax error" แต่หากเป็นข้อผิดพลาดด้านตรรกะ โปรแกรมจะทำงาน แต่จะทำงานผิดพลาด ตัวอย่างเช่น หากต้องการตรวจสอบว่าคำตอบของผู้เรียนไม่มีทั้งคำว่า Therefore และไม่มี Thus และเขียนคำสั่งต่อไปนี้

```
If   InStr(TopicSent$), "Therefore" = 0 Or _
     InStr(TopicSent$), "Thus"  =  0  Then
```

คำสั่งนี้จะผิดตรรกะ เนื่องจากใช้คำเชื่อม "Or" แทนที่จะใช้ "And" เพราะเมื่อเขียนเช่นนี้ ถ้าในคำตอบของผู้เรียนมี Therefore ก็จะเข้าสู่เงื่อนไขนี้ หรือเมื่อมีคำว่า Thus ก็จะเข้าสู่เงื่อนไขนี้ และหากในคำตอบของผู้เรียนมีทั้ง Therefore และ Thus ก็จะเข้าสู่เงื่อนไขนี้อีกเช่นกัน ซึ่งไม่ใช่สิ่งที่เราต้องการ เพราะเงื่อนไขที่ต้องการคือเมื่อไม่มีทั้ง Therefore และ Thus ดังนั้น จึงต้องเปลี่ยน "Or" เป็น "And" ซึ่งจะได้ผลตามที่ต้องการ เพราะตามหลักตรรกะนั้น เงื่อนไขนี้จะเป็นจริงได้กรณีเดียวคือ เมื่อไม่มีทั้ง Therefore และ Thus (เพราะเมื่อ P = True, Q = True นั้น P ^ Q = True) ดังนั้น ผู้เขียนโปรแกรมโดยเฉพาะผู้ที่เพิ่งเริ่มเขียนโปรแกรม ควรระวังเรื่องนี้เป็นอย่างยิ่ง (Bradley, 2002)

ในการตรวจคำตอบ ท่านควรเขียนคำสั่งตรวจการเขียนส่วนต่างๆ ของย่อหน้าตามหลักการกว้างๆ เกี่ยวกับการตรวจดังนี้

1. ตรวจว่ามีคำตอบหรือไม่ เพราะบางครั้งผู้เรียนอาจคลิกปุ่มคำสั่งให้ตรวจโดยที่ไม่ได้พิมพ์ข้อความใดๆ

   ```
   f Len(TopicSent$) = 0 Then
   ```

   คำสั่งนี้หมายความว่า "ถ้าประโยค topic sentence มีความยาวเท่ากับศูนย์ (คือไม่มีคำตอบ)" จากนั้นเป็นคำสั่งให้โปรแกรมแสดงคำแนะนำผู้เรียน

2. ตรวจการพิมพ์อักษรตัวพิมพ์ใหญ่ และตัวพิมพ์เล็ก

```
If Left$(TopicSent$, 1) >= Chr(97) And
Left$(TopicSent$,1) <= Chr(122) Then
```

คำสั่งนี้หมายความว่า "ถ้าอักษรตัวแรกของคำตอบเป็นอักษรตัวใดตัวหนึ่งตั้งแต่อักษร a ไปจนถึงอักษร z (ตัวพิมพ์เล็กทั้งหมด) " จากนั้นจึงเป็นคำแนะนำให้ผู้เรียนเปลี่ยนอักษรตัวแรกนั้นเป็นอักษรตัวใหญ่ ในที่นี้ Chr(97) คืออักษร a และ Chr(122) คือ อักษร z ซึ่งเป็นตัวพิมพ์เล็ก และ Left$ เป็นคำสั่งให้ตรวจสอบอักษรจากด้านซ้าย จากคำสั่ง Left$(TopicSent$, 1) นั้นเลข 1 ระบุว่าเป็นอักษรตัวแรก

3. ตรวจสอบการพิมพ์คำตอบด้วยอักษรตัวพิมพ์ใหญ่ทั้งหมด

```
If TopicSent$ = UCase(TopicSent$) Then
```

คำสั่งนี้หมายความว่า "ถ้าคำตอบเป็นอักษรตัวใหญ่ (ทั้งหมด)" จากนั้นจะเป็นคำสั่งให้โปรแกรมแสดงคำแนะนำผู้เรียนให้พิมพ์อักษรตัวใหญ่เฉพาะเท่าที่จำเป็นเท่านั้น

