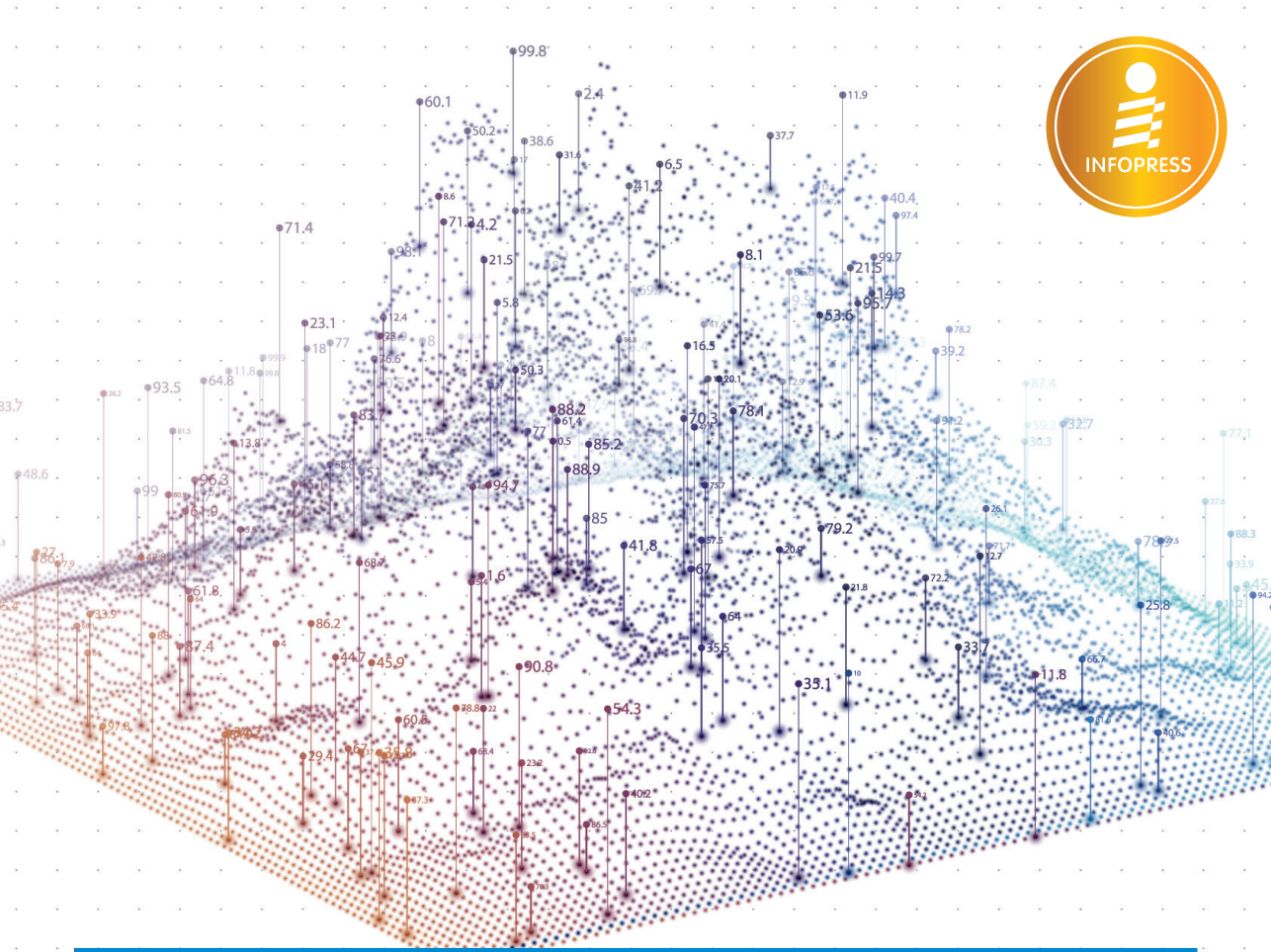


INFOPRESS

คู่มือเรียนโครงสร้างข้อมูลและอัลกอริทึม

Data Structure and Algorithm

- ครอบคลุมเนื้อหาทั้งโครงสร้างข้อมูล อัลกอริทึม ตลอดจนการวิเคราะห์ประสิทธิภาพของอัลกอริทึม
- อ่านเข้าใจง่าย มีตัวอย่างอธิบายการทำงานทุกหัวข้อ
- มีตัวอย่างโค้ดโปรแกรมทั้งภาษา C และภาษา Java พร้อมคำอธิบายอย่างละเอียด
- เหมาะสำหรับนิสิต นักศึกษา อาจารย์และผู้ที่สนใจ

ฉบับสมบูรณ์
2nd Edition

ฟรี ไฟล์ตัวอย่างภายในเล่ม
https://serazu.com/
9786164870062

วิษณุ ช้างเนียม บรรณาธิการ กิตินันท์ พลสวัสดิ์

IDC
PREMIER

มีเพียง “ความรู้” เท่านั้นที่มนุษย์ใช้พลิก “โลก” และเปลี่ยน “ชีวิต”
เราจึงสร้างสรรค์ และส่งมอบ “ความรู้” ในรูปแบบที่ดีกว่า
เพื่อให้คนไทย “เรียนรู้” ได้ตลอดชีวิต

INFOPRESS Think Beyond Read Comics Little Heart

คู่มือเรียนโครงสร้างข้อมูลและอัลกอริทึม

(Data Structure and Algorithm) ฉบับสมบูรณ์ 2nd Edition

AUTHOR

วิษณุ ช้างเนียม data_algo@hotmail.com

Editorial

กิตินันท์ พลสวัสดิ์ kitinan_p@idcpremier.com

GRAPHIC DESIGNERS

วสันต์ พึ่งพูลผล, วุฒิพันธ์ สมพระเมฆ

PAGE LAYOUT

กมลชนก ชัยสาครสมุทร

PROOFREADER

มนฤดี ศรีอุทโยภาส

PUBLISHING COORDINATORS

วรพล ณธิกุล, สุพัตรา อาจปรุ, มงคล แก้วพลอย

PUBLISHED AND DISTRIBUTED BY

IDC PREMIER

บริษัท ไอดีซี พรีเมียร์ จำกัด

200 หมู่ 4 ชั้น 19 ห้อง 1901

อาคารจัสมินอินเตอร์เนชั่นแนลทาวเวอร์

ถ.แจ้งวัฒนะ อ.ปากเกร็ด จ.นนทบุรี 11120

โทรศัพท์ 0-2962-1081 (อัตโนมัติ 10 คู่สาย)

