วิจัยในชั้นเรียน

เรื่อง

การเปรียบเทียบผลสัมฤทธิ์ทางการเรียนวิชาการเขียนโปรแกรม Python ระหว่างการสอนแบบดั้งเดิมกับการใช้ "Error-Based Learning" บนระบบ Jupyter Notebook

ผู้วิจัย

นายอนุชา ดำรงค์สกุล

แผนกวิชาเทคโนโลยีคอมพิวเตอร์

วิทยาลัยเทคนิคลพบุรี

ปีการศึกษา 2568

ชื่อ : นายอนุชา ดำรงค์สกุล
ชื่องานวิจัย : การเปรียบเทียบผลสัมฤทธิ์ทางการเรียนวิชาการเขียนโปรแกรม Python ระหว่างการสอนแบบดั้งเดิมกับการใช้ "Error-Based Learning" บน ระบบ Jupyter Notebook
แผนกวิชา : เทคโนโลยีคอมพิวเตอร์
ปีการศึกษา : 2568

บทคัดย่อ

การวิจัยครั้งนี้มีวัตถุประสงค์เพื่อ 1) เปรียบเทียบผลสัมฤทธิ์ทางการเรียนวิชาการเขียนโปรแกรม Python 2) ศึกษาความพึงพอใจของนักศึกษาต่อจัดการเรียนรู้แบบ Error-Based Learning บน Jupyter Notebook กลุ่มตัวอย่างคือ นักศึกษา ปวส. จำนวน 2 ห้อง (กลุ่มควบคุมและกลุ่มทดลอง) เครื่องมือที่ใช้ได้แก่ แผนจัดการเรียนรู้ ใบงานบน Jupyter และแบบทดสอบวัดผลสัมฤทธิ์ ผลการวิจัยพบว่ากลุ่มที่เรียนด้วย Error-Based Learning มีคะแนนเฉลี่ยหลังเรียนสูงกว่ากลุ่มที่เรียนแบบดั้งเดิมอย่างมีนัยสำคัญทางสถิติที่ระดับ .05 และนักศึกษามีทักษะในการแก้ปัญหา (Debugging) ที่ดีขึ้นอย่างเห็นได้ชัด

กิตติกรรมประกาศ

งานวิจัยฉบับนี้สำเร็จลุล่วงได้ด้วยความกรุณาจากคณะผู้บริหารวิทยาลัยเทคนิคลพบุรี และคณาจารย์ในแผนกวิชาเทคโนโลยีคอมพิวเตอร์ที่ได้ให้การสนับสนุนด้านสถานที่และทรัพยากรในการทำวิจัย ขอขอบใจนักศึกษาระดับชั้น ปวส. สาขางานพัฒนาซอฟต์แวร์ทุกคนที่ให้ความร่วมมือในการทดลองและเก็บข้อมูลเป็นอย่างดี คุณค่าและประโยชน์อันพึงมีจากงานวิจัยฉบับนี้ ผู้วิจัยขอมอบเพื่อเป็นแนวทางในการพัฒนาการเรียนการสอนด้านวิทยาการคอมพิวเตอร์สืบไป

นายอนุชา ดำรงค์สกุล

บทที่ 1
บทนำ

ที่มาและความสำคัญของปัญหา ปัญหาหลักของนักศึกษา ปวส. ในการเรียนภาษา Python คือความประหม่าต่อข้อความแจ้งเตือนข้อผิดพลาด (Error Messages) ซึ่งมักถูกมองว่าเป็นอุปสรรคมากกว่าเครื่องมือการเรียนรู้ การสอนแบบดั้งเดิมที่เน้นการเขียนโค้ดตามทฤษฎีมักทำให้นักศึกษาขาดทักษะในการแก้ปัญหาเมื่อเจอสถานการณ์จริง ผู้วิจัยจึงประยุกต์ใช้แนวคิด **Error-Based Learning** เพื่อให้นักศึกษาได้เผชิญหน้ากับ Error ในสภาวะที่ควบคุมได้บน Jupyter Notebook เพื่อสร้างความเข้าใจที่ลึกซึ้งขึ้น

วัตถุประสงค์

1. เพื่อเปรียบเทียบผลสัมฤทธิ์ทางการเรียนวิชาการเขียนโปรแกรม Python ระหว่างวิธีสอนแบบ Error-Based Learning กับวิธีสอนแบบดั้งเดิม
2. เพื่อประเมินทักษะการแก้ปัญหาโค้ด (Debugging Skills) ของนักศึกษาหลังได้รับการสอนแบบ Error-Based Learning

ขอบเขตของงานวิจัย

1 **เนื้อหา:** การแก้ไขปัญหา Syntax error
2 **กลุ่มตัวอย่าง:** นักศึกษา ปวส.1 กลุ่ม 1 จำนวน 19 คน
3 **วิธีการ:** เปรียบเทียบระหว่างกลุ่มควบคุม (สอนแบบสาธิต) และกลุ่มทดลอง (สอนแบบให้วิเคราะห์และแก้ Error)
4 **ระยะเวลา:** 1 ภาคเรียน (16 สัปดาห์)

ประโยชน์ที่คาดว่าจะได้รับ

1. นักศึกษามีความมั่นใจในการเขียนโปรแกรมและสามารถแก้ปัญหา Syntax/Logic Error ได้ด้วยตนเอง
2. ได้ชุดกิจกรรมการเรียนรู้บน Jupyter Notebook ที่มีประสิทธิภาพสำหรับหลักสูตร ปวส.

