April 2023

Fundamental IT Engineer Examination (Afternoon)

ให้ทำข้อสอบตามรายละเอียดต่อไปนี้

หมายเลขคำถาม	Q1	Q2 – Q5	Q6	Q7, Q8
การเลือกคำถาม	คำถามบังคับ	เลือก 2 ใน 4	คำถามบังคับ	เลือก 1 ใน 2
เวลาสอบ	13:30 – 16:00 (150 นาที)			

ข้อปฏิบัติ:

1. ให้ใช้ดินสอตอบ ถ้าต้องการเปลี่ยนคำตอบ ให้ลบคำตอบเก่าให้สะอาดก่อนโดยไม่ให้มีคราบยางลบหลงเหลือ
2. ให้ทำเครื่องหมายบอกข้อมูลผู้สอบและคำตอบของแบบทดสอบ ตามคำสั่งด้านล่างอย่างเคร่งครัด หากทำเครื่องหมายไม่เหมาะสม คำตอบของท่านอาจไม่ได้รับการตรวจ ห้ามทำเครื่องหมาย หรือเขียนตอบนอกพื้นที่ที่กำหนดไว้

(1) **หมายเลขผู้สอบ (Examinee Number)**
ให้เขียนหมายเลขผู้สอบลงในช่องที่เตรียมไว้ให้ และทำเครื่องหมายในช่องว่างที่เหมาะสมที่อยู่ใต้ตัวเลขแต่ละตัว

(2) **วันเกิด (Date of Birth)**
ให้เขียนวันเกิดของผู้สอบ (เป็นตัวเลข) ลงในช่องที่เตรียมไว้ ให้ตรงกับที่พิมพ์อยู่ในบัตรเข้าห้องสอบ และทำเครื่องหมายในช่องว่างที่เหมาะสมที่อยู่ใต้ตัวเลขแต่ละตัว

(3) **การเลือกคำตอบ**
สำหรับ Q2 ถึง Q5 และ Q7 กับ Q8 ให้เลือก Ⓢ ของข้อที่คุณเลือกที่จะตอบในช่อง "Selection Column" บนกระดาษคำตอบของคุณ

(4) **คำตอบ**
ให้ทำเครื่องหมายตรงคำตอบที่เลือกตามตัวอย่างที่แสดงอยู่ด้านล่าง

[คำถามตัวอย่าง]
ข้อใดต่อไปนี้เป็นสิ่งที่ควรใช้ทำเครื่องหมายเพื่อเลือกข้อที่ต้องการในกระดาษคำตอบ

กลุ่มคำตอบ
a) ปากกาลูกลื่น b) สีเทียน c) ปากกาหมึกซึม d)ดินสอ

เนื่องจากคำตอบที่ถูกคือ "d)" (ดินสอ), ดังนั้นให้ทำเครื่องหมายดังแสดงด้านล่างนี้:

[ตัวอย่างคำตอบ]

Sample	ⓐ ⓑ ⓒ ● ⓔ ⓕ ⓖ ⓗ ⓘ ⓙ

ห้ามเปิดดูข้อสอบก่อนได้รับอนุญาต
ข้อสงสัยที่เกี่ยวข้องกับคำถามในข้อสอบอาจจะไม่ถูกตอบ

Notations used in the pseudo-language

In questions that use pseudo-language, the following notations are used unless otherwise stated:

[Declaration, comment, and process]

Notation		Description
type: *var1*, ... , *array1*[], ...		Declares variables *var1*, ... , and/or arrays *array1*[], ... , by data *type* such as INT and CHAR.
FUNCTION: *function*(*type*: *arg1*, ...)		Declares a *function* and its arguments *arg1*, ...
/* comment */ or // comment		Describes a comment.
Process	*variable* ← *expression*;	Assigns the value of the *expression* to the *variable*.
	function (*arg1*, ...);	Calls the *function* by passing / receiving the arguments *arg1*, ...

	IF (*condition*) {
	process1
	}
	ELSE {
	process2
	}
		Indicates the selection process.
		If the *condition* is true, then *process1* is executed.
		If the *condition* is false, then *process2* is executed, when the optional ELSE clause is present.

	WHILE (*condition*) {
	process
	}
		Indicates the "WHILE" iteration process.
		While the *condition* is true, the *process* is executed repeatedly.

	DO { *process* } WHILE (*condition*);	Indicates the "DO-WHILE" iteration process. The *process* is executed once, and then while the *condition* is true, the *process* is executed repeatedly.

	FOR (*init*; *condition*; *incr*) {
	process
	}
		Indicates the "FOR" iteration process.
		While the *condition* is true, the *process* is executed repeatedly.
		At the start of the first iteration, the process *init* is executed before testing the *condition*.
		At the end of each iteration, the process *incr* is executed before testing the *condition*.

[Logical constants]

true, false

[Operators and their precedence]

Type of operation	Unary	Arithmetic		Relational	Logical	
Operators	+, −, not	×, ÷, %	+, −	>, <, ≥, ≤, =, ≠	and	or
Precedence	High ←———					———→ Low

Note: With division of integers, an integer quotient is returned as a result.
The "%" operator indicates a remainder operation.

คำถาม **Q1** เป็น **คำถามบังคับ**

Q1. อ่านคำอธิบายของกระบวนการเปิดการเชื่อมต่อ TLS เวอร์ชัน 1.2 ต่อไปนี้ แล้วตอบคำถามย่อย

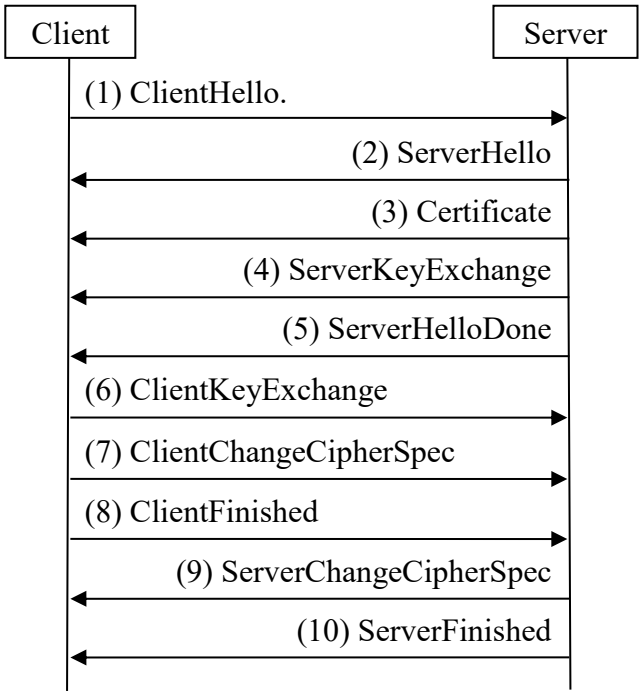
HTTPS คือการนำ HTTP มารวมเข้ากับ Secure Socket Layer (SSL) หรือ Transport Layer Security (TLS) โดย TLS เป็นโพรโทคอลที่สืบทอดมาจาก SSL และจัดเป็นมาตรฐานในปัจจุบัน โพรโทคอลนี้ทำหน้าที่ด้านการเข้ารหัสลับและสามารถนำมาใช้เพื่อสร้างการเชื่อมต่อการสื่อสารที่ปลอดภัยระหว่างเว็บเบราว์เซอร์กับเว็บเซิร์ฟเวอร์โดยการใช้แพคเก็ต TLS มาห่อหุ้มข้อมูล HTTP โดย HTTPS จะช่วยให้ไคลเอนต์ซึ่งในที่นี้คือเว็บเบราว์เซอร์ สามารถสื่อสารกับเว็บเซิร์ฟเวอร์ได้อย่างเป็นความลับ รวมทั้งยังพิสูจน์ตัวจริงของเซิร์ฟเวอร์ได้ด้วย ซึ่งตัวตนของเว็บเซิร์ฟเวอร์นั้นต้องได้รับการยืนยันล่วงหน้าด้วยการส่งใบรับรองเว็บเซิร์ฟเวอร์ (web server certificate) ไปให้ผู้ออกใบรับรองซึ่งเป็นบุคคลที่สามที่เป็นที่ยอมรับ (CA: Certification Authority) โดยข้อมูลที่ต้องถูกส่งไปเพื่อทำการยืนยันตัวตนนั้นได้แก่ชื่อโดเมนฉบับเต็ม (FQDN: Fully Qualified Domain Name) และกุญแจสาธารณะ (public key) ที่เป็นคู่กับกุญแจส่วนตัว (private key) ซึ่งถูกติดตั้งอยู่บนเว็บเซิร์ฟเวอร์ จากนั้น CA จะใช้กุญแจส่วนตัวของตนเซ็นรับรองใบรับรองเว็บเซิร์ฟเวอร์นั้นพร้อมทั้งกำหนดระยะเวลาที่สามารถใช้งานได้ และในที่สุด ใบรับรองที่ถูกเซ็นกำกับแล้วนั้น ก็จะถูกส่งคืนไปเพื่อติดตั้งลงในเว็บเซิร์ฟเวอร์ดังกล่าว รูปแบบมาตรฐานของใบรับรองดังกล่าวนี้คือ X.509

โดยทั่วไปแล้ว TLS มุ่งเป้าไปที่การพิสูจน์ตัวจริงของเซิร์ฟเวอร์ แล้วจึงเริ่มทำการเข้ารหัสระหว่างไคลเอนต์กับเซิร์ฟเวอร์ ดังนั้นจึงมีหลายกลไกที่สามารถนำมาใช้เพื่อให้มั่นใจว่าทั้งไคลเอนต์และเซิร์ฟเวอร์มีกุญแจเดียวกันสำหรับใช้ในเซสชันหรือการเชื่อมต่อนั้น ๆ กุญแจนี้เรียกว่ากุญแจเซสชัน (session key) หรืออาจเรียกว่ากุญแจที่ใช้ร่วมกัน (shared key) ซึ่งจะถูกคำนวณขึ้นมาจากข้อมูลที่ได้แลกเปลี่ยนกันระหว่างไคลเอนต์ (Client) กับเซิร์ฟเวอร์ (Server) โดยกระบวนการนี้จะเกิดขึ้นในระหว่างการเปิดการเชื่อมต่อ TLS
รูปที่ 1 แสดงกระบวนการเปิดการเชื่อมต่อ TLS ที่ได้สรุปให้เรียบง่ายขึ้น

[กระบวนการเปิดการเชื่อมต่อ TLS]

(1) ClientHello: ไคลเอนต์ส่งข้อความ client hello ซึ่งมีข้อมูลความสามารถในการเข้ารหัสลับต่าง ๆ ของตน เวอร์ชันสูงที่สุดของ TLS ที่ไคลเอนต์รองรับ และข้อมูลอื่น ๆ ที่เกี่ยวข้อง รวมไปถึงตัวเลขที่ถูกสุ่มขึ้นมาซึ่งจะถูกนำใช้สร้างมาสเตอร์ซีเครต (master secret) ไปยังเซิร์ฟเวอร์

(2) ServerHello: เซิร์ฟเวอร์เลือกค่ากำหนดต่าง ๆ สำหรับการเชื่อมต่อ ซึ่งรวมถึงกลไกการเข้ารหัสลับและการบีบอัดข้อมูลที่รองรับโดยทั้งเซิร์ฟเวอร์และไคลเอนต์ จากนั้นจึงสุ่มตัวเลขขึ้นมาอีกตัวหนึ่ง แล้วจึงส่งข้อมูลเหล่านี้ไปยังไคลเอนต์



รูปที่ 1 กระบวนการเปิดการเชื่อมต่อ TLS

(3) Certificate: ข้อความนี้โดยทั่วไปแล้วจะบรรจุสายโซ่ใบรับรอง X.509 (X.509 certificate chain) ซึ่งรวมถึงกุญแจสาธารณะของเซิร์ฟเวอร์และข้อมูลอื่น ๆ เอาไว้ เมื่อไคลเอนต์ได้รับข้อความนี้แล้ว ก็จะสามารถตรวจสอบใบรับรอง X.509 ของเซิร์ฟเวอร์กับ CA ที่เกี่ยวข้อง เนื่องจากใบรับรองของเซิร์ฟเวอร์นั้นถูกเซ็นกำกับด้วย [A] ดังนั้นจึงสามารถตรวจสอบเพื่อยืนยันได้ด้วยกุญแจสาธารณะของ CA ซึ่งหากพิสูจน์ตัวจริงไม่ผ่านแล้วนั้น ไคลเอนต์ก็จะแจ้งเตือนผู้ใช้ว่าใบรับรองไม่ถูกต้อง หรือไม่สามารถยืนยันความถูกต้องได้ และจะถามผู้ใช้ว่าต้องการดำเนินการต่อหรือไม่

(4) ServerKeyExchange: ข้อความนี้บรรจุข้อมูลเพิ่มเติมที่จำเป็นต่อการสร้างมาสเตอร์ซีเครตตามชุดการเข้ารหัส (cipher suite) ที่ได้เลือกไว้ แต่ข้อความนี้อาจไม่มีความจำเป็นต้องใช้และอาจไม่ถูกส่งไปหากกลไกแลกเปลี่ยนกุญแจ (key exchange) ไม่ต้องใช้ข้อมูลเพิ่มเติมใด ๆ

(5) ServerHelloDone: ข้อความนี้ระบุให้ทราบว่าเซิร์ฟเวอร์ได้ส่งข้อมูลต่าง ๆ ไปครบทั้งหมดแล้ว จากนั้นจึงจะรอการตอบกลับจากไคลเอนต์

(6) ClientKeyExchange: ข้อความนี้บรรจุข้อมูลต่าง ๆ ที่สร้างขึ้นจากทางไคลเอนต์ที่จำเป็นต่อการสร้างมาสเตอร์ซีเครตตามชุดการเข้ารหัสที่ได้เลือกไว้ อย่างไรก็ตาม เมื่อเปรียบเทียบกับข้อความ ServerKeyExchange แล้ว ข้อความ ClientKeyExchange นี้เป็นข้อความบังคับ (mandatory)

(7) ClientChangeCipherSpec: ไคลเอนต์ส่งข้อความนี้ไปยังเซิร์ฟเวอร์เพื่อแจ้งว่าไคลเอนต์มีข้อมูลเพียงพอที่จะสร้างกุญแจสำหรับการเข้ารหัส (encryption key) รวมทั้งค่ากำหนดต่าง ๆ เพื่อเริ่มต้นการเข้ารหัสแล้ว โดยไคลเอนต์จะสลับไปเข้ารหัสลับหลังจากข้อความนี้