โทรสาร 0-2962-1084

สมาชิกสัมพันธ์

โทรศัพท์ 0-2962-1081-3 ต่อ 121

โทรสาร 0-2962-1084

ร้านค้าและตัวแทนจำหน่าย

โทรศัพท์ 0-2962-1081-3 ต่อ 112-114

โทรสาร 0-2962-1084

INFOPRESS

พิมพ์ครั้งที่ 1 กุมภาพันธ์ 2562

2 4 6 8 10 9 7 5 3 1

ข้อมูลทางบรรณานุกรม

วิษณุ ช้างเนียม

คู่มือเรียนโครงสร้างข้อมูลและอัลกอริทึม

(Data Structure and Algorithm) ฉบับสมบูรณ์ 2nd Edition

นนทบุรี : ไอดีซีฯ, 2562

408 หน้า

1. โครงสร้างข้อมูล

I ชื่อเรื่อง

005.73

ISBN 885-916-100-536-0

บริษัท ไอดีซี พรีเมียร์ จำกัด จัดตั้งขึ้นเพื่อเผยแพร่ความรู้ที่มีคุณภาพสู่ผู้อ่านชาวไทย เรายินดีรับงานเขียนของนักวิชาการและนักเขียนทุกท่าน ท่านผู้สนใจกรุณาติดต่อผ่านทางอีเมลที่ infopress@idcpremier.com หรือทางโทรศัพท์หมายเลข 0-2962-1081 (อัตโนมัติ 10 คู่สาย) โทรสาร 0-2962-1084

ราคา 395 บาท

คำนำ

หนังสือเล่มนี้ปรับปรุงจากหนังสือเนื้อหาคู่มือเรียนโครงสร้างข้อมูลและอัลกอริทึม (Data Structure & Algorithm) ฉบับสมบูรณ์ เพื่อใช้ประกอบการเรียนการสอนในรายวิชาโครงสร้างข้อมูลและอัลกอริทึม สำหรับนักเรียน นักศึกษา และครูอาจารย์ที่สนใจ

หนังสือเล่มนี้ประกอบด้วยเนื้อหา การวิเคราะห์ประสิทธิภาพของอัลกอริทึม อาร์เรย์ ลิงค์ลิสต์ สแต็ก คิว ทรี แฮช ไทรส์ การจัดเรียงข้อมูล กราค้นหาข้อมูล และกราฟ ในทุกๆ หัวข้อจะมีตัวอย่างประกอบการอธิบายเพื่อเพิ่มความเข้าใจ มีตัวอย่างโค้ดโปรแกรมทั้งภาษา C และภาษา Java สำหรับทดสอบการทำงานของอัลกอริทึม พร้อมทั้งคำอธิบายโค้ดอย่างละเอียด

สุดท้ายนี้ ขอขอบพระคุณคุณพ่อคุณแม่และครอบครัวของผู้เขียนที่ให้กำลังใจและให้การสนับสนุนมาตลอด ขอขอบคุณอาจารย์ทุกท่านที่ได้ประสิทธิประสาทวิชาให้กับผู้เขียน ประโยชน์ของหนังสือเล่มนี้ขอมอบให้อาจารย์ผู้ประสิทธิประสาทวิชาทุกท่าน บิดา-มารดา และครอบครัว

วิษณุ ช้างเนียม

บทบรรณาธิการ

ในการเรียนหรือการทำงานทางคอมพิวเตอร์ คงหลีกหนีไม่ได้ที่จะต้องเรียนหรือใช้ความรู้เกี่ยวกับโครงสร้างข้อมูลและอัลกอริทึม เนื่องด้วยเนื้อหาความรู้ในรายวิชานี้ เป็นความรู้พื้นฐานที่ถือได้ว่าสำคัญที่สุดในการเรียนหรือทำงานทางคอมพิวเตอร์ ไม่ว่าจะเป็นการใช้งานโครงสร้างข้อมูลอย่าง อาร์เรย์, ลิงค์ลิสต์, สแต็ก, คิว, ทรี หรือจะเป็นอัลกอริทึมในการจัดเรียงข้อมูลอย่าง Selection sort, Insertion sort, Merge sort หรือจะเป็นอัลกอริทึมในการค้นหาข้อมูลอย่าง Binary search, Rabin-Karp String search, Boyer-Moore String search

ด้วยความสำคัญดังกล่าว หนังสือเล่มนี้จึงได้รวบรวมและเรียบเรียง พร้อมยกตัวอย่างและอธิบายการทำงานของโครงสร้างข้อมูลและอัลกอริทึมต่างๆ ไว้อย่างครบถ้วน ทั้งการวิเคราะห์ประสิทธิภาพของอัลกอริทึม, อาร์เรย์และการอ้างอิงข้อมูล, ลิงค์ลิสต์, สแต็ก, คิว, ทรี, แฮช, ไทรส์, การจัดเรียงข้อมูล, การค้นหาข้อมูล และกราฟ อีกทั้งยังแสดงการทำงานของอัลกอริทึมด้วยโค้ด 2 ภาษา ทั้งภาษา C และภาษา Java เพื่อเป็นแนวทางให้ผู้อ่านได้ศึกษาและนำไปใช้งานได้จริง ที่สำคัญหนังสือเล่มนี้ได้อธิบายการทำงานของโค้ดโปรแกรมไว้อย่างละเอียด เพื่อช่วยให้ผู้อ่านได้ทำความเข้าใจการทำงานของอัลกอริทึมได้อย่างถูกต้อง

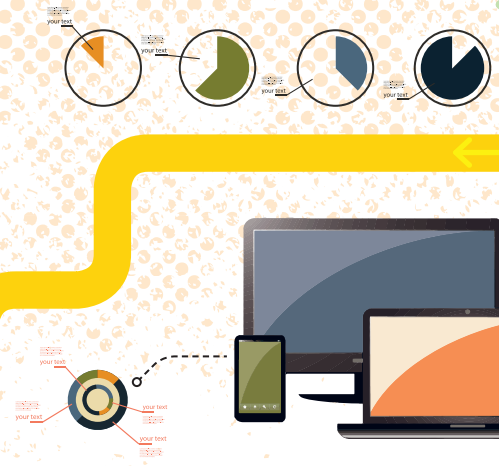
สุดท้ายนี้ ทางสำนักพิมพ์หวังเป็นอย่างยิ่งว่า หนังสือคู่มือเรียนและใช้งานโครงสร้างข้อมูลและอัลกอริทึม (Data Structrue and Algorithm) เล่มนี้ จะช่วยให้ผู้อ่านทุกท่านมีความเข้าใจโครงสร้างข้อมูลและอัลกอริทึมได้ดียิ่งขึ้น และสามารถนำไปใช้งานได้จริง