4. ตรวจการสะกดคำ เพราะถ้าสะกดไม่ถูกต้องแล้วตรวจอย่างไรก็จะไม่มีคำตอบถูก เพราะคำตอบที่เรากำหนดไว้นั้นสะกดคำถูกต้อง

```
If InStr(UCase(TopicSent$),
"DRAWBACKS") = 0 Then
```

คำสั่งนี้หมายความว่า "ถ้าในคำตอบของผู้เรียนไม่มีคำว่า DRAWBACKS" ให้โปรแกรมแจ้งให้ผู้เรียนพิมพ์คำนี้ในคำตอบ หรือหากมีคำนี้แล้วให้สะกดคำให้ถูกต้อง แต่การใช้คำสั่งนี้จะใช้ได้ในกรณีที่คำตอบมีคำว่า drawbacks เท่านั้น หากผู้เรียนใช้คำว่า problems หรือ adverse effects โปรแกรมจะบอกให้ผู้เรียนพิมพ์คำว่า drawbacks ทุกครั้ง หากโปรแกรมให้ใช้คำอื่นแทนคำว่า drawbacks ได้ ก็อาจใช้คำสั่งต่อไปนี้

```
If InStr(UCase(TopicSent$),
"DRAWSBACKA") <> 0 Or _

InStr(UCase(TopicSent$),
DRWABACKS") <> 0 Or _

InStr(UCase(TopicSent$),
"DRAWBAKCS") <> 0  Or _

InStr(UCase(TopicSent$),
"DRAWBACSK") <>0  Or _

InStr(UCase(TopicSent$),
"DRAWBAKSC") <>0  Or _

InStr(UCase(TopicSent$),
"DRAWBAKCS") <>0  Or _

InStr(UCase(TopicSent$),
"DARWBACKS") <>0  Then
```

5. ตรวจว่าผู้เรียนใช้คำเอกพจน์หรือพหูพจน์ถูกต้องหรือไม่

```
If InStr(UCase(TopicSent$),
"DRAWBACK.") <> 0 Or _

InStr(UCase(TopicSent$), "DRAWBACK ")
<> 0 Or _
```

คำสั่งนี้หมายความว่า "ถ้าในคำตอบของผู้เรียนไม่มีคำว่า DRAWBACK. หรือ DRAW-BACK" ให้โปรแกรมแจ้งให้ผู้เรียนทราบ ว่าต้องใช้รูปพหูพจน์

6. ตรวจว่าผู้เรียนใช้ข้อมูลที่ให้ครบถ้วนหรือไม่ เพราะถ้าข้อมูลไม่ครบ คำตอบจะผิดเสมอ

```
If InStr(UCase(TopicSent$), "USING
MOBILE PHONES") = 0  And _

InStr(UCase(TopicSent$), "THE USE OF
MOBILE PHONES") = 0  And _

InStr(UCase(TopicSent$), "THE USAGE
OF MOBILE PHONES") = 0  And _
```

InStr(UCase(TopicSent$), "MOBILE PHONE USAGE") = 0 Then

คำสั่งนี้หมายความว่า "ถ้าในคำตอบของผู้เรียนไม่มีข้อความ (ที่อยู่ในเครื่องหมายคำพูด)" ให้โปรแกรมแจ้งให้ผู้เรียนใช้ข้อความใดข้อความหนึ่งในสี่ข้อความนี้ในคำตอบ

7. ตรวจว่าผู้เรียนใช้คำบ่งชี้ (marker) ถูกต้องเหมาะสมหรือไม่

   If InStr(UCase(MajorSupp1$), "TO BEGIN WITH") = 0 And _

   InStr(UCase(MajorSupp1$), "FIRST OF ALL") = 0 And _

   InStr(UCase(MajorSupp1$), "FIRST") = 0 And _

   InStr(UCase(MajorSupp1$), "FIRSTLY") = 0 Then

   คำสั่งนี้หมายความว่า "ถ้าในคำตอบของผู้เรียนไม่มีข้อความ .... (ที่อยู่ในเครื่องหมายคำพูด)" ให้โปรแกรมแจ้งให้ผู้เรียนใช้คำใดคำหนึ่งในคำตอบ

8. ตรวจว่าคำตอบของผู้เรียนผิดที่ใดบ้าง

   If InStr(MajorSupp1$, "First, using mobile phones cause many drawbacks.") <> 0 Then

   คำสั่งนี้หมายความว่า "ถ้าในคำตอบของผู้เรียนมีข้อความ (ที่อยู่ในเครื่องหมายคำพูด)" ให้โปรแกรมแจ้งให้ผู้เรียนตรวจสอบ subject-verb agreement

