การพัฒนากรอบงานความสามารถด้านการเขียนโปรแกรม จากความถนัดและทักษะ

นายชัยวัฒน์ ฉวีวรรณ

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

บทคัดย่อและแฟ้มข้อมูลฉบับเต็มของวิทยานิพนธ์ตั้งแต่ปีการศึกษา 2554 ที่ให้บริการในคลังปัญญาจุฬาฯ (CUIR)
เป็นแฟ้มข้อมูลของนิสิตเจ้าของวิทยานิพนธ์ ที่ส่งผ่านทางบัณฑิตวิทยาลัย
The abstract and full text of theses from the academic year 2011 in Chulalongkorn University Intellectual Repository (CUIR)
are the thesis authors' files submitted through the University Graduate School.

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2560
ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

# Development of Programming Capability Framework Based on Aptitude and Skill

Mr. Chaiwat Chaweewan

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2017

Copyright of Chulalongkorn University

| | |
|---|---|
| หัวข้อวิทยานิพนธ์ | การพัฒนากรอบงานความสามารถด้านการเขียนโปรแกรมจากความถนัดและทักษะ |
| โดย | นายชัยวัฒน์ ฉวีวรรณ |
| สาขาวิชา | วิทยาศาสตร์คอมพิวเตอร์ |
| อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก | ผู้ช่วยศาสตราจารย์ ดร.อรรถสิทธิ์ สุรฤกษ์ |
| อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม | รองศาสตราจารย์ ดร.อานนท์ รุ่งสว่าง |

---

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ นครทิพย์ พร้อมพูล)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ผู้ช่วยศาสตราจารย์ ดร.อรรถสิทธิ์ สุรฤกษ์)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม

(รองศาสตราจารย์ ดร.อานนท์ รุ่งสว่าง)

.....กรรมการภายนอกมหาวิทยาลัย

(ผู้ช่วยศาสตราจารย์ ดร.โกเมศ อัมพวัน)

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ชัยวัฒน์ ฉวีวรรณ : การพัฒนากรอบงานความสามารถด้านการเขียนโปรแกรม จากความถนัดและทักษะ (Development of Programming Capability Framework Based on Aptitude and Skill) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: ผศ. ดร.อรรถสิทธิ์ สุรฤกษ์, อ.ที่ปรึกษาวิทยานิพนธ์ร่วม: รศ. ดร.อานนท์ รุ่งสว่าง, 50 หน้า.

การเรียนการสอนการเขียนโปรแกรมจะมุ่งเน้นให้ผู้เรียนสามารถเขียนโปรแกรมเพื่อนำไปใช้แก้ไขปัญหาต่าง ๆ ที่สามารถแก้ด้วยภาษาโปรแกรมได้อย่างมีประสิทธิภาพ ทางผู้สอนจึงมักจะสอนการเขียนโปรแกรมในรูปแบบต่าง ๆ รวมถึงให้คำแนะนำการใช้เครื่องมือทางภาษาโปรแกรมให้แก่ผู้เรียน เพื่อให้ผู้เรียนมีทักษะในการเขียนโปรแกรมเพื่อแก้ไขปัญหาที่หลากหลาย แต่เนื่องจากผู้เรียนแต่ละคนนั้นมีพื้นฐาน และความถนัดในการเขียนโปรแกรมที่แตกต่างกัน จึงมีความจำเป็นที่จะต้องมีการค้นหาความถนัด และประเมินทักษะของผู้เรียน เพื่อเป็นข้อมูลให้กับครูผู้สอนรับทราบถึงทักษะของผู้เรียนแต่ละคน และยังทำให้สามารถเข้าไปเสริมในเรื่องที่ผู้เรียนเกิดความไม่เข้าใจให้มีความเข้าใจมากขึ้นเพื่อที่จะสามารถนำความรู้ที่ได้ไปใช้ในการเขียนโปรแกรมได้อย่างมีประสิทธิภาพ

ในงานวิจัยชิ้นนี้ได้มีการนำเสนอแนวคิด และวิธีการในการค้นหาความถนัด และประเมินทักษะในการเขียนโปรแกรม โดยที่ทางด้านการค้นหาความถนัดนี้จะพิจารณาจาก 4 คุณลักษณะได้แก่ 1) ผลจากการเปรียบเทียบระหว่างวัตถุประสงค์การสอนกับโปรแกรมที่ผู้เรียนการเขียนโปรแกรมเขียนขึ้น 2) การใช้เครื่องมือทางภาษาเขียนโปรแกรม 3) หมายเหตุที่ได้จากการตรวจ และ 4) ความยากง่ายทางการอ่านของรหัสต้นฉบับ และทางด้านการประเมินทักษะทางด้านการเขียนโปรแกรมจะพิจารณาจาก เวลาที่ผู้เรียนการเขียนโปรแกรมใช้ไปในการเขียนโปรแกรม 1 ข้อ และจำนวนครั้งของการส่งโปรแกรม โดยผลลัพธ์ที่ได้จากงานวิจัยชิ้นนี้คือแนวคิด และวิธีการในการค้นหาความถนัด และประเมินทักษะในการเขียนโปรแกรมที่สามารถนำไปปรับใช้ในการเรียนการสอนการเขียนโปรแกรมได้ อีกทั้งผลลัพธ์ที่ได้จากการค้นหาความถนัด และประเมินทักษะทางด้านการเขียนโปรแกรมยังเป็นข้อมูลอันเป็นประโยชน์ต่อครูผู้สอนการเขียนโปรแกรมในการเสริมสร้างทักษะในการเขียนโปรแกรมให้แก่ผู้เรียนเขียนโปรแกรม

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

| | | | |
|---|---|---|---|
| ภาควิชา | วิศวกรรมคอมพิวเตอร์ | ลายมือชื่อนิสิต | ..... |
| สาขาวิชา | วิทยาศาสตร์คอมพิวเตอร์ | ลายมือชื่อ อ.ที่ปรึกษาหลัก | ..... |
| ปีการศึกษา | 2560 | ลายมือชื่อ อ.ที่ปรึกษาร่วม | ..... |

# # 5770149421 : MAJOR COMPUTER SCIENCE
KEYWORDS: PROGRAMMING / CAPABILITY / APTITUDE / SKILL

CHAIWAT CHAWEEWAN: Development of Programming Capability Framework Based on Aptitude and Skill. ADVISOR: ASST. PROF. ATHASIT SURARERKS, Ph.D., CO-ADVISOR: ASSOC. PROF. ARNON RUNGSAWANG, Ph.D., 50 pp.

Programming courses aim at making students be able to create programs that can efficiently solve problems using programming languages. Therefore, the instructors provide various forms of programming courses and guidelines to use the programming languages tools to the students in order to enable them to solve various problems by writing a program. However, different students have different backgrounds and programming aptitudes. Thus, it is necessary to find the student aptitude and evaluate their skills in order to obtain an information for the instructors to strengthen the points that students do mostly not understand so that students themselves are able to efficiently improve their programming capability.

This research introduces concepts and methodologies for aptitude finding and skills evaluation in writing the computer programs. The aptitude finding is based on considering 4 main features i.e. 1) the result of an objective comparison, 2) programming tools, 3) grader comment and 4) source code readability, respectively. Meanwhile, the skill evaluation takes into account time consumption on writing a program and frequency of code submission to the grader system. The result of this research are the concept and methods that can be applied to find student's programming aptitude and to evaluate students' programming skill. Moreover, the results of aptitude finding and skill evaluation will be beneficial for the instructors to improve the students' programming capability during a programming course.

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

| | | |
|---|---|---|
| Department: | Computer Engineering | Student's Signature ..... |
| Field of Study: | Computer Science | Advisor's Signature ..... |
| Academic Year: | 2017 | Co-Advisor's Signature ..... |

# กิตติกรรมประกาศ

ขอขอบคุณ ผศ.ดร.อรรถสิทธิ์ สุรฤกษ์ ที่ได้มอบวิชาความรู้อันเป็นประโยชน์ และคำปรึกษาในการทำงานวิจัยชิ้นนี้จนสำเร็จด้วยดี

ขอขอบคุณ รศ.ดร.อานนท์ รุ่งสว่าง และผศ.ดร.บัณฑิต มนัสเกษมศักดิ์ ที่ได้ให้คำแนะนำและแนวทางการแก้ไขปัญหาที่เกิดขึ้นกับงานวิจัยชิ้นนี้ให้ลุล่วงผ่านไปได้ด้วยดี

ขอขอบคุณ ผศ.นครทิพย์ พร้อมพูล ที่ได้สละเวลามาเป็นประธานกรรมการในการสอบป้องกันวิทยานิพนธ์

ขอขอบคุณ ผศ.ดร.โกเมศ อัมพวัน ที่ได้ให้คำแนะนำที่เป็นประโยชน์เกี่ยวกับงานวิจัย และสละเวลามาเป็นกรรมการในการสอบป้องกันวิทยานิพนธ์

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

# สารบัญ

หน้า




จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

หน้า





จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

# สารบัญตาราง





จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

# สารบัญภาพ



จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

# บทที่ 1
# บทนำ

## 1.1 ที่มาของงานวิจัย

ทางด้านการเรียนการสอนเขียนโปรแกรมในปัจจุบันนั้นมุ่งเน้นไปที่การเรียนการสอนเพื่อนำความรู้ที่เรียนไปประยุกต์ใช้ในด้านต่าง ๆ ไม่ว่าจะเป็นทางด้านการเรียน การทำงาน หรือแม้กระทั่งชีวิตประจำวัน โดยในระหว่างที่ได้มีการเรียนการสอนการเขียนโปรแกรมนั้นอาจไม่ได้มีการประเมินหรือการบันทึกประวัติว่าผู้เรียนการเขียนโปรแกรมมีลักษณะการเขียนโปรแกรมแบบใด มีความถนัดอะไรบ้าง หรือแม้แต่การประเมินทักษะเพื่อที่จะแสดงถึงความก้าวหน้าระหว่างการเรียนที่มากพอ ทำให้ไม่สามารถบ่งบอกได้ว่าผู้เรียนมีทักษะการเขียนโปรแกรมเป็นอย่างไร ถนัดการใช้เครื่องมือทางภาษาอะไรบ้าง หรือไม่ถนัดการใช้เครื่องมือทางภาษาอะไรบ้าง ผู้เรียนได้ทำการเขียนโปรแกรมตามที่ครูผู้สอนได้สอนไปหรือไม่อย่างไร ทำให้ไม่สามารถเข้าไปเสริมในส่วนที่ผู้เรียนไม่ถนัดทางด้านการเลือกใช้เครื่องมือทางภาษาเขียนโปรแกรมให้ผู้เรียนมีความถนัดเพิ่มมากขึ้น เพื่อความหลากหลายในการเขียนโปรแกรมเนื่องจากการเขียนโปรแกรมจริง ๆ นั้น หนึ่งโปรแกรมสามารถที่จะเขียนได้มากกว่าหนึ่งรูปแบบซึ่งในแต่ละรูปแบบก็จะมีข้อดีข้อเสียต่างกันไป เช่น ความต่างของระยะเวลาที่โปรแกรมใช้ประมวลผลก่อนที่จะส่งคืนค่าผลลัพธ์ออกมาให้กับผู้ใช้ ในส่วนนี้รูปแบบวิธีการในการเขียนโปรแกรมนั้นถือว่ามีส่วนเป็นอย่างมาก เพราะถ้าเขียนด้วยรูปแบบที่เหมาะสมแล้วก็จะสามารถทำให้โปรแกรมที่ผู้เขียนเขียนขึ้นมานั้นทำงานได้อย่างรวดเร็ว และยังมีประสิทธิภาพมากขึ้นตามไปด้วย ซึ่งในจุดนี้การค้นหาความถนัดในการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรมเป็นสิ่งสำคัญที่จะบอกว่าผู้เขียนถนัดในการใช้เครื่องมือทางภาษาเขียนโปรแกรมอะไรบ้าง หรือไม่ถนัดการใช้เครื่องมือทางภาษาเขียนโปรแกรมอะไรบ้าง ควรเสริมจุดไหนเพื่อให้ผู้เรียนการเขียนโปรแกรมมีพัฒนาการทางด้านการเรียนเขียนโปรแกรมที่ดีขึ้น และในจุด ๆ นี้ยังส่งผลกระทบไปถึงทักษะโดยตรงของผู้เขียน เนื่องจากที่ถ้ารู้ว่าผู้เขียนไม่ถนัดอะไรแล้วไปเสริมในจุด ๆ นั้นอย่างตรงประเด็นก็จะส่งผลทำให้ผู้เขียนได้เกิดการเรียนรู้ และพัฒนาในส่วนที่ตัวผู้เรียนการเขียนโปรแกรมมีความถนัดเพิ่มขึ้นมาทำให้ตัวผู้เรียนเองได้มีทักษะในการเขียนโปรแกรมมากยิ่งขึ้น และเมื่อผู้เรียนการเขียนโปรแกรมมีทักษะมากขึ้นแล้ว ก็จะส่งผลให้เกิดความชำนาญในการเขียนโปรแกรมที่มากขึ้นตามไปด้วยและในท้ายที่สุดเมื่อผู้เรียนได้เจอกับโจทย์ปัญหาใหม่ ๆ ที่จะเข้ามาในอนาคตแล้ว ก็จะสามารถเขียนโปรแกรมได้อย่างมีประสิทธิภาพ และใช้เวลาในการเขียนโปรแกรมที่น้อยลงตามไปด้วย

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

จากที่ได้กล่าวมาในข้างต้นงานวิจัยในวิทยานิพนธ์นี้จึงมีแนวความคิดที่จะทำการค้นหาความถนัดของผู้เรียนการเขียนโปรแกรม และการประเมินทักษะการเขียนโปรแกรม ของผู้เรียนการเขียนโปรแกรม โดยแนวความคิดนี้ได้เริ่มมากจากการศึกษาทางด้านการตรวจสอบความเหมือนกันของรหัสต้นฉบับของโปรแกรม (program source code) ที่เริ่มมาจากการตรวจสอบการคัดลอก (plagiarism) จากงานวิจัยที่ได้รับการตีพิมพ์เมื่อปี 1989 [1] ที่ว่าด้วยการตรวจสอบการคัดลอกของรหัสต้นฉบับของโปรแกรมโดยใช้การนับจำนวนคุณลักษณะ (features) ที่ในช่วงเวลานั้นถูกเรียกว่าการวัดชุดคำสั่ง (program measure) ภายในโปรแกรมที่ถูกเขียนขึ้นโดยใช้ภาษาซี (C programming language) ต่อมาได้มีการเสนอการตรวจสอบการคัดลอกของรหัสต้นฉบับของโปรแกรมโดยใช้การตรวจสอบการเปลี่ยนแปลงของคำ หรือคำศัพท์ที่ใช้ภายในตัวโปรแกรม (lexical changes) ร่วมกับการเปลี่ยนแปลงทางด้านโครงสร้างของตัวโปรแกรม (structural change) และการเปลี่ยนแปลงทางด้านความหมายของคำ (sematic) [2], [3], [4], [5], [6]โดยการตรวจสอบโดยใช้เปลี่ยนแปลงทั้งสองด้านที่ว่ามาเพื่อที่จะแก้ปัญหาในกรณีที่มีการคัดลอกรหัสต้นฉบับของโปรแกรมมาแล้วมีการปรับเปลี่ยนการใช้คำ อาทิเช่น การเปลี่ยนชื่อตัวแปร หรือการเปลี่ยนลำดับของบรรทัดของชุดคำสั่งภายในรหัสต้นฉบับของโปรแกรม เป็นต้น หรือปรับเปลี่ยนโครงสร้าง อาทิเช่น การเปลี่ยนชนิดของวังวน (loop) จากวังวน while ไปเป็นวังวน for หรือการเปลี่ยนลับดับของตัวดำเนินการ (operands) อาทิเช่น “x < y” เป็น “y >= x” เป็นต้น แล้วนำไฟล์ที่ได้รับการแก้ไขนั้นไปส่งใหม่ ต่อมาได้มีการเสนอแนวทางการตรวจสอบการคัดลอกกันของรหัสต้นฉบับของโปรแกรมโดยใช้การหาลายนิ้วมือ (Fingerprint) ของรหัสต้นฉบับของโปรแกรมที่มีชื่อว่า “Winnowing algorithm” [7] ต่อมาได้มีการนำเสนอการตรวจสอบการคัดลอกของรหัสต้นฉบับออกมาหลากหลายวิธี อาทิเช่น การเปลี่ยนรหัสต้นฉบับเป็นโทเค็น (token) [8] และนำโทเค็นดังกล่าวมาเปรียบเทียบกับรหัสต้นฉบับอื่น ๆ โดยการใช้ขั้นตอนวิธีที่มีชื่อว่า คาร์บ-ราบิน (Karp-Rabin algorithm) ที่ว่าด้วยการเปรียบเทียบสายอักขระด้วยการนำเอาสายอักขระทั้งสายอัขระที่เป็นต้นฉบับ และสายอักขระที่จะนำมาเปรียบเทียบไปเข้าฟังก์ชั่นแฮช (hash function) เพื่อที่จะแปลงสายอักขระให้เป็นค่าที่เป็นตัวเลขเพื่อที่จะนำค่าเหล่านี้ไปค้นหาภายในชุดลำดับของค่าที่ได้จากฟังก์ชั่นแฮชโดยการสร้างหน้าต่างขนาดที่เท่ากับขนาดของสายอักขระต้นฉบับ โดยจะทำการเลื่อนหน้าต่างดังกล่าวไปทีละ 1 ตำแหน่งจนพบค่าที่เหมือนกัน (match) จึงจะรายงานผลการจับคู่ออกมาโดยที่ผลการจับคู่นี้จะสะท้อนให้เห็นถึงจำนวนของการคัดลอกของชุดคำสั่งของโปรแกรม การใช้การเปรียบเทียบระยะทางเปลี่ยนแปลงของโหนดภายในต้นไม้ (tree) [9], [10], [11] โดยการใช้การหาระยะทางการเปลี่ยนแปลงของจำนวนปม (node) ภายในต้นไม้จากต้นหนึ่งไปสู่อีกต้นหนึ่งซึ่งการเปลี่ยนแปลงที่จะนำมาพิจารณาจะประกอบไปด้วย การแทรกของปมภายในต้นไม้ (insertion) การหลุดหายไปของปมภายในต้นไม้ (deletion) และการเปลี่ยนชื่อของปมภายในต้นไม้ (renaming) โดยจะทำการนับจำนวนครั้งของการเปลี่ยนแปลงของต้นไม้จากต้นหนึ่งไปสู่อีกต้นหนึ่ง

ซึ่งจำนวนครั้งของการเปลี่ยนแปลงของต้นไม้นี้จะสะท้อนให้เห็นถึงความเหมือน (similarity) ระหว่างต้นไม้ 2 ต้น ต่อมาได้มีการนำเสนอการเปรียบเทียบรหัสต้นฉบับของโปรแกรมโดยใช้กราฟ [12] ด้วยการใช้ขั้นตอนวิธีที่มีชื่อว่า การเปรียบเทียบ และจับคู่กราฟ (graph similarity scoring and matching) และภายในเวลาต่อมาได้มีการนำเสนอขั้นตอนวิธีการการเปรียบเทียบรหัสต้นฉบับของโปรแกรมโดยใช้การเปรียบเทียบคุณลักษณะ (features) ของโปรแกรม [13], [14], [15] โดยเฉพาะการเปรียบเทียบรหัสต้นฉบับโดยการใช้การสกัดคุณลักษณะนั้นเป็นเทคนิคที่มีความน่าสนใจเนื่องจากคุณลักษณะดังกล่าวสามารถนำออกมาวิเคราะห์เพื่อบ่งบอกถึงสิ่งที่ผู้เขียนโปรแกรมใช้ในการเขียนโปรแกรมตลอดช่วงระยะเวลาหนึ่งได้ หรืออาจเรียกได้ว่าเป็นความถนัดทางด้านการเขียนโปรแกรมของผู้เขียนตลอดหนึ่งช่วงระยะเวลาการเขียนโปรแกรม นอกจากนี้ยังได้นำคุณลักษณะทางด้านความยากง่ายทางการอ่านของชุดคำสั่งของโปรแกรม [15] มาใช้ในการวิเคราะห์ถึงความยากง่ายทางการอ่านของรหัสต้นฉบับของโปรแกรมของผู้เขียนที่เขียนขึ้นมาว่าเขียนขึ้นมาแล้วมีความยากง่ายทางการอ่านมากน้อยเพียงใด

โดยในงานวิจัยชิ้นนี้มุ่งเน้นในการนำเสนอแนวคิด และวิธีการในการค้นหาความถนัดในการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรม เพื่อเป็นข้อมูลให้ครูผู้สอน หรือผู้ช่วยสอนการเขียนโปรแกรมได้รับรู้ถึงความถนัดของผู้เรียนแต่ละคนเพื่อที่จะได้เสริมสร้างความถนัดเพิ่มเติมจากสิ่งที่ผู้เรียนถนัดอยู่แล้วให้เกิดความถนัดที่หลากหลายมากยิ่งขึ้น หรือแนะนำในสิ่งที่ผู้เรียนการเขียนโปรแกรมเพื่อให้เกิดประโยชน์โดยตรงกับผู้เรียนการเขียนโปรแกรม ทั้งนี้ยังได้นำเสนอวิธีการในการประเมินทักษะการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรมจากเปรียบเทียบเวลาที่ใช้ไปในการเขียนโปรแกรมในแต่ละข้อร่วมกับการพิจารณาจำนวนครั้งของการส่งโปรแกรมขึ้นตรวจในแต่ละข้อเพื่อที่จะสามารถบ่งบอกได้ถึงทักษะของผู้เรียนแต่ละคนว่าผู้เรียนแต่ละคนมีทักษะการเขียนโปรแกรมเป็นอย่างไรอีกทั้งยังสามารถบ่งบอกถึงพัฒนาการของผู้เรียนการเขียนโปรแกรมได้ตลอดระยะเวลาที่มีการเรียนการสอนเขียนโปรแกรมได้อย่างต่อเนื่อง

### 1.2 แนวทางการพัฒนางานวิจัย

1) ศึกษาวิธีการสกัดข้อมูลออกมาจากไฟล์ข้อมูลการเขียนโปรแกรม
2) ศึกษาวิธีการคิดคำนวณเวลารวมที่ใช้ไปในการเขียนโปรแกรม
3) ศึกษาการวิเคราะห์ข้อมูลความถนัด และประเมินทักษะ
4) กำหนดวิธีในการประเมินทักษะการเขียนโปรแกรม

5) ออกแบบระบบที่ใช้ในการค้นหาความถนัด และประเมินทักษะทางด้านการเขียนโปรแกรมวิเคราะห์ และสรุปผลลัพธ์ที่ได้จากระบบ