กิตินันท์ พลสวัสดิ์

บรรณาธิการ

สารบัญ

Data Structure & Algorithm



CHAPTER 01

รู้จักกับโครงสร้างข้อมูลและอัลกอริทึม

หนังสือเล่มนี้เป็นหนังสือสอนเรื่องโครงสร้างข้อมูลและอัลกอริทึม มุ่งเน้นในเรื่องวิธีการและขั้นตอนการแก้ไขปัญหาต่างๆ โดยใช้คุณสมบัติและความสามารถของโครงสร้างข้อมูล ซึ่งถือได้ว่าเป็นรายวิชาหลักในการเรียนทางด้านคอมพิวเตอร์ ดังนั้น ผู้อ่านควรศึกษาเนื้อหาและตัวอย่างต่างๆ ในหนังสือให้เข้าใจ เพื่อใช้เป็นพื้นฐานในการศึกษาเรื่องอื่นๆ ต่อไป

โครงสร้างข้อมูลและอัลกอริทึมคืออะไร

โครงสร้างข้อมูล (Data Structure) คือ การจัดการข้อมูลในหน่วยความจำภายในเครื่องคอมพิวเตอร์ หรือบางครั้งเป็นการจัดการข้อมูลในดิสก์ให้มีความสัมพันธ์กันภายในกลุ่มข้อมูล ให้มีรูปแบบและข้อกำหนดที่ชัดเจนในการกำหนดคุณสมบัติเพื่อสร้างความสัมพันธ์ภายในกลุ่มข้อมูล

รูปแบบการเก็บข้อมูลที่อยู่ในรูปแบบโครงสร้างข้อมูล เช่น อาร์เรย์ (Array), ลิงค์ลิสต์ (Linked-List), สแต็ก (Stack), ไบนารีทรี (Binary Tree) เป็นต้น

อัลกอริทึม (Algorithm) หรือเรียกอีกอย่างว่า **ขั้นตอนวิธี** เป็นวิธีการแสดงลำดับขั้นตอนในการทำงานหรือการลำดับขั้นตอนการแก้ไขปัญหา เช่น การกำหนดขั้นตอนเพื่อแก้ไขปัญหาการจัดเรียงเอกสารในแฟ้มข้อมูล หรือการกำหนดอัลกอริทึมในการค้นหาข้อมูลในแฟ้มข้อมูลทั้งหมด เป็นต้น

อัลกอริทึมหรือขั้นตอนวิธีพบได้ในชีวิตประจำวัน เช่น ขั้นตอนการเติมเงินในโทรศัพท์มือถือผ่านตู้เติมเงินอัตโนมัติ, ขั้นตอนการใช้งานตู้กดเงินอัตโนมัติ (ATM) เพื่อทำธุรกรรมทางการเงิน เป็นต้น

ตัวอย่างที่ 1.1 แสดงอัลกอริทึมหรือขั้นตอนวิธีการใช้ตู้กดเงินอัตโนมัติ (ATM) เพื่อโอนเงิน ดังนี้

1. ใส่บัตร ATM
2. ป้อนรหัสผ่านของบัตร ATM
3. ในหน้าบริการ เลือกบริการรายการโอนเงิน
4. เลือกรูปแบบการโอนเงินว่าจะโอนเงินเข้าบัญชีอื่นธนาคารเดียวกัน หรือธนาคารอื่นๆ
5. กดหมายเลขบัญชีที่ต้องการโอนเงินเข้าบัญชี
6. ตรวจสอบเลขที่บัญชีที่ป้อนถูกต้องหรือไม่ ถ้าถูกต้องให้กดตกลง
7. กรอกจำนวนเงินที่ต้องการโอนเงิน แล้วกดตกลง
8. ชื่อบัญชีและจำนวนเงินที่ต้องการโอนจะปรากฏขึ้นเพื่อยืนยันความถูกต้องการโอนเงิน เมื่อตรวจสอบบัญชีและจำนวนเงินถูกต้อง กดตกลง เป็นการเสร็จสิ้นการโอนเงิน
9. รับบัตร ATM
10. รับใบสลิปการโอนเงิน

ตัวอย่างที่ 1.2 แสดงอัลกอริทึมการหาข้อมูลในอาร์เรย์ขนาด n ข้อมูล ดังนี้

1. รับข้อมูลตัวเลขที่ต้องการค้นหา
2. เปรียบเทียบข้อมูลในอาร์เรย์ทีละตัวตั้งแต่ข้อมูลในตำแหน่งที่ 0 จนถึงตำแหน่งที่ n-1
3. ถ้าข้อมูลในอาร์เรย์ตรงกับข้อมูลที่ต้องการค้นหา แสดงว่าเจอข้อมูล จบการค้นหาข้อมูล
4. ถ้าเปรียบเทียบข้อมูลจนถึงตำแหน่งที่ n-1 แล้วไม่พบข้อมูลตัวใดในอาร์เรย์ที่มีข้อมูลตรงกับข้อมูลที่ต้องการค้นหาเลย แสดงว่าไม่มีข้อมูลที่ต้องการค้นหาในอาร์เรย์

ประโยชน์ของโครงสร้างข้อมูลและอัลกอริทึม

โครงสร้างข้อมูลและอัลกอริทึม เป็นวิธีการจัดการกับข้อมูลที่ทำให้คอมพิวเตอร์สามารถจัดการกับข้อมูล เพื่อนำมาใช้งานได้อย่างมีประสิทธิภาพ ตัวอย่างเช่น

- การจัดเก็บข้อมูลจำนวนมากในรูปแบบของโครงสร้างข้อมูลที่เหมาะสม จะทำให้สามารถนำข้อมูลไปใช้ได้อย่างมีประสิทธิภาพ โดยใช้ทรัพยากรที่มีได้อย่างคุ้มค่าที่สุด (ใช้พื้นที่หน่วยความจำน้อยที่สุด) อีกทั้งยังใช้เวลาในการประมวลผลน้อยที่สุดด้วย ส่งผลให้คอมพิวเตอร์ทำงานเร็วขึ้นนั่นเอง เช่น การใช้โครงสร้างทรีแบบสมดุลมาเก็บข้อมูลเพื่อเป็นดัชนีในการเข้าถึงข้อมูล เป็นต้น
- การอ่านข้อมูลเพื่อใช้ประมวลผล คอมพิวเตอร์จะอ่านข้อมูลมาเก็บในหน่วยความจำหลัก (หน่วยความจำแรม) และในกรณีที่ข้อมูลมีขนาดมากกกว่าขนาดของหน่วยความจำ คอมพิวเตอร์จะใช้โครงสร้างข้อมูลและอัลกอริทึมเข้ามาช่วยในการจัดการกับข้อมูลเหล่านั้น เพื่อให้สามารถจัดสรรหน่วยความจำได้อย่างเหมาะสมและสามารถทำงานได้อย่างต่อเนื่อง เช่น นำสแต็ก (Stack) มาเก็บเครื่องหมายทางคณิตศาสตร์เมื่อต้องการประมวลผลก็นำข้อมูลที่ต้องใช้มาเก็บไว้ในความจำหลักก่อนการประมวลผล
- การพัฒนาโปรแกรมโดยใช้โครงสร้างข้อมูลและอัลกอริทึมที่เหมาะสม สามารถเพิ่มประสิทธิภาพการทำงานของโปรแกรมได้เช่นกัน (ใช้หน่วยความจำน้อยและประมวลผลได้รวดเร็ว)

