โครงสร้างคำสั่งควบคุม
structure control

ผู้ช่วยศาสตราจารย์สมเกียรติ ช่อเหมือน (tko@webmail.npru.ac.th)

สาขาวิชาวิศวกรรมซอฟต์แวร์ คณะวิทยาศาสตร์และเทคโนโลยี

บทนำ

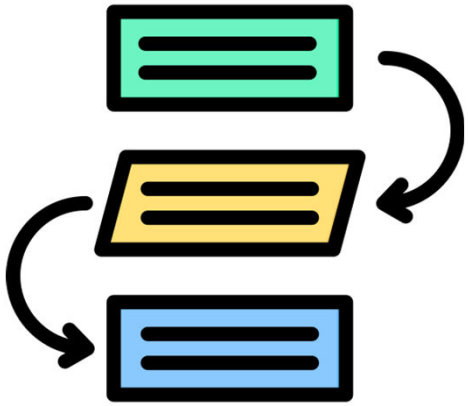
- การทำงานแบบลำดับ
  - ความหมายของการทำงานแบบลำดับ
  - ลักษณะของการทำงานแบบลำดับ
- การทำงานแบบเงื่อนไข
  - if: การตั้งเงื่อนไข
  - switch-case: การจัดการกับเคสต่าง ๆ
- การทำงานแบบวนซ้ำ
  - for: การกำหนดเงื่อนไขและการปรับปรุงค่า
  - while: การตั้งเงื่อนไข
  - do-while: การตั้งเงื่อนไข

การทำงานแบบลำดับ

- การทำงานแบบลำดับ **(Sequential Execution)**
  - กระบวนการทำงานโดยลำดับ จากบรรทัดแรกไปยังบรรทัดสุดท้าย
  - โดยไม่มีการกระโดดข้ามหรือวนซ้ำ ในทุก ๆ ขั้นตอน
  - โปรแกรมทำงานตามคำสั่งที่กำหนด
  - แล้วเลื่อนไปทำงานในคำสั่งถัดไป
- ลักษณะของการทำงานแบบลำดับ

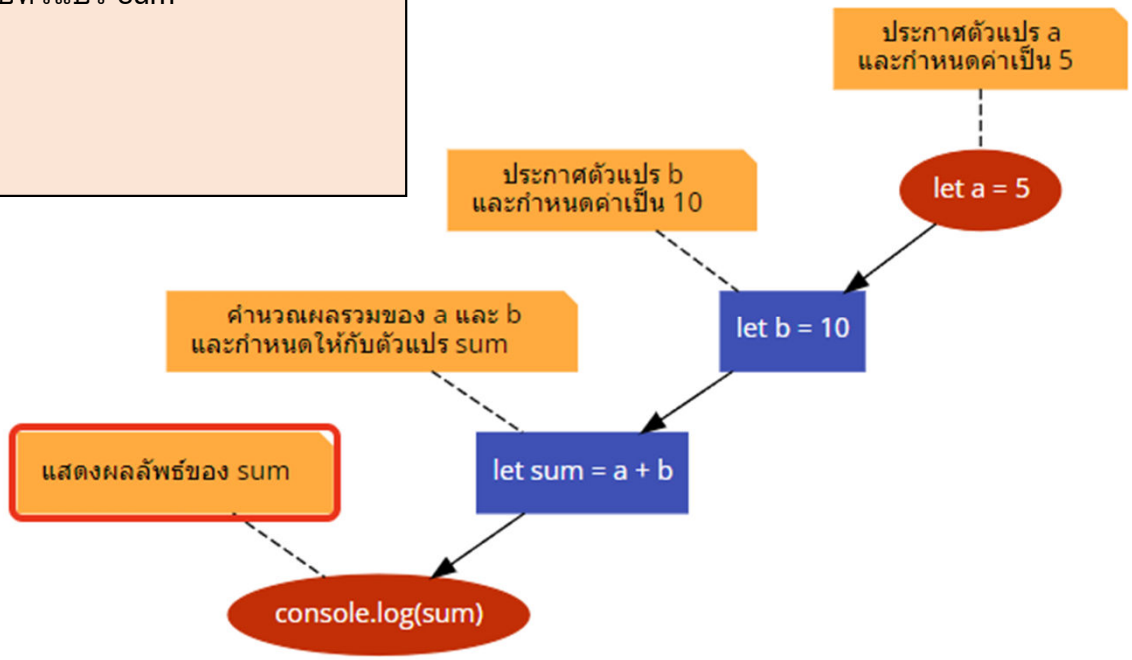
  ลำดับ: คำสั่งทำงานตามลำดับที่กำหนด จากบนลงล่าง

