การบ้านครั้งที่ 2 Basic Methods, Decision, and Array

จุดมุ่งหมายของปฏิบัติการ

- เพื่อฝึกทักษะเบื้องต้นในการ implement คลาสอย่างง่าย
- เพื่อให้สามารถเขียนโปรแกรมที่ใช้คำสั่งทางเลือก และคำสั่งการทำซ้ำได้
- เพื่อฝึกทักษะการ implement คลาสอย่างง่ายที่นำเข้าและพิมพ์ผลลัพธ์ทางหน้าจอ
- ในทุกข้อสามารถเขียนเพื่อให้วนซ้ำรับค่าจนกว่าผู้ใช้ไม่ต้องการทำงานต่อไป

1. การ implement ทางเลือก โดยใช้ switch-case

ให้สร้างโปรแกรมเพื่อรับค่าตัวเลขโรมัน 1-6 จากผู้ใช้ จากนั้นพิมพ์ตัวเลขเป็ นข้อความ ในกรณีที่ใส่ตัวเลขไม่ถูกต้อง หรือไม่อยู่ในช่วง ให้พิมพ์ข้อความ “Unknown Number.”

ข้อกำหนด

- โค้ดข้อนี้ให้ใช้คำสั่ง switch-case ในการทำงาน
- ให้แยกเป็ นเมท็อดสถิตย์สำหรับการรับค่า และเมท็อดสถิตย์การแปลงตัวเลขเป็ นข้อความ

ตัวอย่างการทำงานของโปรแกรม เช่น กรณีเลขโรมันที่เป็ นได้

```
Input the roman number: II
It is Two.
```

กรณีเลขโรมันที่ไม่ถูกต้อง

```
Input the roman number: 9
Unknown Number.
```

ตารางเลขโรมัน

เลขโรมัน	I	II	III	IV	V	VI
ข้อความ	One	Two	Three	Four	Five	Six

2. การ implement ทางเลือกแบบซ้อน ให้ออกแบบและเขียนคลาสเพื่อใช้ในการคำนวณค่าใช้บริการโทรศัพท์มือถือ โดยมีเกณฑ์การคำนวณดังนี้

- การใช้บริการที่เป็ นเศษของนาที (น้อยกว่านาที) คิดเป็ น 1 นาที (ให้ใช้ Math.ceil เพื่อปัดเศษขึ้น)
- บริการน้อยกว่า 10 นาที คิดนาทีละ 3.50 บาท
- บริการตั้งแต่ 10 นาทีแต่ไม่ถึง 1 ชั่วโมง คิด 20 นาทีแรก นาทีละ 3 บาท นาทีถัดไปนาทีละ 1 บาท
- บริการตั้งแต่ 1 ชั่วโมงขึ้นไป คิดอัตราตายตัวนาทีละ 1.5 บาท

กรณีเวลามีค่าน้อยกว่า 0 ให้คิดค่าบริการ 0 บาท โปรแกรมนี้รับค่าเวลาจากผู้ใช้(ชนิด double) เพียงหนึ่งครั้ง แล้วพิมพ์ค่าใช้บริการ ปรับปรุงให้วนซ้ำเพื่อรับค่าเวลาจนกว่าผู้ใช้จะ พิมพ์ Q

ตัวอย่างการทำงานของโปรแกรมกรณีต่าง ๆ เช่น

```
Enter minutes(Q= Quit):8.5
You have to pay 31.5

Enter minutes(Q= Quit):18.5
You have to pay 57.0

Enter minutes(Q= Quit):21.4
You have to pay 62.0

Enter minutes(Q= Quit):60.1
You have to pay 91.5

Enter minutes(Q= Quit):-60.1
You have to pay 0.0
```

จุดมุ่งหมายของการบ้านข้อ 3

- เพื่อให้สามารถใช้ ArrayList ในการเก็บกลุ่มของวัตถุ รวมถึงการเข้าถึงแต่ละวัตถุในกลุ่มได้
- เพื่อให้สามารถ implement algorithm พื้นฐานที่ทำงานกับกลุ่มของวัตถุได้
- เพื่อสร้าง array สำหรับเก็บข้อมูลได้
- เพื่อให้สามารถส่งผ่าน arrays เข้า/คืนจาก method ได้

ข้อกำหนดให้**ใช้ ArrayList ในการเก็บกลุ่มของวัตถุ รวมถึงการเข้าถึงแต่ละวัตถุในกลุ่มได้** และ ให้ใช้โค้ดของคลาส **Account มีรายละเอียดของโค้ดดังนี้ในการทำการบ้านข้อที่ 3**

```
public class Account {
 private double balance;
 private int accountNumber;
 private static int lastAccountNumber = 0;
 public Account(double intialBalance){
  balance = intialBalance;
  accountNumber = lastAccountNumber + 1;
  lastAccountNumber = accountNumber;
 }

 public void deposit(double depositAmount){
  balance += depositAmount;
 }

 public boolean withdraw(double withdrawAmount){
  if (withdrawAmount > balance){
   System.out.println("Insufficient Funds!!!");
   return false;
  } else {
   balance -= withdrawAmount;
   return true;
  }
 }
```