ทั้งนี้ ประโยชน์ของโครงสร้างข้อมูลและอัลกอริทึมที่กล่าวมาเป็นเพียงตัวอย่างเล็กน้อยเท่านั้น ซึ่งโครงสร้างข้อมูลและอัลกอริทึมยังสามารถประยุกต์เพื่อใช้งานได้อีกมากมาย ไม่ว่าจะเป็นการเรียงลำดับข้อมูล การค้นหาข้อมูล เป็นต้น

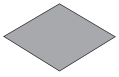
ผังงาน (Flow Chart)

ผังงาน หรือเรียกว่า **โฟลว์ชาร์ต (Flow Chart)** เป็นเครื่องมือที่ใช้ออกแบบระบบงานด้วยสัญลักษณ์ ช่วยให้มีโครงสร้างของระบบงานที่เป็นลำดับขั้นตอน และเข้าใจได้ง่าย

การนำผังงานมาใช้ออกแบบโปรแกรม สามารถตรวจสอบได้ว่า มีลำดับขั้นตอนการทำงานถูกต้องหรือไม่ และสามารถเปลี่ยนแปลงแก้ไขข้อผิดพลาดของระบบงานภายในผังงานได้ง่ายกว่าการหาข้อผิดพลาดที่เกิดจากการเขียนโปรแกรม อีกทั้งยังช่วยลดความสับสนในการพัฒนาโปรแกรมอีกด้วย

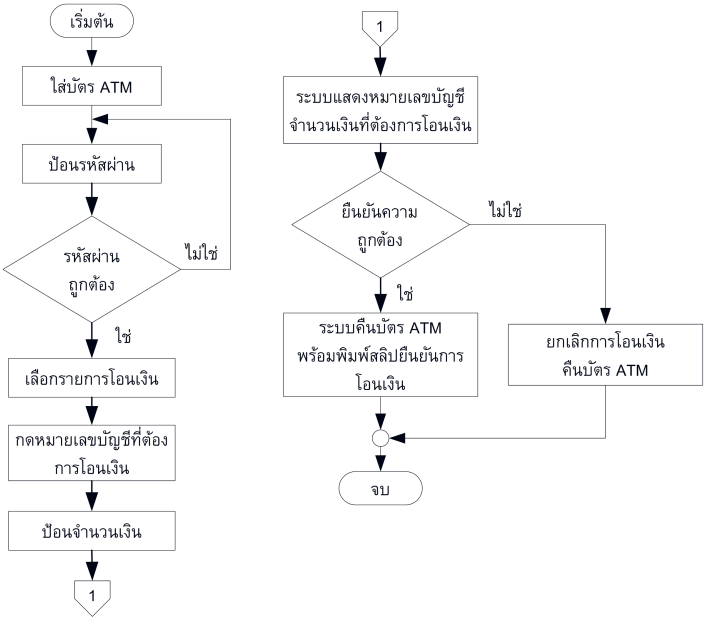
วัตถุประสงค์ที่สำคัญที่สุดในการนำผังงานมาเป็นเครื่องมือในการพัฒนาโปรแกรม คือ เพื่อให้บุคคลอื่นสามารถทำความเข้าใจถึงลำดับขั้นตอนการทำงานของโปรแกรม และผลลัพธ์ที่ได้จากการทำงานของโปรแกรมที่พัฒนาขึ้นได้

สัญลักษณ์ที่นำมาใช้ในเขียนผังงานมีอยู่ 8 สัญลักษณ์ดังนี้

สัญลักษณ์	ความหมายสัญลักษณ์ในการใช้งานในผังงาน
 Terminator	จุดเริ่มต้นและจุดสิ้นสุดของโปรแกรม
 Data	รับข้อมูลเข้ามาในโปรแกรม หรือส่งค่าออกไปจากโปรแกรม
 Process	การประมวลผลหรือการคำนวณของโปรแกรม
 Decision	ตรวจสอบเงื่อนไข แล้วเลือกการทำงานของโปรแกรม
 Document	แสดงผลออกทางเอกสาร
 On-page reference	จุดเชื่อมต่อหลายเส้นทางของโปรแกรมให้เหลือการเข้ามาเพียงเส้นทางเดียว
 Off-page reference	ขึ้นหน้าใหม่หรือสิ้นสุดภายในหนึ่งหน้ากระดาษในกรณีที่ผังงานมีความยาวเกินกว่าที่จะแสดงพอในหนึ่งหน้า
	ลูกศรแสดงทิศทางการทำงานของโปรแกรมและการไหลของข้อมูล

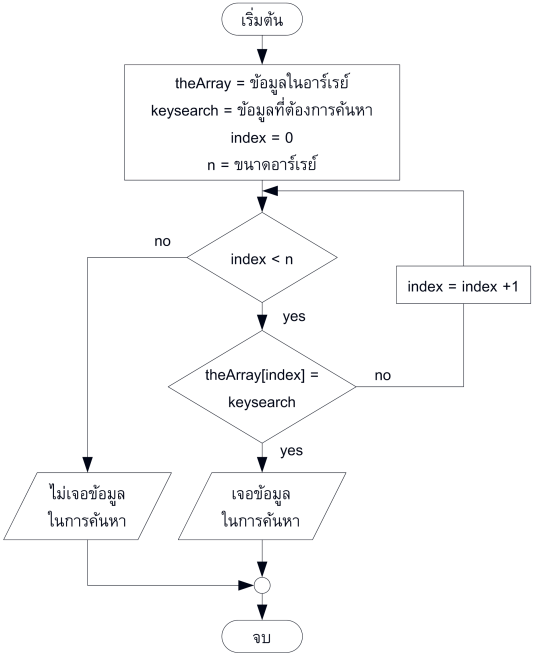
ตัวอย่างที่ 1.3 แสดงผังงานขั้นตอนการใช้ตู้กดเงินอัตโนมัติ (ATM) เพื่อโอนเงิน ดังนี้

