Lab 5 เขียนส่วนควบคุมตัวรถในเกม CodeRally

เริ่มโดย เอาไฟล์จาก

CodeRally [*from* *http://www.alphaworks.ibm.com/tech/coderally]***

หรือโหลดไฟล์จากลิ้งค์นี้ unzip jar ทั้งหมดไปไว้ในplugins folder ของ eclipse

CodeRally คือการเขียนโปรแกรมบังคับรถแข่ง แข่งกัน นิสิตจะต้องเขียนคลาสและเมธอดของจาวา เพื่อทำ AI ของรถ พอเราเขียน AI เสร็จก็เอาไปทดลองในสนามแข่ง แข่งกับเครื่องหรือเพื่อนๆ

รถของเรา (คลาสจะเรียกว่า Rally car) สามารถรับรู้ ตำแหน่งของสิ่งต่างๆในสนามแข่งได้ รวมถึงสามารถรับรู้สภาพของรถคันอื่นได้ ด้วย รถนั้นชนกันได้ และ ยังโยนยางสำรอง (เก็บได้ ในฉาก) เพื่อทำให้รถคันอื่นเสียได้ ด้วย นอกจากนี้ยังเปลี่ยนเป็นรถตำรวจ (เรียกอีกอย่างว่า เข้าprotected mode) ซึ่งเป็นอมตะชั่วคราวได้ อีกด้วย

ในการแข่งนั้น หนึ่งแมทช์จะมีรถแข่งได้ หกคัน ซึ่งแต่ละคันจะสุ่มโผล่และสุ่มหันหน้าที่ตำแหน่งต่างๆกันไป แต่ละคันจะมีน้ำมันและยางสำรองเท่ากัน การขับเคลื่อนต้องเสียน้ำมัน ในสนามแข่งจะมีที่เติมน้ำมันอยู่ ถ้าเอารถไปจอดทับได้ ก็จะได้ เติมน้ำมัน นอกจากนี้ยังมีที่เก็บยางอีกด้วย ถ้าน้ำมันหมดรถจะเคลื่อนไปไหนไม่ได้

ในการคิดคะแนน วิธีคิดของคนที่ทำCodeRally เป็นดังนี้

- ขว้างยางถูกรถอีกคัน
- วิ่งผ่านธง (check point)
- น้ำมันที่เหลือในตอนแข่งจบ (การแข่งใช้เวลาประมาณสองนาที)

ซึ่งรถที่ได้ คะแนนมากที่สุดจะถือว่าชนะ

เอาล่ะ มาเริ่มเล่นกันดีกว่า

1. ไปที่ File > New > Project
2. เลือก Other > Game Project คลิ้กที่Next ตามรูป Figure 1.