  ไม่มีการกระโดด: ไม่มีการข้ามคำสั่งหรือวนซ้ำ



การทำงานแบบลำดับ

```
let a = 5;  // ประกาศตัวแปร a และกำหนดค่าเป็น 5
let b = 10;  // ประกาศตัวแปร b และกำหนดค่าเป็น 10
let sum = a + b;  // คำนวณผลรวมของ a และ b และกำหนดให้กับตัวแปร sum
console.log(sum);  // แสดงผลลัพธ์ของ sum
```



การทำงานแบบเงื่อนไข

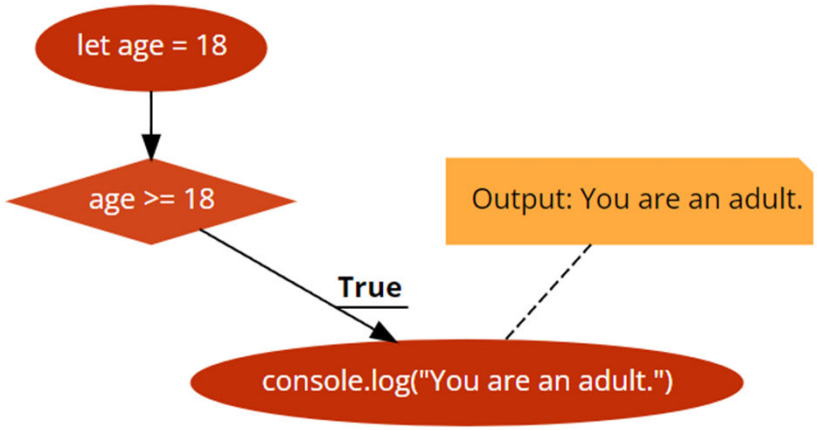
if

- การใช้งาน

  การตรวจสอบเงื่อนไขถ้าเงื่อนไขเป็นจริง **(true)**

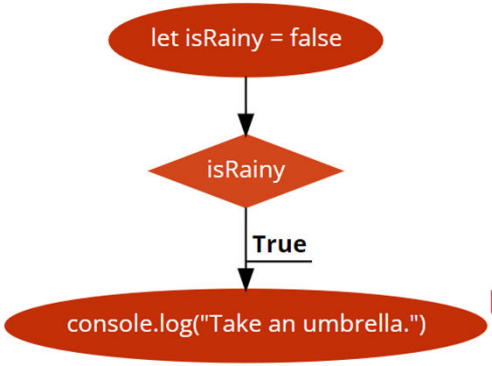
- การตั้งเงื่อนไข

  การเปรียบเทียบหรือการประเมินค่า




```
let age = 18;
if (age >= 18) {
    console.log("You are an adult.");  // Output: You are
an adult.
}
```

```
let isRainy = false;
if (isRainy) {
    console.log("Take an umbrella.");
}
```



การทำงานแบบเงื่อนไข **if else**

- การใช้งาน

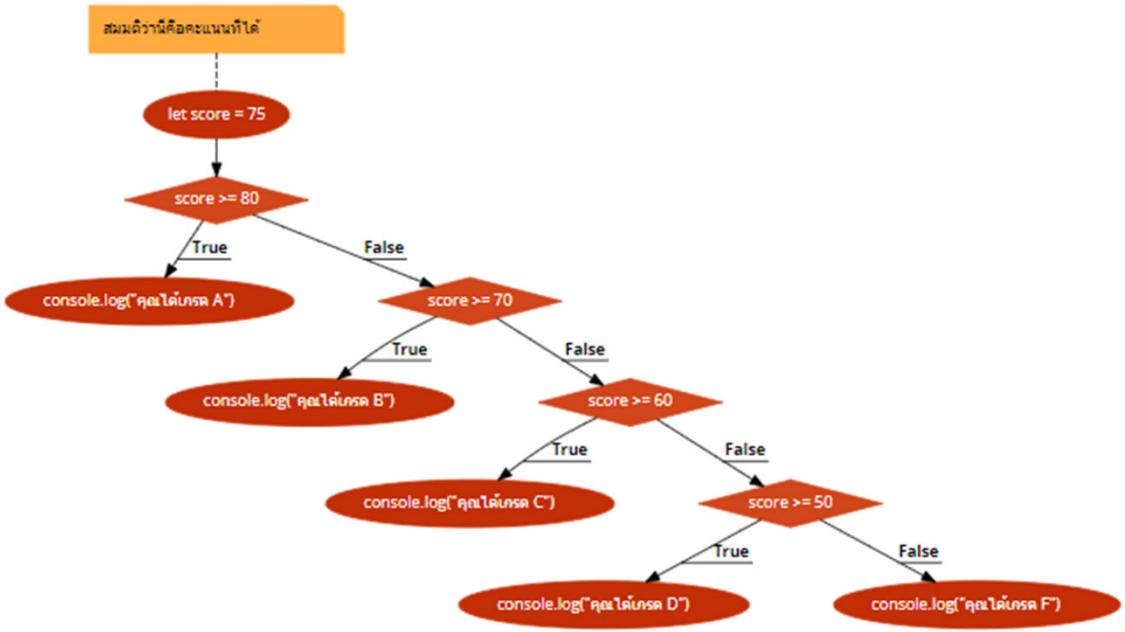
  การตรวจสอบถ้าเงื่อนไขเป็นจริง **(true)**