**1.3 วัตถุประสงค์ของงานวิจัย**

1) เพื่อค้นหาแนวทางในการค้นหาความถนัดในการเขียนโปรแกรมของผู้เขียนแต่ละคน
2) เพื่อค้นหาแนวทางในการประเมินทักษะในการเขียนโปรแกรมจากโปรแกรมที่เขียนขึ้นของผู้เขียนแต่ละคน
3) เพื่อเป็นแนวทางในการวิเคราะห์ และอธิบายข้อมูลจากทั้งทางด้านความถนัด และทักษะทางด้านการเขียนโปรแกรม

**1.4 ขอบเขตของงานวิจัย**

1) ข้อมูลที่นำมาทำการวิจัยเป็นข้อมูลการเขียนโปรแกรมที่ถูกเขียนขึ้นโดยใช้ภาษาจาวา (java) เท่านั้น
2) ข้อมูลที่นำมาทำการวิจัยเป็นข้อมูลการเรียนการสอนการเขียนโปรแกรมของปีการศึกษา 2555 เท่านั้น

**1.5 ข้อจำกัดของงานวิจัย**

1) ข้อมูลที่นำมาทำการทดลองนั้นเป็นข้อมูลที่มีแผ่นแบบ (template) ส่งผลทำให้ไม่สามารถสกัดข้อมูลที่เป็นประโยชน์ต่อการค้นหาความถนัด และประเมินทักษะในการเขียนโปรแกรมได้เท่าที่ควร

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

**1.6 ประโยชน์ที่คาดว่าจะได้รับ**

1) สามารถหาความถนัดของการเขียนโปรแกรมจากข้อมูลการเขียนโปรแกรมได้
2) สามารถประเมินทักษะของผู้เขียนโปรแกรมจากข้อมูลการเขียนโปรแกรมได้
3) สามารถนำแนวคิดและวิธีการที่นำเสนอไปใช้ในการค้นหาความถนัด และประเมินทักษะจากการเขียนโปรแกรมภาษาอื่น ๆ ได้ และเป็นแนวทางให้กับผู้ที่สนใจศึกษา หรือวิจัยทางด้านการวิเคราะห์ข้อมูลได้นำไปใช้เพื่อต่อยอดงานวิจัยต่อไปได้

# บทที่ 2
# ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในหัวข้อจะกล่าวถึงเรื่องทฤษฎีต่าง ๆ ที่นำมาใช้ในวิทยานิพนธ์ชิ้นนี้โดยทฤษฎีที่จะนำมาใช้นั้นเป็นทฤษฎีที่เกี่ยวข้องกับกรรมวิธีในการนำเอาข้อมูลต่าง ๆ ที่อยู่ในรหัสต้นฉบับออกมาประมวลผลและวิเคราะห์ โดยทฤษฎีที่จะถูกนำมาใช้นั้นจะประกอบไปด้วย 2 ทฤษฎี ได้แก่ 1. การสกัดคุณลักษณะ (feature extraction) และ 2. การประเมินความยากง่ายทางการอ่านของรหัสต้นฉบับของโปรแกรม (readability) โดยจะขออธิบายตามลำดับ ดังนี้

**2.1 การสกัดคุณลักษณะ** (feature extraction) [1], [13], [14] เป็นวิธีที่นำมาใช้กันอย่างแพร่หลายในการสกัดเอาข้อมูลที่จำเป็นออกมาจากรหัสต้นฉบับ (source code) ของโปรแกรมออกมาวิเคราะห์ โดยสามารถกระทำได้หลากหลายวิธี แต่ในวิทยานิพนธ์ชิ้นนี้ได้มีการหยิบเอาวิธีการสกัดคุณลักษณะออกมาใช้ด้วยกัน 2 วิธีคือ 1. การค้นหาจากคำหรือกลุ่มคำที่ได้เตรียมไว้ (scanning from reserved word) และ 2. การนับจำนวน (counting) โดยจะขออธิบายเป็นข้อ ๆ ดังนี้

2.1.1 การค้นหาจากคำหรือกลุ่มคำที่ได้เตรียมไว้เป็นการกำหนดกำหนดคำหลัก (keyword) สำหรับใช้ในการค้นหา โดยคำหลักเหล่านี้จะสื่อถึงสิ่งที่เราต้องการที่จะค้น เช่น การค้นหาชนิดของวังวน (loop) ที่ใช้ภายในรหัสต้นฉบับของโปรแกรม ก็จะประกอบไปด้วยคำหลักดังนี้ for, while, do หรือจะค้นหาชนิดของการเลือกทำที่ใช้ในรหัสต้นฉบับของโปรแกรม ก็จะประกอบไปด้วยคำหลักดังนี้ if, else-if, else เป็นต้น โดยคำหลักดังกล่าวอาจถูกเก็บไว้ในรูปแบบเอกสาร หรือเก็บไว้เป็นเซตของแถวลำดับ (array) เพื่อความสะดวกในการเข้าถึง และแก้ไขโดยผลลัพธ์ที่ออกมาจากการค้นหาจากกลุ่มคำที่ได้กำหนดไว้ (reserved word) อาจมีด้วยกันหลายรูปแบบ เช่น พบ หรือ ไม่พบ หรือ ตำแหน่งที่พบ เป็นต้น ทั้งนี้ขึ้นอยู่กับจุดประสงค์ในการนำผลลัพธ์ไปใช้ว่าต้องการที่จะนำไปใช้ทำอะไร เช่น ต้องการตรวจสอบว่ามีการใช้การเลือกทำ (conditional statements) แบบใดในรหัสต้นฉบับของโปรแกรม ก็จะใช้ผลลัพธ์ที่อยู่ในรูปของการพบ หรือไม่พบ เป็นต้น

2.1.2 การนับจำนวน โดยการสกัดคุณลักษณะแบบนี้จะใช้ร่วมกันกับการค้นหาคำหรือกลุ่มคำที่ได้เตรียมไว้ที่ได้อธิบายไปในข้างต้นโดยหลังจากที่ได้ทำการค้นหา และพบคำหลักที่ได้กำหนดไว้แล้ว จะเริ่มทำการค้นหาต่อไปเพื่อทำการนับจำนวนที่พบคำหลัก หรือสิ่งที่ได้กำหนดไว้ในการนับจำนวนหลังจากที่พบคำหลักคำนั้น ๆ ปรากฏอยู่ เช่น การตรวจสอบลักษณะของการประกาศตัวแปร ก็จะทำ

การค้นหาบรรทัดที่มีการประกาศตัวแปร โดยใช้คำหลักเป็นชนิดของตัวแปรต่าง ๆ เช่น int, double, string เป็นต้น และเมื่อพบคำหลักดังกล่าวปรากฏ ณ ตำแหน่งใด ๆ ของรหัสต้นฉบับของโปรแกรมแล้ว จะทำการค้นหาเครื่องหมาย จุลภาค “,” (comma) หรือเครื่องหมายอัฒภาค “;” (semicolon) เพื่อที่จะนับจำนวนเครื่องหมายดังกล่าว เพื่อที่จะนำจำนวนของเครื่องหมายดังกล่าวมาบอกประเภทของลักษณะการประกาศตัวแปรของรหัสต้นฉบับของโปรแกรมนั้นๆ

งานวิจัย [1], [13], [14] ได้มีการใช้ทฤษฎีทางด้านการสกัดคุณลักษณะของโปรแกรมออกมานั้น ได้เริ่มต้นจากงานวิจัยของ อลัน พาร์คเกอร์ และคณะ (Alan Parker et al.) [1] ที่ได้มีการเสนอการตรวจสอบการคัดลอกของรหัสต้นฉบับของโปรแกรมเมื่อปี 1989 ที่ว่าด้วยการเปรียบเทียบขั้นตอนวิธี (algorithm) สำหรับการตรวจสอบการคัดลอกของรหัสต้นฉบับที่ประกอบด้วยขั้นตอนวิธีจำนวน 7 ขั้นตอนวิธี โดยในแต่ละขั้นตอนวิธีนั้นก็จะประกอบไปด้วยคุณลักษณะที่ถูกสกัดออกมาเพื่อทำการตรวจสอบการคัดลอกของรหัสต้นฉบับโดยที่ในเอกสารงานวิจัยเรียกว่าการวัดชุดคำสั่ง (program measure) ซึ่งในแต่ละขั้นตอนวิธีจะประกอบไปด้วยคุณลักษณะต่าง ๆ ดังตารางที่ 2.1

ตารางที่ 2.1 รายชื่อคุณลักษณะที่ถูกใช้ในการเปรียบเทียบขั้นตอนวิธีในงานวิจัยของอลัน พาร์คเกอร์และคณะ

| | Algorithm Number | | | | | | |
|---|---|---|---|---|---|---|---|
| **Program Measure** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| Number of characters per line | | | | | | m1 | z1 |
| Number of comment lines | | a3 | | f2 | | m2 | |
| Number of indent lines | | a20 | | | | m3 | |
| Number of blank line | | | | | | m4 | |
| Average procedure/function length | | | | | | m5 | |
| Number of reserve words | | | | f4 | c3 | m6 | y3, z2 |
| Average identifier length | | | | | | m7 | |
| Average space percentage per line | | | | | c2 | m8 | |
| Number of labels and gotos | | a15 | | | | m9 | |
| Unique operances | n2 | a17, a6, a9, a10, a11 | b1 | f5, f6, f10, f11, f15 | c4 | m10 | y2, z3 |
| Numberof program interval | | | | | | m11 | |
| Number of color used in coloring the control graph | | | | | | m12 | |

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ตารางที่ 2.1 รายชื่อคุณลักษณะที่ถูกใช้ในการเปรียบเทียบขั้นตอนวิธีในงานวิจัยของอลัน พาร์คเกอร์ และคณะ(ต่อ)

| | Algorithm number | | | | | | |
|---|---|---|---|---|---|---|---|
| **Program Measure** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| Number of vertices colored with color3 in coloring the controll graph | | | | | | m13 | |
| Number of vertics colored with color4 in coloring the controll graph | | | | | | m14 | |
| Total Operands | N2 | a19 | b6 | f7, f13 | | m15 | y4 |
| Unique Operators | n1 | a16, a5 | | f8, f9, f14 | | m16 | y4 |
| Total Operators | N1 | a18 | | f12 | | m17 | |
| Program factor structure percentage | | | | | | m18 | |
| Program impority percentage | | a7 | | | | m19 | |
| Module contribution percentage | | a8 | b7 | | | m20 | y10 |
| Number of module | | | b2 | | c6 | m21 | |
| Conditional statement percentage | | | b4 | | | m22 | y5, y8, z4, z7 |
| Repetitive statement percentage | | a12, a13, a14 | b5 | f3 | | m23 | y6, y7, z5, z6 |
| Number of program statement | | a1, a2 | b8 | f1 | c5, c1 | | y10 |
| Reference sequence order | | | | | | | z9 |
| Multiple statement lines | | a4 | | | | | |

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

โดยที่ตัวอักษรที่อยู่ในแต่ละแนวตั้ง (column) คือคุณลักษณะของขั้นตอนวิธีนั้น ๆ ที่ใช้ในการตรวจสอบการคัดลอกของรหัสต้นฉบับโดยงานวิจัยชิ้นนี้ได้แสดงให้เห็นถึงคุณลักษณะต่าง ๆ ที่ใช้ในขั้นตอนวิธีในแต่ละขั้นตอนวิธีในการตรวจสอบการคัดลอกของรหัสต้นฉบับของโปรแกรม

ต่อมาภายในปี 2012 อารับยาโมฮามาดี และคณะ (S. Arabyarmohamady et al.) [13] ได้นำเสนองานวิจัยที่เกี่ยวกับการตรวจสอบการคัดลอกของชุดคำสั่งของโปรแกรมจากคุณลักษณะของโปรแกรม (program features) ซึ่งงานวิจัยชิ้นนี้ได้มีการใช้การสกัดคุณลักษณะของโปรแกรมโดย

คุณลักษณะของโปรแกรมที่ถูกสกัดออกมานั้นมีหลากหลายคุณลักษณะ โดยจะขอยกตัวอย่างให้เห็นทั้งหมด 4 คุณลักษณะ ได้แก่ 1. การเขียนชุดคำสั่งให้อยู่ในบรรทัดเดียวกัน (each command in one line) หรือมีการขึ้นบรรทัดใหม่ในกรณีที่มีชุดคำสั่งใหม่เกิดขึ้น (several command in one line) 2. การเลือกใช้ประเภทของหมายเหตุ (comment) ระหว่างหมายเหตุแบบกลุ่ม (block comment) หรือหมายเหตุแบบเดี่ยว (single comment) 3. การใช้เครื่องหมายเส้นใต้อักขระ (underscore) ภายในชื่อตัวแปร และ 4. การใช้การเว้นวรรค (space) ก่อน และหลังใช้เครื่องหมายพิเศษ ยกตัวอย่างเช่น เครื่องหมายบวก เครื่องหมายลบ เครื่องหมายดอกจัน หรือเครื่องหมายเท่ากับ เป็นต้น โดยที่คุณลักษณะเหล่านี้จะถูกนำไปรวมกันเพื่อทำเป็นเวกเตอร์คุณลักษณะ (features vector) โดยที่เวกเตอร์ของคุณลักษณะเหล่านี้จะสามารถบ่งบอกได้ว่า ผู้เขียนโปรแกรมได้เขียนโปรแกรมออกมาในลักษณะใดบ้าง หลังจากที่ได้เวกเตอร์ของคุณลักษณะแล้วก็จะนำเอาเวกเตอร์ดังกล่าวไปเก็บไว้เป็นประวัติการเขียนโปรแกรม หรือที่ในงานวิจัยเรียกว่าโพรไฟล์ผู้เขียน (author's profile) หลังที่ได้มีการจัดเก็บโพรไฟล์ของผู้เขียนเสร็จแล้วเวลาที่มีโปรแกรมที่ต้องการที่จะตรวจสอบเข้ามา ก็จะทำการสกัดคุณลักษณะของโปรแกรมแล้วนำมาสร้างเวกเตอร์คุณลักษณะของโปรแกรม และจะนำเวกเตอร์คุณลักษณะเหล่านั้นมาเปรียบเทียบหาความเหมือนกัน (similarity comparison) โดยใช้การคำนวณระยะทางแบบยูคลิด (Euclidean distance) กับโพรไฟล์ผู้เขียนทั้งหมดที่ถูกจัดเก็บไว้ในระบบ และดูผลลัพธ์ที่ออกมาว่ามีค่าความเหมือนกันกับโพรไฟล์ของผู้เขียนคนใดที่อยู่ในระบบ โดยที่จะมีการตั้งเกณฑ์การตัดสินค่าความเหมือนกันไว้ตัดสินว่ามีการคัดลอกกันหรือไม่ จากค่าความเหมือนที่ถูกคำนวณออกมา และนอกจากนี้ยังได้นำเอาเวกเตอร์คุณลักษณะของโปรแกรมที่นำเข้ามาตรวจ กับโพรไฟล์ผู้เขียนไปเข้า SVM (support vector machine) เพื่อคำนวณหาความเป็นไปได้ของการคัดลอก (likelihood of plagiarism) โดยงานวิจัยชิ้นนี้ได้นำเสนอขั้นตอนวิธีการในการตรวจสอบการคัดลอกของรหัสต้นฉบับของโปรแกรม โดยใช้คุณลักษณะของโปรแกรมมาเป็นตัวแทนของโปรแกรมที่ผู้เขียนเขียนขึ้นมา เพื่อใช้ในการคำนวณค่าความเหมือนของรหัสต้นฉบับของโปรแกรมและโอกาสในการคัดลอกของรหัสต้นฉบับ

งานวิจัยของอารานี และคณะ (N. Alanee et al.) [14] ได้นำเสนอวิธีการในการแยกตัวระบุ (identifier) ออกมาจากรหัสต้นฉบับโดยใช้การสกัดคุณลักษณะของรหัสต้นฉบับ ร่วมกับการใช้ตัวจำแนกประเภทแบบเบย์ (Naïve Bayes) เพื่อช่วยในการจำแนกตัวระบุ และได้มีการประเมินประสิทธิผลในการจำแนกด้วยเทคนิคทางด้านการค้นคืนสารสนเทศ โดยในงานวิจัยชิ้นนี้ได้มีการนำเสนอคุณลักษณะที่ได้นำมาใช้ด้วยกัน 4 คุณลักษณะได้แก่ 1. จำนวนของตัวอักษรตัวพิมพ์ใหญ่ (capital latter) ในตัวระบุ 2. จำนวนครั้งของการใช้เครื่องหมายวรรคตอน (punctuation) ในตัวระบุ 3. จำนวนตัวเลข (digit) ในตัวระบุ และ 4. จำนวนของคำที่แยกได้จากการใช้การตรวจสอบ

ความถูกต้องของคำด้วยโปรแกรม “Microsoft word” โดยข้อมูลทั้งหมดจะถูกแบ่งออกเป็นข้อมูลสำหรับฝึกสอน (training data) และ ข้อมูลสำหรับทดสอบ (testing data) ซึ่งในส่วนของข้อมูลที่ใช้ในการฝึกสอนนั้นได้มีการกำหนดประเภทของกลุ่ม (class) ลงบนข้อมูลโดยที่ประเภทของกลุ่มข้อมูลนี้ได้มาจาก การนำจำนวนคำของตัวระบุที่สามารถแยกออกมาเป็นคำได้ เช่น “GetTokenAt” สามารถแยกออกมาเป็นคำได้ทั้งหมดสามคำด้วยกันได้แก่ “Get” “Token” และ “At” ในที่นี้ก็จะสามารถกำหนดประเภทของกลุ่มข้อมูลได้เป็นประเภทที่ 3 หลังจากที่ได้ประเภทของกลุ่มข้อมูลแล้วก็จะนำข้อมูลที่ได้มีการแยกประเภทแล้วทั้งหมดไปทำการฝึกสอนให้กับตัวจำแนกแบบ Naïve Bayes และจากนั้นจะนำเอาข้อมูลสำหรับทดสอบ มาทดสอบกับตัวจำแนกที่ได้มีการถูกฝึกสอนมาแล้ว ในท้ายที่สุดจะนำค่าที่ได้จากการจำแนกมาประเมินประสิทธิภาพด้วยการใช้การประเมินประสิทธิผลทางด้านการค้นคืนสารสนเทศซึ่งได้แก่ การประเมินด้วยค่าความเที่ยง (precision) การประเมินด้วยค่าระลึก (recall) และการประเมินด้วยค่าวัดประสิทธิภาพ (F-measure) โดยการประเมินนี้จะถูกแบ่งออกเป็นการประเมินสามครั้งโดยที่แต่ละครั้งของการประเมินนั้นแตกต่างกันตรงที่จำนวนของคุณลักษณะที่ได้นำมารวมกัน อาทิเช่น การประเมินครั้งที่หนึ่งได้ทำการประเมินโดยใช้ จำนวนตัวอักษรพิมพ์ใหญ่ในตัวระบุเพียงอย่างเดียว การประเมินครั้งที่สองได้ทำการประเมินโดยใช้ จำนวนตัวอักษรพิมพ์ใหญ่ในตัวระบุร่วมกับจำนวนสัญลักษณ์ในตัวระบุ และในการประเมินครั้งที่สามได้ทำการประเมินโดยใช้ จำนวนตัวอักษรพิมพ์ใหญ่ในตัวระบุ จำนวนสัญลักษณ์ในตัวระบุ และ จำนวนตัวเลขในตัวระบุ เป็นต้น โดยงานวิจัยชิ้นนี้ได้แสดงให้เห็นถึงประสิทธิผลในการจำแนกตัวระบุออกจากรหัสต้นฉบับของโปรแกรมโดยใช้คุณลักษณะที่ถูกสกัดออกมาจากรหัสต้นฉบับของโปรแกรม

**2.2 การประเมินความยากง่ายทางการอ่านของรหัสต้นฉบับ** กรรมวิธีในการประเมินรหัสต้นฉบับของโปรแกรมว่ารหัสต้นฉบับนั้น ๆ มีความยากง่ายทางการอ่านเป็นอย่างไรบ้าง โดยที่ความยากง่ายทางการอ่านนี้จะเป็นตัวชี้วัดถึงคุณภาพของรหัสต้นฉบับนั้น ๆ โดยเฉพาะเวลาที่มีผู้เขียนโปรแกรมคนอื่นที่ไม่ใช่เจ้าของของรหัสต้นฉบับนั้น ๆ ได้นำรหัสต้นฉบับนั้นไปใช้ต่อ หรือแม้กระทั้งนำไปศึกษาถึงแนวทางการเขียนโปรแกรมของเจ้าของรหัสต้นฉบับดังกล่าว ซึ่งความยากง่ายทางการอ่านจะมีผลต่อความน่าสนใจ รวมถึงความเข้าใจในเวลาที่มีผู้เขียน หรือผู้สนใจคนอื่น ๆ มาดูหรือมาศึกษารหัสต้นฉบับนั้น ๆ ถ้ารหัสต้นฉบับนั้นเป็นรหัสต้นฉบับที่อ่านง่ายก็จะส่งผลทำให้ผู้ที่มาดูหรือศึกษาเกิดความเข้าใจในตัวรหัสต้นฉบับได้ง่ายขึ้นว่ารหัสต้นฉบับของโปรแกรมดังกล่าวนั้นทำงานอย่างไร หรือมีวิธีในการเขียนอย่างไร แต่ในทางกลับกันถ้ารหัสต้นฉบับนั้นเป็นรหัสต้นฉบับที่อ่านยากผู้ที่มาดู หรือศึกษาก็จะเกิดความไม่เข้าใจต่อสิ่งที่ผู้เขียนโปรแกรมต้องการที่จะสื่อ ส่งผลให้เกิดความไม่เข้าใจ และความไม่น่าสนใจของตัวรหัสต้นฉบับนั้น ๆ

โดยงานวิจัยที่จะนำมาเสนอดังต่อไปนี้เป็นงานวิจัยที่ได้มีการประเมินความยากง่ายทางการอ่านของรหัสต้นฉบับไว้โดยนักวิจัยที่มีชื่อว่า เรย์ม่อน บรูซ และคณะ (Raymond P.L. Buse et al.) [15] โดยงานวิจัยชิ้นนี้ได้ทำการประเมินความยากง่ายทางการอ่านของรหัสต้นฉบับของโปรแกรมโดยการใช้การตัดสินจากผู้เชี่ยวชาญ (expert judgment) ผ่านทางการทำการสำรวจจากผู้เชี่ยวชาญผ่านทางโปรแกรมที่มีชื่อว่า “Sniped Sniper” ซึ่งเป็นโปรแกรมที่ทำงานผ่านทางเว็บ หรือเรียกง่าย ๆ ว่าเป็นเว็บโปรแกรมประยุกต์ (web application) โดยจะให้ผู้เชี่ยวชาญเข้ามาให้คะแนนของตัวอย่างของรหัสต้นฉบับของโปรแกรมที่จัดเตรียมไว้ให้ และจะนำคะแนนที่ได้ไปทำการจัดกลุ่มระดับความยากง่ายทางการอ่าน และจะนำกลุ่มแต่ละกลุ่มไปวิเคราะห์หาถึงองค์ประกอบของความยากง่ายทางการอ่านของรหัสต้นฉบับ โดยใช้การสกัดข้อมูลเพื่อที่จะนำเอาคุณลักษณะของรหัสต้นฉบับของโปรแกรมออกมาวิเคราะห์ด้วยการใช้วิธีการนับจำนวนในการสกัดคุณลักษณะออกมา ยกตัวอย่างเช่น จำนวนของอักขระในหนึ่งบรรทัด (number of characters) จำนวนของตัวระบุ (number of identifiers) เป็นต้น หลังจากนั้นจะนำคุณลักษณะเหล่านี้มาทำการจำแนกประเภทเพื่อให้ทราบถึงความสำคัญของแต่ละคุณลักษณะที่ส่งผลโดยตรงต่อระดับของความยากง่ายทางการอ่านของโปรแกรม

