สัปดาห์ที่ 5: JavaScript Fundamentals - ปฏิบัติการ

ปฏิบัติการ: 5 กิจกรรม

วัตถุประสงค์

- เข้าใจ Variables (const, let) และ Data Types
- เขียน Functions ได้อย่างถูกต้อง
- ใช้ Control Flow (if/else, switch)
- ทำความเข้าใจ Scope และ Closure
- ใช้ Loops และ Array Methods

กิจกรรมที่ 1: Variables & Data Types Practice

วัตถุประสงค์

ฝึกการใช้ variables, data types, และ การบังคับ type (coercion)

ขั้นตอน

1. **สร้างไฟล์ 01-variables.js:**

```
// ============================================
// Activity 1: Variables & Data Types
// ============================================

console.log("=== Variables & Data Types Practice ===\n");

// ─────────────────────────────────
// 1. Using const vs let
// ─────────────────────────────────

// Good: const for constants
const MAX_USERS = 100;
const PI = 3.14159;
const GRAVITY = 9.8;

console.log("Constants:");
console.log("MAX_USERS:", MAX_USERS);
console.log("PI:", PI);

// Good: let for variables that change
let count = 0;
count++;
count++;
console.log("\nVariable (let):");
console.log("count after increment:", count);

// ❌ Bad: var (avoid!)
// var oldStyle = "Don't use this";

// ─────────────────────────────────
// 2. Primitive Data Types
// ─────────────────────────────────

console.log("\n=== Primitive Data Types ===");

// Numbers
const age = 25;
const height = 5.9;
const temperature = -10;
```

```
console.log("Numbers:", age, height, temperature);

// Strings
const firstName = "John";
const lastName = "Doe";
const fullName = `${firstName} ${lastName}`; // Template literal
console.log("Strings:", fullName);

// Booleans
const isStudent = true;
const isTeacher = false;
console.log("Booleans:", "isStudent:", isStudent, "isTeacher:", isTeacher);

// null & undefined
const emptyValue = null;
let noValue;
console.log("null:", emptyValue);
console.log("undefined:", noValue);

// ------------------------------------
// 3. Object Data Types
// ------------------------------------

console.log("\n=== Object Data Types ===");

// Array
const fruits = ["apple", "banana", "orange"];
console.log("Array:", fruits);
console.log("First fruit:", fruits[0]);
console.log("Array length:", fruits.length);

// Object
const person = {
  name: "John",
  age: 25,
  city: "Bangkok",
  isStudent: true,
};
console.log("Object:", person);
console.log("Person name:", person.name);
console.log("Person age:", person.age);

// ------------------------------------
// 4. typeof Operator
// ------------------------------------

console.log("\n=== typeof Operator ===");
console.log("typeof 25:", typeof 25);
console.log("typeof 'hello':", typeof "hello");
console.log("typeof true:", typeof true);
console.log("typeof undefined:", typeof undefined);
console.log("typeof []:", typeof []); // "object" (ทำไม ถึงเป็น object…)
// วิธีตรวจสอบ Array อย่างถูกต้อง:
// Array.isArray(arr)      // true
// arr instanceof Array    // true
console.log("typeof {}:", typeof {}); // "object"
console.log("typeof (() => {}):", typeof (() => {})); // "function"
```

```
// ──────────────────────────────────
// 5. Type Coercion Examples
// ──────────────────────────────────

console.log("\n=== Type Coercion ===");

// Implicit coercion (auto)
console.log("'5' + 2:", "5" + 2); // "52" (string concat)
console.log("'5' - 2:", "5" - 2); // 3 (numeric)
console.log("'5' * 2:", "5" * 2); // 10 (numeric)
console.log("true + 1:", true + 1); // 2

// Explicit coercion
console.log("\nExplicit coercion:");
console.log("String(25):", String(25));
console.log("Number('25'):", Number("25"));
console.log("Boolean(1):", Boolean(1));
console.log("Boolean(0):", Boolean(0));
console.log("Boolean(''):", Boolean(""));
console.log("Boolean('hello'):", Boolean("hello"));

// ──────────────────────────────────
// 6. Challenge: Create a Person Object
// ──────────────────────────────────

console.log("\n=== Challenge: Person Object ===");

const student = {
  firstName: "Alice",
  lastName: "Smith",
  age: 20,
  gpa: 3.8,
  courses: ["HTML", "CSS", "JavaScript"],
  isActive: true,

  // Method (function in object)
  getFullName: function () {
    return `${this.firstName} ${this.lastName}`;
  },

  getInfo: function () {
    return `${this.getFullName()}, Age: ${this.age}, GPA: ${this.gpa}`;
  },
};

console.log("Student object:");
console.log(student);
console.log("Full name:", student.getFullName());
console.log("Info:", student.getInfo());
console.log("Courses:", student.courses.join(", "));
```

```
// ──────────────────────────────────
// 7. Truthy vs Falsy
// ──────────────────────────────────

console.log("\n=== Truthy vs Falsy ===");

// Falsy values
const falsyValues = [0, "", null, undefined, false, NaN];
console.log("Falsy values:");
falsyValues.forEach((val) => {
  console.log(
    `  ${typeof val === "string" ? `"${val}"` : val}: ${Boolean(val)}`
  );
});

// Truthy values
const truthyValues = [1, "hello", true, [], {}, () => {}];
console.log("\nTruthy values:");
truthyValues.forEach((val) => {
  const display = Array.isArray(val)
    ? "[]"
    : typeof val === "function"
    ? "() => {}"
    : typeof val === "object"
    ? "{}"
    : val;
  console.log(`  ${display}: ${Boolean(val)}`);
});

console.log("\n☑ Activity 1 completed!");
```

