โจทย์ปัญหาข้อที่ 1 :
A => มี Object เศษส่วนที่1, เศษส่วนที่2, ผลรวมของเศษส่วนทั้ง2
B => มี Attribute fraction ไว้เก็บค่าเศษ และ part ไว้เก็บค่าส่วน
C => มี Method sum ไว้ใช้หาผลรวมของเศษส่วนที่1 และเศษส่วนที่2
Method getValue ไว้return ค่าของเศษส่วนนั้นๆ
Constructor
D => ต้องสร้าง Class 1 Class ชื่อว่า Fraction
E => UML

Fraction
- fraction : int - part : int
+ Fraction(fraction: int, part: int) + sum(second:Fraction): void + getValue(): Fraction

โจทย์ปัญหาข้อที่ 2 :
A => มี Object ของลูกค้า (customer)
B => มี Attribute collectedStamp ไว้เก็บจำนวนของแสตมป์ที่มี
money ให้ทราบจำนวนเงินที่ลูกค้ามี
name ชื่อของลูกค้า
C => มี Method calculateStamp ไว้ใช้คำนวณว่าลูกค้าจะได้แสตมป์กี่ดวงและนำเงินที่ใช้หักจากการแลกแสตมป์
getStamp ไว้ใช้คืนค่าว่ามีแสตมป์กีดวง
getMoney ไว้คืนค่าว่ามีเงินกี่บาท
constructor
D => ต้องสร้าง Class 1 Class ชื่อว่า Customer
E => UML

Customer
- collectedStamp: int - name : String - money : float
+ Customer(name: String, money: float) + calculateStamp(usedMoney: float): void + getStamp(): int + getMoney(): float

โจทย์ปัญหาข้อที่ 3 :
A => มี Object number คือ object ของเลขที่รับเข้ามา
B => มี Attribute number เก็บค่าของตัวเองนั้นๆ
C => มี Method compare ไว้ใช้เปรียบเทียบตัวเลขที่รับมาว่าตรงกับเลขเป้าหมายหรือไม่
constructor
D => ต้องสร้าง Class 1 Class ชื่อว่า Number
E => UML

Number
- Number: int
+ Number(number: int) + compare(input: Number): String

โจทย์ปัญหาข้อที่ 4 :
A => มี Object playerOne, PlayerTwo
B => มี Attribute hidden เก็บเลขที่ผู้เล่นที่ 1 และ 2 ให้ทาย
seek เก็บค่าเลขที่ผู้เล่น 1 และ 2 ทาย
C => มี Method checkNum ไว้เช็คค่าที่ทายกับตัวเลือกที่ให้ทาย
constructor
D => ต้องสร้าง Class 1 Class ชื่อว่า Player
E => UML

Player
- hidden: int - seek: int
+ Player(hidden: int, seek: int) + checkNum(seek: int): String

โจทย์ปัญหาข้อที่ 5 :
A => มี Object hangManWord
B => มี Attribute hangManWord เก็บคำที่ทาย
alpha เก็บอักษรที่ทาย
life จำนวนชีวิตที่เหลือ
C => มี Method getLife คืนค่าชีวิตที่เหลือ
inputAlpha รับตัวอักษรที่ทายและเช็คว่าถูกหรือผิดถ้าผิดให้ลดชีวิตไป1
constructor
D => ต้องสร้าง Class 1 Class ชื่อว่า HangMan
E => UML

HangMan
- alpha: char - life: int - hangManWord: String
+ HangMan(hangManWord: String) + getLife(): int + inputAlpha(alpha:char): void

โจทย์ปัญหาข้อที่ 6 :
A => มี Object pokeBall, greatBall, ultraBall, masterBall
B => มี Attribute price ราคา
sellFor ราคาขาย
ballValue
C => มี Method getPrice คืนค่าราคา
getSellFor คืนค่าราคาขาย
getBallValue คืนค่าball value
constructor
D => ต้องสร้าง Class 1 Class ชื่อว่า PokeBall
E => UML

MonsterBall
- price: int - sellFor: int - ballValue: int
+ MonsterBall() + getPrice(): int + getSellFor(): int + getBallValue: int

โจทย์ปัญหาข้อที่ 7 :

A => มี Object dictionary

B => มี Attribute word[] เก็บคำศัพท์

word เก็บศัพท์

meaning เก็บความหมาย

example เก็บตัวอย่างการใช้

C => มี Method getType คืนค่าชนิดของศัพท์

getMeaning คืนค่าความหมายของศัพท์

getExample คืนค่าตัวอย่างการใช้

addWord เพิ่มคำศัพท์ลงในdictionary

constructor

D => ต้องสร้าง Class 2 Class ชื่อว่า Dictionary, Word

E => UML

Dictionary
- word: Word[]
+ Dictionary() + getType(word: String): String + getMeaning(word: String): String + getExample(word: String): String + addWord(word: Word): void

Word
- word: String - meaning: String - example: String
+ Word(word:String, meaning: String, example: String)