(8) ClientFinished: ไคลเอนต์ส่งข้อความนี้เพื่อแจ้งว่าการเปิดการเชื่อมต่อเสร็จสิ้นแล้ว ข้อความนี้บรรจุแฮช (hash) ของข้อความทั้งหมดที่เคยรับและส่งก่อนหน้านี้รวมเข้ากับมาสเตอร์ซีเครต ทำให้เซิร์ฟเวอร์สามารถยืนยันความถูกต้องสมบูรณ์ของกระบวนการเปิดการเชื่อมต่อได้

(9) ServerChangeCipherSpec: เซิร์ฟเวอร์ส่งข้อความนี้ไปยังไคลเอนต์เพื่อแจ้งว่าเซิร์ฟเวอร์มีข้อมูลเพียงพอที่จะสร้างกุญแจสำหรับการเข้ารหัส (encryption key) รวมทั้งค่ากำหนดต่าง ๆ เพื่อเริ่มต้นการเข้ารหัสแล้ว โดยเซิร์ฟเวอร์จะสลับไปเข้ารหัสลับหลังจากข้อความนี้

(10) ServerFinished: เซิร์ฟเวอร์ส่งข้อความนี้เพื่อแจ้งว่าการเปิดการเชื่อมต่อเสร็จสิ้นแล้ว ข้อความนี้บรรจุแฮช (hash) ของข้อความทั้งหมดที่เคยรับและส่งก่อนหน้านี้รวมเข้ากับมาสเตอร์ซีเครต ทำให้ไคลเอนต์สามารถยืนยันความถูกต้องสมบูรณ์ของกระบวนการเปิดการเชื่อมต่อได้

ตารางที่ 1 แสดงการสรุปกระบวนการดังกล่าวข้างต้น

ตารางที่ 1 สรุปกระบวนการเปิดการเชื่อมต่อ TLS

ขั้นตอน	คำอธิบาย
(ไม่แสดง)	แลกเปลี่ยนข้อมูลความสามารถที่มี แล้วจึงเลือกค่ากำหนดในการเชื่อมต่อที่ต้องการ ซึ่งจะเสร็จสิ้นในขั้นตอนที่ [B]
(ไม่แสดง)	ตรวจสอบความถูกต้องของใบรับรองหรือทำการพิสูจน์ตัวจริงด้วยวิธีการอื่น
(4), (5), (6), (7) และ (9)	ทำความตกลงร่วมกันถึงมาสเตอร์ซีเครตที่จะถูกใช้เป็น [C] ทั้งนี้ เซิร์ฟเวอร์จะมีข้อมูลเพียงพอที่จะสร้างมาสเตอร์ซีเครตหลังจากได้รับข้อความในขั้นตอนที่ [D]
(8) และ (10)	เริ่มส่งข้อความผ่านช่องทางการสื่อสารที่ถูกเข้ารหัสลับ และทำการยืนยันข้อความต่าง ๆ ที่ถูกใช้ในกระบวนการนี้เพื่อให้มั่นใจว่าไม่มีการแทรกแซงจากมือที่สาม

ในที่นี้ ขั้นตอนที่ (3), (4) และ (6) สามารถนำมาพิจารณารวมกันเป็นหนึ่งกระบวนการที่เรียกว่าการแลกเปลี่ยนกุญแจ (key exchange) ได้ เป้าหมายของการแลกเปลี่ยนกุญแจคือการสร้างพรีมาสเตอร์ซีเครต (premaster secret) ซึ่งจะถูกนำไปใช้สร้างมาสเตอร์ซีเครตต่อไป หนึ่งในอัลกอริทึมในการแลกเปลี่ยนกุญแจที่เป็นที่รู้จักกันดีคือ การแลกเปลี่ยนกุญแจแบบ RSA โดยกลไกของการแลกเปลี่ยนกุญแจแบบ RSA นั้นถูกอธิบายไว้ด้านล่างนี้:

[การแลกเปลี่ยนกุญแจ RSA]

RSA เป็นอัลกอริทึมแลกเปลี่ยนกุญแจที่เป็นมาตรฐานสากล ซึ่งในกรณีนี้ ไคลเอนต์เป็นผู้ทำหน้าที่สร้างพรีมาสเตอร์ซีเครตแล้วจึงเข้ารหัสลับด้วย [E] โดยผลที่ได้จะถูกส่งไปยังเซิร์ฟเวอร์ในข้อความ ClientKeyExchange จากนั้น เซิร์ฟเวอร์จะถอดรหัสพรีมาสเตอร์ซีเครตแล้วนำไปใช้สร้างมาสเตอร์ซีเครตในภายหลัง

ความเข้มแข็งของ HTTPS มาจากข้อเท็จจริงที่ว่ามาสเตอร์ซีเครตหรือกุญแจเซสชันต้องถูกสร้างขึ้นมาใหม่ในทุกครั้งที่เกิดเซสชันใหม่แม้ว่าทั้งกุญแจส่วนตัวและกุญแจสาธารณะของเว็บเซิร์ฟเวอร์นั้นไม่มี

การเปลี่ยนแปลง การเข้ารหัสลับกุญแจสาธารณะถูกใช้เฉพาะในช่วงเริ่มต้นเซสชันสำหรับกระบวนการแลกเปลี่ยนกุญแจเท่านั้น การสื่อสารต่อ ๆ มาในเซสชันนั้นจะดำเนินการด้วยกุญแจแบบสมมาตร (symmetric key) ดังนั้น แม้ว่าข้อมูลการสื่อสารจะถูกดักจับและกุญแจที่ใช้ในเซสชันนั้นจะถูกค้นพบ กุญแจดังกล่าวก็จะไม่สามารถนำไปใช้กับเซสชันอื่นได้ แต่ถ้าหาก [F] รั่วไหลออกไป เซสชันที่เกี่ยวข้องทั้งหมดไม่ว่าจะเป็นในอดีตหรือในอนาคตจะตกอยู่ในอันตราย เนื่องจากสามารถนำไปใช้เพื่อหากุญแจเซสชันต่าง ๆ จากข้อมูลการสื่อสารที่ดักจับได้

อีกกลไกในการแลกเปลี่ยนกุญแจที่เหนือกว่าได้แก่การแลกเปลี่ยนคีย์ชั่วคราวดิฟฟีเฮลแมนด้วย RSA (DHE-RSA: Diffie-Hellman ephemeral key exchange with RSA) วิธีนี้ช่วยให้สามารถแลกเปลี่ยนค่าความลับที่ใช้ร่วมกัน (shared secret) ได้ในช่องทางการสื่อสารที่ไม่ปลอดภัย โดยใช้ฟังก์ชันทางคณิตศาสตร์ที่คำนวณขึ้นมาได้ง่ายในทิศทางเดียวแต่ยากต่อการหาค่าย้อนกลับแทนที่จะใช้การเข้ารหัสลับกุญแจสาธารณะ ด้วยวิธีนี้ RSA จะถูกใช้เฉพาะสำหรับการเซ็นกำกับการแลกเปลี่ยนค่ากำหนดต่าง ๆ เท่านั้นโดยไม่ถูกนำมาใช้กับการเข้ารหัสลับ โดยค่าความลับที่ใช้ร่วมกันที่จะกำหนดขึ้นมานั้นจะถูกสุ่มเลือกขึ้นมาใหม่ในทุกเซสชัน จึงยากต่อการถอดรหัสลับเนื้อหาในเซสชันนั้น ๆ แม้ผู้ไม่หวังดีจะมี [F] ที่รั่วไหลออกไป ดังนั้น จึงควรคำนึงถึงกลไกการแลกเปลี่ยนกุญแจในลักษณะนี้ในการนำไปใช้งานจริง

คำถามย่อย

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในคำอธิบายข้างต้น

กลุ่มคำตอบสำหรับ A, C, E และ F

a) ตัวเลขสุ่ม (random number)
b) กุญแจส่วนตัวของ CA
c) กุญแจสาธารณะของ CA
d) กุญแจส่วนตัวของเซิร์ฟเวอร์
e) กุญแจสาธารณะของเซิร์ฟเวอร์
f) กุญแจเซสชัน

กลุ่มคำตอบสำหรับ B และ D

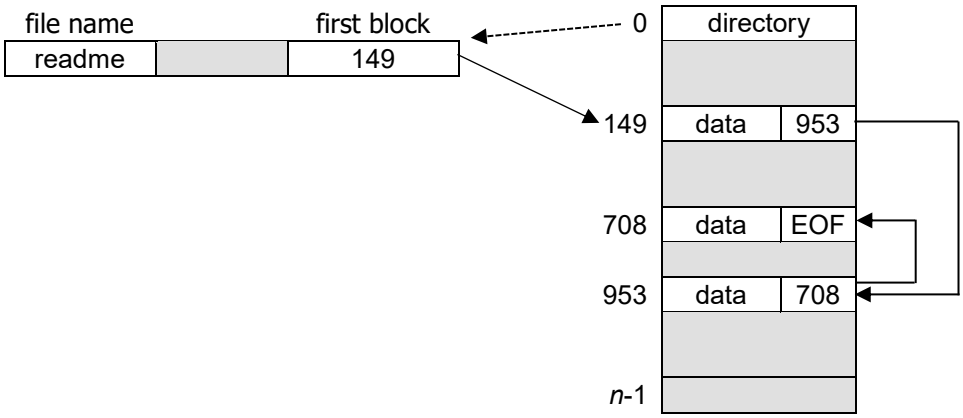
a) (2) ServerHello
b) (3) Certificate
c) (4) ServerKeyExchange
d) (5) ServerHelloDone
e) (6) ClientKeyExchange
f) (7) ClientChangeCipherSpec
g) (8) ClientFinished
h) (10) ServerFinished

สำหรับข้อสอบข้อที่ **Q2** ถึง **Q5 ให้เลือกทำเพียงสอง**ในสี่ข้อ
จากนั้น ให้ระบายทับในวงกลม Ⓢ ในกระดาษคำตอบสำหรับข้อที่เลือกทำ
หากเลือกตั้งแต่สามข้อขึ้นไป จะตรวจให้คะแนนเฉพาะสองข้อแรกเท่านั้น

Q2. อ่านคำอธิบายเกี่ยวกับระบบไฟล์ต่อไปนี้ แล้วตอบคำถามย่อย 1 และ 2
หมายเหตุ: ในคำถามนี้ หนึ่งกิโลไบต์ (1 kB) คือ 1024 ไบต์

ไฟล์และไดเรกทอรีที่ถูกจัดเก็บอยู่ในอุปกรณ์จัดเก็บข้อมูลสามารถนำมาบริหารจัดการได้หลายวิธี ขึ้นอยู่กับระบบไฟล์ (file system) ที่ใช้
อุปกรณ์จัดเก็บข้อมูลนั้น โดยทั่วไปแล้วจะถูกฟอร์แมตให้อยู่ในรูปแบบของบล็อก (block) จำนวนมากที่มีขนาดเท่า ๆ กัน โดยไฟล์และไดเรกทอรีทั้งหมดจะถูกใส่ลงในบล็อกที่ว่างอยู่ในอุปกรณ์จัดเก็บข้อมูลนั้น ๆ อย่างไรก็ตาม หากไฟล์มีขนาดใหญ่กว่าขนาดของบล็อก ไฟล์นั้นก็จะถูกแบ่งออกเป็นหลาย ๆ ส่วนแล้วจึงใส่ลงในบล็อกต่าง ๆ ที่ว่างอยู่ ตัวอย่างเช่นในระบบไฟล์ที่แต่ละบล็อกมีขนาด 4kB ไฟล์ที่มีขนาด 10 kB จะถูกแบ่งออกเป็นสามส่วนแล้วใส่ลงในสามบล็อก

[วิธีจัดสรรพื้นที่แบบเชื่อมโยง (Linked allocation method)]
ในวิธีจัดสรรพื้นที่แบบเชื่อมโยง รายการของไฟล์จะถูกเก็บในตารางที่เรียกว่าไดเรกทอรี (directory) โดยแต่ละรายการจะจัดเก็บชื่อไฟล์ (file name) และข้อมูลอื่น ๆ รวมทั้งหมายเลขบล็อกที่แสดงถึงบล็อกแรก (first block) ของไฟล์
นอกจากข้อมูลของไฟล์นั้น ๆ แล้ว แต่ละบล็อกยังมีหมายเลขของบล็อกที่เก็บข้อมูลส่วนต่อไปและเก็บต่อเนื่องไปในแบบสายโซ่จนถึงบล็อกสุดท้าย ซึ่งจะเก็บค่าสิ้นสุดไฟล์ (EOF: End of file) แทนการเก็บหมายเลขบล็อก



หมายเหตุ: ส่วนที่แรเงาถูกซ่อนไว้ และ n คือจำนวนของบล็อกที่มีในอุปกรณ์จัดเก็บข้อมูล
รูปที่ 1 ตัวอย่างของระบบไฟล์ที่ใช้วิธีจัดสรรพื้นที่แบบเชื่อมโยง

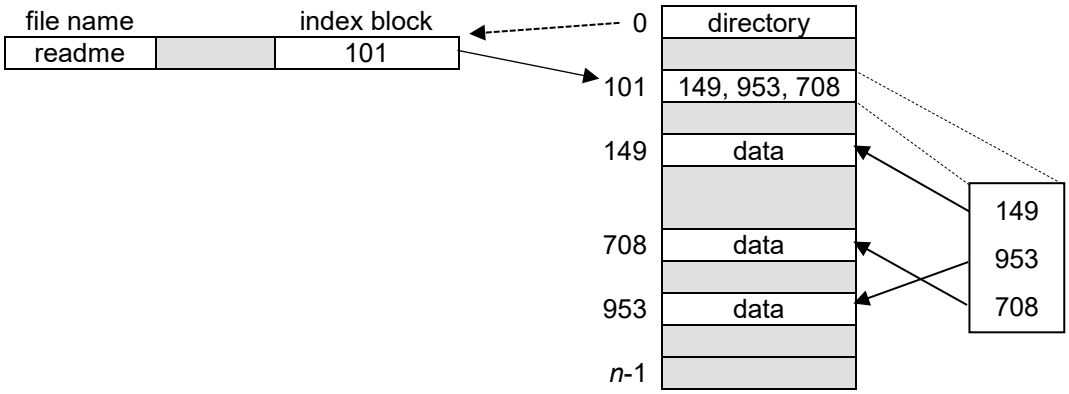
ในรูปที่ 1 ส่วนบนซ้ายแสดงถึงรายการหนึ่งในไดเรกทอรีที่แสดงถึงไฟล์ชื่อ readme และทางด้านขวาแสดงถึงบล็อกต่าง ๆ ในอุปกรณ์จัดเก็บข้อมูล ลูกศรแสดงถึงการเชื่อมโยงจากหมายเลขบล็อกไปยังบล็อกที่บรรจุข้อมูลอยู่จริงบนอุปกรณ์จัดเก็บข้อมูล โดยไฟล์ readme ถูกจัดเก็บอยู่ในบล็อก 149, 953, และ 708 ตามลำดับ