2. **เรียกใช้ script บน Node.js:**

```
// เปิด Terminal และพิมพ์คำสั่ง
node 01-variables.js
```

3. **ดูผลลัพธ์** - ควรเห็น output ของ variables, data types, และ type coercion

Git Commit - Activity 1

```
git add 01-variables.js
git commit -m "Add: JavaScript variables and data types practice"
```

กิจกรรมที่ 2: Functions & Arrow Functions

วัตถุประสงค์

ฝึกการเขียน functions แบบต่างๆ และเข้าใจ arrow functions

ขั้นตอน

1. **สร้างไฟล์ 02-functions.js:**

```
// ============================================
// Activity 2: Functions & Arrow Functions
// ============================================

console.log("=== Functions & Arrow Functions Practice ===\n");

// ─────────────────────────────────
// 1. Function Declaration
// ─────────────────────────────────

function greet(name) {
  return `Hello, ${name}!`;
}

console.log("Function Declaration:");
console.log(greet("John"));
console.log(greet("Alice"));

// ─────────────────────────────────
// 2. Function Expression  --  สำคัญมาก
// ─────────────────────────────────

const add = function (a, b) {
  return a + b;
};

console.log("\nFunction Expression:");
console.log("add(5, 3):", add(5, 3));
console.log("add(10, 20):", add(10, 20));

// ─────────────────────────────────
// 3. Arrow Functions (Modern!) --  สำคัญมาก
// ─────────────────────────────────

// Full syntax
const multiply = (a, b) => {
  return a * b;
};
console.log("\nArrow Function (full syntax):");
console.log("multiply(4, 5):", multiply(4, 5));

// Shorthand (implicit return)
const square = (x) => x * x;
console.log("Arrow Function (shorthand):");
console.log("square(5):", square(5));

// Single parameter (no parens)
const double = (x) => x * 2;
console.log("double(10):", double(10));

// No parameters
```

```js
const getRandom = () => Math.floor(Math.random() * 100);
console.log("getRandom():", getRandom());

// ──────────────────────────────────
// 4. Default Parameters
// ──────────────────────────────────

function introduce(name = "Anonymous", age = 0, city = "Unknown") {
  return `${name} is ${age} years old from ${city}`;
}

console.log("\nDefault Parameters:");
console.log(introduce()); // All defaults
console.log(introduce("John")); // Name only
console.log(introduce("John", 25)); // Name + age
console.log(introduce("John", 25, "Bangkok")); // All

// ──────────────────────────────────
// 5. Rest Parameters (...args)  --  สำคัญมาก
// ──────────────────────────────────

function sum(...numbers) {
  let total = 0;
  for (const num of numbers) {
    total += num;
  }
  return total;
}

console.log("\nRest Parameters:");
console.log("sum(1, 2, 3):", sum(1, 2, 3));
console.log("sum(5, 10, 15, 20):", sum(5, 10, 15, 20));
console.log("sum():", sum()); // 0

// Better way: reduce
const sumWithReduce = (...nums) => nums.reduce((total, n) => total + n, 0);
console.log("sumWithReduce(2, 4, 6, 8):", sumWithReduce(2, 4, 6, 8));

// ──────────────────────────────────
// 6. Destructuring Parameters  --  สำคัญมาก
// ──────────────────────────────────

function printUser({ name, age, city }) {
  console.log(`${name}, ${age} years old, from ${city}`);
}

console.log("\nDestructuring Parameters:");
const user = { name: "Alice", age: 22, city: "Chiang Mai" };
printUser(user);
```

```
// ————————————————————————————
// 7. Validation Function (Early Return)
// ————————————————————————————

function validateEmail(email) {
  // Early returns for error cases
  if (!email) {
    return { valid: false, message: "Email is required" };
  }

  if (email.indexOf("@") === -1) {
    return { valid: false, message: "Invalid email format" };
  }

  if (email.indexOf(".") === -1) {
    return { valid: false, message: "Missing domain extension" };
  }

  return { valid: true, message: "Email is valid" };
}

console.log("\nValidation Function:");
console.log(validateEmail(""));
console.log(validateEmail("invalidemail"));
console.log(validateEmail("invalid@email"));
console.log(validateEmail("valid@email.com"));

// ————————————————————————————
// 8. Returning Objects
// ————————————————————————————

function createUser(firstName, lastName, age) {
  return {
    firstName, // shorthand for firstName: firstName  --  สำคัญมาก
    lastName,
    age,
    email: `${firstName.toLowerCase()}.${lastName.toLowerCase()}@example.com`,
    getFullName() {
      // shorthand for getFullName: function() {}
      return `${this.firstName} ${this.lastName}`;
    },
    getAge() {
      return this.age;
    },
  };
}

console.log("\nReturning Objects:");
const newUser = createUser("John", "Doe", 30);
console.log(newUser);
console.log("Email:", newUser.email);
console.log("Full name:", newUser.getFullName());
```

```
// ──────────────────────────────────
// 9. Function as Parameter (Callback)
// ──────────────────────────────────

function processArray(arr, callback) {
  const result = [];
  for (const item of arr) {
    result.push(callback(item));
  }
  return result;
}

const numbers = [1, 2, 3, 4, 5];
const doubled = processArray(numbers, (x) => x * 2);
const squared = processArray(numbers, (x) => x * x);

console.log("\nCallback Function:");
console.log("Original:", numbers);
console.log("Doubled:", doubled);
console.log("Squared:", squared);

// ──────────────────────────────────
// 10. Challenge: Calculator
// ──────────────────────────────────

const calculator = {
  add: (a, b) => a + b,
  subtract: (a, b) => a - b,
  multiply: (a, b) => a * b,
  divide: (a, b) => (b === 0 ? "Cannot divide by zero" : a / b),
  power: (a, b) => Math.pow(a, b),

  operate(operation, a, b) {
    if (this[operation]) {
      return this[operation](a, b);
    }
    return "Unknown operation";
  },
};

console.log("\nChallenge: Calculator");
console.log("5 + 3 =", calculator.add(5, 3));
console.log("10 - 4 =", calculator.subtract(10, 4));
console.log("6 * 7 =", calculator.multiply(6, 7));
console.log("20 / 4 =", calculator.divide(20, 4));
console.log("2 ^ 10 =", calculator.power(2, 10));
console.log("Using operate('add', 5, 3) =", calculator.operate("add", 5, 3));

console.log("\n☑ Activity 2 completed!");
```

2. **เรียกใช้ script:**

```
node 02-functions.js
```

3. **สังเกตผลลัพธ์** - Functions แบบต่างๆ ทำงานอย่างไร

Git Commit - Activity 2

```
git add 02-functions.js
git commit -m "Add: Functions and arrow functions practice"
```

กิจกรรมที่ 3: Control Flow & Logic

วัตถุประสงค์

ฝึกการใช้ if/else, switch, และ logical operators

ขั้นตอน

1. **สร้างไฟล์ 03-control-flow.js:**

```
// ============================================
// Activity 3: Control Flow & Logic
// ============================================

console.log("=== Control Flow & Logic Practice ===\n");

// ——————————————————————————
// 1. if/else Statements
// ——————————————————————————

function checkAge(age) {
  if (age < 13) {
    return "Child";
  } else if (age < 18) {
    return "Teenager";
  } else if (age < 60) {
    return "Adult";
  } else {
    return "Senior";
  }
}

console.log("Age Classification:");
console.log("Age 5:", checkAge(5));
console.log("Age 15:", checkAge(15));
console.log("Age 25:", checkAge(25));
console.log("Age 65:", checkAge(65));

// ——————————————————————————
// 2. Switch Statement
// ——————————————————————————

function getDayName(dayNum) {
  switch (dayNum) {
    case 1:
      return "Monday";
    case 2:
      return "Tuesday";
    case 3:
      return "Wednesday";
    case 4:
      return "Thursday";
```

```
    case 5:
      return "Friday";
    case 6:
      return "Saturday";
    case 7:
      return "Sunday";
    default:
      return "Unknown day";
  }
}

console.log("\nDay Names:");
for (let i = 1; i <= 8; i++) {
  console.log(`Day ${i}:`, getDayName(i));
}

// ──────────────────────────────────
// 3. Ternary Operator
// ──────────────────────────────────

const isWeekend = (day) => (day === 6 || day === 7 ? "Weekend" : "Weekday");
console.log("\nWeekday/Weekend:");
console.log("Monday (1):", isWeekend(1));
console.log("Saturday (6):", isWeekend(6));

// ──────────────────────────────────
// 4. Logical Operators (&&, ||, !)
// ──────────────────────────────────

console.log("\nLogical Operators:");

const age = 25;
const hasLicense = true;
const hasInsurance = true;

// AND (&&) - all must be true
const canDrive = age >= 18 && hasLicense && hasInsurance;
console.log("Can drive:", canDrive);

// OR (||) - at least one must be true
const isSpecial = age === 18 || age === 21 || age === 25;
console.log("Is special age:", isSpecial);

// NOT (!)
const isNotAdult = !(age >= 18);
console.log("Is not adult:", isNotAdult);
```

```
// ──────────────────────────────────
// 5. Short-Circuit Evaluation -- JavaScript หยุดประเมินค่าตรงกลางเมื่อรู้ผลลัพธ์แล้ว
// ──────────────────────────────────

console.log("\nShort-Circuit Evaluation:");

const user = { name: "John", age: 25 };
const admin = null;

// OR: use default value
const userName = admin?.name || user.name || "Anonymous";
console.log("User name:", userName);
// ?. คือการใช้ Optional Chaining - เป็นวิธีที่ปลอดภัยในการเข้าถึง properties ของ object ที่อาจ
เป็น null หรือ undefined
// admin?.name ก็คือ ถ้า admin มีค่า ให้เข้าถึง .name ไม่เช่นนั้นให้คืนค่า undefined
// 1. admin?.name
//    - admin คือ null ❌
//    - ไม่ error, ส่งคืน undefined
// 2. undefined || user.name
//    - user.name คือ "John" ☑
//    - ใช้ค่านี้ → "John"
// 3. ผลลัพธ์: "John"

// AND: check before accessing
const userProfile = user && user.profile;
console.log("User profile:", userProfile); // undefined

// ──────────────────────────────────
// 6. Grading System
// ──────────────────────────────────

function getGrade(score) {
  if (score >= 90) {
    return "A";
  } else if (score >= 80) {
    return "B";
  } else if (score >= 70) {
    return "C";
  } else if (score >= 60) {
    return "D";
  } else {
    return "F";
  }
}

console.log("\nGrading System:");
const scores = [95, 85, 75, 65, 55];
scores.forEach((score) => {
  console.log(`Score ${score}: Grade ${getGrade(score)}`);
});
```

```
// ————————————————————————————
// 7. Form Validation
// ————————————————————————————

function validateRegistration(formData) {
  // Create validation result object
  const errors = [];

  // Validate name
  if (!formData.name || formData.name.trim() === "") {
    errors.push("Name is required");
  } else if (formData.name.length < 3) {
    errors.push("Name must be at least 3 characters");
  }


  // Validate email
  if (!formData.email || formData.email.indexOf("@") === -1) {
    errors.push("Valid email is required");
  }

  // Validate age
  if (!formData.age || formData.age < 18) {
    errors.push("Must be 18 or older");
  }

  // Validate password
  if (!formData.password || formData.password.length < 6) {
    errors.push("Password must be at least 6 characters");
  }

  // Check if agree to terms
  if (!formData.agreeToTerms) {
    errors.push("Must agree to terms");
  }

  return {
    isValid: errors.length === 0,
    errors: errors,
  };
}

console.log("\nForm Validation:");

const validUser = {
  name: "John Doe",
  email: "john@example.com",
  age: 25,
  password: "securepass123",
  agreeToTerms: true,
};

const invalidUser = {
  name: "Jo",
  email: "invalidemail",
  age: 15,
  password: "pass",
```

```
  agreeToTerms: false,
};

console.log("Valid user:", validateRegistration(validUser));
console.log("Invalid user:", validateRegistration(invalidUser));

// ──────────────────────────────────────
// 8. Challenge: Traffic Light
// ──────────────────────────────────────

function getTrafficAction(color) {
  switch (color) {
    case "red":
      return "🛑 STOP";
    case "yellow":
      return "⚠️ SLOW DOWN";
    case "green":
      return "✅ GO";
    default:
      return "❓ INVALID COLOR";
  }
}

console.log("\nChallenge: Traffic Light");
const lights = ["red", "yellow", "green", "blue"];
lights.forEach((light) => {
  console.log(`${light}: ${getTrafficAction(light)}`);
});

console.log("\n☑️ Activity 3 completed!");
```

2. **เรียกใช้ script:**

```
node 03-control-flow.js
```

Git Commit - Activity 3

```
git add 03-control-flow.js
git commit -m "Add: Control flow and logical operators practice"
```

กิจกรรมที่ 4: Loops & Array Methods

วัตถุประสงค์

ฝึกการใช้ loops และ array methods (map, filter, reduce)

ขั้นตอน

1. **สร้างไฟล์ 04-loops.js:**

```
// ==========================================
// Activity 4: Loops & Array Methods
// ==========================================

console.log("=== Loops & Array Methods Practice ===\n");

// ------------------------------------
// 1. Traditional for loop
// ------------------------------------

console.log("For loop (0-4):");
for (let i = 0; i < 5; i++) {
  console.log(`  i = ${i}`);
}

// ------------------------------------
// 2. while loop
// ------------------------------------

console.log("\nWhile loop (count down):");
let count = 5;
while (count > 0) {
  console.log(`  ${count}...`);
  count--;
}
console.log("  Blastoff! 🚀");

// ------------------------------------
// 3. for...of loop (values)
// ------------------------------------

const fruits = ["apple", "banana", "orange"];
console.log("\nFor...of loop (fruits):");
for (const fruit of fruits) {
  console.log(`  - ${fruit}`);
}

// ------------------------------------
// 4. for...in loop (keys - for objects)
// ------------------------------------

const person = { name: "John", age: 25, city: "Bangkok" };
console.log("\nFor...in loop (person properties):");
for (const key in person) {
  console.log(`  ${key}: ${person[key]}`);
}
```

```
// ──────────────────────────────────
// 5. forEach (side effects)
// ──────────────────────────────────

console.log("\nforEach (with index):");
fruits.forEach((fruit, index) => {
  console.log(`  ${index}: ${fruit}`);
});

// ──────────────────────────────────
// 6. map (transform to new array) --  สำคัญมาก
// ──────────────────────────────────

const numbers = [1, 2, 3, 4, 5];
console.log("\nmap - transform elements:");
console.log("Original:", numbers);

const doubled = numbers.map((n) => n * 2);
console.log("Doubled:", doubled);

const squared = numbers.map((n) => n * n);
console.log("Squared:", squared);

const asStrings = numbers.map((n) => `Number: ${n}`);
console.log("As strings:", asStrings);

// ──────────────────────────────────
// 7. filter (select matching elements)  --  สำคัญมาก
// ──────────────────────────────────

console.log("\nfilter - select elements:");

const evens = numbers.filter((n) => n % 2 === 0);
console.log("Even numbers:", evens);

const odds = numbers.filter((n) => n % 2 !== 0);
console.log("Odd numbers:", odds);

const greaterThan2 = numbers.filter((n) => n > 2);
console.log("Numbers > 2:", greaterThan2);

// ──────────────────────────────────
// 8. reduce (accumulate to single value)  --  สำคัญมาก
// ──────────────────────────────────

console.log("\nreduce - accumulate:");

const sum = numbers.reduce((total, n) => total + n, 0);
console.log("Sum:", sum);

const product = numbers.reduce((total, n) => total * n, 1);
console.log("Product:", product);

const concatenated = numbers.reduce((str, n) => str + n, "");
console.log("Concatenated:", concatenated);
```

```
// Count occurrences
const words = ["apple", "banana", "apple", "orange", "apple"];
const wordCount = words.reduce((counts, word) => {
  counts[word] = (counts[word] || 0) + 1;
  return counts;
}, {});
console.log("Word count:", wordCount);

// ──────────────────────────────────
// 9. Chaining methods  --  สำคัญมาก
// ──────────────────────────────────

console.log("\nMethod chaining:");

const data = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

// Filter even > map to string > join
const evenStrings = data
  .filter((n) => n % 2 === 0)           // [2, 4, 6, 8, 10]
  .map((n) => `${n}²=${n * n}`)         // ["2²=4", "4²=16", ...]
  .join(", "); // "2²=4, 4²=16, ..."

console.log("Even numbers squared:", evenStrings);

// Calculate average with reduce and length
const numbers2 = [10, 20, 30, 40, 50];
const average = numbers2.reduce((sum, n) => sum + n, 0) / numbers2.length;
console.log("Average:", average);

// ──────────────────────────────────
// 10. Challenge: Student Grades
// ──────────────────────────────────
const students = [
  { name: "Alice", score: 95 },
  { name: "Bob", score: 75 },
  { name: "Charlie", score: 85 },
  { name: "Diana", score: 92 },
  { name: "Eve", score: 88 },
];

console.log("\nChallenge: Student Analysis");
console.log("Students:", students);

// 1. Get all names
const names = students.map((s) => s.name);
console.log("Names:", names.join(", "));

// 2. Filter high scorers (>= 85)
const highScorers = students.filter((s) => s.score >= 85);
console.log(
  "High scorers:",
  highScorers.map((s) => `${s.name} (${s.score})`).join(", ")
);

// 3. Calculate class average
const classAverage =
  students.reduce((sum, s) => sum + s.score, 0) / students.length;
console.log("Class average:", classAverage.toFixed(2));
```

```
// 4. Find top scorer
const topScorer = students.reduce((top, s) => (s.score > top.score ? s : top));
console.log("Top scorer:", `${topScorer.name} (${topScorer.score})`);

// 5. Create summary
const summary = students
  .map((s) => ({
    ...s,
    grade: s.score >= 90 ? "A" : s.score >= 80 ? "B" : "C",
  }))
  .sort((a, b) => b.score - a.score);
console.log("Summary (sorted):");
summary.forEach((s) => console.log(`  ${s.name}: ${s.score} (${s.grade})`));

console.log("\n☑ Activity 4 completed!");
```

2. **เรียกใช้ script:**

```
node 04-loops.js
```

Git Commit - Activity 4

```
git add 04-loops.js
git commit -m "Add: Loops and array methods (map, filter, reduce) practice"
```

กิจกรรมที่ 5: Integration & Summary

วัตถุประสงค์

รวมทั้งหมดเข้าด้วยกันในโปรแกรม "Quiz Application"

ขั้นตอน

1. **สร้างไฟล์ 05-integration.js (Quiz Application):**

```
// ============================================
// Activity 5: Integration - Quiz Application
// ============================================

console.log("🎯 === QUIZ APPLICATION === 🎯\n");

// Quiz data
const quizzes = [
  {
    question: "What is 5 + 3?",
    options: ["8", "7", "6", "9"],
    correctAnswer: 0,                  // Index of correct option
  },
  {
    question: "What is the capital of Thailand?",
    options: ["Phuket", "Bangkok", "Chiang Mai", "Pattaya"],
    correctAnswer: 1,
  },
  {
    question: "What is the largest planet?",
    options: ["Mars", "Saturn", "Jupiter", "Neptune"],
    correctAnswer: 2,
```

```
  },
  {
    question: "What is 2^8?",
    options: ["128", "256", "64", "512"],
    correctAnswer: 1,
  },
  {
    question: "Which is NOT a JavaScript data type?",
    options: ["string", "class", "symbol", "boolean"],
    correctAnswer: 1,
  },
];

// Quiz results
let results = [];

// Process each quiz
quizzes.forEach((quiz, index) => {
  const userAnswer = Math.floor(Math.random() * 4);     // จำลองการทำ quiz
  const isCorrect = userAnswer === quiz.correctAnswer;

  results.push({
    questionNum: index + 1,
    question: quiz.question,
    userAnswer: quiz.options[userAnswer],
    correctAnswer: quiz.options[quiz.correctAnswer],
    isCorrect: isCorrect,
  });
});

// Display results
console.log("QUIZ RESULTS:");
console.log("=".repeat(60));

results.forEach((result) => {
  const status = result.isCorrect ? "☑ CORRECT" : "❌ WRONG";
  console.log(`Q${result.questionNum}: ${result.question}`);
  console.log(`  Your answer: ${result.userAnswer}`);
  if (!result.isCorrect) {
    console.log(`  Correct answer: ${result.correctAnswer}`);
  }
  console.log(`  ${status}`);
  console.log();
});

// Calculate score
const correctCount = results.filter((r) => r.isCorrect).length;
const score = (correctCount / results.length) * 100;

console.log("=".repeat(60));
console.log(
  `FINAL SCORE: ${correctCount}/${results.length} (${score.toFixed(1)}%)`
);
```

```
// Grade assignment
let grade;
if (score >= 90) {
  grade = "A";
} else if (score >= 80) {
  grade = "B";
} else if (score >= 70) {
  grade = "C";
} else if (score >= 60) {
  grade = "D";
} else {
  grade = "F";
}

console.log(`GRADE: ${grade}`);

// Feedback
console.log("\nFEEDBACK:");
if (score === 100) {
  console.log("🌟 Perfect score! Excellent work!");
} else if (score >= 80) {
  console.log("👍 Great job! Keep practicing.");
} else if (score >= 60) {
  console.log("📚 Good effort. Review the material and try again.");
} else {
  console.log("💪 Keep practicing. You'll improve!");
}

// Statistics
console.log("\n📊 STATISTICS:");
console.log(`Total questions: ${results.length}`);
console.log(`Correct: ${correctCount}`);
console.log(`Incorrect: ${results.length - correctCount}`);
console.log(`Success rate: ${score.toFixed(1)}%`);

// Category breakdown (if applicable)
const byCorrectness = results.reduce(
  (acc, r) => {
    acc[r.isCorrect ? "correct" : "incorrect"]++;
    return acc;
  },
  { correct: 0, incorrect: 0 }
);

console.log("\nAnswer breakdown:");
console.log(`  ☑ Correct: ${byCorrectness.correct}`);
console.log(`  ❌ Incorrect: ${byCorrectness.incorrect}`);

console.log("\n☑ All activities completed!");
console.log("=".repeat(60));
```

2. **เรียกใช้ script:**

```
node 05-integration.js
```

3. **ดูผลลัพธ์** - โปรแกรมจำลอง quiz และแสดงผลลัพธ์

Git Commit - Activity 5

```
git add 05-integration.js
git commit -m "Add: Integration project - quiz application with all fundamentals"
```

สร้างไฟล์ README.md

ใส่เนื้อหา:

```
# Week 5: JavaScript Fundamentals - Lab Summary

## 📊 ปฏิบัติการที่สำเร็จ

### Activity 1: Variables & Data Types ☑

- ใช้ `const` และ `let` อย่างถูกต้อง
- Primitive types: number, string, boolean, null, undefined
- Object types: array, object, function
- Type coercion: implicit vs explicit

**Key file:** `01-variables.js`

### Activity 2: Functions & Arrow Functions ☑

- Function declarations, expressions, arrow functions
- Default parameters
- Rest parameters (...args)
- Destructuring parameters
- Callbacks

**Key file:** `02-functions.js`

### Activity 3: Control Flow & Logic ☑

- if/else statements
- switch statements
- Ternary operators
- Logical operators (&&, ||, !)
- Form validation

**Key file:** `03-control-flow.js`

### Activity 4: Loops & Array Methods ☑
```

- for, while, for...of, for...in loops
- forEach, map, filter, reduce
- Method chaining
- Student analysis example

Key file: `04-loops.js`

Activity 5: Integration Project ☑

- Quiz application combining all concepts
- Data processing
- Results calculation
- Statistics generation

Key file: `05-integration.js`

🎯 Learning Outcomes

Variables & Scoping

☑ Use `const` by default
☑ Use `let` when value changes
☑ Avoid `var`
☑ Understand block scope

Data Types

☑ Primitive: number, string, boolean, null, undefined
☑ Objects: array, object, function, date
☑ Type checking with `typeof`
☑ Type coercion (implicit & explicit)

Functions

☑ Declaration, expression, arrow syntax
☑ Parameters: default, rest, destructuring
☑ Return values
☑ Callbacks & higher-order functions
☑ Scope & closure

Control Flow

```
☑ Conditional: if/else, switch, ternary
☑ Logical operators: &&, ||, !
☑ Short-circuit evaluation

### Loops & Iteration

☑ Traditional loops: for, while
☑ Iteration: for...of, for...in
☑ Array methods: forEach, map, filter, reduce
☑ Method chaining

---

## 🗀 Files Created
01-variables.js - Variables and data types
02-functions.js - Functions and arrow functions
03-control-flow.js - Control flow and logic
04-loops.js - Loops and array methods
05-integration.js - Quiz application README.md - Summary (this file)
```

Git Commit

```
git add README.md
git commit -m "Add: Week 5 summary - JavaScript fundamentals"
```

O]O