โครงการ

โปรแกรมแสดงการเปรียบเทียบวิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก

ชื่อโครงการ	โปรแกรมแสดงการเปรียบเทียบวิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก
	Comparison of Bubble Sort and Insertion Sort Program
ชื่อนิสิต	นายชัยฤทธิ์ ขาวผ่อง 553 35604 23
ภาควิชา	คณิตศาสตร์และวิทยาการคอมพิวเตอร์
	สาขาวิชา คณิตศาสตร์
ปีการศึกษา	2561

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

โปรแกรมแสดงการเปรียบเทียบวิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก

นายชัยฤทธิ์ ขาวผ่อง

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต
สาขาวิชาคณิตศาสตร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2561

COMPARISON OF BUBBLE SORT AND INSERTION SORT PROGRAM

Mr. Chaiyarit Karwpong

A Project Submitted in Partial Fulfillment of the Requirements

for the Degree of Bachelor of Science Program in Mathematics

Department of Mathematics and Computer Science

Faculty of Science

Chulalongkorn University

Academic Year 2018

หัวข้อโครงงาน	โปรแกรมแสดงการเปรียบเทียบวิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก
โดย	นายชัยฤทธิ์ ขาวผ่อง
สาขาวิชา	คณิตศาสตร์
อาจารย์ที่ปรึกษาโครงงานหลัก	รองศาสตราจารย์ จิตรจวบ เปาอินทร์
อาจารย์ที่ปรึกษาโครงงานร่วม	ผู้ช่วยศาสตราจารย์ วาสนา สุขกระสานติ

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับโครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาบัณฑิต ในรายวิชา 2301499 โครงงานวิทยาศาสตร์ (Senior Project)

.....
(ศาสตราจารย์ ดร.กฤษณะ เนียมมณี) หัวหน้าภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

คณะกรรมการสอบโครงงาน

.....
(รองศาสตราจารย์ จิตรจวบ เปาอินทร์) อาจารย์ที่ปรึกษาโครงงานหลัก

.....
(ผู้ช่วยศาสตราจารย์ วาสนา สุขกระสานติ) อาจารย์ที่ปรึกษาโครงงานร่วม

.....
(ผู้ช่วยศาสตราจารย์ ศศิภา พันธุวดีธร) กรรมการ

.....
(ผู้ช่วยศาสตราจารย์ ดร.รตินันท์ บุญเคลือบ) กรรมการ

นายชัยฤทธิ์ ขาวผ่อง: โปรแกรมแสดงการเปรียบเทียบวิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก. (Comparison of Bubble Sort and Insertion Sort Program) อ.ที่ปรึกษาโครงงานหลัก : รศ.จิตรจวบ เปาอินทร์, อ.ที่ปรึกษาโครงงานร่วม : ผศ.วาสนา สุขกระสานติ, 94 หน้า.

โครงงานนี้มีวัตถุประสงค์เพื่อวิเคราะห์และนำเสนอวิธีการจัดเรียงลำดับข้อมูลแบบฟอง และการจัดเรียงลำดับข้อมูลแบบแทรก พร้อมทั้งเปรียบเทียบประสิทธิภาพของการเรียงลำดับทั้งสองวิธี โดยนำเสนอวิธีการจัดเรียงลำดับข้อมูลแบบฟองและวิธีการจัดเรียงลำดับข้อมูลแบบแทรก ผ่านโปรแกรมคอมพิวเตอร์โดยใช้ภาษาซี

ภาควิชา คณิตศาสตร์และวิทยาการคอมพิวเตอร์ ลายมือชื่อนิสิต ชัยฤทธิ์ ขาวผ่อง

สาขาวิชา คณิตศาสตร์ ลายมือชื่อ อ.ที่ปรึกษาโครงงานหลัก

ปีการศึกษา 2561 ลายมือชื่อ อ.ที่ปรึกษาโครงงานร่วม

5533560423 : MAJOR MATHEMATICS

KEYWORDS : DATA SORTING, PROGRAMMING ALGORITHM

CHAIYARIT KARWPONG : COMPARISON OF BUBBLE SORT AND INSERTION SORT PROGRAM SORT. ADVISOR : ASSOC. PROF. CHITCHUAB PAOIN, CO-ADVISOR : ASST. PROF. VASANA SUKKRASANTI, 94 pp.

The objectives of this research are to analyze and present bubble sort and insertion sort. We display comparison of bubble sort and insertion sort. We also present the bubble sort and insertion sort via computer program using C language.

Department : Mathematics and Computer Science Student's Signature Chaiyarit Karwpong

Field of Study : Mathematics Advisor's Signature Chitchuab Paoin

Academic Year : 2018 Co-advisor's Signature Vasana Sukkrasanti

กิตติกรรมประกาศ

โครงงานนี้เสร็จสมบูรณ์ได้โดยได้รับความกรุณาเป็นอย่างยิ่งจากรองศาสตราจารย์ จิตรจวบ เปาอินทร์ และผู้ช่วยศาสตราจารย์ วาสนา สุขกระสานติ ที่ได้สละเวลาอันมีค่าให้คำปรึกษาในการทำโครงงาน พร้อมทั้งอบรมสั่งสอนและให้ความรู้เป็นอย่างดียิ่ง อีกทั้งแนะนำ ตรวจ และแก้ไขข้อบกพร่องต่าง ๆ อย่างดีตลอดมา จนโครงงานนี้สำเร็จสมบูรณ์ นิสิตผู้จัดทำโครงงานจึงขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณบิดาและมารดา ที่ให้คำปรึกษาและการสนับสนุนในการทำโครงงานเสมอมา

ขอขอบพระคุณอาจารย์ทุกท่านที่ได้ให้ความกรุณาและให้ความรู้

สุดท้ายนี้ผู้จัดทำโครงงานหวังเป็นอย่างยิ่งว่า โครงงานโปรแกรมแสดงการเปรียบเทียบ วิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก จะเป็นประโยชน์ต่อการศึกษาและค้นคว้าสำหรับผู้ที่สนใจ

นายชัยฤทธิ์ ขาวผ่อง

ผู้จัดทำโครงงาน

สารบัญ

หน้า

หน้า

สารบัญภาพ

หน้า

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและเหตุผลการวิจัย

ปัจจุบันนี้เป็นยุคของข้อมูลข่าวสาร ซึ่งแน่นอนว่าข้อมูลมีจำนวนเพิ่มมากขึ้นเรื่อย ๆ การมีวิธีการจัดการข้อมูลอย่างเป็นระบบ จะทำให้สามารถทำงานได้อย่างมีประสิทธิภาพ การเรียงลำดับข้อมูล เป็นวิธีการที่มีบทบาทสำคัญในการจัดการข้อมูลอย่างยิ่ง นักคณิตศาสตร์ และนักเทคโนโลยีสารสนเทศ คิดค้นวิธีการเรียงลำดับมากมายหลายวิธี แต่ละวิธีต่างมีข้อดีข้อเสียแตกต่างกัน

ผู้ดำเนินงานจึงสนใจที่จะวิเคราะห์การเรียงลำดับข้อมูล โดยเลือกวิธีการเรียงลำดับมา 2 วิธี แล้วอธิบายขั้นตอนการเรียงลำดับข้อมูลของแต่ละวิธีที่ได้เลือกมานั้น ผ่านจอภาพที่มีรูปภาพสีสันให้ผู้สนใจสามารถเข้าใจได้โดยง่าย พร้อมทั้งแสดงการเปรียบเทียบข้อดีและข้อด้อยของทั้งสองวิธีนั้น

1.2 วัตถุประสงค์ของการวิจัย

โครงงานนี้มีวัตถุประสงค์เพื่อวิเคราะห์และนำเสนอวิธีการจัดเรียงลำดับข้อมูลสองวิธี คือ วิธีเรียงลำดับข้อมูลแบบฟอง (Bubble sort) และวิธีเรียงลำดับข้อมูลแบบแทรก (Insertion sort) พร้อมทั้งเปรียบเทียบและวิเคราะห์ข้อดีและข้อด้อยของทั้งสองวิธีนี้

1.3 ขอบเขตการวิจัย

1. โครงงานนี้จะแสดงวิธีการจัดเรียงข้อมูลสองวิธี คือ วิธีเรียงลำดับข้อมูลแบบฟอง และวิธีเรียงลำดับข้อมูลแบบแทรก โดยนำเสนอในลักษณะของภาพเคลื่อนไหว เป็นขั้นตอน โดยใช้ข้อมูลไม่น้อยกว่า 20 ตัว
2. โครงงานนี้พัฒนาการเรียงลำดับข้อมูลโดยใช้ภาษาซี และใช้งานได้บนระบบปฏิบัติการไมโครซอฟท์วินโดว์

1.4 ขั้นตอนการวิจัย

1. ศึกษาวิธีการจัดเรียงลำดับข้อมูลในแบบต่าง ๆ
2. ศึกษาความรู้พื้นฐานเกี่ยวกับภาษาซี
3. ศึกษาวิธีการจัดเรียงลำดับข้อมูล
4. เขียนโปรแกรมแสดงการจัดเรียงลำดับข้อมูลแบบฟองและแบบแทรก
5. วิเคราะห์และทดสอบโปรแกรมการจัดเรียงลำดับข้อมูล
6. วิเคราะห์ข้อดีและข้อด้อยของการจัดเรียงข้อมูลแบบฟองและแบบแทรก
7. สรุปผลและเขียนรายงาน

1.5 ประโยชน์ที่คาดว่าจะได้รับ

สำหรับผู้ดำเนินงาน

1. ได้ความรู้พื้นฐานเกี่ยวกับการจัดเรียงลำดับข้อมูล และการเขียนโปรแกรมด้วยภาษาซี
2. ได้ศึกษาและทำความเข้าใจเกี่ยวกับการเรียงลำดับแบบฟอง และการเรียงลำดับแบบแทรก
3. ได้โปรแกรมแสดงขั้นตอนและวิธีการจัดเรียงลำดับข้อมูลแบบฟองและแบบแทรก

สำหรับผู้ใช้งาน

1. ได้ความรู้เกี่ยวกับการเรียงลำดับแบบฟอง และการเรียงลำดับแบบแทรก
2. ได้ทราบข้อดีและข้อด้อยของการจัดเรียงข้อมูลแบบฟองและแบบแทรก

1.6 โครงสร้างของรายงาน

บทที่ 2 จะกล่าวถึงหลักการเรียงลำดับข้อมูลแบบฟองและแบบแทรก ความรู้พื้นฐานที่เกี่ยวข้องกับโปรแกรมภาษาซีในส่วนที่ใช้ในโครงงาน คือ ส่วนอัลกอริทึมของการทำงานแบบวนซ้ำ และฟังก์ชันการสุ่มตัวเลขของภาษาซี และความรู้เกี่ยวกับการเปรียบเทียบวิธีการจัดเรียงข้อมูล

บทที่ 3 จะกล่าวถึงขั้นตอนการทำโครงงาน ซึ่งจะประกอบไปด้วย การออกแบบ ขั้นตอนการจัดทำโครงงาน อัลกอริทึมที่ใช้ในการเรียงลำดับแบบฟองและแบบแทรก พร้อมทั้งการเปรียบเทียบผลที่ได้

บทที่ 4 จะกล่าวถึงผลการดำเนินโครงงาน คือ โปรแกรมที่แสดงถึงขั้นตอนและวิธีการจัดเรียงลำดับข้อมูลแบบฟองและแบบแทรก

บทที่ 5 จะกล่าวถึงบทสรุป และข้อเสนอแนะของการจัดทำโครงงานนี้

บทที่ 2
งานที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงหลักการเรียงลำดับข้อมูลแบบฟองและแบบแทรก ความรู้พื้นฐานที่เกี่ยวข้องกับโปรแกรมภาษาซีในส่วนที่ใช้ในโครงงาน คือ ส่วนอัลกอริทึมของการทำงานแบบวนซ้ำ และฟังก์ชันการสุ่มตัวเลขของภาษาซี และความรู้เกี่ยวกับการเปรียบเทียบวิธีการจัดเรียงข้อมูล

2.1 การเรียงลำดับข้อมูล

การเรียงลำดับข้อมูล คือการจัดเรียงข้อมูลให้อยู่ในลำดับที่ต้องการ ซึ่งอาจจะเป็นการเรียงลำดับข้อมูลจากน้อยไปหามาก หรือจากมากไปหาน้อยก็ได้ เพื่อให้มีการเข้าถึงข้อมูลได้อย่างรวดเร็วและมีประสิทธิภาพ การเรียงลำดับข้อมูลมักจะถูกนำมาใช้เมื่อข้อมูลมีจำนวนมากๆ ข้อมูลที่เรียงลำดับแล้วจะทำให้การค้นหาข้อมูลนั้นเป็นไปได้อย่างรวดเร็ว

ปัจจุบันนี้มีวิธีการมากมายที่ใช้ในการจัดเรียงข้อมูล ซึ่งผู้ใช้สามารถเลือกวิธีการจัดเรียงข้อมูลเอง โดยคำนึงถึงปริมาณของข้อมูล เวลาที่ใช้ ลักษณะของข้อมูล เวลาและแรงงานที่ต้องใช้ในการเขียนโปรแกรม

2.1.1 วิธีการจัดเรียงลำดับข้อมูลแบบฟอง (Bubble Sort)

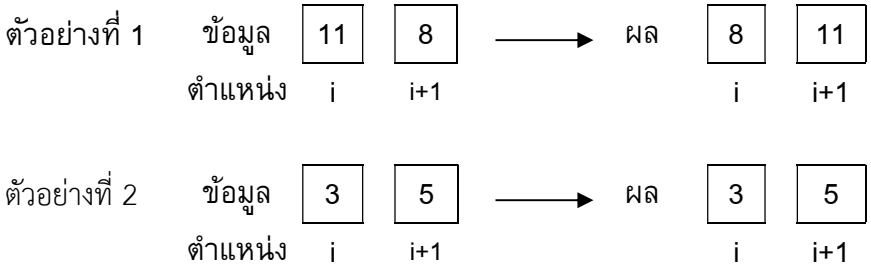
การเรียงลำดับข้อมูลแบบฟอง จะทำโดยการเปรียบเทียบค่าข้อมูลสองค่าที่อยู่ติดกัน แล้วพิจารณาว่าข้อมูลคู่นั้นอยู่ในลำดับที่ต้องการหรือไม่ ถ้าไม่อยู่ในลำดับที่ต้องการให้สลับค่าในตำแหน่งของข้อมูลทั้งสองให้อยู่ในลำดับที่ต้องการและทำเช่นนี้ไปเรื่อย ๆ จนครบจำนวนข้อมูล ดังกรณีตัวอย่างต่อไปนี้

กรณีที่ 1 การเรียงลำดับข้อมูลจากน้อยไปหามาก

พิจารณาข้อมูลใน 2 ตำแหน่งที่อยู่ติดกัน คือ ตำแหน่งที่ i และตำแหน่งที่ i+1 โดยที่ i มีค่าตั้งแต่ 1 ถึง n-1 ดังรูป

เรียงลำดับโดยเปรียบเทียบข้อมูลสองตัวที่อยู่ในตำแหน่งติดกัน โดยพิจารณา ตั้งแต่ i = 1 และขยับ i เพิ่มขึ้นทีละ 1 ไปเรื่อย ๆ เมื่อเปรียบเทียบจนครบข้อมูล คือ i = n-1 ก็จะเปรียบเทียบข้อมูลตัวที่ n-1 กับ n ทำให้ได้ข้อมูลที่เรียงลำดับครบถ้วน

ถ้าข้อมูลตำแหน่งที่ i มีค่ามากกว่าข้อมูลในตำแหน่งที่ i+1 ต้องสลับค่าข้อมูลในตำแหน่งคู่นี้ ดังตัวอย่างที่ 1 ในรูปที่ 2.1 แต่ถ้าข้อมูลในตำแหน่งที่ i น้อยกว่าตัวที่ i+1 ก็ไม่ต้องสลับค่าข้อมูล ดังตัวอย่างที่ 2 ในรูปที่ 2.1



รูปที่ 2.1 การเปรียบเทียบข้อมูลในตำแหน่งที่ i และ i+1 เพื่อเรียงลำดับข้อมูลจากน้อยไปหามาก

ในกรณีเรียงลำดับข้อมูลจากน้อยไปมาก ถ้าค่าที่ i มีค่ามากกว่าค่าที่ i+1 ก็จะทำการสลับที่กัน โดยวิธีการนี้จะทำให้ข้อมูลที่มีค่ามากลอยสูงขึ้นไปทางด้านท้ายของแถวเหมือนฟองสบู่ (bubble) ที่ลอยสูงขึ้นและข้อมูลที่มีค่าน้อยที่สุดก็จะเลื่อนลงมาอยู่ในตำแหน่งแรกของชุดข้อมูล

ตัวอย่าง 2.1 กำหนดข้อมูลตัวเลขจำนวน 5 ตัว คือ 7 3 6 4 2 จงเรียงลำดับข้อมูลจากน้อยไปหามาก วิธีทำในแต่ละขั้นตอนค่าตัวเลขในคอลัมน์ข้อมูลและคอลัมน์ผลลัพธ์ที่เน้นตัวหนา คือ ค่าที่กำลังพิจารณาเปรียบเทียบกัน ค่าตัวเลขในคอลัมน์ข้อมูล และคอลัมน์ผลลัพธ์ที่ขีดเส้นใต้และตัวเอียง คือ ค่าที่อยู่ในตำแหน่งที่เหมาะสมแล้วจึงไม่ต้องนำไปเปรียบเทียบอีก

รอบที่ 1

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**7 3** 6 4 2	ตัวที่ 1 : ตัวที่ 2	ตัวที่ 1 > ตัวที่ 2	สลับที่	**3 7** 6 4 2
2	3 **7 6** 4 2	ตัวที่ 2 : ตัวที่ 3	ตัวที่ 2 > ตัวที่ 3	สลับที่	3 **6 7** 4 2
3	3 6 **7 4** 2	ตัวที่ 3 : ตัวที่ 4	ตัวที่ 3 > ตัวที่ 4	สลับที่	3 6 **4 7** 2
4	3 6 4 **7 2**	ตัวที่ 4 : ตัวที่ 5	ตัวที่ 4 > ตัวที่ 5	สลับที่	3 6 4 **2** *7*

รอบที่ 2

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**3 6** 4 2 *7*	ตัวที่ 1 : ตัวที่ 2	ตัวที่ 1 < ตัวที่ 2	ไม่สลับที่	**3 6** 4 2 *7*
2	3 **6 4** 2 *7*	ตัวที่ 2 : ตัวที่ 3	ตัวที่ 2 > ตัวที่ 3	สลับที่	3 **4 6** 2 *7*
3	3 4 **6 2** *7*	ตัวที่ 3 : ตัวที่ 4	ตัวที่ 3 > ตัวที่ 4	สลับที่	3 4 **2 6** *7*

รอบที่ 3

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**3 4** 2 *6 7*	ตัวที่ 1 : ตัวที่ 2	ตัวที่ 1 < ตัวที่ 2	ไม่สลับที่	**3 4** 2 *6 7*
2	3 **4 2** *6 7*	ตัวที่ 2 : ตัวที่ 3	ตัวที่ 2 > ตัวที่ 3	สลับที่	3 **2 4** *6 7*

รอบที่ 4

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**3 2** *4 6 7*	ตัวที่ 1 : ตัวที่ 2	ตัวที่ 1 < ตัวที่ 2	สลับที่	**2 3** *4 6 7*

จากข้อมูลจำนวน 5 ตัว ทำการเรียงลำดับแบบฟองในแต่ละรอบ จะได้ตำแหน่งที่ถูกต้องของข้อมูลตั้งแต่ลำดับสุดท้ายไปเรื่อย ๆ จนถึงลำดับที่สอง นั่นคือ ข้อมูลจำนวน 5 ตัว จะทำการเรียงลำดับข้อมูลในตำแหน่งต่าง ๆ ดังนี้

รอบที่	ตำแหน่งข้อมูลที่เรียงลำดับแล้ว
1	5
2	4 5
3	3 4 5
4	1 2 3 4 5

ครบ 4 รอบ ข้อมูลทุกค่าอยู่ในตำแหน่งที่เหมาะสมจึงเสร็จสิ้นการเรียงลำดับ

สรุปได้ว่า

ข้อมูลนำเข้าจำนวน 5 ตัว: 7 3 6 4 2

ข้อมูลผลลัพธ์: 2 3 4 6 7
มีการเรียงลำดับรวม 4 รอบ
มีการเปรียบเทียบรวม 4+3+2+1 = 10 ครั้ง
มีการสลับที่รวม 8 ครั้ง

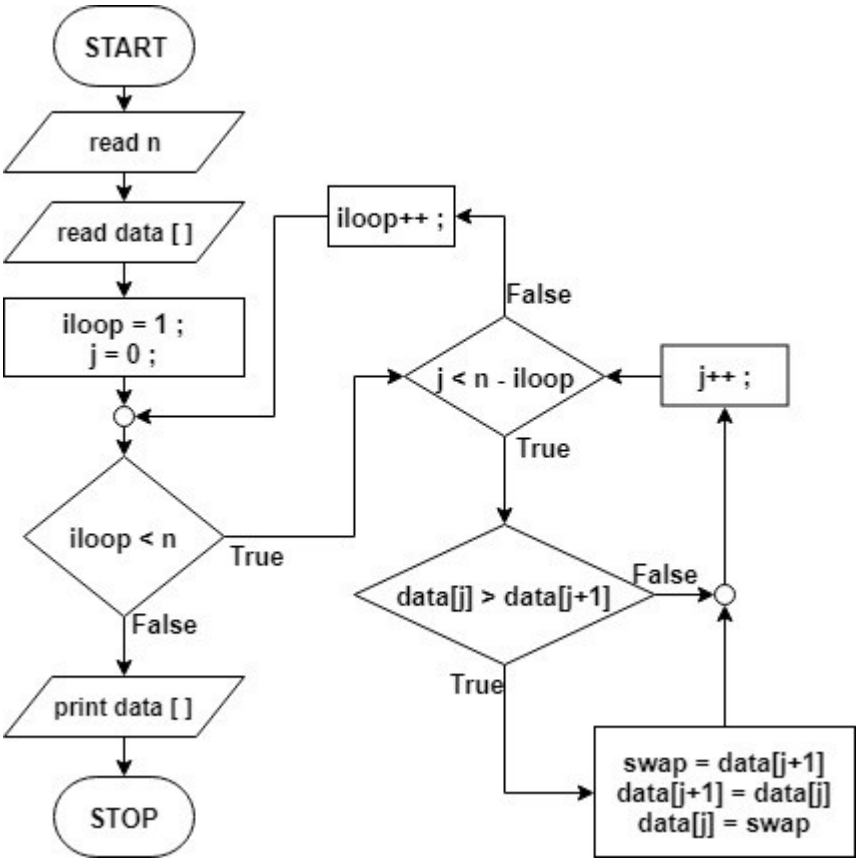
จะเห็นได้ว่าในแต่ละรอบข้อมูลมีการสลับที่กันไปเรื่อยๆ จนถึงท้ายสุดได้ข้อมูลที่มีการเรียงลำดับตามที่ต้องการเป็นที่เรียบร้อยแล้ว ซึ่งมีการจัดเรียง n-1 รอบ โดยในแต่ละรอบจะมีการเปรียบเทียบเป็น n-1, n-2, n-3, ... , 3, 2, 1 ดังนั้นจำนวนในการเปรียบเทียบรวมทั้งหมดจะเป็น

1+2+3+...+(n-2)+(n-1)	=	n(n-1)/2
	=	$(n^2-n)/2$
จากตัวอย่างเมื่อแทนในสูตรจะได้	=	$(5^2 - 5)/2$
	=	(25 - 5)/2
	=	20/2
	=	10

ต่อไปจะแสดงผังงานของการเรียงลำดับข้อมูลแบบฟองโดยเรียงลำดับข้อมูลจากน้อยไปหามาก

กำหนดตัวแปร

data[ ]	:	ตัวแปรที่เก็บข้อมูลที่ต้องการเรียงลำดับ
n	:	จำนวนข้อมูลทั้งหมดในตัวแปร data
iloop	:	ตัวแปรแทนรอบของการวนซ้ำ
j	:	ตัวแปรแทนดัชนีของข้อมูล data
swap	:	ตัวแปรชั่วคราวใช้ช่วยในการสลับค่า



รูปที่ 2.2 ผังงานของโปรแกรมการจัดเรียงข้อมูลแบบฟองโดยให้เรียงลำดับจากน้อยไปหามาก

ตัวอย่าง 2.2 กำหนดข้อมูลตัวเลขจำนวน 10 ตัว คือ 12 50 68 1 13 40 98 36 14 54 จงเรียงลำดับข้อมูลจากน้อยไปหามาก

วิธีทำ ในแต่ละขั้นตอนค่าตัวเลขในคอลัมน์ข้อมูลและคอลัมน์ผลลัพธ์ที่เน้นตัวหนา คือ ค่าที่กำลังพิจารณาเปรียบเทียบกัน ค่าตัวเลขในคอลัมน์ข้อมูล และคอลัมน์ผลลัพธ์ที่ขีดเส้นใต้และตัวเอียง คือ ค่าที่อยู่ในตำแหน่งที่เหมาะสมแล้วจึงไม่ต้องนำไปเปรียบเทียบอีกแล้ว

กำหนดให้ “[n]” แทนคำว่า “ตัวที่ n”

รอบที่ 1

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**12 50** 68 1 13 40 98 36 14 54	[1] : [2]	[1] < [2]	ไม่สลับที่	**12 50** 68 1 13 40 98 36 14 54
2	12 **50 68** 1 13 40 98 36 14 54	[2] : [3]	[2] < [3]	ไม่สลับที่	12 **50 68** 1 13 40 98 36 14 54
3	12 50 **68 1** 13 40 98 36 14 54	[3] : [4]	[3] > [4]	สลับที่	12 50 **1 68** 13 40 98 36 14 54