รูปที่ 1-1 ผังงานขั้นตอนการใช้ตู้กดเงินอัตโนมัติ (ATM) เพื่อโอนเงิน



ตัวอย่างที่ 1.4 แสดงผังงานอัลกอริทึมค้นหาข้อมูลในอาร์เรย์ขนาด n ข้อมูล ดังนี้

รูปที่ 1-2 ผังงานอัลกอริทึมค้นหาข้อมูลในอาร์เรย์ขนาด n ข้อมูล



โค้ดรหัสเทียม (Pseudo code)

โค้ดรหัสเทียม หรือเรียกว่าซูโดโค้ด (Pseudo code) เป็นรูปแบบการเขียนลำดับขั้นตอนการทำงานของโปรแกรมคอมพิวเตอร์ด้วยการรวมภาษาเขียนกับคำสั่งตรวจสอบเงื่อนไขของภาษาโปรแกรมเข้าไว้ด้วยกัน เพื่ออธิบายโครงสร้างข้อมูลและลำดับขั้นตอนการทำงานของโปรแกรมโดยไม่อ้างอิงภาษาโปรแกรม เพื่อเป็นสื่อกลางแทนการเขียนด้วยโค้ดโปรแกรม (Code Program) ดังตัวอย่างต่อไปนี้

ตัวอย่างที่ 1.5 Pseudo code การค้นหาข้อมูลในอาร์เรย์ ดังนี้

```
1  +searching(in theArray:arrayType,in KeySearch:keyType,in MaxData:integer):boolean
2    for (n = 0 to MaxData-1){
3      if (theArray[n] = KeySearch){
4        data is found then return true
5      }
6    }
7    if (n = MaxData-1){
8      data is not found then return false
9    }
```

อธิบายการทำงานของ Pseudo code

บรรทัดที่ 1 ประกาศฟังก์ชันหรือเมธอด searching โดยกำหนดให้มีตัวแปรอินพุตสามตัว คือ

- theArray ทำหน้าที่เก็บข้อมูลทั้งหมดในรูปแบบของอาร์เรย์
- KeySearch ทำหน้าที่เก็บคีย์ข้อมูลที่ต้องการค้นหา
- MaxData ทำหน้าที่เก็บขนาดของอาร์เรย์

และกำหนดให้ส่งคืนค่าผลการค้นหาข้อมูลเป็นข้อมูลชนิด boolean

บรรทัดที่ 2 วนลูปตั้งแต่ตัวแปร n เท่ากับ 0 จนถึงค่าในตัวแปร MaxData-1

บรรทัดที่ 3-4 ถ้าข้อมูลใน theArray ตำแหน่งที่ n เท่ากับค่าข้อมูล KeySearch แสดงว่าเจอข้อมูลที่ต้องการค้นหา ให้ฟังก์ชันคืนค่าเป็น true

บรรทัดที่ 8-9 ถ้าค่าตัวแปร n เท่ากับ MaxData แสดงว่าไม่พบข้อมูลในอาร์เรย์ให้คืนค่าเป็น false

ตัวอย่างที่ 1.6 Pseudo code แสดงผลการเรียน (Grade) จากคะแนนที่รับเข้ามาดังนี้

```
1  +showgrade(in score:double)
2    if (score >= 80)
3        output “A”
4    else if (score >= 75)
5        output “B+”
6    else if (score >= 70)
7        output “B”
8    else if (score >= 65)
9        output “C+”
10   else if (score >= 60)
11       output “C”
12   else if (score >= 55)
13       output “D+”
14   else if (score >= 50)
15       output “D”
16   else output “F”
```

อธิบายการทำงาน

บรรทัดที่ 1	ประกาศฟังก์ชันหรือเมธอด showgrade กำหนดให้รับค่าคะแนนไว้ในตัวแปร
บรรทัดที่ 2-3	ถ้า score มากกว่าเท่ากับ 80 ให้แสดงเกรด “A”
บรรทัดที่ 4-5	แต่ถ้า score มากกว่าเท่ากับ 75 ให้แสดงเกรด “B+”
บรรทัดที่ 6-7	แต่ถ้า score มากกว่าเท่ากับ 70 ให้แสดงเกรด “B”
บรรทัดที่ 8-9	แต่ถ้า score มากกว่าเท่ากับ 65 ให้แสดงเกรด “C+”
บรรทัดที่ 10-11	แต่ถ้า score มากกว่าเท่ากับ 60 ให้แสดงเกรด “C”
บรรทัดที่ 12-13	แต่ถ้า score มากกว่าเท่ากับ 55 ให้แสดงเกรด “D+”
บรรทัดที่ 14-15	แต่ถ้า score มากกว่าเท่ากับ 50 ให้แสดงเกรด “D”
บรรทัดที่ 16	ถ้า score น้อยกว่า 50 ให้แสดงเกรด “F”

Abstract Data Type

Abstract Data Type หรือเรียกว่า **ADT** เป็นการประกาศถึงคุณสมบัติของโครงสร้างข้อมูลและกลุ่มตัวดำเนินการที่กระทำกับโครงสร้างข้อมูล

คุณสมบัติของโครงสร้างข้อมูลที่กำหนดใน ADT เป็นการแสดงถึงลักษณะของโครงสร้างข้อมูลที่นำมาใช้งาน และกลุ่มตัวดำเนินการจะเป็นฟังก์ชันที่กำหนดการทำงานของโปรแกรมที่กระทำกับโครงสร้างข้อมูล ซึ่งแต่ละกลุ่มตัวดำเนินการจะทำงานอย่างอิสระไม่เกี่ยวเนื่องกัน (ไม่มีการส่งค่าระหว่างกลุ่มตัวดำเนินการ) และมีลักษณะการทำงานที่ชัดเจน

รายการ ADT คือ การรวบรวมลำดับขั้นตอนการทำงาน โดยใช้ตัวเลขแสดงลำดับขั้นตอนการทำงาน ดังตัวอย่างต่อไปนี้

ตัวอย่างที่ 1.7 รายการ ADT แสดงการจัดการรายการสิ่งของในร้านค้าดังนี้

1. Create an empty list. (สร้างรายการว่างเปล่า)
2. Determine whether a list is empty. (ตรวจสอบว่ารายการว่างเปล่า)
3. Determine the number of items on a list. (ตรวจสอบจำนวนสิ่งของในรายการ)
4. Add an item at a given position in the list. (เพิ่มสิ่งของในรายการ ณ ตำแหน่งที่ให้มา)
5. Remove the item at a given position in the list. (ลบสิ่งของในรายการ ณ ตำแหน่งที่ให้มา)
6. Remove all the items from the list. (ลบสิ่งของทั้งหมดในรายการ)
7. Retrieve (get) the item at a given position in the list. (นำสิ่งของกลับคืนมาในตำแหน่งที่ให้มาในรายการ)