จากงานวิจัยที่ได้นำเสนอมาในข้างต้นทั้งงานวิจัยทางด้านการสกัดคุณลักษณะ และงานวิจัยทางด้านการประเมินความยากง่ายทางการอ่านของรหัสต้นฉบับ พบว่างานวิจัยทั้ง 2 ด้านสามารถนำมาประยุกต์ให้ใช้ในวิทยานิพนธ์นี้ได้โดยเฉพาะ งานวิจัยทางด้านการสกัดคุณลักษณะเป็นงานวิจัยที่มีความน่าสนใจเป็นอย่างมากเพราะจะสามารถสร้างตัวแทนของข้อมูลที่จะนำมาวิเคราะห์ได้อย่างมีประสิทธิภาพแต่ถึงกระนั้น การนำคุณลักษณะจากงานวิจัยทางด้านการสกัดคุณลักษณะที่ได้นำเสนอไปนั้นไม่สามารถนำมาใช้ในวิทยานิพนธ์นี้ได้เนื่องจากคุณลักษณะต่าง ๆ ไม่สามารถสื่อถึงสิ่งที่ต้องการจะค้นหา และประเมินได้จึงจำเป็นที่จะต้องค้นหาคุณลักษณะใหม่เพื่อที่จะนำมาใช้ในวิทยานิพนธ์นี้ ส่วนงานวิจัยทางด้านการประเมินความยากง่ายทางการอ่านนั้น สามารถนำมาประยุกต์ใช้กับวิทยานิพนธ์ชิ้นนี้ได้ จากที่กล่าวมาในข้างจึงมีแนวคิดที่จะนำกรรมวิธีการสกัดข้อมูล และการประเมินความยากง่ายทางการอ่านของรหัสต้นฉบับมาประยุกต์ใช้ในวิทยานิพนธ์ชิ้นนี้ โดยขั้นตอน และกรรมวิธีต่าง ๆ จะอธิบายให้เห็นในบทถัดไป

# บทที่ 3
# การศึกษาความถนัดและการประเมินทักษะในการเขียนโปรแกรม

จากที่ได้กล่าวถึงที่มาของวิทยานิพนธ์ชิ้นนี้ในบทที่ 1 จะแสดงให้เห็นว่าที่ผ่านมานักวิจัยต่างได้ทำการตรวจสอบรหัสต้นฉบับในเชิงของการคัดลอก หรือความเหมือนกันของรหัสต้นฉบับเท่านั้น โดยที่ไม่ได้มีการกล่าวถึงเรื่องพฤติกรรม ความถนัด หรือแม้แต่กระทั่งทักษะของผู้เขียนโปรแกรมโดยแม้แต่น้อย ด้วยเหตุนี้จึงเป็นสาเหตุที่ทำให้เกิดเป็นแนวความคิดที่จะทำการค้นหาความถนัด และประเมินทักษะของผู้รียนเขียนโปรแกรมขึ้นมา โดยจุดประสงค์หลักของวิทยานิพนธ์ชิ้นนี้เป็นการค้นหาความถนัด และประเมินทักษะของผู้เรียนการเขียนโปรแกรม เพื่อเป็นข้อมูลให้กับครูผู้สอนการเขียนโปรแกรมได้นำไปใช้ในการเสริมสร้างทักษะทางด้านการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรม

สำหรับเนื้อหาที่จะกล่าวถึงในบทนี้จะเกี่ยวกับแรงจูงใจที่ทำให้เกิดการค้นหาความถนัด และประเมินทักษะของผู้เรียนการเขียนโปรแกรม พร้อมทั้งเสนอแนวคิดสำหรับการค้นหาความถนัดที่ได้มาจากการเปรียบเทียบระหว่างรหัสต้นฉบับของผู้เรียนการเขียนโปรแกรมกับวัตถุประสงค์การสอนการเลือกใช้งานเครื่องมือทางภาษาโปรแกรม (การประกาศตัวแปร การเลือกใช้งานการทำงานแบบเลือกทำ การเลือกใช้งานการทำงานแบบทำซ้ำ) หมายเหตุจากการตอบรับของระบบตรวจอัตโนมัติ ความยากง่ายทางการอ่านของรหัสต้นฉบับ อีกทั้งได้มีการเสนอแนวคิดสำหรับการประเมินทักษะของผู้เรียนการเขียนโปรแกรมด้วยการพิจารณาเวลาที่ผู้เรียนใช้ไปในการเขียนโปรแกรมหนึ่งข้อร่วมกับการพิจารณาจำนวนครั้งของการส่งโปรแกรมเหล่านั้นขึ้นตรวจบนระบบตรวจอัตโนมัติ

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

## 3.1 ความถนัดในการเขียนโปรแกรม

การค้นหาความถนัด และการประเมินทักษะทางด้านการเขียนโปรแกรมเป็นแนวคิดที่จะทำการค้นความถนัดในการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรมว่าผู้เรียนแต่ละคนนั้นมีความถนัดในการเขียนโปรแกรมเป็นอย่างไร ทั้งนี้ความถนัดในการเขียนโปรแกรมนั้นสามารถอธิบายได้ด้วยนิยามที่ 3.1

**นิยามที่ 3.1** ความถนัดในการเขียนโปรแกรมสามารถอธิบายได้ด้วย จำนวนครั้งของการเลือกใช้เครื่องมือทางภาษาเขียนโปรแกรมที่ประกอบไปด้วย ลักษณะการประกาศตัวแปร ลักษณะการเลือกใช้การทำงานแบบเลือกทำ และลักษณะในการเลือกใช้การทำงานแบบทำซ้ำในการเขียนโปรแกรมที่ได้รับคะแนนเต็มจากการตรวจ

นอกจากนี้ยังข้อมูลที่เป็นข้อมูลเพิ่มเติมที่จะสะท้อนให้เห็นถึงลักษณะการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรมได้แก่ 1. การเปรียบเทียบวัตถุประสงค์ของการสอนกับรหัสต้นฉบับของโปรแกรม 2.ความยากง่ายทางการอ่านของรหัสต้นฉบับของโปรแกรม และ 3. หมายเหตุที่ได้จากระบบตรวจอัตโนมัติ โดยข้อมูลเพิ่มเติมทั้ง 3 ส่วนที่ได้กล่าวมานั้นสะท้อนให้เห็นถึงลักษณะในการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรมว่า ได้ปฏิบัติตามสิ่งที่ครูผู้สอนได้ทำการสอนไปหรือไม่ สาเหตุของปัญหาที่เกิดกับโปรแกรมที่ผู้เรียนการเขียนโปรแกรมเขียนขึ้นมา และความยากง่ายทางการอ่านของรหัสต้นฉบับของผู้เรียนการเขียนโปรแกรมที่ได้ทำการเขียนขึ้นมาว่ารหัสต้นฉบับนั้นมีความยากง่ายทางการอ่านเพื่อทำความเข้าใจเป็นอย่างไรบ้าง ส่วนทางด้านการประเมินทักษะการเขียนโปรแกรมเป็นแนวคิดที่เริ่มมาจากในการเรียนการเขียนโปรแกรมนั้นผู้เรียนจะไม่มีทางทราบถึงทักษะในการเขียนโปรแกรมของผู้เรียนเองจนกว่าจะมีการทดสอบ (test) หรือการสอบ (examination) ว่าทักษะของผู้เรียนหลังจากที่ได้ทำการเรียนเขียนโปรแกรมไปในแต่ละครั้งของการสอนนั้นเป็นอย่างไรบ้าง โดยคำว่าทักษะสามารถอธิบายได้ด้วยนิยามที่ 3.2

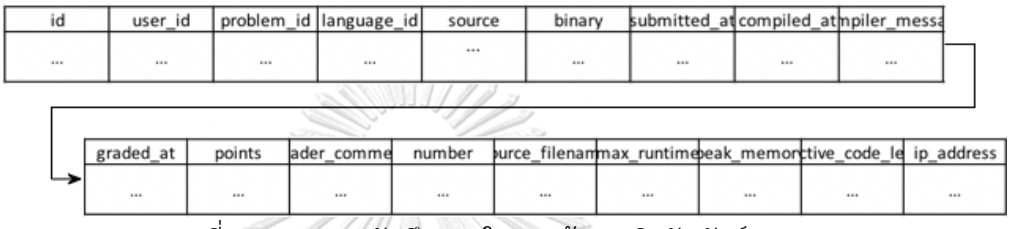
**นิยามที่ 3.2** ทักษะทางด้านการเขียนโปรแกรมสามารถอธิบายได้ด้วย การเปรียบเทียบเวลาที่ใช้ไปในการเขียนโปรแกรม 1 ข้อกับเวลาจำกัด และจำนวนครั้งของการส่งโดยที่ไม่สนใจเวลาที่ใช้ไปในการเขียนโปรแกรมในข้อนั้น ๆ

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

จากที่ได้กล่าวมาในข้างต้น ทำให้ผู้วิจัยได้มีแนวคิดที่จะทำการค้นหาความถนัด และประเมินทักษะในการเขียนโปรแกรมมาใช้ในการเรียนการสอนการเขียนโปรแกรมเพื่อเป็นข้อมูลให้กับครูผู้สอนได้รับทราบถึงความถนัด และทักษะของผู้เรียนเพื่อที่จะได้ปรับเปลี่ยนรูปแบบการเรียนการสอน หรือเข้าไปเสริมในจุดที่ผู้เรียนขาดความเข้าใจเพื่อที่จะทำให้ผู้เรียนเกิดความเข้าใจมากขึ้น ส่งผลทำให้ผู้เรียนมีพัฒนาการทางด้านทักษะการเขียนโปรแกรมที่ดีขึ้น

### 3.2 การค้นหาความถนัดในการเขียนโปรแกรม

การค้นหาความถนัด และพฤติกรรมในการเขียนโปรแกรมจะอาศัยการค้นหาข้อมูล ที่อยู่ภายในรหัสต้นฉบับที่ถูกเขียนขึ้นมาจากภาษาจาวา (java) โดยข้อมูลที่นำมาค้นหาความถนัดในการเขียนโปรแกรมนี้จะจัดเก็บอยู่ในรูปแบบของฐานข้อมูลเชิงสัมพันธ์ (relational database) ที่มาจากระบบตรวจอัตโนมัติ (grader system) โดยข้อมูลจะแบ่งการจัดเก็บออกเป็นจำนวน 18 เขตข้อมูล (field) ดังรูปที่ 3.1




รูปที่ 3.1 รายการบันทึกภายในฐานข้อมูลเชิงสัมพันธ์

แต่เนื่องจากจำนวนเขตข้อมูลที่มีมากถึง 18 รายการนั้นมีข้อมูลที่ไม่เกี่ยวข้องกับการค้นหาความถนัดในการเขียนโปรแกรมรวมอยู่ด้วย จึงจำเป็นที่จะต้องใช้ตัวกรอง (filter) เพื่อที่จะคัดกรองเอาข้อมูลที่สำคัญและจำเป็นต้องใช้ออกมาเพื่อลดเวลาและภาระของเครื่องที่ใช้ในการประมวลผลข้อมูล โดยข้อมูลที่ถูกคัดกรองแล้วจะเหลืออยู่เพียง 7 เขตข้อมูลดังรูปที่ 3.2 โดยทั้ง 7 เขตข้อมูลนี้จะอยู่ในรูปของฐานข้อมูลเชิงสัมพันธ์ และจะถูกแปลงออกมาเป็นแฟ้มข้อมูลข้อมูลนามสกุลจาวา (.java) ดังรูปที่ 3.3 ที่จะประกอบไปด้วย user_id, problem_id, source, submitted_at, point และ grader_comment แต่ในส่วนของ number (ในที่นี้จะหมายถึงจำนวนครั้งของการส่ง) จะใช้การนับจำนวนและแปลงออกมาเป็นข้อมูลเพื่อการประมวลผลในส่วนการประเมินทักษะโดยตรง โดยที่ไม่มีการบันทึกลงแฟ้มข้อมูลนามสกุลจาวาแต่อย่างใด

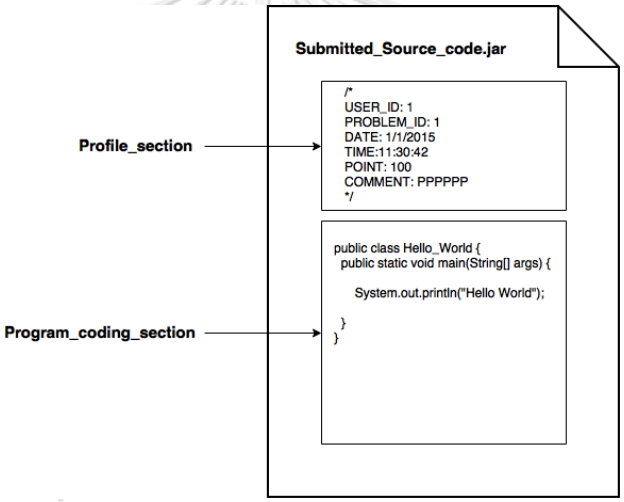
จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

| user_id | problem_id | source | submitted_at | points | grader_comment | number |
|---|---|---|---|---|---|---|
| 451 | 10 | | 2014-09-21 15:05:08 | 80 | PASSED: PPPPPPPP(inconsistent score) | 1 |
| 451 | 12 | | 2014-08-19 02:23:36 | 20 | PP | 1 |
| 451 | 21 | import java.util.Scanner; | 2014-08-19 08:16:46 | 0 | ----- | 1 |
| 451 | 21 | import java.util.Scanner; | 2014-08-19 08:24:21 | 50 | PPPPP | 2 |

รูปที่ 3.2 รายการบันทึกที่นำมาใช้ทดลองจากฐานข้อมูลเชิงสัมพันธ์

โดยในแต่ละเขตข้อมูลจะเก็บข้อมูลดังต่อไปนี้

1. user_id เก็บรหัสผู้เรียนการเขียนโปรแกรม
2. problem_id เก็บรหัสโจทย์ปัญหา
3. source เก็บรหัสต้นฉบับที่ผู้เรียนการเขียนโปรแกรมได้เขียนขึ้นมา
4. submitted_at เก็บวัน และเวลาที่ผู้เรียนการเขียนโปรแกรมส่งโปรแกรมขึ้นตรวจ
5. points เก็บคะแนนที่ได้จากการตรวจโปรแกรม
6. grader_comment เก็บหมายเหตุที่ได้จากการตรวจโปรแกรมของผู้รียนการเขียนโปรแกรม
7. number เก็บจำนวนครั้งของการส่งโปรแกรมขึ้นตรวจ โดยโปรแกรมนั้น ๆ จะต้องเป็นโปรแกรมที่ถูกเขียนขึ้นจากผู้เรียนการเขียนโปรแกรมคนเดียวกัน และจากโจทย์ปัญหาข้อเดียวกัน




รูปที่ 3.3 แฟ้มข้อมูลนามสกุลจาวาที่แปลงมาจากเขตข้อมูลในฐานข้อมูลเชิงสัมพันธ์

CHULALONGKORN UNIVERSITY

โดยข้อมูลนามสกุลจาวานี้จะถูกนำเข้าสู่ขั้นตอนการทำความสะอาดข้อมูล (cleaning data) เนื่องจากข้อมูลที่ได้มาประกอบไปด้วยแผ่นแบบที่จะส่งผลทำให้เกิดความไม่ถูกต้องของผลลัพธ์จากการค้นหาความถนัด จึงจำเป็นที่จะต้องมีการทำความสะอาดข้อมูลรหัสต้นฉบับก่อนเริ่มต้นการค้นหาความถนัดทางด้านการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรม โดยใช้การจับคู่สายอักขระ (string matching) ระหว่างแผ่นแบบที่มีอยู่ในระบบ กับรหัสต้นฉบับที่ถูกเขียนขึ้นโดยผู้เรียนการเขียนโปรแกรม โดยที่ผลลัพธ์ที่ได้จากการทำความสะอาดข้อมูลจะเหลือแต่เพียงส่วนของชุดคำสั่งที่ผู้เรียนการเขียนโปรแกรมได้ทำการเขียนลงไปเท่านั้นโดยจะแสดงให้เห็นดังรูปที่ 3.4

```
USER_ID: 1
PROBLEM_ID: 1
DATE: 1/1/2015
TIME: 11:30:42
POINT: 100
COMMENT: PPPPPP


        int a = 0;
        int b = 2;
        if(a > b){
          System.out.println("TRUE");
        } else {
          System.out.println("FALSE");
        }
```

รูปที่ 3.4 ผลลัพธ์ที่ได้จากการทำความสะอาดข้อมูลโดยการใช้การจับคู่สายอักขระ

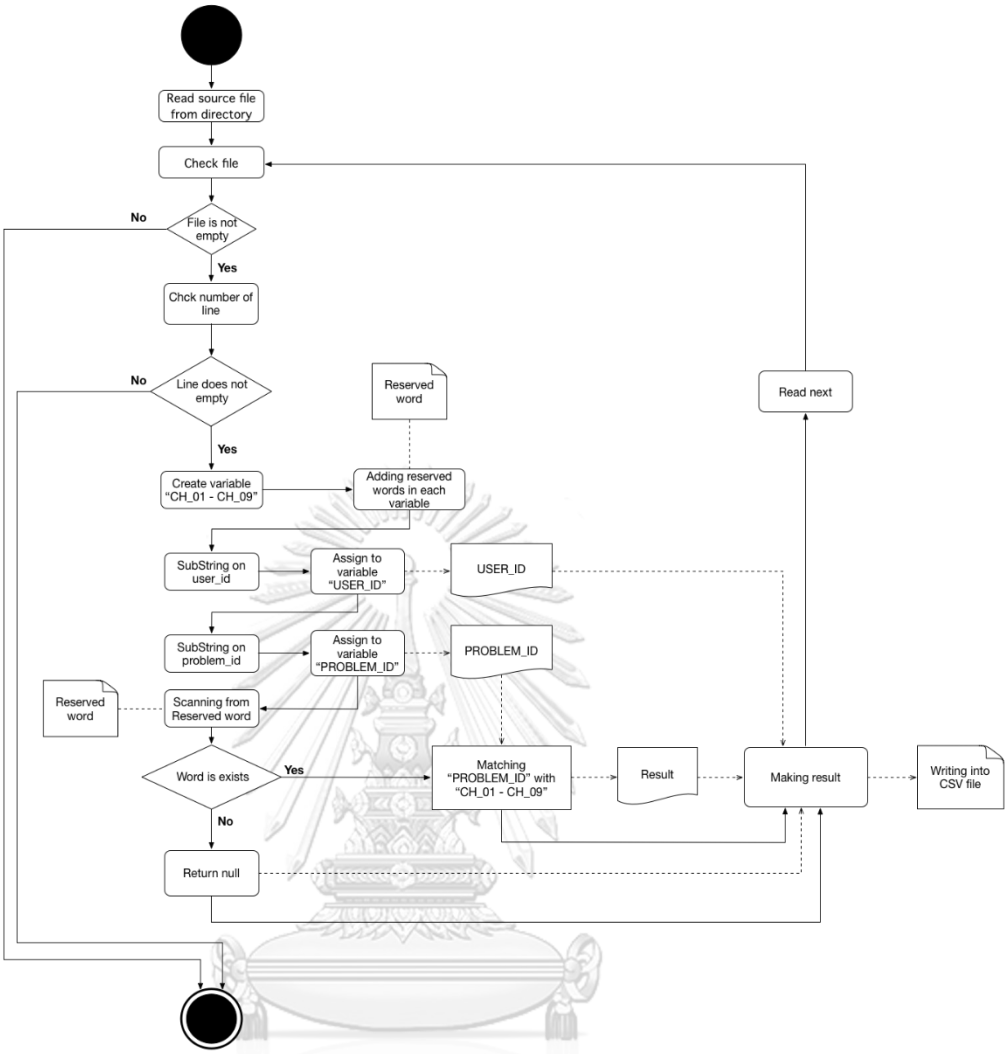
หลังจากที่ได้มีการทำความสะอาดข้อมูลแล้วในขั้นตอนต่อไปจะเป็นการค้นหาความถนัดในการเขียนโปรแกรมซึ่งจะประกอบด้วยส่วนประกอบดังต่อไปนี้

### 3.2.1. เวกเตอร์ของผู้ใช้

ในส่วนนี้จะเป็นการสร้างเวกเตอร์ของผู้ใช้จากโปรแกรมที่ผู้เขียนเขียนขึ้น โดยเวกเตอร์นี้จะเป็นเวกเตอร์ขนาด 2 มิติที่จะประกอบไปด้วย ผลจากการเปรียบเทียบวัตถุประสงค์การสอน (objective) กับโปรแกรมที่ผู้เขียนเขียนขึ้น และคะแนนที่ได้จากการตรวจโดยระบบตรวจอัตโนมัติ (grader system) โดยในส่วนแรกจะอธิบายขั้นตอนและกรรมวิธีการได้มาซึ่งสถานะจากการเปรียบเทียบ โดยจะสร้างแฟ้มข้อมูลเอกสารข้อมูลที่จะประกอบไปด้วยคำหลัก (keyword) โดยคำหลักเหล่านี้คือสิ่งที่จะสอนภายในชั้นเรียนที่ได้มาจากประมวลรายวิชา (course syllabus) โดยจะประกอบไปด้วยคำหลักดังนี้ {“Math.”, “if”, “else”, “else if”, “for”, “while”, “parseInt”, “parseString”, “substring”, “delimeter”, “scanner”, “array[]”, “array[][]” } หลังจากสร้างเอกสารคำหลักแล้วก็จำนำเอกสารดังกล่าวไปไว้ในสารบบ (directory) เพื่อรอการเรียกใช้ต่อไป ต่อไปก็จะจะนำแฟ้มข้อมูลของโปรแกรมที่ต้องการจะตรวจเข้าสู่โปรแกรมเปรียบเทียบวัตถุประสงค์ โดยโปรแกรมจะเริ่มทำงานเมื่อมีการนำเข้าแฟ้มข้อมูลของโปรแกรมที่ต้องการจะเปรียบเทียบ จากนั้นโปรแกรมเริ่มทำการเก็บข้อมูลผู้เขียนโดยใช้สายอักขระย่อย (substring) จากส่วนข้อมูลผู้เขียน (Profile_section) และนำไปเก็บไว้ในตัวแปร โดยจะแบ่งการเก็บเป็น 2 ตัวแปรคือ รหัสผู้ใช้ (USER_ID) ไปเก็บไว้ที่ตัวแปรชนิดอักขระที่มีชื่อว่า “user_id” และรหัสโจทย์ (PROBLEM_ID) ไปเก็บไว้ที่ตัวแปรชนิดอักขระที่มีชื่อว่า “problem_id” ภายหลังจากการเก็บข้อมูลผู้ใช้เสร็จแล้วโปรแกรมจะไปอ่านข้อมูลจากเอกสารคำหลักที่ได้เตรียมไว้ก่อนหน้า เพื่อนำคำหลักเหล่านั้นมาค้นหา