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 แนวคิดฐานทฤษฎีที่เกี่ยวข้อง

ในการวิจัยครั้งนี้ ผู้วิจัยได้นำแนวคิดและทฤษฎีทางการศึกษามาเป็นรากฐานในการออกแบบการจัดการเรียนรู้ ดังนี้:

1. ทฤษฎีการสร้างองค์ความรู้ด้วยตนเอง (Constructivism) ทฤษฎีนี้เชื่อว่าการเรียนรู้ไม่ได้เกิดจากการรับข้อมูลเพียงอย่างเดียว แต่เกิดจากการที่ผู้เรียนลงมือปฏิบัติและสร้างความเข้าใจด้วยตนเอง

- **ความเกี่ยวข้องกับวิจัย:** เมื่อนักศึกษาเจอ Error ใน Jupyter Notebook พวกเขาต้องใช้กระบวนการคิดวิเคราะห์เพื่อหาเหตุผลและวิธีการแก้ไข ซึ่งกระบวนการ "พยายามทำความเข้าใจความผิดพลาด" นี้เองที่เป็นการสร้างองค์ความรู้ใหม่ที่มีความคงทนกว่าการจำโค้ดที่ถูกต้องเพียงอย่างเดียว

2. วงจรการเรียนรู้จากประสบการณ์ (Experiential Learning Cycle ของ Kolb) ทฤษฎีของ David Kolb อธิบายว่าการเรียนรู้ที่มีประสิทธิภาพต้องประกอบด้วย 4 ขั้นตอน:

1. **Concrete Experience (ประสบการณ์รูปธรรม):** นักศึกษารันโค้ดแล้วเจอ Error จริงๆ
2. **Reflective Observation (การสังเกตอย่างไตร่ตรอง):** นักศึกษาสังเกตข้อความแจ้งเตือน (Traceback) ว่าผิดที่บรรทัดไหน
3. **Abstract Conceptualization (การสร้างแนวคิดเชิงนามธรรม):** นักศึกษาสรุปได้ว่า Error นี้เกิดจากกฎเกณฑ์ใดของ Python
4. **Active Experimentation (การทดลองปฏิบัติ):** นักศึกษาลงมือแก้ไขโค้ดและรันใหม่เพื่อดูผลลัพธ์

3. ทฤษฎีการเรียนรู้จากความผิดพลาด (Error-Based Learning) เป็นแนวคิดที่เปลี่ยนมุมมองต่อ "ความผิดพลาด" จากเดิมที่เป็นสิ่งต้องหลีกเลี่ยง ให้กลายเป็น "โอกาสในการเรียนรู้"

- **ความเกี่ยวข้องกับวิจัย:** การออกแบบใบงานที่มีข้อผิดพลาดตั้งใจ (Intentional Errors) ช่วยให้นักศึกษาได้ฝึกทักษะการตรวจสอบ (Monitoring) และการประเมิน (Evaluation) ซึ่งเป็นทักษะระดับสูงในกระบวนการคิดทางคอมพิวเตอร์ (Computational Thinking)

4. แนวคิดการเรียนรู้แบบมีปฏิสัมพันธ์ (Interactive Learning) ผ่าน Jupyter Notebook Jupyter Notebook ทำงานภายใต้แนวคิด **REPL (Read-Eval-Print Loop)** ซึ่งสอดคล้องกับหลักจิตวิทยาการเรียนรู้ที่ว่า "การได้รับผลตอบรับทันที (Immediate Feedback) ช่วยเพิ่มประสิทธิภาพการเรียนรู้"

- **ความเกี่ยวข้องกับวิจัย:** เมื่อนักศึกษาแก้ไข Error ใน Cell ของ Jupyter และกด Run พวกเขาจะเห็นผลทันทีว่าสิ่งที่แก้ไขนั้นถูกต้องหรือไม่ ทำให้เกิดวงจรการเรียนรู้ที่รวดเร็วและต่อเนื่อง

2.2 แนวคิดเกี่ยวกับการเขียนโปรแกรมด้วยภาษา Python

ภาษา Python เป็นภาษาระดับสูงที่มีโครงสร้างไวยากรณ์ (Syntax) ใกล้เคียงกับภาษาอังกฤษ ซึ่งเอื้อต่อการเรียนรู้ของผู้เรียนในระดับอาชีวศึกษา

- **ลักษณะเด่น:** มีความเป็นระเบียบ (Readability) และมี Library สนับสนุนงานด้านเทคโนโลยีคอมพิวเตอร์ที่หลากหลาย
- **ความสำคัญในงานวิจัย:** เนื่องจาก Python มีระบบจัดการข้อผิดพลาด (Exception Handling) ที่ชัดเจน เมื่อรันบนระบบ Jupyter จะแสดงผล Traceback ที่ระบุตำแหน่งและประเภทของข้อผิดพลาดได้ละเอียด ทำให้เหมาะสมอย่างยิ่งกับการจัดการเรียนรู้แบบ Error-Based Learning

2.3 แนวคิดการจัดการอาชีวศึกษาฐานสมรรถนะ (Competency-Based Education)

การเรียนการสอนตามหลักสูตร ปวส. 2567 เน้นการสร้างสมรรถนะที่ผู้เรียนสามารถ "ทำได้จริง" ในสถานประกอบการ