  else ทำเมื่อเงื่อนไขเป็นเท็จ **(false)**

- การตั้งเงื่อนไข

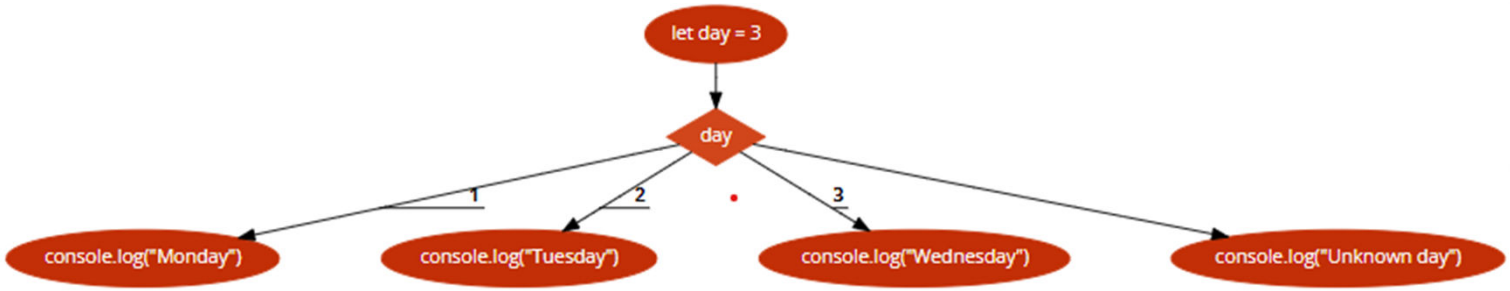
  การเปรียบเทียบหรือการประเมินค่า

```
let score = 75; // สมมติว่านี่คือคะแนนที่ได้
if (score >= 80) {
  console.log("คุณได้เกรด A");
} else if (score >= 70) {
  console.log("คุณได้เกรด B");
} else if (score >= 60) {
  console.log("คุณได้เกรด C");
} else if (score >= 50) {
  console.log("คุณได้เกรด D");
} else {
  console.log("คุณได้เกรด F");
}
```



การทำงานแบบเงื่อนไข**switch-case**

- การตรวจสอบค่าของตัวแปรต่อหลายกรณี
- แต่ละกรณี **case** ตรวจสอบค่าที่เท่ากับค่าของตัวแปร
- ส่งผ่านไปยัง **switch** มีการจับคู่
- **break** ใช้เพื่อหยุดการประมวลผลและออกจาก switch
- **default** ใช้สำหรับการจัดการเคสที่ไม่ตรงกับกรณีใด




```
let day = 3;
switch (day) {
    case 0:
        console.log("Sunday");
        break;
    case 1:
        console.log("Monday");
        break;
    case 2:
        console.log("Tuesday");
        break;
    case 3:
        console.log("Wednesday");
        // Output: Wednesday
        break;
    default:
        console.log("Unknown day");
}
```