ในแฟ้มข้อมูลของโปรแกรมที่นำเข้ามาเพื่อเปรียบเทียบ เมื่อโปรแกรมทำการค้นหาแล้วพบว่าคำหลักที่นำมาค้นนั้นปรากฏอยู่ ณ ตำแหน่งใด ๆ ในแฟ้มข้อมูลของโปรแกรมที่นำเข้ามาแล้ว ก็จะไปเปลี่ยนค่าของตัวแปร “checking” จาก เท็จ (false) เป็น จริง (true) ที่เป็นตัวแปรชนิดตรรกะ (boolean) ที่ประกาศไว้สำหรับตรวจสอบสถานะการค้นหาว่าพบหรือไม่พบ หลังจากนั้นโปรแกรมจะเข้าสู่เงื่อนไขการเปรียบเทียบ โดยอ้างอิงจากค่าสถานะของตัวแปร checking โดยที่ถ้ามีค่าสถานะเป็นจริง แล้วโปรแกรมจะนำ ค่าของรหัสโจทย์ที่เก็บไว้ในตัวแปร ไปค้นหาในแถวลำดับ (array) ที่ชื่อว่า “CH_” โดยตัวแปรนี้จะมีทั้งหมด 9 ตัวไล่ตั้งแต่ “CH_01 – CH_09” โดยในแต่ละแถวลำดับคือหมายเลขบทเรียนที่สอนไปในชั้นเรียนที่จะประกอบไปด้วยรหัสของโจทย์ที่อยู่ในบทเรียนนั้น ๆ โดยที่ถ้าค่าสถานะเป็นจริง และนำหมายเลขโจทย์ที่ได้มาจากส่วนข้อมูลผู้ใช้มาค้นหาแล้วพบในแถวลำดับใด ๆ แล้วจะส่งคืนค่า (return) ที่มีชื่อว่า “Successful” ออกมา แต่ในทางกลับกัน ถ้าค่าสถานะเป็นจริงแล้ว และนำรหัสโจทย์จากส่วนของผู้ใช้มาค้นหาในแล้วไม่พบในแถวลำดับใด ๆ แล้วจะส่งคืนค่าที่มีชื่อว่า “Unsuccessful” ออกมา และนำค่าดังกล่าวไปบันทึกลงแฟ้มข้อมูลเอกสารต่อไป โดยจะบันทึกเป็นรายการ “รหัสผู้ใช้, รหัสโจทย์, ผลจากการเปรียบเทียบ” โดยโปรแกรมจะทำงานไปเรื่อย ๆ จนกว่าจะไม่มีแฟ้มข้อมูลนำเข้าแล้ว โดยการทำงานทั้งหมดจะแสดงให้เห็นดังรูปที่ 3.5

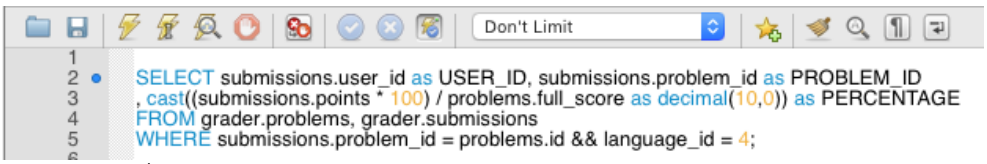
จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University




รูปที่ 3.5 การทำงานของโปรแกรมเปรียบเทียบวัตถุประสงค์การสอน

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

หลังจากที่ได้ผลลัพธ์จากการเปรียบเทียบมาแล้วก็จะนำในสารบบ เพื่อรอการเรียกใช้ต่อไป และอีกหนึ่งส่วนก็คือ อัตราส่วนคะแนนร้อยละ (percentage) โดยคะแนนในส่วนนี้จะเป็นการใช้การสอบถามโดยตรงไปยังระบบฐานข้อมูลเชิงสัมพันธ์ และบันทึกผลออกมาโดยตรงจากการสอบถามที่ใช้ภาษาซีเควล (SQL) ดังรูปที่ 3.6




รูปที่ 3.6 ภาษาซีเควลสำหรับการใช้สอบถามไปยังฐานข้อมูลเชิงสัมพันธ์

โดยที่จะมีการคำนวณค่าประสิทธิภาพ (Performance) ที่อยู่ในรูปของอัตราร้อยละ (Percentage) เพื่อที่จะทำให้ข้อมูลนั้นเป็นแบบเดียวกันโดยใช้คำสั่ง “cast((submissions.points * 100) / problems.full_score as decimal(10,0)) as PERCENTAGE” คำนวณจาก 2 ตารางได้แก่ ตาราง “submissions” ที่เก็บคะแนนที่ได้จากการตรวจจากระบบตรวจอัตโนมัติ และตาราง“problem” ที่เก็บคะแนนเต็มของแต่ละโจทย์ปัญหา โดยการปรับข้อมูลในครั้งนี้มีสาเหตุมาจากค่าของคะแนนเต็มของแต่ละโจทย์ปัญหานั้นมีไม่เท่ากันโดยผลลัพธ์ที่ออกมาจะประกอบไปด้วยกัน 3 เขตข้อมูลได้แก่ รหัสผู้ใช้ (USER_ID) รหัสโจทย์ (PROBLEM_ID) และอัตราส่วนคะแนนร้อยละ (PERCENTAGE) ดังรูปที่ 3.7 และภายหลังจากที่ได้คำนวณค่าประสิทธิภาพก็จะนำผลลัพธ์ที่ได้จากการเปรียบเทียบวัตถุประสงค์การสอนกับรหัสต้นฉบับที่ผู้เรียนการเขียนโปรแกรมได้เขียนขึ้นมารวมกับค่าประสิทธิภาพ ดังรูปที่ 3.8

| USER... | PROBLEM... | PERCENTAGE |
|---|---|---|
| 451 | 21 | 0 |
| 451 | 21 | 100 |
| 451 | 23 | 100 |
| 451 | 24 | 0 |
| 451 | 24 | 0 |
| 451 | 24 | 100 |
| 451 | 25 | 100 |
| 451 | 26 | 100 |
| 451 | 27 | 100 |
| 451 | 28 | 0 |

รูปที่ 3.7 ผลลัพธ์ที่ได้จากการคำนวณหาอัตราส่วนร้อยละของคะแนนที่ได้จากระบบตรวจอัตโนมัติ

จุฬาลงกรณ์มหาวิทยาลัย

| USER_ID | PROBLEM_ID | OBJECTIVE_STATUS | PERFORMANCE_SCORE |
|---|---|---|---|
| 451 | 21 | Pass | 0 |
| 451 | 21 | Pass | 100 |
| 451 | 23 | Pass | 100 |
| 451 | 24 | Pass | 0 |
| 451 | 24 | Pass | 0 |
| 451 | 24 | Pass | 100 |
| 451 | 25 | Pass | 100 |
| 451 | 26 | Pass | 100 |
| 451 | 27 | Pass | 100 |
| 451 | 28 | Pass | 0 |
| 451 | 28 | Pass | 100 |

รูปที่ 3.8 ผลลัพธ์จากการรวมข้อมูลระหว่างการเปรียบเทียบวัตถุประสงค์ และอัตราส่วนคะแนนร้อยละ

ภายหลังจากเสร็จสิ้นการรวมข้อมูลแล้วก็จะนำแฟ้มข้อมูลดังกล่าวไปเก็บไว้ในสารบบโดยใช้ชื่อเวกเตอร์ข้อมูลผู้ใช้ (User_vector_profile) เพื่อรอการเรียกใช้งานต่อไป

### 3.2.2 เครื่องมือทางภาษาโปรแกรม (Programming language tools)

ในทางการตรวจสอบการใช้เครื่องมือทางภาษาโปรแกรมนั้น จะมีการตรวจสอบด้วยกัน 3 ด้านด้วยกัน ได้แก่ 1. ด้านการประกาศตัวแปร (variable declaration) 2. ด้านการทำงานแบบเลือกทำ (selection statement) และ 3. ด้านการทำงานแบบวนซ้ำ (iteration statement) โดยทั้ง 3 ด้านจะมีวิธีการตรวจสอบที่แตกต่างกัน โดยจะขออธิบายตามลำดับดังนี้

**3.2.2.1 การประกาศตัวแปร** จะเป็นการตรวจสอบลักษณะของการประกาศตัวแปรที่จะแบ่งออกเป็น 3 ประเภทได้แก่ การประกาศตัวแปรแบบเดี่ยวต่อ 1 บรรทัด (single declaration) การประกาศตัวแปรแบบหลายตัวต่อ 1 บรรทัด (multiple declaration) และการประกาศตัวแปรแบบผสม (mixed declaration) โดยการตรวจสอบนั้นจะใช้การค้นหาจากคำ หรือกลุ่มคำที่ได้กำหนดไว้ และการนับจำนวนโดยจะมีการทำงานดังนี้ เริ่มต้นจากทำการป้อนเอกสารรหัสต้นฉบับเข้าสู่ระบบ ต่อมาระบบจะนำชื่อของไฟล์มาเก็บไว้ในตัวแปรที่ชื่อว่า “name” ต่อมาจะทำการค้นหาจากกลุ่มคำที่ได้กำหนดไว้ซึ่งจะแบ่งตามประเภทของตัวแปร ได้แก่ ตัวแปรประเภทจำนวนเต็ม (integer) ที่มีจะการกำหนดคำเป็น “int” ตัวแปรประเภททศนิยม (double) ที่จะมีการกำหนดคำเป็น “double” และตัวแปรประเภทอักขระ (string) ที่จะมีการกำหนดคำเป็น “string” เมื่อระบบทำการค้นหาจนเจอคำที่ได้กำหนดไว้แล้ว ณ ตำแหน่งใด ๆ ของรหัสต้นฉบับแล้ว ก็จะใช้การนับจำนวนเครื่องหมายจุลภาค “,” หรือเครื่องหมายอัฒภาค “;” ไปจนสิ้นสุดบรรทัดนั้น ๆ โดยจะมีเงื่อนไขการนับจำนวนดังนี้

- ถ้ามีการใช้เครื่องหมายอัฒภาคอยู่เพียง 1 ครั้งก่อนที่จะมีการกำหนดค่าตัวแปรภายในบรรทัดที่มีการประกาศตัวแปร โดยไม่มีการใช้เครื่องหมายจุลภาคเลยในบรรทัดที่มีการประกาศตัวแปรแล้ว ถือว่าเป็นการประกาศตัวแปรนั้นเป็นการประกาศตัวแปรแบบเดี่ยว
- ถ้ามีการใช้เครื่องหมายอัฒภาคมากกว่า 1 ครั้งหลังการกำหนดค่าตัวแปรถือว่าการประกาศตัวแปรภายในบรรทัดที่มีการประกาศตัวแปรแล้ว ถือว่าเป็นการประกาศตัวแปรแบบหลายตัว
- ถ้ามีการใช้เครื่องหมายจุลภาคมากกว่า 1 ครั้ง ก่อนการกำหนดค่าตัวแปรภายในบรรทัดที่มีการประกาศตัวแปรและมีเครื่องหมายอัฒภาคปิดท้ายบรรทัดที่มีการประกาศตัวแปรแล้ว ถือว่าเป็นการประกาศตัวแปรนั้นเป็นการประกาศตัวแปรแบบหลายตัว

จากนั้นจะนำผลที่ได้จากการนับจำนวนตัวแปรมาเก็บไว้ในแถวลำดับ (array) ที่มีชื่อว่า “var_seq” ขนาด n มิติโดยจำนวนมิตินี้จะถูกกำหนดโดยจำนวนบรรทัดของการประกาศตัวแปรทั้งหมดภายในโปรแกรม โดยจะทำการตรวจสอบลักษณะการประกาศตัวแปรในทุกบรรทัดที่มีการประกาศตัวแปรภายในโปรแกรม และนำผลมาเก็บไว้ในแถวลำดับจนครบทุกบรรทัด และจะนำแถวลำดับนี้มาสรุปผลในขั้นตอนที่มีชื่อว่า “Variable Declaration summary” โดยจะมีเงื่อนไขการสรุปลักษณะการประกาศตัวแปรดังนี้

- ถ้าภายในแถวลำดับประกอบไปด้วยลักษณะการประกาศตัวแปรแบบเดี่ยวเพียงอย่างเดียวแล้วถือว่าโปรแกรมนี้มีการใช้การประกาศตัวแปรแบบเดี่ยวหรือหนึ่งตัวต่อหนึ่งบรรทัด
- ถ้าภายในแถวลำดับประกอบไปด้วยลักษณะการประกาศตัวแปรแบบหลายตัวเพียงอย่างเดียวแล้วถือว่าโปรแกรมนี้มีการใช้การประกาศตัวแปรแบบหลายตัวต่อหนึ่งบรรทัด
- ถ้าภายในแถวลำดับประกอบไปด้วยลักษณะการประกาศตัวแปรทั้งแบบเดี่ยวและแบบหลายตัวอยู่ด้วยกันแล้วถือว่าโปรแกรมนี้มีการใช้การประกาศตัวแปรแบบผสม

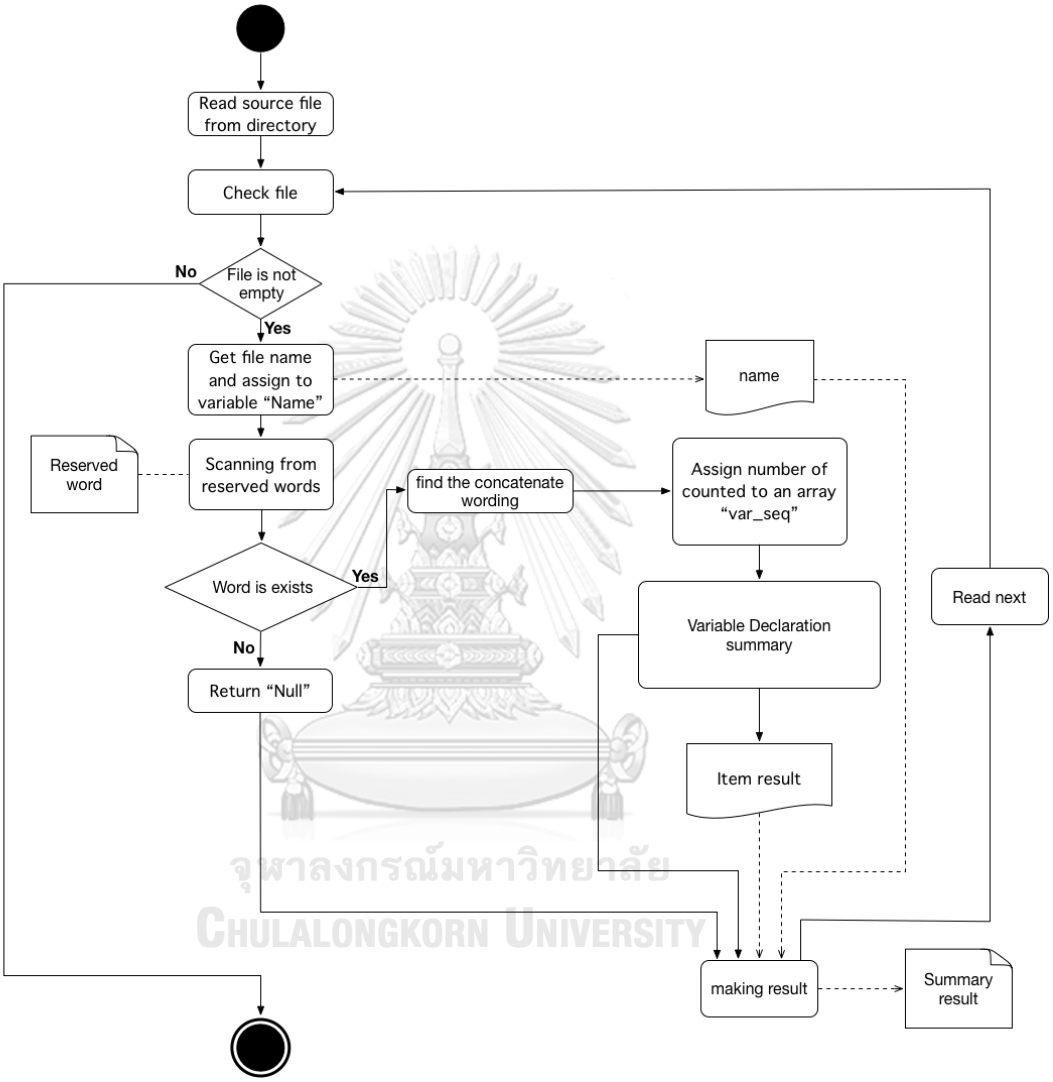
หลังจากมีการสรุปผลเสร็จแล้วถือว่าจบการทำงานของระบบตรวจสอบลักษณะการประกาศตัวแปรโดยที่การจบการทำงานของโปรแกรมนี้จะประกอบด้วย 2 กรณีคือ

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

- การที่โปรแกรมตรวจพบว่าโปรแกรมที่นำเข้ามาตรวจนั้นพบการประกาศตัวแปรและได้ทำการตรวจสอบจนหมดทั้งโปรแกรมแล้วจึงเป็นการสิ้นสุดของโปรแกรม
- การที่โปรแกรมตรวจพบว่าโปรแกรมที่นำเข้ามาตรวจนั้นไม่พบการประกาศตัวแปรภายในโปรแกรมแล้วถือว่าเป็นการสิ้นสุดการทำงานทันที

โดยที่การทำงานที่ได้กล่าวมาทั้งหมดจะแสดงดังรูปที่ 3.9 หลังจากที่โปรแกรมทำการสรุปผลเรียบร้อยแล้วจะนำเอาชื่อของไฟล์ที่เก็บไว้ในตัวแปร “name” มาเขียนลงบนแฟ้มข้อมูลนามสกุลซีเอสวี (CSV) โดยชื่อไฟล์ดังกล่าวจะมีการใช้สารอักขระย่อยในการแบ่งข้อมูลออกเป็น รหัสผู้ใช้ และรหัสโจทย์ในขั้นตอนที่มีชื่อว่า “making result” จากนั้นระบบจะทำ

การเขียนข้อมูลทั้งหมดลงบนไฟล์นามสกุลซีเอสวี โดยจะมีรูปแบบการเขียนเป็น รหัสผู้ใช้, รหัสโจทย์, ลักษณะการประกาศตัวแปร ดังรูปที่ 3.10 และนำไปเก็บไว้ในสารบบเพื่อรอการเรียกใช้ต่อไป




จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

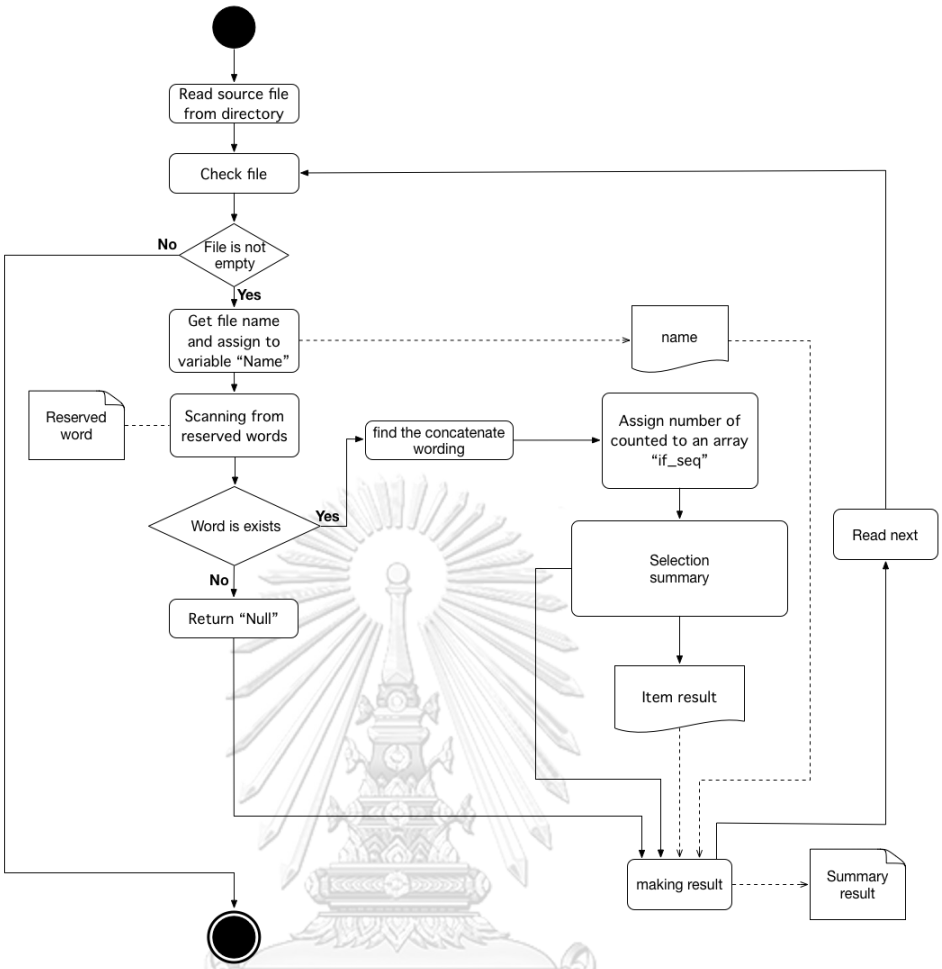
รูปที่ 3.9 การทำงานของโปรแกรมตรวจสอบลักษณะการประกาศตัวแปร

| USER_ID | PROBLEM_ID | VARIABLE_STYLE |
|---|---|---|
| 451 | 21 | Multiple |
| 451 | 21 | Multiple |
| 451 | 23 | Multiple |
| 451 | 24 | Single |
| 451 | 24 | Single |
| 451 | 24 | Multiple |