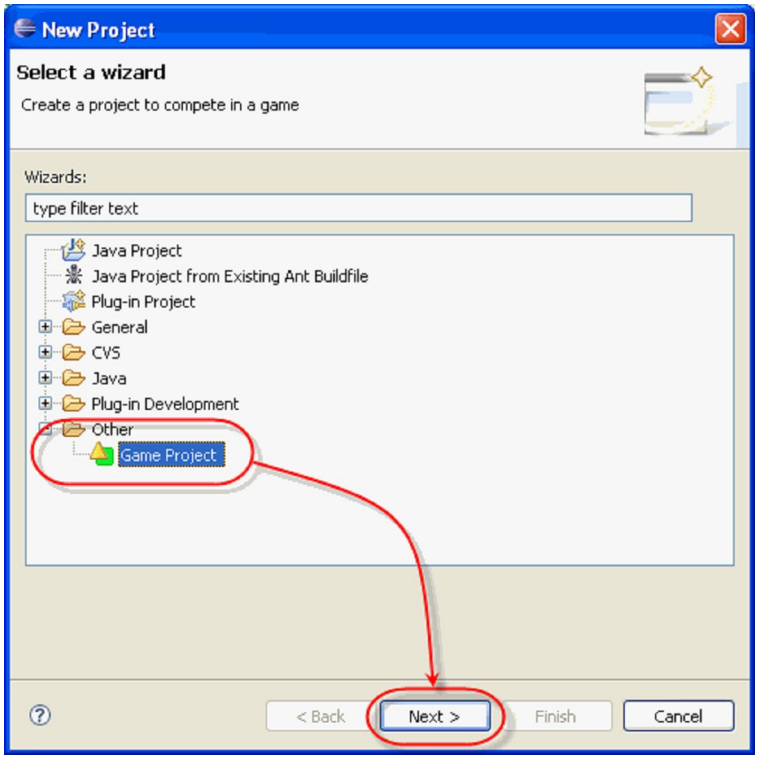

Figure 1 Create new Game Project

3. เลือกให้ เกมเป็นCodeRally ใส่ ID นิสิ ตลงตรงproject name จากนั้นคลิ ก Finish ตามรูป Figure 2.

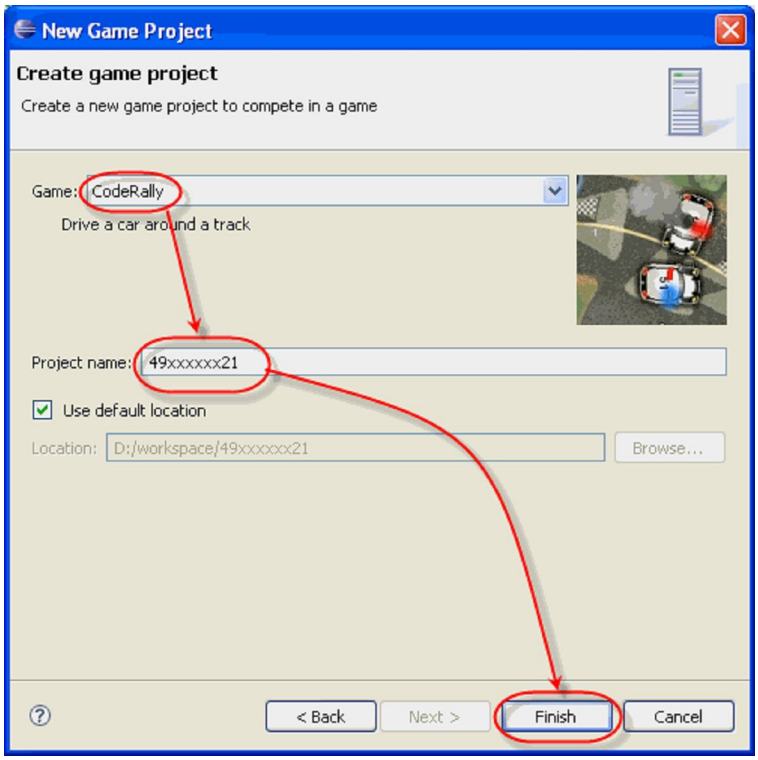

Figure 2 New Game Project name

4. โครงสร้ างโปรเจ็คจะเป็นดังFigure 3.

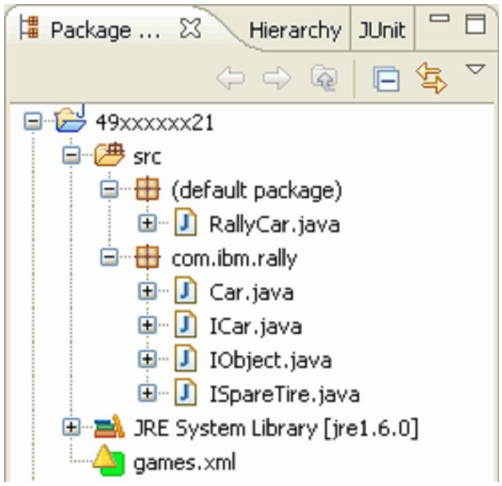

Figure 3 Your game project

5. เลือก RallyCar.java ไฟล์นี้จะเป็นไฟล์ที่เราต้ องแก้ ไฟล์เดียวเท่านั้น ในเมธอดgetColor() ให้ เปลี่ยนสี จาก *CAR_BLUE* เป็นสี ใดก็ได้ ที่นิยามไว้ ในCar.java (อันนี้ทำเพื่อให้ เราได้ ใส่ สี ที่ชอบเฉยๆ ไ มีอะไรพิ เศษ)