การทำงานแบบวนซ้ำ

for

- การใช้งาน

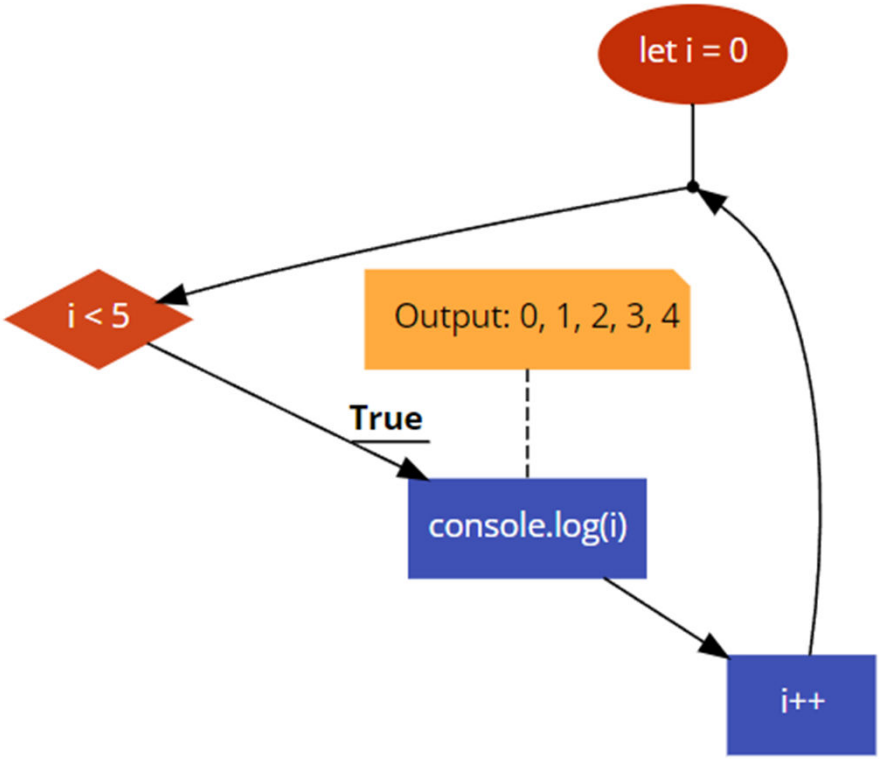
  เพื่อวนซ้ำโค้ดบล็อกตามจำนวนครั้งที่กำหนด

- เงื่อนไขและการปรับค่า

  for ลูปประกอบด้วย 3 ส่วน

  - การกำหนดค่าเริ่มต้น
  - เงื่อนไข
  - การปรับค่า

```
for (let i = 0; i < 5; i++) {
    console.log(i);  // Output: 0, 1, 2, 3, 4
}
```



การทำงานแบบวนซ้ำ

while

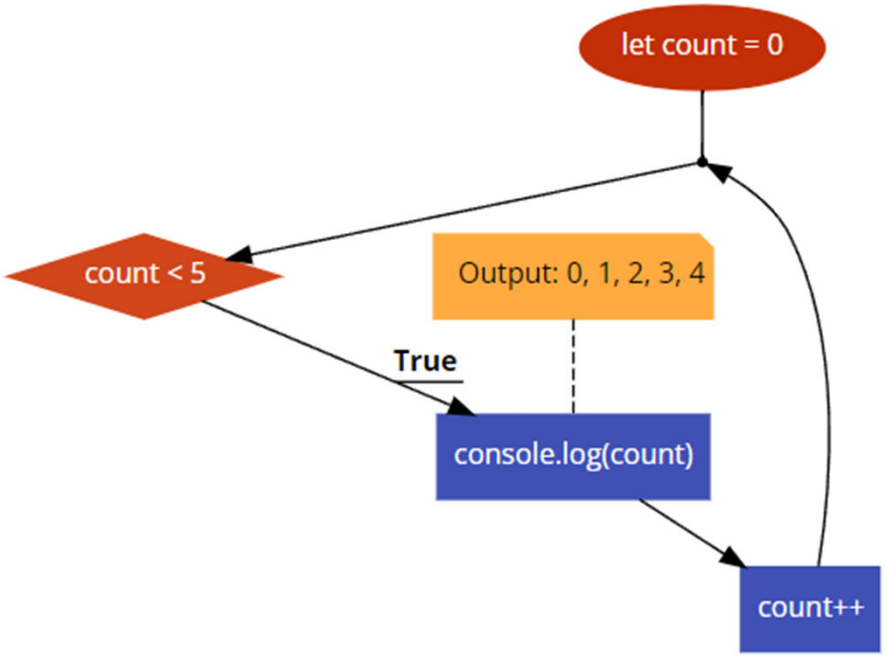
- การใช้งาน

  วนซ้ำโค้ดบล็อกตราบใดที่เงื่อนไขยังคงเป็นจริง

- การกำหนดเงื่อนไข

  ตรวจสอบก่อนที่จะประมวลผลโค้ดในลูป

```
let count = 0;
while (count < 5) {
    console.log(count);  // Output: 0, 1, 2, 3, 4
    count++;
}
```

