Manual คู่มือการใช้งาน

1 ขั้นตอนแรก หลังจากที่มี ไฟล์ Graph.java แล้ว ให้สร้างไฟล์สำหรับ main ขึ้นมาเพื่อทำการเรียกใช้งาน class Graph (หรือใช้ไฟล์ main ที่มีให้ในไฟล์ rar)

```
public class runGraph {
    Run | Debug
    public static void main(String[] args) {

    }
}
```

2 หากเราสร้าง class main ด้วยตัวเอง ให้ทำการ import java.util.Scanner; และสร้าง Object => Scanner ชื่อตัวแปร = new Scanner(System.in); เพื่อนำมาใช้รับค่าผ่านทางแป้นพิมพ์ หรือ ใช้วิธีอื่นสำหรับกรณีที่ไม่ต้องการรับค่าผ่านทางแป้นพิมพ์ เช่น สร้างตัวแปร มาเก็บ ข้อมูลที่ต้องการใช้

ตัวอย่าง

```
import java.util.Scanner;

public class runGraph {
    Run | Debug
    public static void main(String[] args) {

        Scanner sn = new Scanner(System.in);
        String all = sn.nextLine();
        sn.close();

    }
}
```

สร้างตัวแปรประเภท String ชื่อ all มารับค่าผ่านทางแป้นพิมพ์

3 วิธีการเรียกใช้งานตัว class Graph ให้สร้างObject จาก Graph โดยให้พารามิเตอร์ เป็น String ที่เรารับค่าเข้ามา หลังจากนั้นนำมาเรียกใช้ method checkGraph

```
import java.util.Scanner;

public class runGraph {
    Run | Debug
    public static void main(String[] args) {

        Scanner sn = new Scanner(System.in);
        String all = sn.nextLine();
        sn.close();

        Graph G = new Graph(all);
        System.out.println(G.checkGraph());
    }
}
```

ใช้คำสั่ง System.out.print(); เพื่อแสดงคำตอบที่ได้จาก method

วิธีการทำงานของGraph.java

class Graph ประกอบไปด้วย attribute constructor และ method ดังนี้

Attribute

- private String[] graphWalk;
- private int repeatedEdge = 0;
- private int repeatedVertex = 0;
- private int startendSamepoint = 0;

repeatedEdge สำหรับตรวจสอบ การซ้ำเส้น
repeatedVertex สำหรับตรวจสอบการซ้ำจุด
startendSamepoint สำหรับตรวจสอบว่าเริ่มและสิ้นสุดที่จุดเดียวกัน

กำหนดให้ตัวแปรตรวจสอบทั้งสามตัวเป็น 0 โดยให้ 0 หมายถึง false และ 1 เป็น true เพื่อนำไปเทียบตรวจสอบข้อมูลที่อ้างอิงจากตาราง

```java
public class Graph {

    private String[] graphWalk;
    private int repeatedEdge = 0;
    private int repeatedVertex = 0;
    private int startendSamepoint = 0;
    // 0 = false 1 = true
```

	Repeated Edge?	Repeated Vertex?	Starts and Ends at Same Point?
Walk	allowed	allowed	allowed
Trail	no	allowed	allowed
Path	no	no	no
Closed walk	allowed	allowed	yes
Circuit	no	allowed	yes
Simple circuit	no	first and last only	yes

Constructor

```java
    Graph(String all){

        this.graphWalk = all.split(" ");


    }
```

รับค่าตัวแปร String จาก main เข้ามาด้วยชื่อ all จากนั้นทำการ split เพื่อเก็บข้อมูลเข้า Array ชื่อ graphWalk ที่เป็น attribute ของ class โดยแบ่งจากช่องว่าง

Method

- public boolean Trail(int repeatedEdge, int repeatedVertex, int startendSamepoint)

Method Trail ใช้สำหรับตรวจสอบเงื่อนไขที่ว่า Trail จะต้องไม่มีการทับเส้น

- public boolean circuit(int repeatedEdge, int repeatedVertex, int startendSamepoint)