6. ลองรันดูเลย วิ ธีการคือ เลือกgames.xml จากนั้นใส่ ชื่อและสถาบัน แล้ วคลิ ก Add My Team จาก นั้นก็เลือกผู้ เล่นที่บังคับด้ วยคอมอื่นๆ แล้ วAdd ไปด้ วย พอเสร็จแล้ วให้ รันด้ วยการคลิ ก Run ดัง Figure 4.

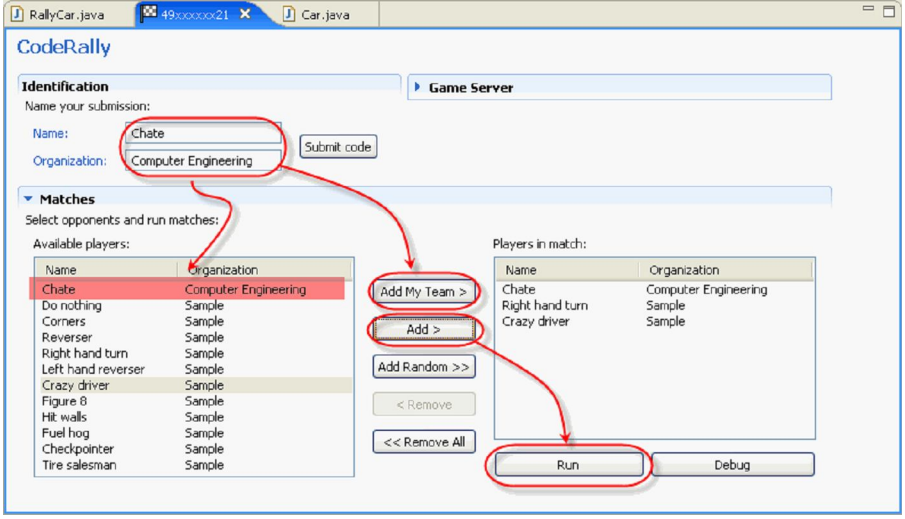

Figure 4 Setup your car to the race

7. สภาพการแข่งจะเป็นดังรูป Figure 5

8. ถ้าจะเอาไฟล์เอกสารของเมธอดต่างๆ และคลาสต่างๆ

   8.1. คลิ้กชื่อโปรเจ็คแล้วเลือกเมนู **File** > **Export**

   8.2. เลือก **Java** > **Javadoc**

   8.3. คลิ้ก **Next** แล้วก็**Finish**. (อาจต้องมีการหา javadoc.exe ให้เจอ ซึ่งปกติจะอยู่ในโฟลเดอของ jdk)

9. เท่านี้ก็จะได้โฟลเดอร์doc ออกมา ซึ่งเราจะดูคำอธิบายของทุกคลาสได้ (คำอธิบายเมธอดและคลาส เราจะเรียกว่า javadoc)

10. สิ่งที่เราต้องเขียนก็คือ เมธอดmove() ใน RallyCar.java

โดยเราต้องเขียนให้

10.1 ให้รถวิ่งตามเช็คพอยต์

10.2 ให้รถหยุดเติมน้ำมันได้

10.3 เขียนพฤติกรรมอื่นๆเพิ่มได้ตามใจ แต่ต้องไม่ขัดกับพฤติกรรมสองอย่างข้างต้น แปลงร่างเป็นรถตำรวจเมื่อชน แล้วไล่ชนคนอื่นจนกว่าจะกลับเป็นรถธรรมดา หรือ แปลงเป็นรถตำรวจ พอชนแล้วถอยเข้าชนอีกเพื่อซ้ำให้ตายจนกว่าจะกลับเป็นรถธรรมดา เป็นต้น