9. ตรวจว่าคำตอบของผู้เรียนถูกต้องหรือไม่

   If InStr(MajorSupp1$, "First, using mobile phones are many drawbacks.") <> 0 Then

   คำสั่งนี้หมายความว่า "ถ้าในคำตอบของผู้เรียนมีข้อความ (ที่อยู่ในเครื่องหมายคำพูด)" ให้โปรแกรมแจ้งให้ผู้เรียนตรวจสอบการใช้คำกริยา (verb)

10. ตรวจความซ้ำซ้อนของการใช้คำ และ/หรือ รูปประโยค

    If UCase(ConcludingSent$) = "THEREFORE, " + UCase(TopicSent$) Then

    คำสั่งนี้หมายความว่า "ถ้าคำตอบประโยค Concluding sentence เท่ากับคำว่า Therefore บวก กับประโยค Topic sentence ที่เขียนไปแล้ว" ให้โปรแกรมแจ้งให้ผู้เรียนแก้รูปประโยค หรือ คำในประโยค Concluding sentence ให้แตกต่างไปจากรูปประโยค หรือ คำในประโยค ที่ใช้ในประโยค Topic sentence

จากที่กล่าวมาเกี่ยวกับการตรวจคำตอบ และตัวอย่างคำสั่ง จะเห็นได้ว่าผู้สอน และ/หรือ ผู้เขียนโปรแกรม เป็นผู้ตรวจคำตอบ คอมพิวเตอร์เพียงทำตามคำสั่งเท่านั้น ผู้สอนจึงเป็นผู้กำหนดว่าจะรับหรือไม่รับคำตอบใด ซึ่งรวมทั้งการกำหนดรายละเอียดต่างๆ ของการตรวจ ผู้สอนสามารถกำหนดรายละเอียดต่างๆ ได้ว่าประโยคอย่างไรจึงจะรับว่าถูกต้อง และประโยคอย่างไรไม่ถูกต้อง หากไม่ถูกต้องควรอธิบายให้ผู้เรียนเห็นข้อผิดพลาด จนสามารถแก้ไขข้อผิดพลาดนั้นได้ กล่าวโดยสรุปก็คือผู้สอนเป็นผู้ตรวจงานเขียนที่แท้จริง โปรแกรมคอมพิวเตอร์เป็นเพียงเครื่องมือที่ทำงานตามที่เรากำหนดเท่านั้น แต่ปัญหาสำคัญคือ จะเขียนคำสั่งอย่างไร จึงจะสามารถตรวจงานเขียนได้ดังที่เราต้องการ

เมื่อตรวจคำตอบแล้ว โปรแกรมต้องให้ข้อมูลป้อนกลับที่เป็นประโยชน์กับผู้เรียน ข้อมูลที่สำคัญได้แก่ผลการตรวจ ข้อผิดพลาด และข้อแนะนำสำหรับการแก้ไขข้อผิดพลาด Harmer (2007) ได้เสนอแนะไว้ว่าข้อแนะนำที่ให้ควรเป็นข้อแนะนำที่ช่วยให้ผู้เรียนได้เรียนรู้จากข้อผิดพลาด

การประเมินโปรแกรม

เมื่อเขียนโปรแกรมเสร็จเรียบร้อยและได้ตรวจสอบการทำงานทุกขั้นตอนของโปรแกรมรวมทั้งได้ตรวจความถูกต้องของคำแนะนำ และการสะกดคำแล้ว ขั้นต่อไปเป็นการทดลองใช้โปรแกรมกับกลุ่มเป้าหมาย เพื่อที่จะนำปัญหาต่างๆ ที่พบขณะทดลองมาแก้ไขปรับปรุงโปรแกรมก่อนนำไปใช้จริง ก่อนการทดลอง ท่านควรสร้างแบบสอบถามสำหรับการประเมินที่ครอบคลุมรายละเอียดที่ต้องการเก็บข้อมูล และควรให้ผู้เชี่ยวชาญ หรือ ผู้ที่มีประสบการณ์เกี่ยวกับการสร้างและการใช้แบบสอบถามตรวจสอบความถูกต้องเหมาะสมของแบบสอบถามก่อนนำไปใช้ สำหรับแบบสอบถามเพื่อประเมินโปรแกรมเรื่อง Writing a Cause-Effect Paragraph ได้ผ่านการตรวจสอบของผู้เชี่ยวชาญ 4 ท่าน ซึ่งได้ให้ข้อเสนอแนะเพื่อแก้ไขปรับปรุงหลายประการ ท่านอาจใช้แบบสอบถามในภาคผนวกที่ 1 เพื่อประกอบการพิจารณาสร้างแบบสอบถามที่เหมาะสมกับการประเมินโปรแกรมของท่าน