วิธีจัดสรรพื้นที่แบบนี้มีความเรียบง่ายและง่ายต่อการพัฒนา อย่างไรก็ตาม แนวทางนี้มีข้อเสียเนื่องจาก [A]

[วิธีจัดสรรพื้นที่แบบดัชนี (Indexed allocation method)]

ในการจัดสรรพื้นที่แบบดัชนี แต่ละไฟล์จะถูกเชื่อมโยงกับบล็อกดัชนี (index block) โดยในบล็อกดัชนีนี้จะจัดเก็บรายการของหมายเลขบล็อกที่สัมพันธ์กับไฟล์ที่ระบุ

รูปที่ 2 แสดงตัวอย่างของระบบไฟล์ที่ใช้วิธีจัดสรรพื้นที่แบบดัชนี



หมายเหตุ: ส่วนที่แรเงาถูกซ่อนไว้ และ n คือจำนวนของบล็อกที่มีในอุปกรณ์จัดเก็บข้อมูล

รูปที่ 2 ตัวอย่างของระบบไฟล์ที่ใช้วิธีจัดสรรพื้นที่แบบดัชนี

ในรูปที่ 2 ส่วนบนซ้ายแสดงถึงรายการหนึ่งในไดเรกทอรีที่แสดงถึงไฟล์ชื่อ readme โดยระบุว่าบล็อกดัชนีของไฟล์ readme จัดเก็บอยู่ที่บล็อก 101 โดยในบล็อก 101 นั้น ก็แสดงว่าข้อมูลของไฟล์ readme ถูกจัดเก็บอยู่ในบล็อก 149, 953 และ 708 ตามลำดับ ดังนั้น หลังจากอ่านบล็อกดัชนีแล้วก็จะสามารถค้นหาหมายเลขบล็อกที่จัดเก็บแต่ละส่วนของไฟล์ได้โดยง่าย อย่างไรก็ตาม แนวทางนี้ก็ยังมีข้อเสียเช่นกัน เนื่องจาก [B]

[การคำนวณหาจำนวนของบล็อกที่จำเป็นต้องใช้]

ในวิธีจัดสรรข้อมูลแบบเชื่อมโยง แต่ละบล็อกต้องจัดเก็บค่าดัชนี (index value) ของหมายเลขบล็อกต่อไป ในขณะที่วิธีจัดสรรข้อมูลแบบดัชนี จำเป็นต้องใช้บล็อกดัชนีโดยเฉพาะหนึ่งบล็อกแทนการเก็บค่าดัชนีไว้ในแต่ละบล็อก

พิจารณาถึงกรณีที่ไฟล์หนึ่งมีข้อมูล 40 kB เมื่อขนาดของบล็อกคือ 4 kB และขนาดของค่าดัชนี (index value) คือ 4 ไบต์แล้ว วิธีจัดสรรข้อมูลแบบเชื่อมโยงจำเป็นต้องใช้ [C1] บล็อกสำหรับจัดเก็บไฟล์นี้ และวิธีจัดสรรข้อมูลแบบดัชนีจำเป็นต้องใช้ [C2] บล็อก ในที่นี้ ไม่ต้องคำนึงถึงบล็อกสำหรับจัดเก็บรายการไดเรกทอรี

คำถามย่อย 1

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่เหมาะสมที่สุดเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในคำอธิบายข้างต้น ในที่นี้ คำตอบที่จะถูกเติมลงใน C1 และ C2 นั้นให้เลือกจากการจับคู่กันที่ถูกต้องในกลุ่มคำตอบสำหรับ C

กลุ่มคำตอบสำหรับ A และ B

a) ไม่สามารถจัดเก็บไฟล์ต่อเนื่อง (sequential file) ได้หากมีจำนวนบล็อกที่ว่างอยู่ต่อเนื่องกันไม่เพียงพอ
b) ขาดประสิทธิภาพในการเข้าถึงแบบโดยตรง (direct-access) เนื่องจากจำเป็นต้องเริ่มต้นจากจุดแรกของไฟล์แล้วจึงไล่ตามไปในแต่ละบล็อกเพื่อค้นหาส่วนที่ต้องการในไฟล์
c) มักสูญเสียพื้นที่โดยเปล่าประโยชน์มากกว่าวิธีอื่น โดยเฉพาะในระบบที่มีไฟล์ขนาดเล็ก ๆ เป็นจำนวนมาก
d) เนื้อหาของไฟล์ไม่สามารถถูกจัดเก็บลงในบล็อกที่ต่อเนื่องกันได้เนื่องจากข้อจำกัดในกลไกการอ้างอิง ส่งผลให้ประสิทธิภาพโดยรวมลดลง

กลุ่มคำตอบสำหรับ C

	C1	C2
a)	10	10
b)	10	11
c)	11	10
d)	11	11

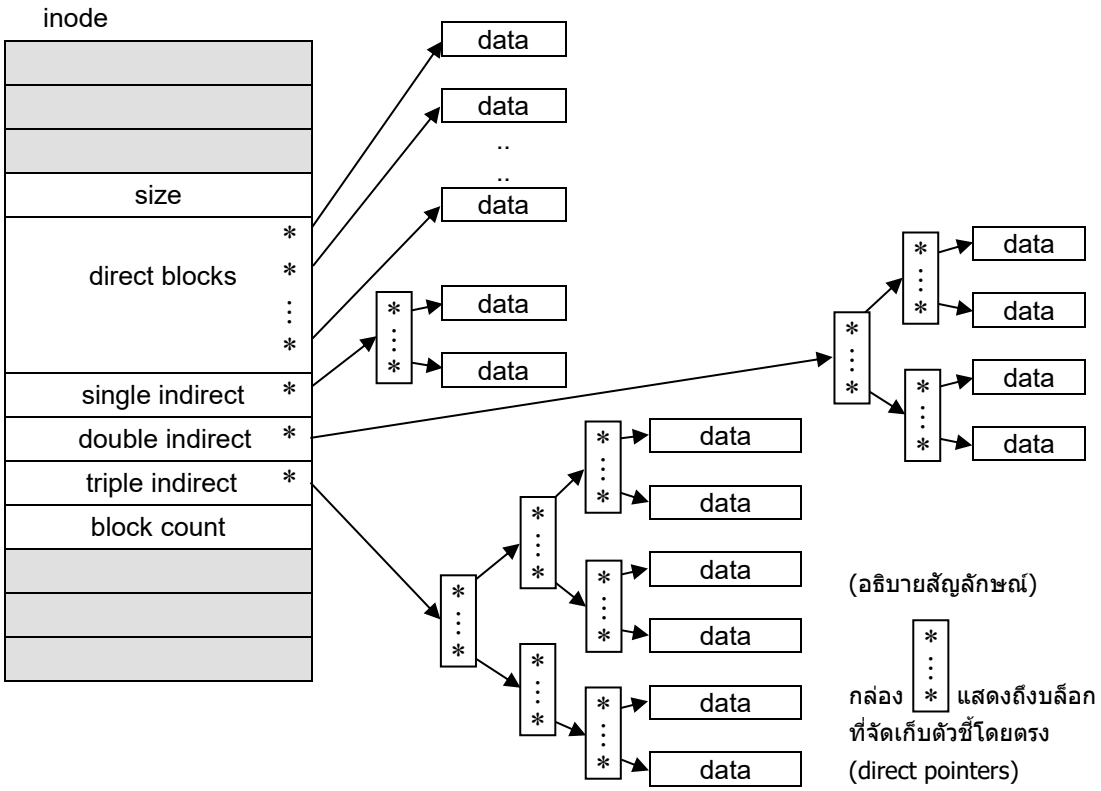
คำถามย่อย 2

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่เหมาะสมที่สุดเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในคำอธิบายต่อไปนี้

[วิธีการจัดสรรพื้นที่แบบไอโหนด (Inode allocation method)]
วิธีการจัดสรรพื้นที่แบบไอโหนดถูกใช้อยู่ในระบบที่มีพื้นฐานจากยูนิกส์ (UNIX-based system) วิธีจัดสรรพื้นที่แบบไอโหนดรวมเอาวิธีการแบบเชื่อมโยงและแบบดัชนีมาไว้ด้วยกัน แทนที่จะใช้ตารางเพื่อค้นหาไฟล์ เอนทิตี้ที่เรียกว่าไอโหนด (inode) จะถูกใช้เพื่อจัดเก็บข้อมูลต่าง ๆ เกี่ยวกับไฟล์ในลักษณะเดียวกันกับบล็อกดัชนีในรูปที่ 2 ซึ่งที่จริงวิธีนี้ก็คือวิธีจัดสรรพื้นที่แบบดัชนีที่เพิ่มจุดแข็งให้ยืดหยุ่นยิ่งขึ้น โดยสามารถเชื่อมโยงโหนดดัชนีได้สูงสุดถึงสี่โหนดด้วยกัน

แนวคิดของไอโหนดที่ถูกใช้ในระบบที่มีพื้นฐานจากยูนิกส์ถูกแสดงอยู่ในรูปที่ 3
ในไอโหนดมีตัวชี้ (pointer) อยู่ทั้งสิ้น 15 ตัว โดย 12 ตัวแรกชี้ไปที่บล็อกข้อมูลโดยตรง นั่นคือแต่ละส่วนของไฟล์นั้นสามารถเข้าถึงได้โดยตรงหากใช้พื้นที่เท่ากับหรือน้อยกว่า 12 บล็อก สำหรับไฟล์ที่ใหญ่ขึ้นนั้น ก็สามารถใช้ตัวชี้โดยอ้อม (single indirect), ตัวชี้โดยอ้อมสองชั้น (double indirect) และ

ตัวชี้โดยอ้อมสามชั้น (triple indirect) ขึ้นอยู่กับขนาดของไฟล์ ในที่นี้ ตัวชี้โดยอ้อมจะชี้ไปยังบล็อกที่จัดเก็บตัวชี้โดยตรง ขณะที่ในบล็อกที่จัดเก็บตัวชี้โดยอ้อมสองชั้นนั้น ก็จะชี้ไปยังบล็อกที่จัดเก็บตัวชี้โดยอ้อมอีกทอดหนึ่ง



หมายเหตุ: ส่วนที่แรเงาถูกซ่อนไว้ และ “*” แสดงถึงตัวชี้ที่ชี้ไปยังบล็อกใด ๆ

รูปที่ 3 แนวคิดของวิธีจัดสรรพื้นที่แบบไอโหนดในระบบที่มีพื้นฐานจากยูนิกส์

ตารางที่ 1 แสดงขนาดของข้อมูลที่ใหญ่ที่สุดที่สามารถเข้าถึงได้ของตัวชี้แต่ละระดับ ในที่นี้ ขนาดของบล็อกคือ 4096 ไบต์ (4 kB) และขนาดของตัวชี้คือ 4 ไบต์ นั่นคือในหนึ่งบล็อกจะเก็บตัวชี้ได้ 1024 ตัว

ตารางที่ 1 ขนาดของข้อมูลที่ใหญ่ที่สุดที่เข้าถึงได้ของตัวชี้แต่ละระดับ

ระดับของตัวชี้	จำนวนของตัวชี้	ขนาดข้อมูลที่ใหญ่ที่สุดที่เข้าถึงได้
ตัวชี้โดยตรง (direct blocks)	12	48 kB
ตัวชี้โดยอ้อม (single indirect)	1024	4 MB
ตัวชี้โดยอ้อมสองชั้น (double indirect)	1024 × 1024	4 GB
ตัวชี้โดยอ้อมสามชั้น (triple indirect)	1024 × 1024 × 1024	4 TB

พิจารณาถึงกรณีที่ไฟล์มีขนาด 1 MB เมื่อขนาดของบล็อกเป็น 4 kB และขนาดของตัวชี้เป็น 4 ไบต์ แล้ว จะต้องใช้บล็อกทั้งสิ้น [D] บล็อกในการจัดเก็บไฟล์ด้วยวิธีจัดสรรพื้นที่แบบไอโหนด ในที่นี้ ไม่ต้องคำนึงถึงบล็อกที่ใช้จัดเก็บไอโหนด และ 1MB ÷ 4 kB = 256

การกำหนดขนาดบล็อกอื่น ๆ นอกเหนือจาก 4 kB นั้น สามารถทำได้ขณะทำการฟอร์แมตระบบไฟล์ บล็อกที่มีขนาดเล็กจะช่วยลดขนาดการจับจองพื้นที่จัดเก็บต่อไฟล์สำหรับไฟล์ขนาดเล็ก ขณะที่บล็อกขนาดใหญ่ช่วยลดจำนวนบล็อกที่ต้องใช้ต่อไฟล์สำหรับไฟล์ขนาดใหญ่
การเปลี่ยนแปลงขนาดบล็อกส่งผลต่อขนาดสูงสุดของไฟล์ที่สามารถเข้าถึงได้อยู่สองปัจจัย: จำนวนของไบต์ต่อหนึ่งบล็อกข้อมูล และจำนวนของตัวชี้ต่อหนึ่งบล็อกดัชนี ตัวอย่างเช่น หากแต่ละบล็อกมีขนาด 1 kB และขนาดของตัวชี้เป็น 4 ไบต์แล้ว ขนาดสูงสุดของไฟล์ที่เข้าถึงได้จะเป็น [E] โดยประมาณ ด้วยวิธีการจัดสรรพื้นที่แบบไอโหนด

กลุ่มคำตอบสำหรับ D

a) 256 b) 257 c) 258 d) 259

กลุ่มคำตอบสำหรับ E

a) 16 GB b) 32 GB c) 64 GB d) 128 GB
e) 256 GB f) 512 GB g) 1 TB

สำหรับข้อสอบข้อที่ **Q2** ถึง **Q5 ให้เลือกทำเพียงสอง**ในสี่ข้อ

Q3. อ่านคำอธิบายของฐานข้อมูลเชิงสัมพันธ์สำหรับร้าน DVD ต่อไปนี้ แล้วตอบคำถามย่อย 1 ถึง 3

ร้าน DVD U ให้บริการจัดส่ง ให้เช่า และรับซื้อ DVD
แผนกบัญชีของร้าน DVD U ใช้ฐานข้อมูลเพื่อบจัดการด้านการชำระเงิน โดยฐานข้อมูลนี้ประกอบด้วยสี่ตาราง: ลูกค้า (Customer), ร้าน (Store), พนักงาน (Staff) และการชำระเงิน (Payment)
โครงสร้างของตารางและข้อมูลตัวอย่างเป็นดังต่อไปนี้:

ตาราง Customer

CustomerID	FirstName	LastName	Email	City
C001	Bill	Smith	billsmith@example.net	South lake
C002	Baelfire	Grehn	baelfiregrehn@example.net	East wood
C003	Bill	Gretan	billgretan@example.net	West hill
C004	Brendon	Green	brendongreen@example.net	East wood
C005	John	Doe	johndoe@example.net	North coast

ตาราง Store

StoreID	StoreName	ManagerStaffID
10	South-west shop	S001
20	North-east shop	S002

ตาราง Staff

StaffID	FirstName	LastName	Email	StoreID
S001	Mike	Hillyer	Mike.Hillyer@example.com	10
S002	Jon	Stephens	Jon.Stephens@example.com	20

ตาราง Payment

PaymentID	CustomerID	StaffID	Amount	PayDate
P001	C002	S002	2.99	2023-04-05
P002	C003	S001	0.99	2023-04-08
P003	C001	S001	5.99	2023-04-14
P004	C003	S001	4.99	2023-04-15
P005	C005	S002	9.99	2023-04-21

คำถามย่อย 1

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในข้อความสั่ง SQL ต่อไปนี้

ข้อความสั่ง SQL1 ต่อไปนี้ สร้างผลลัพธ์ว่าแต่ละร้านสามารถทำยอดขาย (sales amount) ได้เท่าใด โดยผลลัพธ์จะถูกเรียงลำดับด้วยยอดขายจากมากไปหาน้อย

```
-- SQL1 --
SELECT st.StoreID, st.StoreName, [   A   ] AS Sales
FROM [   B   ]
INNER JOIN Staff s  ON s.StaffID = p.StaffID
INNER JOIN Store st ON st.StoreID = s.StoreID
GROUP BY st.StoreID, st.StoreName
ORDER BY [   C   ]
```

คำสำคัญ `INNER JOIN` ในข้อความสั่งนี้จะทำการเลือกทุกแถวจากทั้งสองตารางในกรณีที่ค่าในคอลัมน์นั้นตรงกัน

จากข้อมูลตัวอย่างที่แสดงในตารางข้างต้น SQL1 จะแสดงผลลัพธ์ดังนี้:

StoreID	StoreName	Sales
20	North-east shop	12.98
10	South-west shop	11.97

กลุ่มคำตอบสำหรับ A ถึง C

a) `Amount`
b) `Payment`
c) `Payment p`
d) `Sales`
e) `Sales ASC`
f) `Sales DESC`
g) `SUM(Amount)`

คำถามย่อย 2

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] ในข้อความสั่ง SQL ต่อไปนี้

วันหนึ่ง พนักงานพบว่ามีลูกค้ารายหนึ่งลืมเครื่องพีซีไว้ในร้าน บนเครื่องพีซีนั้นมีสติ๊กเกอร์ระบุชื่อเจ้าของเครื่องติดอยู่ แต่ตัวอักษรลบเลือนไปจนพนักงานไม่สามารถอ่านชื่อได้ชัดเจน อย่างไรก็ตาม พนักงานทราบว่า:

o ตัวอักษรตัวแรกของชื่อ (First Name) คือ "B"

o นามสกุล (Last Name) ยาวห้าตัวอักษร โดยสามตัวแรกคือ "Gre" และตัวอักษรสุดท้ายคือ "n"

ข้อความสั่ง SQL2 ต่อไปนี้จะดึงข้อมูลลูกค้าที่มีชื่อใกล้เคียงกับที่ปรากฏอยู่บนเครื่องพีซี

```
-- SQL2 --
SELECT CustomerID, FirstName, LastName, Email
FROM Customer
WHERE [ D ]
```

อักขระตัวแทน (wildcard) สองตัวที่ที่มักถูกนำมาใช้ในแบบแผนการค้นหาได้แก่:

'%' (เครื่องหมายเปอร์เซ็นต์) ... ใช้แทนสตริงใด ๆ ที่มีตัวอักษรตั้งแต่ศูนย์ตัวขึ้นไป

'_' (เส้นใต้) ... ใช้แทนตัวอักษรใด ๆ หนึ่งตัว

จากข้อมูลตัวอย่างที่แสดงในตารางข้างต้น SQL2 จะแสดงผลลัพธ์ดังนี้:

CustomerID	FirstName	LastName	Email
C002	Baelfire	Grehn	baelfiregrehn@example.net
C004	Brendon	Green	brendongreen@example.net

กลุ่มคำตอบสำหรับ D

a) `(FirstName LIKE 'B%') AND (LastName LIKE 'Gre%n')`

b) `(FirstName LIKE 'B%') AND (LastName LIKE 'Gre_n')`

c) `(FirstName LIKE 'B_') AND (LastName LIKE 'Gre%n')`

d) `(FirstName LIKE 'B_') AND (LastName LIKE 'Gre_n')`

คำถามย่อย 3

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] ในข้อความสั่ง SQL ต่อไปนี้

ร้าน DVD U กำลังวางแผนจัดแคมเปญให้กับลูกค้าประจำ ข้อความสั่ง SQL3 ต่อไปนี้จะดึงข้อมูลลูกค้าที่ชำระเงินตั้งแต่สองครั้งขึ้นไประหว่างวันที่ 1 เมษายน ถึง 15 เมษายน 2023 ขึ้นมา

```
-- SQL3 --
SELECT c.CustomerID, c.LastName, COUNT(*) AS PayCount
   FROM Customer c, Payment p
   WHERE c.CustomerID = p.CustomerID
```

[E]

จากข้อมูลตัวอย่างที่แสดงในตารางข้างต้น SQL3 จะแสดงผลลัพธ์ดังนี้:

CustomerID	LastName	PayCount
C003	Gretan	2

กลุ่มคำตอบสำหรับ E

a) `AND p.PayDate >= '2023-04-01' AND p.PayDate <= '2023-04-15'`
`AND COUNT(*) >= 2`

b) `AND p.PayDate BETWEEN '2023-04-01' AND '2023-04-15'`
`GROUP BY c.CustomerID, c.LastName`
`HAVING COUNT(*) >= 2`

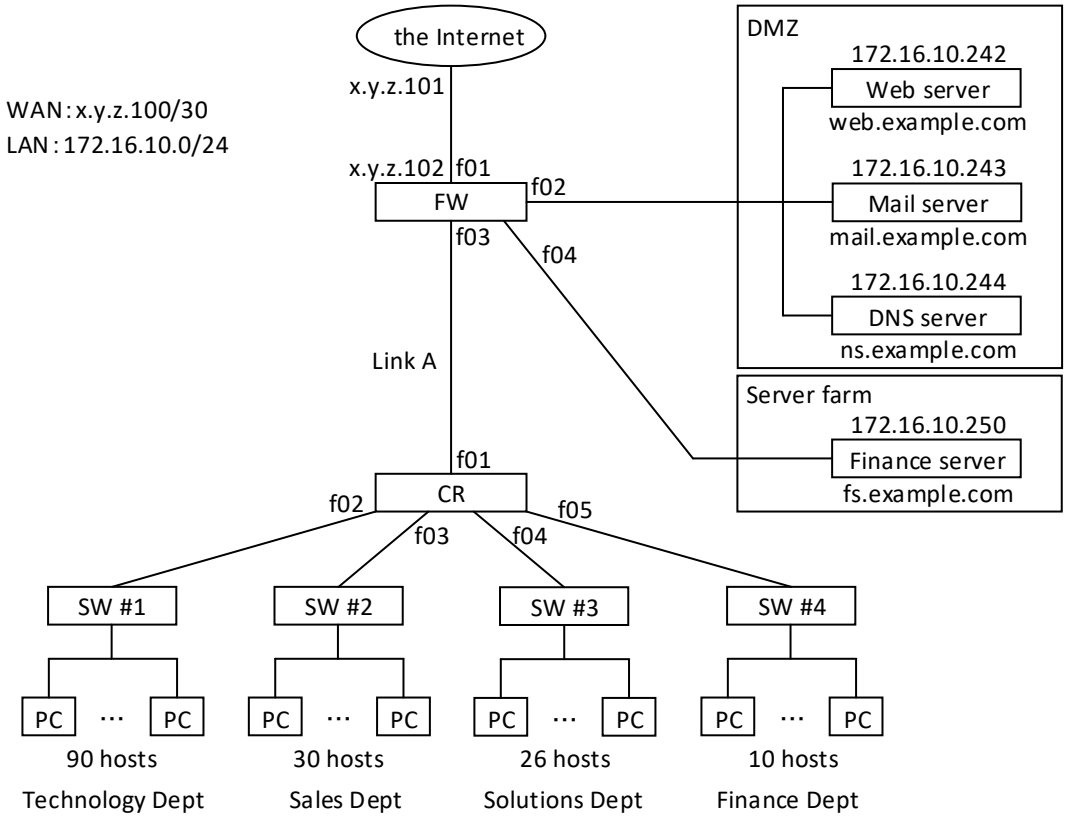
c) `GROUP BY c.CustomerID, c.LastName, p.PayDate`
`HAVING p.PayDate >= '2023-04-01' AND p.PayDate <= '2023-04-15'`
`AND COUNT(*) >= 2`

d) `HAVING p.PayDate BETWEEN '2023-04-01' AND '2023-04-15'`
`AND COUNT(*) >= 2`

สำหรับข้อสอบข้อที่ **Q2** ถึง **Q5 ให้เลือกทำเพียงสอง**ในสี่ข้อ

Q4. อ่านคำอธิบายเกี่ยวกับการออกแบบเครือข่ายของบริษัทขึ้นมาใหม่ต่อไปนี้ แล้วตอบคำถามย่อย 1 และ 2

บริษัท V เป็นบริษัทผู้ให้บริการพัฒนาและสร้างระบบที่มีพนักงาน 100 คน
ผู้จัดการของบริษัท V วางแผนเพื่อจะขยายบริการต่าง ๆ โดยการเพิ่มจำนวนพนักงานและปรับปรุงการควบคุมด้านความมั่นคงปลอดภัยในแต่ละแผนก ผู้ดูแลเครือข่ายได้วางแผนในการออกแบบโครงสร้างพื้นฐานของเครือข่ายภายในขึ้นมาใหม่โดยนำการกำหนดที่อยู่เครือข่ายด้วยซับเน็ตมาสก์ที่ปรับเปลี่ยนความยาวได้ (variable-length subnet mask) เข้ามาใช้ (ในส่วนที่แตกต่างกันของเครือข่าย) โดยใช้ที่อยู่ไอพีส่วนตัว (172.16.10.0/24) ส่วนต่าง ๆ ของเครือข่ายนี้ถูกเชื่อมต่อกันด้วยเราเตอร์หลัก (CR: Core Router) และไฟร์วอลล์ (FW)
รูปที่ 1 แสดงการกำหนดค่าเครือข่ายของบริษัท V



หมายเหตุ: ชื่อโดเมนของบริษัท V คือ example.com
รูปที่ 1: การกำหนดค่าเครือข่ายของบริษัท V

ฟังก์ชันแปลงที่อยู่เครือข่าย (Network Address Translation) ถูกใช้ที่ FW เพื่อให้เครือข่ายภายใน (internal network) เข้าใช้อินเทอร์เน็ตได้ เมลเซิร์ฟเวอร์ (Mail Server), เซิร์ฟเวอร์ DNS (DNS Server) และเว็บเซิร์ฟเวอร์ (Web Server) อยู่ใน DMZ และ FW จะปิดกั้นการสื่อสารขาเข้าไปยังเครือข่ายภายในทั้งจากอินเทอร์เน็ตและจาก DMZ เว้นแต่เฉพาะการสื่อสารที่อนุญาตไว้อย่างชัดเจนเท่านั้น และที่อยู่ไอพี (x.y.z.100/30) ถูกใช้สำหรับเครือข่ายแวน (WAN) ระหว่างอินเทอร์เน็ตกับ FW

คำถามย่อย 1

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในคำอธิบายต่อไปนี้และในตารางที่ 1

ในการออกแบบเครือข่ายแบบใช้ซับเน็ตมาสก์ปรับเปลี่ยนความยาวได้ (แบ่งเครือข่ายออกเป็นส่วน ๆ) ผู้ดูแลเครือข่ายได้ใช้แนวทางดังต่อไปนี้:

- ส่วนต่าง ๆ ของเครือข่ายถูกออกแบบตามความต้องการจำนวนโฮสต์ในแต่ละเครือข่าย โดยเริ่มจากแลนที่มีขนาดใหญ่ที่สุดไปยังแลนที่มีขนาดเล็กที่สุด โปรดทราบว่าจำนวนของโฮสต์ดังกล่าวนี้นับรวมทั้งพีซี เซิร์ฟเวอร์ และอินเทอร์เฟสเครือข่ายของอุปกรณ์ต่าง ๆ ที่เชื่อมต่ออยู่ในเครือข่ายนั้น
- หลังจากทราบที่อยู่ของทุก ๆ ซับเน็ตของแลนแล้ว ที่อยู่แรกที่สามารถใช้งานได้จะถูกกำหนดให้กับอินเทอร์เฟสของ FW และที่อยู่ถัดมาจะถูกใช้กับอินเทอร์เฟสของ CR สำหรับ Link A
- ในเครือข่ายแต่ละส่วน ที่อยู่ไอพีแรกที่สามารถใช้งานได้จะถูกใช้เป็นเกตเวย์ของเครือข่ายนั้น โดยจะถูกกำหนดให้กับอินเทอร์เฟสของเราเตอร์ แต่สำหรับ DMZ และเซิร์ฟเวอร์ฟาร์ม (Server Farm) เกตเวย์ของส่วนเครือข่ายนั้นจะถูกกำหนดไว้ที่อินเทอร์เฟสของ FW ดังนั้น เกตเวย์ของเซิร์ฟเวอร์ต่าง ๆ ใน DMZ ก็คือ [A]
- ตารางที่ 1 แสดงความต้องการจำนวนโฮสต์ของเครือข่ายที่ต้องใช้ในแต่ละส่วน

ตารางที่ 1 แนวทางกำหนดที่อยู่เครือข่ายแบบใช้ซับเน็ตมาสก์ปรับเปลี่ยนความยาวได้

ลำดับที่	ชื่อเครือข่าย	จำนวนโฮสต์ที่ต้องใช้	มาสก์เครือข่าย
1	Technology department	92	172.16.10.0/25
2	Sales department	32	[B]
3	Solutions department	28	[C]
4	Finance department	12	172.16.10.224/28
5	DMZ	4	172.16.10.240/29
6	Server farm	2	172.16.10.248/30
7	Link A	2	172.16.10.252/30