10.4 เขียนคอมเม้นกำกับ เมธอด move ด้วย ว่ารถของเรามีพฤติกรรมอย่างไร

การตั้งชื่อproject และการตั้งชื่อไฟล์รวมถึงการขึ้นต้นไฟล์ใช้เกณฑ์เดียวกับแล็บที่แล้ว

ไฟล์ที่ส่งให้เป็นjar file ที่มีซอร์สโค้ดอยู่ภายในตั้งชื่อด้วยเกณฑ์เดียวกับแล็บที่แล้ว

subject ของอีเมล์ที่ส่งให้ใช้เกณฑ์ของแล็บที่แล้วเช่นเดียวกัน

กำหนดส่งคือวันอาทิตย์ภายในเวลา 0.00 (เวลาเที่ยงคืนของคืนวันเสาร์)

ขอบอกว่า อย่าประมาท ให้อ่านเดี๋ยวนี้เพราะมีเนื้อหาให้อ่านเยอะไม่งั้นกว่าจะได้เริ่มจะทำไม่ทัน

ต่อไปนี้เป็นคำแนะนำ อ่านไว้จะได้ช่วยในการเขียนโค้ด

Coding Your Car – Overview

RallyCar.java นั้นมีโครงอยู่แล้ว เราสามารถเติมตัวแปรและเมธอดเพิ่มเติมในคลาสนี้ได้ ตามใจ สร้างคลื่นส
ก็ยังได้

พอต้องการจะทดสอบ อย่าลืมเซฟซะก่อน แล้วเลือกgames.xml แล้วรัน

คลาส Car.java เป็นซุปเปอร์คลาสของ RallyCar จะมีเมธอดหลายๆอย่างที่ RallyCar รับสืบทอดไป จงใช้ เมธ
อดเหล่านี้ให้ เป็นประโยชน์

ห้ามแก้ คลาสCar จริงๆแล้ว เวลารันCodeRally รันคนละ Car กับที่เรามี ที่เราได้ มาเป็นแค่ตัวโครงเฉยๆ

นอกจากนี้ยังมี อินเตอร์เฟสอีกสามอัน (โปรดไปอ่านเรื่องInterface และ abstract class เพื่อความเข้าใจตรง
นี้ จะช่วยได้ มาก)

IObject.java

เป็นอินเตอร์เฟสของทุกออบเจ็กต์ในสนามแข่ง ทุกออบเจ็กต์จะimplement อินเตอร์เฟสนี้ (ไป
อ่านว่า การ implement คืออะไร) อินเตอร์เฟสนี้นิยามเมธอด

getX() และ getY() ซึ่งรีเทิร์นตำแหน่งของออบเจ็กต์ในสนามแข่ง ซึ่งเลขของx กับ y นั้นไม่มีค่า
ลบ และเป็นไทป์ double

ISpareTire.java

อินเตอร์เฟสนี้ extend มาจาก IObject ซึ่งอินเตอร์เฟสนี้นิยามยางสำรองที่อยู่ในสนามแข่ง ยาง
ทุกเส้นจะimplement อินเตอร์เฟสนี้ อินเตอร์เฟสนี้มีเมธอดgetHeading() และ getSpeed() ดัง
นั้นเราสามารถรู้ข้อมูลของยางระหว่างที่ถูกขว้างไปได้

ICar.java

อินเตอร์เฟสนี้ extend มาจาก IObject ซึ่งอินเตอร์เฟสนี้นิยามรถที่อยู่ในสนามแข่ง นิยามเมธอดที่
รถสามารถเรียกใช้ ได้ (ทั้งเรียกของตัวมันเอง และเรียกของรถคันอื่น) เมธอดเหล่านี้ได้ ถูกอธิบาย
ไว้ข้างล่างและในJavaDocs