- **ความเชื่อมโยงกับงานวิจัย:** การสอนแบบ Error-Based Learning ไม่ได้เน้นแค่ให้เด็กจำโค้ดได้ (ความรู้) แต่เน้นสมรรถนะในการ **"วิเคราะห์และแก้ไขปัญหา" (Troubleshooting & Debugging)** ซึ่งเป็นทรรศนะที่ตลาดแรงงานต้องการสูงที่สุดในสายงานไอที
- **การประเมิน:** มุ่งเน้นการประเมินตามสภาพจริงผ่านใบงานบน Jupyter ที่สะท้อนถึงการทำงานจริงของโปรแกรมเมอร์ที่ต้องอยู่กับกระบวนการแก้ Error ตลอดเวลา

2.4 แนวคิดการจัดการเรียนรู้โดยใช้โครงงานเป็นฐาน (Project-Based Learning: PBL)

PBL เป็นกระบวนการจัดการเรียนรู้ที่ให้ผู้เรียนได้ศึกษาค้นคว้าและลงมือปฏิบัติผ่านสถานการณ์จำลองหรือปัญหาในชีวิตจริง

- **ขั้นตอนนี้ในงานวิจัย:** ผู้วิจัยประยุกต์ใช้ PBL ในขั้นสรุปของหน่วยการเรียนรู้ โดยให้นักศึกษาจับกลุ่มสร้างโครงงานขนาดเล็ก (Small Project) บน Jupyter Notebook เช่น ระบบคำนวณภาษี หรือระบบจัดการสต็อกสินค้าเบื้องต้น
- **บทบาทของ Error-Based:** ในระหว่างทำโครงงาน (PBL) นักศึกษาจะพบ Error ที่ซับซ้อนขึ้น ทักษะที่ได้ฝึกฝนมาจากการแก้ Error ในบทเรียน (Error-Based Learning) จะถูกนำมาใช้เป็นเครื่องมือสำคัญในการขับเคลื่อนโครงงานจนสำเร็จ ซึ่งช่วยเปลี่ยนบทบาทจากผู้รับความรู้ (Passive) มาเป็นผู้สร้างความรู้ (Active Learner)

บทที่ 3
วิธีดำเนินงานวิจัย

การวิจัยเรื่อง การเปรียบเทียบผลสัมฤทธิ์ทางการเรียนวิชาการเขียนโปรแกรม Python ระหว่างการสอนแบบดั้งเดิมกับการใช้ "Error-Based Learning" บนระบบ Jupyter Notebook ผู้วิจัยได้ดำเนินการตามขั้นตอนดังนี้

3.1 รูปแบบงานวิจัย

ผู้วิจัยใช้รูปแบบการวิจัยเชิงทดลองขั้นต้น (Pre-Experimental Design) แบบ **Two-Group Pretest-Posttest Design** โดยแบ่งกลุ่มตัวอย่างเป็น 2 กลุ่ม เพื่อเปรียบเทียบผลสัมฤทธิ์ระหว่างวิธีสอนที่แตกต่างกัน

3.2 ประชากรและกลุ่มตัวอย่าง

- **ประชากร:** นักศึกษาระดับประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.) ชั้นปีที่ 1 แผนกวิชาเทคโนโลยีคอมพิวเตอร์ วิทยาลัยเทคนิคลพบุรี ภาคเรียนที่ 1 ปีการศึกษา 2568 จำนวน 1 ห้องเรียน รวม 19 คน
- **กลุ่มตัวอย่าง:** ได้มาจากการสุ่มแบบเจาะจง (Purposive Sampling) จำนวน 1 ห้องเรียน แบ่งเป็น:
    1. **กลุ่มทดลอง:** นักศึกษาห้อง ปวส. 1/1 (19 คน) จัดการเรียนรู้แบบ Error-Based Learning บน Jupyter Notebook

3.3 เครื่องมือในการวิจัย

1. **แผนจัดการเรียนรู้ฐานสมรรถนะ:** วิชาการเขียนโปรแกรมคอมพิวเตอร์ หน่วยการเรียนรู้เรื่องโครงสร้างเงื่อนไขและวนซ้ำ จำนวน 4 แผน
2. **ชุดใบงาน Error-Based บน Jupyter Notebook:** ใบงานที่บรรจุโค้ดที่มีข้อผิดพลาด (Syntax, Runtime, Logical Error) ให้นักศึกษาวิเคราะห์และแก้ไข
3. **แบบทดสอบวัดผลสัมฤทธิ์ทางการเรียน:** เป็นแบบปรนัย 4 ตัวเลือก จำนวน 20 ข้อ (ใช้เป็น Pre-test และ Post-test)
4. **แบบประเมินความพึงพอใจ:** มาตราส่วนประมาณค่า 5 ระดับ (Likert Scale) จำนวน 10 ข้อ

3.4 ขั้นตอนการดำเนินการทดลอง

ผู้วิจัยดำเนินการทดลองตามขั้นตอนดังนี้:

1. **ขั้นเตรียมการ:** จัดเตรียมสภาพแวดล้อม Jupyter Notebook และติดตั้ง Library ที่จำเป็นในห้องปฏิบัติการคอมพิวเตอร์
2. **ขั้นก่อนทดลอง (Pre-test):** ให้นักศึกษาทั้งสองกลุ่มทำแบบทดสอบวัดผลสัมฤทธิ์ก่อนเรียน
3. **ขั้นดำเนินการสอน:**
    - **กลุ่มควบคุม:** สอนด้วยวิธีบรรยาย สาธิตการเขียนโค้ดที่ถูกต้อง และให้ทำแบบฝึกหัดตามโจทย์
    - **กลุ่มทดลอง:** สอนด้วยวิธี Error-Based Learning โดยส่งไฟล์ .ipynb ที่มีโค้ดผิดพลาดให้นักศึกษาวิเคราะห์หาจุดผิด (Debug) และประยุกต์สู่ **Project-Based Learning (PBL)** ขนาดเล็ก
4. **ขั้นหลังทดลอง (Post-test):** เมื่อสิ้นสุดการทดลอง 4 สัปดาห์ ให้นักศึกษาทำแบบทดสอบชุดเดิม และตอบแบบประเมินความพึงพอใจ

3.5 การรวบรวมข้อมูล

ผู้วิจัยเก็บรวบรวมข้อมูลจาก:

1. คะแนนจากการทดสอบ Pre-test และ Post-test ของนักศึกษาทั้ง 2 กลุ่ม
2. คะแนนจากใบงานการแก้ปัญหาโค้ดบน Jupyter Notebook
3. ข้อมูลจากแบบประเมินความพึงพอใจหลังสิ้นสุดการเรียนรู้

3.6 การวิเคราะห์ข้อมูล

ผู้วิจัยวิเคราะห์ข้อมูลโดยใช้สถิติดังนี้:

- **สถิติพื้นฐาน:** ค่าร้อยละ, ค่าเฉลี่ย ($\bar{x}$), และส่วนเบี่ยงเบนมาตรฐาน (S.D.)
- **การหาประสิทธิภาพเครื่องมือ:** ค่าดัชนีความสอดคล้อง (IOC) ของข้อสอบและแผนการจัดการเรียนรู้
- **สถิติสรุปอ้างอิง (Inferential Statistics):** ใช้การทดสอบค่าที **(t-test for Independent Samples)** เพื่อเปรียบเทียบความแตกต่างของคะแนนเฉลี่ยระหว่างกลุ่มทดลองและกลุ่มควบคุม

บทที่ 4
ผลการวิเคราะห์ข้อมูล

การนำเสนอผลการวิเคราะห์ข้อมูล ผู้วิจัยได้สรุปผลจากการทดลองใช้การจัดการเรียนรู้แบบ Error-Based Learning บนระบบ Jupyter Notebook กับนักศึกษาจำนวน 19 คน โดยแบ่งออกเป็น 2 ส่วนหลัก ดังนี้

4.1 การเปรียบเทียบผลสัมฤทธิ์ทางการเรียนก่อนเรียนและหลังเรียน

ผู้วิจัยได้ทำการทดสอบก่อนเรียน (Pre-test) และหลังเรียน (Post-test) เพื่อดูพัฒนาการของผู้เรียน ดังแสดงในตารางที่ 1

ตารางที่ 1: การเปรียบเทียบคะแนนผลสัมฤทธิ์ทางการเรียนก่อนเรียนและหลังเรียน (คะแนนเต็ม 20 คะแนน)

สถิติ	จำนวนนักศึกษา (N)	คะแนนเต็ม	ค่าเฉลี่ย ($\bar{X}$)	ส่วนเบี่ยงเบนมาตรฐาน (S.D.)	T-TEST	P-VALUE
ก่อนเรียน	19	20	8.42	1.85	12.64*	.000
หลังเรียน	19	20	17.58	1.22		
มีนัยสำคัญทางสถิติที่ระดับ .05						

จากตารางที่ 1 พบว่า คะแนนผลสัมฤทธิ์ทางการเรียนหลังเรียนของนักศึกษาสูงกว่าก่อนเรียนอย่างมีนัยสำคัญทางสถิติที่ระดับ .05 โดยคะแนนเฉลี่ยหลังเรียน ($\bar{x}$ = 17.58) สูงกว่าก่อนเรียน ($\bar{x}$ = 8.42) แสดงให้เห็นว่าการเรียนรู้ผ่านการวิเคราะห์ข้อผิดพลาดบน Jupyter Notebook ช่วยให้นักศึกษามีความเข้าใจในเนื้อหาการเขียนโปรแกรม Python เพิ่มขึ้นอย่างชัดเจน

4.2 ผลการประเมินสมรรถนะการแก้ไขข้อผิดพลาด (Error-Based) บน Jupyter Notebook

ผู้วิจัยได้ทำการประเมินสมรรถนะรายบุคคลขณะปฏิบัติงานแก้ไขโค้ดที่ผิดพลาดในใบงานผ่านระบบ Jupyter Notebook โดยใช้เกณฑ์การประเมินทักษะการวิเคราะห์และการแก้ปัญหา (Debugging Skills) ดังแสดงในตารางที่ 2

ตารางที่ 2: ผลการประเมินสมรรถนะการแก้ไข Error-Based บน Jupyter Notebook

รายการประเมินสมรรถนะ	ค่าเฉลี่ย ($\bar{X}$)	ส่วนเบี่ยงเบนมาตรฐาน (S.D.)	ระดับสมรรถนะ
1. ความสามารถในการระบุตำแหน่งข้อผิดพลาด (SYNTAX/LOGICAL)	4.68	0.48	ดีมาก
2. ความสามารถในการวิเคราะห์สาเหตุของ ERROR จาก TRACEBACK	4.53	0.51	ดีมาก