4	12 50 1 **68 13** 40 98 36 14 54	[4] : [5]	[4] > [5]	สลับที่	12 50 1 **13 68** 40 98 36 14 54
5	12 50 1 13 **68 40** 98 36 14 54	[5] : [6]	[5] > [6]	สลับที่	12 50 1 13 **40 68** 98 36 14 54
6	12 50 1 13 40 **68 98** 36 14 54	[6] : [7]	[6] < [7]	ไม่สลับที่	12 50 1 13 40 **68 98** 36 14 54
7	12 50 1 13 40 68 **98 36** 14 54	[7] : [8]	[7] > [8]	สลับที่	12 50 1 13 40 68 **36 98** 14 54
8	12 50 1 13 40 68 36 **98 14** 54	[8] : [9]	[8] > [9]	สลับที่	12 50 1 13 40 68 36 **14 98** 54
9	12 50 1 13 40 68 36 14 **98 54**	[9] : [10]	[9] > [10]	สลับที่	12 50 1 13 40 68 36 14 **54** ***98***

รอบที่ 2

i	**ข้อมูล**	**เปรียบเทียบข้อมูล**	**เปรียบเทียบค่า**	**การสลับข้อมูล**	**ผลลัพธ์**
1	**12 50** 1 13 40 68 36 14 54 *98*	[1] : [2]	[1] < [2]	ไม่สลับที่	**12 50** 1 13 40 68 36 14 54 *98*
2	12 **50 1** 13 40 68 36 14 54 *98*	[2] : [3]	[2] > [3]	สลับที่	12 **1 50** 13 40 68 36 14 54 *98*
3	12 1 **50 13** 40 68 36 14 54 *98*	[3] : [4]	[3] > [4]	สลับที่	12 1 **13 50** 40 68 36 14 54 *98*
4	12 1 13 **50 40** 68 36 14 54 *98*	[4] : [5]	[4] > [5]	สลับที่	12 1 13 **40 50** 68 36 14 54 *98*
5	12 1 13 40 **50 68** 36 14 54 *98*	[5] : [6]	[5] < [6]	ไม่สลับที่	12 1 13 40 **50 68** 36 14 54 *98*
6	12 1 13 40 50 **68 36** 14 54 *98*	[6] : [7]	[6] > [7]	สลับที่	12 1 13 40 50 **36 68** 14 54 *98*
7	12 1 13 40 50 36 **68 14** 54 *98*	[7] : [8]	[7] > [8]	สลับที่	12 1 13 40 50 36 **14 68** 54 *98*
8	12 1 13 40 50 36 14 **68 54** *98*	[8] : {9]	[8] > [9]	สลับที่	12 1 13 40 50 36 14 **54** ***68*** *98*

รอบที่ 3

i	**ข้อมูล**	**เปรียบเทียบข้อมูล**	**เปรียบเทียบค่า**	**การสลับข้อมูล**	**ผลลัพธ์**
1	**12 1** 13 40 50 36 14 54 *68 98*	[1] : [2]	[1] > [2]	สลับที่	**1 12** 13 40 50 36 14 54 *68 98*
2	1 **12 13** 40 50 36 14 54 *68 98*	[2] : [3]	[2] < [3]	ไม่สลับที่	1 **12 13** 40 50 36 14 54 *68 98*
3	1 12 **13 40** 50 36 14 54 *68 98*	[3] : [4]	[3] < [4]	ไม่สลับที่	1 12 **13 40** 50 36 14 54 *68 98*
4	1 12 13 **40 50** 36 14 54 *68 98*	[4] : [5]	[4] < [5]	ไม่สลับที่	1 12 13 40 **50 36** 14 54 *68 98*
5	1 12 13 40 50 **36 14** 54 *68 98*	[5] : [6]	[5] > [6]	สลับที่	1 12 13 40 50 **14 36** 54 *68 98*
6	1 12 13 40 50 14 **36 54** *68 98*	[6] : [7]	[6] < [7]	ไม่สลับที่	1 12 13 40 50 14 **36 54** *68 98*
7	1 12 13 40 50 14 36 **54** ***68*** *98*	[7] : [8]	[7] < [8]	ไม่สลับที่	1 12 13 40 50 14 36 ***54 68*** *98*

รอบที่ 4

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**1 12** 13 40 50 14 36 *54 68 98*	[1] : [2]	[1] < [2]	ไม่สลับที่	**1 12** 13 40 50 14 36 *54 68 98*
2	1 **12 13** 40 50 14 36 *54 68 98*	[2] : [3]	[2] < [3]	ไม่สลับที่	1 **12 13** 40 50 14 36 *54 68 98*
3	1 12 **13 40** 50 14 36 *54 68 98*	[3] : [4]	[3] < [4]	ไม่สลับที่	1 12 **13 40** 50 14 36 *54 68 98*
4	1 12 13 **40 50** 14 36 *54 68 98*	[4] : [5]	[4] < [5]	ไม่สลับที่	1 12 13 **40 50** 14 36 *54 68 98*
5	1 12 13 40 **50 14** 36 *54 68 98*	[5] : [6]	[5] > [6]	สลับที่	1 12 13 40 **14 50** 36 *54 68 98*
6	1 12 13 40 14 **50 36** *54 68 98*	[6] : [7]	[6] > [7]	สลับที่	1 12 13 40 14 **36** ***50*** *54 68 98*

รอบที่ 5

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**1 12** 13 40 14 36 *50 54 68 98*	[1] : [2]	[1] < [2]	ไม่สลับที่	**1 12** 13 40 14 36 *50 54 68 98*
2	1 **12 13** 40 14 36 *50 54 68 98*	[2] : [3]	[2] < [3]	ไม่สลับที่	1 **12 13** 40 14 36 *50 54 68 98*
3	1 12 **13 40** 14 36 *50 54 68 98*	[3] : [4]	[3] < [4]	ไม่สลับที่	1 12 **13 40** 14 36 *50 54 68 98*
4	1 12 13 **40 14** 36 *50 54 68 98*	[4] : [5]	[4] > [5]	สลับที่	1 12 13 **14 40** 36 *50 54 68 98*
5	1 12 13 14 **40 36** *50 54 68 98*	[5] : [6]	[5] > [6]	สลับที่	1 12 13 14 **36** ***40*** *50 54 68 98*

รอบที่ 6

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**1 12** 13 14 36 *40 50 54 68 98*	[1] : [2]	[1] < [2]	ไม่สลับที่	**1 12** 13 14 36 *40 50 54 68 98*
2	1 **12 13** 14 36 *40 50 54 68 98*	[2] : [3]	[2] < [3]	ไม่สลับที่	1 **12 13** 14 36 *40 50 54 68 98*
3	1 12 **13 14** 36 *40 50 54 68 98*	[3] : [4]	[3] < [4]	ไม่สลับที่	1 12 **13 14** 36 *40 50 54 68 98*
4	1 12 13 **14 36** *40 50 54 68 98*	[4] : [5]	[4] < [5]	ไม่สลับที่	1 12 13 **14** ***36*** *40 50 54 68 98*

รอบที่ 7

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**1 12** 13 14 *36 40 50 54 68 98*	[1] : [2]	[1] < [2]	ไม่สลับที่	**1 12** 13 14 *36 40 50 54 68 98*
2	1 **12 13** 14 *36 40 50 54 68 98*	[2] : [3]	[2] < [3]	ไม่สลับที่	1 **12 13** 14 *36 40 50 54 68 98*
3	1 12 **13 14** *36 40 50 54 68 98*	[3] : [4]	[3] < [4]	ไม่สลับที่	1 12 **13** ***14*** *36 40 50 54 68 98*

รอบที่ 8

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**1 12** 13 *14 36 40 50 54 68 98*	[1] : [2]	[1] < [2]	ไม่สลับที่	**1 12** 13 *14 36 40 50 54 68 98*
2	1 **12 13** *14 36 40 50 54 68 98*	[2] : [3]	[2] < [3]	ไม่สลับที่	1 **12** ***13*** *14 36 40 50 54 68 98*

รอบที่ 9

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**1 12** *13 14 36 40 50 54 68 98*	[1] : [2]	[1] < [2]	ไม่สลับที่	**1** ***12*** *13 14 36 40 50 54 68 98*

จากข้อมูลจำนวน 10 ตัว ทำการเรียงลำดับแบบฟองในแต่ละรอบ จะได้ตำแหน่งที่ถูกต้องของข้อมูลตั้งแต่ลำดับสุดท้ายไปเรื่อย ๆ จนถึงลำดับที่สอง นั่นคือ ข้อมูลจำนวน 10 ตัว จะทำการเรียงลำดับข้อมูลในตำแหน่งต่าง ๆ ดังนี้

รอบที่	ตำแหน่งข้อมูลที่เรียงลำดับแล้ว
1	**10**
2	9 10
3	8 9 10
4	7 8 9 10
5	6 7 8 9 10
6	5 6 7 8 9 10
7	4 5 6 7 8 9 10
8	3 4 5 6 7 8 9 10
9	1 2 3 4 5 6 7 8 9 10

ครบ 9 รอบ ข้อมูลทุกค่าอยู่ในตำแหน่งที่เหมาะสมจึงเสร็จสิ้นการเรียงลำดับ

สรุปได้ว่า

ข้อมูลนำเข้าจำนวน 10 ตัว: 12 50 68 1 13 40 98 36 14 54

ข้อมูลผลลัพธ์: 1 12 13 14 36 40 50 54 68 98

มีการเรียงลำดับรวม 9 รอบ

มีการเปรียบเทียบรวม 9+8+7+6+5+4+3+2+1 = 45 ครั้ง

จะเห็นได้ว่าในแต่ละรอบข้อมูลมีการสลับที่กันไปเรื่อยๆ จนถึงท้ายสุดได้ข้อมูลที่มีการเรียงลำดับตามที่ต้องการเป็นที่เรียบร้อยแล้ว ซึ่งมีการจัดเรียง n-1 รอบ โดยในแต่ละรอบจะมีการเปรียบเทียบเป็น n-1, n-2, n-3, ... , 3, 2, 1 ดังนั้นจำนวนในการเปรียบเทียบรวมทั้งหมดจะเป็น

1+2+3+...+(n-2)+(n-1)	=	n(n-1)/2
	=	$(n^2-n)/2$
จากตัวอย่างเมื่อแทนในสูตรจะได้	=	$(10^2 - 10)/2$
	=	(100 - 10)/2
	=	90/2
	=	45

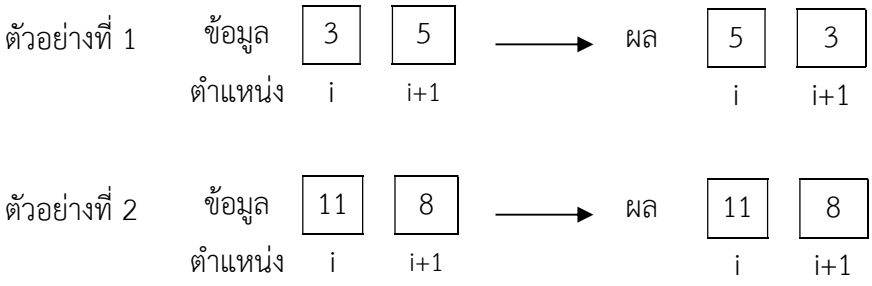
กรณีที่ 2 การเรียงลำดับข้อมูลจากมากไปหาน้อย

พิจารณาข้อมูลใน 2 ตำแหน่งที่อยู่ติดกัน คือ ตำแหน่งที่ i และตำแหน่งที่ i+1 ดังรูป

ข้อมูล				.....			.....		
ตำแหน่ง	1	2	3	.....	i	i+1	.....	n-1	n

ข้อมูลที่มีค่ามาก ต้องอยู่ในตำแหน่งก่อน (i) และ ข้อมูลที่มีค่าน้อย จะอยู่ในตำแหน่งหลัง (i+1) เรียงลำดับโดยเปรียบเทียบข้อมูลสองตัวที่อยู่ในตำแหน่งติดกัน โดยพิจารณา ตั้งแต่ i=1 และขยับ i เพิ่มขึ้นทีละ 1 ไปเรื่อย ๆ เมื่อเปรียบเทียบจนครบข้อมูล คือ i = n-1 ก็จะได้ข้อมูลที่เรียงลำดับเรียบร้อย

ถ้าข้อมูลตำแหน่งที่ i น้อยกว่าข้อมูลในตำแหน่งที่ i+1 ต้องสลับค่าข้อมูลในตำแหน่งคู่นี้ดังตัวอย่างที่ 1 ในรูปที่ 2.3 แต่ถ้าข้อมูลในตำแหน่งที่ i มีค่ามากกว่าข้อมูลในตำแหน่งที่ i+1 ก็ไม่ต้องสลับค่าข้อมูล ดังตัวอย่างที่ 2 ในรูปที่ 2.3



รูปที่ 2.3 การเปรียบเทียบข้อมูลในตำแหน่งที่ i และ i+1 โดยเรียงลำดับจากมากไปหาน้อย

ในกรณีเรียงลำดับข้อมูลจากมากไปน้อย ถ้าค่าแรกมีค่าน้อยกว่าค่าที่สอง ก็จะทำการสลับที่กัน โดยวิธีการนี้จะทำให้ข้อมูลที่มีค่าน้อยลอยสูงขึ้นไปทางด้านท้ายของแถวเหมือนฟองสบู่ (bubble) ที่ลอยขึ้นที่สูงและข้อมูลที่มีค่ามากที่สุดก็จะเลื่อนลงมาอยู่ในตำแหน่งแรกของชุดข้อมูล

ตัวอย่าง 2.3 กำหนดข้อมูลตัวเลขจำนวน 5 ตัว คือ 1 3 5 2 6 จงเรียงลำดับข้อมูลจากมากไปหาน้อย วิธีทำในแต่ละขั้นตอนค่าตัวเลขในคอลัมน์ข้อมูลและคอลัมน์ผลลัพธ์ที่เน้นตัวหนา คือ ค่าที่กำลังถูกเปรียบเทียบกัน ค่าตัวเลขคอลัมน์ในข้อมูล และผลลัพธ์ที่ขีดเส้นใต้และตัวเอียง คือ ค่าที่อยู่ในตำแหน่งที่เหมาะสมแล้วจึงไม่ต้องนำไปเปรียบเทียบอีกแล้ว

รอบที่ 1

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**1 3** 5 2 6	ตัวที่ 1 : ตัวที่ 2	ตัวที่ 1 < ตัวที่ 2	สลับที่	**3 1** 5 2 6
2	3 **1 5** 2 6	ตัวที่ 2 : ตัวที่ 3	ตัวที่ 2 < ตัวที่ 3	สลับที่	3 **5 1** 2 6
3	3 5 **1 2** 6	ตัวที่ 3 : ตัวที่ 4	ตัวที่ 3 < ตัวที่ 4	สลับที่	3 5 **2 1** 6
4	3 5 2 **1 6**	ตัวที่ 4 : ตัวที่ 5	ตัวที่ 4 < ตัวที่ 5	สลับที่	3 5 2 **6** ***1***

รอบที่ 2

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**3 5** 2 6 *1*	ตัวที่ 1 : ตัวที่ 2	ตัวที่ 1 < ตัวที่ 2	สลับที่	**5 3** 2 6 *1*
2	5 **3 2** 6 *1*	ตัวที่ 2 : ตัวที่ 3	ตัวที่ 2 > ตัวที่ 3	ไม่สลับที่	5 **3 2** 6 *1*
3	5 3 **2 6** *1*	ตัวที่ 3 : ตัวที่ 4	ตัวที่ 3 < ตัวที่ 4	สลับที่	5 3 **6 2** *1*

รอบที่ 3

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**5 3** 6 *2 1*	ตัวที่ 1 : ตัวที่ 2	ตัวที่ 1 > ตัวที่ 2	ไม่สลับที่	**5 3** 6 *2 1*
2	5 **3 6** *2 1*	ตัวที่ 2 : ตัวที่ 3	ตัวที่ 2 < ตัวที่ 3	สลับที่	5 **6 3** *2 1*

รอบที่ 4

i	ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
1	**5 6** *3 2 1*	ตัวที่ 1 : ตัวที่ 2	ตัวที่ 1 < ตัวที่ 2	สลับที่	**6 5** *3 2 1*

ข้อมูล จำนวน 5 ตัว การเรียงลำดับแบบฟองวรรคในแต่ละรอบจะได้ตำแหน่งที่ถูกต้องของข้อมูลตั้งแต่ลำดับสุดท้ายไปเรื่อย ๆ จนถึงลำดับที่สอง นั่นคือ ข้อมูลจำนวน 5 ตัว จะทำการเรียงลำดับข้อมูลในตำแหน่งต่าง ๆ ดังนี้

รอบที่	ตำแหน่งข้อมูลที่เรียงลำดับแล้ว
1	5
2	4 5
3	3 4 5
4	1 2 3 4 5

ครบ 4 รอบ ข้อมูลทุกค่าอยู่ในตำแหน่งที่เหมาะสมจึงเสร็จสิ้นการเรียงลำดับ

สรุปได้ว่า

ข้อมูลนำเข้าจำนวน 5 ตัว: 1 3 5 2 6

ข้อมูลผลลัพธ์: 6 5 3 2 1

มีการเรียงลำดับรวม 4 รอบ

มีการเปรียบเทียบรวม 4+3+2+1 = 10 ครั้ง

มีการสลับที่รวม 8 ครั้ง

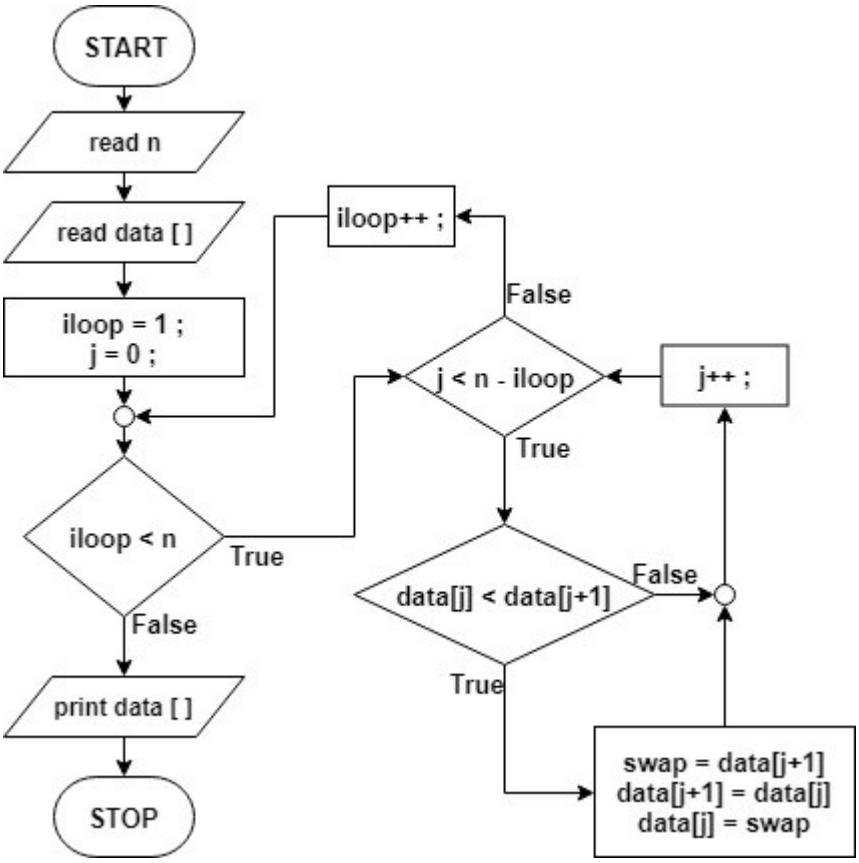
จะเห็นได้ว่าในแต่ละรอบข้อมูลมีการสลับที่กันไปเรื่อยๆ จนถึงท้ายสุดได้ข้อมูลที่มีการเรียงลำดับตามที่ต้องการเป็นที่เรียบร้อยแล้ว ซึ่งมีการจัดเรียง n-1 รอบ โดยในแต่ละรอบจะมีการเปรียบเทียบเป็น n-1, n-2, n-3, ... , 3, 2, 1 ดังนั้นจำนวนในการเปรียบเทียบรวมทั้งหมดจะเป็น

1+2+3+...+(n-2)+(n-1)	=	n(n-1)/2
	=	$(n^2-n)/2$
จากตัวอย่างเมื่อแทนในสูตรจะได้	=	$(5^2 - 5)/2$
	=	(25 - 5)/2
	=	20/2
	=	10

ต่อไปจะแสดงผังงานของการเรียงลำดับข้อมูลแบบฟองโดยเรียงลำดับข้อมูลจากมากไปหาน้อย

กำหนดตัวแปร

data[ ]	:	ตัวแปรที่เก็บข้อมูลที่ต้องการเรียงลำดับ
n	:	จำนวนข้อมูลทั้งหมดในตัวแปร data
iloop	:	ตัวแปรแทนรอบของการวนซ้ำ
j	:	ตัวแปรแทนดัชนีของข้อมูล data
swap	:	ตัวแปรชั่วคราวใช้ช่วยในการสลับค่า



รูปที่ 2.4 ผังงานของโปรแกรมการจัดเรียงข้อมูลแบบฟองโดยให้เรียงลำดับจากมากไปหาน้อย

ตัวอย่าง 2.4 กำหนดข้อมูลตัวเลขจำนวน 15 ตัว คือ 39 24 45 58 67 77 45 68 71 35 52 30 30 10 23 จงเรียงลำดับข้อมูลจากมากไปหาน้อย

วิธีทำ ในแต่ละขั้นตอนค่าตัวเลขในคอลัมน์ข้อมูลและคอลัมน์ผลลัพธ์ที่เน้นตัวหนา คือ ค่าที่กำลังพิจารณาเปรียบเทียบกัน ค่าตัวเลขในคอลัมน์ข้อมูล และคอลัมน์ผลลัพธ์ที่ขีดเส้นใต้และตัวเอียง คือ ค่าที่อยู่ในตำแหน่งที่เหมาะสมแล้วจึงไม่ต้องนำไปเปรียบเทียบอีกแล้ว

รอบที่ 1

i	ข้อมูล	การสลับข้อมูล	ผลลัพธ์
1	**39 24** 45 58 67 77 45 68 71 35 52 30 30 10 23	ไม่สลับที่	**39 24** 45 58 67 77 45 68 71 35 52 30 30 10 23
2	39 **24 45** 58 67 77 45 68 71 35 52 30 30 10 23	สลับที่	39 **45 24** 58 67 77 45 68 71 35 52 30 30 10 23
3	39 45 **24 58** 67 77 45 68 71 35 52 30 30 10 23	สลับที่	39 45 **58 24** 67 77 45 68 71 35 52 30 30 10 23
4	39 45 58 **24 67** 77 45 68 71 35 52 30 30 10 23	สลับที่	39 45 58 **67 24** 77 45 68 71 35 52 30 30 10 23
5	39 45 58 67 **24 77** 45 68 71 35 52 30 30 10 23	สลับที่	39 45 58 67 **77 24** 45 68 71 35 52 30 30 10 23

6	39 45 58 67 77 **24 45** 68 71 35 52 30 30 10 23	สลับที่	39 45 58 67 77 **45 24** 68 71 35 52 30 30 10 23
7	39 45 58 67 77 45 **24 68** 71 35 52 30 30 10 23	สลับที่	39 45 58 67 77 45 **68 24** 71 35 52 30 30 10 23
8	39 45 58 67 77 45 68 **24 71** 35 52 30 30 10 23	สลับที่	39 45 58 67 77 45 68 **71 24** 35 52 30 30 10 23
9	39 45 58 67 77 45 68 71 **24 35** 52 30 30 10 23	สลับที่	39 45 58 67 77 45 68 71 **35 24** 52 30 30 10 23
10	39 45 58 67 77 45 68 71 35 **24 52** 30 30 10 23	สลับที่	39 45 58 67 77 45 68 71 35 **52 24** 30 30 10 23
11	39 45 58 67 77 45 68 71 35 52 **24 30** 30 10 23	สลับที่	39 45 58 67 77 45 68 71 35 52 **30 24** 30 10 23
12	39 45 58 67 77 45 68 71 35 52 30 **24 30** 10 23	สลับที่	39 45 58 67 77 45 68 71 35 52 30 **30 24** 10 23
13	39 45 58 67 77 45 68 71 35 52 30 30 **24 10** 23	ไม่สลับที่	39 45 58 67 77 45 68 71 35 52 30 30 **24 10** 23
14	39 45 58 67 77 45 68 71 35 52 30 30 24 **10 23**	สลับที่	39 45 58 67 77 45 68 71 35 52 30 30 24 **23 *10***

รอบที่ 2

i	ข้อมูล	การสลับข้อมูล	ผลลัพธ์
1	**39 45** 58 67 77 45 68 71 35 52 30 30 24 23 *10*	สลับที่	**45 39** 58 67 77 45 68 71 35 52 30 30 24 23 *10*
2	45 **39 58** 67 77 45 68 71 35 52 30 30 24 23 *10*	สลับที่	45 **58 39** 67 77 45 68 71 35 52 30 30 24 23 *10*
3	45 58 **39 67** 77 45 68 71 35 52 30 30 24 23 *10*	สลับที่	45 58 **67 39** 77 45 68 71 35 52 30 30 24 23 *10*
4	45 58 67 **39 77** 45 68 71 35 52 30 30 24 23 *10*	สลับที่	45 58 67 **77 39** 45 68 71 35 52 30 30 24 23 *10*
5	45 58 67 77 **39 45** 68 71 35 52 30 30 24 23 *10*	สลับที่	45 58 67 77 **45 39** 68 71 35 52 30 30 24 23 *10*
6	45 58 67 77 45 **39 68** 71 35 52 30 30 24 23 *10*	สลับที่	45 58 67 77 45 **68 39** 71 35 52 30 30 24 23 *10*
7	45 58 67 77 45 68 **39 71** 35 52 30 30 24 23 *10*	สลับที่	45 58 67 77 45 68 **71 39** 35 52 30 30 24 23 *10*
8	45 58 67 77 45 68 71 **39 35** 52 30 30 24 23 *10*	ไม่สลับที่	45 58 67 77 45 68 71 **39 35** 52 30 30 24 23 *10*
9	45 58 67 77 45 68 71 39 **35 52** 30 30 24 23 *10*	สลับที่	45 58 67 77 45 68 71 39 **52 35** 30 30 24 23 *10*
10	45 58 67 77 45 68 71 39 52 **35 30** 30 24 23 *10*	ไม่สลับที่	45 58 67 77 45 68 71 39 52 **35 30** 30 24 23 *10*
11	45 58 67 77 45 68 71 39 52 35 **30 30** 24 23 *10*	ไม่สลับที่	45 58 67 77 45 68 71 39 52 35 **30 30** 24 23 *10*
12	45 58 67 77 45 68 71 39 52 35 30 **30 24** 23 *10*	ไม่สลับที่	45 58 67 77 45 68 71 39 52 35 30 **30 24** 23 *10*
13	45 58 67 77 45 68 71 39 52 35 30 30 **24 23** *10*	ไม่สลับที่	45 58 67 77 45 68 71 39 52 35 30 30 **24 *23*** *10*

รอบที่ 3

i	ข้อมูล	การสลับข้อมูล	ผลลัพธ์
1	**45 58** 67 77 45 68 71 39 52 35 30 30 24 *23 10*	สลับที่	**58 45** 67 77 45 68 71 39 52 35 30 30 24 *23 10*
2	58 **45 67** 77 45 68 71 39 52 35 30 30 24 *23 10*	สลับที่	58 **67 45** 77 45 68 71 39 52 35 30 30 24 *23 10*
3	58 67 **45 77** 45 68 71 39 52 35 30 30 24 *23 10*	สลับที่	58 67 **77 45** 45 68 71 39 52 35 30 30 24 *23 10*
4	58 67 77 **45 45** 68 71 39 52 35 30 30 24 *23 10*	ไม่สลับที่	58 67 77 **45 45** 68 71 39 52 35 30 30 24 *23 10*
5	58 67 77 45 **45 68** 71 39 52 35 30 30 24 *23 10*	สลับที่	58 67 77 45 **68 45** 71 39 52 35 30 30 24 *23 10*

6	58 67 77 45 68 **45 71** 39 52 35 30 30 24 *23 10*	สลับที่	58 67 77 45 68 **71 45** 39 52 35 30 30 24 *23 10*
7	58 67 77 45 68 71 **45 39** 52 35 30 30 24 *23 10*	ไม่สลับที่	58 67 77 45 68 71 **45 39** 52 35 30 30 24 *23 10*
8	58 67 77 45 68 71 45 **39 52** 35 30 30 24 *23 10*	สลับที่	58 67 77 45 68 71 45 **52 39** 35 30 30 24 *23 10*
9	58 67 77 45 68 71 45 52 **39 35** 30 30 24 *23 10*	ไม่สลับที่	58 67 77 45 68 71 45 52 **39 35** 30 30 24 *23 10*
10	58 67 77 45 68 71 45 52 39 **35 30** 30 24 *23 10*	ไม่สลับที่	58 67 77 45 68 71 45 52 39 **35 30** 30 24 *23 10*
11	58 67 77 45 68 71 45 52 39 35 **30 30** 24 *23 10*	ไม่สลับที่	58 67 77 45 68 71 45 52 39 35 **30 30** 24 *23 10*
12	58 67 77 45 68 71 45 52 39 35 30 **30 24** *23 10*	ไม่สลับที่	58 67 77 45 68 71 45 52 39 35 30 **30** ***24*** *23 10*

.