3.1 ให้นักศึกษาสร้าง class Bank ที่จะจัดเก็บ list ของบัญชี โดยคลาส Bank มีคุณสมบัติดังต่อไปนี้

a) accountList เป็นตัวแปรวัตถุที่มีชนิดเป็น ArrayList ใช้เก็บ Account

b) method void addAccount(Account acct) ซึ่งจะเพิ่มวัตถุ Account acct ต่อท้ายใน accountList

c) method Account getAccount(int index) ซึ่งจะคืนวัตถุ Account ที่ตำแหน่งที่ระบุโดย index

d) method ArrayList<Account> findAccounts(double amount) คืน ArrayList ที่เก็บบัญชีที่มียอดเงินคงเหลือในบัญชีมากกว่าหรือเท่ากับ amount หากไม่พบบัญชีตามเงื่อนไข ให้คืน null

e) method Account findMin() คืน null หากไม่มี account เลย ไม่เช่นนั้นคืน Account ที่มียอดเงินน้อยที่สุดอันดับแรกที่พบใน accountList

f) method Account findMax() คืน null หากไม่มี account เลย ไม่เช่นนั้นคืน Account ที่มียอดเงินมากที่สุดอันดับแรกที่พบใน accountList

g) method void addInterest() ซึ่งจะเพิ่มดอกเบี้ย 3% ให้แต่ละ account ใน accountList

h) method double getTotal() ซึ่งจะหาค่าผลรวมของเงินใน accountList ทั้งหมด

i) method double getAverage() ซึ่งจะหาค่าเฉลี่ยของเงินใน accountList ทั้งหมด

3.2 สร้าง class BankTest เพื่อทดสอบสร้าง Bank และให้วนลูปเพื่อรับคำสั่งจาก User ในการจัดการ accounts ของ Bank ที่สร้าง โดย

กด 1 เพื่อเพิ่มบัญชีใหม่ โดยให้ user กำหนดรายละเอียดของ Account สร้าง Account ใหม่ และ เรียก method addAccount เพื่อเพิ่มบัญชีใหม่ให้ Bank

กด 2 เพื่อดึงวัตถุ Account ในตำแหน่ง index ใน array list ที่เก็บบัญชีทั้งหมดของ Bank และ พิมพ์แสดงรายละเอียดของบัญชี

กด 3 เพื่อค้นหาและพิมพ์รายละเอียดบัญชีที่มียอดมากกว่าหรือเท่ากับ amount ที่ user ระบุ โดยให้ user ใส่ค่า amount แล้วเรียกใช้ method findAccounts และ แสดงรายละเอียดของ Account ที่มียอดเงินในบัญชีน้อยที่สุด และ เรียกใช้ method findMax() เพื่อค้นหาบัญชีที่มียอดสูงสุดและแสดงรายละเอียดของบัญชีนั้น

กด 4 เพื่อเพิ่มดอกเบี้ยให้แก่ทุกบัญชี โดยให้เรียก method addInterest() และ แสดงยอดเงินทุกบัญชีหลังเพิ่มดอกเบี้ย

กด 5 เพื่อแสดงยอดเงินทั้งหมดรวมจากทุกบัญชี โดยเรียกใช้ method getTotal() และ แสดงยอดเงินเฉลี่ยจากทุกบัญชี โดยเรียกใช้ method getAverage()

กด Q เพื่อสิ้นสุดโปรแกรม

เมื่อสิ้นสุดแต่ละเมนู 1-5 ให้ user ตัดสินว่าจะทำเมนูใดต่อ โดยให้ขึ้นข้อความแสดงเมนู 1-5 และ Q ตามรายละเอียดที่อธิบาย

3.3 ให้นักศึกษา capture screen แสดงภาพการเรียกใช้ Class BankTest และผลการทดสอบการเรียกใช้เมนูต่างๆเพื่อทดสอบการสร้าง Account 3 บัญชีที่มียอด 120, 2567, 250,000 ด้วยเมนูที่ 1 และแสดงรายละเอียดของแต่ละบัญชีหลังเพิ่มเข้า Bank ด้วยเมนูที่ 2 แสดงบัญชีที่มียอดมากกว่า 1,000 แสดงบัญชีที่ยอดเงินนิ้ยที่สุด และ มากที่สุดด้วยเมนู 3 เพิ่มยอดเงินจากการคำนวณดอกเบี้ยที่ได้รับให้แก่บัญชีทุกบัญชี ด้วยเมนู 4 จากนั้น แสดงยอดเงินรวมทั้งหมดจากทุกปัญชี ละยอดเงินเฉลี่ย ด้วยเมนู 5 จากนั้นสิ้นสุดโปรแกรมด้วยการกดเมนู Q

ข้อกำหนดการส่งข้อที่ 3 ให้ส่ง 2 ไฟล์.java และ ไฟล์ pdf แสดงรูปภาพที่ capture จากหน้าจอที่แสดงผลการรันโปรแกรมตามรายละเอียดข้อ 3.3