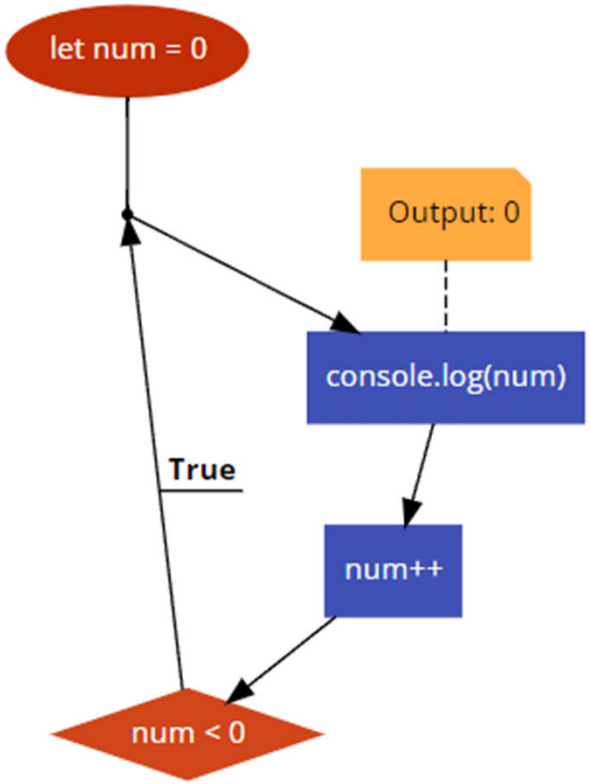


การทำงานแบบวนซ้ำ

do-while

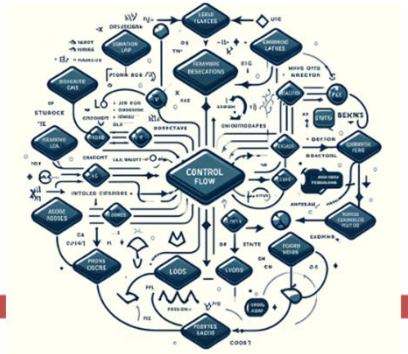
- การใช้งาน
  - วนซ้ำโค้ดบล็อกอย่างน้อยหนึ่งครั้ง
  - แล้วจึงตรวจสอบเงื่อนไข
- การกำหนดเงื่อนไข
  - ตรวจสอบหลังจากประมวลผลโค้ดลูปครั้งแรก

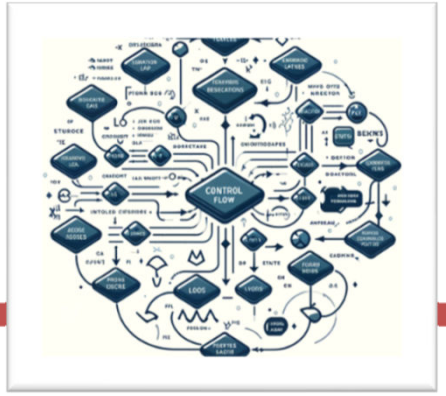
```
let num = 0;
do {
    console.log(num);  // Output: 0
    num++;
} while (num < 0);
```

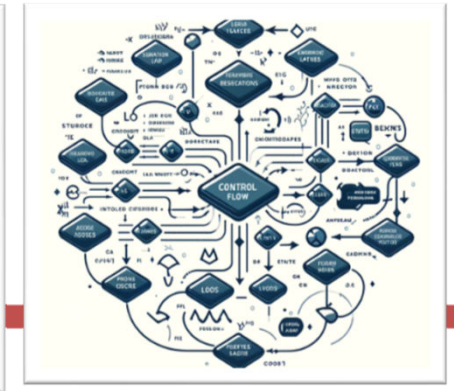


สรุปท้ายบท

- คำสั่งจะถูกดำเนินการตามลำดับที่กำหนด
- การกำหนดเงื่อนไข if ตรวจสอบเงื่อนไขและเลือกทำคำสั่งที่เป็นจริง ส่วนเงื่อนไขที่เป็นเท็จ ใช้ else
- switch-case ใช้ในการจัดการกับหลายเงื่อนไข
- การทำงานแบบวนซ้ำอย่าง for, while, do-while
- โปรแกรมควบคุมการทำงานได้และตอบสนองต่อข้อมูลและเหตุการณ์







ถาม-ตอบ