ปัญหา อารีน่าของ SU 3 (AoSU3) [10 คะแนน]

[ทรัพยากรการคำนวณ: 1 วินาที, 32 MB]

ในข้อที่แล้ว เราได้ศึกษากลไกในการตรวจสอบทีมว่าถูกต้องตามกฎหรือไม่ ส่วนในข้อนี้ เราจะมาจัดการประเด็นเรื่องทีมที่สมดุล ซึ่งเกิดจากการที่ทีมมีฮีโร่ที่หลากหลายชนิด

สำหรับการแยกชนิดของฮีโร่นั้น ในข้อนี้เราต้องการสร้างคลาสที่รับทอด (inherit) คลาส `Hero` จากข้อที่แล้วมาเพื่อแยกเป็นฮีโร่ชนิดต่าง ๆ โดยคลาสที่สืบทอดมานี้มีทั้งหมด 6 คลาสคือ `1. Carry, 2. Fighter, 3. Mage, 4. Tank, 5. Support, 6. Assassin`[1]

ในปัญหานี้ เราจะยังไม่เพิ่มเติมสิ่งใดให้กับคลาสทั้งหกนี้หลังการรับทอด นอกจากเรียกใช้ตัวสร้างของคลาส `Hero` แต่จะเน้นไปเฉพาะเรื่องทีมที่สมดุลแทน นั่นคือเราจะต่อเติมคลาส `Team` ที่จัดการทีมผู้เล่นขนาด 5 คนโดยมีเมธอดสำคัญที่เราจะเขียนขึ้นมาดังนี้

1. ตัวสร้าง ซึ่งจะรับพารามิเตอร์เป็นอาเรย์หนึ่งตัวเช่นเดิมกับข้อที่แล้ว และจะทำการตรวจสอบว่าทีมถูกกฎหรือไม่ (ทำผ่านเมธอดสองอันจากข้อที่แล้ว) ในกรณีที่ทีมไม่ถูกกฎก็จะ `return` โดยไม่บันทึกข้อมูลของพารามิเตอร์ที่ส่งมา แต่หากถูกกฎก็จะรายงานว่าทีมสมดุลหรือไม่ผ่านการเรียกใช้เมธอด `isBalanced` อย่างไรก็ตาม ไม่ว่าทีมจะสมดุลหรือไม่ หากทีมถูกกฎ ตัวสร้างก็จะบันทึกข้อมูลทีมลง `members` ด้วยวิธีที่เดียวกับข้อที่แล้ว

---

[1] เป็นไปได้ว่าผู้พัฒนาเกมอย่าง RoV จะไม่ใช้วิธีนี้ เพราะที่จริงมันอาจจะสร้างปัญหาต่อการพัฒนาเกมในบริบทอื่น ๆ และตัวเกมจะใช้ประโยชน์จากวิธีการรับทอดได้ไม่ดีนัก ต่างกับเกมอย่าง Pokemon Go ที่การรับทอดจะทำให้การพัฒนาง่ายขึ้นมาก อย่างไรก็ตามแนวคิดนี้มีความสำคัญกับการศึกษาด้านการเขียนโปรแกรมมาก จึงยกมาเป็นประเด็นในคำถามนี้

2. เมธอด `isBalanced` สามารถตรวจสอบได้ว่าทีมสมดุลหรือไม่ หากทีมสมดุลจะคืนค่า `true` และหากไม่สมดุลจะคืนค่า `false` ซึ่งความสมดุลแบบที่เราจะพิจารณานี้ จะนับง่าย ๆ ว่าหากทีมมีฮีโร่ 4 ชนิดแรกอยู่ครบ[2] จะถือว่าสมดุล เช่น ถ้าฮีโร่ทั้ง 5 คนที่เลือกมาเป็นประเภท `Carry, Mage, Tank, Assassin` และ `Fighter` ตามลำดับ หรือ `Mage, Tank, Mage, Carry,` และ `Fighter` ตามลำดับ ทั้งสองแบบนี้ถือว่าทำให้ทีมสมดุล และเมธอดจะแสดงข้อความว่า `balanced team` และคืนค่า `true`

แต่หากฮีโร่ทั้ง 5 ที่เลือกมาเป็นชนิด `Carry, Fighter, Fighter, Assassin,` และ `Carry` ตามลำดับ หรือ `Fighter, Mage, Tank, Tank,` และ `Support` ตามลำดับ สองแบบหลังนี้ถือว่าไม่สมดุล เพราะมีฮีโร่ 4 ชนิดหลักไม่ครบ ซึ่งในแบบแรก เมธอดจะต้องแสดงข้อความแจ้งต่อผู้เล่นว่าทีมต้องการ `Mage` และ `Tank` ส่วนแบบที่สองระบบจะต้องแสดงข้อความว่าทีมต้องการ `Carry` และจะคืนค่า `false` ไม่ว่าทีมจะไม่สมดุลในรูปแบบใดก็ตาม

หมายเหตุ ในตอนต้นเมธอดนี้ก็จะตรวจจำนวนสมาชิกในทีมก่อนในลักษณะเดียวกันกับ `is5MemberTeam` ซึ่งถ้าไม่ได้มีสมาชิก 5 คนพอดี เมธอด `isBalanced` จะคืนค่า `false` กลับไปโดยไม่พิมพ์ข้อความใด ๆ ออกมา และเมธอด `isBalanced` จะยังทำการตรวจสอบความสมดุลของทีม แม้ว่าฮีโร่ในทีมจะมีซ้ำกันก็ตาม