สำหรับประเมินโปรแกรมเรื่อง Writing a Cause-Effect Paragraph นั้น ผู้เขียนใช้กลุ่มทดลองสองกลุ่ม คือกลุ่มนิสิตชั้นปีที่ 1 จำนวน 30 คน และนิสิตระดับบัณฑิตศึกษาอีกหนึ่งกลุ่มจำนวน 20 คน ผลการประเมินโดยรวมพบว่าผู้เรียนส่วนใหญ่มีความเห็นว่าโปรแกรมนี้มีประโยชน์ต่อการฝึกเขียนย่อหน้าแสดงความสัมพันธ์แบบเหตุและผล ดังรายละเอียดในภาคผนวกที่ 2

แม้การพัฒนาโปรแกรมโดยใช้ภาษาคอมพิวเตอร์โดยตรงนั้นจะยาก แต่เป็นสิ่งที่เป็นไปได้ ในระยะเริ่มต้นอาจมีอุปสรรคบ้าง แต่หากท่านมีความ ตั้งใจที่จะเป็นผู้สร้างโปรแกรม ท่านจะสามารถทำได้เช่นกัน อย่างไรก็ตาม เพื่อช่วยให้ท่านประสบผล สำเร็จในการพัฒนาโปรแกรมได้สะดวกขึ้น ท่านควรมีที่ปรึกษาด้านการเขียนโปรแกรม และควรวางแผนการเขียนโปรแกรมอย่างเป็นระบบ และดำเนินตามแผนการไปทีละขั้นตอน แต่ที่สำคัญยิ่งก็คือ ก่อนเขียนโปรแกรม ท่านควรเก็บข้อมูลเกี่ยวกับข้อผิดพลาดต่างๆ ในการใช้ภาษาของผู้เรียน และนำข้อมูลนั้นมาใช้ให้เกิดประโยชน์

เอกสารอ้างอิง

Bradley, Julia Case and Millspaugh, Anita C. 2002. **Programming in Visual Basic 6.0**, Boston, Massachusetts: McGraw-Hill/ Irwin.

Harmer, Jeremy (Ed.). 2007. **How to Teach English with Technology**, Pearson/Longman: Essex, England.

.....

ภาคผนวกที่ 1

แบบประเมินโปรแกรมช่วยการเขียนย่อหน้าแสดงเหตุและผล

Writing a Cause-Effect Paragraph Evaluation Form

คำแนะนำ โปรดทำเครื่องหมาย √ ในช่องที่ตรงกับความคิดเห็นของท่านมากที่สุด ให้ครบทุกข้อ

หัวข้อการประเมิน	มากที่สุด	มาก	ปานกลาง	น้อย	น้อยที่สุด
1. วัตถุประสงค์ของโปรแกรมชัดเจนเพียงใด					
2 หน้าจอ (screen) เหมาะสมเพียงใด					
3. โปรแกรมนี้ ให้ความรู้ทั่วไปเกี่ยวกับการเขียนย่อหน้าแสดงเหตุและผลเพียงใด					
4. คำแนะนำเฉพาะสำหรับการเขียนแต่ละส่วนของย่อหน้าของแบบฝึกหัดชัดเจนเพียงใด					
5. โปรแกรมนี้รับประโยคคำตอบที่ถูกต้อง (acceptable sentences) ได้มากน้อยเพียงใด					
6. โปรแกรมนี้อธิบายและแนะนำวิธีแก้ไขข้อผิดพลาดด้านไวยากรณ์มากน้อยเพียงใด					
7. ภาษาที่ให้คำแนะนำการแก้ไขข้อผิดพลาดต่าง ๆ เหมาะสมเพียงใด					
8. ลักษณะการนำเสนอข้อแนะนำการแก้ข้อผิดพลาด ช่วยให้ท่านสามารถแก้ข้อผิดพลาดต่าง ๆ ได้มากน้อยเพียงใด					
9. โปรแกรมให้คำแนะนำการใช้คำบ่งชี้หรือคำเชื่อมโยง (marker/transition) มากน้อยเพียงใด					
10. โปรแกรมสามารถแจ้งข้อผิดพลาดในประเด็นต่าง ๆ ได้ชัดเจนเพียงใด					
11. โปรแกรมเปิดโอกาสให้ท่านเขียนประโยคสำหรับแต่ละส่วนของย่อหน้าโดยใช้ศัพท์ และ รูปประโยค (sentence structures) สำหรับการเขียนเกี่ยวกับเหตุและผล ได้หลากหลายเพียงใด					
12. การที่โปรแกรมเปิดโอกาสให้ท่านสามารถเขียนส่วนใดของย่อหน้าก่อนก็ได้ โดยไม่ต้องเขียนแต่ละส่วนไปตามลำดับนั้น มีประโยชน์มากน้อยเพียงใด					
13. ความยากง่ายของแบบฝึกหัดเหมาะสมกับความสามารถของท่านเพียงใด					
14. ท่านสามารถควบคุม (control) โปรแกรมได้มากน้อยเพียงใด เช่น สามารถเลือกที่จะเขียน เลือกที่จะศึกษาวิธีเขียน หรือ จะออกจากโปรแกรมเมื่อใดก็ได้					
15. โปรแกรมนี้มีประโยชน์สำหรับการฝึกเขียนย่อหน้าแสดงเหตุและผลมากน้อยเพียงใด					