รูปที่ 3.10 รูปแบบในการจัดเก็บผลลัพธ์จากโปรแกรมตรวจสอบลักษณะการประกาศตัวแปร

**3.2.2.2 การทำงานแบบเลือกทำ** จะเป็นการตรวจสอบลักษณะของการใช้การเลือกทำ โดยใช้การค้นหาจากกลุ่มคำที่ได้เตรียมไว้ที่จะประกอบไปด้วยคำดังต่อไปนี้ “if”, “else if”, “else” โดยการทำงานของโปรแกรมจะเริ่มจากการนำเข้าเอกสารรหัสต้นฉบับที่ต้องการตรวจสอบ ต่อมาระบบจะนำชื่อของไฟล์มาเก็บไว้ในตัวแปรที่ชื่อว่า “name” หลังจากที่โปรแกรมเก็บชื่อไฟล์เสร็จแล้วก็จะเริ่มทำการค้นหาจากคำที่ได้กำหนดไว้ และถ้าเจอคำที่ได้กำหนดไว้ คำใดคำหนึ่ง ณ ตำแหน่ง ใด ๆ ของรหัสต้นฉบับแล้ว ก็จะใช้สายอักขระย่อยในการแยกคำ ๆ นั้นออกมาแล้วนำไปเก็บไว้ในแถวลำดับที่ชื่อว่า “if_seq” เพื่อรอการสรุปผลในส่วนท้ายต่อไป โดยจะทำแบบนี้ซ้ำ ๆ ไปเรื่อย ๆ จนกว่าจะสิ้นสุดบรรทัดของเอกสารนั้น ๆ และเมื่อสิ้นสุดบรรทัดแล้วโปรแกรมก็จะทำการนำเอาแถวลำดับที่ได้มีการเก็บคำที่ได้แยกออกมา มาสรุปผลโดยจะแสดงการทำงานให้เห็นดังรูปที่ 3.11 โดยมีเงื่อนไขดังต่อไปนี้

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University




รูปที่ 3.11 การตรวจสอบลักษณะการทำงานแบบเลือกทำ

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

- ถ้าภายในแถวลำดับ if_seq นั้นประกอบด้วย “if” เพียงอย่างเดียวจะสามารถสรุปได้ว่า เอกสารรหัสต้นฉบับที่นำเข้ามาตรวจนั้นมีการใช้การเลือกทำแบบใช้ “if” เพียงอย่างเดียว
- ถ้าภายในแถวลำดับ if_seq นั้นประกอบด้วย “if” และ “else” ผสมกับอยู่ในแถวลำดับจะสามารถสรุปได้ว่า เอกสารรหัสต้นฉบับที่นำเข้ามาตรวจนั้นมีการใช้การเลือกทำแบบใช้ “if” และ “else” ผสมกันอยู่ภายในเอกสารรหัสต้นฉบับ
- ถ้าภายในแถวลำดับ if_seq นั้นประกอบด้วย “if” และ “else if” ผสมกับอยู่ในแถวลำดับจะสามารถสรุปได้ว่า เอกสารรหัสต้นฉบับที่นำเข้ามาตรวจนั้นมีการใช้การเลือกทำแบบใช้ “if” และ “else if” ผสมกันอยู่ภายในเอกสารรหัสต้นฉบับ

- ถ้าภายในแถวลำดับ if_seq นั้นประกอบด้วย “if”, “else if” และ “else” ผสมกับอยู่ในแถวลำดับจะสามารถสรุปได้ว่า เอกสารรหัสต้นฉบับที่นำเข้ามาตรวจนั้นมีการใช้การเลือกทำแบบใช้ “if” และ “else if” และ “else” ผสมกันอยู่ภายในเอกสารรหัสต้นฉบับ

หลังจากที่โปรแกรมทำการสรุปผลเรียบร้อยแล้วจะนำเอาชื่อของไฟล์ที่เก็บไว้ในตัวแปร “name” มาเขียนลงบนแฟ้มข้อมูลนามสกุลซีเอสวี โดยชื่อไฟล์ดังกล่าวจะมีการใช้สายอักขระย่อยในการแบ่งข้อมูลออกเป็น รหัสผู้ใช้ และรหัสโจทย์ในขั้นตอนที่มีชื่อว่า “making result” จากนั้นระบบจะทำการเขียนข้อมูลทั้งหมดลงบนไฟล์นามสกุลซีเอสวี โดยจะมีรูปแบบการเขียนเป็น รหัสผู้ใช้, รหัสโจทย์, ลักษณะการการใช้การเลือกทำ ดังรูปที่ 3.12 และนำไปเก็บไว้ในสารบบเพื่อรอการเรียกใช้ต่อไป

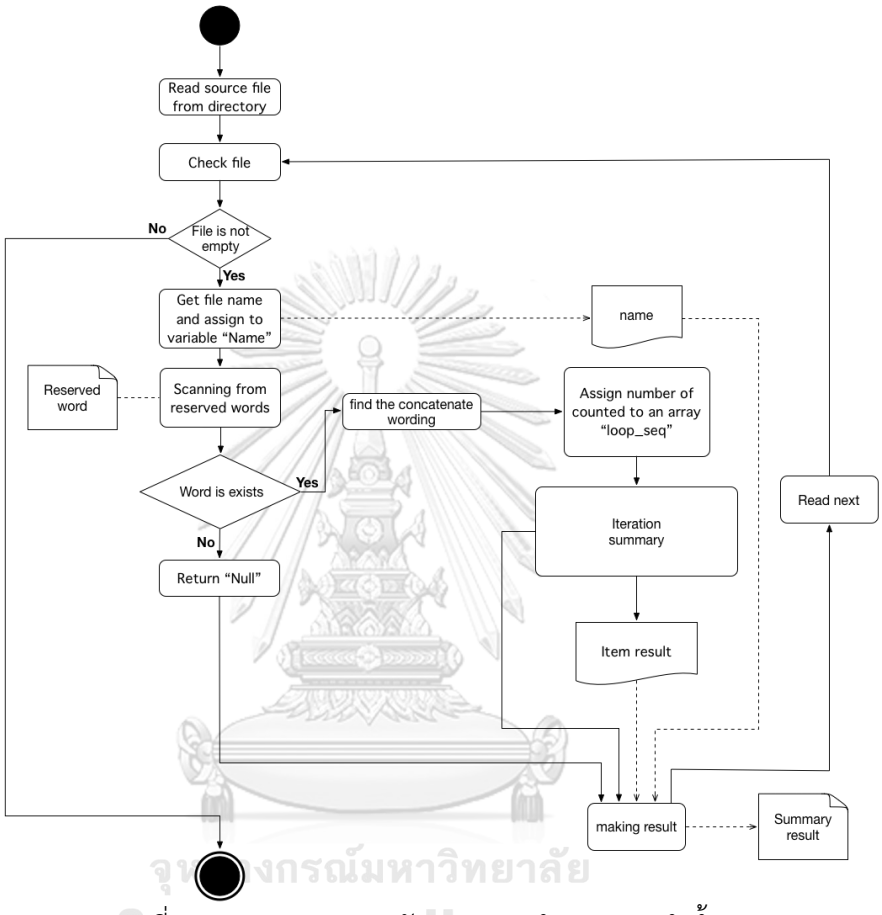
| USER_ID | PROBLEM_ID | SELECTION_STYLE |
|---|---|---|
| 451 | 21 | null |
| 451 | 21 | null |
| 451 | 23 | null |
| 451 | 24 | null |
| 451 | 24 | null |
| 451 | 24 | null |
| 451 | 25 | null |
| 451 | 26 | null |
| 451 | 27 | Only if |
| 451 | 28 | Only if |
| 451 | 28 | Only if |
| 451 | 29 | Only if |

รูปที่ 3.12 ผลลัพธ์จากการตรวจสอบลักษณะการเลือกทำ

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

**3.2.2.3 การทำงานแบบทำซ้ำ** จะเป็นการตรวจสอบลักษณะของการเลือกใช้ชนิดของการทำซ้ำ โดยในที่นี้จะตรวจสอบด้วยกันอยู่ 2 ประเภทคือ 1. การทำซ้ำแบบ “for” และ 2. การทำซ้ำแบบ “while” โดยใช้การค้นหาจากคำที่ได้กำหนดไว้ซึ่งได้แก่ “for(” และ “while(” โดยเริ่มต้นโปรแกรมจะไปอ่านแฟ้มข้อมูลเอกสารรหัสต้นฉบับจากสารบบที่เก็บไว้ ต่อมาระบบจะนำชื่อของไฟล์มาเก็บไว้ในตัวแปรที่ชื่อว่า “name” หลังจากที่จัดเก็บชื่อไฟล์ไปยังตัวแปร “name” เสร็จแล้วก็จะทำการค้นหาคำจากกลุ่มคำที่ได้กำหนดไว้ และเมื่อเจอคำที่ได้กำหนดไว้ปรากฏ ณ ตำแหน่งใด ๆ ในเอกสารหัสต้นฉบับแล้ว จะทำการใช้สายอักขระย่อยเพื่อที่จะแยกคำนั้น ๆ ออกมาเก็บไว้ในแถวลำดับที่มีชื่อว่า “loop_seq” เพื่อรอการสรุปผลในตอนท้ายต่อไป โดยจะทำแบบนี้ไปเรื่อย ๆ จนกว่าจะสิ้นสุดบรรทัดของเอกสารรหัสต้นฉบับนั้น ๆ หลังจากโปรแกรมทำการค้นหา

จนสิ้นสุดบรรทัดของเอกสารรหัสต้อนฉบับแล้ว จะนำเอาแถวลำดับที่ได้ทำการเก็บค่าที่ได้มาจากการใช้สายอักขระแยกคำออกมานั้น มาทำการสรุปผลโดยการทำงานทั้งหมด จะแสดงให้เห็นดังรูปที่ 3.13 ซึ่งมีเงื่อนไขดังต่อไปนี้




รูปที่ 3.13 การตรวจสอบลักษณะการทำงานแบบทำซ้ำ

- ถ้าภายในแถวลำดับประกอบไปด้วยการทำซ้ำชนิด “for” เพียงอย่างเดียว แสดงว่า เอกสารรหัสต้นฉบับนั้นมีการใช้งานการทำซ้ำแบบ “for” เพียงอย่างเดียว
- ถ้าภายในแถวลำดับประกอบไปด้วยการทำซ้ำชนิด “while” เพียงอย่างเดียว แสดงว่า เอกสารรหัสต้นฉบับนั้นมีการใช้งานการทำซ้ำแบบ “while” เพียงอย่างเดียว

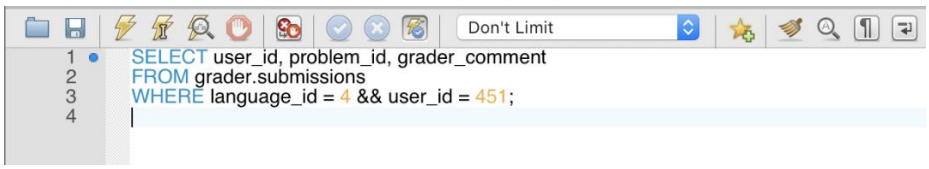
- ถ้าภายในแถวลำดับประกอบไปด้วยการทำซ้ำชนิด “for” และ “while” ผสมกันแสดงว่า เอกสารรหัสต้นฉบับนั้นมีการใช้งานการทำซ้ำแบบ “for” ผสมกับการทำซ้ำแบบ “while”

หลังจากที่โปรแกรมทำการสรุปผลเรียบร้อยแล้วจะนำเอาชื่อของไฟล์ที่เก็บไว้ในตัวแปร “name” มาเขียนลงบนแฟ้มข้อมูลนามสกุลซีเอสวี โดยชื่อไฟล์ดังกล่าวจะมีการใช้สารอักขระย่อยในการแบ่งข้อมูลออกเป็น รหัสผู้ใช้ และรหัสโจทย์ในขั้นตอนที่มีชื่อว่า “making result” จากนั้นระบบจะทำการเขียนข้อมูลทั้งหมดลงบนไฟล์นามสกุลซีเอสวี โดยจะมีรูปแบบการเขียนเป็น รหัสผู้ใช้, รหัสโจทย์, ลักษณะการการใช้การทำซ้ำ ดังรูปที่ 3.14 และนำไปเก็บไว้ในสารบบเพื่อรอการเรียกใช้ต่อไป

| USER_ID | PROBLEM_ID | REPETITION_STYLE |
|---|---|---|
| 451 | 21 | null |
| 451 | 21 | null |
| 451 | 23 | null |
| 451 | 24 | null |
| 451 | 24 | null |
| 451 | 24 | null |
| 451 | 25 | null |
| 451 | 26 | null |
| 451 | 27 | While |
| 451 | 28 | While |
| 451 | 28 | While |
| 451 | 29 | While |

รูปที่ 3.14 ผลลัพธ์จากการตรวจสอบลักษณะการทำซ้ำ

**3.2.2.4 หมายเหตุจากระบบตรวจอัตโนมัติ (grader comment)** การทำงานในส่วนนี้จะใช้การดึงข้อมูลโดยตรงจากฐานข้อมูลเชิงสัมพันธ์ของระบบตรวจอัตโนมัติโดยใช้การสอบถาม (query) จากฐานข้อมูลโดยตรง ดังรูปที่ 3.15 โดยหมายเหตุที่ดึงออกมาจะอยู่ในรูปของลำดับของสายอักขระ (sequence of string) ประกอบด้วยสัญลักษณ์ต่าง ๆ ได้แก่ “Pass”, “x”, “-” และ “Compilation Error” และใช้การบันทึกผลลัพธ์ออกมาเป็นแฟ้มข้อมูลซีเอสวีโดยตรงจากตัวโปรแกรม MySQL Workbench โดยผลลัพธ์ที่ถูกบันทึกนั้นจะบันทึกให้อยู่ในรูป “รหัสผู้ใช้, รหัสโจทย์, หมายเหตุ” ดังรูปที่ 3.16 ตามลำดับ



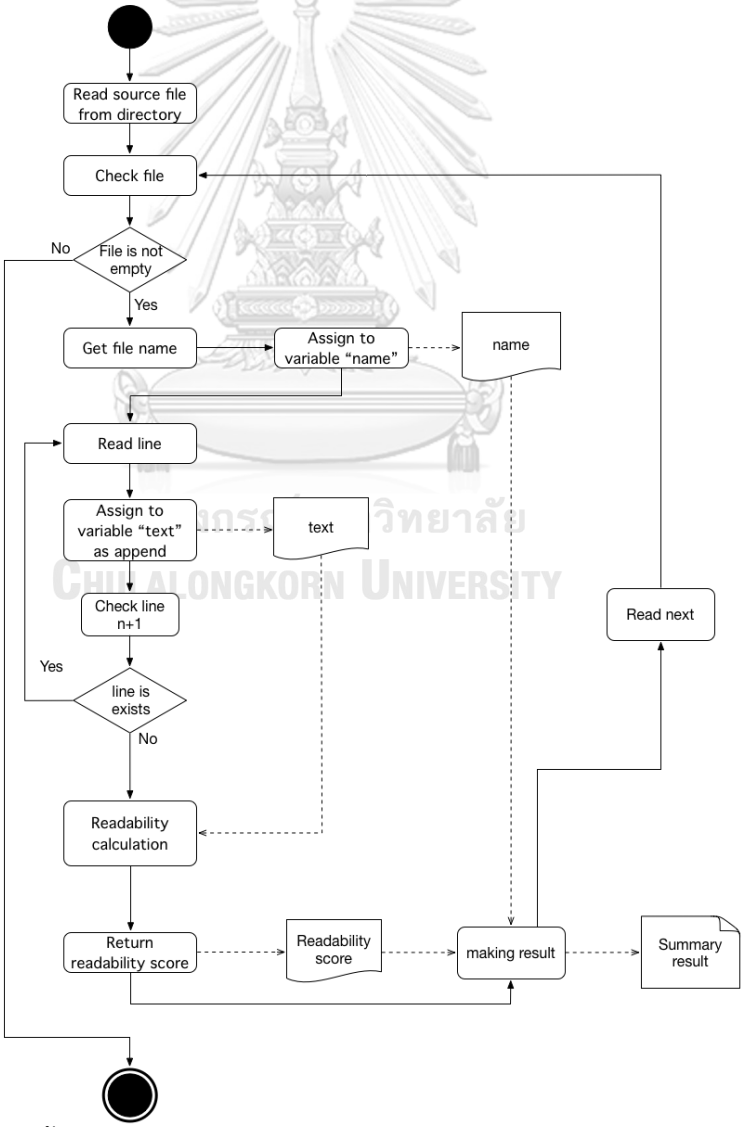

รูปที่ 3.15 การสอบถามหมายเหตุจากระบบตรวจอัตโนมัติจากฐานข้อมูล

| user_id | problem_id | grader_comment |
|---|---|---|
| 451 | 10 | PASSED: PPPPPPPP(inconsistent score) |
| 451 | 12 | PP |
| 451 | 21 | ----- |
| 451 | 21 | PPPPP |
| 451 | 23 | PPPPP |
| 451 | 24 | compilation error |
| 451 | 24 | ---- |
| 451 | 24 | PPPP |

รูปที่ 3.16 ผลลัพธ์ที่ได้มาจากการสอบถาม

**3.2.2.5 ความยากง่ายทางการอ่านของรหัสต้นฉบับ (source code readability)** การทำงานส่วนนี้จะเป็นการเรียกใช้งานโปรแกรมจากงานวิจัยของเรย์ม่อน บรูซ และคณะ (Raymond P.L. Buse et al.) [15] ที่ศึกษาเกี่ยวกับความยากง่ายทางการอ่านของรหัสต้นฉบับโดยโปรแกรมดังกล่าวมีชื่อว่า “readability” ซึ่งเป็นโปรแกรมนามสกุล “.jar” โดยในงานวิจัยชิ้นนี้จะใช้การเขียนโปรแกรมเพื่อไปเรียกใช้งานโปรแกรมดังกล่าว [15] โดยมีการทำงานดังนี้ เริ่มต้นโปรแกรมจะไปอ่านแฟ้มข้อมูลจากสารบบ ต่อมาระบบจะนำชื่อของไฟล์มาเก็บไว้ในตัวแปรที่ชื่อว่า “name” หลังจากที่จัดเก็บชื่อไฟล์ไปยังตัวแปร “name” เสร็จแล้วก็จะทำการอ่านข้อมูลภายในแฟ้มข้อมูลในส่วนของรหัสต้นฉบับทั้งหมดทุกบรรทัด และนำค่าไปเก็บไว้ในตัวแปรที่มีชื่อว่า “text” ต่อมาโปรแกรมจะไปทำการเรียกใช้คลังภายนอก (external library) ในขั้นตอนที่มีชื่อว่า “readability calculation” เพื่อมาใช้ในการคำนวณค่าความยากง่ายทางการอ่าน หลังจากโปรแกรมทำการคำนวณเสร็จจะเอาค่าที่ได้ไปเก็บไว้ที่ตัวแปรที่มีชื่อว่า “readability_score” ต่อมาโปรแกรมจะนำค่าของตัวแปรทั้งสามตัว ได้แก่ name, problemid และ readability_score มาบันทึกลงไฟล์ หลังจากที่โปรแกรมทำการคำนวณค่าความยากง่ายทางการอ่านเป็นที่เรียบร้อยแล้วจะนำเอาชื่อของไฟล์ที่เก็บไว้ในตัวแปร “name” มาเขียนลงบนแฟ้มข้อมูลนามสกุลซีเอสวี โดยชื่อไฟล์ดังกล่าวจะมีการใช้สารอักขระย่อยในการแบ่งข้อมูลออกเป็น รหัสผู้ใช้ และรหัสโจทย์ในขั้นตอนที่มีชื่อว่า “making result” โดยจะมีรูปแบบการเขียนเป็น “รหัสผู้ใช้ (USER_ID) รหัสโจทย์ (PROBLEM_ID) และคะแนนความยากง่ายทางการอ่าน

(READABILITY_SCORE)” ดังรูปที่ 3.17 หลังจากที่ได้ผลลัพธ์จาก 4 คุณลักษณะที่กล่าวมา ได้แก่ 1. เวกเตอร์ของผู้ใช้ 2. ผลจากการตรวจสอบการใช้เครื่องมือทางภาษาโปรแกรม 3. หมายเหตุที่ตอบกลับจากระบบตรวจอัตโนมัติ และ 4. ผลจากการตรวจสอบความยากง่ายทางการอ่านของรหัสต้นฉบับก็จะนำข้อมูลคุณลักษณะทั้ง 4 คุณลักษณะมารวมกันโดยใช้การคัดลอกข้อมูลจากทั้ง 4 แฟ้มข้อมูลมาวางลงบนแฟ้มข้อมูลใหม่ผ่านทางการใช้โปรแกรม MS Excel โดยจะบันทึกเป็นแฟ้มข้อมูลนามสกุลซีเอสวีจะได้ผลลัพธ์ดังรูปที่ 3.18 จากนั้นจะนำแฟ้มข้อมูลดังกล่าวเข้าสู่ฐานข้อมูลเชิงสัมพันธ์ที่ชื่อว่า “USER_APTITUDE” ที่จะประกอบไปด้วยเขตข้อมูล และชนิดของข้อมูลของแต่ละเขตข้อมูลดังรูปที่ 3.19 ด้วยโปรแกรม MySQL Workbench ตามลำดับ




รูปที่ 3.17 ขั้นตอนการเรียกใช้เครื่องมือเพื่อคำนวณค่าความยากง่ายทางการอ่าน

| ID | USER_ID | PROBLEM_ID | SUBMITTED_DATE | SUBMITTED_TIME | LANGUAGE_TOOL | PERFORMANCE_SCORE | READABILITY_SCORE | OBJECTIVE_STATUS | GRADER_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 451 | 21 | 19/08/2014 | 8:16:46 | 0-0-3 | 0 | 0 | Pass | ----- |
| 2 | 451 | 21 | 19/08/2014 | 8:24:21 | 0-0-3 | 100 | 0 | Pass | PPPPP |
| 3 | 451 | 23 | 19/08/2014 | 8:04:38 | 0-0-3 | 100 | 0 | Pass | PPPPP |
| 4 | 451 | 24 | 19/08/2014 | 7:35:23 | 0-0-2 | 0 | 1 | Pass | compilation error |
| 5 | 451 | 24 | 19/08/2014 | 7:38:30 | 0-0-2 | 0 | 1 | Pass | ---- |
| 6 | 451 | 24 | 19/08/2014 | 7:48:01 | 0-0-3 | 100 | 1 | Pass | PPPP |
| 7 | 451 | 25 | 19/08/2014 | 7:54:20 | 0-0-3 | 100 | 1 | Pass | PPPP |
| 8 | 451 | 26 | 19/08/2014 | 7:31:40 | 0-0-3 | 100 | 1 | Pass | PPPPP |
| 9 | 451 | 27 | 14/09/2014 | 16:27:31 | 2-1-3 | 100 | 1 | Pass | PPPPP |
| 10 | 451 | 28 | 26/08/2014 | 2:15:28 | 2-1-3 | 0 | 0 | Pass | compilation error |
| 11 | 451 | 28 | 26/08/2014 | 2:16:36 | 2-1-3 | 100 | 1 | Pass | PPPPPPPPPP |

รูปที่ 3.18 ผลจากการรวมข้อมูลจากสี่แฟ้มข้อมูล

| Column | Datatype | PK | NN | UQ | BIN | UN | ZF | AI | G | Default / Expression |
|---|---|---|---|---|---|---|---|---|---|---|
| ID | INT(11) | ✓ | ✓ | | | | | ✓ | | |
| USER_ID | INT(11) | | | | | | | | | NULL |
| PROBLEM_ID | INT(11) | | | | | | | | | NULL |
| SUBMITTED_... | MEDIUMTEXT | | | | | | | | | NULL |
| SUBMITTED_... | MEDIUMTEXT | | | | | | | | | NULL |
| LANGUAGE_T... | MEDIUMTEXT | | | | | | | | | NULL |
| PERFORMAN... | INT(11) | | | | | | | | | NULL |
| READABILITY... | DECIMAL(3,0) | | | | | | | | | NULL |
| OBJECTIVE_... | MEDIUMTEXT | | | | | | | | | NULL |
| GRADER_CO... | MEDIUMTEXT | | | | | | | | | NULL |
| <click to edit> | | | | | | | | | | |