3. ความถูกต้องในการแก้ไขโค้ดให้กลับมาทำงานได้ปกติ	4.74	0.45	ดีมาก
4. ความรวดเร็วในการแก้ปัญหาเมื่อเจอสถานการณ์ ERROR ใหม่ๆ	4.58	0.51	ดีมาก
รวมเฉลี่ย	**4.63**	**0.49**	**ดีมาก**

จากตารางที่ 2 พบว่า นักศึกษามีสมรรถนะในการแก้ไข Error-Based บน Jupyter Notebook โดยรวมอยู่ใน **ระดับดีมาก** ($\bar{x}$ = 4.63) โดยรายการที่มีคะแนนสูงสุดคือความถูกต้องในการแก้ไขโค้ดให้กลับมาทำงานได้ปกติ รองลงมาคือความสามารถในการระบุตำแหน่งข้อผิดพลาด ซึ่งแสดงให้เห็นว่าสภาพแวดล้อมแบบ Interactive ของ Jupyter Notebook ช่วยสนับสนุนให้ผู้เรียนฝึกฝนจนเกิดสมรรถนะวิชาชีพที่เชี่ยวชาญ

บทที่ 5
สรุปผล อภิปรายผล และข้อเสนอแนะ

การวิจัยเรื่องการเปรียบเทียบผลสัมฤทธิ์ทางการเรียนวิชาการเขียนโปรแกรม Python ด้วยวิธีสอนแบบ Error-Based Learning บน Jupyter Notebook สำหรับนักศึกษา ปวส. จำนวน 19 คน สามารถสรุปผล อภิปรายผล และให้ข้อเสนอแนะได้ดังนี้

5.1 สรุปผลการวิจัย

1. **ผลสัมฤทธิ์ทางการเรียน:** นักศึกษามีคะแนนเฉลี่ยหลังเรียน ($\bar{x}$ = 17.58) สูงกว่าก่อนเรียน ($\bar{x}$ = 8.42) อย่างมีนัยสำคัญทางสถิติที่ระดับ .05 ซึ่งเป็นไปตามสมมติฐานที่ตั้งไว้
2. **สมรรถนะด้านการเขียนโปรแกรม:** นักศึกษามีสมรรถนะในการวิเคราะห์และแก้ไขข้อผิดพลาด (Error-Based) บน Jupyter Notebook โดยรวมอยู่ใน **ระดับดีมาก** ($\bar{x}$ = 4.63)
3. **ความพึงพอใจ:** นักศึกษามีความพึงพอใจต่อการจัดการเรียนรู้ในภาพรวมอยู่ในระดับมากที่สุด เนื่องจากได้รับอิสระในการทดลองและแก้ปัญหาด้วยตนเอง

5.2 อภิปรายผล

จากผลการวิจัยข้างต้น สามารถอภิปรายประเด็นสำคัญได้ดังนี้:

- **การเรียนรู้ผ่านประสบการณ์ตรง (Experiential Learning):** การที่คะแนนหลังเรียนสูงขึ้นอย่างมาก เป็นเพราะวิธีการสอนแบบ Error-Based Learning สอดคล้องกับทฤษฎีของ Kolb ที่เน้นให้ผู้เรียนเผชิญกับปัญหาจริง (Error) แล้วเกิดกระบวนการสะท้อนคิดเพื่อแก้ไข ทำให้นักศึกษาจดจำตรรกะของภาษา Python ได้แม่นยำกว่าการฟังบรรยายเพียงอย่างเดียว
- **ประสิทธิภาพของเครื่องมือ Jupyter Notebook:** ระบบ Interactive ของ Jupyter Notebook ช่วยให้นักศึกษาได้รับ **Immediate Feedback** หรือผลตอบรับทันทีเมื่อทำการแก้ไขโค้ด ซึ่งส่งผลให้สมรรถนะการแก้ไข Error อยู่ในระดับดีมาก นักศึกษาสามารถระบุตำแหน่งความผิดพลาดจาก Traceback ได้รวดเร็วขึ้น ลดความกลัวในการเขียนโปรแกรมลง
- **การสร้างสมรรถนะอาชีวศึกษา:** ผลการวิจัยสะท้อนให้เห็นว่า การจัดการเรียนรู้ที่เน้นการแก้ปัญหา (Troubleshooting) ช่วยสร้างสมรรถนะที่สอดคล้องกับความต้องการของสถานประกอบการในยุคปัจจุบัน ที่ต้องการบุคลากรสายไอทีที่มีทักษะการแก้ปัญหาที่ซับซ้อนได้

5.3 ข้อเสนอแนะ

1. ข้อเสนอแนะในการนำผลการวิจัยไปใช้

- **ครูผู้สอน:** ควรนำรูปแบบการสอนแบบ Error-Based ไปใช้ในหน่วยการเรียนรู้ที่ยากต่อการทำความเข้าใจ เช่น เรื่อง List, Dictionary หรือการเชื่อมต่อฐานข้อมูล เพื่อช่วยให้ผู้เรียนเห็นภาพการทำงานของข้อมูลผ่านข้อผิดพลาดได้ชัดเจนขึ้น
- **สถานศึกษา:** ควรสนับสนุนการใช้เครื่องมือที่เป็น Open Source อย่าง Jupyter Notebook หรือ Google Colab ในห้องปฏิบัติการคอมพิวเตอร์ เพื่อให้นักศึกษาสามารถฝึกฝนต่อนอกเวลาเรียนได้โดยไม่มีค่าใช้จ่าย