.

.

รอบที่ 11

i	ข้อมูล	การสลับข้อมูล	ผลลัพธ์
1	**77 71** 68 67 58 *52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	**77 71** 68 67 58 *52 45 45 39 35 30 30 24 23 10*
2	77 **71 68** 67 58 *52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	77 **71 68** 67 58 *52 45 45 39 35 30 30 24 23 10*
3	77 71 **68 67** 58 *52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	77 **71 68 67** 58 *52 45 45 39 35 30 30 24 23 10*
4	77 71 68 **67 58** *52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	77 71 68 **67** ***58*** *52 45 45 39 35 30 30 24 23 10*

รอบที่ 12

i	ข้อมูล	การสลับข้อมูล	ผลลัพธ์
1	**77 71** 68 67 *58 52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	**77 71** 68 67 *58 52 45 45 39 35 30 30 24 23 10*
2	77 **71 68** 67 *58 52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	77 **71 68** 67 *58 52 45 45 39 35 30 30 24 23 10*
3	77 71 **68 67** *58 52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	77 71 **68** ***67*** *58 52 45 45 39 35 30 30 24 23 10*

รอบที่ 13

i	ข้อมูล	การสลับข้อมูล	ผลลัพธ์
1	**77 71** 68 *67 58 52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	**77 71** 68 *67 58 52 45 45 39 35 30 30 24 23 10*
2	77 **71 68** *67 58 52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	77 **71** ***68*** *67 58 52 45 45 39 35 30 30 24 23 10*

รอบที่ 14

i	ข้อมูล	การสลับข้อมูล	ผลลัพธ์
1	**77 71** *68 67 58 52 45 45 39 35 30 30 24 23 10*	ไม่สลับที่	**77 71** *68 67 58 52 45 45 39 35 30 30 24 23 10*

จากข้อมูลจำนวน 15 ตัว ทำการเรียงลำดับแบบฟองในแต่ละรอบ จะได้ตำแหน่งที่ถูกต้องของข้อมูลตั้งแต่ลำดับสุดท้ายไปเรื่อย ๆ จนถึงลำดับที่สอง นั่นคือ ข้อมูลจำนวน 15 ตัว จะทำการเรียงลำดับข้อมูลในตำแหน่งต่าง ๆ ดังนี้

รอบที่	ตำแหน่งข้อมูลที่เรียงลำดับแล้ว
1	15
2	14 15
3	13 14 15
4	12 13 14 15
5	11 12 13 14 15
6	10 11 12 13 14 15
7	9 10 11 12 13 14 15
8	8 9 10 11 12 13 14 15
9	7 8 9 10 11 12 13 14 15
10	6 7 8 9 10 11 12 13 14 15
11	5 6 7 8 9 10 11 12 13 14 15
12	4 5 6 7 8 9 10 11 12 13 14 15
13	3 4 5 6 7 8 9 10 11 12 13 14 15
14	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

ครบ 14 รอบ ข้อมูลทุกค่าอยู่ในตำแหน่งที่เหมาะสมจึงเสร็จสิ้นการเรียงลำดับ

สรุปได้ว่า

ข้อมูลนำเข้าจำนวน 15 ตัว: 39 24 45 58 67 77 45 68 71 35 52 30 30 10 23

ข้อมูลผลลัพธ์: 77 71 68 67 58 52 45 45 39 35 30 30 24 23 10

มีการเรียงลำดับรวม 14 รอบ

มีการเปรียบเทียบรวม 14+13+12+11+10+9+8+7+6+5+4+3+2+1 = 105 ครั้ง

จะเห็นได้ว่าในแต่ละรอบข้อมูลมีการสลับที่กันไปเรื่อยๆ จนถึงท้ายสุดได้ข้อมูลที่มีการเรียงลำดับตามที่ต้องการเป็นที่เรียบร้อยแล้ว ซึ่งมีการจัดเรียง n-1 รอบ โดยในแต่ละรอบจะมีการเปรียบเทียบเป็น n-1, n-2, n-3, ... , 3, 2, 1 ดังนั้นจำนวนในการเปรียบเทียบรวมทั้งหมดจะเป็น

$1+2+3+...+(n-2)+(n-1) = n(n-1)/2$

$= (n^2-n)/2$

จากตัวอย่างเมื่อแทนในสูตรจะได้ $= (15^2 - 15)/2$

$= (225 - 15)/2$

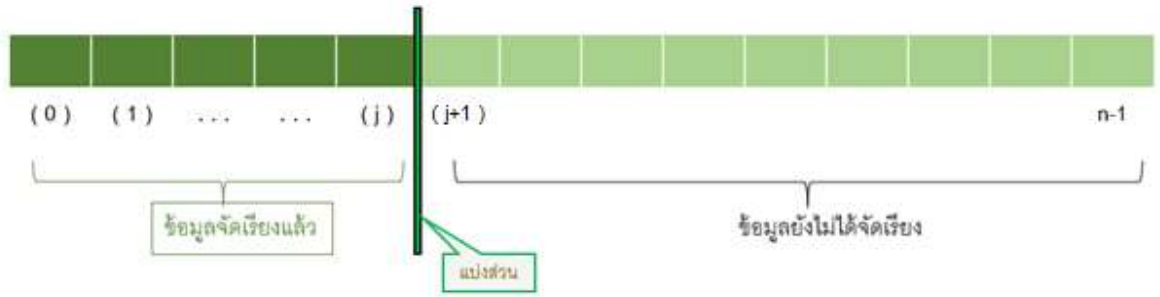
$= 210/2$

$= 105$

เมื่อพิจารณาวิธีการจัดเรียงแบบฟอง (Bubble Sort) กรณีที่แย่ที่สุดในการจัดเรียงจะเป็นกรณีที่มีการสลับที่ตำแหน่งค่าของข้อมูลและการเปรียบเทียบค่าข้อมูลในทุกครั้งๆ ที่มีการเปรียบเทียบ

2.1.2 วิธีการจัดเรียงลำดับข้อมูลแบบแทรก (Insertion Sort)

การเรียงข้อมูลแบบแทรก จะทำโดยการนำค่าข้อมูลที่ยังไม่ได้จัดเรียง ไปเปรียบเทียบกับค่าข้อมูลที่จัดเรียงแล้วทีละตัว โดยแทรกข้อมูลที่ต้องการจะจัดเรียงลงในตำแหน่งที่เหมาะสมของข้อมูลส่วนที่มีการจัดเรียงแล้วให้ถูกต้อง โดยการเรียงข้อมูลแบบแทรกนี้จะกำหนดแบ่งข้อมูลออกเป็นสองส่วน คือ ข้อมูลส่วนที่จัดเรียงแล้ว และข้อมูลส่วนที่ยังไม่ได้จัดเรียง โดยข้อมูลส่วนที่จัดเรียงแล้วจะอยู่ทางด้านซ้าย และข้อมูลส่วนที่ยังไม่ได้จัดเรียงจะอยู่ทางด้านขวา ดังรูปที่ 2.5

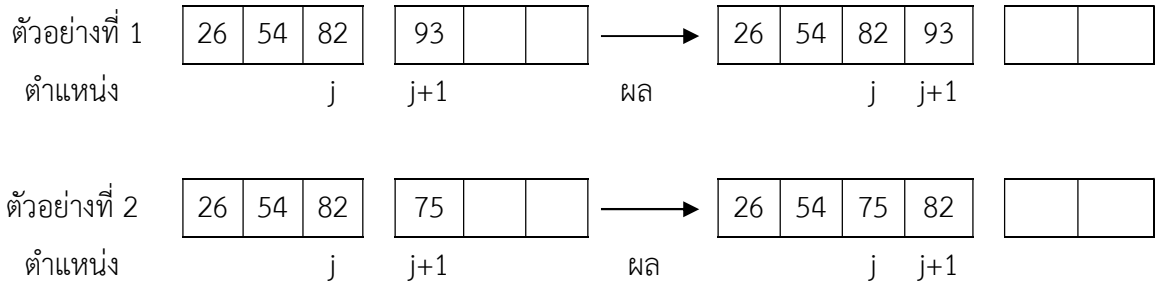


รูปที่ 2.5 แสดงการแบ่งข้อมูลในวิธีการจัดเรียงลำดับข้อมูลแบบแทรก

กรณีที่ 1 การเรียงลำดับข้อมูล จากน้อยไปหามาก คือ เปรียบเทียบค่ากับตำแหน่งถัดไปแล้วสลับตำแหน่งให้ข้อมูลอยู่ในที่เหมาะสม โดยข้อมูลที่มีค่าน้อยต้องอยู่ในตำแหน่งก่อน (j) และข้อมูลที่มีค่ามากจะอยู่ในตำแหน่งหลัง (j+1) ดังตัวอย่าง 2.4 เปรียบเทียบข้อมูลที่ต้องการจัดเรียงทีละตัวจากซ้ายไปขวาโดยข้อมูลส่วนที่จัดเรียงแล้วจะอยู่ทางด้านซ้าย

ทำต่อเนื่องไปเรื่อย ๆ ในกรณีเรียงลำดับข้อมูลจากน้อยไปหามาก ถ้าค่าที่พิจารณามีค่าน้อยกว่าค่าก่อนหน้า ก็จะทำการสลับที่กัน โดยวิธีการนี้จะทำให้ข้อมูลแทรกลงในตำแหน่งที่เหมาะสมของข้อมูลส่วนที่จัดเรียงแล้ว ลักษณะเหมือนการเรียงไพ่ในมือ ผู้เล่นรับไพ่มาทีละใบและแทรกลงในกองไพ่ในมือในตำแหน่งที่เหมาะสม โดยกองไพ่ในมือนั้นจัดเรียงไว้แล้ว อาจจะเรียงจากน้อยไปมากหรือมากไปน้อยก็ได้ เมื่อรับไพ่และแทรกลงจนครบ กองไพ่ในมือจะจัดเรียงกันได้อย่างถูกต้อง

ตัวอย่าง 2.5 การนำข้อมูลในตำแหน่งที่ j+1 มาเรียงลำดับกับข้อมูลที่เรียงลำดับแล้วในตำแหน่งที่ 1 ถึง j



รูปที่ 2.6 การเรียงลำดับข้อมูลตำแหน่งที่ j+1

ตัวอย่าง 2.6 กำหนดข้อมูลตัวเลขจำนวน 5 ตัวคือ 7 3 6 4 2 จงเรียงลำดับจากน้อยไปหามาก ในแต่ละขั้นตอนค่าตัวเลขที่อยู่ในเครื่องหมายวงเล็บ ถ้าอยู่ในคอลัมน์ข้อมูลนำเข้า คือ ค่าตัวเลขที่กำลังพิจารณา ถ้าอยู่ในคอลัมน์ผลลัพธ์ คือ ค่าตัวเลขที่ถือว่าเป็นข้อมูลที่จัดเรียงแล้ว และค่าตัวเลขที่ถูกเน้นตัวหนา คือ ค่าตัวเลขที่กำลังถูกเปรียบเทียบ

รอบที่ 1 พิจารณาข้อมูลตำแหน่งที่ 1-2

ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
(7 **3**) 6 4 2	ตัวที่ 2 : ตัวที่ 1	ตัวที่ 2 < ตัวที่ 1	สลับที่	(**3** 7) 6 4 2

รอบที่ 2 พิจารณาข้อมูลตำแหน่งที่ 1-3

ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
(3 7 **6**) 4 2	ตัวที่ 3 : ตัวที่ 2	ตัวที่ 3 < ตัวที่ 2	สลับที่	(3 **6** 7) 4 2
(3 **6** 7) 4 2	ตัวที่ 2 : ตัวที่ 1	ตัวที่ 2 > ตัวที่ 1	ไม่สลับที่	(3 **6** 7) 4 2

รอบที่ 3 พิจารณาข้อมูลตำแหน่งที่ 1-4

ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
(3 6 7 **4**) 2	ตัวที่ 4 : ตัวที่ 3	ตัวที่ 4 < ตัวที่ 3	สลับที่	(3 6 **4** 7) 2
(3 6 **4** 7) 2	ตัวที่ 3 : ตัวที่ 2	ตัวที่ 3 < ตัวที่ 2	สลับที่	(3 **4** 6 7) 2
(3 **4** 6 7) 2	ตัวที่ 2 : ตัวที่ 1	ตัวที่ 2 > ตัวที่ 1	ไม่สลับที่	(3 **4** 6 7) 2

รอบที่ 4 พิจารณาข้อมูลตำแหน่งที่ 1-5

ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
(3 4 6 7 **2**)	ตัวที่ 5 : ตัวที่ 4	ตัวที่ 5 < ตัวที่ 4	สลับที่	(3 4 6 **2** 7)
(3 4 6 **2** 7)	ตัวที่ 4 : ตัวที่ 3	ตัวที่ 4 < ตัวที่ 3	สลับที่	(3 4 **2** 6 7)
(3 4 **2** 6 7)	ตัวที่ 3 : ตัวที่ 2	ตัวที่ 3 < ตัวที่ 2	สลับที่	(3 **2** 4 6 7)
(3 **2** 4 6 7)	ตัวที่ 2 : ตัวที่ 1	ตัวที่ 2 < ตัวที่ 1	สลับที่	(**2** 3 4 6 7)

ข้อมูลจำนวน 5 ตัว การเรียงลำดับแบบแทรกในแต่ละรอบ จะได้ตำแหน่งค่าที่ถูกจัดเรียงของข้อมูลตั้งแต่ลำดับที่สองไปเรื่อย ๆ จนถึงลำดับสุดท้าย นั่นคือในที่นี้ ข้อมูลจำนวน 5 ตัว จะทำการเรียงลำดับข้อมูลในตำแหน่งต่าง ๆ ดังนี้

รอบที่	จำนวนข้อมูลที่จัดเรียง
1	2
2	3
3	4
4	5

ครบ 4 รอบ ข้อมูลทุกค่าอยู่ในตำแหน่งที่เหมาะสมจึงเสร็จสิ้นการเรียงลำดับ

สรุปได้ว่า

ข้อมูลนำเข้าจำนวน 5 ตัว: 7 3 6 4 2

ข้อมูลผลลัพธ์: 2 3 4 6 7

มีการเรียงลำดับรวม 4 รอบ

มีการเปรียบเทียบรวม 1+2+3+4 = 10 ครั้ง

มีการสลับที่รวม 8 ครั้ง

การจัดเรียงลำดับข้อมูลแบบแทรก จะมีการจัดเรียงลำดับข้อมูลทั้งหมด n-1 รอบ แต่มักใช้เวลาน้อยกว่าการจัดเรียงลำดับข้อมูลแบบฟอง ซึ่งในการจัดเรียงข้อมูลแต่ละรอบนั้น จำนวนการเปรียบเทียบจะไม่แน่นอน เพราะในแต่ละรอบการเปรียบเทียบจะสิ้นสุดเมื่อไม่มีการสลับตำแหน่ง ดังนั้น จะพิจารณาจำนวนการเปรียบเทียบแบ่งเป็น 3 กรณีคือ

1. กรณีดีที่สุด ข้อมูลนำเข้าถูกจัดเรียงลำดับเรียบร้อยแล้ว กรณีนี้ในการจัดเรียงแต่ละรอบจะมีการเปรียบเทียบค่าข้อมูลเพียงครั้งเดียว เพราะฉะนั้นจำนวนการเปรียบเทียบรวม คือ n-1
2. กรณีแย่ที่สุด คือ ข้อมูลนำเข้าเรียงลำดับค่าจากมากไปหาน้อย ในกรณีนี้แต่ละรอบจะมีจำนวนครั้งในการเปรียบเทียบ เป็นดังนี้

   รอบที่ 1 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 1 ครั้ง
   รอบที่ 2 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 2 ครั้ง
   รอบที่ 3 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 3 ครั้ง

   .
   .
   .

   รอบที่ n - 2 จำนวนการเปรียบเทียบทั้งหมดจะเป็น n - 2 ครั้ง
   รอบที่ n - 1 จำนวนการเปรียบเทียบทั้งหมดจะเป็น n - 1 ครั้ง

   ดังนั้น จำนวนการเปรียบเทียบทั้งหมดจะเป็น

   $1+2+3+\ldots+(n-2)+(n-1) = n(n-1)/2$

3. กรณีทั่วไป จำนวนการเปรียบเทียบทั้งหมด จะเฉลี่ยจากจำนวนการเปรียบเทียบของกรณีที่ดีที่สุดและกรณีที่แย่ที่สุด ซึ่งจะทำให้จำนวนการเปรียบเทียบทั้งหมดถูกประมาณได้จากการคำนวณดังนี้

   $((n-1) + (n(n-1)/2)) /2 = (n-1) (n+2) /4$

   $= (n^2 + n - 2)/4$

จากตัวอย่างซึ่งตรงกับกรณีเฉลี่ย จึงใช้สูตร = $(n^2 + n - 2)/4$

และเมื่อแทนในสูตรเพื่อประมาณค่าจะได้ = $(5^2 + 5 - 2)/4$

= $(25 + 5 - 2)/4$

= $28/4$

= 7

ตัวอย่าง 2.7 กำหนดข้อมูลตัวเลขจำนวน 20 ตัว คือ 195 254 44 555 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577 จงเรียงลำดับจากน้อยไปหามาก

ในแต่ละขั้นตอนค่าตัวเลขที่อยู่ในเครื่องหมายวงเล็บ ถ้าอยู่ในคอลัมน์ข้อมูลนำเข้า คือ ค่าตัวเลขที่กำลังพิจารณา ถ้าอยู่ในคอลัมน์ผลลัพธ์ คือ ค่าตัวเลขที่ถือว่าเป็นข้อมูลที่จัดเรียงแล้ว และค่าตัวเลขที่ถูกเน้นตัวหนา คือ ค่าตัวเลขที่กำลังถูกเปรียบเทียบ

รอบที่ 1 พิจารณาข้อมูลตำแหน่งที่ 1-2

ข้อมูล	(195 **254**) 44 555 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577
เปรียบเทียบข้อมูล	ตัวที่ 2 : ตัวที่ 1
เปรียบเทียบค่า	ตัวที่ 2 > ตัวที่ 1
การสลับข้อมูล	ไม่สลับที่
ผลลัพธ์	(195 **254**) 44 555 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

รอบที่ 2 พิจารณาข้อมูลตำแหน่งที่ 1-3

ข้อมูล	(195 254 **44**) 555 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577
เปรียบเทียบข้อมูล	ตัวที่ 3 : ตัวที่ 2
เปรียบเทียบค่า	ตัวที่ 3 < ตัวที่ 2
การสลับข้อมูล	สลับที่
ผลลัพธ์	(195 **44** 254) 555 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

ข้อมูล	(195 **44** 254) 555 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577
เปรียบเทียบข้อมูล	ตัวที่ 2 : ตัวที่ 1
เปรียบเทียบค่า	ตัวที่ 2 < ตัวที่ 1
การสลับข้อมูล	สลับที่
ผลลัพธ์	(**44** 195 254) 555 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

รอบที่ 3 พิจารณาข้อมูลตำแหน่งที่ 1-4

ข้อมูล	(44 195 254 **555**) 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577
เปรียบเทียบข้อมูล	ตัวที่ 4 : ตัวที่ 3
เปรียบเทียบค่า	ตัวที่ 4 > ตัวที่ 3
การสลับข้อมูล	ไม่สลับที่
ผลลัพธ์	(44 195 254 **555**) 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

รอบที่ 4 พิจารณาข้อมูลตำแหน่งที่ 1-5

ข้อมูล	(44 195 254 555 **102**) 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577
เปรียบเทียบข้อมูล	ตัวที่ 5 : ตัวที่ 4
เปรียบเทียบค่า	ตัวที่ 5 < ตัวที่ 4
การสลับข้อมูล	สลับที่
ผลลัพธ์	(44 195 254 **102** 555) 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

ข้อมูล	(44 195 254 **102** 555) 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577
เปรียบเทียบข้อมูล	ตัวที่ 4 : ตัวที่ 3
เปรียบเทียบค่า	ตัวที่ 4 < ตัวที่ 3
การสลับข้อมูล	สลับที่
ผลลัพธ์	(44 195 **102** 254 555) 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

ข้อมูล	(44 195 **102** 254 555) 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577
เปรียบเทียบข้อมูล	ตัวที่ 3 : ตัวที่ 2
เปรียบเทียบค่า	ตัวที่ 3 < ตัวที่ 2
การสลับข้อมูล	สลับที่
ผลลัพธ์	(44 **102** 195 254 555) 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

ข้อมูล	(44 **102** 195 254 555) 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577
เปรียบเทียบข้อมูล	ตัวที่ 2 : ตัวที่ 1
เปรียบเทียบค่า	ตัวที่ 2 > ตัวที่ 1
การสลับข้อมูล	ไม่สลับที่
ผลลัพธ์	(44 **102** 195 254 555) 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

.

.

.