The CodeRally Track Simulation

Identification

มีโครงเมธอดใน RallyCar เพื่อไว้ สำหรับให้ รถเราดูต่างจากคันอื่น เมธอดนี้ก็คือgetColor() ซึ่งต้องรีเทิร์น
byte constant ที่เลือกมาจากสีรถที่นิยามไว้ ในคลาสCar เราใช้ เมธอดนี้ในการให้ สีรถของเรา ค่าdefault
ของ getColor() นั้นคือ CAR_BLUE

เมธอดนี้ต้องไม่ทำอะไรนอกจากรีเทิร์นค่าคงที่

Initialization

ในตอนที่รถเราถูกวางในสนามแข่ง ตัวซิมูเลเตอร์จะเรียกเมธอดinitialize() ของรถเรา ดังนั้นให้ เราใส่โค้ด
สำหรับการเซ็ตค่าเริ่มต้นต่างๆของรถเราได้ ที่เมธอดนี้ เราสามารถเรียกใช้ เมธอดอะไรก็ได้ว่า ซิมูเลเตอร์
จะรัน initialize() เป็นเวลาหนึ่งวินาทีเท่านั้น ดังนั้นถ้ารันเมธอดinitialize() ไม่เสร็จ รถของเราอาจจะเข้ามาใน
สนามในสภาพที่ค่าต่างๆเละได้

Moving Your Car

หลังจากซิมูเลเตอร์เรียกใช่ initialize() เสร็จไปแล้ว มันจะเรียกเมธอดmove() ของรถแต่ละคันเรียงลำดับกันไป ซึ่งการเรียกนี้เกิดทุกclock tick เราสามารถเรียกเมธอดต่างๆเพื่อดูสภาพรถ เปลี่ยนตัวแปร ดูสถานะรถคันอื่น หาตำแหน่งของต่างๆในสนามแข่ง(เช่นหาปั๊มน้ำมัน) และ โยนยางสำรองออกไป

move() นั้นมีพารามิเตอร์สี่ตัว ซึ่งเป็นการบอกค่าต่างๆจากcycle การทำงานรอบที่แล้ว ซึ่งพารามิเตอร์เหล่านี้บอกถึง:

- การใช้ เวลาของเมธอดmove() (หน่วยเป็น milliseconds) ในรอบการทำงานที่แล้ว
- รอบที่แล้วชนขอบสนามหรือเปล่า
- รอบที่แล้ว เราชนกับรถคันอื่นหรือเปล่า
- รอบที่แล้ว เราโดนยางขว้างโดนหรือเปล่า

พารามิเตอร์ตัวแรกเป็น int ส่วนตัวที่สองเป็นboolean ตัวที่สามและสี่นั้นเป็นICar (หรือเป็น null ถ้าไม่ได้ ชนหรือโดนปา) พารามิเตอร์ตัวแรกจะใช้ บอกได้ว่า โค้ดเรายาวเกินไป นานเกินไปหรือไม่ เพราะ ซิมูเลเตอร์มีเวลาให้ move() จำกัดเหมือนกัน

CodeRally Track Details

สนามแข่งนั้นเป็นสองมิติ แกนx ยาว 1010 หน่วย ส่วนแกนy ยาว 580 หน่วย จุด (0,0) อยู่ซ้ายบนของจอ สนามมีผนังล้อมรอบ ซึ่งรถออกนอกผนังนี้ไม่ได้ ไม่มีผนังอยู่ข้างในสนามแต่อย่างใด รถสามารถเคลื่อนที่ไปไหนก็ได้ ในสนาม ยกเว้นแต่ว่าชน หรือ ติดรถคันอื่น ออบเจ็กต์เคลื่อนที่ในทิศทางที่เรียกว่าheadings ซึ่งวัดเป็นจำนวนเต็ม หน่วยเป็นองศา (0 องศา หมายถึง ตรงขึ้นข้างบน) องศานั้นมีค่าตั้งแต่0 ถึง 359 ค่าขององศาเพิ่มตามเข็มนาฬิกา