ข้อเสนอแนะ

_____

_____

_____

_____

_____

ภาคผนวกที่ 2

ผลการประเมินโปรแกรมช่วยการเขียนย่อหน้าแสดงเหตุและผล

คำแนะนำ โปรดทำเครื่องหมาย √ ในช่องที่ตรงกับความคิดเห็นของท่านมากที่สุด ให้ครบทุกข้อ

หัวข้อการประเมิน	ค่าเฉลี่ย
1. วัตถุประสงค์ของโปรแกรมชัดเจนเพียงใด	4.40
2 หน้าจอ (screen) เหมาะสมเพียงใด	3.84
3. โปรแกรมนี้ให้ความรู้ทั่วไปเกี่ยวกับการเขียนย่อหน้าแสดงเหตุและผลเพียงใด	3.84
4. คำแนะนำเฉพาะสำหรับการเขียนแต่ละส่วนของย่อหน้าของแบบฝึกหัดชัดเจนเพียงใด	3.74
5. โปรแกรมนี้รับประโยคคำตอบที่ถูกต้อง (acceptable sentences) ได้มากน้อยเพียงใด	3.28
6. โปรแกรมนี้อธิบายและแนะนำวิธีแก้ไขข้อผิดพลาดด้านไวยากรณ์มากน้อยเพียงใด	3.66
7. ภาษาที่ให้คำแนะนำการแก้ไขข้อผิดพลาดต่าง ๆ เหมาะสมเพียงใด	3.70
8. ลักษณะการนำเสนอข้อแนะนำการแก้ข้อผิดพลาดช่วยให้ท่านสามารถแก้ข้อผิดพลาดต่าง ๆ ได้มากน้อยเพียงใด	3.42
9. โปรแกรมให้คำแนะนำการใช้คำบ่งชี้หรือคำเชื่อมโยง (marker/transition) มากน้อยเพียงใด	3.60
10. โปรแกรมสามารถแจ้งข้อผิดพลาดในประเด็นต่าง ๆ ได้ชัดเจนเพียงใด	3.60
11. โปรแกรมเปิดโอกาสให้ท่านเขียนประโยคสำหรับแต่ละส่วนของย่อหน้าโดยใช้ศัพท์และ รูปประโยค (sentence structures) สำหรับการเขียนเกี่ยวกับเหตุและผล ได้หลากหลายเพียงใด	3.10
12. การที่โปรแกรมเปิดโอกาสให้ท่านสามารถเขียนส่วนใดของย่อหน้าก่อนก็ได้โดยไม่ต้องเขียนแต่ละส่วนไปตามลำดับนั้น มีประโยชน์มากน้อยเพียงใด	3.80
13. ความยากง่ายของแบบฝึกหัดเหมาะสมกับความสามารถของท่านเพียงใด	3.68
14. ท่านสามารถควบคุม (control) โปรแกรมได้มากน้อยเพียงใด เช่น สามารถเลือกที่จะเขียน เลือกที่จะศึกษาวิธีเขียน หรือ จะออกจากโปรแกรมเมื่อใดก็ได้	3.66
15. โปรแกรมนี้มีประโยชน์สำหรับการฝึกเขียนย่อหน้าแสดงเหตุและผลมากน้อยเพียงใด	4.06