รูปที่ 3.19 รายชื่อเขตข้อมูลและชนิดของข้อมูลของแต่ละเขตข้อมูลภายในฐานข้อมูลเชิงสัมพันธ์

### 3.3 การประเมินทักษะการเขียนโปรแกรม

การประเมินทักษะด้านการเขียนโปรแกรมนี้จะพิจารณา 2 ด้านคือ 1. เวลาที่ใช้เขียนโปรแกรมในแต่ละข้อ (time consumption) และ 2. จำนวนครั้งของการส่งโปรแกรมขึ้นตรวจในแต่ละข้อการเขียน (submission frequency) โดยไม่สนใจเวลาที่ใช้ไปในการเขียนโปรแกรม โดยจะอธิบายตามลำดับดังนี้

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

**3.3.1 เวลาที่ใช้เขียนโปรแกรม** นี้จะหมายถึงเวลาที่ใช้ไปในการเขียนโปรแกรม 1 ข้อโดยไม่สนใจจำนวนครั้งของการส่ง (versions) ก็ตามโดยข้อมูลที่นำเข้ามาตรวจสอบนี้จะถูกนำมาจากฐานข้อมูลเชิงสัมพันธ์ (relational database) เช่นเดียวกันกับที่ใช้ในการค้นหาความถนัดในการเขียนโปรแกรม โดยเขียนโปรแกรมเพื่อเรียกใช้งานส่วนจำเพาะ (module) ที่เรียกว่า “Python database connector” สำหรับการนำข้อมูลออกมาจากฐานข้อมูลเพื่อที่จะนำมาใช้ในการประเมินทักษะ โดยที่จะนำเขตข้อมูลจากที่เคยดึงมาแล้วจากขั้นตอนการค้นหาความถนัดของการเขียนโปรแกรมมาใช้ต่อ แต่เนื่องจากมีข้อมูลบางส่วนที่ไม่ได้ใช้งานจึงใช้ตัวกรองให้เหลือเฉพาะข้อมูลที่ใช้งานจริง ๆ จึงทำให้ข้อมูลที่ดึงออกมาในช่วงที่ทำการค้นหาความถนัดในการเขียนโปรแกรมนั้นที่มีอยู่ทั้งหมด 7 เขตข้อมูลจะถูกลดลงมาเหลืออยู่เพียง 3 เขตข้อมูลประกอบด้วย “user_id”, “problem_id” และ “submitted_at” เพื่อที่จะนำไปใช้ในการประเมินทักษะต่อไป โดยเขตข้อมูลที่ชื่อว่า “submitted_at” นี้คือเขตข้อมูลที่เก็บวันและเวลาของผู้เขียนที่ส่งโปรแกรมขึ้นตรวจบนระบบตรวจ

อัตโนมัติที่จะเก็บอยู่ในรูปของ “yyyy-mm-dd hh:mm:ss” ดังรูปที่ 3.20 โดยที่สัญลักษณ์ต่าง ๆ จะมีความหมายดังนี้

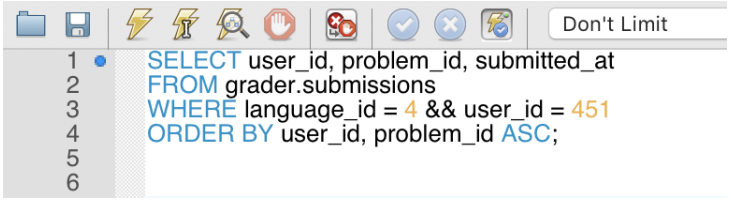
- yyyy คือ ปี (year) ในที่นี้จะเก็บอยู่ในรูปของปี ค.ศ. หรือ คริสต์ศักราช ตัวอย่างเช่น 2014 เป็นต้น
- mm คือ เดือน (month) ที่ผู้เขียนส่งโปรแกรมขึ้นตรวจจะเป็นตัวเลขเริ่มต้นตั้งแต่ 01 - 12
- dd คือ วันที่ (day) ที่ผู้เขียนส่งโปรแกรมขึ้นตรวจจะเป็นตัวเลขเริ่มตั้งแต่ 01 - 31
- hh คือ เวลาที่ผู้เขียนส่งโปรแกรมขึ้นตรวจมีหน่วยเป็น ชั่วโมง (hour) เริ่มตั้งแต่ 00 - 23
- mm คือ เวลาที่ผู้เขียนส่งโปรแกรมขึ้นตรวจ มีหน่วยเป็นนาที (minute) เริ่มตั้งแต่ 00 – 59
- ss คือ เวลาที่ผู้เขียนส่งโปรแกรมขึ้นตรวจ มีหน่วยเป็นวินาที (second) เริ่มตั้งแต่ 00 – 59

| user_id | problem_id | submitted_at |
|---|---|---|
| 451 | 34 | 2014-09-02 02:16:49 |
| 451 | 34 | 2014-09-09 03:39:24 |
| 451 | 34 | 2014-09-09 04:19:47 |
| 451 | 34 | 2014-09-23 14:14:47 |

รูปที่ 3.20 วันและเวลาของการส่งที่เก็บอยู่ในฐานข้อมูลเชิงสัมพันธ์

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

หลังจากที่ได้ทำการดึงข้อมูลออกมาจากฐานข้อมูลแล้วจะนำข้อมูลดังกล่าวไปบันทึกเป็นแฟ้มข้อมูลนามสกุลซีเอสวี (comma-separated values) ในขั้นตอนการนำข้อมูลออกมานั้นจะใช้การเรียงลำดับ (sort) ด้วยการสอบถามด้วยภาษาซีเควล (SQL) ดังรูปที่ 3.21 โดยบันทึกแยกเป็นรายคนดังรูปที่ 3.22 ตามลำดับ

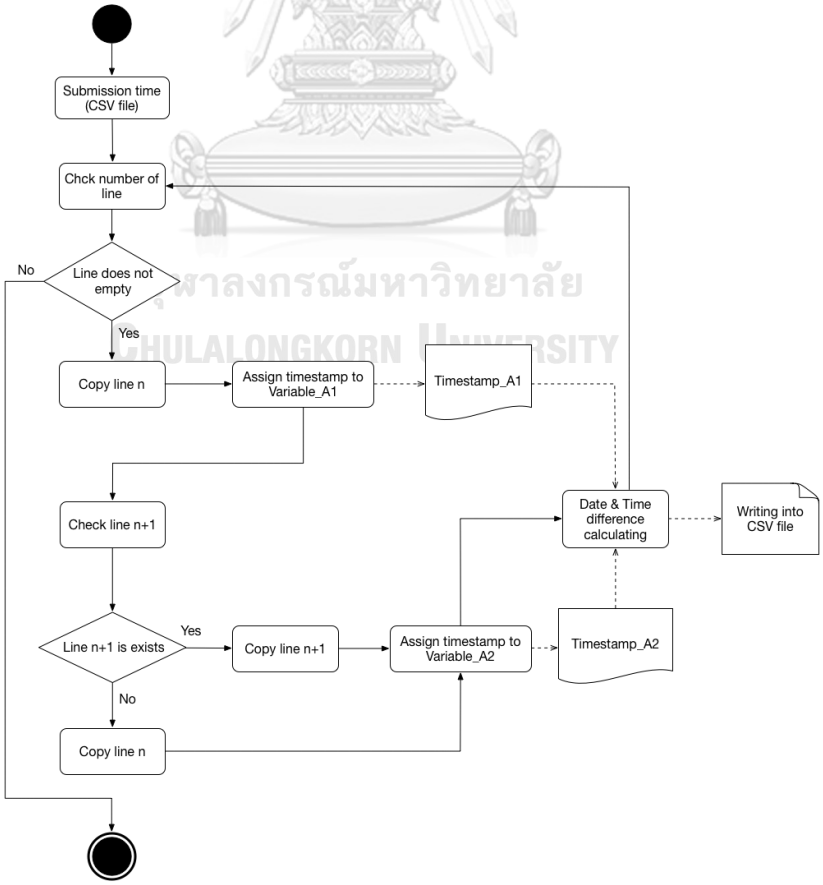



รูปที่ 3.21 ภาษาซีเควลที่ใช้ในการสอบถามวันเวลาที่ผู้เขียนส่งโปรแกรม

| user_id | problem_id | submitted_at |
|---|---|---|
| 451 | 10 | 9/21/14 15:05 |
| 451 | 12 | 8/19/14 2:23 |
| 451 | 21 | 8/19/14 8:16 |
| 451 | 21 | 8/19/14 8:24 |
| 451 | 23 | 8/19/14 8:04 |
| 451 | 24 | 8/19/14 7:35 |
| 451 | 24 | 8/19/14 7:38 |
| 451 | 24 | 8/19/14 7:48 |
| 451 | 25 | 8/19/14 7:54 |
| 451 | 26 | 8/19/14 7:31 |
| 451 | 27 | 9/14/14 16:27 |
| 451 | 28 | 8/26/14 2:15 |

รูปที่ 3.22 การจัดเก็บเวลาที่ผู้เขียนส่งโปรแกรมขึ้นมาตรวจ

หลังจากที่ได้เวลาในการส่งแล้วจะทำการป้อน (feed) เข้าสู่โปรแกรมคำนวณหาความต่างของเวลา (time difference) ที่ถูกเขียนขึ้นด้วยภาษาจาวาซึ่งจะมีการทำงานดังรูปที่ 3.23




รูปที่ 3.23 การทำงานของการบันทึกเวลาเพื่อการคำนวณเวลาที่ใช้ไปในการเขียนโปรแกรม

จากรูปที่ 3.23 แสดงให้เห็นถึงการทำงานของโปรแกรมการคำนวณหาความต่างของเวลาเพื่อที่จะนำไปในการคำนวณหาระยะเวลาที่ใช้ในการเขียนโปรแกรมจะมีการทำงานดังต่อไปนี้ เริ่มจากโปรแกรมจะไปอ่านแฟ้มข้อมูลซีเอสวีที่ได้มาจาก การนำข้อมูลวัน และเวลาออกมาจากฐานข้อมูลเชิงสัมพันธ์ของระบบตรวจอัตโนมัติ (grader database) หลังจากนั้นโปรแกรมจะเริ่มทำการวนลูปเพื่ออ่านบรรทัดไปทีละบรรทัด โดยจะมีการข้ามไปอ่านบรรทัดถัดไปเพื่อทำการคัดลอกค่าของแต่ละบรรทัดไปกำหนดค่า (assign) ให้กับตัวแปร “Variable_A1 & Variable_A2” เพื่อใช้ในการคำนวณหาความต่างของเวลา ตัวอย่างเช่น เมื่อโปรแกรมอ่านบรรทัดที่ 1 แล้วจะนำบรรทัดที่ 1 ไปเก็บไว้ในตัวแปรที่มีชื่อว่า “Variable_A1” และจะมีการข้ามไปอ่านบรรทัดที่ 2 แล้วคัดลอกบรรทัดที่ 2 เก็บไว้ในตัวแปรที่มีชื่อว่า “Variable_A2” แล้วนำค่าที่เก็บไว้ในตัวแปรทั้ง “Variable_A1” และ “Variable_A2” มาทำการหาความต่างของเวลาโดยจะนำเอา “Variable_A1” เป็นตัวตั้งและลบด้วย “Variable_A2” โดยผลลัพธ์ที่ออกมาจะอยู่ในรูปของ “USER_ID, PROBLEM_ID, HOURS, MINUTES, SECONDS” ดังรูปที่ 3.24

| USER_ID | PROBLEM_ID | HOURS | MINUTES | SECONDS |
|---|---|---|---|---|
| 451 | 12 | 5 | 8 | 4 |
| 451 | 26 | 0 | 3 | 43 |
| 451 | 24 | 0 | 3 | 7 |
| 451 | 24 | 0 | 9 | 31 |
| 451 | 24 | 0 | 6 | 19 |
| 451 | 25 | 0 | 10 | 18 |
| 451 | 23 | 0 | 12 | 8 |
| 451 | 21 | 0 | 7 | 35 |
| 451 | 21 | 6 | 8 | 53 |
| 451 | 28 | 0 | 1 | 8 |

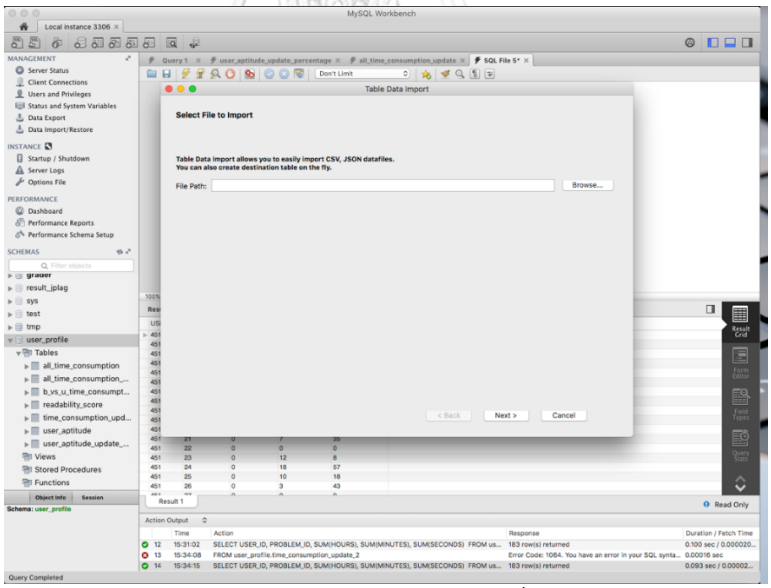
รูปที่ 3.24 ผลลัพธ์ที่ถูกบันทึกจากโปรแกรมคำนวณหาความต่างของเวลา

ภายหลังจากการคำนวณเสร็จสิ้นแล้วจะบันทึกผลลัพธ์ลงแฟ้มข้อมูลนามสกุลซีเอสวี (CSV) ในลำดับถัดไป โดยการทำงานของโปรแกรมหาค่าความต่างของเวลานี้จะทำงานอยู่ภายใต้เงื่อนไขสองข้อ ได้แก่

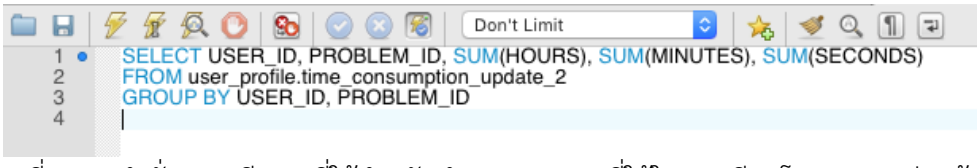
- ถ้าโปรแกรมมีการตรวจพบบรรทัดของวันที่และเวลาที่ส่ง (submitted date & time) ครั้งแรก และครั้งถัดไปแล้ว โปรแกรมจะนำวัน และเวลาที่ส่งของแต่ละครั้งมาคำนวณหาความต่างของเวลา

- ถ้าโปรแกรมมีการตรวจพบบรรทัดของวันที่และเวลาที่ส่ง (submitted date & time) แล้วไม่เจอการส่งครั้งถัดไปแล้ว โปรแกรมจะหยุดการทำงานทันที และโปรแกรมจะบันทึกค่าความต่างของเวลาเป็น “0” ลงแฟ้มข้อมูลในลำดับถัดไป

หลังจากที่ได้ความต่างของเวลาจากการส่งโปรแกรมขึ้นตรวจมาแล้วต่อไปก็จะเป็นการคำนวณหาเวลารวมของการเขียนโปรแกรมของโปรแกรมแต่ละข้อ โดยนำข้อมูลที่ส่งออกจากโปรแกรมคำนวณหาความต่างของเวลาที่อยู่ในรูปของแฟ้มข้อมูลซีเอสวี นำไปนำเข้าสู่ฐานข้อมูลเชิงสัมพันธ์โดยใช้การนำเข้าจากเมนู “Table Data Import” ของโปรแกรม MySQL Workbench ดังรูปที่ 3.25 โดยภายหลังจากกระบวนการนำเข้าสำเร็จแล้ว ต่อมาก็ใช้คำสั่งของภาษาซีเควล (SQL) ทำการคำนวณหาเวลาที่ใช้ไปในการเขียนโปรแกรมหนึ่งข้อโดยคำสั่งที่ใช้งานจะใช้คำสั่งการหาผลรวม (summation) โดยจะมีการหาผลรวมแบบแยกเป็นเขตข้อมูล ได้แก่ เขตข้อมูล ชั่วโมง (HOURS) นาที (MINUTES) และ วินาที (SECONDS) และในบรรทัดสุดท้ายจะเป็นการสั่งให้รวมกลุ่มของข้อมูลที่เหมือนกันภายใต้เงื่อนไขที่ได้กำหนดไว้ ในที่นี้คือ กำหนดให้รวมข้อมูลภายใต้ผู้ใช้คนเดียวกัน และข้อเดียวกัน “USER_ID, PROBLEM_ID” ดังรูปที่ 3.26




รูปที่ 3.25 รายการเลือกการนำเข้าข้อมูลจากแฟ้มข้อมูลข้อมูลที่ได้จากโปรแกรมคำนวณความต่างของเวลา




รูปที่ 3.26 คำสั่งภาษาซีเควลที่ใช้สำหรับคำนวณหาเวลาที่ใช้ในการเขียนโปรแกรมแต่ละข้อ

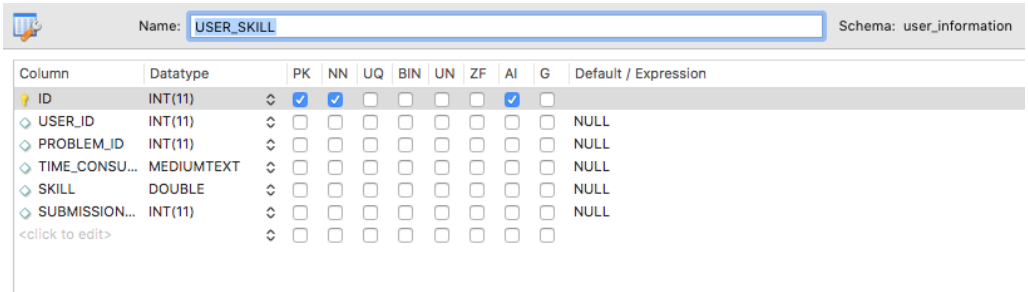
หลังจากการคำนวณหาเวลารวมของการเขียนโปรแกรมหนึ่งข้อแล้ว จะได้ผลลัพธ์ดังรูปที่ 3.27

| USER_ID | PROBLEM_ID | HOURS | MINUTES | SECONDS |
|---|---|---|---|---|
| 451 | 12 | 5 | 8 | 4 |
| 451 | 26 | 0 | 3 | 43 |
| 451 | 24 | 0 | 3 | 7 |
| 451 | 24 | 0 | 9 | 31 |
| 451 | 24 | 0 | 6 | 19 |
| 451 | 25 | 0 | 10 | 18 |
| 451 | 23 | 0 | 12 | 8 |
| 451 | 21 | 0 | 7 | 35 |
| 451 | 21 | 6 | 8 | 53 |
| 451 | 28 | 0 | 1 | 8 |

| USER_ID | PROBLEM_ID | SUM(HOURS) | SUM(MINUTES) | SUM(SECONDS) |
|---|---|---|---|---|
| 451 | 10 | 0 | 0 | 0 |
| 451 | 11 | 0 | 0 | 0 |
| 451 | 12 | 5 | 8 | 4 |
| 451 | 13 | 0 | 0 | 0 |
| 451 | 14 | 0 | 0 | 0 |
| 451 | 15 | 0 | 0 | 0 |
| 451 | 16 | 0 | 0 | 0 |
| 451 | 17 | 0 | 0 | 0 |
| 451 | 18 | 0 | 0 | 0 |
| 451 | 19 | 0 | 0 | 0 |
| 451 | 20 | 0 | 0 | 0 |
| 451 | 21 | 0 | 7 | 35 |
| 451 | 22 | 0 | 0 | 0 |
| 451 | 23 | 0 | 12 | 8 |
| 451 | 24 | 0 | 18 | 57 |

รูปที่ 3.27 ข้อมูลที่เปลี่ยนแปลงไปหลังจากการคำนวณหาเวลารวมในการเขียนโปรแกรมผ่านการใช้ภาษาซีเควล จากโปรแกรม MySQL Workbench

หลังจากที่ได้ข้อมูลที่ได้มาจากการคำนวณแล้วจะนำ เอาข้อมูลนี้เข้าไปเก็บไว้ในฐานข้อมูลเชิงสัมพันธ์ของผู้ใช้ทั้งหมดโดยฐานข้อมูลจะถูกสร้างขึ้นใหม่จะประกอบไปด้วย 6 เขตข้อมูลได้แก่ รหัสชี้ตำแหน่ง (ID) รหัสผู้ใช้ (USER_ID) รหัสโจทย์ (PROBLEM_ID) เวลารวมที่ใช้ไปในการเขียนโปรแกรมข้อนั้น ๆ (TIME_CONSUMPTION) คะแนนทักษะ (SKILL) และจำนวนครั้งของการส่ง (SUBMISSION_FREQUENCY) โดยที่จะมีรหัสชี้ตำแหน่งเป็นกุญแจหลัก (primary key) และจะมีการกำหนดชนิดของข้อมูลในแต่ละเขตข้อมูลดังรูปภาพที่ 3.28




รูปที่ 3.28 ชนิดของข้อมูลในฐานข้อมูลเชิงสัมพันธ์ในส่วนของทักษะ

หลังจากที่ได้สร้างฐานข้อมูลเสร็จแล้ว จะเป็นการคำนวณเพื่อหาค่าของทักษะเพื่อประเมินทักษะในลำดับถัดไป จากผลลัพธ์ที่ได้มาจากการคำนวณเพื่อหาความต่างของเวลาจากการใช้โปรแกรมคำนวณ

ดังรูปที่ 3.27 นั้น สังเกตว่าในส่วนของเวลานั้นจะอยู่ในรูปของ ชั่วโมง นาที และวินาที เช่นนั้นจึงจำเป็นที่จะต้องปรับรูปของเวลาให้อยู่ในหน่วยเดียวกันให้หมด เพื่อที่จะได้เกิดความสะดวกในการคำนวณหาค่าของทักษะ โดยหน่วยที่เลือกมานั้นจะใช้หน่วย นาที (minute) และจะมีวิธีการทำดังนี้