รอบที่ 18 พิจารณาข้อมูลตำแหน่งที่ 1-19

ข้อมูล	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 761 802 828 997 **328**) 577
เปรียบเทียบข้อมูล	ตัวที่ 19 : ตัวที่ 18
เปรียบเทียบค่า	ตัวที่ 19 < ตัวที่ 18
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 761 802 828 **328** 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 761 802 828 **328** 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 18 : ตัวที่ 17
เปรียบเทียบค่า	ตัวที่ 18 < ตัวที่ 17
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 761 802 **328** 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 761 802 **328** 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 17 : ตัวที่ 16
เปรียบเทียบค่า	ตัวที่ 17 < ตัวที่ 16
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 761 **328** 802 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 761 **328** 802 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 16 : ตัวที่ 15
เปรียบเทียบค่า	ตัวที่ 16 < ตัวที่ 15
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 **328** 761 802 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 446 491 553 555 738 750 **328** 761 802 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 15 : ตัวที่ 14
เปรียบเทียบค่า	ตัวที่ 15 < ตัวที่ 14
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 491 553 555 738 **328** 750 761 802 828 997) 577
ข้อมูล	(24 44 55 102 144 195 254 410 446 491 553 555 738 **328** 750 761 802 828 997) 577

เปรียบเทียบข้อมูล	ตัวที่ 14 : ตัวที่ 13
เปรียบเทียบค่า	ตัวที่ 14 < ตัวที่ 13
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 491 553 555 **328** 738 750 761 802 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 446 491 553 555 **328** 738 750 761 802 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 13 : ตัวที่ 12
เปรียบเทียบค่า	ตัวที่ 13 < ตัวที่ 12
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 491 553 **328** 555 738 750 761 802 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 446 491 553 **328** 555 738 750 761 802 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 12 : ตัวที่ 11
เปรียบเทียบค่า	ตัวที่ 12 < ตัวที่ 11
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 491 **328** 553 555 738 750 761 802 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 446 491 **328** 553 555 738 750 761 802 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 11 : ตัวที่ 10
เปรียบเทียบค่า	ตัวที่ 11 < ตัวที่ 10
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 446 **328** 491 553 555 738 750 761 802 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 446 **328** 491 553 555 738 750 761 802 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 10 : ตัวที่ 9
เปรียบเทียบค่า	ตัวที่ 10 < ตัวที่ 9
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 410 **328** 446 491 553 555 738 750 761 802 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 410 **328** 446 491 553 555 738 750 761 802 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 9 : ตัวที่ 8
เปรียบเทียบค่า	ตัวที่ 9 < ตัวที่ 8
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 **328** 410 446 491 553 555 738 750 761 802 828 997) 577

ข้อมูล	(24 44 55 102 144 195 254 **328** 410 446 491 553 555 738 750 761 802 828 997) 577
เปรียบเทียบข้อมูล	ตัวที่ 8 : ตัวที่ 7
เปรียบเทียบค่า	ตัวที่ 8 > ตัวที่ 7
การสลับข้อมูล	ไม่สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 **328** 410 446 491 553 555 738 750 761 802 828 997) 577

รอบที่ 19 พิจารณาข้อมูลตำแหน่งที่ 1-20

ข้อมูล	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 761 802 828 997 **577**)
เปรียบเทียบข้อมูล	ตัวที่ 20 : ตัวที่ 19
เปรียบเทียบค่า	ตัวที่ 20 < ตัวที่ 19
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 761 802 828 **577** 997)

ข้อมูล	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 761 802 828 **577** 997)
เปรียบเทียบข้อมูล	ตัวที่ 19 : ตัวที่ 18
เปรียบเทียบค่า	ตัวที่ 19 < ตัวที่ 18
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 761 802 **577** 828 997)

ข้อมูล	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 761 802 **577** 828 997)
เปรียบเทียบข้อมูล	ตัวที่ 18 : ตัวที่ 17
เปรียบเทียบค่า	ตัวที่ 18 < ตัวที่ 17
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 761 **577** 802 828 997)

ข้อมูล	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 761 **577** 802 828 997)
เปรียบเทียบข้อมูล	ตัวที่ 17 : ตัวที่ 16
เปรียบเทียบค่า	ตัวที่ 17 < ตัวที่ 16
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 **577** 761 802 828 997)

ข้อมูล	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 750 **577** 761 802 828 997)
เปรียบเทียบข้อมูล	ตัวที่ 16 : ตัวที่ 15
เปรียบเทียบค่า	ตัวที่ 16 < ตัวที่ 15
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 **577** 750 761 802 828 997)

ข้อมูล	(24 44 55 102 144 195 254 328 410 446 491 553 555 738 **577** 750 761 802 828 997)
เปรียบเทียบข้อมูล	ตัวที่ 15 : ตัวที่ 14
เปรียบเทียบค่า	ตัวที่ 15 < ตัวที่ 14
การสลับข้อมูล	สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 328 410 446 491 553 555 **577** 738 750 761 802 828 997)

ข้อมูล	(24 44 55 102 144 195 254 328 410 446 491 553 555 **577** 738 750 761 802 828 997)
เปรียบเทียบข้อมูล	ตัวที่ 14 : ตัวที่ 13
เปรียบเทียบค่า	ตัวที่ 14 > ตัวที่ 13
การสลับข้อมูล	ไม่สลับที่
ผลลัพธ์	(24 44 55 102 144 195 254 328 410 446 491 553 555 **577** 738 750 761 802 828 997)

สรุปได้ว่า

ข้อมูลนำเข้าจำนวน 20 ตัว:

195 254 44 555 102 761 750 446 55 738 491 997 828 802 24 144 553 410 328 577

ข้อมูลผลลัพธ์:

24 44 55 102 144 195 254 328 410 446 491 553 555 577 738 750 761 802 828 997

มีการเรียงลำดับรวม 19 รอบ

มีการเปรียบเทียบรวม 101 ครั้ง

มีการสลับที่รวม 84 ครั้ง

การจัดเรียงลำดับข้อมูลแบบแทรก จะมีการจัดเรียงลำดับข้อมูลทั้งหมด n-1 รอบ แต่มักใช้เวลาน้อยกว่าการจัดเรียงลำดับข้อมูลแบบฟอง ซึ่งในการจัดเรียงข้อมูลแต่ละรอบนั้น จำนวนการเปรียบเทียบจะไม่แน่นอน เพราะในแต่ละรอบการเปรียบเทียบจะสิ้นสุดเมื่อไม่มีการสลับตำแหน่ง ดังนั้น จะพิจารณาจำนวนการเปรียบเทียบแบ่งเป็น 3 กรณีคือ

1. กรณีดีที่สุด ข้อมูลนำเข้าถูกจัดเรียงลำดับเรียบร้อยแล้ว กรณีนี้ในการจัดเรียงแต่ละรอบจะมีการเปรียบเทียบค่าข้อมูลเพียงครั้งเดียว เพราะฉะนั้นจำนวนการเปรียบเทียบรวม คือ n-1
2. กรณีแย่ที่สุด คือ ข้อมูลนำเข้าเรียงลำดับค่าจากมากไปหาน้อย ในกรณีนี้แต่ละรอบจะมีจำนวนครั้งในการเปรียบเทียบ เป็นดังนี้

   รอบที่ 1 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 1 ครั้ง
   รอบที่ 2 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 2 ครั้ง
   รอบที่ 3 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 3 ครั้ง

   .
   .
   .

   รอบที่ n - 2 จำนวนการเปรียบเทียบทั้งหมดจะเป็น n - 2 ครั้ง
   รอบที่ n - 1 จำนวนการเปรียบเทียบทั้งหมดจะเป็น n - 1 ครั้ง

   ดังนั้น จำนวนการเปรียบเทียบทั้งหมดจะเป็น

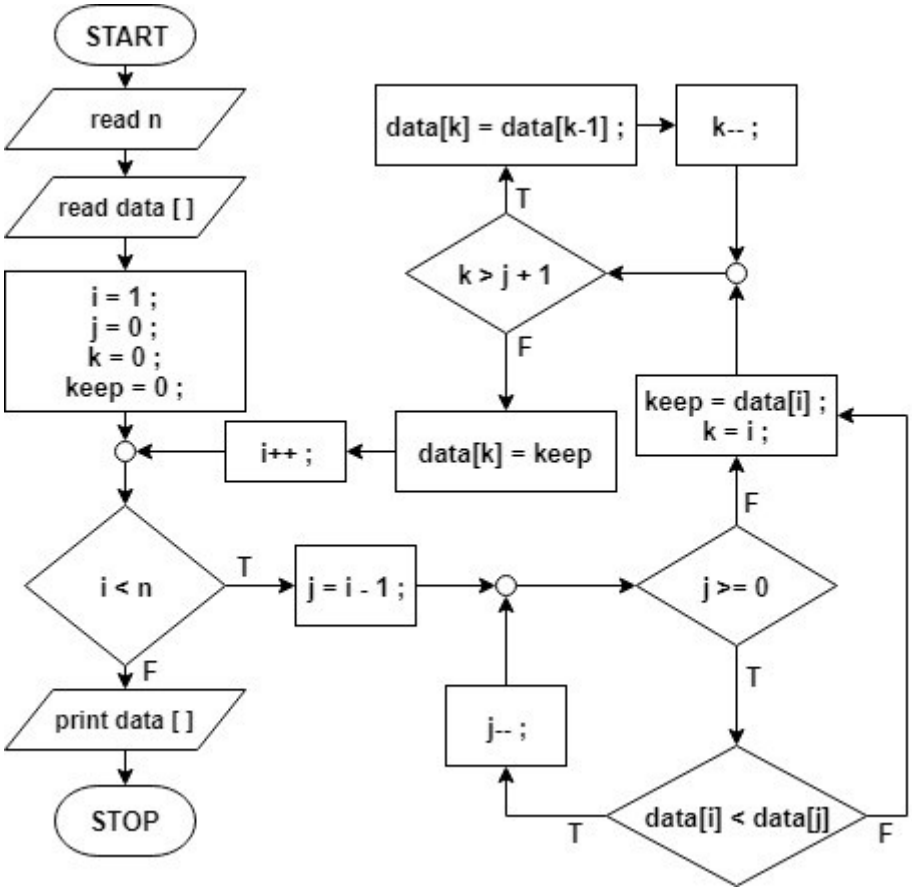
   $1+2+3+\ldots+(n-2)+(n-1) = n(n-1)/2$

3. กรณีทั่วไป จำนวนการเปรียบเทียบทั้งหมด จะเฉลี่ยจากจำนวนการเปรียบเทียบของกรณีที่ดีที่สุดและกรณีที่แย่ที่สุด ซึ่งจะทำให้จำนวนการเปรียบเทียบทั้งหมดถูกประมาณได้จากการคำนวณดังนี้

   $$((n-1) + (n(n-1)/2)) /2 = (n-1) (n+2) /4$$
   $$= (n^2 + n - 2)/4$$

   จากตัวอย่างซึ่งตรงกับกรณีเฉลี่ย จึงใช้สูตร $= (n^2 + n - 2)/4$
   และเมื่อแทนในสูตรเพื่อประมาณค่าจะได้ $= (20^2 + 20 - 2)/4$
   $= (400 + 20 - 2)/4$
   $= 418/4$
   $= 104.5$

กำหนดตัวแปร data[ ] : ที่เก็บข้อมูลที่ต้องการเรียงลำดับ
n : จำนวนข้อมูลทั้งหมดในตัวแปร data
i , j , k : ตัวแปรแทนดัชนีของข้อมูล data
keep : ตัวแปรชั่วคราวใช้ช่วยในการสลับค่าข้อมูล

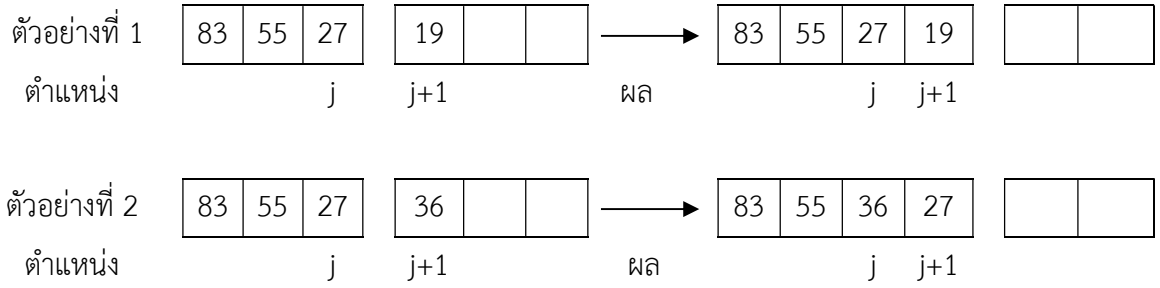


รูปที่ 2.7 ผังงานของโปรแกรมการจัดเรียงข้อมูลแบบโดยแทรกให้เรียงลำดับจากน้อยไปหามาก

กรณีที่ 2 การเรียงลำดับข้อมูล จากมากไปหาน้อย คือ เปรียบเทียบค่ากับตำแหน่งถัดไปแล้วสลับตำแหน่งให้ข้อมูลอยู่ในที่เหมาะสม โดยข้อมูลที่มีค่ามากต้องอยู่ในตำแหน่งก่อน (j) และข้อมูลที่มีค่าน้อยจะอยู่ในตำแหน่งหลัง (j+1) ดังตัวอย่าง 2.6 เปรียบเทียบข้อมูลที่ต้องการจัดเรียงทีละตัวจากซ้ายไปขวาโดยข้อมูลส่วนที่จัดเรียงแล้วจะอยู่ทางด้านซ้าย

ทำต่อเนื่องไปเรื่อย ๆ ในกรณีเรียงลำดับข้อมูลจากมากไปหาน้อย ถ้าค่าที่พิจารณามีค่ามากกว่าค่าก่อนหน้าก็จะทำการสลับที่กัน โดยวิธีการนี้จะทำให้ข้อมูลแทรกลงในตำแหน่งที่เหมาะสมของข้อมูลส่วนที่จัดเรียงแล้ว

ตัวอย่าง 2.8 การนำข้อมูลในตำแหน่งที่ j+1 มาเรียงลำดับกับข้อมูลที่เรียงลำดับแล้วในตำแหน่งที่ 1 ถึง j



รูปที่ 2.8 การเรียงลำดับข้อมูลตัวที่ j+1

ตัวอย่าง 2.9 กำหนดข้อมูลตัวเลขจำนวน 5 ตัวคือ 1 3 5 2 6 จงเรียงลำดับจากมากไปหาน้อย ในแต่ละขั้นตอนค่าตัวเลขที่อยู่ในเครื่องหมายวงเล็บ ถ้าอยู่ในช่องข้อมูลนำเข้าคือค่าตัวเลขที่กำลังพิจารณา ถ้าอยู่ในช่องผลลัพธ์คือค่าตัวเลขที่ถือว่าเป็นข้อมูลที่จัดเรียงแล้ว และค่าตัวเลขที่ถูกเน้นตัวหนาคือค่าตัวเลขที่กำลังถูกเปรียบเทียบ

รอบที่ 1

ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
(1 **3**) 5 2 6	ตัวที่ 2 : ตัวที่ 1	ตัวที่ 2 > ตัวที่ 1	สลับที่	(**3** 1) 5 2 6

รอบที่ 2

ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
(3 1 **5**) 2 6	ตัวที่ 3 : ตัวที่ 2	ตัวที่ 3 > ตัวที่ 2	สลับที่	(3 **5** 1) 2 6
(3 **5** 1) 2 6	ตัวที่ 2 : ตัวที่ 1	ตัวที่ 2 > ตัวที่ 1	สลับที่	(**5** 3 1) 2 6

รอบที่ 3

ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
(5 3 1 **2**) 6	ตัวที่ 4 : ตัวที่ 3	ตัวที่ 4 > ตัวที่ 3	สลับที่	(5 3 **2** 1) 6
(5 3 **2** 1) 6	ตัวที่ 3 : ตัวที่ 2	ตัวที่ 3 < ตัวที่ 2	ไม่สลับที่	(5 3 **2** 1) 6

รอบที่ 4

ข้อมูล	เปรียบเทียบข้อมูล	เปรียบเทียบค่า	การสลับข้อมูล	ผลลัพธ์
(5 3 2 1 **6**)	ตัวที่ 5 : ตัวที่ 4	ตัวที่ 5 > ตัวที่ 4	สลับที่	(5 3 2 **6** 1)
(5 3 2 **6** 1)	ตัวที่ 4 : ตัวที่ 3	ตัวที่ 4 > ตัวที่ 3	สลับที่	(5 3 **6** 2 1)
(5 3 **6** 2 1)	ตัวที่ 3 : ตัวที่ 2	ตัวที่ 3 > ตัวที่ 2	สลับที่	(5 **6** 3 2 1)
(5 **6** 3 2 1)	ตัวที่ 2 : ตัวที่ 1	ตัวที่ 2 > ตัวที่ 1	สลับที่	(**6** 5 3 2 1)

ข้อมูล จำนวน 5 ตัว การเรียงลำดับแบบแทรกในแต่ละรอบ จะได้ตำแหน่งค่าที่ถูกจัดเรียงของข้อมูลตั้งแต่ลำดับที่สองไปเรื่อย ๆ จนถึงลำดับสุดท้าย นั่นคือในที่นี้ ข้อมูล จำนวน 5 ตัว จะทำการเรียงลำดับข้อมูลในตำแหน่งต่าง ๆ ดังนี้

รอบที่	จำนวนข้อมูลที่จัดเรียง
1	**2**
2	**3**
3	**4**
4	**5**

ครบ 4 รอบ ข้อมูลทุกค่าอยู่ในตำแหน่งที่เหมาะสมจึงเสร็จสิ้นการเรียงลำดับ

สรุปได้ว่า

ข้อมูลนำเข้าจำนวน 5 ตัว: 1 3 5 2 6

ข้อมูลผลลัพธ์: 6 5 3 2 1

มีการเรียงลำดับรวม 4 รอบ

มีการเปรียบเทียบรวม 1+2+2+4 = 9 ครั้ง

มีการสลับที่รวม 8 ครั้ง

การจัดเรียงลำดับข้อมูลแบบแทรก จะมีการจัดเรียงลำดับข้อมูลทั้งหมด n-1 รอบ แต่มักใช้เวลาน้อยกว่าการจัดเรียงลำดับข้อมูลแบบฟอง ซึ่งในการจัดเรียงข้อมูลแต่ละรอบนั้น จำนวนการเปรียบเทียบจะไม่แน่นอน เพราะในแต่ละรอบการเปรียบเทียบจะสิ้นสุดเมื่อไม่มีการสลับตำแหน่ง ดังนั้น จะพิจารณาจำนวนการเปรียบเทียบแบ่งเป็น 3 กรณีคือ

1. กรณีดีที่สุด ข้อมูลนำเข้าถูกจัดเรียงลำดับเรียบร้อยแล้ว กรณีนี้ในการจัดเรียงแต่ละรอบจะมีการเปรียบเทียบค่าข้อมูลเพียงครั้งเดียว เพราะฉะนั้นจำนวนการเปรียบเทียบรวม คือ n-1
2. กรณีแย่ที่สุด คือ ข้อมูลนำเข้าเรียงลำดับค่าจากน้อยไปหามาก ในกรณีนี้แต่ละรอบจะมีจำนวนครั้งในการเปรียบเทียบ เป็นดังนี้

   รอบที่ 1 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 1 ครั้ง
   รอบที่ 2 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 2 ครั้ง
   รอบที่ 3 จำนวนการเปรียบเทียบทั้งหมดจะเป็น 3 ครั้ง

   .
   .
   .

   รอบที่ n - 2 จำนวนการเปรียบเทียบทั้งหมดจะเป็น n - 2 ครั้ง
   รอบที่ n - 1 จำนวนการเปรียบเทียบทั้งหมดจะเป็น n - 1 ครั้ง

   ดังนั้น จำนวนการเปรียบเทียบทั้งหมดจะเป็น

   $1+2+3+\ldots+(n-2)+(n-1) = n(n-1)/2$

3. กรณีทั่วไป จำนวนการเปรียบเทียบทั้งหมด จะเฉลี่ยจากจำนวนการเปรียบเทียบของกรณีที่ดีที่สุดและกรณีที่แย่ที่สุด ซึ่งจะทำให้จำนวนการเปรียบเทียบทั้งหมดถูกประมาณได้จากการคำนวณดังนี้

   $$((n - 1) + (n(n-1)/2)) /2 = (n - 1) (n + 2) /4$$
   $$= (n^2 + n - 2)/4$$

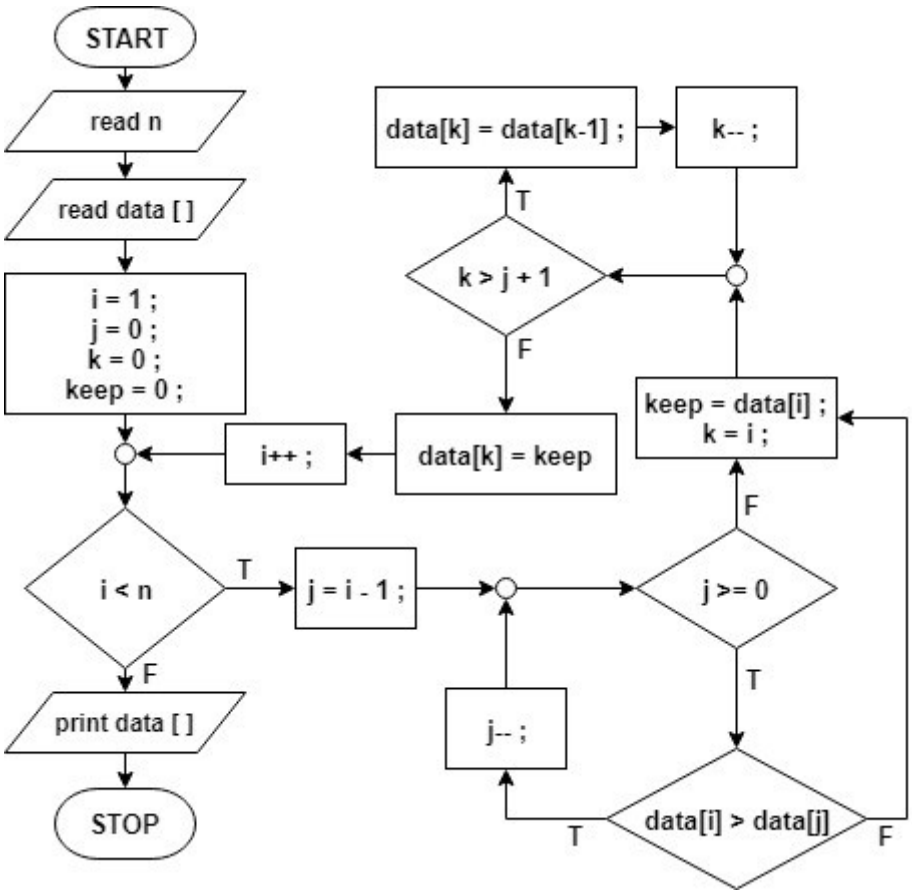
   จากตัวอย่างซึ่งตรงกับกรณีเฉลี่ย จึงใช้สูตร $= (n^2 + n - 2)/4$
   และเมื่อแทนในสูตรเพื่อประมาณค่าจะได้ $= (5^2 + 5 - 2)/4$
   $= (25 + 5 - 2)/4$
   $= 28/4$
   $= 7$

กำหนดตัวแปร data[ ] : ที่เก็บข้อมูลที่ต้องการเรียงลำดับ

n : จำนวนข้อมูลทั้งหมดในตัวแปร data

i , j , k : ตัวแปรแทนดัชนีของข้อมูล data

keep : ตัวแปรชั่วคราวใช้ช่วยในการสลับค่า

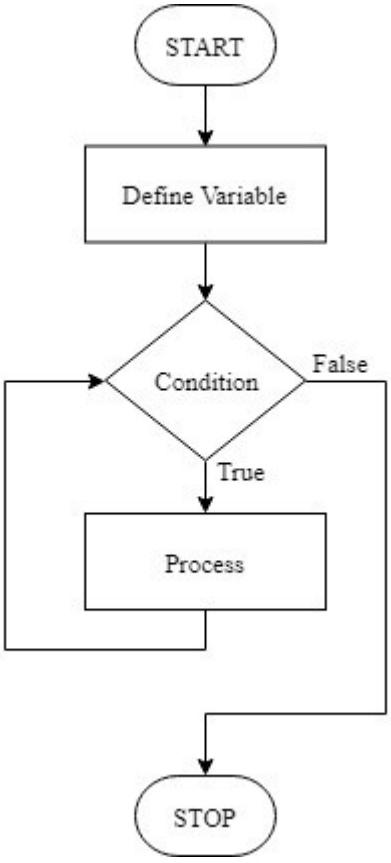


รูปที่ 2.9 ผังงานของโปรแกรมการจัดเรียงข้อมูลแบบโดยแทรกให้เรียงลำดับจากมากไปหาน้อย

2.2 ภาษาซีที่ใช้ในโครงงานนี้

2.2.1 การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for-to

การทำงานแบบวนซ้ำโดยใช้คำสั่ง for-to คือ การควบคุมโปรแกรมเพื่อให้ชุดคำสั่งหนึ่งของโปรแกรมเกิดการทำงานซ้ำตามเงื่อนไขที่กำหนดโดยใช้คำสั่ง for-to ดังรูปที่ 2.10



รูปที่ 2.10 ผังงานและตัวอย่างซอร์สโค้ดการทำงานแบบวนซ้ำโดยใช้คำสั่ง for-to (ก)

```
//---START---

#include <stdio.h>

int main()
{

//---Define Variable---

      int i=0, n=10 ;

//---Condition---

      for( i=1 ; i<10 ; i++ )
```

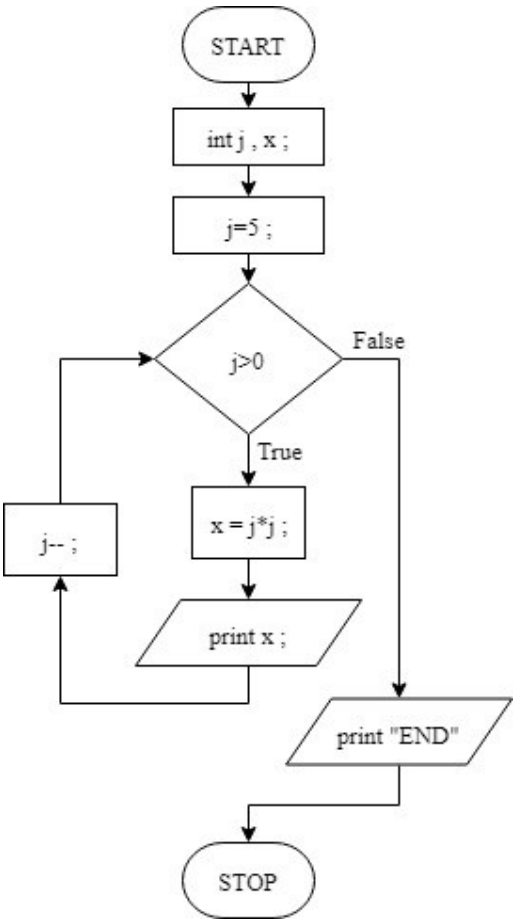
```
//---Process---
      {
         printf("%d ",n+i);
      }

//---STOP---

  return 0;
}
```

รูปที่ 2.10 ผังงานและตัวอย่างซอร์สโค้ดการทำงานแบบวนซ้ำโดยใช้คำสั่ง for-to (ข)

จากรูปที่ 2.10 (ก)
เริ่ม กำหนดค่าตัวแปร (Define Variable) แล้วจึงนำมาตรวจสอบเงื่อนไข (Condition)
ถ้า เงื่อนไขมีค่าเป็นจริง (True) ให้ทำชุดคำสั่ง (Process)
แล้ววนกลับไปตรวจสอบเงื่อนไข (Condition) อีกครั้งหนึ่ง
ถ้าเงื่อนไขเป็นจริงก็จะทำชุดคำสั่งซ้ำวนไปจนกว่า เงื่อนไขจะมีค่าเป็นเท็จ (False)
จึงจบการทำงาน

จากรูปที่ 2.10 (ข)
เริ่ม กำหนดค่าตัวแปร i = 0 และ n=10 เริ่มต้นใช้คำสั่ง for โดยกำหนดค่าตัวแปร i=1 แล้ว ตรวจสอบเงื่อนไขเพื่อทำงานแบบวนซ้ำโดยเปรียบเทียบเงื่อนไข i < 10 ว่าเป็นจริง (True) หรือเท็จ (False)
ถ้า เงื่อนไข i < 10 มีค่าเป็นจริง ให้ทำชุดคำสั่งคือ แสดงผลค่าตัวเลข n+i ออกทางหน้าจอ เสร็จแล้ว ทำคำสั่งเปลี่ยนค่าตัวแปร i++
แล้วจึงแล้ววนกลับไปตรวจสอบเงื่อนไข i < 10 ว่าเป็นจริงหรือเท็จ
ถ้า เงื่อนไข i < 10 มีค่าเป็นจริง ให้ทำชุดคำสั่งคือ แสดงผลค่าตัวเลข n+i ออกทางหน้าจอ เสร็จแล้ว จึงทำคำสั่งเปลี่ยนค่าตัวแปร i++
ทำวนซ้ำจนกว่า เงื่อนไขจะมีค่าเป็นเท็จ (False) จึงจบการทำงาน
ตัวอย่างซอร์สโค้ดนี้จะแสดงผลเป็น 1 2 3 4 5 6 7 8 9 10

ตัวอย่าง 2.10 การทำงานแบบวนซ้ำโดยใช้คำสั่ง for-to




```
#include <stdio.h>
int main()
{
       int j , x ;

       for( j=5 ; j>0 ; j-- )
       {
          x = j*j;
          printf("%d ",x);
       }

       printf("END");
  return 0;
}
```

รูปที่ 2.11 ผังงานและซอร์สโค้ดตัวอย่างการทำงานแบบวนซ้ำโดยใช้คำสั่ง for-to

จากรูปที่ 2.11

เริ่ม กำหนดตัวแปร j , x

เริ่มต้นการใช้คำสั่ง for โดยกำหนด j = 5;

แล้วทำลำดับที่ 1 คือ ตรวจสอบเงื่อนไขว่า j > 0 หรือไม่

ถ้า j > 0 ก็จะทำชุดคำสั่ง กำหนดค่าตัวแปร x = j*j คือ x = 5*5

และแสดงผลค่า x คือ พิมพ์ 25

แล้วจึงเปลี่ยนค่าตัวแปร j = j-1 จะได้ j = 4

ลำดับที่ 2 วนกลับไปตรวจสอบเงื่อนไขอีกครั้งหนึ่งว่า j > 0 หรือไม่

ถ้า j > 0 ก็จะทำชุดคำสั่ง กำหนดค่าตัวแปร x = j*j คือ x = 4*4

และแสดงผลค่า x คือ พิมพ์ 16

แล้วจึงเปลี่ยนค่าตัวแปร j = j-1 จะได้ j = 3

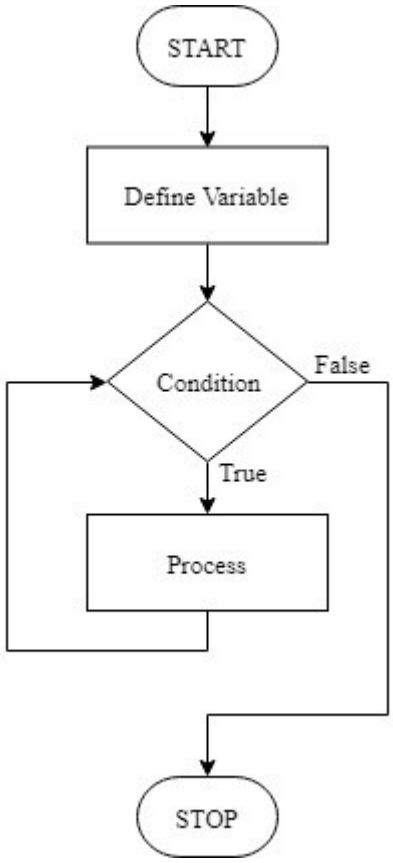
ทำวนซ้ำจนกว่า เงื่อนไขจะมีค่าเป็นเท็จคือ j มีค่าไม่มากกว่า 0 จึงจบการทำงาน คือลำดับที่ 6 ซึ่งมีลำดับการทำงานดังตารางต่อไปนี้

ลำดับการทำงาน	กำหนดตัวแปร	ตรวจสอบเงื่อนไข j > 0	ผลการตรวจสอบเงื่อนไข j > 0	กำหนดค่าตัวแปร	พิมพ์	เปลี่ยนค่าตัวแปร
1	j = 5	5 > 0	จริง	x = 5*5	25	j = 4
2		4 > 0	จริง	x = 4*4	16	j = 3
3		3 > 0	จริง	x = 3*3	9	j = 2
4		2 > 0	จริง	x = 2*2	4	j = 1
5		1 > 0	จริง	x = 1*1	1	j = 0
6		0>0	เท็จ		END	

จากรูปที่ 2.11 จะได้ผลลัพธ์เป็นการพิมพ์ 25 16 9 4 1 END

2.2.2 การทำงานแบบวนซ้ำ โดยใช้คำสั่ง while

การทำงานแบบวนซ้ำโดยใช้คำสั่ง while คือ การควบคุมโปรแกรมเพื่อให้ชุดคำสั่งหนึ่งของโปรแกรมเกิดการทำงานซ้ำตามเงื่อนไขที่กำหนดโดยใช้คำสั่ง while ดังรูปที่ 2.12



รูปที่ 2.12 ผังงานและตัวอย่างซอร์สโค้ดการทำงานแบบวนซ้ำโดยใช้คำสั่ง while (ก)

```
//---START---
#include <stdio.h>
int main()
{

//---Define Variable---

      int i=10 ;

//---Condition---

      while( i>=0 )

//---Process---
      {
         i--;
      }
```

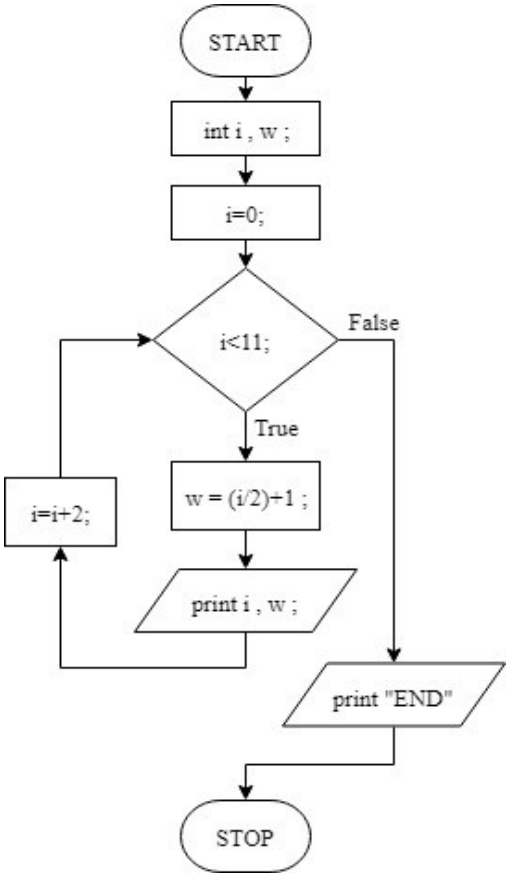
```
//---STOP---

  return 0;
}
```

รูปที่ 2.12 ผังงานและตัวอย่างซอร์สโค้ดการทำงานแบบวนซ้ำโดยใช้คำสั่ง while (ข)

จากรูปที่ 2.12 (ก)
เริ่มกำหนดค่าตัวแปร (Define Variable) แล้วตรวจสอบเงื่อนไข (Condition)
ถ้า เงื่อนไขมีค่าเป็นจริง (True) ให้ทำชุดคำสั่ง (Process)
แล้ววนกลับไปตรวจสอบเงื่อนไข (Condition) อีกครั้งหนึ่ง
ถ้าเงื่อนไขเป็นจริงก็จะทำชุดคำสั่ง (Process) ซ้ำวนไปจนกว่า เงื่อนไขจะมีค่าเป็นเท็จ (False)
จึงจบการทำงาน

จากรูปที่ 2.12 (ข)
เริ่มกำหนดค่าตัวแปร i = 10 แล้วตรวจสอบเงื่อนไขเพื่อทำงานแบบวนซ้ำ
คือคำสั่ง while เพื่อเปรียบเทียบเงื่อนไข i >= 0 ว่าเป็นจริง (True) หรือเท็จ (False)
ถ้า เงื่อนไข i >= 0 มีค่าเป็นจริงให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ i--;
แล้วจึงแล้ววนกลับไปตรวจสอบเงื่อนไข i >= 0 ว่าเป็นจริงหรือเท็จ
ถ้า เงื่อนไข i >= 0 มีค่าเป็นจริงให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ i--;
ทำวนซ้ำจนกว่า เงื่อนไขจะมีค่าเป็นเท็จ จึงจบการทำงาน

ตัวอย่าง 2.11 การทำงานแบบวนซ้ำโดยใช้คำสั่ง while




```
#include <stdio.h>
int main()
{
       int i,w ;

       i=0;

       while( i<11 )
       {

          w=(i/2)+1;

          printf("%d %d ",i,w);

          i=i+2;
       }


  return 0;
}
```

รูปที่ 2.13 ผังงานและซอร์สโค้ดตัวอย่างการทำงานแบบวนซ้ำโดยใช้คำสั่ง while

จากรูปที่ 2.13

เริ่ม กำหนดตัวแปร i , w

กำหนดค่าตัวแปร i = 0;

ลำดับที่ 1 เริ่มต้นการใช้คำสั่ง while โดยตรวจสอบเงื่อนไขว่า i < 11 หรือไม่

ถ้า เงื่อนไขมีค่าเป็นจริง ให้ทำชุดคำสั่ง กำหนดค่าตัวแปร w = (i/2)+1; คือ w=(0/2)+1

และแสดงผลค่า i และ w คือ พิมพ์ 0 1

แล้วจึงเปลี่ยนค่าตัวแปร i = i+2 จะได้ i = 2

แล้ววนกลับไปทำลำดับที่ 2 คือตรวจสอบเงื่อนไขอีกครั้งหนึ่งว่า i < 11 หรือไม่

ถ้า i < 20 ก็จะทำชุดคำสั่ง กำหนดค่าตัวแปร w = (i/2)+1; คือ w=(2/2)+1

และแสดงผลค่า i และ w คือ พิมพ์ 2 2

ทำวนซ้ำจนกว่า เงื่อนไขจะมีค่าเป็นเท็จคือ i มีค่าไม่น้อยกว่า 11 จึงจบการทำงาน

ลำดับการทำงาน	กำหนดตัวแปร	ตรวจสอบเงื่อนไข i < 11	ผลการตรวจสอบเงื่อนไข i < 11	กำหนดค่าตัวแปร	พิมพ์	เปลี่ยนค่าตัวแปร
1	i = 0	0 < 11	จริง	w = (0/2)+1	0 1	i = 2
2		2 < 11	จริง	w = (2/2)+1	2 2	i = 4
3		4 < 11	จริง	w = (4/2)+1	4 3	i = 6
4		6 < 11	จริง	w = (6/2)+1	6 4	i = 8
5		8 < 11	จริง	w = (8/2)+1	8 5	i = 10
6		10 < 11	จริง	w = (10/2)+1	10 6	i = 12
7		12 < 11	เท็จ		END	

จากรูปที่ 2.13 จะได้ผลลัพธ์เป็นการพิมพ์เลข 0 1 2 2 4 3 6 4 8 5 10 6 END

ในตัวโปรแกรมแสดงการเปรียบเทียบวิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรกที่จัดทำนั้นเราจะทำการสุ่มตัวเลขมาใช้เป็นข้อมูลนำเข้า เพื่อให้สะดวกต่อการวิเคราะห์และเปรียบเทียบ โดยไม่ต้องป้อนค่าเองทุกตัวในทุกครั้งที่ทำการใช้โปรแกรม

2.2.3 ฟังก์ชันการสุ่มตัวเลข

การสุ่มตัวเลข คือฟังก์ชันสำหรับการสุ่ม (random) ตัวเลขซึ่งจะได้ผลลัพธ์เป็นเลขจำนวนเต็มมีค่าตั้งแต่ 0 ถึง 32767 หากต้องการสุ่มเลขจำนวนที่มีค่าน้อย หรืออยู่ในช่วงที่กำหนดจะใช้ % (mod) เข้ามาช่วย

% (mod) หรือ modulo คือ การหารเอาเศษ เช่น การหาร 12 ด้วย 7 จะได้ผลลัพธ์คือ 1 และเศษ 5 ในโปรแกรมนั้น ถ้าเราหาค่าของ 12/7 ก็จะได้คำตอบเป็นจำนวนเต็ม คือ 1 เท่านั้น หากอยากได้เศษของการหารต้องใช้ modulo เช่น ถ้าเราหาค่าของ 12%7 จะได้คำตอบเป็นจำนวนเต็ม คือ 5 ซึ่งคือค่าเศษของการหาร

สำหรับการใช้ฟังก์ชัน rand() จะต้องมีการกำหนด #include<stdlib.h> ที่ด้านบนของโปรแกรมด้วย

ตัวอย่างที่ 2.12 การสุ่มตัวเลขโดยใช้ฟังก์ชัน rand()

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
      int A = rand();
      printf("%d",A);
      return 0;
}
```

รูปที่ 2.14 โค้ดตัวอย่างของฟังก์ชัน rand()

จากรูปที่ 2.14 เป็นการกำหนดค่าให้ตัวแปร A โดยการสุ่มค่าโดยใช้ฟังก์ชัน rand() และแสดงผล

รันโปรแกรมครั้งที่ 1 จะได้ผลลัพธ์เป็น 41

รันโปรแกรมครั้งที่ 2 จะได้ผลลัพธ์เป็น 41

รันโปรแกรมครั้งที่ 3 จะได้ผลลัพธ์เป็น 41

รันโปรแกรมครั้งที่ 4 จะได้ผลลัพธ์เป็น 41

จะเห็นว่าค่าตัวแปรจะได้รับการสุ่มออกมาเป็นเลข 41 ทุกครั้ง

การใช้ฟังก์ชัน rand() นี้จะทำการสุ่มเลขออกมาเป็นเลขเดียวกันต่อการรันหนึ่งครั้งเสมอ ดังนั้น หากต้องการสุ่มตัวเลขให้ได้ผลเป็นเลขจำนวนที่ไม่ซ้ำกัน จะต้องกำหนดค่า seed ให้กับฟังก์ชัน rand() ใหม่ โดยใช้ฟังก์ชัน srand() สำหรับการใช้ฟังก์ชัน time() จะต้องมีการกำหนด #include<time.h> ที่ด้านบนของโปรแกรมด้วย

ตัวอย่างที่ 2.13 การสุ่มตัวเลขโดยใช้ฟังก์ชัน rand() และกำหนดค่าเริ่มต้นโดยใช้ฟังก์ชัน srand()

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int main()
{
	srand(time(NULL));

	int A = rand();
	printf("%d",A);
	return 0;
}
```

รูปที่ 2.15 โค้ดตัวอย่างของฟังก์ชัน srand()

จากรูปที่ 2.15 เป็นการกำหนดค่าเริ่มต้นให้กับฟังก์ชัน rand() เพื่อให้สุ่มตัวเลขได้จำนวนไม่ซ้ำกันในแต่ละครั้ง ซึ่งการกำหนดค่า seed นั้นทำโดยการใช้ฟังก์ชันเวลา time(NULL) ซึ่งเป็นค่าที่นำมาจากนาฬิกาภายในของคอมพิวเตอร์ โดยที่เวลานั้นมีการเปลี่ยนแปลงอยู่ตลอด จึงทำให้ค่า seed นั้นจะเปลี่ยนแปลงไปตลอดเช่นเดียวกันด้วย

รันโปรแกรมครั้งที่ 1 จะได้ผลลัพธ์เป็น 25484

รันโปรแกรมครั้งที่ 2 จะได้ผลลัพธ์เป็น 25497

รันโปรแกรมครั้งที่ 3 จะได้ผลลัพธ์เป็น 25572

จะเห็นว่าค่าที่ได้จากการสุ่มตัวเลขจะไม่ซ้ำกันในการรันแต่ละครั้ง

ตัวอย่างที่ 2.14 ถ้าต้องการเลขสุ่ม 0-9 จะเขียน code ได้ดังนี้

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int main()
{
	srand(time(NULL));
	int randomNumber09 = rand()%10;
	printf("%d",randomNumber09);
	return 0;
}
```

รูปที่ 2.16 โค้ดตัวอย่างของการเลขสุ่ม 0-9

จากรูปที่ 2.16 เป็นการกำหนดค่าให้ตัวแปร randomNumber09 โดยค่านั้นเกิดจากการสุ่มตัวเลข แล้วนำตัวเลขนั้นไป %10 หรือคือ นำเลขนั้นไปหารด้วย 10 แล้วเอาแต่เศษ

โดยเมื่อนำตัวเลขใด ๆ มาหารด้วย10 ตัวเศษก็จะมีค่าตั้งแต่ 0 ถึง 9 การหารเอาเศษด้วย 10 จึงเป็นการยืนยันว่าค่าตัวเลขที่สุ่มได้มาจะถูกปรับให้อยู่ในช่วงจำนวนเต็ม 0 ถึง 9 นั่นเอง

รันโปรแกรมครั้งที่ 1 จะได้ผลลัพธ์เป็น 1

รันโปรแกรมครั้งที่ 2 จะได้ผลลัพธ์เป็น 7

รันโปรแกรมครั้งที่ 3 จะได้ผลลัพธ์เป็น 4

รันโปรแกรมครั้งที่ 4 จะได้ผลลัพธ์เป็น 7

รันโปรแกรมครั้งที่ 5 จะได้ผลลัพธ์เป็น 0

จะเห็นว่าค่าที่ได้จากการสุ่มตัวเลขจะไม่ซ้ำกันในการรันแต่ละครั้งและค่าอยู่ในช่วง 0-9 ตามต้องการ

ตัวอย่างที่ 2.15 ถ้าต้องการเลขสุ่ม 50-200 จะเขียน code ได้ดังนี้

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int main()
{
	srand(time(NULL));

	int randomNumber50200 = 50+(rand()%(200-50));
	printf("%d",randomNumber50200);
	return 0;
}
```

รูปที่ 2.17 โค้ดตัวอย่างของการเลขสุ่ม 50-200

จากรูปที่ 2.17 เป็นการกำหนดค่าให้ตัวแปร randomNumber50200 โดยค่านั้นเกิดจากการสุ่มตัวเลข แล้วนำตัวเลขนั้นไป %(200-50) หรือคือ %150 นั่นเอง

ซึ่งก็คือการนำเลขนั้นไปหารด้วย 150 แล้วเอาแต่เศษ ค่าที่ได้ก็จะอยู่ในช่วง 0-150

หรือคือ 0, 1, 2, 3, 4, 5, … , 149, 150

จึงนำไปบวกกับ 50 ค่าที่ได้ก็จะเป็น 0+50, 1+50, 2+50, 3+50, … , 149+50, 150+50

ซึ่งก็คือ 50, 51, 52, 53, 54, 55, … , 199, 200

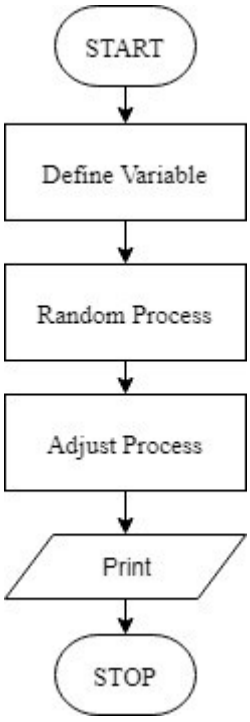
ได้เป็นค่าตัวเลขที่อยู่ในช่วง 50-200 ตามต้องการ

รันโปรแกรมครั้งที่ 1 จะได้ผลลัพธ์เป็น 168

รันโปรแกรมครั้งที่ 2 จะได้ผลลัพธ์เป็น 51

รันโปรแกรมครั้งที่ 3 จะได้ผลลัพธ์เป็น 74

จะเห็นว่าค่าที่ได้จากการสุ่มตัวเลขจะมีค่าอยู่ในช่วง 50-200 ตามต้องการ




```
//---START---
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
int main()
{
      srand(time(NULL));

//--Define Variable--
      int r ;

//---Random---
      r = rand();

//---Adjust---
      r = r%1000 ;

//---Print---
      printf("%d",r);

//---STOP---
  return 0;
}
```

รูปที่ 2.18 ผังงานและซอร์สโค้ดตัวอย่างของฟังก์ชันการสุ่มตัวเลขจำนวนเต็มสามหลัก

จากรูปที่ 2.18

เริ่ม กำหนดตัวแปร r เพื่อรับค่าจากการสุ่ม

แล้วกำหนดค่าตัวแปร r โดยใช้ฟังก์ชัน rand() เพื่อสุ่มค่า

โดย r = rand() ซึ่งจะได้ว่า r มีค่าเป็นจำนวนเต็มจำนวนหนึ่ง เช่น r = 18,467

แล้วนำค่าตัวแปรที่ได้มาปรับตามความต้องการเช่น r = r%1000

จะเป็นการปรับค่า r ให้เป็นจำนวนเต็มที่มีค่าไม่เกิน 1000

ซึ่ง r%1000 นั้นคือการหารเอาเศษ เช่น 18467 % 1000 = 467

จะได้ว่า r = 467 ซึ่งได้ r เป็นจำนวนเต็มสามหลักตามต้องการ
และแสดงผลค่า r ออกทางหน้าจอ
แล้วจึงจบการทำงาน

Asymtotic notation คือ เครื่องหมายที่ใช้อธิบายการเติบโตของฟังก์ชันในการวิเคราะห์อัลกอริทึม จะนำเครื่องหมายนี้มาใช้ในการระบุประสิทธิภาพของอัลกอริทึม โดยในโครงงานนี้ใช้สัญลักษณ์ดังนี้

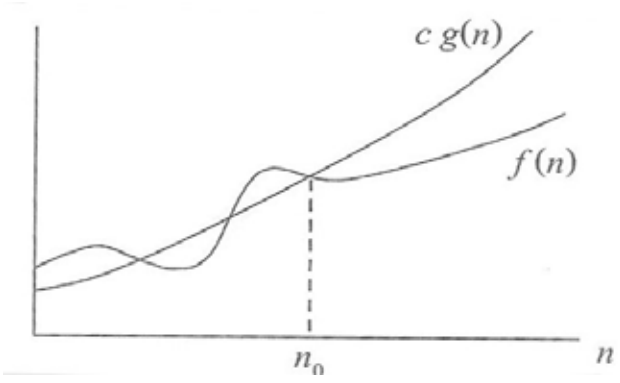
2.3 Big-O Notation : O

บิ๊ก-โอ ใช้ในการระบุเวลาที่ใช้ในการทำงานของอัลกอริทึมเมื่อมีขนาดของอินพุทเปลี่ยนไป คือ ความสัมพันธ์ระหว่าง เวลา กับ ขนาดของอินพุท หรืออินพุทที่ขนาดใดขนาดหนึ่ง เวลาที่ใช้ในการทำงานมากที่สุด (Upper bound) จะเป็นเท่าไร ซึ่งฟังก์ชันบิ๊ก-โอเป็นที่นิยมใช้มากที่สุดในการระบุประสิทธิภาพของอัลกอริทึม

ตัวอย่างที่ 2.16

O(n) คือฟังก์ชันที่ใช้เวลาทำงานช้าที่สุด ≤ n เช่น อัลกอริทึม A มีประสิทธิภาพเป็น $O(n^2)$ ถ้า n = 20 แล้ว ฟังก์ชัน A จะใช้เวลาทำงานช้าที่สุด 400 หน่วยเวลา (อาจจะเร็วกว่า400ได้ แต่ช้าสุดไม่เกิน400)

จะเขียนได้ว่า f(n) **∈** O(g(n)) เพื่อบอกว่า f(n) เป็นฟังก์ชันที่ไม่โตเร็วกว่า g(n)



รูปที่ 2.19 ตัวอย่างกราฟ f(n) และ g(n)

f(n) **∈** O(g(n))

f(n) ≤ (g(n)

สัญกรณ์โอใหญ่ มาตรฐาน	**ชื่อฟังก์ชัน**	**หมายเหตุ**
O(1)	ค่าคงที่	ไม่ใช้ค่าคงที่อื่นในการแสดงสัญกรณ์ เช่น ไม่มีการใช้ O(2)
O(log n)	ลอการิทึม	ลอการิทึมทุกฐานอยู่ในระดับเดียวกัน เพราะเปลี่ยนฐานได้โดยคูณค่าคงที่
$O(k^n)$, 0<k<1	เอกซ์โพเนลเชียล ฐานเศษส่วนแท้	ยิ่งค่าฐานมากยิ่งใหญ่
$O((\log n)^m)$	โพลิลอกาลิทึม	ยิ่งเลขชี้กำลังมากระดับยิ่งใหญ่
$O(n^k)$, 0<k<1	ยกกำลังที่เป็นเศษส่วนแท้ (ติดราก)	ยิ่งเลขชี้กำลังมากระดับยิ่งใหญ่
O(n)	เชิงเส้น	จริงๆแล้วเป็นพหุนามรูปแบบหนึ่ง แยกมาเรียกเพราะใช้บ่อย
$O(n^k)$, k>1	พหุนาม	ยิ่งเลขชี้กำลังมากระดับยิ่งใหญ่
$O(k^n)$, k>1	เอกซ์โพเนลเชียล	ยิ่งค่าฐานมากยิ่งใหญ่
O(n!)	แฟกทอเรียล	อาจรวมถึงการเรียงลำดับสับเปลี่ยน (permutation)
$O(n^n)$	n ยกกำลัง n	มีบางครั้งคนใช้ $O(n^n)$ แทน O(n!) แต่ที่จริง $O(n^n)$ ใหญ่กว่า O(n!) เล็กน้อย