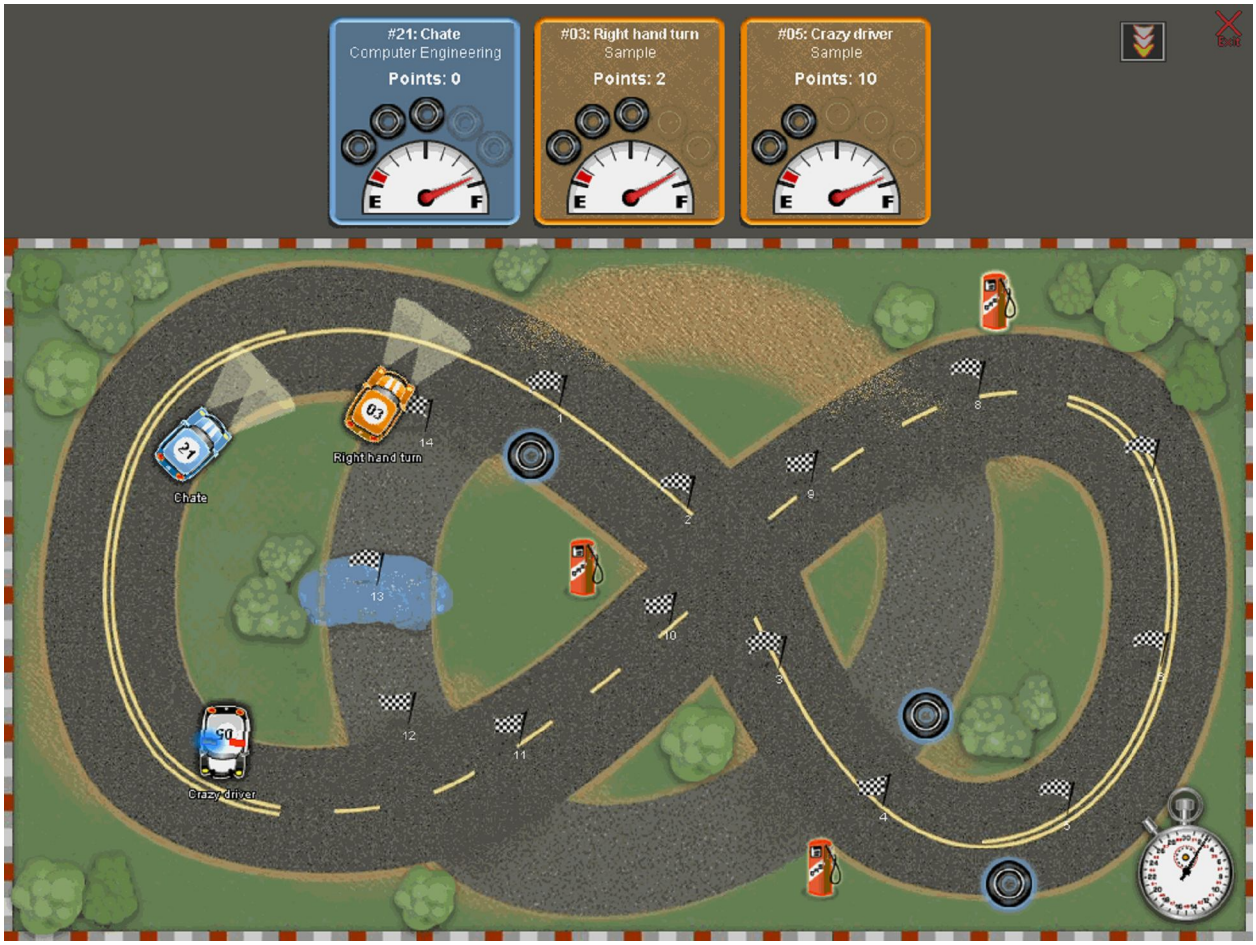

Figure 5 Simulated track world

ในโลกของเกมนั้นมีสิ่งต่างๆดังต่อไปนี้

มีนาฬิกา ticking clock ซึ่งค่าของมันอ่านได้ด้วย getClockTicks()