- เปิดแฟ้มข้อมูลข้อมูลด้วย MS Excel
- คลิกเลือกเขตข้อมูล (field) ที่ต้องการที่จะแปลง
- ใส่สูตรการแปลง “((จำนวนชั่วโมง * 60) + (จำนวนนาที) + (จำนวนวินาที / 60))”
- ทำกับทุก ๆ ข้อมูลที่จะทำการแปลง

โดยข้อมูลหลังจากการแปลงเสร็จสิ้นจะออกมาดังรูปที่ 3.29 หลังจากที่ได้ระยะเวลารวมที่มีหน่วยเป็นนาทีแล้ว ต่อไปก็จะเป็นการคำนวณค่าของทักษะต่อไป โดยจะใช้ข้อมูลของเวลารวมกับข้อมูลของเวลาจำกัด นำทั้งสองข้อมูลเข้าสู่สมการการคำนวณโดยสมการดังกล่าวมีที่มาแนวคิดที่จะสร้างคะแนนทักษะที่อยู่ในช่วง -1 ถึง 0 และ 0 ถึง 1 โดยจะนำเอาเวลารวมของผู้เขียนที่ใช้ไปในการเขียนโปรแกรมหนึ่งข้อมาหารด้วยค่าเฉลี่ยของเวลาจำกัดที่ได้มาจากการนำ เวลาที่ใช้ไปในการเขียนโปรแกรมหนึ่งข้อของกลุ่มนิสิตที่เป็นนิสิตระดับโอลิมปิคจำนวน 7 คนมาหาค่าเฉลี่ยของเวลาที่ใช้ไปในการเขียนโปรแกรมหนึ่งข้อ และนำผลลัพธ์จากการหารครั้งนี้มาลบออกจากหนึ่งที่มีค่าเป็นจำนวนเต็มบวก แต่จะมีตัวแปรเสริมขึ้นมาชื่อว่าเดลต้าที สำหรับการปรับค่าคะแนนทักษะโดยที่ค่าของตัวแปรเดลต้าทีนี้จะถูกกำหนดโดยผู้ใช้ โดยที่ค่าของตัวแปรเดลต้าทีนี้จะมีค่าไม่เกินสองเท่าของเวลาจำกัด ($2B$) เนื่องจากในขั้นตอนการคำนวณทักษะในการเขียนโปรแกรมนั้น การที่ผู้เรียนการเขียนโปรแกรมใช้เวลาในการเขียนโปรแกรม 1 ข้อมากกว่าเวลาจำกัดถึง 1 เท่าตัวของเวลาจำกัดแล้วถือว่าผู้เรียนการเขียนโปรแกรมคนนั้นมีทักษะในการเขียนโปรแกรมที่ไม่ดีเป็นอย่างยิ่งเนื่องจากใช้เวลาในการเขียนโปรแกรม 1 ข้อมากเกินไป ซึ่งจะมีขั้นตอนดังนี้

1. เปิดแฟ้มข้อมูลข้อมูลด้วย MS Excel
2. คัดลอกข้อมูลของเวลาจำกัดมาวางต่อท้ายข้อมูลเวลาเดิม
3. คลิกเลือกในเขตข้อมูลที่ต้องการคำนวณภายใต้หัวข้อ Skill
4. ใส่สูตรการคำนวณ “(1 - (เวลารวมที่ใช้ไป (หน่วยเป็นนาที) / เวลาจำกัด (หน่วยเป็นนาที)))” ดังสมการที่ 3.1
5. ทำกับทุก ๆ ข้อมูลที่ต้องการจะคำนวณ

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

$$Skill = \left(1 - \frac{U}{B}\right) \pm \Delta T$$

**สมการที่ 3.1 การคำนวณหาค่าทักษะ**

โดยที่

$U$ คือ เวลาของผู้เรียนการเขียนโปรแกรมที่ใช้ไปในการเขียนโปรแกรม 1 ข้อ

$B$ คือ เวลาจำกัดที่ได้มาจากการหาค่าเฉลี่ยเวลาที่ใช้ไปในการเขียนโปรแกรม 1 ข้อของกลุ่มนิสิตที่เป็นนิสิตระดับโอลิมปิคจำนวน 7 คน

$\Delta T$ คือ ตัวแปรที่ใช้สำหรับการปรับค่าทักษะ (กำหนดโดยผู้ใช้ และค่าที่กำหนดห้ามเกิน 2 เท่าของเวลาจำกัด)

หลังจากคำนวณเสร็จแล้วจะได้ผลลัพธ์ดังรูปที่ 3.30 ตามลำดับโดยผลลัพธ์นี้จะไม่สามารถนำไปใช้ได้ เนื่องจากยังขาดข้อมูลอยู่อีก 1 ส่วนนั้นก็คือจำนวนครั้งของการส่ง

**3.3.2 จำนวนครั้งของการส่ง** นั้นได้มาจากการสอบถามจากฐานข้อมูลเชิงสัมพันธ์ของระบบตรวจอัตโนมัติ โดยจะใช้รายการสอบถามดังรูปที่ 3.31 และภายหลังจากได้ผลลัพธ์จากการสอบถามแล้ว ก็จะคัดลอกผลลัพธ์ดังกล่าวไปต่อท้ายค่าของทักษะที่ได้คำนวณไว้ก่อนหน้า ดังรูปที่ 3.32 หลังจากนั้นจึงนำข้อมูลทั้งดังกล่าวเข้าสู่ฐานข้อมูลเชิงสัมพันธ์ที่ได้สร้างไว้ก่อนหน้านี้

| USER_ID | PROBLEM_ID | HOUR | MINUTE | SECOND | TIME_CONSUMPTION |
|---|---|---|---|---|---|
| 451 | 10 | 1 | 6 | 32 | 66.53 |
| 451 | 11 | 0 | 0 | 0 | 0 |
| 451 | 12 | 5 | 8 | 4 | 308.07 |
| 451 | 13 | 0 | 0 | 0 | 0 |
| 451 | 14 | 0 | 0 | 0 | 0 |
| 451 | 15 | 0 | 0 | 0 | 0 |
| 451 | 16 | 0 | 0 | 0 | 0 |
| 451 | 17 | 0 | 0 | 0 | 0 |
| 451 | 18 | 0 | 0 | 0 | 0 |
| 451 | 19 | 0 | 0 | 0 | 0 |
| 451 | 20 | 0 | 0 | 0 | 0 |
| 451 | 21 | 17 | 58 | 42 | 1078.70 |
| 451 | 22 | 0 | 0 | 0 | 0 |
| 451 | 23 | 0 | 12 | 8 | 12.13 |
| 451 | 24 | 0 | 18 | 57 | 18.95 |
| 451 | 25 | 0 | 10 | 18 | 10.3 |

รูปที่ 3.29 ข้อมูลหลังจากการแปลงหน่วยของเวลา

| USER_ID | PROBLEM_ID | HOUR | MINUTE | SECOND | TIME_CONSUMPTION | TIME_LIMIT | SKILL |
|---|---|---|---|---|---|---|---|
| 451 | 10 | 1 | 6 | 32 | 66.53 | 0 | 0 |
| 451 | 11 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 12 | 5 | 8 | 4 | 308.07 | 15.39 | -19.02 |
| 451 | 13 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 14 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 15 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 16 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 17 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 18 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 19 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 20 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 21 | 17 | 58 | 42 | 1078.70 | 7.07 | -151.57 |
| 451 | 22 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 23 | 0 | 12 | 8 | 12.13 | 224.5 | 0.95 |
| 451 | 24 | 0 | 18 | 57 | 18.95 | 119.95 | 0.84 |
| 451 | 25 | 0 | 10 | 18 | 10.3 | 7.07 | -0.46 |

รูปที่ 3.30 ผลลัพธ์จากการคำนวณหาค่าทักษะ

```
1
2 •  SELECT user_id, problem_id, COUNT(*) as SUBMISSION_FREQUENCY
3    FROM grader.submissions
4    WHERE language_id = 4 && user_id >= 451
5    GROUP BY user_id, problem_id;
6
```

รูปที่ 3.31 การสอบถามจำนวนครั้งของการส่งจากฐานข้อมูลเชิงสัมพันธ์

| USER_ID | PROBLEM_ID | HOUR | MINUTE | SECOND | TIME_CONSUMPTION | TIME_LIMIT | SKILL | SUBMISSION_FREQUENCY |
|---|---|---|---|---|---|---|---|---|
| 451 | 10 | 1 | 6 | 32 | 66.53 | 0 | 0 | 1 |
| 451 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 12 | 5 | 8 | 4 | 308.07 | 15.39 | -19.02 | 1 |
| 451 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 21 | 17 | 58 | 42 | 1078.70 | 7.07 | -151.57 | 2 |
| 451 | 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 451 | 23 | 0 | 12 | 8 | 12.13 | 224.5 | 0.95 | 1 |
| 451 | 24 | 0 | 18 | 57 | 18.95 | 119.95 | 0.84 | 3 |
| 451 | 25 | 0 | 10 | 18 | 10.3 | 7.07 | -0.46 | 1 |

รูปที่ 3.32 ข้อมูลทั้งหมดของผู้ใช้ด้านทักษะการเขียนโปรแกรม

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

หลังจากที่ได้ข้อมูลทางด้านทักษะของการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรมแล้วจะนำข้อมูลดังกล่าวมาพักไว้ เพื่อรอการวิเคราะห์ร่วมกับ ความถนัดทางด้านการเขียนโปรแกรมต่อไป ซึ่งในส่วนถัดไปจะเป็นการอธิบายถึงวิธีการและขั้นตอนในการได้มาซึ่งความถนัดในการเขียนโปรแกรม

# บทที่ 4
# การวิเคราะห์ผล

หลังจากที่ได้ข้อมูลทั้งความถนัดในการเขียนโปรแกรม และทักษะในการเขียนโปรแกรมแล้ว ขั้นตอนต่อไปก็จะเป็นการวิเคราะห์ผลที่ได้มาโดยจะแบ่งการวิเคราะห์เป็น 2 ด้านคือ 1. การวิเคราะห์ทางด้านความถนัดทางด้านการเขียนโปรแกรม และ 2. การวิเคราะห์ทางด้านทักษะทางการเขียนโปรแกรมซึ่งจะอธิบายตามลำดับดังนี้

**4.1 การวิเคราะห์ทางด้านความถนัดในการเขียนโปรแกรม** จะเป็นการวิเคราะห์ถึงพฤติกรรมในการเขียนโปรแกรมในด้านต่าง ๆ อาทิ ผู้เขียนโปรแกรมได้มีการเขียนโปรแกรมตามที่ได้สอนไปหรือไม่ และได้ค่าประสิทธิผลเป็นอัตราส่วนร้อยละเท่าใด ผู้เขียนโปรแกรมมีความถนัดในการเลือกใช้เครื่องมือทางภาษาโปรแกรมเป็นอย่างไรบ้าง โปรแกรมที่ผู้เรียนเขียนขึ้นนั้นมีความถูกต้องสมบูรณ์หรือมีข้อผิดพลาดหรือไม่อย่างไร และรหัสต้นฉบับที่ถูกเขียนขึ้นนั้นอ่านง่ายหรือยากเพียงใด โดยที่ได้กล่าวมาทั้งหมดนี้จะสามารถบ่งบอกถึงพฤติกรรมการเขียนโปรแกรมของแต่ละบุคคลได้ รวมถึงยังสามารถบอกได้ว่าในอนาคตนั้นผู้เรียนการเขียนโปรแกรมจะมีแนวทางในการเขียนโปรแกรมอย่างไร โดยจะมีขั้นตอนในการวิเคราะห์ดังต่อไปนี้

**4.1.1 การเปรียบเทียบวัตถุประสงค์ทางด้านการเขียนโปรแกรม และคะแนนที่ได้** จากที่ผลที่ได้ทำการเปรียบเทียบระหว่างโปรแกรมที่ถูกเขียนขึ้นกับวัตถุประสงค์ของการสอนมาแล้ว ก็จะนำผลที่ได้มาวิเคราะห์ในภาพรวมว่าเมื่อครูผู้สอนได้มีการสอนการเขียนโปรแกรมแก่ผู้เรียนแล้ว ผู้เรียนได้มีการเขียนโปรแกรมตามที่สอนไปกี่ครั้ง โดยจะนับจำนวนครั้งที่ผู้เรียนได้ทำการเขียนโปรแกรมตามที่ผู้สอนสอนไปกับจำนวนครั้งที่ผู้เรียนไม่ได้เขียนโปรแกรมตามที่ผู้สอนได้สอนไป แล้วนำจำนวนครั้งมาคิดเป็นอัตราส่วนร้อยละเพื่อที่จะบอกว่า เมื่อท้ายที่สุดแล้วเมื่อผู้เรียนการเขียนโปรแกรมได้ทำการเรียนการเขียนโปรแกรมจนสิ้นสุดภาคการศึกษาแล้วนั้น ผู้เรียนได้นำสิ่งที่ครูผู้สอนได้ทำการสอนไปมาใช้ในการเขียนโปรแกรมมากน้อยขนาดไหนเมื่อเปรียบเทียบกับคะแนนที่ได้รับ และค่าประสิทธิผลที่ได้จากการนำคะแนนที่ได้มาจากการตรวจจากระบบตรวจอัตโนมัติมาคิดคำนวณกับคะแนนเต็มของแต่ละโจทย์ปัญหา โดยค่าประสิทธิผลจะถูกคิดออกมาเป็นอัตราส่วนร้อยละโดยค่าประสิทธิภาพส่วนนี้จะบอกว่าผู้เรียนการเขียนโปรแกรมทำการเขียนโปรแกรมออกมาแล้วมีความถูกต้องเพียงใด ยกตัวอย่างเช่น ผู้เรียนหมายเลขที่ 451 มีการเขียนโปรแกรมทั้งหมด 94 ครั้งจากจำนวนโจทย์ปัญหาทั้งหมด 48 ข้อ โดยมีการเขียนโปรแกรมตามวัตถุประสงค์การสอนทั้งสิ้น 94 ครั้งจาก 99 ครั้งโดยคิดเป็นร้อยละ 96

หรือ 96% จากการเขียนโปรแกรมทั้งหมด และไม่ได้เขียนโปรแกรมตามวัตถุประสงค์การสอนทั้งสิ้น 5 ครั้งจาก 99 ครั้งโดยคิดเป็นร้อยละ 4 หรือ 4% จากการเขียนโปรแกรมทั้งหมด และ ได้ค่าประสิทธิภาพ 0% เป็นจำนวน 52 ครั้งคิดเป็นร้อยละ 52.48 หรือประมาณ 52% จากจำนวนการเขียนโปรแกรม 99 ครั้ง ได้ค่าประสิทธิภาพ 10% จำนวน 1 ครั้งคิดเป็นร้อยละ 0.99 หรือประมาณ 1% จากการเขียนโปรแกรม 99 ครั้ง ได้ค่าประสิทธิภาพ 20% จำนวน 1 ครั้ง คิดเป็นร้อยละ 0.99 หรือประมาณ 1% จากการเขียนโปรแกรมจำนวน 99 ครั้ง ได้ค่าประสิทธิภาพ 80% จำนวน 1 ครั้ง คิดเป็นร้อยละ 0.99 หรือประมาณ 1% จากการเขียนโปรแกรม 99 ครั้ง และ ได้ค่าประสิทธิภาพ 100% จำนวน 44 ครั้งจากการเขียนโปรแกรม 99 ครั้งคิดเป็นร้อยละ 43.56 หรือประมาณ 44%

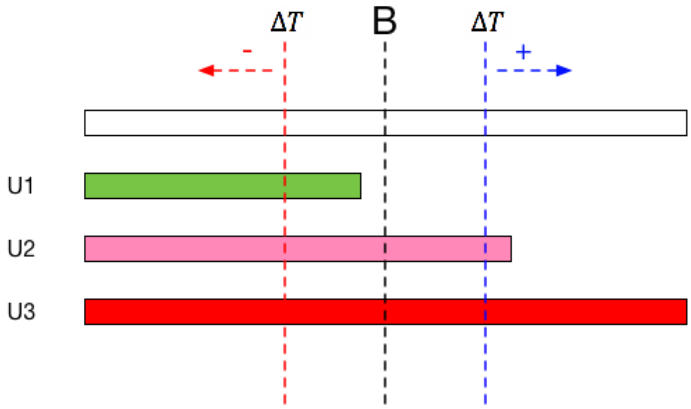
**4.1.2 ด้านความถนัดในการเลือกใช้เครื่องมือทางภาษาโปรแกรม** ในการวิเคราะห์ทางด้านความถนัดในการเลือกใช้เครื่องมือทางภาษาโปรแกรมนั้นจะพิจารณาด้วยกันอยู่ 3 ส่วนด้วยกันคือ 1. ลักษณะของการการประกาศตัวแปร 2. ลักษณะของการเลือกใช้การทำงานแบบเลือกทำ และ 3. ลักษณะของการเลือกใช้การทำงานแบบทำซ้ำ โดยในแต่ละส่วนจะใช้การนับจำนวนครั้งของการเลือกใช้ที่ได้ค่าประสิทธิผลเต็ม 100% แล้วนำมาคิดเป็นอัตราส่วนร้อยละ และนำลักษณะการเลือกใช้ที่มีอัตราส่วนมากที่สุดมาสรุปเป็นความถนัดในการเลือกใช้เครื่องทางภาษาโปรแกรมของผู้เขียน ยกตัวอย่างเช่น ผู้เรียนหมายเลข 451 ที่มีการเขียนโปรแกรมทั้งหมด 99 ครั้งได้มีการเลือกใช้การประกาศตัวแปรแบบ 1 ตัวต่อ 1 บรรทัด เลือกใช้การเลือกทำแบบใช้ “if” อย่างเดียว และ เลือกใช้การทำซ้ำประเภท “for” เป็นจำนวน 5 ครั้ง ที่ได้ค่าประสิทธิภาพเต็ม 100% เป็นจำนวน 5 ครั้งจากการเขียนโปรแกรมที่ได้ค่าประสิทธิภาพเต็ม 100% จำนวน 44 ครั้ง จากการเขียนโปรแกรมทั้งหมดจำนวน 99 ครั้ง คิดเป็นร้อยละ 2.2 หรือประมาณ 2% จากจำนวนการเขียนโปรแกรมที่ได้ค่าประสิทธิภาพเต็ม 100% จำนวน 44 ครั้ง หรือ คิดเป็นร้อยละ 4.95 หรือประมาณ 5% จากการเขียนโปรแกรมทั้งหมด 99 ครั้ง

**4.1.3 ด้านหมายเหตุจากระบบ** ในการวิเคราะห์ทางด้านการหมายเหตุจากการตรวจจากระบบนั้นจะสามารถบ่งบอกถึงข้อผิดพลาดของผู้เรียนในการเขียนโปรแกรมได้ในเบื้องต้น อาทิเช่น เขียนโปรแกรมแล้วไม่มีผลลัพธ์ออกมา หรือเขียนโปรแกรมแล้วผลลัพธ์ที่ได้ไม่ตรงกับเฉลยที่มีอยู่ในระบบ เป็นต้น ซึ่งหมายเหตุในส่วนนี้จะมีอยู่ด้วยกัน 4 ประเภทประกอบไปด้วยผ่านโดยไม่มีข้อผิดพลาด 1 ประเภท และมีข้อผิดพลาดอีก 3 ประเภท โดยในการวิเคราะห์จะใช้การนับจำนวนหมายเหตุทั้งผ่าน และไม่ผ่าน มาคิดเป็นอัตราส่วนร้อยละจากการเขียนโปรแกรมทั้งหมดของผู้เรียนการเขียนโปรแกรมที่สามารถบ่งบอกได้ถึงความผิดพลาดในเบื้องต้นของผู้เขียนว่าเกิดจากอะไร ยกตัวอย่างเช่น ผู้เรียนหมายเลข 451 ที่มีการเขียนโปรแกรมทั้งหมด 99 ครั้ง ผ่าน 44 ครั้งจากการเขียนโปรแกรมทั้งหมด

99 ครั้งคิดเป็นอัตราส่วนร้อยละ 43.56 หรือประมาณ 44% ไม่ผ่าน 55 ครั้งจากการเขียนโปรแกรมทั้งหมด 99 ครั้งคิดเป็นอัตราส่วนร้อยละ 56.44 หรือประมาณ 56% โดยที่ความผิดพลาดที่ทำให้ผู้เรียนหมายเลข 451 เขียนโปรแกรมไม่ผ่านนั้น เกิดมาจาก การที่เขียนโปรแกรมแล้วแปลโปรแกรม (compile) ไม่ผ่าน เป็นจำนวน 21 ครั้งจาก 55 ครั้ง เขียนโปรแกรมแล้วไม่มีผลลัพธ์ออกมาเป็นจำนวน 15 ครั้ง จาก 55 ครั้งและ เขียนโปรแกรมแล้วผลลัพธ์ที่ออกมาไม่ตรงกับเฉลยอีก 19 ครั้ง จาก 55 ครั้ง โดยจะเห็นว่าความผิดพลาดส่วนใหญ่ของผู้เรียนหมายเลข 451 นั้นเกิดมาจากการเขียนโปรแกรมที่ไม่สมบูรณ์ และผลลัพธ์ของโปรแกรมไม่ตรงกับเฉลย

**4.1.4 ด้านความยากง่ายทางการอ่านของรหัสต้นฉบับ** ในการวิเคราะห์ทางด้านความยากง่ายทางการอ่านของรหัสต้นฉบับจะเป็นการวิเคราะห์จากคะแนนที่ได้จากการใช้โปรแกรมที่มีชื่อว่า “Readability” ในการตรวจโดยผลที่ออกมาจะเป็นคะแนนที่อยู่ในช่วง 0 – 1 โดยยิ่งได้คะแนนที่ต่ำมากเท่าไหร่แสดงว่า รหัสต้นฉบับฉบับนั้นยากต่อการทำความเข้าใจ แต่ในทางกลับกันถ้ายิ่งได้คะแนนสูงมากเท่าไหร่แสดงว่ารหัสต้นฉบับนั้นง่ายต่อการทำความเข้าใจ ยกตัวอย่างเช่น ผู้เรียนหมายเลข 451 มีการเขียนโปรแกรมทั้งหมด 99 ครั้ง ได้ค่าความยากง่ายทางการอ่านของรหัสต้นฉบับเฉลี่ยอยู่ที่ 0.356 หรือคิดเป็น 35% จะสามารถอธิบายได้ว่ารหัสต้นฉบับที่ถูกเขียนขึ้นโดยผู้เขียนหมายเลข 451 นั้นอ่านยาก และยากต่อการทำความเข้าใจ