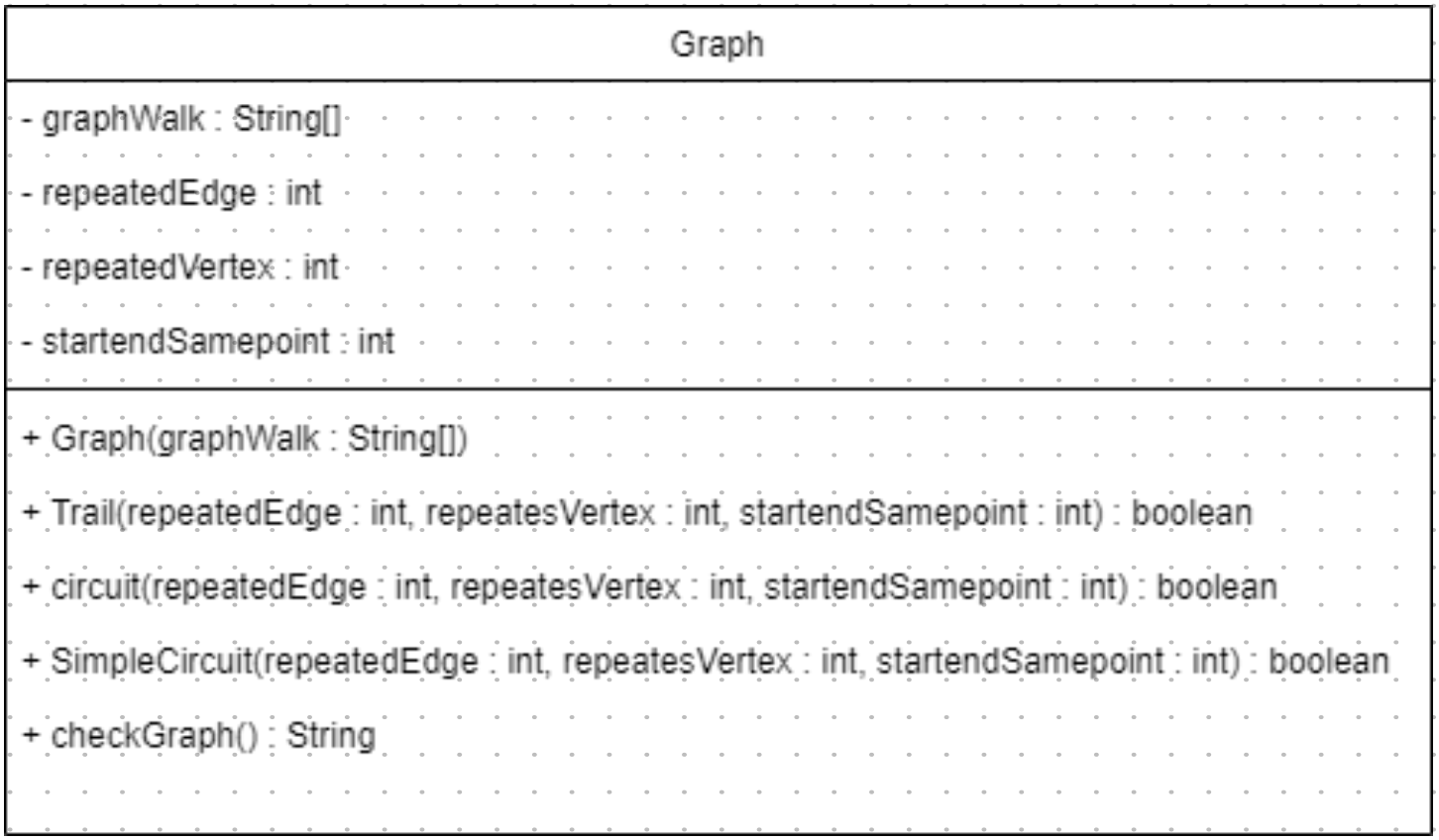
Method circuit ใช้สำหรับตรวจสอบเงื่อนไขที่ว่า circuit จะต้องไม่มีการทับเส้น และจะต้องมีจุดเริ่มต้นและจุดจบเป็นจุดเดียวกัน

- public boolean simpleCircuit(int repeatedEdge, int repeatedVertex, int startendSamepoint)

Method simpleCircuit ใช้สำหรับตรวจสอบเงื่อนไขที่ว่า simpleCircuit จะต้องไม่มีการทับเส้น และจะต้องมีจุดเริ่มต้นและจุดจบเป็นจุดเดียวกัน จะต้องไม่มีการทับเส้น จะต้องไม่มีการทับจุดนอกจากจุดเริ่มและจุดจบ

- public String checkGraph()

Method checkGraph ใช้สำหรับหาว่าข้อมูล graph ที่รับเข้ามาเป็นรูปแบบใด



หลักการทำงานของMedthod checkGraph

```
53      public String checkGraph(){
54
55          int sumOf = 0;
56          List<String> list = Arrays.asList(graphWalk);
57          if(graphWalk[0].equals(graphWalk[graphWalk.length-1])){
58              repeatedVertex = 1;
59              startendSamepoint = 1;
60          }
61          for(int i = 0; i < graphWalk.length; i++){
62
63              sumOf += Collections.frequency(list, graphWalk[i]);
64              if(Collections.frequency(list, graphWalk[i]) > 1){
65
66                  if( i % 2 == 0 ){
67                      if(i == 0 ) repeatedVertex = 4;
68                      else if(i != graphWalk.length-1) repeatedVertex = 1;
69                  }
70                  else if(i % 2 != 0 ) repeatedEdge = 1;
71
72              }
73
74          }
75
76          if(sumOf == graphWalk.length) return "Path";
77          if(simpleCircuit(repeatedEdge, repeatedVertex, startendSamepoint)) return "Simple Circuit";
78          if(circuit(repeatedEdge, repeatedVertex, startendSamepoint)) return "Circuit";
79          if(Trail(repeatedEdge, repeatedVertex, startendSamepoint)) return "Trail";
80
81          return "Walk";
82
83      }
84
85  }
```