บทที่ 3
วิธีการวิจัย

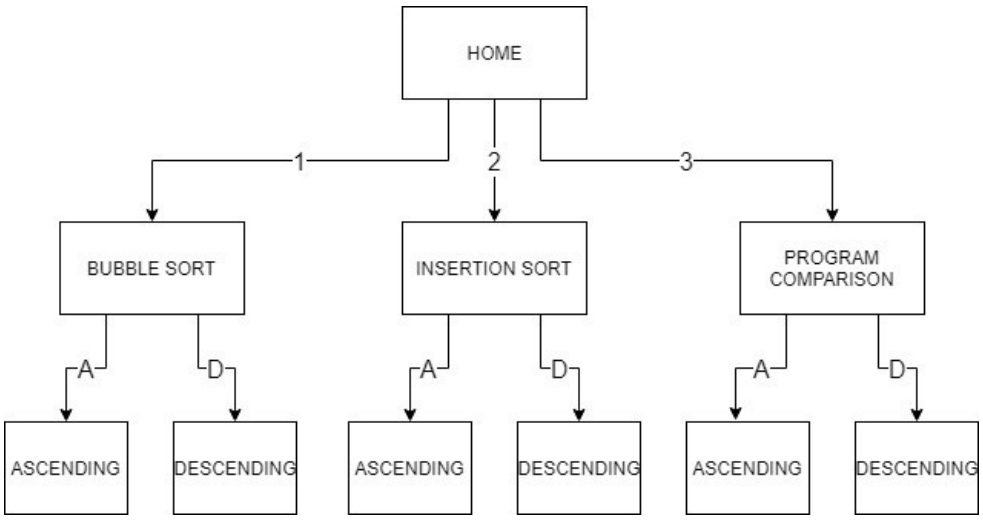
ในบทนี้จะกล่าวถึงขั้นตอนการทำโครงงาน ซึ่งจะประกอบไปด้วย ออกแบบระบบของโครงงาน ออกแบบและเขียนโปรแกรมการเรียงลำดับแบบฟองและแบบแทรก พร้อมทั้งการเปรียบเทียบผลที่ได้

3.1 การออกแบบระบบ

ในโปรแกรมนี้ประกอบด้วย โปรแกรมย่อย 3 โปรแกรม คือ

1. โปรแกรมแสดงการจัดเรียงลำดับแบบฟอง
2. โปรแกรมแสดงการจัดเรียงลำดับแบบแทรก
3. โปรแกรมแสดงการเปรียบเทียบการจัดเรียงแบบฟองและแบบแทรก

กำหนดให้แต่ละโปรแกรมถามผู้ใช้ว่าต้องการเรียงลำดับเลขจำนวนกี่จำนวน (n) แล้วโปรแกรมจะสุ่มตัวเลขจำนวเต็มไม่เกิน 3 หลัก จำนวนตามที่ผู้ใช้ต้องการ เพื่อนำมาเรียงลำดับตามความต้องการของผู้ใช้ โดยผู้ใช้ เลือก A ถ้าต้องการเรียงลำดับจากน้อยไปหามาก (Ascending) และเลือก D ถ้าต้องการเรียงลำดับจากมากไปหาน้อย (Descending) ดังรูป



รูปที่ 3.1 ผังงานแสดงการออกแบบโปรแกรม

3.2 โปรแกรมแสดงการจัดเรียงลำดับข้อมูลแบบฟอง

กำหนดตัวแปร	data[ ]	:	ตัวแปรชุดสำหรับเก็บข้อมูลที่ต้องการเรียงลำดับ
	i , j	:	ตัวแปรแทนดัชนีของข้อมูล data
	swap	:	ตัวแปรชั่วคราวที่ใช้ช่วยในการสลับค่าของข้อมูล
ข้อมูลนำเข้า	n	:	จำนวนข้อมูลทั้งหมดในตัวแปร data
ผลลัพธ์	data[ ]	:	ตัวแปรชุดที่เก็บข้อมูลที่จัดเรียงแล้ว

```
//--BUBBLE SORT--
//----START----

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
int main()
{

//----Define Variable---

  int data[1000]={};     //…(i)
  int n=0, i=0, j=0, swap=0;   //…(ii)


//----Set Seed of function rand()----

      srand(time(NULL));


//----Input----

  printf(">Enter amount of number\n");
  scanf("%d", &n);


//----Random Process----

  for (i = 0; i < n ; i++)     //…(iii)
  {
      data[i] = rand() %1000;
  }

//----Print array data----
  printf("\n>DATA IN: ");
  for (i = 0; i < n ; i++)
     printf("   %d", data[i]);


//----Sorting Process----

  for (i = 0 ; i < ( n - 1 ); i++)   //…(iv)
  {
    for (j = 0 ; j < n - i - 1; j++)       //…(v)
```

ใช้ // นำ ข้างหลังจะเป็นหมายเหตุ ไม่เอามาคำนวณ

ฟังก์ชัน srand(TIME(NULL)) ใช้กำหนดค่าเริ่มต้นของการสุ่ม เพื่อให้การรันแต่ละครั้งได้ข้อมูลที่ไม่ซ้ำกัน

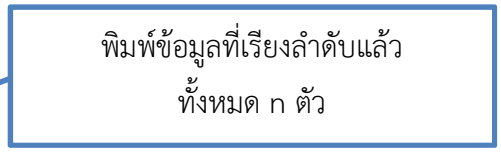
รับค่าตัวเลขจำนวนข้อมูลที่ต้องการเรียงลำดับมาเก็บในตัวแปร n เพื่อเป็นการกำหนดจำนวนของตัวแปร data[]

สุ่มข้อมูลตัวเลขจำนวน n ตัว โดยแต่ละตัวจะเป็นเลขจำนวนเต็มมีค่าไม่เกินสามหลัก

พิมพ์ข้อมูลเริ่มต้นที่สุ่มได้ทั้งหมด n ตัว

ขั้นตอนการเรียงลำดับแบบฟอง

```
    {
      if (data[j] > data[j+1])       //…(vi)
      {
          swap       =   data[j];
          data[j]     =   data[j+1];
          data[j+1]   =   swap;
      }
    }
  }


//----Print Output----

   printf("\n>DATA OUT: "); //…(vii)
  for (i = 0; i < n ; i++)
     printf("   %d", data[i]);


//----STOP----

  return 0;
}
```



รูปที่ 3.2 ซอร์สโค้ดโปรแกรมแสดงการจัดเรียงลำดับข้อมูลแบบฟอง

จากรูปที่ 3.2 ซอร์สโค้ดโปรแกรมแสดงการจัดเรียงลำดับข้อมูลแบบฟอง

เริ่มการทำงาน

(i) กำหนดค่าตัวแปร data[1000]={} เพื่อเป็นการจองพื้นที่ไว้เป็นจำนวน 1000 ช่อง และทำให้ค่าเริ่มต้นในตัวแปรเป็นศูนย์ทั้งหมด เพื่อกันการมีค่าข้อมูลตกค้าง

(ii) กำหนดค่าตัวแปร n=0 , i=0 , j=0 , swap=0 เพื่อให้ค่าเริ่มต้นเป็น 0

(iii) ทำการสุ่มค่าให้ตัวแปร data[i] เป็นจำนวน n ตัว โดยใช้ data[i] = rand() %1000 เพื่อปรับค่า data[i] ให้เป็นจำนวนเต็มที่มีค่าไม่เกินสามหลัก

(iv) แสดงผล data[] ที่ได้รับการสุ่มค่าแล้วออกทางหน้าจอเพื่อให้ผู้ใช้สามารถตรวจสอบ

(v) ใช้การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for ซึ่งกำหนดค่า i = 0 และเปรียบเทียบกับเงื่อนไข i < n-1 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นจริง จะใช้การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for ซึ่งกำหนดค่า j = 0 และเปรียบเทียบกับเงื่อนไข j < n-i-1 ว่าเป็นจริงหรือเท็จ

(vi) ถ้าเป็นจริง จะทำการตรวจสอบเงื่อนไข data[j] > data[j+1] ว่าเป็นจริงหรือเท็จ

ถ้าเป็นจริงจะทำการสลับที่ค่าของ data[j] กับ data[j+1]

ถ้าเป็นเท็จ จะไม่เกิดการสลับที่ แล้วจึงให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ j++; แล้วจึงแล้ววนกลับไปตรวจสอบเงื่อนไข j < n-i-1 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นเท็จ จะให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ i++; แล้วจึงแล้ววนกลับไปตรวจสอบเงื่อนไข i < n-1 ว่าเป็นจริงหรือเท็จ

ทำวนซ้ำจนกว่า เงื่อนไขจะมีค่าเป็นเท็จ จึงจบการทำงานแบบวนซ้ำ

(vii) แสดงผลลัพธ์ของ data[] ที่ได้รับการจัดเรียงเรียบร้อยแล้วออกทางหน้าจอ

จึงจบการทำงาน

3.3 โปรแกรมแสดงการจัดเรียงแบบแทรก

กำหนดตัวแปร	data[ ]	: ที่เก็บข้อมูลที่ต้องการเรียงลำดับ
	i , j	: ตัวแปรแทนดัชนีของข้อมูล data
	swap	: ตัวแปรชั่วคราวใช้ช่วยในการสลับค่า
ข้อมูลนำเข้า	n	: จำนวนข้อมูลทั้งหมดในตัวแปร data
ผลลัพธ์	data[ ]	: ข้อมูลที่จัดเรียงแล้ว

```
//--INSERTION SORT--
//----START----

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
int main()
{

//----Define Variable---

  int data[1000]={}, n=0, i=0, j=0, swap=0; //…(i)


//----Set Seed of function rand()----

      srand(time(NULL)); //…(ii)


//----Input----

  printf(">Enter amount of number\n");
  scanf("%d", &n);


//----Random Process----

  for (i = 0; i < n ; i++)
  {
      data[i] = rand() %1000; //…(iii)
  }

//----Print Array Data----
  printf("\n>DATA IN: ");
```

ใช้ฟังก์ชัน srand(TIME(NULL)) เพื่อเป็นการกำหนดค่าเริ่มต้นของการสุ่มให้ไม่ซ้ำในการรันแต่ละครั้ง

รับค่าและนำเก็บในตัวแปร n เพื่อเป็นการกำหนดจำนวนของตัวแปร data[]

```
  for (i = 0; i < n ; i++)
     printf("   %d", data[i]);


//----Sorting Process----

  for (i = 1 ; i <= ( n - 1 ); i++) //…(iv)
  {
    for ( j=i ; j > 0 ; j-- )
    {
      if (data[j] < data[j-1]) //…(v)
      {
          swap        =   data[j];
          data[j]     =   data[j-1];
          data[j-1]   =   swap;
      }
      else
          break;
    }
  }


//----Print Output---- //…(vi)

   printf("\n>DATA OUT: ");
  for (i = 0; i < n ; i++)
     printf("   %d", data[i]);


//----STOP----

  return 0;
}
```

รูปที่ 3.3 ซอร์สโค้ดโปรแกรมแสดงการจัดเรียงแบบแทรก

จากรูปซอร์สโค้ดที่ 3.3

เริ่มการทำงาน

(i) กำหนดค่าตัวแปร data[1000]={} เพื่อเป็นการจองพื้นที่ไว้เป็นจำนวน 1000 ช่อง และเคลียร์ค่าตัวแปร และกำหนดค่าตัวแปร n=0 , i=0 , j=0, swap=0 เพื่อให้ค่าเริ่มต้นเป็น 0

(ii) ทำการสุ่มค่าให้ตัวแปร data[] เป็นจำนวน n ตัว โดยใช้ฟังก์ชั่น rand() และใช้การ % 1000 (mod 1000) เพื่อปรับค่า data[] ให้เป็นจำนวนเต็มที่มีค่าไม่เกิน 1000

(iii) แสดงผล data[] ที่ได้รับการสุ่มค่าแล้วออกทางหน้าจอเพื่อให้ผู้ใช้สามารถตรวจสอบ

(iv) ใช้การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for ซึ่งกำหนดค่า i = 1 และเปรียบเทียบกับเงื่อนไข i <= n-1 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นจริง จะใช้การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for ซึ่งกำหนดค่า j = i และเปรียบเทียบกับเงื่อนไข j > 0 ว่าเป็นจริงหรือเท็จ

(v) ถ้าเป็นจริง จะทำการตรวจสอบเงื่อนไข data[j] < data[j-1] ว่าเป็นจริงหรือเท็จ
ถ้าเป็นจริงจะทำการสลับที่ค่าของ data[j] กับ data[j+1]
ถ้าเป็นเท็จ จะไม่เกิดการสลับที่ แล้ว break ออกจากการทำงานแบบวนซ้ำ
แล้วจึงให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ j--; แล้วจึงแล้ววนกลับไปตรวจสอบ
เงื่อนไข j > 0 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นเท็จ จะให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ i++; แล้วจึงแล้ววนกลับไปตรวจสอบ
เงื่อนไข i <= n-1 ว่าเป็นจริงหรือเท็จ
ทำวนซ้ำจนกว่า เงื่อนไขจะมีค่าเป็นเท็จ จึงจบการทำงานแบบวนซ้ำ

(vi) แสดงผลลัพธ์ของ data[] ที่ได้รับการจัดเรียงเรียบร้อยแล้วออกทางหน้าจอ
จึงจบการทำงาน

3.4 โปรแกรมแสดงการเปรียบเทียบวิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก

กำหนดตัวแปร	datab[ ], datai[]	: ที่เก็บข้อมูลที่ต้องการเรียงลำดับ
	i , j	: ตัวแปรแทนดัชนีของข้อมูล data
	swap	: ตัวแปรชั่วคราวใช้ช่วยในการสลับค่า
	bcompare , icompare	: ตัวแปรเก็บจำนวนครั้งที่ข้อมูลเกิดการเปรียบเทียบ
	bsort , isort	: ตัวแปรเก็บจำนวนครั้งที่ข้อมูลเกิดการสลับที่
ข้อมูลนำเข้า	n	: จำนวนข้อมูลทั้งหมดในตัวแปร data
ผลลัพธ์	datab[ ], datai[]	: ข้อมูลที่จัดเรียงแล้ว

```
//-- PROGRAM COMPARISION--
//----START----

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
int main()
{

-Define Variable---

  int datab[1000]={}, datai[1000]={}, n=0, i=0, j=0, swap=0; //…(i)
  int bcompare=0, icompare=0, bsort=0, isort=0;

//----Set Seed of function rand()----

      srand(time(NULL));


//----Input----
```

ใช้ฟังก์ชัน srand(TIME(NULL)) เพื่อเป็นการกำหนดค่าเริ่มต้นของการสุ่มให้ไม่ซ้ำในการรันแต่ละครั้ง

```
  printf(">Enter amount of number\n");
  scanf("%d", &n);


//----Random Process----

  for (i = 0; i < n ; i++) //…(ii)
  {
      datab[i] = rand() %1000;
      datai[i] = datab[i];
  }

//----Print array data----

     printf("\n>DATA BUBBLE IN: "); //…(iii)
  for (i = 0; i < n ; i++)
     printf("   %d", datab[i]);

   printf("\n>DATA INSERTION IN: ");
  for (i = 0; i < n ; i++)
     printf("   %d", datai[i]);



//----Bubble Sorting Process----

  for (i = 0 ; i < ( n - 1 ); i++) //…(iv)
  {
    for (j = 0 ; j < n - i - 1; j++)
    {
      bcompare++;
      if (datab[j] > datab[j+1])
      {
          bsort++;

          swap       =   datab[j];
          datab[j]     =   datab[j+1];
          datab[j+1]   =   swap;
      }
    }
  }


//----Insertion Sorting Process----

  for (i = 1 ; i <= ( n - 1 ); i++) //…(v)
  {
    for ( j=i ; j > 0 ; j-- )
    {
      icompare++;
      if (datai[j] < datai[j-1])
      {
          isort++;

          swap        =   datai[j];
          datai[j]     =   datai[j-1];
          datai[j-1]   =   swap;
      }
      else
```

รับค่าและนำเก็บในตัวแปร n เพื่อเป็นการกำหนดจำนวนของตัวแปร data[]

```
            break;
      }
   }

//----Print Output----

    printf("\n>DATA BUBBLE OUT: "); //…(vi)
   for (i = 0; i < n ; i++)
      printf("   %d", datab[i]);

    printf("\n>DATA INSERTION OUT: ");
   for (i = 0; i < n ; i++)
      printf("   %d", datai[i]);

    printf("\n>Bubble Compare: %d ",bcompare);
    printf("\n>Bubble Sort: %d ",bsort);
    printf("\n>Insertion Compare: %d ",icompare);
    printf("\n>Insertion Sort: %d ",isort);



//----STOP----

   return 0;
}
```

รูปที่ 3.4 ซอร์สโค้ดโปรแกรมแสดงการเปรียบเทียบ

จากรูปซอร์สโค้ดที่ 3.4

(i) เริ่ม กำหนดค่าตัวแปร datab[1000]={} , datai[1000]={} เพื่อเป็นการจองพื้นที่ไว้เป็นจำนวน 1000 ช่องเพื่อใช้สำหรับการจัดเรียงแบบฟองและแบบแทรก และเคลียร์ค่าตัวแปร และกำหนดค่าตัวแปร n=0 , i=0 , j=0 , swap=0 , bcompare=0, icompare=0, bsort=0, isort=0 เพื่อให้ค่าเริ่มต้นเป็น 0

(ii) ทำการสุ่มค่าให้ตัวแปร datab[] เป็นจำนวน n ตัว โดยใช้ฟังก์ชั่น rand() และใช้การ % 1000 (mod 1000) เพื่อปรับค่า datab[] ให้เป็นจำนวนเต็มที่มีค่าไม่เกิน 1000

และทำการกำหนดให้ตัวแปร datai[] มีค่าเท่ากับตัวแปร datab[] เพื่อที่จะสามารถนำค่า datai[] ไปใช้ในการจัดเรียงแบบแทรกได้ ซึ่งเป็นข้อมูลชุดเดียวกันจึงจะเกิดการเปรียบเทียบได้

(iii) แสดงผล datab[] และ datai[] ที่ได้รับการสุ่มค่าแล้วออกทางหน้าจอเพื่อให้ผู้ใช้สามารถตรวจสอบ

(iv) เริ่มการเรียงลำดับแบบฟอง ใช้การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for ซึ่งกำหนดค่า i = 0 และเปรียบเทียบกับเงื่อนไข i < n-1 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นจริง จะใช้การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for ซึ่งกำหนดค่า j = 0 และเปรียบเทียบกับเงื่อนไข j < n-i-1 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นจริง จะให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ bcompare++;

และจะทำการตรวจสอบเงื่อนไข datab[j] > datab[j+1]

ถ้าเป็นจริงจะทำการสลับที่ค่าของ datab[j] กับ datab[j+1]

และ จะให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ bsort++;

ถ้าเป็นเท็จ จะไม่เกิดการสลับที่ แล้วจึงให้เปลี่ยนค่าตัวแปรคือ j++; แล้วจึง

แล้ววนกลับไปตรวจสอบเงื่อนไข j < n-i-1 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นเท็จ จะให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ i++; แล้วจึงแล้ววนกลับไปตรวจสอบเงื่อนไข i < n-1 ว่าเป็นจริงหรือเท็จ

ทำวนซ้ำจนกว่า เงื่อนไขจะมีค่าเป็นเท็จ จึงจบการทำงานแบบวนซ้ำ

(v) เริ่มการเรียงลำดับแบบแทรก ใช้การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for ซึ่งกำหนดค่า i = 1 และเปรียบเทียบกับเงื่อนไข i <= n-1 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นจริง จะใช้การทำงานแบบวนซ้ำ โดยใช้คำสั่ง for ซึ่งกำหนดค่า j = i และเปรียบเทียบกับเงื่อนไข j > 0 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นจริง จะให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ icompare++;

และจะทำการตรวจสอบเงื่อนไข datai[j] < datai[j-1] ว่าเป็นจริงหรือเท็จ

ถ้าเป็นจริงจะทำการสลับที่ค่าของ datai[j] กับ datai[j+1]

และ จะให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ bsort++;

ถ้าเป็นเท็จ จะไม่เกิดการสลับที่ แล้ว break ออกจากการทำงานแบบวนซ้ำ

แล้วจึงให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ j--; แล้วจึงแล้ววนกลับไปตรวจสอบเงื่อนไข j > 0 ว่าเป็นจริงหรือเท็จ

ถ้าเป็นเท็จ จะให้ทำชุดคำสั่งเปลี่ยนค่าตัวแปรคือ i++; แล้วจึงแล้ววนกลับไปตรวจสอบเงื่อนไข i <= n-1 ว่าเป็นจริงหรือเท็จ

ทำวนซ้ำจนกว่า เงื่อนไขจะมีค่าเป็นเท็จ จึงจบการทำงานแบบวนซ้ำ

(vi) แสดงผล datab[] และ datai[] ที่ได้รับการจัดเรียงเรียบร้อยแล้วออกทางหน้าจอ และแสดงผล bcompare, icompare, bsort, isort เพื่อเปรียบเทียบจำนวนครั้งของการเปรียบเทียบข้อมูลและจำนวนครั้งของการสลับที่ข้อมูล ของการจัดเรียงแบบฟองและแบบแทรก

แล้วจึงจบการทำงาน

บทที่ 4
ข้อสรุปและข้อเสนอแนะ

โครงงานนี้มีวัตุประสงค์เพื่อวิเคราะห์และนำเสนอวิธีการจัดเรียงลำดับข้อมูลทั้ง วิธีการจัดเรียงลำดับข้อมูลแบบฟอง และวิธีการจัดเรียงลำดับข้อมูลแบบแทรก พร้อมทั้งเปรียบเทียบและวิเคราะห์ข้อดีและข้อด้อยของทั้งสองวิธีนี้ ด้วยโปรแกรมคอมพิวเตอร์

4.1 ข้อสรุป

จากข้อมูลของโปรแกรม ทำให้ทราบว่าสำหรับข้อมูลชุดเดียวกัน นำมาจัดเรียงลำดับโดยวิธีการจัดเรียงลำดับข้อมูลแบบฟองและวิธีการจัดเรียงลำดับข้อมูลแบบแทรก ทั้งสองวิธีมีจำนวนครั้งของการสลับที่เท่ากัน และการจัดเรียงลำดับข้อมูลแบบแทรกจะมีจำนวนครั้งในการเปรียบเทียบน้อยกว่าการจัดเรียงลำดับข้อมูลแบบฟอง

การจัดเรียงลำดับข้อมูลแบบฟอง (Bubble Sort) และ การจัดเรียงลำดับข้อมูลแบบแทรก (Insertion Sort) มีประสิทธิภาพในการจัดเรียงลำดับแตกต่างกันไปในแต่กรณีของข้อมูลแต่ละชุด ดังนี้

วิธีการเรียงลำดับข้อมูล	กรณีดีที่สุด (Best-case)	กรณีทั่วไป (Average-case)	กรณีแย่ที่สุด (Worst-case)
Bubble Sort	O(n)	$O(n^2)$	$O(n^2)$
Insertion Sort	O(n)	$O(n^2)$	$O(n^2)$

เช่น n=5 ถ้าข้อมูลอยู่ในกรณีทั่วไป การจัดเรียงลำดับข้อมูลแบบฟอง และ การจัดเรียงลำดับข้อมูลแบบแทรก จะมีประสิทธิภาพอยู่ที่ $O(5^2)$ คือ อาจจะไวกว่า 25 หน่วยเวลา แต่ช้าสุดไม่เกิน 25 เวลา

จากตัวอย่างที่ 2.3 และ 2.9 กรณีชุดข้อมูลเดียวกัน จำนวนเท่ากัน เรียงลำดับจากมากไปน้อยเหมือนกัน จะได้ว่า การจัดเรียงลำดับข้อมูลแบบแทรกนั้นมีจำนวนครั้งของการเปรียบเทียบน้อยกว่าการจัดเรียงลำดับข้อมูลแบบฟอง ในตัวอย่างที่ 2.3 การจัดเรียงแบบฟองนั้นเกิดการเปรียบเทียบข้อมูลทั้งหมด 10 ครั้ง ในตัวอย่างที่ 2.9 การจัดเรียงแบบแทรกนั้นเกิดการเปรียบเทียบข้อมูลทั้งหมด 9 ครั้ง

ข้อแตกต่างเห็นได้ชัดในกรณีที่ชุดข้อมูลอยู่ในกรณีดีที่สุด ถ้าชุดข้อมูลเดียวกัน จำนวน n ตัวเท่ากัน เรียงลำดับแบบเดียวกัน จากสูตรการคำนวณ จำนวนในการเปรียบเทียบรวมทั้งหมดของการเรียงลำดับข้อมูลแบบฟองจะมีค่าเท่ากับ n(n-1)/2 แต่จำนวนในการเปรียบเทียบรวมทั้งหมดของการเรียงลำดับข้อมูลแทรกจมีค่าเท่ากับ (n-1)

4.2 ข้อเสนอแนะ

จากการได้ศึกษาวิธีการเรียงลำดับข้อมูลแบบฟองและแบบแทรกแล้ว ได้ทราบว่าวิธีการเรียงลำดับข้อมูลสองวิธีนี้นั้นมีความเรียบง่ายในตัวเอง ง่ายต่อการทำความเข้าใจ และง่ายต่อการนำมาใช้งาน แต่เป็นที่แน่นอนว่าการเรียงลำดับข้อมูลที่เรียบง่ายนั้นมีประสิทธิภาพต่ำกว่าวิธีการเรียงลำดับข้อมูลแบบอื่นที่ซับซ้อนมากกว่า

แต่ในวิธีการเรียงลำดับข้อมูลแบบอื่นที่ให้ประสิทธิภาพที่ดีกว่า อาจจะช่วยให้ตัวโปรแกรมทำงานได้เร็วขึ้น แต่ก็มีความซับซ้อนมากกว่า ต้องใช้เวลาในการทำความใจมากกว่าการเรียงลำดับข้อมูลที่เรียบง่ายและเข้าใจได้ง่าย ในบางกรณีเราอาจจำเป็นต้องใช้งานอัลกอริทึมที่เรียบง่ายต่อการศึกษาหรือปรับปรุง แต่ก็ต้องแลกมาด้วยกับการที่โปรแกรมจะทำงานช้ากว่า ใช้ทรัพยากรมากกว่า

ดังนั้นประสิทธิภาพอาจไม่ได้เป็นสิ่งเดียวที่เราต้องการเสมอไป เพราะว่าในทุกเรื่องต่างมีความสำคัญพอๆกัน ขึ้นอยู่กับสภาพแวดล้อมและความเหมาะสม