**4.2 การวิเคราะห์ทางด้านทักษะทางด้านการเขียนโปรแกรม** จะเป็นการวิเคราะห์ทักษะของการเขียนโปรแกรมจากการประเมินทักษะโดยใช้การเปรียบเทียบเวลาที่ใช้ไปในการเขียนโปรแกรมหนึ่งข้อปัญหา กับเวลาจำกัดที่ได้มาจากการหาค่าเฉลี่ยของกลุ่มนิสิตโอลิมปิคจำนวน 7 คน และจำนวนครั้งการส่งของแต่ละข้อโดยจากสมการที่ 3.1 นั้นจะสังเกตว่าจะมีตัวแปร $\Delta T$ โดยตัวแปรนี้จะเป็นตัวแปรที่ไว้ใช้สำหรับการปรับค่าทักษะเพื่อทำการชี้วัดค่าทักษะเพื่อประเมินว่าค่าทักษะนั้นจะออกมาเป็นอย่างไรโดยถ้าทำการบวก $\Delta T$ เข้าไปจะหมายถึงการขยายช่วงของค่าทักษะให้กว้างขึ้นส่งผลทำให้ค่าทักษะของผู้เขียนดีขึ้นเพราะในกรณีที่ค่าทักษะของผู้เขียนติดลบก็จะทำให้ค่าทักษะที่ติดลบอยู่นั้นสามารถกลายเป็นค่าบวกหรือค่าทักษะเพิ่มขึ้นได้ แต่ในทางกลับกันถ้าทำการลบ $\Delta T$ ออกจากค่าทักษะก็จะเป็นการลดช่วงของค่าทักษะลง ส่งผลให้ค่าทักษะของผู้เขียนนั้นถูกลดลงไป โดยตัวแปร $\Delta T$ นี้จะขึ้นอยู่กับผู้ประเมินว่าจะกำหนดค่าเป็นเท่าไหร่ แต่ในการวิจัยครั้งนี้จะกำหนดค่าไว้เป็นศูนย์เพื่อไม่ให้ค่าของทักษะที่ได้มาจากการคำนวณคลาดเคลื่อนไปจากความเป็นจริงโดยจะสามารถอธิบายได้ดังรูปที่ 4.1




รูปที่ 4.1 การปรับค่าทักษะโดยใช้ค่า ΔT

และในการวิเคราะห์ทางด้านทักษะของการเขียนโปรแกรมนั้นจะประกอบไปด้วยเงื่อนไข 5 ข้อได้แก่

1. ถ้าค่าทักษะมีค่าเป็นค่าบวกที่น้อยกว่า 1 และมีจำนวนครั้งของการส่งเพียง 1 ครั้งแล้ว ถือว่าผู้เขียนโปรแกรมมีทักษะการเขียนโปรแกรมอยู่ในช่วงดีมาก
2. ถ้าค่าทักษะมีค่าเท่ากับ 1 และมีจำนวนครั้งของการส่งเท่ากับ 1 ครั้งแล้ว ถือว่าผู้เขียนโปรแกรมมีทักษะการเขียนโปรแกรมอยู่ในช่วงดี
3. ถ้าค่าทักษะมีค่าเป็นค่าบวกที่มากกว่า 0 ไปจนถึง 1 และมีจำนวนครั้งของการส่งมากกว่า 1 ครั้งแล้ว ถือว่าผู้เขียนโปรแกรมมีทักษะการเขียนโปรแกรมอยู่ในช่วงปานกลาง
4. ถ้าค่าทักษะมีค่าเป็นค่าที่ติดลบที่ไม่เกิน -1 และมีจำนวนครั้งของการส่งมากกว่าหรือเท่ากับ 1 ครั้งแล้ว ถือว่าผู้เขียนโปรแกรมมีทักษะการเขียนโปรแกรมอยู่ในช่วงไม่ดี
5. ถ้าค่าทักษะมีค่าเป็นค่าติดลบที่เกิน -1 และมีจำนวนครั้งของการส่งมากกว่า 1 ครั้งแล้ว ถือว่าผู้เขียนโปรแกรมมีทักษะการเขียนโปรแกรมอยู่ในช่วงไม่ดี สมควรได้รับการแก้ไข

ภายหลังจากการที่ได้ค่าทักษะจากการคำนวณมาแล้วก็จะนำค่าทักษะดังกล่าวมาวิเคราะห์โดยอ้างอิงตามเงื่อนไขทั้ง 5 ข้อเพื่อที่จะสามารถบอกได้ถึงทักษะของผู้เรียนการเขียนโปรแกรมแต่ละคน ยกตัวอย่างเช่น ผู้เรียนหมายเลข 451 มีการเขียนโปรแกรมทั้งหมด 50 โปรแกรม ได้ค่าทักษะดีมาก 13 ข้อจาก 50 ข้อ คิดเป็นอัตราส่วนร้อยละ 0.26 หรือ 26% จากการเขียนโปรแกรมทั้งหมด 50 ข้อ ได้ค่าทักษะ ดี 9 ครั้ง จากการเขียนโปรแกรม 50 ข้อจาก 50 ข้อ คิดเป็นอัตราส่วนร้อยละ 0.18 หรือ 18% จากการเขียนโปรแกรมทั้งหมด 50 ข้อ ได้ค่าทักษะปานกลาง 2 ข้อ จากการเขียนโปรแกรม 50 ข้อ คิดเป็นอัตราส่วนร้อยละ 0.04 หรือ 4% จากการเขียนโปรแกรมทั้งหมด 50 ข้อ ได้ค่าทักษะไม่ดี จำนวน 7 ข้อ จากการเขียนโปรแกรมทั้งหมด 50 ข้อ คิดเป็นอัตราส่วนร้อยละ 0.14 หรือ 14% จาก

การเขียนโปรแกรมทั้งหมด 50 ข้อ และได้รับค่าทักษะไม่ดีจำนวน 19 ข้อ จากการเขียนโปรแกรมทั้งหมด 50 ข้อ คิดเป็นอัตราส่วนร้อยละ 0.38 หรือ 38% จากการเขียนโปรแกรมทั้งหมด 50 ข้อ จากข้อมูลที่ได้มาแสดงให้เห็นว่าผู้เขียนหมายเลข 451 นั้นมีทักษะการเขียนโปรแกรมอยู่ในระดับปานกลาง เนื่องจากอัตราส่วนของค่าทักษะที่ดีกับ อัตราส่วนของค่าทักษะที่ควรปรับปรุงนี้ค่อนข้างที่จะใกล้เคียงกันมาก ถึงแม้ว่าจะนำเอาจำนวนค่าทักษะของทั้งกลุ่มดีมาก และดีมาคิดรวมกัน ก็จะออกมาใกล้เคียงกับการที่เอาจำนวนค่าทักษะของกลุ่มไม่ดี และควรปรับปรุงมาคิดรวมกันโดยจะออกมาเป็น 22 ต่อ 26 ซึ่งจะค่อนข้างใกล้เคียงกัน จึงสรุปว่าผู้เขียนหมายเลข 451 นั้นมีทักษะการเขียนโปรแกรมอยู่ในระดับปานกลาง



จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

# บทที่ 5
# สรุปผลการวิจัยและวิจารณ์งานวิจัย

## 5.1 สรุปผลการวิจัย

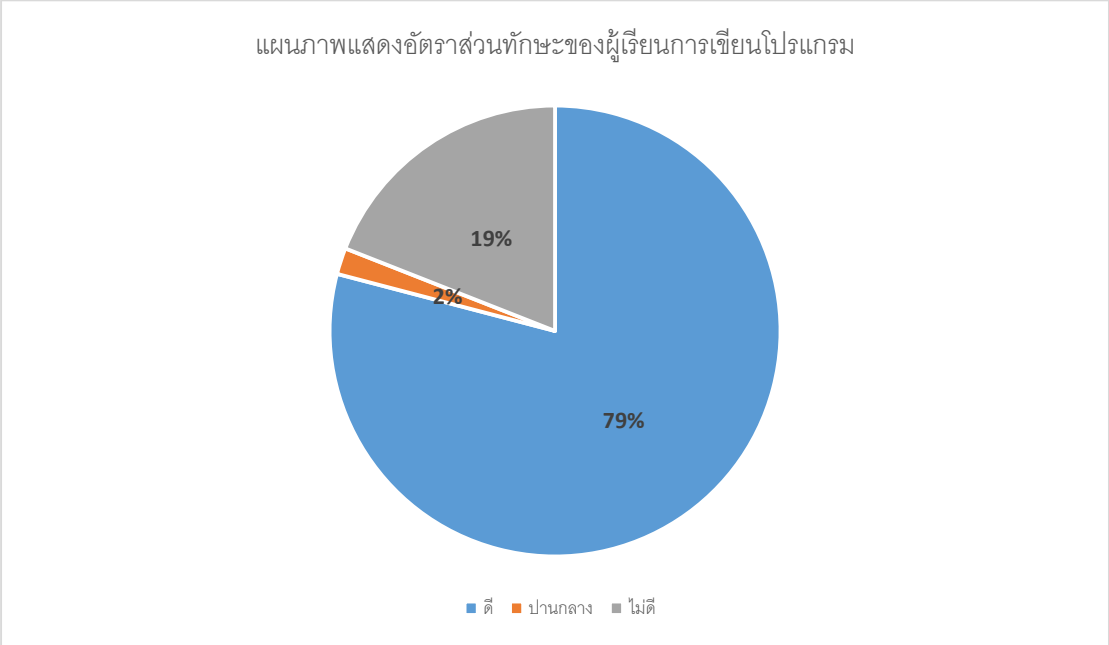
ในการค้นหาความถนัดทางด้านการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรมแต่ละคนนั้นจะสามารถแสดงให้เห็นถึงความถนัดในการเลือกใช้เครื่องมือทางภาษาการเขียนโปรแกรมของแต่ละคน รวมถึงสามารถบ่งบอกถึงแนวทางการเลือกใช้เครื่องมือทางภาษาการเขียนโปรแกรมในการเขียนโปรแกรมที่จะเกิดขึ้นในอนาคต และยังสามารถบอกถึงรหัสต้นฉบับของโปรแกรมที่ผู้เขียนเขียนออกมานั้นอ่านได้ยาก หรือง่ายอย่างไรโดยคะแนนจากการตรวจความยากง่ายทางการอ่านจะเป็นตัวบอก โดยในส่วนนี้จะมีความสำคัญมากในกรณีที่มีผู้เขียนคนอื่นนำรหัสต้นฉบับไปใช้ต่อ ส่วนในทางด้านการประเมินทักษะทางด้านการเขียนโปรแกรมพบว่า มีผู้เรียนจำนวน 57% นิยมใช้การประกาศตัวแปรแบบ 1 ตัวต่อ 1 บรรทัด และอีก 40% เป็นผู้เขียนที่นิยมใช้การประกาศตัวแปรแบบผสมระหว่าง 1 ตัวต่อ 1 บรรทัด และหลายตัวต่อ 1 บรรทัด ในด้านของการใช้การเลือกทำ พบว่ามีผู้เรียนมากถึง 59% ในการเลือกใช้การเลือกทำแบบใช้คำสั่ง “if” เพียงอย่างเดียวในโปรแกรม 22% เป็นการเลือกใช้คำสั่ง “if” และตามด้วย “else” 9% เป็นการเลือกใช้คำสั่ง “if” และตามด้วย “else-if” และอีก 10% เป็นการเลือกใช้คำสั่ง “if” ตามด้วย “else-if” และจบด้วย “else” ทั้งหมดภายในโปรแกรม ทางด้านการเลือกใช้การทำซ้ำพบว่ามีผู้เรียนจำนวนมากถึง 49% เลือกใช้การทำซ้ำแบบ “for” 32% เป็นผู้เขียนที่เลือกใช้การทำซ้ำแบบ “while” และอีก 19% เป็นผู้ใช้ที่เลือกใช้การทำซ้ำแบบผสมระหว่าง “for” และ “while” ผสมกันภายในหนึ่งโปรแกรมดังตารางที่ 5.1

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ตารางที่ 5.1 อัตราส่วนของลักษณะการประกาศตัวแปรของผู้เขียนทั้งหมด

| Language tools | | | | | |
|---|---|---|---|---|---|
| Declaration | | Selection | | Repetition | |
| TYPE | PERCENTAGE | TYPE | PERCENTAGE | TYPE | PERCENTAGE |
| SINGLE | 57% | IF | 59% | FOR | 49% |
| MULTIPLE | 3% | IF-ELSE | 22% | WHILE | 32% |
| MIX | 40% | IF-ELSE_IF | 9% | MIX | 19% |
| | | MIX | 10% | | |

ในด้านการเปรียบเทียบวัตถุประสงค์ของการสอนนั้นพบว่า ผู้เรียนทั้งหมด 100% ได้ทำการเขียนโปรแกรมตามวัตถุประสงค์ของการสอนทั้งหมด ทางด้านความยากง่ายทางการอ่าน พบว่าผู้เรียนส่วนใหญ่เขียนโปรแกรมออกมาได้ค่อนข้างที่จะอ่านยากซึ่งมีจำนวนมากถึง 836 คน หรือคิดเป็นอัตราส่วนร้อยละ 99% ของผู้เรียนทั้งหมด 848 คน และอีก 128 คน หรือคิดเป็น 1% เป็นผู้เรียนที่เขียนโปรแกรมออกมาได้ค่อนข้างที่จะอ่านง่ายไปจนถึงอ่านง่าย ทางด้านการประเมินทักษะในการเขียนโปรแกรมพบว่า มีผู้เรียนจำนวน 79% ของผู้เขียนทั้งหมด มีทักษะทางด้านการเขียนโปรแกรมที่ดี 19% เป็นผู้เรียนที่มีทักษะทางด้านการเขียนโปรแกรมที่ไม่ดี และที่เหลืออีก 2% เป็นผู้เรียนที่มีทักษะทางด้านการเขียนโปรแกรมปานกลาง ดังรูปที่ 5.1 และตารางที่ 5.2 ตามลำดับ




รูปที่ 5.1 อัตราส่วนแสดงจำนวนผู้เรียนที่มีทักษะดี ไม่ดี และปานกลาง

ตารางที่ 5.2 ตารางแสดงกลุ่มของทักษะของผู้เรียนการเขียนโปรแกรม

| | ระดับของทักษะ | | | | |
|---|---|---|---|---|---|
| | **ดีมาก** | **ดี** | **ปานกลาง** | **ไม่ดี** | **ควรปรับปรุง** |
| จำนวนของผู้เรียน (คน) | 0 | 669 | 16 | 161 | 0 |

## 5.2 วิจารณ์งานวิจัย

งานวิจัยชิ้นนี้ได้นำเสนอแนวคิด และวิธีการในการค้นหาความถนัด และการประเมินทักษะทางด้านการเขียนโปรแกรม จากข้อมูลจากการเรียนการสอนวิชา Computer Programming I ของนิสิตชั้นปีที่ 1 คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย โดยอาศัยคุณลักษณะต่าง ๆ อาทิเช่น การเลือกใช้เครื่องมือทางภาษาเขียนโปรแกรม ความยากง่ายทางการอ่านของรหัสต้นฉบับ เวลาที่ใช้ในการเขียนโปรแกรมหนึ่งข้อ เป็นต้น โดยคุณลักษณะที่ใช้บางคุณลักษณะอาจทำให้เกิดความลำเอียงในผลลัพธ์ที่ออกมา ยกตัวอย่างเช่น ความยากง่ายทางการอ่านของรหัสต้นฉบับที่ถูกพัฒนาขึ้นโดย บรูส และไวเมอร์ [15] โดยการประเมินความยากง่ายทางการอ่านนั้นเป็นเรื่องที่ต่างคนต่างมองไม่เหมือนกันบางคนมองว่าการเขียนรหัสต้นฉบับแบบนี้อ่านง่าย แต่ในทางกลับกัน อีกคนกลับบอกว่าการเขียนแบบนี้อ่านยากควรที่จะเขียนอีกแบบ ซึ่งในจุดนี้จะขึ้นอยู่กับมุมมองของแต่ละบุคคลซึ่งไม่มีมาตรฐาน หรือกฎเกณฑ์มารองรับว่าการเขียนแบบไหนที่เรียกว่า อ่านง่าย หรืออ่านยาก ถ้ามีนักวิจัยที่สามารถคิดค้นมาตรฐาน หรือกฎเกณฑ์ดังกล่าวขึ้นมาได้จะสามารถทำให้ความลำเอียงดังกล่าวลดลงไปได้เป็นอย่างมาก อีกทั้งจำนวนของคุณลักษณะที่มีผลต่อการวิเคราะห์นั้นอาจมีได้มากว่านี้เพื่อที่จะทำให้ผลจากการวิเคราะห์จะได้ชัดเจนมากขึ้น อาทิเช่น การกรอกแบบสอบถาม (questionnaires) ก่อนเรียนว่าได้มีการเตรียมตัวก่อนเรียน หรือรู้เรื่องที่จะเรียนมากน้อยแค่ไหนบ้าง เพื่อที่จะได้เป็นข้อมูลประกอบในการวิเคราะห์ทักษะได้อย่างชัดเจนมากขึ้น และจำเป็นที่จะต้องมีแบบทดสอบก่อนที่จะเริ่มเรียนการเขียนโปรแกรม (pre-test) เพื่อที่จะได้รู้ว่าพื้นฐานการเขียนโปรแกรมของผู้เรียนการเขียนโปรแกรมเป็นอย่างไรบ้าง เนื่องจากในการเรียนการเขียนโปรแกรมนั้นจะมีความหลากหลายทางด้านทักษะ และความรู้ในการเขียนโปรแกรมอยู่มาก ผู้เขียนบางคนเคยเขียนโปรแกรมมาเป็นระเวลานาน ต่างจากผู้เขียนบางคนที่ไม่เคยเขียนโปรแกรมมาเลยซึ่งข้อมูลในส่วนนี้จะมาช่วยในการประเมินทักษะ และยังบ่งบอกความก้าวหน้า หรือประสิทธิผลที่ได้จากการเรียนเขียนโปรแกรมได้อีกด้วยซึ่งในเรื่องของแบบทดสอบก่อนเรียน และแบบสอบถามหลังเรียนเป็นเรื่องที่น่าสนใจ และควรทำเป็นอย่างยิ่ง เพราะจะสามารถบ่งบอกทักษะและพัฒนาการทางด้านการเรียนได้เป็นอย่างดี

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

### 5.3 ข้อเสนอแนะสำหรับงานวิจัยในอนาคต

1) ควรที่จะมีการจัดการทดสอบ ก่อนเริ่มการเรียนการสอนการเขียนโปรแกรมในครั้งแรกของการเรียนการสอน เพื่อเป็นข้อมูลให้แก่ครูผู้สอนการเขียนโปรแกรมได้รับทราบถึงทักษะในการเขียนโปรแกรมของผู้เรียนทุกคน
2) ควรที่จะมีแบบสอบถามรวมถึงแบบฝึกหัด ก่อนที่จะเริ่มการเรียนการสอนการเขียนโปรแกรมในแต่ละครั้ง เพื่อเป็นข้อมูลให้กับครูผู้สอนได้รับทราบถึงการเตรียมความพร้อมของผู้เรียนการเขียนโปรแกรมว่า ผู้เรียนได้มีการทบทวนในสิ่งที่ครูผู้สอนได้สอนไปในการเรียนการสอนการเขียนโปรแกรมครั้งที่ผ่านมาหรือไม่
3) ไม่ควรที่จะมีการใช้แผ่นแบบในการเรียนการสอนการเขียนโปรแกรม เนื่องจากการใช้แผ่นแบบไม่ได้ช่วยพัฒนาทักษะทางด้านการเขียนโปรแกรมแต่อย่างใด

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

**รายการอ้างอิง**


1. Hamblen, A.P.a.J., *Computer Algorithm for Plagiarism Detection*, in *IEEE Transections on Education*. May 1989. p. 94 - 99.
2. Luck, M.J.a.M., *Plagiarism in Programming Assignments*, in *IEEE Transections on Education*. 1999. p. 129 - 133.
3. S. Yousuf, M.A.a.S.N., *A Review of Plagiarism Detection Based on Lexical and Semantic Approach*, in *Emerging Trends in Communication, Control, Signal Process & Computing Application (C2SPCA)*. 2013: Bangalore, India.
4. L. Kong, Z.L., H. Qi and Z. Han, *High Obfuscation Plagiarism Detection Using Multi Feature Fusion Based on Logical Regression Model*, in *The 4th international Conference on Computer Scienced and Network Technology (ICCSNT)*. 2015: Harbin, Chaina.
5. O. Ajmal, M.M.S.M., T. Hashmat, M. Moosa and T. Ali, *EPlag: A two-layer source code plagiarism detection system*, in *The 8th International Conference on Digital Information Management (ICDIM)*. 2013: Islamabad Pakistan. p. 256 - 261.
6. W. Chunhui, L.Z.a.L.D., *Preventing and detecting plagiarism in programming course*, in *International Journal of Security and Its Applications*. 2013. p. 269 - 278.
7. S. Schleimer, D.W.a.A.A., *Winnowing: Local Algorithms for Document Fingerprinting*, in *ACM SIGMOD International Conference on Management of Data*. 2003: San Diego CA. p. 76 - 85.
8. L. Prechelt, G.M.a.M.P., *JPlag: Finding Plagiarism among a set of programs.* Journal of universal Computer Science, 2002: p. 1016 - 1038.
9. J. Zhao, K.X., Y. Fu, Baojiang Cui, *An AST-Based Code Plagiarism Detection Algorithm*, in *10th International Conference on Broadband and wireless Computing, Communication and Application*. 2015.
10. Augsten, M.P.a.N., *Efficient Computation of the Tree Edit Distance*, in *ACM Transactions on Database System*. 2015.


จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

11. R. Roy Choudhury, H.Y., J. Moghadam, A. Fox, *AutoStyle: Toward Coding Style Feedback at Scale*, in *CSCW'16 Companion*. 2016: San Francisco, USA.
12. Kusumoto, Y.H.a.S., *Code Clone Detection Non-Specialized PDGs with Heuristic*, in *15th European Conference on Software Maintenance and Reengineering*. 2011.
13. S. Arabyarmohamady, H.M., M. Asadpour, *A Coding Style- based Plagiarism Detection*, in *2012 International Conference on Interactive Mobile and Computer Aided Learning (IMCL)*. 2012: Amman, Jordan.
14. Murad, N.A.a.M.A.A., *A Hybrid Method of Feature Extraction and Nai¨ve Bayes Classification for Splitting Identifiers.* Journal of Theoretical and Applied Information Technology, 2017. **96**(7): p. 1549 - 1557.
15. Weimer, R.B.a.W., *Learning a Matric for Code Readability*, in *IEEE Transections on Software Engineering*. 2010. p. 546 - 558.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

# ภาคผนวก

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

# ประวัติผู้เขียนวิทยานิพนธ์

นายชัยวัฒน์ ฉวีวรรณ

เกิดเมื่อวันที่ 4 ธันวาคม พ.ศ. 2533

จบการศึกษาระดับประถมศึกษาจาก โรงเรียน อำนวยวิทย์ จังหวัด จันทบุรี

จบการศึกษาระดับมัธยมศึกษาตอนต้นจาก โรงเรียน ท่าใหม่พูลสวัสดิ์ ราษฎร์นุกูล จังหวัด จันทบุรี

จบการศึกษาระดับมัธยมศึกษาตอนปลายจาก โรงเรียน อัสสัมชัญศรีราชา จังหวัด ชลบุรี

จบการศึกษาระดับปริญญาตรีจาก มหาวิทยาลัย เทคโนโลยีราชมงคลกรุงเทพ จังหวัด กรุงเทพมหานคร

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY