ส่วนประกอบของโปรแกรม

Program components

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

สาขาวิชาวิศวกรรมซอฟต์แวร์ คณะวิทยาศาสตร์และเทคโนโลยี

บทนำ

- ตัวแปร และ ค่าคงที่
- ชนิดข้อมูล
  - ข้อมูลพื้นฐาน
  - ข้อมูลเชิงประกอบ
- การดำเนินการ
  - การคำนวณ
  - การเปรียบเทียบ
  - การกำหนดค่า

ตัวแปร (Variables)

การสร้างตัวบ่งชี้หรือที่อยู่ในหน่วยความจำที่เปลี่ยนแปลงค่าได้

- **การประกาศตัวแปร (Declaration)**

  การกำหนดให้รู้ว่ามีตัวแปรและชนิดข้อมูล

- **การกำหนดค่าให้ตัวแปร (Assignment)**

  การให้ค่าเริ่มต้นหรือค่าใหม่ให้ตัวแปร

- **ขอบเขตของตัวแปร (Scope)**

  ขอบเขตที่ตัวแปรสามารถถูกเข้าถึงได้

imageURL

ตัวแปร (Variables)

โค้ดภาษา JavaScript

```
let name = "John";  // การประกาศตัวแปรและการกำหนดค่า
name = "Doe";       // การกำหนดค่าใหม่

function greet() {
    let greeting = "Hello";
// ตัวแปร greeting มีขอบเขตภายในฟังก์ชัน greet เท่านั้น
    console.log(greeting + " " + name);
}

greet();  // Output: Hello Doe
```

ค่าคงที่ (Constants)

ตัวแปรที่ไม่สามารถเปลี่ยนแปลงค่าได้หลังจากที่ถูกกำหนดค่าเริ่มต้น

- **ความหมาย:** การเก็บค่าที่ไม่เปลี่ยนแปลงตลอดการทำงาน
- **การกำหนด:** การกำหนดค่าคงที่จะต้องกำหนดค่าทันทีที่ประกาศ
- **การใช้งาน:** ค่าคงที่สามารถใช้งานได้เหมือนตัวแปร แต่ไม่เปลี่ยนแปลงค่าได้

imageURL

ค่าคงที่ (Constants)

โค้ดภาษา JavaScript

```
const PI = 3.14159;  // การประกาศและการกำหนดค่าคงที่


function calculateArea(radius) {
    return PI * radius * radius;  // การใช้งานค่าคงที่
}


let area = calculateArea(5);
console.log(area);  // Output: 78.53975
```

ข้อมูลพื้นฐาน (Primitive Data Types)

JavaScript มีข้อมูลพื้นฐาน 3 ประเภท

- ตัวเลข (Number) ประเภทข้อมูลนี้ใช้เก็บค่าตัวเลข

```
let age = 25;

let price = 19.99;
```

- ข้อความ (String) ประเภทข้อมูลนี้ใช้เก็บข้อความหรือตัวอักษร

```
let name = "John";
```

- บูลีน (Boolean) ประเภทข้อมูลนี้มีค่าเป็น true หรือ false

```
let isApproved = true;
```

ข้อมูลเชิงประกอบ (Composite Data Types)

ข้อมูลเชิงประกอบเป็นการรวมกลุ่มของข้อมูลพื้นฐาน

- **อาร์เรย์ (Array)** การจัดเก็บข้อมูลหลาย ๆ ค่าในตัวแปรเดียว

```
let fruits = ["Apple", "Banana", "Cherry"];
```

- **สตริง (String)** สามารถจัดการได้เหมือนข้อมูลเชิงประกอบ เข้าถึงตัวอักษรแต่ละตัวได้

```
let greeting = "Hello, World!";

console.log(greeting[0]); // Output: H
```

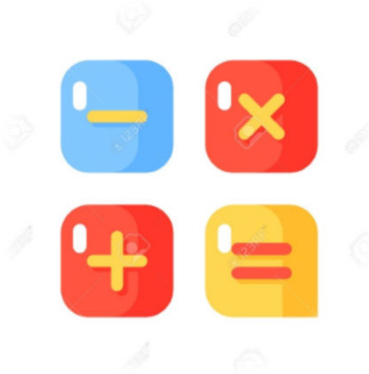
ข้อมูลเชิงประกอบ (Composite Data Types)

- **สตรัคเจอร์ (Structure)** JavaScript ไม่มีประเภทข้อมูลสตรัคเจอร์
- แต่ใช้ Object หรือ Class

```
let person = { // อ็อบเจ็คที่ใช้เป็นสตรัคเจอร์ข้อมูล
firstName: "John",
lastName: "Doe",
age: 25
};
class Car { // คลาสที่ใช้เป็นสตรัคเจอร์ข้อมูล
    constructor(brand, model, year) {
                this.brand = brand;
                this.model = model;
                this.year = year;
                }
}
let myCar = new Car("Toyota", "Corolla", 2020)
```

การคำนวณ

- การดำเนินการทางคณิตศาสตร์ใน JavaScript
  - การบวก (+) let sum = 5 + 3; // sum is now 8
  - การลบ (-) let difference = 5 - 3; // difference is now 2
  - การคูณ (*) let product = 5 * 3; // product is now 15
  - การหาร (/) let quotient = 6 / 3; // quotient is now 2
  - การยกกำลัง (**) let power = 5 ** 3; // power is now 125



imageURL

การเปรียบเทียบ

- การเปรียบเทียบความสัมพันธ์ระหว่างค่าต่าง ๆ
  - มากกว่า (>) console.log(5 > 3); // Output: true
  - น้อยกว่า (<) console.log(5 < 3); // Output: false
  - เท่ากับ (== หรือ ===) console.log(5 == 5); // Output: true
  - ไม่เท่ากับ (!= หรือ !==) console.log(5 !== 3); // Output: true

การกำหนดค่า

- การกำหนดค่าแบบตรง (Direct Assignment)
  - let x = 5; // x is now 5
- การกำหนดค่าผ่านการดำเนินการ (Assignment via Operations)
  - let y = 10;
  - y += 5; // y is now 15 (same as y = y + 5)
  - y -= 3; // y is now 12 (same as y = y - 3)
  - y *= 2; // y is now 24 (same as y = y * 2)
  - y /= 4; // y is now 6 (same as y = y / 4)

imageURL

สรุปท้ายบท

- ตัวแปรและค่าคงที่เป็นส่วนพื้นฐานที่จำเป็นในการเก็บและจัดการข้อมูล
- ค่าคงที่เป็นข้อมูลที่ไม่เปลี่ยนแปลง
- ชนิดข้อมูลที่ตัวแปรสามารถเก็บได้ เช่น ตัวเลข, ตัวอักษร, หรือวัตถุ
- กระบวนการที่จัดการและแปรข้อมูลที่เก็บในตัวแปร
- การคำนวณพื้นฐานเช่นการบวก ลบ คูณ และหาร การเปรียบเทียบ
- การกำหนดค่าที่ตัวแปรรับได้ถูกต้องตามเงื่อนไข
- การเขียนโค้ดที่มีโครงสร้างดี ง่ายต่อการอ่านและบำรุงรักษา

ถาม-ตอบ