- ตัวแปร sumOf สร้างมาเพื่อตรวจสอบว่าเส้นทาง Graph ที่รับเข้ามา มีเส้นหรือจุดซ้ำหรือไม่ โดยใช้คำสั่งของ List => Collections.frequency() วน loopนับว่าในจุดหรือเส้นนั้นมีตัวที่ซ้ำกันมากกว่า 1 ไหม เนื่องจาก 1 คือนับตัวมันเอง หลังจบ loop sumOf ที่มีค่าเท่ากับจำนวนของ Array ที่เก็บข้อมูล ถือว่าไม่มีตัวซ้ำเลย จึง คืนค่า เป็น "Path"

- เงื่อนไขแรกใน Method ที่เราจะเจอคือ ให้ค่าเส้นทาง Graph ที่มีจุดแรกเป็นตัวเดียวกันกับจุดสุดท้าย มีการซ้ำจุดจริง และ มีจุดเริ่มต้นและสิ้นสุดที่เดียวกันจริง

- เพื่อที่จะเก็บข้อมูลไปเช็คตามตารางที่มีเราจำเป็นต้องวน loop เพื่อรวบรวมข้อมูล โดยใน loop จะมีการใช้ตัวแปร sumOfเพื่อตรวจสอบว่าเป็น Path หรือไม่และยังมีการตรวจสอบในกรณีที่มีจุดซ้ำหรือเส้นซ้ำกันมากกว่าหนึ่ง(นับตัวเองด้วย) ตรวจสอบว่าเป็นจุดหรือเส้นจาก index ของ Array เนื่องจาก ข้อมูลที่รับเข้ามาเป็นการสลับระหว่าง จุดและเส้น จึงให้ รอบที่หมุนแล้วเจอตัวซ้ำมากกว่าหนึ่ง มาเป็นตัวตรวจสอบ ถ้าเป็น index เลขคู่ ถือว่าเป็น จุดซ้ำจริง และตรวจสอบว่าเป็นการซ้ำของจุดเริ่มและจุดจบอย่างเดียวหรือไม่ ถ้าเป็นเลขคี่ ถือว่าเป็นเส้นซ้ำกันจริง

- เมื่อออกจาก loop เราจะได้ข้อมูลตามที่ตารางต้องการคือ มีการซ้ำเส้นจริงไหม ซ้ำจุดจริงไหม และจุดเริ่มและจุดสิ้นสุดเป็นจุดเดียวกันจริงไหม จากนั้นเราจะนำเอาข้อมูลทั้งหมดไปเทียบผ่าน method แต่ละ method เรียงลำดับการตรวจสอบข้อมูล โดยอ้างอิงข้อมูลจากตารางโดยจะให้ค่าการเดินทางรูปแบบที่ต้องการข้อมูลที่เป็นจริงหรือเท็จมากกว่า การเดินทางรูปแบบที่ต้องการข้อมูลที่สามารถเป็นได้ทั้งจริงและเท็จ

Input / Output

Input

รับค่าเข้ามาเป็น String 1 บรรทัด โดยที่ต้องเป็นข้อมูลที่จุดและเส้นสลับกันไปมา และจบด้วยจุดเสมอ ชื่อจุดและเส้นจะต้องถูกคั่นด้วยช่องว่าง spacebar 1 ครั้ง

Output

แสดงผลเป็นชื่อรูปแบบของการเดินทางของ Graph โดยตรวจสอบอ้างอิงจากตาราง

ตัวอย่างการแสดงผล

ตัวอย่าง Input	ตัวอย่าง Output
v1 e1 v2 e3 v3 e4 v3 e5 v4	Trail
v1 e1 v2 e3 v3 e7 v6 e8 v2 e2 v1	Circuit
v1 e1 v2 e2 v3 e2 v2 e1 v1	Walk
v1 e1 v2 e2 v1	Simple Circuit

สมาชิกกลุ่ม

64050050	จุติพร	เขมกรรม
64050060	ชลลดา	แซ่ลิ้ม
64050162	พลอยชมพู	ตุลสุข