จงสร้างคลาสฮีโร่เพิ่มเติมอีก 6 ชนิดและต่อเติมคลาส `Team` จากข้อที่แล้วให้ทำงานได้ตามข้อกำหนดดังกล่าว และใช้เมธอด `main` ที่กำหนดให้ และห้ามทำการเปลี่ยนแปลงใด ๆ ในคลาส `AoSU3` ยกเว้นส่วนที่มีการเว้นไว้สำหรับการเติมโค้ดโดยเฉพาะ ถ้าหากมีการแก้ไขใด ๆ ในคลาส `AoSU3` นอกเหนือไปจากนั้นและมีผลต่อคำตอบแล้วได้

[2] คือมี `Carry, Fighter, Mage` และ `Tank` ครบทั้งสี่ชนิดนี้

คะแนน คะแนนดังกล่าวจะกลายเป็นแต้มลบ เช่น ถ้าได้คะแนน 60 ก็จะถูกปรับเป็น -60 แทน

```
import java.util.Scanner;

class Team {
// Enter your code here.
}
class Carry // Enter your code here.
{
  // Enter your code here.
}
// อย่าลืมคลาสฮีโร่อีก 5 ชนิดด้วยนะ

public class AoSU3 {
  public static void main(String[] args) {
    Scanner scan = new Scanner(System.in);
    int N = scan.nextInt();
    ________ members ________;
    if(N > 0) {
      members = new ________;
      for(int i = 0; i < N; ++i) {
        int type = scan.nextInt();
        String name = scan.next();
        int hp = scan.nextInt();
        int attkType = scan.nextInt();
        int attkDmg = scan.nextInt();
        int phyDef = scan.nextInt();
        int magDef = scan.nextInt();

        if(type == 1)
          ________ = new Carry(name, hp, attkType,
                               attkDmg, phyDef, magDef);
        ________ (มีโค้ดตรงนี้ตามมาอีกหลายบรรทัด) ________
      }
    }
    Team t = new Team(members);
    System.out.println(t.is5MemberTeam(members));
    System.out.println(t.isValid(members));
```

```
    System.out.println(t.isBalanced(members));
  }
}
```

ข้อมูลเข้า

บรรทัดแรก	เป็นค่า N ซึ่งคือจำนวนฮีโร่ที่จะอยู่ในทีม โดยที่ 0 <= N <= 20
บรรทัดที่สอง	เป็นข้อมูลฮีโร่ตัวแรก ซึ่งอยู่ในลักษณะเดียวกันกับข้อ AoSU2 แต่ก่อนจะเป็นข้อมูลชื่อ จะมีเลขจำนวนเต็ม 1 ถึง 6 ค่าใดค่าหนึ่งมาด้วยเพื่อระบุชนิดของฮีโร่ โดยเลขกับชนิดเป็นแบบเดียวกับที่เขียนไว้ในตอนต้นของคำถาม
อีก N-1 บรรทัด	เป็นข้อมูลฮีโร่ที่เหลือ หนึ่งตัวต่อบรรทัด

ผลลัพธ์

เป็นสิ่งที่ตัวสร้างพิมพ์ออกมา ตามด้วยสิ่งที่พิมพ์และค่าที่คืนจาก `is5MemberTeam`, `isValid` และ `isBalanced` ตามลำดับ

ตัวอย่าง

ข้อมูลเข้า	ผลลัพธ์
5 1 Van 1000 1 70 50 20 3 Zill 800 1 90 40 40 4 Taara 1000 1 -70 50 20 6 Murad 1000 2 70 50 20 2 Maloch 1000 1 70 50 20	warning: value cannot be negative full team valid hero selection balanced team full team true valid hero selection true balanced team true

5 3 Zill 800 1 90 40 40 4 Taara 1000 1 -70 50 20 1 Van 1000 1 70 50 20 3 Krixi 1000 2 70 50 20 2 Maloch 1000 1 70 50 20	warning: value cannot be negative full team valid hero selection balanced team full team true valid hero selection true balanced team true
5 1 Van 1000 1 70 50 20 2 Zuka 800 1 90 40 40 2 Maloch 1000 1 70 50 20 6 Murad 1000 1 70 50 20 1 Violet 1000 2 70 50 20	full team valid hero selection team needs mage tank full team true valid hero selection true team needs mage tank false
5 2 Maloch 1000 1 70 50 20 3 Zill 800 1 90 40 40 4 Taara 1000 1 70 50 20 4 Thane 1000 1 70 50 20 5 Alice 1000 1 70 50 20	full team valid hero selection team needs carry full team true valid hero selection true team needs carry false
3 1 Van 100 1 70 50 20 4 Taara 100 1 70 50 20 5 Alice 100 2 70 50 20	member is missing member is missing false false false
6 2 Zuka 1000 1 70 50 20 4 Taara 1000 1 -70 50 20 5 Alice 1000 2 70 50 20 2 Maloch 1000 1 70 50 20 3 Zill 800 1 90 40 40 1 Violet 1000 1 70 50 20	warning: value cannot be negative too many members too many members false false false

คำแนะนำ การตรวจสอบว่าวัตถุเป็นชนิดเดียวกันกับที่สนใจหรือไม่ ทำได้โดยการใช้คำสำคัญ `instanceof` เช่น ถ้าต้องการตรวจสอบว่า `m` แท้จริงมีชนิดเป็น `Carry` หรือไม่

เราจะเขียนว่า `if(m instanceof Carry)` ซึ่งจุดนี้อาจจะดูแปลกตาไปบ้าง แต่เป็นฟีทเจอร์ในภาษาคอมพิวเตอร์ที่สำคัญอันหนึ่ง ปัจจุบันมีแทบทุกภาษา

เงื่อนไขการให้คะแนน: โปรแกรมจะต้องทำงานถูกต้องอย่างน้อย **50%** ของชุดข้อมูลทดสอบจึงจะได้คะแนน