รายการอ้างอิง

1. TED-Ed, An introduction to sorting algorithms (วันที่ 13 มกราคม 2561)
https://www.facebook.com/TEDEducation/videos/1369636223049580/

2. สถาบันส่งเสริมการสอนวิทยาศาสตร์และเทคโนโลยี, หลักสูตรโปรแกรมภาษา C เบื้องต้น (วันที่ 15 มีนาคม 2561)
https://www.programming.in.th/tutorial/index.php

3. โปรแกรมเมอร์ตัวน้อย, สอนจาวา (Java) ตอนที่ 1-6 แบบ Basic คัดพิเศษ (วันที่ 8 สิงหาคม 2561)
https://www.facebook.com/Programmerbaby/posts/140936066107108

4. สมชาย ประสิทธิ์จูตระกูล, การเรียงลำดับข้อมูล (วันที่ 9 ตุลาคม 2561)
https://www.cp.eng.chula.ac.th/~somchai/ULearn/DataStructures/Topics/index_ch14.htm

5. สำนักงานวิทยบริการและเทคโนโลยีสารสนเทศ มหาวิทยาลัยราชภัฏสงขลา, บทที่ 8 การเรียงลำดับข้อมูล (วันที่ 15 ตุลาคม 2561)
http://oservice.skru.ac.th/ebookft/459/chapter%208%20%A1%D2%C3%E0%C3%D5%C2%A7%C5%D3%B4%D1%BA%A2%E9%CD%C1%D9%C5.pdf

6. ทศพล ธนะทิพานนท์ และ วรเศรษฐ สุวรรณิก, เขียนโปรแกรม JAVA เบื้องต้น, ซีเอ็ดยูเคชั่น, บมจ. 2549

7. อรพิน ประวัติบริสุทธิ์, คู่มือเขียนโปรแกรมด้วย ภาษา C ฉบับสมบูรณ์ (ปรับปรุงใหม่), ซีเอ็ดยูเคชั่น , บมจ. 2559

ภาคผนวก

ภาคผนวก ก

แบบเสนอหัวข้อโครงงาน Project Proposal

ปีการศึกษา 2559

ชื่อโครงงาน (ภาษาไทย)	โปรแกรมแสดงการเปรียบเทียบวิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก
ชื่อโครงงาน (ภาษาอังกฤษ)	Program Display Comparison Of Bubble Sort And Insertion Sort
อาจารย์ที่ปรึกษา	รองศาสตราจารย์ จิตรจวบ เปาอินทร์
	ผู้ช่วยศาสตราจารย์ วาสนา สุขกระสานติ
ผู้ดำเนินงาน	นายชัยฤทธิ์ ขาวผ่อง รหัสประจำตัวนิสิต 5533560423
	สาขาวิชาคณิตศาสตร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
	คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

หลักการและเหตุผล

ปัจจุบันนี้เป็นยุคของข้อมูลข่าวสาร ซึ่งแน่นอนว่า ข้อมูลมีจำนวนมากขึ้นเรื่อยๆ การทำงานเรียงลำดับข้อมูล เป็นวิธีการที่มีบทบาทในการจัดการข้อมูลอย่างยิ่ง มีนักคณิตศาสตร์ นักเทคโนโลยีสารสนเทศ คิดค้นวิธีการเรียงลำดับมากมาย แต่ละวิธีก็มีข้อดีแตกต่างกันไป

ผู้ดำเนินโครงการจึงสนใจที่จะนำเสนอขั้นตอนการเรียงลำดับข้อมูล โดยเลือกวิธีการเรียงลำดับมา 2 วิธี แล้วอธิบายขั้นตอนการเรียงลำดับข้อมูลของแต่ละวิธีที่ได้เลือกมานั้น ผ่านจอภาพที่มีรูปภาพสีสันให้ผู้สนใจสามารถเข้าใจได้โดยง่าย พร้อมทั้งแสดงการเปรียบเทียบ ข้อดี ข้อด้อย ของทั้งสองวิธีนั้น

จุดประสงค์

โครงงานนี้มีวัตถุประสงค์เพื่อวิเคราะห์และนำเสนอ การแสดงการจัดเรียงข้อมูล วิธีเรียงข้อมูลแบบฟอง (Bubble sort) และวิธีเรียงข้อมูลแบบแทรก (Insertion sort) พร้อมทั้งเปรียบเทียบและวิเคราะห์ข้อดีและข้อด้อยของทั้งสองวิธีนี้ ด้วยโปรแกรมผ่านภาษาคอมพิวเตอร์

ขอบเขตของโครงงาน

โครงงานนี้จะแสดงการจัดเรียงข้อมูลสองวิธี คือ วิธีเรียงข้อมูลแบบฟอง และวิธีเรียงข้อมูลแบบแทรก โดยนำเสนอในลักษณะของภาพเคลื่อนไหว โดยใช้ข้อมูลไม่น้อยกว่า 20 ตัว ผ่านโปรแกรม ด้วยภาษาซี หรือ จาวา พร้อมทั้งเปรียบเทียบผลการเรียงลำดับของแต่ละวิธีด้วย

โปรแกรมนี้เหมาะสำหรับผู้ที่สามารถใช้งานคอมพิวเตอร์ในระบบไมโครซอฟท์ วินโดวส์ได้เป็นอย่างดี

วิธีการดำเนินงาน

1. ศึกษาความรู้เกี่ยวกับ วิธีการจัดเรียงข้อมูลในแบบต่างๆ
2. ศึกษาความรู้พื้นฐานเกี่ยวกับ ภาษาซี และ จาวา
3. ศึกษาความรู้และความสัมพันธ์ของ วิธีการจัดเรียงข้อมูล
4. เขียนโปรแกรมแสดงการจัดเรียงข้อมูล
5. วิเคราะห์และทดสอบโปรแกรมการจัดเรียงข้อมูล
6. สรุปผลและเขียนรายงาน

ระยะเวลาการทำงาน

ขั้นตอนการดำเนินการ	พฤศจิกายน 2559 - เมษายน 2560					
	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.
1. ศึกษาความรู้เกี่ยวกับ การจัดเรียงข้อมูลในแบบต่าง ๆ						
2. ศึกษาความรู้พื้นฐานเกี่ยวกับ ภาษาซี และ จาวา						
3. ศึกษาความรู้และความสัมพันธ์ของวิธีการจัดเรียงข้อมูล						
4. เขียนโปรแกรมแสดงการจัดเรียงข้อมูล						
5. วิเคราะห์และทดลองโปรแกรมการจัดเรียงข้อมูล						
6. สรุปผลและเขียนรายงาน						

ประโยชน์ที่คาดว่าจะได้รับ

สำหรับผู้ดำเนินงาน

1. ได้ความรู้พื้นฐานเกี่ยวกับ การจัดเรียงข้อมูล และการเขียนโปรแกรมด้วยภาษาซีหรือจาวา
2. ได้ศึกษาและทำความเข้าใจเกี่ยวกับการเรียงลำแบบฟอง และการเรียงลำดับแบบแทรก
3. ได้โปรแกรมนำเสนอ ขั้นตอนและวิธีการของการจัดเรียงข้อมูลที่ต่างกัน สองวิธี

สำหรับผู้ใช้งาน

1. ได้ศึกษาและทำความเข้าใจเกี่ยวกับการเรียงลำแบบฟอง และการเรียงลำดับแบบแทรก

อุปกรณ์และเครื่องมือที่ใช้

1. เครื่องคอมพิวเตอร์ Dell รุ่น Inspiron 7447, Intel Core i7, RAM 8 GB, HDD 1 TB
2. เครื่องพิมพ์ HP

เอกสารอ้างอิง

1. TED-Ed, An introduction to sorting algorithms (วันที่ 30 พฤศจิกายน 2559)
   https://www.facebook.com/TEDEducation/videos/1369636223049580/
2. สถาบันส่งเสริมการสอนวิทยาศาสตร์และเทคโนโลยี, หลักสูตรโปรแกรมภาษา C เบื้องต้น (วันที่ 15 ธันวาคม 2559)
   https://www.programming.in.th/tutorial/index.php
3. เว็บไทยครีเอท, Java Programming (วันที่ 9 มกราคม 2560)
   http://www.thaicreate.com/java.html
4. ทศพล ธนะทิพานนท์,และ วรเศรษฐ สุวรรณิก, เขียนโปรแกรม JAVA เบื้องต้น, ซีเอ็ดยูเคชั่น, บมจ. 2549
5. อรพิน ประวัติบริสุทธิ์, คู่มือเขียนโปรแกรมด้วย ภาษา C ฉบับสมบูรณ์ (ปรับปรุงใหม่),ซีเอ็ดยูเคชั่น, บมจ. 2559

ภาคผนวก ข
ซอร์สโค้ดโปรแกรมแสดงการเปรียบเทียบ
วิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก

```
/*All Program */

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <conio.h>


#include <windows.h>
#define BLACK         0
#define DARKBLUE      1
#define DARKGREEN     2
#define AQUA          3
#define DARKRED       4
#define DARKVOILET    5
#define DARKYELLOW    6
#define GRAY          7
#define DARKGRAY      8
#define BLUE          9
#define GREEN         10
#define CYAN          11
#define RED           12
#define VIOLET        13
#define YELLOW        14
#define WHITE         15

#define textcolor(txt,back)
SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), back*16+txt)
#define resetcolor()
SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15)

void insertion()
{
  int num[1000],keep[1000], n=0, i=0, j=0, k=0 , s=0, bold=0, sort=0, round=0
,ip=0,l=0,m=0,count=0,check=0,icheck=0,icount=0;
 int checkinput=0, checksort=0, checkspace=0;

      srand(time(NULL));
      int r = rand() %100;
      char c;

      textcolor(WHITE,BLACK);
```

```
printf("-------------------------------------------------");
printf("\n\n");
printf("            I N S E R T I O N     S O R T");
printf("\n\n");
printf("-------------------------------------------------");
printf("\n\n");

textcolor(GRAY,BLACK);
printf(">Enter amount of number\n\n> ");
 textcolor(WHITE,BLACK);
scanf("%d", &n);
printf("\n\n");

printf("\n\n");
textcolor(GRAY,BLACK);
printf(">Choose how to input data 1 by random or 2 by yourself\n\n ");
 textcolor(WHITE,BLACK);
scanf("%d", &checkinput);
printf("\n\n");

printf("\n\n");
textcolor(GRAY,BLACK);
printf(">Choose how to sort 1 ascending or 2 descending\n\n ");
 textcolor(WHITE,BLACK);
scanf("%d", &checksort);
printf("\n\n");

printf("\n\n");
textcolor(GRAY,BLACK);
printf(">Choose when to pause 1 step by step or 2 no pause\n\n ");
 textcolor(WHITE,BLACK);
scanf("%d", &checkspace);
printf("\n\n");


if (checkinput == 2)
{
    printf("\n\n");
    textcolor(GRAY,BLACK);
    printf(">Enter input data %d numbers \n\n > ",n);
 textcolor(WHITE,BLACK);
    printf("\n\n");
}


for (i = 0; i < n ; i++)
{
    if (checkinput == 1)
    {
          num[i] = rand() %1000;
          keep[i] = num[i];
    }
    else
    {
```

```
            scanf("%d", &num[i]);
      }
 }
  printf("\n>Input: ");
textcolor(WHITE,BLACK);
 for (i = 0; i < n ; i++)
    printf("   %d", num[i]);

   scanf("%c",&c);
    scanf("%c",&c);


textcolor(GRAY,BLACK);
 printf("\n\n");
 printf("\n\n");
 printf("---------------------------------");
 printf("\n\n");
 printf("           S O R T I N G");
 printf("\n\n");
 printf("---------------------------------");
 printf("\n\n");
 textcolor(WHITE,BLACK);
 printf("\n\n");

 for (i = 1 ; i <=  n-1 ; i++)
 {
     textcolor(WHITE,BLACK);
     if (checkspace==1)
     scanf("%c",&c);


   round = i;
   for(j=0;j<n;j++)
   printf("---");
   printf(" ROUND %d ", round);
   for(j=0;j<n;j++)
   printf("---");
   printf("\n\n");


     textcolor(GRAY,BLACK);
     printf("                ");
     for(j=0;j<n;j++)
     {
     printf("(%d) ",j+1);
     }
     printf("\n\n");


   for (s = i , j=0 ; s > 0; s-- , j++)
   {
     icheck++;
     textcolor(GREEN,BLACK);
     printf("No.%d & No.%d ",s,s+1);
     textcolor(GRAY,BLACK);
```

```
  printf("--> ");


if (checksort==1 && num[s] < num[s-1])
{
icount++;

for (k = 0; k < n ; k++)
    {
            if(k==s-1)
            {
                  textcolor(GREEN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }
    if (checkspace==1)
     scanf("%c",&c);
     printf("\n");

textcolor(GREEN,BLACK);

printf("            ");
textcolor(GRAY,BLACK);
  printf("--> ");

  bold = s;
    sort     = num[s] ;
  num[s]   = num[s-1];
  num[s-1]=sort ;


    for (k = 0; k < n ; k++)
    {
            if(k==s-1)
            {
                  textcolor(GREEN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }
```

```
 printf("\n");
}

else if (checksort==2 && num[s] > num[s-1])
{
icount++;

for (k = 0; k < n ; k++)
    {
            if(k==s-1)
            {
                  textcolor(GREEN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }
    if (checkspace==1)
     scanf("%c",&c);
     printf("\n");

textcolor(GREEN,BLACK);

printf("            ");
textcolor(GRAY,BLACK);
  printf("--> ");

  bold = s;
    sort     = num[s] ;
  num[s]   = num[s-1];
  num[s-1]= sort ;


    for (k = 0; k < n ; k++)
    {
            if(k==s-1)
            {
                  textcolor(GREEN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}
```

```
          }


       printf("\n");
      }


      else
      {



              for (k = 0; k < n ; k++)
          {
                  if(k==s-1)
                  {
                        textcolor(GREEN,BLACK);
            printf("%d ", num[k]);
            printf("%d ", num[k+1]);
            k++;
      }
      else
      {
            textcolor(WHITE,BLACK);
            printf("%d ", num[k]);
      }

          }

           printf("\n");
           break;
        }
        if (checkspace==1)
        scanf("%c",&c);
    }


     printf("\n");

}


  printf("\n\n");
   textcolor(GRAY,BLACK);
  for(j=0;j<n;j++)
    printf("---");
    printf(" E N D ");
    for(j=0;j<n;j++)
    printf("---");

    printf("\n\n");
 textcolor(WHITE,BLACK);

   printf("\n>Output: ");
```

```
  for (i = 0; i < n ; i++)
     printf("   %d", num[i]);

   printf("\n\n");
   printf("\n\n");

   textcolor(WHITE,BLACK);
   printf("COMPARE --> ");
   textcolor(GREEN,BLACK);
    printf("%d ", icheck);
    printf("\n\n");

    textcolor(WHITE,BLACK);
    printf("SORTING --> ");
    textcolor(GREEN,BLACK);
    printf("%d ", icount);
    printf("\n\n");
      printf("\n\n");

      scanf("%c");
      scanf("%c");

}

void bubble()
{
   int num[1000],keep[1000], n, i, j, k , s, bold, sort, round
,ip,l,m,count=0,check=0,icheck=0,icount=0;
  int checkinput=0, checksort=0, checkspace=0;

      srand(time(NULL));
      int r = rand() %100;
      char c;

      textcolor(WHITE,BLACK);

  printf("-------------------------------------------------");
  printf("\n\n");
  printf("            B U B B L E     S O R T");
  printf("\n\n");
  printf("-------------------------------------------------");

  printf("\n\n");
  textcolor(GRAY,BLACK);
  printf(">Enter amount of number\n\n> ");
   textcolor(WHITE,BLACK);
  scanf("%d", &n);
  printf("\n\n");

  printf("\n\n");
  textcolor(GRAY,BLACK);
  printf(">Choose how to input data 1 by random or 2 by yourself\n\n ");
   textcolor(WHITE,BLACK);
  scanf("%d", &checkinput);
```

```
printf("\n\n");

printf("\n\n");
textcolor(GRAY,BLACK);
printf(">Choose how to sort 1 ascending or 2 descending\n\n ");
 textcolor(WHITE,BLACK);
scanf("%d", &checksort);
printf("\n\n");

printf("\n\n");
textcolor(GRAY,BLACK);
printf(">Choose when to pause 1 step by step or 2 no pause\n\n ");
 textcolor(WHITE,BLACK);
scanf("%d", &checkspace);
printf("\n\n");


if (checkinput == 2)
{
    printf("\n\n");
    textcolor(GRAY,BLACK);
    printf(">Enter input data %d numbers \n\n > ",n);
 textcolor(WHITE,BLACK);
    printf("\n\n");
}

for (i = 0; i < n ; i++)
{
    if (checkinput == 1)
    {
          num[i] = rand() %1000;
          keep[i] = num[i];
    }
    else
    {
          scanf("%d", &num[i]);
    }
}
 printf("\n>Input: ");
textcolor(WHITE,BLACK);
 for (i = 0; i < n ; i++)
   printf("   %d", num[i]);

  scanf("%c",&c);


 printf("\n\n");

textcolor(GRAY,BLACK);
printf("\n\n");
printf("\n\n");

printf("---------------------------------");
printf("\n\n");
printf("           S O R T I N G");
```

```
printf("\n\n");
printf("---------------------------------");
printf("\n\n");
textcolor(WHITE,BLACK);
printf("\n\n");

for (i = 0 ; i < ( n - 1); i++)
{
    textcolor(WHITE,BLACK);
    if (checkspace==1)
    scanf("%c",&c);

  round = i;
  for(j=0;j<n;j++)
  printf("---");
  printf(" ROUND %d ", round+1);
  for(j=0;j<n;j++)
  printf("---");
  printf("\n\n");


    textcolor(GRAY,BLACK);
    printf("                 ");
    for(j=0;j<n;j++)
    {
    printf("(%d) ",j+1);
    }
    printf("\n\n");


  for (j = 0 ; j < n - i - 1; j++)
  {
    check++;
    textcolor(CYAN,BLACK);
    printf("No.%d & No.%d ",j+1,j+2);
    textcolor(GRAY,BLACK);
      printf("--> ");


    if ( checksort==1 && num[j] > num[j+1])
    {
    count++;

    for (k = 0; k < n ; k++)
        {
                if(k==j)
                {
                      textcolor(CYAN,BLACK);
          printf("%d ", num[k]);
          printf("%d ", num[k+1]);
          k++;
    }
    else
    {
```

```
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }
    if (checkspace==1)
     scanf("%c",&c);
     printf("\n");

textcolor(CYAN,BLACK);

printf("            ");
textcolor(GRAY,BLACK);
  printf("--> ");

  bold = j;
    sort       =   num[j];
    num[j]     =   num[j+1];
    num[j+1] = sort;


    for (k = 0; k < n ; k++)
    {
            if(k==j)
            {
                  textcolor(CYAN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }


 printf("\n");
}

else if ( checksort==2 && num[j] < num[j+1])
{
count++;

for (k = 0; k < n ; k++)
    {
            if(k==j)
            {
                  textcolor(CYAN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
```

```
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }
    if (checkspace==1)
     scanf("%c",&c);
     printf("\n");

textcolor(CYAN,BLACK);

printf("            ");
textcolor(GRAY,BLACK);
  printf("--> ");

  bold = j;
    sort       =   num[j];
    num[j]     =   num[j+1];
    num[j+1] = sort;


    for (k = 0; k < n ; k++)
    {
            if(k==j)
            {
                  textcolor(CYAN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }


 printf("\n");
}

else
{



         for (k = 0; k < n ; k++)
    {
            if(k==j)
            {
                  textcolor(CYAN,BLACK);
      printf("%d ", num[k]);
```

```
            printf("%d ", num[k+1]);
            k++;
      }
      else
      {
            textcolor(WHITE,BLACK);
            printf("%d ", num[k]);
      }

          }

           printf("\n");
        }
        if (checkspace==1)
        scanf("%c",&c);
    }


     printf("\n");

  }
  printf("\n\n");
   textcolor(GRAY,BLACK);
  for(j=0;j<n;j++)
    printf("---");
    printf(" E N D ");
    for(j=0;j<n;j++)
    printf("---");

    printf("\n\n");
textcolor(WHITE,BLACK);

   printf("\n>Output: ");

  for (i = 0; i < n ; i++)
     printf("   %d", num[i]);

   printf("\n\n");
   printf("\n\n");

   textcolor(WHITE,BLACK);
   printf("COMPARE --> ");
   textcolor(CYAN,BLACK);
    printf("%d ", check);
    printf("\n\n");

    textcolor(WHITE,BLACK);
    printf("SORTING --> ");
    textcolor(CYAN,BLACK);
    printf("%d ", count);
    printf("\n\n");

      printf("\n\n");

      scanf("%c",&c);
```

```
}

void comparison()
{
  int num[1000]={},keep[1000]={}, n=0, i=0, j=0, k=0 , s=0, bold=0, sort=0, round=0
,ip=0,l=0,m=0,count=0,check=0,icheck=0,icount=0;
   int checkinput=0, checksort=0, checkspace=0;

      srand(time(NULL));
      int r = rand() %100;
      char c;

      textcolor(WHITE,BLACK);

  printf("--------------------------------------------------");
  printf("\n\n");
  printf("            PROGRAM COMPARISION");
  printf("\n\n");
  printf("--------------------------------------------------");
  printf("\n\n");
  textcolor(GRAY,BLACK);
  printf(">Enter amount of number\n\n ");
   textcolor(WHITE,BLACK);
  scanf("%d", &n);
  printf("\n\n");


  for (i = 0; i < n ; i++)
  {
      num[i] = rand() %1000;
      keep[i] = num[i];
  }
   printf("\n>Input: ");
 textcolor(WHITE,BLACK);
  for (i = 0; i < n ; i++)
     printf("   %d", num[i]);

    if (checkspace==1)
     scanf("%c",&c);

   printf("\n\n\n\n");



   textcolor(CYAN,BLACK);

       printf("--------------------------------------------------");
  printf("\n\n");
  printf("            B U B B L E     S O R T");
  printf("\n\n");
  printf("--------------------------------------------------");
  printf("\n\n");
  textcolor(CYAN,BLACK);
```

```
   printf("\n>Input: ");
//textcolor(WHITE,BLACK);
  for (i = 0; i < n ; i++)
     printf("   %d", num[i]);

 textcolor(GRAY,BLACK);
  printf("\n\n");
  printf("\n\n");
  printf("---------------------------------");
  printf("\n\n");
  printf("           S O R T I N G");
  printf("\n\n");
  printf("---------------------------------");
  printf("\n\n");
  textcolor(WHITE,BLACK);
  printf("\n\n");

  for (i = 0 ; i < ( n - 1); i++)
  {
      textcolor(WHITE,BLACK);
      if (checkspace==1)
      scanf("%c",&c);

    round = i;
    for(j=0;j<n;j++)
    printf("---");
    printf(" ROUND %d ", round+1);
    for(j=0;j<n;j++)
    printf("---");
    printf("\n\n");


      textcolor(GRAY,BLACK);
      printf("                 ");
      for(j=0;j<n;j++)
      {
      printf("(%d) ",j+1);
      }
      printf("\n\n");


    for (j = 0 ; j < n - i - 1; j++)
    {
      check++;
      textcolor(CYAN,BLACK);
      printf("No.%d & No.%d ",j+1,j+2);
      textcolor(GRAY,BLACK);
        printf("--> ");


      if (num[j] > num[j+1])
      {
      count++;
```

```
for (k = 0; k < n ; k++)
    {
            if(k==j)
            {
                  textcolor(CYAN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }
    if (checkspace==1)
     scanf("%c",&c);
     printf("\n");

textcolor(CYAN,BLACK);

printf("             ");
textcolor(GRAY,BLACK);
  printf("--> ");

  bold = j;
    sort       =   num[j];
    num[j]     =   num[j+1];
    num[j+1] = sort;


    for (k = 0; k < n ; k++)
    {
            if(k==j)
            {
                  textcolor(CYAN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }


 printf("\n");
}

else
{
```

```
            for (k = 0; k < n ; k++)
        {
                if(k==j)
                {
                      textcolor(CYAN,BLACK);
          printf("%d ", num[k]);
          printf("%d ", num[k+1]);
          k++;
    }
    else
    {
          textcolor(WHITE,BLACK);
          printf("%d ", num[k]);
    }

        }

         printf("\n");
      }
      if (checkspace==1)
      scanf("%c",&c);
  }


   printf("\n");

 }
 printf("\n\n");
  textcolor(GRAY,BLACK);
 for(j=0;j<n;j++)
   printf("---");
   printf(" E N D ");
   for(j=0;j<n;j++)
   printf("---");

   printf("\n\n");
textcolor(WHITE,BLACK);

  printf("\n>Output: ");

 for (i = 0; i < n ; i++)
    printf("   %d", num[i]);

  printf("\n\n");
  printf("\n\n");

  textcolor(WHITE,BLACK);
  printf("COMPARE --> ");
  textcolor(CYAN,BLACK);
   printf("%d ", check);
   printf("\n\n");
```

```
    textcolor(WHITE,BLACK);
    printf("SORTING --> ");
    textcolor(CYAN,BLACK);
    printf("%d ", count);
    printf("\n\n");

      printf("\n\n");

      if (checkspace==1)
      scanf("%c",&c);

      for (i = 0; i < n ; i++)
   {
      num[i] = keep[i];
   }

      //-----------------------------------------------------------------------------------------------------------------------------



            textcolor(GREEN,BLACK);

  printf("-------------------------------------------------");
  printf("\n\n");
  printf("            I N S E R T I O N    S O R T");
  printf("\n\n");
  printf("-------------------------------------------------");
  printf("\n\n");


   printf("\n>Input: ");
 textcolor(WHITE,BLACK);
  for (i = 0; i < n ; i++)
     printf("   %d", num[i]);

 textcolor(GRAY,BLACK);
  printf("\n\n");
  printf("\n\n");
  printf("---------------------------------");
  printf("\n\n");
  printf("           S O R T I N G");
  printf("\n\n");
  printf("---------------------------------");
  printf("\n\n");
  textcolor(WHITE,BLACK);
  printf("\n\n");

  for (i = 1 ; i <=  n-1 ; i++)
  {
      textcolor(WHITE,BLACK);
      if (checkspace==1)
      scanf("%c",&c);

    round = i;
```

```
for(j=0;j<n;j++)
printf("---");
printf(" ROUND %d ", round);
for(j=0;j<n;j++)
printf("---");
printf("\n\n");


  textcolor(GRAY,BLACK);
  printf("                 ");
  for(j=0;j<n;j++)
  {
  printf("(%d) ",j+1);
  }
  printf("\n\n");


for (s = i , j=0 ; s > 0; s-- , j++)
{
  icheck++;
  textcolor(GREEN,BLACK);
  printf("No.%d & No.%d ",s,s+1);
  textcolor(GRAY,BLACK);
    printf("--> ");


  if (num[s] < num[s-1])
  {
  icount++;

  for (k = 0; k < n ; k++)
      {
              if(k==s-1)
              {
                    textcolor(GREEN,BLACK);
       printf("%d ", num[k]);
       printf("%d ", num[k+1]);
       k++;
  }
  else
  {
       textcolor(WHITE,BLACK);
       printf("%d ", num[k]);
  }

      }
      if (checkspace==1)
       scanf("%c",&c);
       printf("\n");

  textcolor(GREEN,BLACK);

  printf("            ");
  textcolor(GRAY,BLACK);
```