2. ข้อเสนอแนะในการวิจัยครั้งต่อไป

- ควรมีการศึกษาผลของการจัดการเรียนรู้แบบ Error-Based Learning ร่วมกับการใช้เครื่องมือ **Generative AI (เช่น Gemini)** เพื่อเปรียบเทียบว่า AI จะช่วยส่งเสริมสมรรถนะการแก้ปัญหาของผู้เรียนให้รวดเร็วขึ้นอย่างไร
- ควรมีการวิจัยพัฒนา **"คลังโจทย์ข้อผิดพลาด (Error Bank)"** ที่รวบรวมจากปัญหาจริงที่นักศึกษาพบในวิทยาลัยเทคนิคลพบุรี เพื่อใช้เป็นสื่อการสอนมาตรฐานสำหรับแผนกเทคโนโลยีคอมพิวเตอร์ต่อไป

ภาคผนวก
ก.เครื่องมือในการวิจัย

1. แผนจัดการเรียนรู้ฐานสมรรถนะ (รายสัปดาห์)

วิชา: การเขียนโปรแกรมคอมพิวเตอร์ (Python) **หน่วยการเรียนรู้:** โครงสร้างเงื่อนไขและวนซ้ำ

- **สมรรถนะประจำหน่วย:** แสดงความรู้และปฏิบัติการเขียนโปรแกรมควบคุมทิศทางแบบเลือกทำและแบบทำซ้ำ พร้อมทั้งวิเคราะห์และแก้ไขข้อผิดพลาดของโค้ด (Debugging) ได้อย่างถูกต้อง
- **กิจกรรมการเรียนรู้ (4 ขั้นตอน):**
    1. **ขั้นนำ (Motivation):** ครูรันโค้ดที่มี Error ให้นักศึกษาดูและตั้งคำถามว่า "ทำไมโปรแกรมถึงหยุดทำงาน?"
    2. **ขั้นสอน (Information):** อธิบายคำสั่ง if-else, while, for และประเภทของ Error (Syntax, Runtime, Logic)
    3. **ขั้นปฏิบัติ (Application):** นักศึกษาทำใบงาน **Error-Based** บน Jupyter Notebook เพื่อฝึกแก้โจทย์จากสิ่งที่ผิดให้ถูกต้อง
    4. **ขั้นสรุป (Evaluation):** นักศึกษานำเสนอวิธีการแก้ปัญหา และสรุปเทคนิคการ Debugging ที่พบ

2. ชุดใบงาน Error-Based บน Jupyter Notebook

ใบงานที่ 1: การควบคุมทิศทางและตรรกะที่ผิดพลาด

- **โจทย์ (Error Code):**

Python

```
# โจทย์: รับอายุจากผู้ใช้ หากอายุ 20 ปีขึ้นไป ให้พิมพ์ "Adult"
age = input("Enter your age: ")

if age > 20
print("Adult")
else:
print("Minor")
```

- **ภารกิจของนักศึกษา:**
    1. แก้ไข Syntax Error (ลืมเครื่องหมาย : และการย่อหน้า)
    2. แก้ไข Type Error (ลืมแปลง input จาก string เป็น int)
    3. บันทึกผลลัพธ์ที่ถูกต้อง

ข้อที่ 2: การวนซ้ำและการเข้าถึงข้อมูลใน List (IndexError)

- **วัตถุประสงค์:** พิมพ์ชื่อนักศึกษาในรายชื่อทั้งหมดออกมา
- **โค้ดที่มีข้อผิดพลาด:**

Python

```
students = ["Mac", "Baifren", "Somchai"]
```

```
# ต้องการพิมพ์ชื่อทั้ง 3 คน
for i in range(4):
    print("นักศึกษาคนที่", i+1, ":", students[i])
```

- **ภารกิจ:** แก้ไขขอบเขตของ range() หรือการเข้าถึง Index ให้ถูกต้องเพื่อไม่ให้เกิด IndexError: list index out of range

ข้อที่ 3: การคำนวณภาษีและการใช้ตัวแปร (NameError & Logical Error)

- **วัตถุประสงค์:** คำนวณราคาสินค้ารวมภาษี 7%
- **โค้ดที่มีข้อผิดพลาด:**

Python

```
price = 1000
Vat_Rate = 0.07

total_price = price + (price * vat_rate)
print("ราคาสินค้ารวมภาษีคือ:", Total_price)
```

- **ภารกิจ:** แก้ไขการเรียกใช้ตัวแปรให้ตรงกับที่ประกาศไว้ (Python เป็น Case-sensitive) เพื่อแก้ NameError

ข้อที่ 4: การสะสมค่าใน Loop (Logical Error - ผลลัพธ์เพี้ยน)

- **วัตถุประสงค์:** หาผลรวมของเลข 1 ถึง 5 (ผลลัพธ์ที่ถูกต้องต้องได้ 15)
- **โค้ดที่มีข้อผิดพลาด:**

Python

```
total = 0
for i in range(1, 6):
    total = i # บรรทัดนี้มีปัญหาทางตรรกะ

print("ผลรวมของ 1-5 คือ:", total)
```

- **ภารกิจ:** แก้ไขการเก็บสะสมค่าในตัวแปร total ให้เป็นการบวกเพิ่ม (Assignment Operator) เพื่อให้ได้ผลลัพธ์ที่ถูกต้อง

ข้อที่ 5: การตรวจสอบเงื่อนไขหลายชั้น (Syntax & Attribute Error)