ในตอนเริ่มแข่งแต่ละนัด รถแต่ละคันจะมีน้ำมัน100 หน่วยและยางเอาไว้ขว้าง3 เส้น

เมื่อเราเซ็ตพวงมาลัยและคันเร่ง รถจะเคลื่อนตามนั้นจนกว่าจะมีคำสั่งใหม่ (แม้ว่าจะเกิดการชนก็ตาม จะพยายามเคลื่อนที่ตามคำสั่งเดิมอยู่ดี นอกจากเราเขียนแก้)

พวงมาลัยและคันเร่งเปลี่ยนค่าได้ทันที แต่ว่าความเร็วและทิศทางจะไม่เปลี่ยนในทันทีเพราะนั้นมีความเฉื่อย

ค่าคันเร่งน้อยสุด(MIN_THROTTLE) คือ -50 และค่าคันเร่งมากสุด(MAX_THROTTLE) คือ 100 อัตราการเปลี่ยนความเร็ว(นับจากสภาพรถหยุด) คือ 8 หน่วยต่อ tick (ยกเว้นกรณีที่เกิดการชนกัน)

ค่าหักพวงมาลัยน้อยที่สุด(MAX_STEER_LEFT) คือ -10 และค่าหักพวงมาลัยมากสุด (MAX_STEER_RIGHT) คือ 10 อัตราการเปลี่ยนทิศนั้นขึ้นกับความเร็วของรถ เราสามารถหาอัตรานี้ได้โดยใช้getChangeInHeading()

ตำแหน่งของรถ บนสนามแข่งนั้น เป็นจุด รถมีความยาว60 หน่วย และกว้าง40 หน่วย โดยตำแหน่งคือจุดกลางรถนั้นเอง

เราถือว่า ยางที่ขว้างออกไป ชนรถ เมื่อตำแหน่งของมันอยู่ใกล้จุดกลางรถ40 หน่วย หรือน้อยกว่า

ยางที่ขว้างนั้น มีความเร็วคงที่ เท่ากับ12 หน่วย/tick เมื่อชนกับรถหรือผนังแล้ว ยางจะหายไป

จุดเช็คพอยต์ ที่เติมน้ำมัน และที่เติมยาง ไม่มีผลต่อยางที่ขว้างออกมา ยางที่ขว้างออกมาไม่มีวันชนกัน

รถมีน้ำมันได้มากที่สุด100 หน่วย

รถมียางไว้ขว้างได้มากสุด5 เส้น

เมื่อตำแหน่งรถอยู่ใกล้ที่เติมน้ำมันเป็นระยะ25 หน่วยหรือใกล้กว่า น้ำมันจะเพิ่มด้วยอัตรา1 หน่วยต่อ clock tick จนเต็มถัง (ถ้ารถออกจากระยะไปแล้ว น้ำมันก็ไม่เพิ่ม) ในสนามแข่งจะมีที่เติมน้ำมันอยู่สามที่ ตำแหน่งจะเป็นการสุ่ม

เมื่อรถอยู่ใกล้ที่เติมยางเป็นระยะ25 หน่วยหรือใกล้กว่า (และรถมียางสำหรับขว้างน้อยกว่า5 เส้น) รถจะเก็บยางด้วยอัตรา1 เส้นต่อ5 clock ticks ในสนามแข่งจะมีที่เติมยางอยู่3 ที่ ตำแหน่งของที่เติมยางจะเป็นการสุ่ม

รถสามารถป้องกันตัวเองได้ด้วยการเข้า"protect mode" แปลงเป็นรถตำรวจ ซึ่งรถที่เข้าโหมดนี้นั้น จะกินน้ำมันสองเท่าของทั่วไป โหมดนี้นั้นจะคงอยู่50 clock tick

ถ้ารถเราชนกับรถอีกคัน โมเม้นตั้มจะถ่ายถึงกัน รถที่ชนจะเสียน้ำมัน10 หน่วย (ยกเว้นถ้ารถนั้นอยู่ใน protect mode)