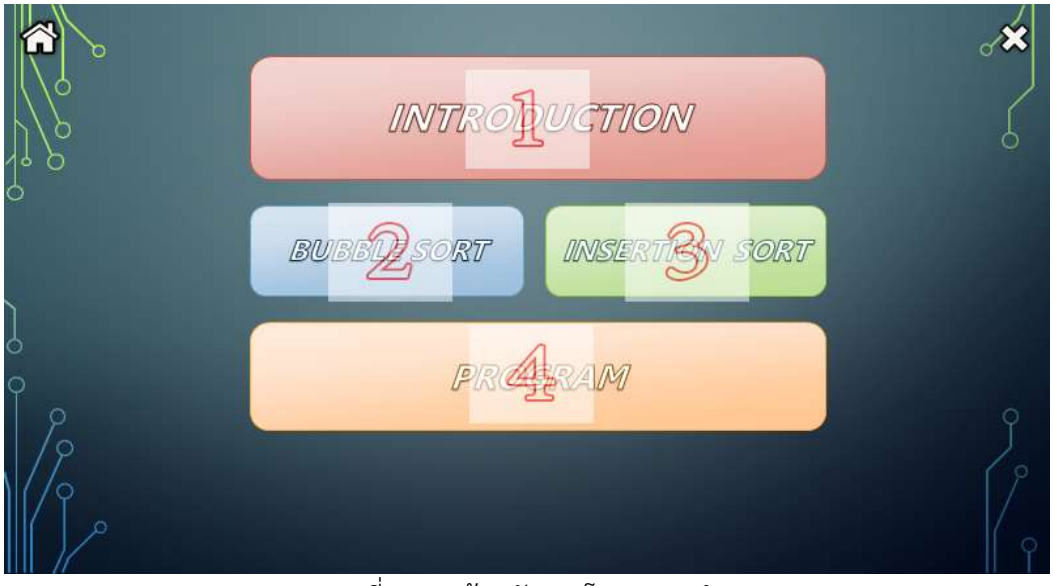
```
  printf("--> ");

  bold = s;
    sort     = num[s] ;
  num[s]   = num[s-1];
  num[s-1]= sort ;


    for (k = 0; k < n ; k++)
    {
            if(k==s-1)
            {
                  textcolor(GREEN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }


 printf("\n");
}

else
{



        for (k = 0; k < n ; k++)
    {
            if(k==s-1)
            {
                  textcolor(GREEN,BLACK);
      printf("%d ", num[k]);
      printf("%d ", num[k+1]);
      k++;
}
else
{
      textcolor(WHITE,BLACK);
      printf("%d ", num[k]);
}

    }

     printf("\n");
     break;
  }
  if (checkspace==1)
```

```
        scanf("%c",&c);
    }


     printf("\n");

}


  printf("\n\n");
   textcolor(GRAY,BLACK);
  for(j=0;j<n;j++)
    printf("---");
    printf(" E N D ");
    for(j=0;j<n;j++)
    printf("---");

    printf("\n\n");
 textcolor(WHITE,BLACK);

   printf("\n>Output: ");

  for (i = 0; i < n ; i++)
     printf("   %d", num[i]);

   printf("\n\n");
   printf("\n\n");

   textcolor(WHITE,BLACK);
   printf("COMPARE --> ");
   textcolor(GREEN,BLACK);
    printf("%d ", icheck);
    printf("\n\n");

    textcolor(WHITE,BLACK);
    printf("SORTING --> ");
    textcolor(GREEN,BLACK);
    printf("%d ", icount);
    printf("\n\n");
      printf("\n\n");
      //=============================================
      if (checkspace==1)
       scanf("%c",&c);

      textcolor(GRAY,BLACK);
      printf("-------------------------------------------------------------------
-----------------------------------------");

       printf("\n\n");
   printf("\n\n");

   textcolor(WHITE,BLACK);
   printf("BUBBLE COMPARE --> ");
   textcolor(CYAN,BLACK);
    printf("%d ", check);
    printf("\n\n");
```

```
    textcolor(WHITE,BLACK);
    printf("BUBBLE SORTING --> ");
    textcolor(CYAN,BLACK);
    printf("%d ", count);
    printf("\n\n");

      textcolor(WHITE,BLACK);
   printf("INSERTION COMPARE --> ");
   textcolor(GREEN,BLACK);
    printf("%d ", icheck);
    printf("\n\n");

    textcolor(WHITE,BLACK);
    printf("INSERTION SORTING --> ");
    textcolor(GREEN,BLACK);
    printf("%d ", icount);
    printf("\n\n");

      printf("\n\n");
      scanf("%c");
      scanf("%c");

}

int main()
{
      int x,check=0;
      char c;
      while(check==0)
      {
            printf("Choose the program\n");
            printf("1-Bubble Sorting Program\n");
            printf("2-Insertion Sorting Program\n");
            printf("3-Program Comparison\n>");
            scanf("%d",&x);


            if(x==1)bubble();
            if(x==2)insertion();
            if(x==3)comparison();

            textcolor(WHITE,BLACK);
            printf("\nEXIT? (Y/N)\n>");
            scanf("%c",&c);

            if(c=='Y'|| c=='y' ) check++;

            printf("\n\n");
      }

}
```

ภาคผนวก ค
คู่มือการใช้งาน

โปรแกรมแสดงการเปรียบเทียบ วิธีการจัดเรียงข้อมูลแบบฟองและแบบแทรก รันบนระบบ Microsoft Offices ซึ่งพัฒนาจากโปรแกรม PowerPoint เมื่อรันโปรแกรมแล้วจะปรากฏหน้าจอ ดังรูปที่ ค.1 ซึ่งมีแถบให้เลือกทำงานดังนี้



รูปที่ ค.1 หน้าหลักของโปรแกรมนำเสนอ

กลับสู่หน้าหลัก

ปิดโปรแกรมนำเสนอ

1) INTRODUCTION เข้าสู่หน้าอธิบายเบื้องต้นเกี่ยวกับการเรียงลำดับข้อมูล

2) BUBBLE SORT เข้าสู่หน้าอธิบายเบื้องต้นเกี่ยวกับการเรียงลำดับข้อมูลแบบฟอง

3) INSERTION SORT เข้าสู่หน้าอธิบายเบื้องต้นเกี่ยวกับการเรียงลำดับข้อมูลแบบแทรก

4) PROGRAM เปิดโปรแกรมเพื่อทดสอบการเรียงลำดับ โดยจะแสดงการเรียงลำดับข้อมูลแบบฟองและแบบแทรก พร้อมทั้งเปรียบเทียบผลการเรียงลำดับทั้งสองวิธี

สำหรับหน้าที่อธิบายถึงเนื้อหาต่าง ๆ ในโปรแกรมด้านล่าง จะมีปุ่มเลือกทำงาน ดังรูปที่ ค.2 ดังนี้

 กลับสู่หน้าแรกของหัวข้อ

 ย้อนกลับไปหน้าก่อน

 เลื่อนไปหน้าถัดไ

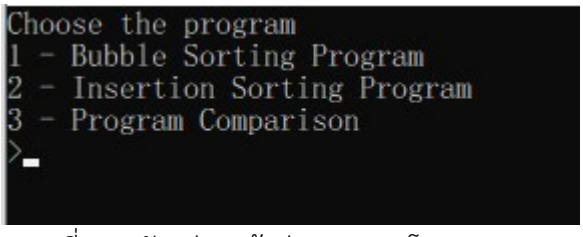
 เลื่อนไปหัวข้อถัดไป

 เปิดโปรแกรมเพื่อทดสอบการเรียงลำดับ

รูปที่ ค.2 ตัวอย่างหน้าต่างในโปรแกรมนำเสนอ

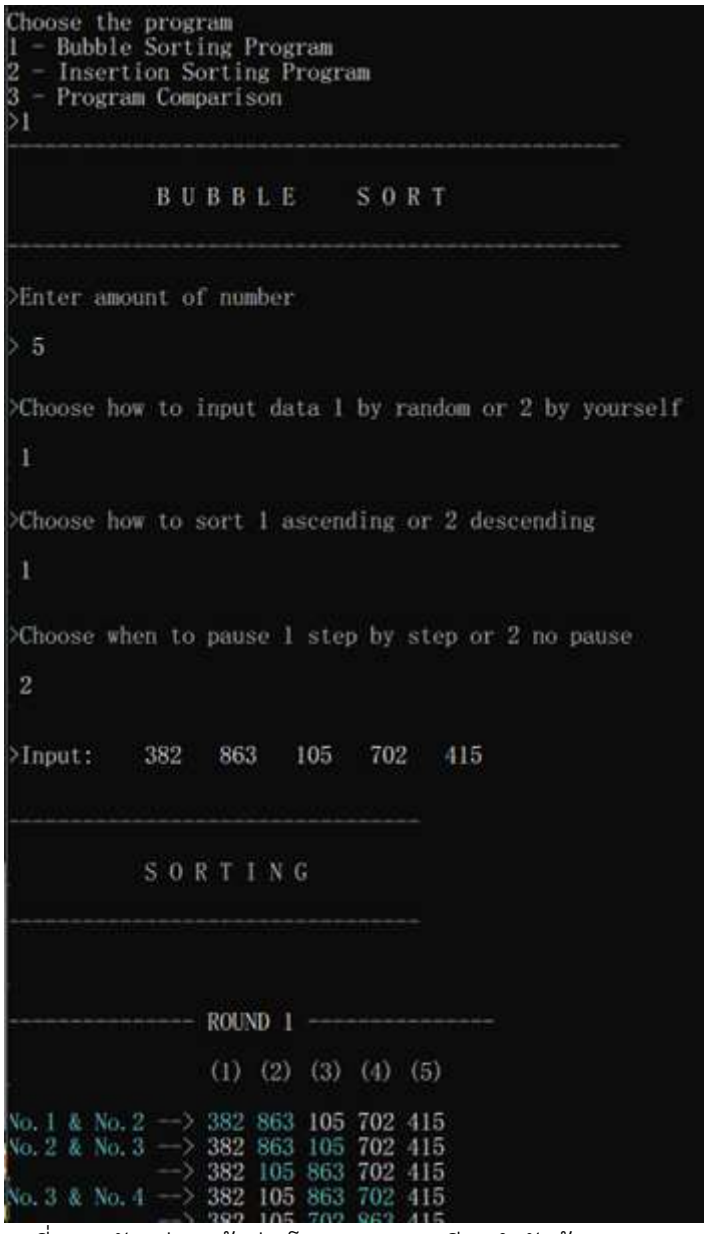
เมื่อเลือกรันเปิดโปรแกรมเพื่อทดสอบการเรียงลำดับ จะปรากฏหน้าต่างดังรูปที่ ค.3 :ซึ่งจะปรากฏรายการเมนูให้เลือกทำงาน 3 อย่างคือ

1. Bubble Sorting Program เพื่อทำการเรียงลำดับแบบฟอง
2. Insertion Sorting Program เพื่อทำการเรียงลำดับแบบแทรก
3. Program Comparision เพื่อเปรียบเทียบการเรียงลำดับแบบฟองและแบบแทรก



รูปที่ ค.3 ตัวอย่างหน้าต่างแรกของโปรแกรม

ให้ผู้ใช้ใส่ตัวเลขที่ต้องการเลือกทำงาน ถ้าเลือกกดเลข 1 เพื่อเลือกทำการเรียงลำดับแบบฟอง จะปรากฏหน้าจอ ดังรูปที่ ค.4



รูปที่ ค.4 ตัวอย่างหน้าต่างโปรแกรมการเรียงลำดับข้อมูลแบบฟอง

ผู้ใช้ต้องใส่ข้อมูลเพื่อทำงานตามลำดับดังนี้

(1) จำนวนข้อมูล (Enter amount of number)

(2) เลือกวิธีการรับข้อมูล 1. รับข้อมูลโดยการสุ่ม ซึ่งโปรแกรมจะสุ่มตัวเลขซึ่งมีค่าตั้งแต่ 0 ถึง 999
2. รับข้อมูลโดยผู้ใช้งานใส่ข้อมูลเอง

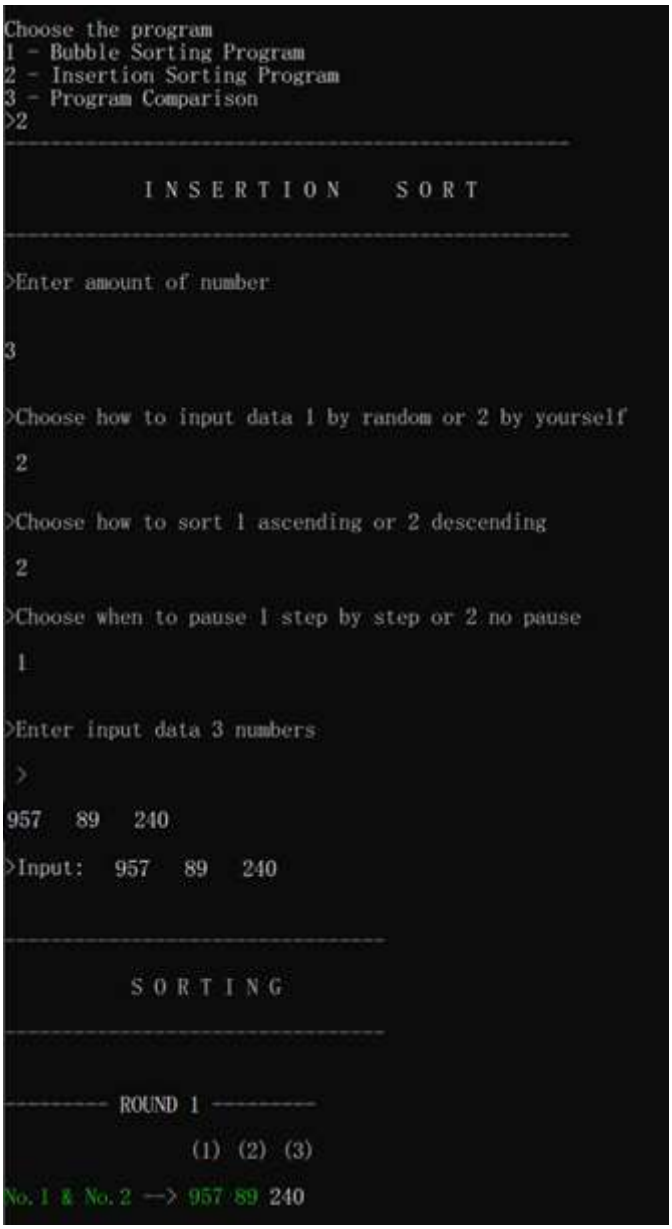
(3) เลือกลำดับการเรียงลำดับข้อมูล 1. จากน้อยไปหามาก 2. จากมากไปหาน้อย

(4) เลือกการหยุด (pause) ของโปรแกรม

1. หยุดทุกครั้งหรือทุกบรรทัดที่มีการสลับค่าข้อมูล เพื่อให้ผู้ใช้ได้ทำความเข้าใจกับการเรียนลำดับได้อย่างเป็นขั้นตอน (ทุกครั้งที่มีการหยุด ผู้ใช้ต้องกด Enter เพื่อให้โปรแกรมดำเนินการต่อ)

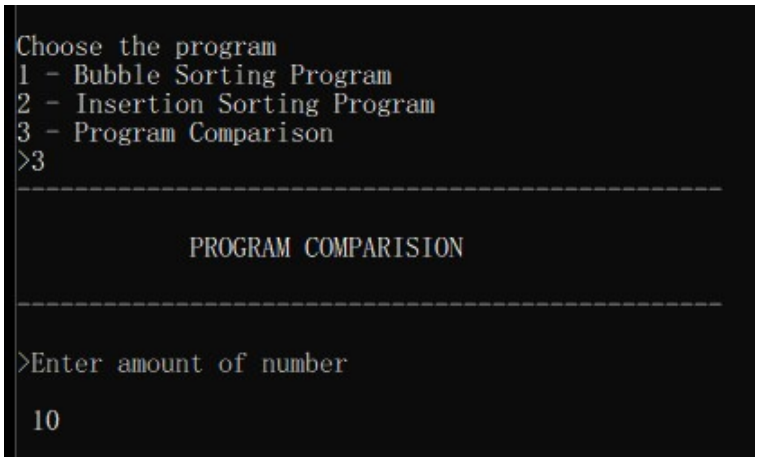
2. ไม่หยุดในระหว่างการเรียงลำดับ

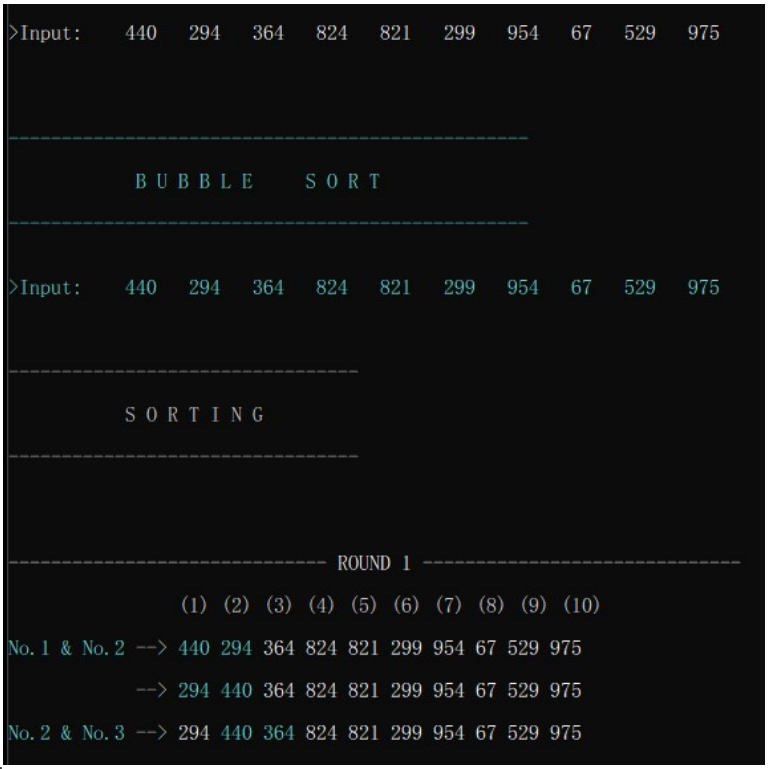
จากรูปที่ ค.3 ถ้าเลือกกดเลข 2 เพื่อเลือกทำการเรียงลำดับแบบแทรก จะปรากฏหน้าจอ ดังรูป ค.5 ผู้ใช้ต้องใส่ข้อมูลเพื่อเลือกการทำงานตามลำดับในทำนองเดียวกับการเรียงลำดับข้อมูลแบบฟอง



รูปที่ ค.5 ตัวอย่างหน้าต่างโปรแกรมการเรียงลำดับข้อมูลแบบแทรก

จากรูปที่ ค.3 ถ้าเลือกกดเลข 3 เพื่อเลือกทำการเปรียบเทียบการเรียงลำดับข้อมูล จะปรากฏหน้าจอดังรูปที่ ค.6 ผู้ใช้ต้องใส่ข้อมูลเพื่อทำงานตามลำดับในทำนองเดียวกับการเรียงลำดับข้อมูลแบบฟอง





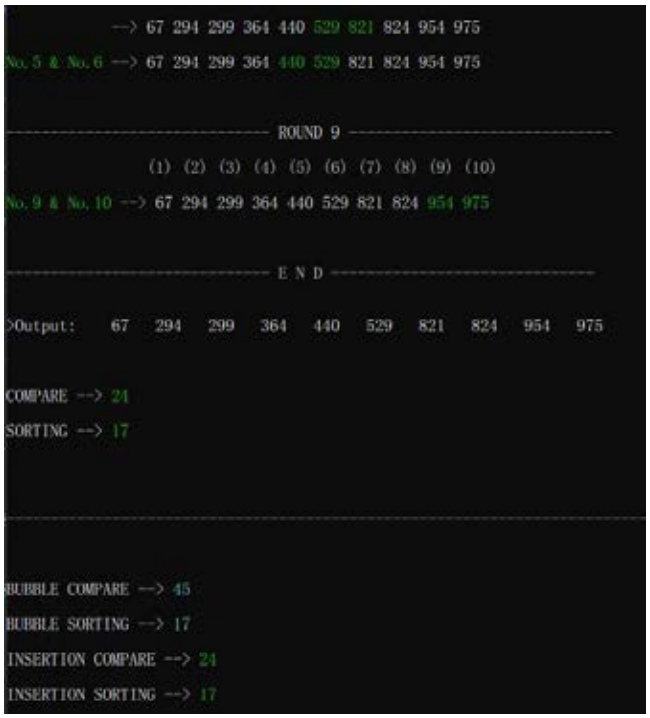
รูปที่ ค.6 ตัวอย่างหน้าต่างโปรแกรมเปรียบเทียบการเรียงลำดับข้อมูล (ช่วงต้น)

โปรแกรมเริ่มการเรียงลำดับด้วยการแสดงการจัดเรียงลำดับข้อมูลแบบฟอง (โดยใช้สีฟ้าเป็นหลัก)

รูปที่ ค.7 ตัวอย่างหน้าต่างโปรแกรมเปรียบเทียบการเรียงลำดับข้อมูล (ช่วงกลาง)

เมื่อจัดเรียงเสร็จแล้ว จะแสดงผล จำนวนครั้งที่เกิดการเปรียบเทียบค่าข้อมูล (COMPARE) และจำนวนครั้งที่เกิดการสลับที่เพื่อจัดเรียงของข้อมูล (SORTING)

จากนั้นจึงเริ่มแสดงการจัดเรียงลำดับข้อมูลแบบแทรก (INSERTION SORT) (โดยใช้สีเขียวเป็นหลัก)



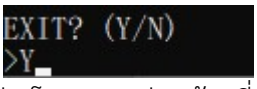
รูปที่ ค.8 ตัวอย่างหน้าต่างโปรแกรมเปรียบเทียบการเรียงลำดับข้อมูล (ช่วงท้าย)

โปรแกรมเปรียบเทียบการเรียงลำดับข้อมูล จะทำการสรุปรวมข้อมูล ดังนี้ จำนวนครั้งที่เกิดการเปรียบเทียบค่าข้อมูลในการเรียงลำดับข้อมูลแบบฟอง (BUBBLE COMPARE) จำนวนครั้งที่เกิดการสลับที่ของข้อมูลหรือเกิดการจัดเรียงข้องมูลในการเรียงลำดับข้อมูลแบบฟอง (BUBBLE SORTING) จำนวนครั้งที่เกิดการเปรียบเทียบค่าข้อมูลในการเรียงลำดับข้อมูลแบบแทรก (INSERTION COMPARE) และจำนวนครั้งที่เกิดการสลับที่ของข้อมูลหรือเกิดการจัดเรียงข้องมูลในการเรียงลำดับข้อมูลแบบแทรก (INSERTION SORTING) เพื่อให้ผู้ใช้งานสามารถเปรียบเทียบการเรียงลำดับข้อมูลทั้งสองวิธีได้

โดยทุกครั้งที่โปรแกรมแสดงผลเสร็จสิ้นแล้ว จะปรากฏทางเลือกให้ผู้ใช้งานว่าต้องการปิดโปรแกรมเลยหรือไม่ ดังรูปที่ ค.9

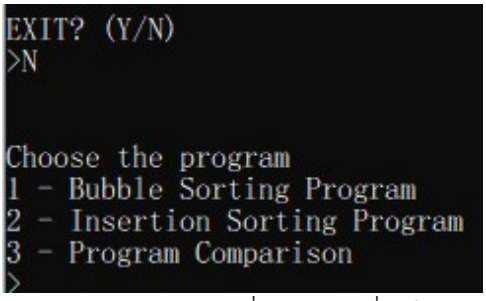
รูปที่ ค.9 ตัวอย่างหน้าต่างโปรแกรมส่วนท้าย

จากรูปที่ ค.9 ถ้าผู้ใช้เลือก Y ซึ่งคือ ใช่ (Yes) ต้องการที่จะปิดโปรแกรมจะเป็นการปิดตัวโปรแกรม ดังรูปที่ ค.10



รูปที่ ค.10 ตัวอย่างหน้าต่างโปรแกรมส่วนท้ายที่เลือก Y เพื่อปิดโปรแกรม

จากรูปที่ ค.9 ถ้าผู้ใช้เลือก N ซึ่งคือ ไม่ (No) ยังไม่ต้องการปิดโปรแกรม และเป็นการเริ่มโปรแกรมใหม่อีกครั้ง ดังรูปที่ ค.11



รูปที่ ค.11 ตัวอย่างหน้าต่างโปรแกรมส่วนท้ายที่เลือก N เพื่อเป็นการเริ่มโปรแกรมใหม่อีกครั้ง

ประวัติผู้เขียน

นายชัยฤทธิ์ ขาวผ่อง เกิดเมื่อวันที่ 18 สิงหาคม พ.ศ.2536 ที่จังหวัดกรุงเทพมหานคร สำเร็จการศึกษา สาขาวิชาคณิตศาสตร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2561