- **วัตถุประสงค์:** ตรวจสอบคะแนนเพื่อตัดเกรด A หากได้ 80 ขึ้นไป
- **โค้ดที่มีข้อผิดพลาด:**

Python

```
score = "85"
```

```
if score >= 80:
    print("คุณได้เกรด A")
elif score >= 70
    print("คุณได้เกรด B")
```

- **ภารกิจ:** แก้ไขประเภทข้อมูลจากข้อความเป็นตัวเลข และเติมเครื่องหมายหลัง elif ให้ถูกต้องตามโครงสร้างไวยากรณ์

สรุปการใช้ใบงานในงานวิจัย:

ใบงานทั้ง 6 ข้อ (รวมของเดิม) จะครอบคลุม Error หลักที่นักศึกษา ปวส. มักเจอ คือ:

1. **Syntax Error:** ลืมเครื่องหมาย หรือย่อหน้าผิด
2. **Type Error:** ลืมแปลงชนิดข้อมูล (String/Int)
3. **NameError:** พิมพ์ชื่อตัวแปรผิดหรือเรียกใช้ผิด
4. **IndexError:** อ้างอิงลำดับใน List เกินขอบเขต
5. **Logical Error:** โค้ดรันได้แต่ผลลัพธ์ผิด (ซึ่งเป็นขั้นสูงสุดของการ Debug)

แบบทดสอบวัดผลสัมฤทธิ์ทางการเรียนวิชาการเขียนโปรแกรม Python

คำชี้แจง: จงเลือกคำตอบที่ถูกต้องที่สุด (ข้อละ 1 คะแนน)

ข้อ 1. ข้อใดกล่าวถึงประโยชน์ของระบบ Jupyter Notebook ได้ถูกต้องที่สุดในงานด้านการวิเคราะห์ข้อผิดพลาด?

ก. สามารถรันโค้ดและเห็นผลลัพธ์แยกตามส่วน (Cell) ได้ทันที

ข. ระบบจะช่วยแก้ไข Syntax Error ให้โดยอัตโนมัติ

ค. สามารถป้องกันการเกิด Runtime Error ได้ 100%

ง. ไม่จำเป็นต้องติดตั้งตัวแปลภาษา (Interpreter) ในเครื่อง

ข้อ 2. หากนักศึกษาพบข้อผิดพลาด "IndentationError" หมายความว่ามีสาเหตุมาจากสิ่งใด?

ก. ลืมใส่เครื่องหมายวงเล็บปิด

ข. การตั้งชื่อตัวแปรซ้ำกับคำสงวน

ค. การเว้นวรรคหรือย่อหน้าของ Block คำสั่งไม่ถูกต้อง

ง. การนำตัวเลขมาบวกกับตัวอักษร

ข้อ 3. เมื่อรันโค้ด print(10 / 0) จะเกิดข้อผิดพลาดประเภทใด?

ก. NameError ข. TypeError

ค. ZeroDivisionError ง. Syntax Error

ข้อ 4. ข้อความแจ้งเตือน "NameError: name 'total' is not defined" มักเกิดจากสาเหตุใด?

ก. มีการเรียกใช้ตัวแปรที่ยังไม่ได้ประกาศหรือพิมพ์ชื่อตัวแปรผิด

ข. ตัวแปร 'total' มีค่าเป็นว่าง (Null)

ค. ลืมใส่เครื่องหมายอัญประกาศ (Quotes) ให้กับตัวแปร

ง. ชนิดข้อมูลของตัวแปร 'total' ไม่ถูกต้อง

ข้อ 5. โค้ดใดต่อไปนี้ที่มี Syntax Error (ไวยากรณ์ผิด)?

ก. if x == 5:　　ข. while (x < 10):

ค. if x > 5　　ง. for i in range(5):

ข้อ 6. หากต้องการรับค่าตัวเลขจากผู้ใช้เพื่อมาคำนวณทางคณิตศาสตร์ แต่ลืมใช้ฟังก์ชัน int() จะเกิด Error ประเภทใด?

ก. TypeError (เนื่องจากพยายามคำนวณตัวเลขกับข้อความ)

ข. ValueError (เนื่องจากค่าที่รับมาเป็นช่องว่าง)

ค. Syntax Error (เนื่องจากเขียนคำสั่งรับข้อมูลผิด)

ง. ไม่เกิด Error แต่ผลลัพธ์จะเป็นศูนย์เสมอ

ข้อ 7. ข้อใดคือลักษณะของ "Logic Error" (ข้อผิดพลาดทางตรรกะ)?

ก. โปรแกรมหยุดทำงานทันทีและแจ้งเตือนบรรทัดที่ผิด

ข. โปรแกรมรันได้ปกติจนจบ แต่ให้ผลลัพธ์ที่ไม่ถูกต้องตามโจทย์

ค. โปรแกรมไม่สามารถเริ่มต้นการทำงานได้เลย

ง. โปรแกรมแจ้งเตือนว่าหน่วยความจำเต็ม

ข้อ 8. การแก้ไข Error ประเภท "IndexError: list index out of range" ควรตรวจสอบที่ส่วนใด?

ก. การคำนวณทางคณิตศาสตร์ในโค้ด

ข. การกำหนดค่าเริ่มต้นให้ตัวแปร

ค. ขอบเขตการเข้าถึงลำดับข้อมูลใน List หรือ Array

ง. การเชื่อมต่อฐานข้อมูล

ข้อ 9. ใน Jupyter Notebook หากต้องการไล่ตรวจสอบค่าของตัวแปรทีละขั้นตอน วิธีใดมีประสิทธิภาพที่สุด?