จากรายการ ADT นำไปเขียนเป็น Pseudo code ได้ดังนี้

```
+createList()
 //สร้างรายการว่างเปล่า
+isEmpty():boolean{query}
 //ตรวจสอบว่ารายการว่างเปล่า
+size():integer{query}
 //คืนค่าขนาดของรายการทั้งหมด
+add(in index:integer, in item:ListItemType)
 //เพิ่มข้อมูล (item) ในตำแหน่ง index ในรายการ ถ้า 0 <= index < size()
 //ถ้า index = size()+1 ไม่เพิ่มข้อมูลในรายการเนื่องจากรายการเต็ม
+remove(in index:integer)
 //ลบข้อมูลในตำแหน่งของ index ในรายการ ถ้า 0 <= index < size()
 //ถ้า index < 0 ไม่ลบข้อมูลในรายการเนื่องจากรายการว่างเปล่า
+removeAll()
 //ลบข้อมูลทั้งหมดในรายการ
+get(in index:ListItemType){queue}
 //คืนค่า item ในตำแหน่ง index ของรายการ ถ้า 0 <= index < size()
 //ให้เลื่อน index ไปทางซ้าย (ลดค่า index ลงหนึ่งตำแหน่ง) และไม่คืนค่าถ้า index เกินขอบเขตที่กำหนด
```

การออกแบบ ADT เป็นการพัฒนามาจากขบวนการแก้ไขปัญหา ตัวอย่างเช่น ต้องการตรวจสอบวันหยุดทั้งหมดในหนึ่งปีซึ่งมีวิธีการ คือ ดูปฏิทินว่ามีวันหยุดวันใดบ้างในหนึ่งปี ด้วยการตรวจสอบทีละวันว่าวันใดเป็นวันหยุด เป็นต้น

เพื่อให้เกิดความเข้าใจให้ศึกษาตัวอย่างต่อไปนี้

ตัวอย่างที่ 1.8 ฟังก์ชันตรวจสอบวันหยุดในหนึ่งปีมีคำสั่งดังนี้

```
 +listHolidays(in year:integer)
//แสดงวันหยุดทั้งหมดในหนึ่งปีของ year ให้มา
    date = วันที่ของวันแรกของปี
    while(date ไม่ใช่วันแรกของปีถัดไป){
          if(date = วันหยุด){
              write(date + "คือวันหยุด")
          } //end if
       date = วันที่ถัดไป
    }//end while
```

จากฟังก์ชันสามารถเขียนในรูปแบบ ADT เพื่อใช้แก้ไขปัญหาการตรวจสอบวันหยุดในหนึ่งปีได้ดังนี้

1. Determine the date of the first day of a given year. (ตรวจสอบว่าเป็นวันแรกของปีที่ให้มา)
2. Determine whether a date is before another date. (ตรวจสอบวันที่ในการตรวจสอบก่อนวันที่อื่นๆ)
3. Determine whether a date is a holiday. (ตรวจสอบว่าวันที่นี้เป็นวันหยุดหรือไม่)
4. Determine the date of the day that follows a given date. (ตรวจสอบวันที่ถัดไปของวันที่กำหนดให้มา)

จาก ADT การตรวจสอบวันหยุดในหนึ่งปีนำไปเขียนเป็น Pseudo code ได้ดังนี้

```
+firstDay(in year:integer):Date{query}
//คืนค่าวันแรกของปีที่ให้มา
+isBefore(in date1:Date,date2:Date):boolean{query}
//คืนค่า true ถ้า date1 เป็นวันที่มาก่อน date2 นอกเหนือจากนั้นให้คืนค่า false
+isHoliday(in aDate:Date):boolean{query}
//คืนค่า true ถ้า aDate เป็นวันหยุด นอกเหนือจากนั้นให้คืนค่า false
+nextDay(in aDate:Date):Date
//คืนค่าวันที่ถัดไปจากวันที่กำหนดให้มาในตัวแปร aDate
```

ดังนั้น การออกแบบ ADT จะเป็นการแสดงข้อมูลและการเลือกตัวดำเนินการให้เหมาะสมกับการแก้ไขปัญหา โดยที่ตัวดำเนินการนั้นมีการทำงานที่อิสระในการกำหนดคุณสมบัติของ ADT

ประเภทของอัลกอริทึม

การแยกประเภทของอัลกอริทึมแยกตามรูปแบบการแก้ไขปัญหาของอัลกอริทึม โดยแยกประเภทของอัลกอริทึมได้ดังนี้

- **Brute force algorithm** เป็นอัลกอริทึมสำหรับแก้ไขปัญหาโดยสั่งให้ทำงานไปเรื่อยๆ จนกระทั่งได้คำตอบของทุกปัญหา การแก้ไขในรูปแบบนี้ไม่เหมาะสำหรับแก้ไขปัญหาที่มีข้อมูลจำนวนมาก ตัวอย่างอัลกอริทึมที่ใช้หลักการ Brute Force algorithm เช่น การจัดเรียงข้อมูลแบบ Bubble sort, Selection sort เป็นต้น
- **Divide and Conquer algorithm** เป็นอัลกอริทึมที่มีหลักการคิดโดยแยกปัญหาออกเป็นสองส่วน คือ ส่วนที่หนึ่งแบ่งปัญหาออกเป็นส่วนเล็กๆ แล้วแก้ไขปัญหาในส่วนเล็กๆ นั้นก่อน และอีกส่วนนำผลที่ได้จากการแก้ไขปัญหาในส่วนเล็กๆ กลับมาร่วมกันใหม่ ตัวอย่างอัลกอริทึมที่ใช้หลักการ Divide and Conquer algorithm เช่น การจัดเรียงข้อมูลแบบ Quick sort, Merge sort เป็นต้น
- **Decrease and Conquer algorithm** เป็นอัลกอริทึมที่แก้ไขปัญหาด้วยการลดขนาดของปัญหาลง และเลือกขนาดของกลุ่มปัญหาที่ต้องการแก้ไขปัญหา โดยละเว้นปัญหาบางส่วนไว้ก่อน เพื่อจะแก้ปัญหาที่มีขนาดเล็กลงกว่าเดิม เนื่องจากการแก้ไขปัญหาที่มีขนาดเล็กกว่าจะสามารถแก้ไขปัญหาได้ง่ายกว่า ตัวอย่างอัลกอริทึมที่ใช้หลักการ Decrease and Conquer algorithm เช่น การค้นหาข้อมูลแบบไบนารี เป็นต้น