ถ้าเรายิงยางชนคันอื่น เราจะได้10 คะแนน ส่วนรถที่โดนยางจะน้ำมันลด10 หน่วย เมธอด move() จะไม่ถูกเรียกไป10 clock tick และรถจะถูกแรงยางพาให้เคลื่อนไปนิดหน่อย เราจะไม่ได้คะแนนถ้ารถที่โดนเราขว้างอยู่ในprotect mode รถที่อยู่ใน protect mode ก็จะไม่เป็นไร ไม่โดนแรงยางด้วย

เมื่อขว้างยางไปแล้ว รถเราจะไม่สามารถโยนยางได้อีก25 clock tick

move() หนึ่งนั้น จำกัดเวลาได้แค่500 มิลลิวินาที ถ้าในช่วงเวลานี้ยังทำmove() ไม่เสร็จ เมธอด move() จะไม่ถูกเรียกอีกต่อไปในการแข่งนั้น รถคันนั้นจะคงค่าต่างๆที่ตั้งไว้ล่าสุด

สำหรับจุดเช็คพอยต์นั้น รถเราจะได้ 2 คะแนนถ้าผ่านในระยะ 25 หน่วยจากเช็คพอยต์นั้น และถ้าเราไปยังเช็คพอยต์ต่อไป (ที่เลขต่อกัน) รถเราจะได้ 6 คะแนน ถ้ารถเราผ่านเช็คพอยต์เดิมสองทีจะไม่ได้ คะแนน

ตารางคะแนนเป็นดังนี้ คะแนนในการแข่งนั้นคิดจากคะแนนรวม

Action	Points Earned
ผ่านเช็คพอยต์	2
ผ่านเช็คพอยต์เรียงลำดับเบอร์	6
ยิงยางโดนรถคันอื่น	10
สำหรับทุกๆ 10 หน่วยของน้ำมันที่เหลือ หลังจากแข่งเสร็จแล้ว	1

General Information, Caveats, Constraints, and Restrictions

Car ไม่สามารถมี constructor ได้

Car ห้ามใช้ static initialization block ในการ initialize

โค้ดทั้งหมดต้องอยู่ในคลาส RallyCar

Car ห้ามสร้าง thread เอง ห้ามปริ๊น ห้ามสร้างไฟล์ ห้ามใช้ system function

Car ใช้ System.out.println() เพื่อพิมพ์ลงEclipse console ได้ แต่ว่าเวลาที่ใช้ ก็นับเป็นส่วนหนึ่งของเวลา move() ด้วย อย่าลืมว่าฟังก์ชั่นพวกนี้ใช้ เวลานาน

ห้ามสร้างโค้ดที่จะทำลายระบบการแข่ง

Example RallyCar Code

ข้างล่างเป็นตัวอย่างโค้ดของ move() เป็นโค้ดจากหลายๆส่วนนะ ไม่ใช่ move() อันเดียว โปรดสังเกตว่าโค้ดเหล่านี้เป็นตัวอย่างเท่านั้น เราสามารถเขียนโค้ดให้ดีกว่านี้อีกมากมายนัก

```
/**
* Go toward the first spare tire depot.
*/
public void move(int lastMoveTime, boolean hitWall,
```

```
    ICar collidedWithCar, ICar hitBySpareTire) {
    // pick a spare tire depot
    IObject st = getSpareTireDepot()[0];
    // go toward the checkpoint
    int h = getHeadingTo(st);
    if (getHeading() > h)
        setSteeringSetting(MAX_STEER_LEFT);
    else
        setSteeringSetting(MAX_STEER_RIGHT);
    setThrottle(MAX_THROTTLE);
}
/**
* Put the car in reverse for a few moves if you collide with another car.
*/
protected int wait;
public void move(int lastMoveTime, boolean hitWall,
    ICar collidedWithCar, ICar hitBySpareTire) {
    if (collidedWithCar != null)
        wait = 10;
    if (wait > 0)
        setThrottle(MIN_THROTTLE);
    else
        setThrottle(MAX_THROTTLE);
    if (wait > 0)
        wait--;
}
```