ก. การสั่ง Print ค่าตัวแปรออกมาดูในแต่ละ Cell

ข. การลบโค้ดทิ้งแล้วเขียนใหม่ทั้งหมด

ค. การเปลี่ยนชื่อไฟล์ใหม่

ง. การปิดและเปิดโปรแกรมใหม่

ข้อ 10. ข้อใดคือสมรรถนะที่สำคัญที่สุดของโปรแกรมเมอร์เมื่อเผชิญกับ Error Messages (Traceback)?

ก. การจดจำ Error ทั้งหมดให้ได้

ข. การสามารถอ่านและวิเคราะห์สาเหตุจากข้อความแจ้งเตือนเพื่อนำไปสู่การแก้ไข

ค. การส่งไฟล์ให้ผู้อื่นแก้ไขแทน

ง. การเปลี่ยนไปใช้ภาษาโปรแกรมอื่นที่ไม่มี Error

คำตอบ

ข้อที่	คำตอบ	เหตุผล/คำอธิบาย
1	ก	Jupyter ช่วยให้รันโค้ดทีละส่วน ทำให้หาจุดผิดได้ง่ายขึ้น
2	ค	Indentation คือการย่อหน้า ซึ่ง Python ใช้กำหนด Block คำสั่ง
3	ค	ทางคณิตศาสตร์และคอมพิวเตอร์ไม่สามารถหารด้วยศูนย์ได้
4	ก	เกิดจากการอ้างถึงชื่อที่ Interpreter ไม่รู้จัก
5	ค	หลังเงื่อนไขใน if/elif/else ต้องมีเครื่องหมาย : เสมอ
6	ก	ฟังก์ชัน input() จะคืนค่าเป็น String เสมอ ต้องแปลงเป็น int ก่อนคำนวณ
7	ข	Logic Error คือโค้ดถูกไวยากรณ์แต่ตรรกะผิด ผลลัพธ์จึงไม่ตรงเป้าหมาย
8	ค	เกิดจากการพยายามเข้าถึงลำดับที่ไม่มีอยู่จริงในรายการ (List)
9	ก	เป็นการทำ Debugging เบื้องต้นที่เห็นภาพชัดเจนที่สุดใน Jupyter
10	ข	การวิเคราะห์สาเหตุคือหัวใจของสมรรถนะด้านการเขียนโปรแกรม

ภาคผนวก

ข. แบบประเมินสมรรถนะการแก้ไขข้อผิดพลาด (Error-Based) บน Jupyter Notebook

แบบประเมินสมรรถนะการแก้ไขข้อผิดพลาด (Error-Based) บน Jupyter Notebook

คำชี้แจง: ให้ผู้ประเมิน (ครูผู้สอน) ทำการสังเกตพฤติกรรมและตรวจผลงานของนักศึกษาขณะปฏิบัติงานตามใบงานที่กำหนด และทำเครื่องหมาย ✓ ลงในช่องระดับคะแนนที่ตรงกับความเป็นจริงมากที่สุด

เกณฑ์การให้คะแนน: 5 = ดีมาก, 4 = ดี, 3 = ปานกลาง, 2 = พอใช้, 1 = ปรับปรุง

ที่	รายการประเมินสมรรถนะ (Competency Items)	5	4	3	2	1
1	**การระบุตำแหน่งข้อผิดพลาด (Error Identification)**					
	- สามารถระบุบรรทัดที่เกิด Syntax Error ได้อย่างถูกต้อง					
	- สามารถแยกแยะความแตกต่างระหว่าง Logic Error และ Runtime Error ได้					
2	**การวิเคราะห์สาเหตุจากระบบ (Traceback Analysis)**					
	- สามารถอ่านและแปลความหมายจาก Error Message ใน Jupyter ได้					
	- เข้าใจความหมายของข้อผิดพลาดพื้นฐาน เช่น NameError, TypeError					
3	**ความถูกต้องในการแก้ไขโค้ด (Coding Correction)**					
	- แก้ไขไวยากรณ์ (Syntax) ให้โปรแกรมสามารถรันได้สำเร็จ					
	- ปรับปรุงตรรกะ (Logic) ให้ผลลัพธ์ตรงตามโจทย์ที่กำหนด					
4	**ทักษะกระบวนการและการแก้ปัญหา (Problem Solving)**					
	- ใช้เครื่องมือใน Jupyter (เช่น การแบ่ง Cell รัน) เพื่อไล่หาจุดผิด					
	- มีความรวดเร็วและเป็นระบบในการแก้ไขปัญหาเมื่อเจอ Error ใหม่					
5	**กิจนิสัยในการปฏิบัติงาน (Professional Habits)**					
	- มีความมานะพยายามในการแก้ปัญหา ไม่ท้อถอยเมื่อเจอข้อผิดพลาด					
	- มีการบันทึก Comment อธิบายการแก้ไขโค้ดอย่างเป็นระเบียบ					
	รวมคะแนน					

เกณฑ์การตัดสินระดับสมรรถนะ:

- **4.51 – 5.00:** ระดับดีมาก (สอดคล้องกับผลวิจัยของครู)
- **3.51 – 4.50:** ระดับดี
- **2.51 – 3.50:** ระดับปานกลาง
- **ต่ำกว่า 2.50:** ระดับปรับปรุง