- **Transform and Conquer algorithm** เป็นอัลกอริทึมสำหรับแก้ไขปัญหาโดยการเปลี่ยนรูปแบบของปัญหาที่ต้องการแก้ไขให้อยู่ในรูปแบบอื่นก่อน ด้วยคาดหวังว่าเมื่อเปลี่ยนรูปแบบของปัญหาแล้วจะสามารถแก้ไขปัญหาได้ง่ายและรวดเร็วขึ้น ตัวอย่างในการเปลี่ยนโครงสร้างข้อมูลก่อนการแก้ไขปัญหา เช่น นำข้อมูลที่ต้องการค้นหามาจัดเรียงข้อมูลก่อนที่จะค้นหา เพื่อให้สามารถใช้อัลกอริทึมที่มีประสิทธิภาพค้นหาข้อมูลได้แทนที่จะค้นข้อมูลทีละตัวจากข้อมูลที่กระจัดกระจาย
- **Greedy algorithm** หรือ**อัลกอริทึมแบบละโมบ** เป็นอัลกอริทึมที่มีลักษณะของการแก้ไขปัญหาด้วยการเพิ่มประสิทธิภาพของการแก้ไขปัญหาให้เหมาะสมที่สุด (Optimization problems) ซึ่งเป็นรูปแบบอัลกอริทึมที่พิจารณาคำตอบที่ดีที่สุดและคุ้มค่าที่สุดในการแก้ไขปัญหานั้นๆ ตัวอย่างอัลกอริทึมที่ใช้หลักการ Greedy algorithm ในการแก้ไขปัญหา เช่น ปัญหาการทอนเหรียญ คือ เลือกทอนเหรียญจากหน่วยที่มีขนาดใหญ่ที่สุดก่อน เป็นต้น
- **Dynamic programming algorithm** หรือ**อัลกอริทึมโปรแกรมพลวัต** เป็นอัลกอริทึมที่มีลักษณะของการแก้ไขปัญหาด้วยการแบ่งปัญหาเป็นส่วนเล็กๆ แล้วนำผลของปัญหาเล็กๆ ที่ดีที่สุดนำมาแก้ไขปัญหาใหญ่ที่เรียกกันว่า การแก้ไขปัญหาจากล่างขึ้นบน **(Bottom-up approach)** ตัวอย่างอัลกอริทึมที่ใช้หลักการ Dynamic programming algorithm ในการแก้ไขปัญหา เช่น การหาค่าตัวเลข Fibonacci เป็นต้น
- **Backtracking algorithm** หรือ**อัลกอริทึมย้อนรอยถอยหลัง** เป็นอัลกอริทึมค้นหาเส้นทางทุกเส้นทางที่เป็นไปได้เพื่อหาคำตอบของปัญหาทีละส่วนย่อยว่า คำตอบนั้นเป็นคำตอบที่ถูกต้องหรือไม่ แต่คำตอบนั้นไม่ใช่ส่วนหนึ่งของคำตอบจะถอยหลังกลับมาจุดเดิม และยกเลิกคำตอบนั้นแล้วค้นหาคำตอบใหม่ ตัวอย่างอัลกอริทึมที่ใช้หลักการ Backtracking algorithm เช่น การกำหนดสีให้กับเมืองในแผนที่, การคิดความเป็นไปได้ทั้งหมดของการเดินหมากกระดาน เป็นต้น
- **Branch and bound algorithm** เป็นอัลกอริทึมที่เพิ่มประสิทธิภาพในการแก้ไขปัญหา ด้วยการนำโครงสร้างทรี (Tree) มาเก็บปัญหาย่อยๆ โดยปัญหาหลักจะอยู่ในตำแหน่งบนสุดของ ทรี คือ โหนดราก (root node) และในแต่ละโหนดจะแก้ไขปัญหาของตัวเอง และถ้าแก้ปัญหาถูกต้องก็จะใช้ผลนั้นเป็นข้อมูลในการแก้ไขปัญหาทั้งหมด แต่ถ้าการแก้ไขปัญหาไม่ถูกต้องให้แบ่งปัญหาออกเป็นสองโหนดย่อยเก็บไว้ในตำแหน่งโหนดลูกของโหนดที่แก้ไขปัญหาไม่ถูกต้อง แล้วกลับไปทำใหม่จนกระทั่งทุกโหนดย่อยในทรีสามารถแก้ไขปัญหาได้ทุกโหนด ตัวอย่างอัลกอริทึมที่ใช้หลักการ Branch and bound algorithm มาใช้ในการแก้ไขปัญหา เช่น ปัญหาในการหาเส้นทางที่เหมาะสมให้กับพนักงานขายสินค้าให้สามารถเดินทางได้ครบทุกที่ได้เร็วที่สุด เป็นต้น
- **Recursive algorithm** หรือ**อัลกอริทึมแบบวนซ้ำ** เป็นอัลกอริทึมสำหรับแก้ไขปัญหาขั้นพื้นฐานด้วยการเรียกใช้ตัวเองซ้ำๆ โดยนำข้อมูลปัญหาส่วนย่อยของปัญหาทั้งหมดกลับมาเป็นข้อมูลในการแก้ไขปัญหา ตัวอย่างอัลกอริทึมแบบเรียกซ้ำ เช่น การหาค่า Factorial, อัลกอริทึมบวกข้อมูลตัวเลขที่อยู่ในกลุ่ม เป็นต้น
- **Randomized algorithm** หรือ**อัลกอริทึมแบบสุ่ม** เป็นอัลกอริทึมที่ใช้หลักการสุ่มข้อมูล แล้วนำข้อมูลที่สุ่มเลือกขึ้นมาได้กระทำกับอัลกอริทึมเพื่อให้ได้ผลตามที่ต้องการตัวอย่างอัลกอริทึมที่ใช้หลักการ Randomized algorithms ไปใช้งาน เช่น พยายามหาข้อมูลที่สำคัญที่สุดด้วยการเลือกข้อมูลจากการสุ่มด้วยการหาร หรือการจัดเรียงข้อมูลแบบ Quicksort ด้วยการสุ่มตัวเลขหนึ่งตัวขึ้นมาเพื่อใช้สำหรับเปรียบ (pivot) ข้อมูลที่ยังไม่ได้จัดเรียงข้อมูล เป็นต้น

สรุปเนื้อหาบทที่ 1

- โครงสร้างข้อมูล คือ การจัดการข้อมูลในหน่วยความจำภายในเครื่องคอมพิวเตอร์
- อัลกอริทึม คือ ลำดับขั้นตอนการทำงานเพื่อใช้ในการแก้ไขปัญหา
- ผังงาน เป็นเครื่องมือที่ช่วยออกแบบขั้นตอนการทำงานด้วยสัญลักษณ์
- โค้ดรหัสเทียม เป็นโครงสร้างรหัสที่มีการร่วมกันทั้งภาษาเขียนกับภาษาคอมพิวเตอร์ เพื่อใช้อธิบายโครงสร้างและลำดับขั้นตอนการทำงานของโปรแกรม โดยไม่อ้างอิงภาษาการเขียนโปรแกรมภาษาใดภาษาหนึ่ง
- Abstract Data Type เป็นการบอกถึงคุณสมบัติของโครงสร้างข้อมูลและกลุ่มตัวดำเนินการที่กระทำกับโครงสร้างข้อมูล
- ประเภทของอัลกอริทึมแยกได้เหมือนกับแยกรูปแบบในการแก้ไขปัญหาของโปรแกรม การแยกประเภทของอัลกอริทึมเพื่อแยกรูปแบบอัลกอริทึมในการแก้ไขปัญหานั่นเอง

แบบฝึกหัดท้ายบทที่ 1

ให้เขียนขั้นตอนวิธี ผังงาน และโค้ดรหัสเทียมของโปรแกรมต่อไปนี้

1. คำนวณเลขยกกำลังกำหนดให้รับตัวเลข 2 ค่า คือ ค่าเลขฐานและเลขยกกำลัง เพื่อนำมาคำนวณเลขยกกำลัง
2. เปลี่ยนค่าหน่วยเวลาจากหน่วยวินาทีที่รับเข้ามาให้เป็นหน่วยชั่วโมง นาที และวินาที
3. คำนวณอัตราแลกเปลี่ยนจากข้อมูลตัวเลขที่ป้อนเข้ามากำหนดให้เป็นสกุลเงินบาท เปลี่ยนเป็นสกุลเงินประเทศสหรัฐอเมริกา ญี่ปุ่น และจีน กำหนดให้อ้างอิงอัตราแลกเปลี่ยนปัจจุบันในการซื้อขาย
4. รับข้อมูลตัวเลขจำนวน 20 ตัว กำหนดให้นำเฉพาะข้อมูลที่เป็นเลขคู่มาบวกกันและแสดงผลการบวก
5. คำนวณการถอนเงิน โดยให้รับข้อมูลตัวเลขเข้ามา 2 จำนวน คือ ยอดเงินที่ลูกค้าซื้อและจำนวนเงินที่ลูกค้าจ่าย แล้วให้คำนวณว่าต้องถอนแบงก์ 1,000 500 100 50 20 เหรียญ 10 5 2 1 .50 และ .25 บาท จำนวนอย่างละเท่าไร
6. รับคะแนนสอบของนักศึกษาจำนวน 20 คนว่ามีนักศึกษาที่ได้เกรดแต่ละเกรดจำนวนกี่คน และหาค่าเฉลี่ยว่านักศึกษากลุ่มนี้มีค่าคะแนนเฉลี่ยอยู่ที่เกรดอะไร โดยกำหนดค่าคะแนนในแต่ละช่วงเกรดดังนี้

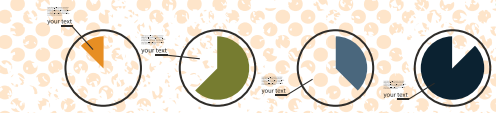
คะแนน	เกรด
ต่ำกว่า 50	F
50-54	D
55-59	D+
60-64	C
65-69	C+
70-74	B
75-79	B+
80 ขึ้นไป	A

7. รับข้อมูลอายุจำนวน 20 คน กำหนดให้นับจำนวนคนในแต่ละช่วงวัยและหาค่าเฉลี่ยจากอายุที่รับเข้ามาว่าอยู่ในช่วงวัยไหน โดยแยกแต่ละช่วงวัยดังนี้
   - วัยเด็กอายุน้อยกว่า 15 ปี
   - วัยรุ่นอายุมากกว่าและเท่ากับ 15 ปี ถึง 20 ปี
   - วัยผู้ใหญ่ตอนต้นอายุมากกว่า 20 ปี ถึง 40 ปี
   - วัยกลางคนอายุมากกว่า 40 ปี ถึง 60 ปี
   - วัยชราอายุมากกว่า 60 ปีขึ้นไป
8. รับข้อมูลตัวเลขจำนวนชั่วโมงการทำงานในแต่ละสัปดาห์จำนวน 4 สัปดาห์ และประเภทของงานที่ทำงานเพื่อคำนวณค่าจ้าง โดยกำหนดให้ประเภทของงานและอัตราค่าจ้างดังนี้

ลำดับประเภทงาน	ชื่อประเภทงาน	อัตราค่าจ้าง/ชั่วโมง
0	แรงงานระดับใช้กำลัง	20
1	แรงงานระดับฝีมือ	25
2	แรงงานระดับชำนาญเฉพาะด้าน	30
3	แรงงานระดับฝีมือชำนาญเฉพาะด้าน	35

9. หาผลคูณจากเมตริกซ์ 2 เมตริกซ์กำหนดเมตริกซ์ที่ 1 มีขนาด n x m และเมตริกซ์ที่ 2 มีขนาด m x n ผลลัพธ์ของการคูณเมตริกซ์ทั้งสองมีขนาด n x n

Data Structure & Algorithm



CHAPTER

02

การวิเคราะห์ประสิทธิภาพของอัลกอริทึม (Performance Analysis of Algorithms)

ในบทนี้จะศึกษาเกี่ยวกับการวิเคราะห์ประสิทธิภาพของอัลกอริทึม (Performance Analysis) ซึ่งเป็นพื้นฐานที่มีความสำคัญเป็นอย่างยิ่งในการเรียนรู้และใช้งานโครงสร้างข้อมูลและอัลกอริทึม

คณิตศาสตร์พื้นฐานสำหรับการวิเคราะห์ประสิทธิภาพของอัลกอริทึม

สมการคณิตศาสตร์ที่ใช้วิเคราะห์ประสิทธิภาพของอัลกอริทึมมี 3 รูปแบบ คือ ลอการิทึม, การหาผลรวม และเลขยกกำลัง โดยสรุปได้ดังนี้

ลอการิทึม (Logarithms)

- ลอการิทึมของค่า y ฐาน b จะเขียนอยู่ในรูปลอการิทึมได้ คือ $\log_b y = x$

  ดังนั้น ถ้า $\log_b y = x$ แล้ว $b^x = y$

  ตัวอย่างการหา $\log_b y = x$

  $4 = 2^2$ ดังนั้น $\log_2 4 = 2$

  $16 = 2^4$ ดังนั้น $\log_2 16 = 4$

  $1000 = 2^{10}$ ดังนั้น $\log_2 1000 = 10$