กลุ่มคำตอบสำหรับ A

a) 172.16.10.1 b) 172.16.10.240 c) 172.16.10.241
d) 172.16.10.248 e) 172.16.10.249

กลุ่มคำตอบสำหรับ B และ C

a) 172.16.10.0/26 b) 172.16.10.0/27 c) 172.16.10.127/26
d) 172.16.10.128/26 e) 172.16.10.128/27 f) 172.16.10.160/27
g) 172.16.10.191/26 h) 172.16.10.191/27 i) 172.16.10.192/26
j) 172.16.10.192/27

คำถามย่อย 2

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในตารางที่ 2

หลังจากที่ได้ออกแบบส่วนต่าง ๆ ของเครือข่ายแล้ว ผู้ดูแลเครือข่ายได้กำหนดค่าของโพรโทคอลหาเส้นทาง (routing protocol) บน CR และ FW ทำให้ฟังก์ชันหาเส้นทางทำงานได้อย่างถูกต้อง และโฮสต์ทั้งหมดในแลนสามารถสื่อสารระหว่างกันและเข้าใช้อินเทอร์เน็ตได้

เพื่อเพิ่มความมั่นคงปลอดภัยให้กับเซิร์ฟเวอร์ใน DMZ และเซิร์ฟเวอร์การเงิน (Finance server) ผู้ดูแลเครือข่ายได้เพิ่มกฎความมั่นคงแบบแบ่งโซน (zone-based security policy) หลายรายการลงใน FW ตามเงื่อนไขต่อไปนี้:

(1) บริการบนเว็บแบบเข้ารหัสลับ (HTTPS) บนที่อยู่ไอพีภายนอก (external IP address) ของ FW จะถูกจับคู่ (map) เข้ากับเว็บเซิร์ฟเวอร์ (Web server) ใน DMZ
(2) บริการส่งเมล (SMTP) และรับเมล (IMAPS) บนที่อยู่ไอพีภายนอกของ FW จะถูกจับคู่เข้ากับเมลเซิร์ฟเวอร์ (Mail server) ใน DMZ
(3) อนุญาตให้ DMZ เข้าใช้อินเทอร์เน็ตได้ผ่าน NAT (การแปลงที่อยู่ IP) ซึ่งจำเป็นต่อการทำงานของบริการที่ถูกจับคู่ไว้ต่าง ๆ
(4) อนุญาตให้เซิร์ฟเวอร์การเงิน (Finance server) สามารถเข้าถึงได้จากแผนกการเงิน (Finance department) ด้วยบริการเว็บแบบเข้ารหัสลับ (secure web service) เท่านั้น
(5) โฮสต์อื่น ๆ ทั้งจากแลนและผู้ใช้จากภายนอกต้องไม่สามารถเข้าถึงเซิร์ฟเวอร์การเงินได้
(6) อนุญาตให้การสื่อสารอื่น ๆ จากแลนเข้าไปยัง DMZ ได้ เพื่อให้ผู้ใช้ภายในเข้าถึงทรัพยากรต่าง ๆ บนเซิร์ฟเวอร์

ตารางที่ 2 แสดงการกำหนดค่าของตารางกฎการสื่อสารบน FW โดยแต่ละแพคเก็ตที่เข้ามาจะถูกตรวจสอบตามกฎต่าง ๆ ดังต่อไปนี้:

(1) อันดับแรก ต้นทาง ปลายทาง และบริการปลายทางจะถูกนำมาเปรียบเทียบกับกฎหมายเลข 1

(i) หากตรงกัน ให้ดำเนินการตามการกระทำที่กำหนดไว้ และถ้าหากได้กำหนดการแปลงเอาไว้ ให้ดำเนินการตามที่ระบุไว้เช่นกัน หลังจากดำเนินการแล้วก็ไม่ต้องตรวจสอบกับกฎข้อต่อไป

(ii) หากไม่ตรงกัน ให้เลื่อนไปยังกฎข้อต่อไป

(2) ทำซ้ำขั้นตอนที่ (1) จนถึงกฎข้อสุดท้ายในตาราง

(3) ปฏิเสธ (Deny) แพคเก็ตนั้น หากไม่มีการดำเนินการใด ๆ ตามกฎในตาราง

ตารางที่ 2 ตารางกฎการสื่อสารบน FW (ไม่แสดงการตั้งค่าอื่น ๆ ที่ไม่เกี่ยวข้อง)

หมายเลข	ต้นทาง	ปลายทาง	บริการปลายทาง	การกระทำ	การแปลง
1	the Internet	x.y.z.102	HTTPS	Allow	MAP: 172.1.10.242
2	the Internet	x.y.z.102	SMTP/IMAPS	Allow	MAP: 172.1.10.243
3	DMZ	D	ANY	Allow	NAT
4	E	172.16.10.250	HTTPS	Allow	
5	F	172.16.10.250	ANY	Deny	
6	LANs	DMZ	ANY	Allow	
⋮	⋮	⋮	⋮	⋮	

หมายเหตุ DMZ: บริเวณของเครือข่ายที่ป้องกันเครือข่ายภายในจากการสื่อสารที่ไม่ไว้ใจ ซึ่งเป็นส่วนของเครือข่ายย่อยที่ตั้งอยู่ระหว่างอินเทอร์เน็ตกับเครือข่ายส่วนตัว

the Internet: เครือข่ายสาธารณะหรือเครือข่ายที่ไม่ไว้ใจ จำเป็นต้องใช้ FW เพื่อปกป้องเครือข่ายส่วนตัวจากการสื่อสารที่ไม่ไว้ใจ

LANs: เครือข่ายส่วนตัวหรือเครือข่ายที่ไว้ใจซึ่งสามารถสื่อสารไปยัง DMZ และอินเทอร์เน็ตได้

ANY: ทุกเครือข่าย

กลุ่มคำตอบสำหรับ D ถึง F

a) 172.16.10.0/24 b) 172.16.10.64/26 c) 172.16.10.128/26

d) 172.16.10.224/28 e) 172.16.10.240/29 f) ANY

g) DMZ h) the Internet

สำหรับข้อสอบข้อที่ **Q2** ถึง **Q5 ให้เลือกทำเพียงสอง**ในสี่ข้อ

Q5. อ่านคำอธิบายของการพัฒนาและการทดสอบโปรแกรมต่อไปนี้ แล้วตอบคำถามย่อย

รัฐ S ประกอบด้วยสี่เมือง: East, North, South และ West โดยสำนักงานสถิติแห่งรัฐ (SSB: State Statistics Bureau) ทำหน้าที่จัดการไฟล์สถิติที่มีข้อมูลดังแสดงในตารางที่ 1

ตารางที่ 1 เนื้อหาในไฟล์สถิติ

ชื่อเมือง (City name)	ประชากรทั้งหมด (Total population)	ประชากรผู้ใหญ่ (Adult population)	มีงานทำ (Employed)	ไม่มีงานทำ (Unemployed)
East	124,000	104,000	90,000	10,000
North	252,000	208,000	176,000	24,000
South	132,000	106,000	86,000	14,000
West	260,000	212,000	168,000	32,000

SSB ได้พัฒนาโปรแกรมที่สร้างผลลัพธ์เป็นรายงานสถิติจากไฟล์สถิตินี้ โดยโปรแกรมได้พัฒนาจนใกล้แล้วเสร็จ ทาง SSB จึงดำเนินการจัดเตรียมกรณีทดสอบต่าง ๆ

[คำอธิบายโปรแกรม]

(1) โปรแกรมอ่านไฟล์สถิติแล้วสร้างผลลัพธ์เป็นรายงานสถิติ

(2) สำหรับแต่ละเรคคอร์ดในไฟล์สถิติ โปรแกรมจะคำนวณและแสดงอัตราแรงงาน (Labor force rate) และอัตราว่างงาน (Unemployment rate) ด้วยสูตรดังนี้

Labor force rate (%) = 100 × (Employed + Unemployed) ÷ Adult population

Unemployment rate (%) = 100 × Unemployed ÷ (Employed + Unemployed)

(3) หลังจากประมวลผลทุกเรคคอร์ดแล้ว โปรแกรมจะแสดงผลรวมของประชากรทั้งหมด, ประชากรผู้ใหญ่, มีงานทำ, ไม่มีงานทำ และอัตราแรงงานกับอัตราว่างงานของทั้งรัฐ จากนั้นโปรแกรมจะแสดงผลอัตราว่างงานสูงสุดและต่ำสุดพร้อมชื่อเมือง

(4) ตัวอย่างของรายงานสถิติเป็นดังต่อไปนี้:

```
City   TotalPop  AdultPop    Employ  Unemploy  Labor%  Unemp%
-----  --------  --------  --------  --------  ------  ------
East    124,000   104,000    90,000    10,000    96.2    10.0
North   252,000   208,000   176,000    24,000    96.2    12.0
South   132,000   106,000    86,000    14,000    94.3    14.0
West    260,000   212,000   168,000    32,000    94.3    16.0
-----  --------  --------  --------  --------  ------  ------
TOTAL   768,000   630,000   520,000    80,000    95.2    13.3

Highest Unemp%: 16.0, City: West
Lowest  Unemp%: 10.0, City: East
```

[โปรแกรม]

```
   STRING: City, CityHigh ← "????", CityLow ← "????"
   INT:  TotalPop, AdultPop, Employ, Unemploy
   INT:  TotalTotalPop ← 0, TotalAdultPop ← 0,
         TotalEmploy ← 0,   TotalUnemploy ← 0
   REAL: LaborRate, UnempRate
X →REAL: UnempRateHigh ← 0.0, UnempRateLow ← 100.0

   OpenFile("statistic");     /* Open the statistic file */
   Print("City   TotalPop  AdultPop    Employ  Unemploy  Labor%  Unemp%");
   Print("-----  --------  --------  --------  --------  ------  ------");
   ReadRecord(City, TotalPop, AdultPop, Employ, Unemploy);
   WHILE (not end-of-file) {     /* Loop until the statistic file reaches the end-of-file */
      LaborRate ← 100.0 × (Employ + Unemploy) ÷ AdultPop;
      UnempRate ← 100.0 × Unemploy ÷ (Employ + Unemploy);
Y →   IF (UnempRate > UnempRateHigh) {
         CityHigh ← City;
         UnempRateHigh ← UnempRate;
      }
Z →   IF (UnempRate < UnempRateLow) {
         CityLow ← City;
         UnempRateLow ← UnempRate;
      }
      TotalTotalPop ← TotalTotalPop + TotalPop;
      TotalAdultPop ← TotalAdultPop + AdultPop;
      TotalEmploy   ← TotalEmploy   + Employ;
      TotalUnemploy ← TotalUnemploy + Unemploy;
      Print(City, TotalPop, AdultPop,
                  Employ, Unemploy, LaborRate, UnempRate);
      ReadRecord(City, TotalPop, AdultPop, Employ, Unemploy);
   }
   Print("-----  --------  --------  --------  --------  ------  ------");
   LaborRate ← 100.0 × (TotalEmploy + TotalUnemploy) ÷ TotalAdultPop;
   UnempRate ← 100.0 × TotalUnemploy ÷ (TotalEmploy + TotalUnemploy);
   Print("TOTAL", TotalTotalPop, TotalAdultPop,
                  TotalEmploy, TotalUnemploy, LaborRate, UnempRate);
   Print();     /* Print a blank line */
   Print("Highest Unemp%:", UnempRateHigh, ", City:", CityHigh);
   Print("Lowest  Unemp%:", UnempRateLow,  ", City:", CityLow );
   CloseFile("statistic");     /* Close the statistic file */
```

หมายเหตุ: ฟังก์ชัน ReadRecord(v_1, v_2, …) อ่านเรคคอร์ดถัดไปขึ้นมาจากไฟล์สถิติแล้วเก็บค่าต่าง ๆ ไว้ในตัวแปร v_1, v_2, … .

ฟังก์ชัน Print(p_1, p_2, …) พิมพ์อาร์กิวเมนต์ p_1, p_2, … ออกมาในหนึ่งบรรทัด กำหนดให้ระยะห่างระหว่างแต่ละอาร์กิวเมนต์ถูกปรับให้เหมาะสมโดยอัตโนมัติ

คำถามย่อย

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในคำอธิบายต่อไปนี้

โปรดทราบว่าส่วนที่ถูกแรเงา [ ] นั้นถูกซ่อนไว้

การพัฒนาโปรแกรมขณะนี้อยู่ในขั้นตอนการทดสอบ และการทดสอบถูกดำเนินการด้วยกรณีทดสอบที่หลากหลาย กรณีทดสอบสี่กรณีและผลการทดสอบถูกแสดงไว้ด้านล่างนี้:

[กรณีทดสอบ 1]

กรณีนี้มีสองเมืองที่มีอัตราการว่างงานสูงที่สุดและอีกสองเมืองมีอัตราว่างงานต่ำที่สุดเท่ากัน ในกรณีนี้โปรแกรมจะแสดงผลรายงานดังต่อไปนี้:

City	TotalPop	AdultPop	Employ	Unemploy	Labor%	Unemp%
East	120,000	104,000	90,000	10,000	96.2	10.0
North	120,000	105,000	90,000	10,000	95.2	10.0
South	120,000	106,000	85,000	15,000	94.3	15.0
West	120,000	107,000	85,000	15,000	93.5	15.0
TOTAL	480,000	422,000	350,000	50,000	94.8	12.5

Highest Unemp%: 15.0, City: [A]

Lowest Unemp%: 10.0, City: [ ]

[กรณีทดสอบ 2]

ในกรณีนี้ ทุกเมืองมีอัตราว่างงานเป็น 0% ซึ่งตามลอจิกของโปรแกรมแล้ว ชื่อของเมืองที่จะถูกแสดงในสองบรรทัดสุดท้ายจะเป็นดังนี้:

City	TotalPop	AdultPop	Employ	Unemploy	Labor%	Unemp%
East	120,000	104,000	90,000	0	86.5	0.0
North	120,000	105,000	90,000	0	85.7	0.0
South	120,000	106,000	85,000	0	80.2	0.0
West	120,000	107,000	85,000	0	79.4	0.0
TOTAL	480,000	422,000	350,000	0	82.9	0.0

Highest Unemp%: 0.0, City: [B]

Lowest Unemp%: 0.0, City: [ ]

[กรณีทดสอบ 3]

ปรากฏการณ์ที่กล่าวถึงในกรณีทดสอบ 2 นั้น สามารถแก้ไขได้ด้วยการเปลี่ยนตัวดำเนินการ “>” ไปเป็น “≥” ที่บรรทัด Y และเปลี่ยนตัวดำเนินการ “<” เป็น “≤” ที่บรรทัด Z

หลังจากดำเนินการแก้ไขแล้ว จึงทำการทดสอบในอีกกรณีหนึ่งซึ่งทุก ๆ เมืองมีอัตราการว่างงานเป็น 100% โดยโปรแกรมจะแสดงผลรายงานดังต่อไปนี้:

City	TotalPop	AdultPop	Employ	Unemploy	Labor%	Unemp%
East	120,000	104,000	0	90,000	86.5	100.0
North	120,000	105,000	0	90,000	85.7	100.0
South	120,000	106,000	0	85,000	80.2	100.0
West	120,000	107,000	0	85,000	79.4	100.0
TOTAL	480,000	422,000	0	350,000	82.9	100.0

Highest Unemp%:100.0, City: [ ]

Lowest Unemp%:100.0, City: [C]

[กรณีทดสอบ 4]

หลังดำเนินการแก้ไขดังที่กล่าวถึงในกรณีทดสอบ 3 แล้ว จึงได้ทดสอบตามกรณีทดสอบ 1 อีกครั้ง เนื่องจากลอจิกของโปรแกรมได้เปลี่ยนแปลงไป ซึ่งโปรแกรมจะแสดงผลรายงานดังต่อไปนี้:

City	TotalPop	AdultPop	Employ	Unemploy	Labor%	Unemp%
East	120,000	104,000	90,000	10,000	96.2	10.0
North	120,000	105,000	90,000	10,000	95.2	10.0
South	120,000	106,000	85,000	15,000	94.3	15.0
West	120,000	107,000	85,000	15,000	93.5	15.0
TOTAL	480,000	422,000	350,000	50,000	94.8	12.5

Highest Unemp%: 15.0, City: [ ]

Lowest Unemp%: 10.0, City: [D]

ในเหตุการณ์นี้ ปรากฏการณ์ดังกล่าวถึงข้างต้นสามารถแก้ไขได้ด้วยวิธีที่แตกต่างกันด้วยการแทนที่ค่าตั้งต้นในบรรทัด X ด้วยค่าต่าง ๆ ดังตัวอย่างนี้:

REAL: UnempRateHigh ← [E], UnempRateLow ← [F]

กลุ่มคำตอบสำหรับ A ถึง D

a) ???? b) East c) North

d) South e) West

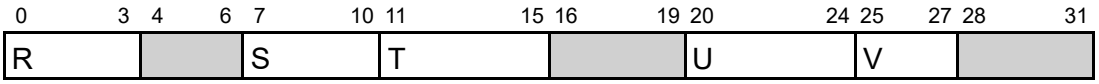
กลุ่มคำตอบสำหรับ E และ F

a) -0.1 b) 0.1 c) 99.9 d) 100.1

คำถาม **Q6** เป็น **คำถามบังคับ**

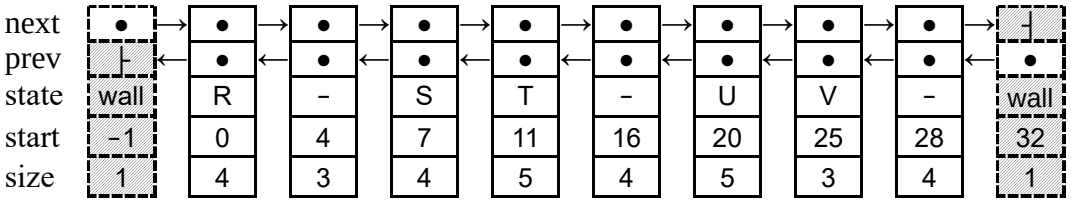
Q6. อ่านคำอธิบายเกี่ยวกับโปรแกรมต่อไปนี้ แล้วตอบคำถามย่อย 1 และ 2

ระบบปฏิบัติการจำเป็นต้องจัดสรรหน่วยความจำให้กับโพรเซสต่าง ๆ ได้อย่างไดนามิก
รูปที่ 1 แสดงตัวอย่างของสถานะในการจัดสรรหน่วยความจำที่มีพื้นที่ของหน่วยความจำอยู่ต่อเนื่องกัน 32 บล็อก ตั้งแต่บล็อกที่ 0 ถึง 31 ในตัวอย่างนี้บล็อก 0 ถึง 3 ได้รับการจัดสรรให้โพรเซส R และบล็อก 4 ถึง 6 ยังว่างอยู่ ในรูปนี้ ▢ แสดงถึงบล็อกของหน่วยความจำที่ยังว่างอยู่



รูปที่ 1 สถานะของการจัดสรรหน่วยความจำ

หนึ่งในวิธีที่สามารถติดตามสถานการณ์จัดสรรหน่วยความจำได้คือการใช้ลิงก์ลิสต์ (linked list) โดยลิงก์ลิสต์ในรูปที่ 2 แสดงสถานะของการจัดสรรหน่วยความจำดังที่นำเสนอในรูปที่ 1 โหนดต่าง ๆ ถูกจัดเก็บโดยเรียงลำดับจากหมายเลขบล็อกเริ่มต้น
ในรูปที่สอง ตัวตรวจสอบ (จุดสีดำ) ถูกใส่ไว้ทั้งสองด้านเพื่อให้แต่ละโหนดถูกใช้เป็นโหนดตรงกลางได้แม้จะเป็นโหนดแรกหรือโหนดสุดท้ายก็ตาม โดยส่วนที่แรเงานั้นคือส่วนที่ไม่สามารถเปลี่ยนแปลงได้เนื่องจากถูกใช้เป็นขอบเขต และ “wall” แสดงให้เห็นว่าสถานะของตัวตรวจสอบนั้นคือ “ไม่ว่าง”



รูปที่ 2 ลิงก์ลิสต์ที่แสดงสถานะของการจัดสรรหน่วยความจำในรูปที่ 1

[คำอธิบายโปรแกรม]
โปรแกรมจัดการหน่วยความจำที่แสดงในรูปที่ 1 ด้วยลิงก์ลิสต์ที่แสดงในรูปที่ 2
โปรแกรมทำงานโดยมีสองกระบวนการดังต่อไปนี้:
(i) เมื่อโพรเซสใหม่เริ่มการทำงาน ให้จัดสรรหน่วยความจำในบล็อกที่ต่อเนื่องกันตามขนาดที่ร้องขอ
(ii) หลังจากโพรเซสสิ้นสุดการทำงาน ให้คืนบล็อกหน่วยความจำที่ได้รับการจัดสรรไป

เพื่อดำเนินการกับโหนดต่าง ๆ ชนิดของข้อมูล Node ได้ถูกกำหนดไว้ดังนี้:

```
Node: node {Node:   next,   // โหนดถัดไป
            Node:   prev,   // โหนดก่อนหน้า
            STRING: state,  // ชื่อโพรเซส (ไม่ว่าง) หรือ "-" (ว่าง)
            INT:    start,  // หมายเลขบล็อกหน่วยความจำเริ่มต้น
            INT:    size    // ขนาดของบล็อกหน่วยความจำ
           }
```

โปรแกรมนี้ใช้สี่ฟังก์ชัน: `createNode`, `removeNode`, `allocate` และ `release`
ลิงก์ลิสต์จะถูกจัดการด้วยสองฟังก์ชันหลัก `allocate` และ `release` โดยฟังก์ชัน `createNode` และ `removeNode` จะถูกเรียกใช้จาก `allocate` และ `release` ตามลำดับ

(1) `FUNCTION: createNode()`
ฟังก์ชัน `createNode` สร้างโหนดใหม่แล้วคืนค่าเป็นโหนดนั้น
ข้อความสั่งด้านล่างนี้จะสร้างโหนดใหม่และให้ชื่อว่า `new`

```
Node: new
new ← createNode();
```

หลังจากสร้างโหนดใหม่แล้ว โปรแกรมจะต้องเพิ่มโหนดดังกล่าวเข้าไปในลิงก์ลิสต์โดยไปกำหนดพอยน์เตอร์ของโหนดที่เกี่ยวข้อง

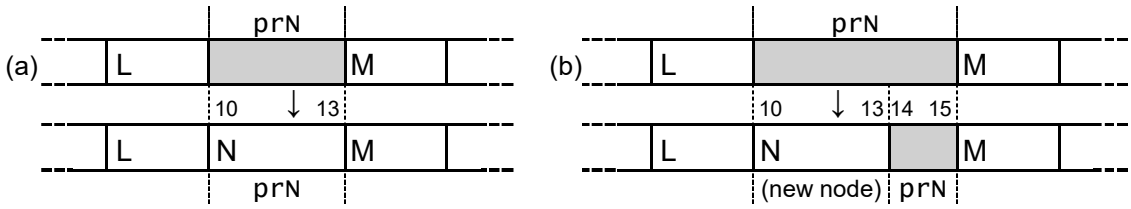
(2) `FUNCTION: removeNode(Node: remove)`
ฟังก์ชัน `removeNode` จะนำโหนดที่ระบุในอาร์กิวเมนต์ `remove` ออกไป
ข้อความสั่งด้านล่างนี้จะนำโหนดที่ชื่อ `garbage` ออกไป

```
Node: garbage
removeNode(garbage);
```

ก่อนที่จะนำโหนดออกไปนั้น โปรแกรมต้องตัดการเชื่อมต่อ จาก/มายัง โหนดนั้น ๆ โดยไปกำหนดพอยน์เตอร์ของโหนดที่เกี่ยวข้อง

(3) `FUNCTION: allocate(Node: free, STRING: procName, INT: reqSize)`
ฟังก์ชัน `allocate` จัดสรรบล็อกของหน่วยความจำขนาด `reqSize` ให้กับโพรเซส `procName` จากบล็อกของหน่วยความจำที่ว่างอยู่ ซึ่งครอบครองอยู่โดยโหนด `free`
เมื่อฟังก์ชันนี้ถูกเรียกใช้ ต้องทำให้มั่นใจว่าเงื่อนไขต่อไปนี้เป็นจริง:
(i) สถานะของโหนด `free` คือ "ว่าง" นั่นคือ `free.state = "-"`
(ii) โหนด `free` มีบล็อกของหน่วยความจำว่างที่เพียงพอ นั่นคือ `free.size ≥ reqSize`

ในฟังก์ชันนี้ มีสองกรณี (กรณี (a) และ (b) ในรูปที่ 3) ที่ต้องคำนึงถึง
กรณี (a): `free.size = reqSize`
บล็อกของหน่วยความจำที่ครอบครองอยู่โดยโหนด `free` จะถูกจัดสรรให้กับโพรเซสที่ร้องขอทั้งบล็อก จำนวนของโหนดที่อยู่ในลิงก์ลิสต์จะไม่มีการเปลี่ยนแปลง
กรณี (b): `free.size > reqSize`
โหนด `free` ต้องถูกแบ่งออกเป็นสองโหนด ทำให้จำนวนของโหนดในลิงก์ลิสต์เพิ่มขึ้น 1 โหนด



รูปที่ 3 แสดงการเรียกใช้ฟังก์ชัน `allocate(prN, "N", 4);` ที่อาจเกิดขึ้นได้สองกรณี

[โปรแกรม allocate]

```
FUNCTION: allocate(Node: free, STRING: procName, INT: reqSize) {
   Node: last, new

   IF (free.size = reqSize) {
      [    A    ];
   }
   ELSE {  /* free.size > reqSize */
      new ← createNode();
      last ← free.prev;

      last.next ← new;
      new.next ← free;
      [    B    ];
      [    C    ];

      new.state ← procName;
      new.start ← free.start;
      new.size ← reqSize;
      free.start ← free.start + reqSize;
      free.size ← [    D    ];
   }
}
```

(4) `FUNCTION: release(Node: proc)`

ฟังก์ชัน `release` คืนบล็อกหน่วยความจำที่ครอบครองอยู่โดยโหนด `proc` บล็อกหน่วยความจำที่เคยถูกครอบครองโดยโหนด `proc` นี้ จะกลายเป็นบล็อกหน่วยความจำว่าง

เมื่อฟังก์ชันนี้ถูกเรียกใช้ ต้องทำให้มั่นใจว่าเงื่อนไขต่อไปนี้เป็นจริง:

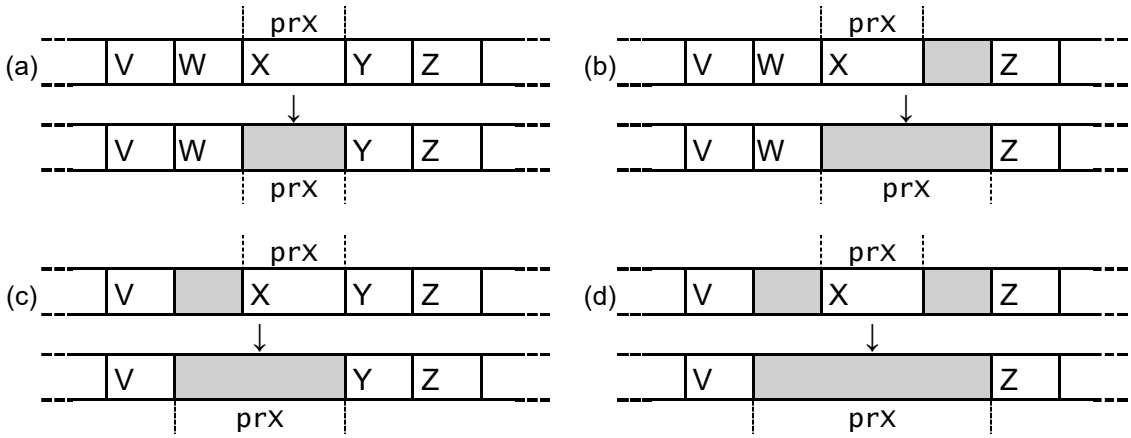
สถานะของโหนด `proc` คือ "ไม่ว่าง" นั่นคือ `proc.state ≠ "-"`

ในฟังก์ชันนี้ มีสี่กรณี (กรณี (a) ถึง (d) ในรูปที่ 4) ที่ต้องคำนึงถึง

กรณี (a): สถานะของโหนดก่อนหน้าและถัดไปคือ "ไม่ว่าง" หลังจากเปลี่ยนโหนด `proc` ให้เป็น "ว่าง" แล้ว จำนวนของโหนดในลิงก์ลิสต์จะไม่มีการเปลี่ยนแปลง

กรณี (b) และ (c): สถานะของโหนดก่อนหน้าเป็น "ไม่ว่าง" และสถานะของโหนดถัดไปคือ "ว่าง" หรือในทางกลับกัน หลังจากเปลี่ยนโหนด `proc` ให้เป็น "ว่าง" แล้ว โหนด "ว่าง" สองโหนดที่อยู่ต่อเนื่องกันจะต้องถูกนำมารวมกันให้กลายเป็นโหนด "ว่าง" เพียงโหนดเดียว ดังนั้น จำนวนของโหนดในลิงก์ลิสต์จึงลดลง 1 โหนด

กรณี (d): สถานะของโหนดก่อนหน้าและโหนดถัดไปคือ “ว่าง” หลังจากเปลี่ยนแปลงโหนด proc ให้เป็น “ว่าง” แล้ว โหนด “ว่าง” ที่อยู่ต่อเนื่องกันสามโหนดจะต้องถูกนำมารวมกันให้เป็นโหนดเดียว ดังนั้น จำนวนของโหนดในลิงก์ลิสต์จึงลดลง 2 โหนด



รูปที่ 4 แสดงการเรียกใช้ฟังก์ชัน release(prX); ที่อาจเกิดขึ้นได้สี่กรณี

[โปรแกรม release]

```
FUNCTION: release(Node: proc) {
   Node: next1, next2, prev1, prev2

   proc.state ← "-";
   next1 ← proc.next;
   IF (next1.state = "-") {  /* รวมโหนด “ว่าง” ที่อยู่ต่อเนื่องกันสองโหนด */
      next2 ← next1.next;
      proc.next ← next2;
      next2.prev ← proc;
      proc.size ← proc.size + next1.size;
      removeNode(next1);
   }
   prev1 ← proc.prev;
   IF (prev1.state = "-") {  /* รวมโหนด “ว่าง” ที่อยู่ต่อเนื่องกันสองโหนด */
      prev2 ← prev1.prev;
      [   E   ];
      [   F   ];
      proc.start ← prev1.start;
      proc.size ← [   G   ];
      removeNode(prev1);
   }
}
```

คำถามย่อย 1

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงใน [ ] แต่ละช่องในโปรแกรมข้างต้น

กลุ่มคำตอบสำหรับ A

a) `/* ไม่ต้องทำอะไร */`
b) `free.start ← free.start + reqSize`
c) `free.state ← "-"`
d) `free.state ← procName`

กลุ่มคำตอบสำหรับ B และ C

a) `free.prev ← last`
b) `free.prev ← new`
c) `last.prev ← free`
d) `last.prev ← new`
e) `new.prev ← free`
f) `new.prev ← last`

กลุ่มคำตอบสำหรับ D

a) `free.size - reqSize`
b) `free.size - reqSize - 1`
c) `reqSize`
d) `reqSize - 1`

กลุ่มคำตอบสำหรับ E และ F

a) `prev1.next ← proc`
b) `prev1.prev ← prev2`
c) `prev2.next ← prev1`
d) `prev2.next ← proc`
e) `proc.prev ← prev1`
f) `proc.prev ← prev2`

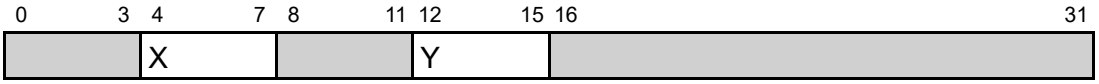
กลุ่มคำตอบสำหรับ G

a) `prev1.size`
b) `prev1.size + next1.size`
c) `prev2.size`
d) `prev2.size + next1.size`
e) `proc.size + prev1.size`
f) `proc.size + prev2.size`

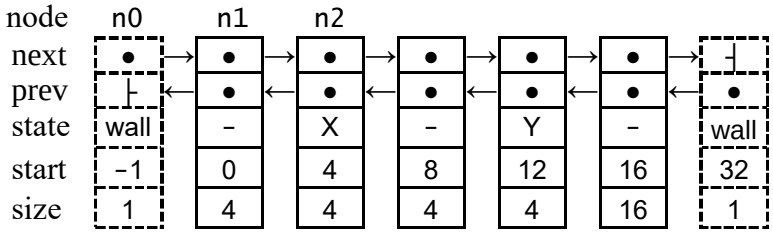
คำถามย่อย 2

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงใน [ ] แต่ละช่องในคำอธิบายต่อไปนี้

รูปที่ 5 แสดงสถานะของการจัดสรรหน่วยความจำในปัจจุบัน และรูปที่ 6 แสดงลิงก์ลิสต์ที่ใช้แทนสถานะของการจัดสรรหน่วยความจำดังที่แสดงในรูปที่ 5 ในที่นี้ ให้สามโหนดแรกในรูปที่ 6 สามารถอ้างถึงได้ด้วยชื่อโหนด n0, n1 และ n2



รูปที่ 5 สถานะของการจัดสรรหน่วยความจำในปัจจุบัน



รูปที่ 6 ลิงก์ลิสต์ที่ใช้แทนสถานะของการจัดสรรหน่วยความจำดังแสดงในรูปที่ 5

จากทั้งสามกรณีต่อไปนี้:

กรณีที่ 1: ฟังก์ชัน `allocate` ถูกเรียกใช้ดังนี้ `allocate(n1, "Z", 2);`

กรณีที่ 2: ฟังก์ชัน `allocate` ถูกเรียกใช้ดังนี้ `allocate(n1, "Z", 4);`

กรณีที่ 3: ฟังก์ชัน `release` ถูกเรียกใช้ดังนี้ `release(n2);`

กรณีที่เกิดการเปลี่ยนแปลงกับค่าของ `n0.next` คือ [H]

กลุ่มคำตอบสำหรับ H

a) กรณีที่ 1 เท่านั้น
b) กรณีที่ 2 เท่านั้น
c) กรณีที่ 3 เท่านั้น
d) กรณีที่ 1 และกรณีที่ 2
e) กรณีที่ 1 และกรณีที่ 3
f) กรณีที่ 2 และกรณีที่ 3

สำหรับข้อสอบข้อที่ **Q7** และ **Q8 ให้เลือกทำหนึ่ง**ในสองข้อ
จากนั้น ให้ระบายทับในวงกลม ⓢ ในกระดาษคำตอบสำหรับข้อที่เลือกทำ
หากเลือกทั้งสองข้อ จะตรวจให้คะแนนเฉพาะข้อแรกเท่านั้น

Q7. อ่านคำอธิบายของโปรแกรม และโค้ดโปรแกรมภาษา C ต่อไปนี้ แล้วตอบคำถามย่อย 1 และ 2

พาลินโดรม (palindrome) เป็นคำหรือลำดับอักขระอื่น ๆ ที่สามารถอ่านได้เหมือนกัน ทั้งแบบย้อนกลับและอ่านไปข้างหน้า เช่น anna, madam, หรือ stats
นอกจากนี้ยังมีพาลินโดรมที่เป็นตัวเลขที่สามารถอ่านได้ค่าเดียวกันทั้งอ่านไปข้างหน้าและอ่านจากข้างหลัง เช่น 353, 787 และ 2332 รวมถึงจำนวนต่าง ๆ ที่ใช้ตัวเลขเดียวกัน เช่น 22, 555 และ 8888 โดยส่วนใหญ่แล้วจำนวนธรรมชาติจะสามารถกลายเป็นจำนวนพาลินโดรม เมื่อนำมาใช้กับกระบวนการ "ย้อนกลับแล้วบวกเพิ่ม reverse and add" อย่างต่อเนื่อง กระบวนการนี้เรียกว่า "กระบวนการ Lychrel" ตัวอย่างเช่น ตัวเลข 13 จะกลายเป็นจำนวนพาลินโดรมเมื่อบวก 31 ซึ่งเป็นค่าย้อนกลับกันของ 13 และในทำนองเดียวกัน 754 จะกลายเป็นเลขพาลินโดรมเมื่อใช้กระบวนการ Lychrel สองครั้ง: 754 + 457 = 1211; 1211 + 1121 = 2332; และ 255 จะกลายเป็นเลขพาลินโดรมเมื่อใช้กระบวนการ Lychrel สามครั้ง: 255 + 552 = 807; 807 + 708 = 1515; 1515 + 5151 = 6666 อย่างไรก็ตามยังมีตัวเลขบางจำนวนที่ยังไม่สามารถทราบได้แน่ชัดว่าต้องใช้กี่ขั้นตอนในการกลายเป็นจำนวนพาลินโดรมด้วยกระบวนการ Lychrel โดยตัวเลขเหล่านี้ได้แก่ 196, 295 เป็นต้น

[คำอธิบายโปรแกรม]
โปรแกรมจะรับค่าตัวเลขที่ไม่เป็นลบ (non-negative number) ซึ่งมีจำนวนหลักน้อยกว่าหรือเท่ากับ 10 หลัก และจะพยายามสร้างจำนวนพาลินโดรมขึ้นมาจากตัวเลขนั้น ในขั้นสุดท้าย โปรแกรมจะแสดงข้อความที่บอกว่าตัวเลขนั้นเป็นจำนวนพาลินโดรมหรือไม่ หรือเป็นจำนวนที่สามารถกลายเป็นจำนวนพาลินโดรมด้วยกระบวนการ Lychrel ภายในจำนวนกี่ครั้ง

โปรแกรมจะใช้สตริงอักขระ (character string) แทนตัวเลข โดยขนาดของสตริงนั้นเพียงพอสำหรับการใช้กระบวนการย้อนกลับแล้วบวกเพิ่มได้สูงสุด 20 ครั้งกับตัวเลขที่ให้มา แต่ละเอลิเมนต์ของสตริงจะเก็บอักขระตัวเลขที่สอดคล้องกับตัวเลขนั้น เอลิเมนต์แรกเก็บหลักสูงสุดของตัวเลข และเอลิเมนต์สุดท้ายเก็บอักขระ NUL ('\0') ในรูปที่ 1 แสดงการเก็บตัวเลข 2023 ในสตริงอักขระ

ดัชนี	0	1	2	3	4
สตริงอักขระ	'2'	'0'	'2'	'3'	'\0'

ภาพที่ 1 การแทนตัวเลข 2023 ในสตริงอักขระ

โปรแกรมประกอบด้วย 6 ฟังก์ชันดังต่อไปนี้:

(1) `int isNumber(char *str)`
คืนค่า 1 หากสตริงประกอบด้วยอักขระที่เป็นตัวเลขเท่านั้น มิฉะนั้น จะคืนค่า 0

(2) `void reverse(char *rev, const char *str)`
ย้อนกลับสตริง `str` แล้วเก็บผลลัพธ์ไว้ในสตริง `rev` ตัวอย่างเช่น การเรียกใช้ `reverse(x, y)` โดยที่ `y` คือสตริง "12345" ผลลัพธ์จะได้ `x` ที่มีค่าเป็น "54321" โดยที่ `y` จะมีค่าเหมือนเดิมไม่เปลี่ยนแปลง ในที่นี้อาร์กิวเมนต์ทั้งสองจะไม่มีทางทับซ้อนกันได้

(3) `int isPalindrome(const char *str)`
(คำอธิบายถูกละไว้)

(4) `void add(char *addend, const char *adder)`
บวกเลขสองตัวที่เป็นสตริง ตัวอย่างเช่น การบวก "12345" และ "98765" จะได้ผลลัพธ์เป็น "111110" อาร์กิวเมนต์ทั้งสองต้องเป็นสตริงที่มีความยาวเท่ากัน ผลลัพธ์ของการบวกจะถูกเก็บไว้ในตัวแปร `addend`

(5) `int doLychrelProcess(char *strNum, int maxAttempts)`
คำนวณจำนวนครั้งของความพยายามที่ตัวเลขที่กำหนดจะกลายเป็นจำนวนพาลินโดรมโดยการใช้กระบวนการย้อนกลับและการบวกเพิ่ม ในที่นี้อาร์กิวเมนต์ `strNum` จัดเก็บตัวเลขที่เป็นสตริงอักขระ และ `maxAttempts` จัดเก็บจำนวนครั้งสูงสุดของความพยายามที่จะใช้กระบวนการนี้ โดยเมธอดจะมีการส่งคืนจำนวนของความพยายาม (ครั้ง) ที่ได้กระทำ หรือส่งคืน -1 (ลบหนึ่ง) หากไม่สามารถระบุได้ว่าตัวเลขที่กำหนดนั้นจะกลายเป็นจำนวนพาลินโดรมได้ หลังจากได้พยายามตามจำนวนครั้งสูงสุดที่กำหนดไว้

(6) `int main()`
(คำอธิบายถูกละไว้)

โปรแกรมใช้ฟังก์ชันไลบรารีมาตรฐานต่อไปนี้

`size_t strlen(const char *str)`
ส่งคืนความยาวของสตริง str ซึ่งความยาวนี้ไม่นับอักขระแสดงจุดสิ้นสุดสตริง NUL (`'\0'`)

ตัวอย่างผลลัพธ์ของโปรแกรมมีดังนี้

(ตัวอย่างที่ 1)

```
Enter a number: 97
[1] 97 + 79 = 176
[2] 176 + 671 = 847
[3] 847 + 748 = 1595
[4] 1595 + 5951 = 7546
[5] 7546 + 6457 = 14003
[6] 14003 + 30041 = 44044
The number becomes a palindrome within 6 steps.
```

(ตัวอย่างที่ 2)

```
Enter a number: 98
[1] 98 + 89 = 187
[2] 187 + 781 = 968
[3] 968 + 869 = 1837
[4] 1837 + 7381 = 9218
[5] 9218 + 8129 = 17347
[6] 17347 + 74371 = 91718
    ⋮ (ละเว้นการแสดงผล)
[15] 664272356 + 653272466 = 1317544822
[16] 1317544822 + 2284457131 = 3602001953
[17] 3602001953 + 3591002063 = 7193004016
[18] 7193004016 + 6104003917 = 13297007933
[19] 13297007933 + 33970079231 = 47267087164
[20] 47267087164 + 46178076274 = 93445163438
The number does not become a palindrome within 20 steps.
```

[โปรแกรม]

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_ATTEMPTS 20
#define MAX_STRLEN (MAX_ATTEMPTS+11)

int isNumber(char *);
void reverse(char *, const char *);
int isPalindrome(const char *);
void add(char *, const char *);
int doLychrelProcess(char *, int);
                                                    /*** α ***/
int isNumber(char *str) {
    int i;
    int len = strlen(str);
    for (i = 0; i < len; i++) {
        if (   A   ) {
            return 0;
        }
    }
    return len != 0;
}
```

```
void reverse(char *rev, const char *str) {
    int i, j;
    int len = strlen(str);
    for (    B    ) {
        rev[j] = str[i];
    }
    rev[j] = '\0';
}

int isPalindrome(const char *str) {
    int i;
    int len = strlen(str);
    for (i = 0; i < len / 2; i++) {
        if (str[i] !=    C    ) {
            return 0;
        }
    }
    return len != 0;
}

void add(char *addend, const char *adder) {
    int i, digit;
    int carry = 0;
    int len = strlen(addend);
    for (i = len - 1; i >= 0; i--) {
        digit = addend[i] - '0' + adder[i] - '0' + carry;
        addend[i] =    D    + '0';
        carry =    E    ;
    }
    if (carry) {
        for (i = len + 1; i > 0; i--) {
            addend[i] = addend[i - 1];
        }
        addend[0] = carry + '0';
    }
}
```

/*** β ***/

```
int doLychrelProcess(char *strNum, int maxAttempts) {
    int numAttempts = 0;
    char strRev[MAX_STRLEN];
    while (!isPalindrome(strNum) && ++numAttempts <= maxAttempts) {
        reverse(strRev, strNum);
        printf("[%d] %s + %s = ",
               numAttempts, strNum, strRev);              /*** γ ***/
        add(strNum, strRev);
        printf("%s\n", strNum);
    }
    if (numAttempts <= maxAttempts) {
        return numAttempts;
    }
    return -1;
}

int main() {
    char strNum[MAX_STRLEN];
    int steps = 0;
    printf("Enter a number: ");
    scanf("%10s", strNum);
    if (!isNumber(strNum)) {
        printf("Please input a non-negative integer.\n");
        return 0;
    }
                                                           /*** δ ***/
    steps = doLychrelProcess(strNum, MAX_ATTEMPTS);
    if (steps == [    F    ]) {
        printf("The number does not become a palindrome within "
               "%d steps.\n", MAX_ATTEMPTS);
    }
    else if (steps == 0) {
        printf("The number is a palindrome number.\n");
    }
    else {
        printf("The number becomes a palindrome within "
               "%d steps.\n", steps);
    }
    return 0;
}
```

คำถามย่อย 1

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในตัวโปรแกรม

กลุ่มคำตอบสำหรับ A

a) `str[i] < '0' && str[i] > '9'`
b) `str[i] < '0' || str[i] > '9'`
c) `str[i] > '0' && str[i] < '9'`
d) `str[i] > '0' || str[i] < '9'`

กลุ่มคำตอบสำหรับ B

a) `i = 0, j = len; i < len - 1; i++, j--`
b) `i = 0, j = len; i < len; i++, j--`
c) `i = len - 1, j = 0; i >= 0; i--, j++`
d) `i = len, j = 0; i >= 0; i--, j++`

กลุ่มคำตอบสำหรับ C

a) `str[i / 2]`
b) `str[len - 1 - i]`
c) `str[len - 1]`
d) `str[len - i]`
e) `str[len]`

กลุ่มคำตอบสำหรับ D และ E

a) `digit % 10`
b) `digit & 1`
c) `digit * 10`
d) `digit / 10`
e) `digit >> 1`

กลุ่มคำตอบสำหรับ F

a) `-1`
b) `0`
c) `1`
d) `MAX_ATTEMPTS`

คำถามย่อย 2

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพิ่มเติมลงในแต่ละช่องว่าง [ ] แต่ละช่องในคำอธิบายด้านล่างนี้ ในที่นี้ ให้สมมติว่าคำตอบที่ถูกต้องได้ถูกเติมลงในช่องว่าง [A] ถึง [F] ในตัวโปรแกรมเรียบร้อยแล้ว

โปรแกรมดำเนินการกับตัวเลขที่มีเลขศูนย์นำหน้าแตกต่างจากตัวเลขเดียวกันที่ไม่มีศูนย์นำหน้า ตัวอย่างเช่น ผลลัพธ์ของโปรแกรมที่มีอินพุตเป็น 0097 จะแตกต่างจากเอาต์พุตของ 97

```
Enter a number: 0097
[1] 0097 + 7900 = 7997
The number becomes a palindrome within 1 steps.
```

ฟังก์ชันใหม่ชื่อว่า skipLeadingZeros ถูกเพิ่มเข้าไปในโปรแกรมตามคำแนะนำต่อไปนี้ เพื่อให้ได้ผลลัพธ์เดียวกัน แม้ว่าตัวเลขนั้นจะมีเลขศูนย์นำหน้าหรือไม่ก็ตาม

(1) แทรกบรรทัดนี้ ณ บรรทัด /*** α ***/

```
const char *skipLeadingZeros(const char *);
```

(2) แทรก 9 บรรทัดต่อไปนี้ ณ บรรทัด /*** β ***/

```
const char *skipLeadingZeros(const char *str) {
    while (*str == '0') {
        str++;
    }
    if (*str == '\0') {
        return [ G ];
    }
    return str;
}
```

ฟังก์ชันนี้ส่งคืนพอยน์เตอร์ที่ชี้ไปยังอักขระตัวแรกที่ไม่ใช่ตัวเลขศูนย์ แต่ถ้า str ประกอบด้วยเลขศูนย์ทั้งหมด ก็จะส่งคืนพอยน์เตอร์ที่ชี้ไปยังอักขระศูนย์ตัวสุดท้าย

(3) แทนที่บรรทัด /*** γ ***/ ด้วยบรรทัดนี้

```
numAttempts, strNum, skipLeadingZeros(strRev));
```

(4) แทรกบรรทัดนี้ ณ บรรทัด /*** δ ***/

```
memmove( [ H ] );
```

ในที่นี้, memmove เป็นฟังก์ชันไลบรารี่มาตรฐานที่ถูกกำหนดไว้ดังนี้

```
void *memmove(void *dest, void *src, size_t len)
```

ฟังก์ชันนี้คัดลอกสตริงอักขระความยาว len จากสตริง src ไปยังสตริง dest โดยที่สตริง src และสตริง dest สามารถทับซ้อนกันได้

กลุ่มคำตอบสำหรับ G

a) *str
b) str + 1
c) str - 1
d) NULL

กลุ่มคำตอบสำหรับ H

a) skipLeadingZeros(strNum), strNum, strlen(strNum)
b) skipLeadingZeros(strNum), strNum, strlen(strNum) + 1
c) strNum, skipLeadingZeros(strNum), strlen(skipLeadingZeros(strNum))
d) strNum, skipLeadingZeros(strNum), strlen(skipLeadingZeros(strNum)) + 1

สำหรับข้อสอบข้อที่ **Q7** และ **Q8 ให้เลือกทำเพียงหนึ่ง**ในสองข้อ

Q8. อ่านคำอธิบายของโปรแกรม และโค้ดโปรแกรมภาษาจาวาต่อไปนี้ แล้วตอบคำถามย่อย 1 และ 2

[คำอธิบายโปรแกรม]

กระเป๋า (bag) เป็นคอลเลกชันของวัตถุ (object) แบบจำกัดที่ไม่ได้มีการจัดเรียงลำดับใดเป็นพิเศษ โดยภายในกระเป๋าสามารถบรรจุไอเท็มที่ซ้ำกันได้ โครงสร้างข้อมูลของกระเป๋าสามารถทำงานได้ดังนี้

i) คืนค่าจำนวนวัตถุทั้งหมดที่อยู่ในกระเป๋า
ii) เพิ่มไอเท็มที่กำหนดลงในกระเป๋า
iii) นับจำนวนไอเท็มที่กำหนดว่ามีกี่อันในกระเป๋า
iv) ตรวจสอบว่ากระเป๋าเต็มหรือไม่
v) ตรวจสอบว่าภายในกระเป๋าว่างเปล่าหรือไม่

โครงสร้างข้อมูลกระเป๋าถูกออกแบบโดยการใช้อินเทอร์เฟซ (interface) ตั้งชื่อว่า `Bag`
ในตารางที่ 1 จะแสดงคำอธิบายโดยสรุปของห้าเมธอดนี้

ตารางที่ 1 แสดงคำอธิบายของห้าเมธอด

เมธอด	คำอธิบาย
`int size()`	ตรวจสอบและคืนค่าเป็นจำนวนไอเท็มทั้งหมดที่อยู่ในกระเป๋า
`boolean add(T entry)`	เพิ่มอาร์กิวเมนต์ entry ลงในกระเป๋า หากสามารถเพิ่มได้สำเร็จ จะคืนค่าจริง (true) หากไม่จะคืนค่าเท็จ (false)
`int count(T entry)`	นับและคืนค่าจำนวนที่ค้นพบอาร์กิวเมนต์ entry ที่ระบุ จากเอลิเมนต์ทั้งหมดที่อยู่ในกระเป๋า
`boolean isFull()`	ตรวจสอบว่ากระเป๋าเต็มหรือไม่ หากกระเป๋าเต็มจะคืนค่าจริง (true) กรณีอื่นจะคืนค่าเท็จ (false)
`boolean isEmpty()`	ตรวจสอบว่าภายในกระเป๋าว่างเปล่าหรือไม่ หากกระเป๋าว่างเปล่าจะคืนค่าจริง (true) กรณีอื่นจะคืนค่าเท็จ (false)

โปรแกรมที่ 1 แสดงถึงอินเทอร์เฟซ (interface) `Bag` โดยที่ `T` เป็นประเภทเจเนริก (generic type)
โปรแกรมที่ 2 แสดงถึงคลาส `ArrayBag` ที่มีการใช้งานอินเทอร์เฟซ `Bag` และมี 2 คอนสตรัคเตอร์: (1) กำหนดค่าเริ่มต้นของความจุ (capacity)ให้เป็น 25 และ (2) กำหนดค่าเริ่มต้นของความจุจากค่าที่ส่งเข้ามา ซึ่งเมธอดต่าง ๆ ที่แสดงในตารางที่ 1 จะถูกนำมาใช้ในโปรแกรมที่ 2

[โปรแกรมที่ 1]

```
public interface Bag<T> {
    public int size();
    public boolean add(T entry);
    public int count(T entry);
    public boolean isFull();
    public boolean isEmpty();
}
```

[โปรแกรมที่ 2]

```
public class [   A   ] implements Bag<T> {
    private final [   B   ] bag;
    private int numberOfEntries = 0;
    private static final int DEFAULT_CAPACITY = 25;

    public ArrayBag() {
        this(DEFAULT_CAPACITY);
    }

    public ArrayBag(int desiredCapacity) {
        @SuppressWarnings("unchecked")
        T[] tempBag = (T[])new Object[desiredCapacity];
        bag = tempBag;
    }

    public int size() {
        return [   C   ];
    }

    public boolean add(T entry) {
        if (entry == null) {
            throw new NullPointerException();
        }
        boolean result = true;
        if (isFull()) {
            result = false;
        } else {
            bag[   D   ] = entry;
        }
        return result;
    }

    public int count(T entry) {
        int counter = 0;
        for (T item : bag) {
            if (entry.equals(item)) {
                counter++;
            }
        }
        return counter;
    }
```

```
    public boolean isFull() {
        return [   C   ] >= [   E   ];
    }

    public boolean isEmpty() {
        return [   C   ] == [   F   ];
    }
}
```

คำถามย่อย 1

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในโปรแกรมที่ 2

กลุ่มคำตอบสำหรับ A และ B

a) `ArrayBag` b) `ArrayBag<T>` c) `ArrayBag<T[]>`
d) `Bag` e) `Bag<T>` f) `Bag<T[]>`
g) `T` h) `T[]`

กลุ่มคำตอบสำหรับ C ถึง F

a) `0` b) `1`
c) `bag.length` d) `bag.length - 1`
e) `bag.length++` f) `numberOfEntries`
g) `numberOfEntries++` h) `numberOfEntries--`

คำถามย่อย 2

มีการเพิ่มเมธอดอีกสี่เมธอดเข้ามาในคลาส ArrayBag

ตารางที่ 2 แสดงคำอธิบายโดยสรุปของสี่เมธอดที่ถูกเพิ่มเข้ามา

ตารางที่ 2 แสดงคำอธิบายของสี่เมธอดที่ถูกเพิ่มเข้ามา

เมธอด	คำอธิบาย
int indexOf(T entry)	ค้นหาและส่งคืนดัชนีตัวแรกของอาร์กิวเมนต์ entry ภายในกระเป๋า หากหาไม่พบ entry จะคืนค่า -1 (ลบหนึ่ง)
T get(int index)	คืนค่าเอลิเมนต์ ณ ตำแหน่งที่ระบุโดยอาร์กิวเมนต์ index
T remove(int index)	ลบเอลิเมนต์ ณ ตำแหน่งที่ระบุโดยอาร์กิวเมนต์ index หากสามารถลบได้สำเร็จจะคืนค่าเอลิเมนต์ที่โดนลบไป หากไม่สำเร็จจะคืนค่า null ในกรณีที่เอลิเมนต์ที่โดนลบไปไม่ใช่เอลิเมนต์ลำดับสุดท้ายในกระเป๋า ให้ย้ายเอลิเมนต์ลำดับสุดท้ายมาแทนที่และกำหนดเอลิเมนต์ลำดับสุดท้ายให้เป็น null
boolean removeEntry(T entry)	ค้นหาและลบเอลิเมนต์ entry ณ ตำแหน่งแรกที่ค้นพบ หากสำเร็จจะค่า true หากไม่สำเร็จจะคืนค่า null ในกรณีที่เอลิเมนต์ที่โดนลบไปไม่ใช่เอลิเมนต์ลำดับสุดท้ายในกระเป๋า ให้ย้ายเอลิเมนต์ลำดับสุดท้ายมาแทนที่และกำหนดเอลิเมนต์ลำดับสุดท้ายให้เป็น null

โปรแกรมที่ 3 แสดงถึงสี่เมธอดที่เพิ่มเข้ามาในคลาส ArrayBag

[โปรแกรมที่ 3]

```
public int indexOf(T entry) {
    for (int i = 0; i < numberOfEntries; i++) {
        if (entry.equals(bag[i])) {
            return i;
        }
    }
    return -1;
}

public T get(int index) {
    if (index < 0 || index >= numberOfEntries) {
        throw new ArrayIndexOutOfBoundsException();
    }
    return bag[index];
}
```

```
public T remove(int index) {
    T result = null;
    if (index >= 0 && index < numberOfEntries) {
        result = [   G   ];
        [   G   ] = [   H   ];
        [   H   ] = null;
        numberOfEntries--;
    }
    return result;
}

public boolean removeEntry(T entry) {
    T result = remove(indexOf(entry));
    return entry.equals(result);
}
```

จากกลุ่มคำตอบด้านล่าง ให้เลือกคำตอบที่ถูกต้องเพื่อเติมลงในช่องว่าง [ ] แต่ละช่องในโปรแกรมที่ 3

กลุ่มคำตอบสำหรับ G และ H

a) `bag[bag.length - 1]`
b) `bag[bag.length]`
c) `bag[index - 1]`
d) `bag[index]`
e) `bag[numberOfEntries - 1]`
f) `bag[numberOfEntries]`

Company names and product names appearing in the test questions are trademarks or registered trademarks of their respective companies. Note that the ® and ™ symbols are not used within the text.