คำอธิบาย

โปรแกรมนี้เป็นโปรแกรมที่ใช้ในการหา Minimum Spanning Tree (MST) ของกราฟที่มีน้ำหนักบวก โดยใช้วิธีการ **Prim's Algorithm** ในการหา MST

โดยโปรแกรมนี้สามารถทำงานได้เฉพาะกราฟที่เป็น connected และมี edges ไม่น้อยกว่า `vertices - 1` เพื่อให้สามารถสร้าง Minimum Spanning Tree ได้ถูกต้อง

1. การสร้าง Graph

- กำหนดค่าคงที่ `inf` เพื่อแทนค่าไม่จำกัดของ **weight** ระหว่าง vertex
- สร้างโครงสร้างข้อมูลแบบ **Matrix** เพื่อเก็บข้อมูลของ **Graph**

2. ฟังก์ชัน `addEdge()`

- เพิ่มเส้นเชื่อม (edge) ระหว่าง vertex และกำหนดค่า weight ลงใน Matrix เพื่อแสดงความสัมพันธ์ของ vertex ต่างๆ

3. ฟังก์ชัน `prim()`

- ในเมทอด `prim()` โปรแกรมจะหา Minimum Spanning Tree ด้วยขั้นตอนดังนี้:
  - เริ่มต้นที่ vertex 0
  - เลือกเส้นเชื่อมที่มีน้ำหนักน้อยที่สุดที่ยังไม่ได้ถูกเลือกและเชื่อมกับ MST ที่กำลังสร้างอยู่
  - ทำกระบวนการเชื่อมต่อ MST โดยเพิ่มเส้นเชื่อมที่มีน้ำหนักน้อยที่สุดลงใน MST และทำเครื่องหมาย vertex ที่ถูกเลือกว่าเป็นส่วนหนึ่งของ MST
  - ทำกระบวนการนี้จนกว่า MST จะเสร็จสมบูรณ์

4. การรับข้อมูลจากผู้ใช้และการตรวจสอบข้อผิดพลาด

- โปรแกรมรับข้อมูลจากผู้ใช้ เช่น จำนวนของ vertices, edges, และรายละเอียดของแต่ละเส้นเชื่อม และทำการตรวจสอบความถูกต้องของข้อมูลที่ผู้ใช้ให้มา เช่น ขอบเขตของข้อมูลที่ถูกต้อง และน้ำหนักของเส้นเชื่อมที่มีค่าเป็นบวก

5. คำนวณและแสดงผลลัพธ์

- หลังจากได้รับข้อมูลทั้งหมดแล้ว โปรแกรมจะทำการคำนวณ Minimum Cost ของ MST และแสดงผลลัพธ์ออกทางหน้าจอ

Pseudocode

```
ฟังก์ชัน prim(กราฟ, จำนวนของจุด):
    เลือกแล้ว = อาร์เรย์ของบูลีนขนาด จำนวนของจุด ที่เริ่มค่าเป็นเท็จ
    เลือกแล้ว[0] = จริง  // เริ่มต้นที่จุดที่ 0

    น้ำหนักขั้นต่ำ = 0

    สำหรับ i ตั้งแต่ 0 ถึง จำนวนของจุด - 1:
        น้อยสุด = อินฟินิตี
        ดัชนีน้อยสุด = -1

        สำหรับ v ตั้งแต่ 0 ถึง จำนวนของจุด - 1:
            ถ้าไม่เลือกแล้ว[v] และ กราฟ[i][v] < น้อยสุด:
                น้อยสุด = กราฟ[i][v]
                ดัชนีน้อยสุด = v

        ถ้า ดัชนีน้อยสุด ไม่เท่ากับ -1:
            เลือกแล้ว[ดัชนีน้อยสุด] = จริง
            น้ำหนักขั้นต่ำ += น้อยสุด

    คืน น้ำหนักขั้นต่ำ

ฟังก์ชัน main():
    พิมพ์ "===== อัลกอริทึม Prim ====="

    จำนวนของจุด = รับข้อมูลจากผู้ใช้("ป้อนจำนวนของจุด: ")
    จำนวนของเส้น = รับข้อมูลจากผู้ใช้("ป้อนจำนวนของเส้น: ")

    ถ้า จำนวนของจุด < 1 หรือ จำนวนของเส้น < 1:
        พิมพ์ "ข้อมูลที่ป้อนไม่ถูกต้อง."
        กลับไป

    Graph.init(จำนวนของจุด)

    สำหรับ i ตั้งแต่ 0 ถึง จำนวนของเส้น - 1:
        u = รับข้อมูลจากผู้ใช้("ป้อนจุดเริ่มต้นสำหรับเส้น " + (i + 1) + ": ")
        v = รับข้อมูลจากผู้ใช้("ป้อนจุดปลายสำหรับเส้น " + (i + 1) + ": ")
        w = รับข้อมูลจากผู้ใช้("ป้อนน้ำหนักสำหรับเส้น " + (i + 1) + ": ")

        u -= 1
        v -= 1

        ถ้า u < 0 หรือ u >= จำนวนของจุด หรือ v < 0 หรือ v >= จำนวนของจุด:
            พิมพ์ "ป้อนดัชนีของจุดไม่ถูกต้อง."
            กลับไป

        ถ้า w <= 0:
            พิมพ์ "ป้อนน้ำหนักไม่ถูกต้อง. น้ำหนักต้องเป็นจำนวนบวก."
            กลับไป
```

```
        Graph.addEdge(u, v, w)

    ถ้า จำนวนของเส้น < จำนวนของจุด - 1:
        พิมพ์ "กราฟไม่เชื่อมกัน."
        กลับไป

    ค่าน้อยสุด = prim(Graph.matrix, จำนวนของจุด)

    พิมพ์ "ค่าน้อยสุด: " + ค่าน้อยสุด
```

วิธีการใช้งาน

1. เปิด Command Prompt (หรือ Terminal) แล้วเข้าไปยัง Path ที่เก็บไฟล์ `prim.go`
2. ใช้คำสั่ง `go run prim.go` เพื่อทำการ Compile และ Run Program
3. ป้อนข้อมูลตามที่โปรแกรมต้องการ โดยข้อมูลจะต้องป้อนเป็นตัวเลขเท่านั้น
4. หากป้อนข้อมูลครบแล้ว โปรแกรมจะแสดงผลลัพธ์ออกทางหน้าจอ