Kiểm tra JavaScript cơ bản

- Mục tiêu: Ôn tập kiến thức và đánh giá kết quả học tập của học viên
- Nội dung: Bài kiểm tra bao gồm 10 bài tập lập trình JavaScript chủ đề khác nhau nhằm đánh giá kiến thức JavaScript căn bản
- Yêu cầu:
 - Làm bài và nộp bài trong thời gian quy định (gửi link github)
 - Comment trong code giải tích chi tiết cách làm (thuật toán, các bước, ...), và links
 tham khảo nếu có
 - Chú ý không đổi tên hàm và giá trị trả về phải đúng định dạng/kiểu dữ liệu theo
 yêu cầu
 - Nếu chỉ có bài làm mà không có giải thích (hoặc giải thích một kiểu làm một kiểu) thì coi như chưa hiểu bài, sẽ bị trừ 1/2 số điểm của bài

Ví du:

```
/**
 * Tîm số nhỏ nhất trong màng
 *
 * @param {number[]} arr - Mảng các số truyên vào
 * @return {number} - Số nhỏ nhất trong mảng
 * @example minOf([2, 5, 1, -1, 4]); // -1
 */

/**
 * Cách làm:
 * Bước 1: Khai báo biến min để giữ kết quả
 * Bước 2: Giả sử phần tử đầu tiên của mảng (arr[0]) là số nhỏ nhất, gán cho min
 * Bước 3: Duyệt qua các giá trị (n) trong mảng
 * - Nếu min < n thì gán lại giá trị cho min</pre>
```

```
* Buớc 4: Trả về kết quả min

*/

function minOf(arr) {
  let min = arr[0];

for (let n of arr) if (n < min) min = n;

return min;
}</pre>
```

Bài tập

```
* Tính số tiền điện theo công thức tính giá điện bậc thang như sau:
 * - Từ kWh thứ 0 đến 50: 1678đ/kWh
 * - Từ kWh thứ 51 - 100: 1734đ/kWh
 * - Từ kWh thứ 101 - 200: 2014đ/kWh
 * - Từ kWh thứ 201 - 300: 2536đ/kWh
 * - Từ kWh thứ 301 - 400: 2834đ/kWh
 * - Từ kWh thứ 401 trở đi: 2927đ/kWh
 * @param {number} kwh - Số điện đã dùng trong tháng
 * @return {number} Tiền điện phải trả
 * @example
 * bill(100); // 170600
function bill(kwh) {}
 * Cho một chuỗi time biểu thị thời gian dạng giờ:phút:giây (VD:
'20:15:45'), và một số n bất kỳ, tính và trả về kết quả là một chuỗi
biểu thị thời gian tương ứng sau n giây
```

```
* @param {string} time - Chuỗi biểu thị thời gian
 * @param {number} n - Số giây cần thêm (bớt)
 * @return {string} Kết quả
 * @example
 * calcTime('20:15:45', 15); // '20:16:00'
 * calcTime('20:15:45', -46); // '20:14:59'
function calcTime(time, n) {}
 * Kiểm tra một chuỗi có phải đối xứng hay không (viết xuôi hay viết
ngược đều giống nhau, không phân biệt chữ hoa chữ thường và không tính
dấu cách)
 * @param {string} str - Chuỗi bất kỳ
 * @return {boolean} Kết quả
 * @example
 * isPalindrome('Race car'); // true, vì Race car = racecar = racecar
function isPalindrome(str) {}
 * Một con ốc sên leo từ đáy giếng lên miệng giếng, biết ban ngày leo
được x mét, ban đêm lại bị tụt xuống y mét, hỏi sau bao nhiều ngày thì
ốc sên sẽ leo lên tới miệng giếng
 * @param {number} h - Chiều cao của giếng (h > 0)
 * @param {number} x - Số mét leo được vào ban ngày (x > y)
 * @param {number} y - Số mét bị tụt xuống vào ban đêm (y > 0)
 * @return {number} Số ngày cần để ốc sên leo lên được miệng giếng
```

```
* @example
 * snail(10, 3, 1); // 5
function snail(h, x, y) {}
 * Sắp xếp các chữ số trong một số nguyên dương bất kỳ để tạo ra số nhỏ
nhất (giữ nguyên số chữ số ban đầu, bao gồm cả số 0)
 * @param {number} n - Số nguyên dương bất kỳ
 * @return {number} Số nhất nhất sau khi sắp xếp các chữ số
 * @example
 * sortNumber(5307510); // 1003557
function sortNumber(n) {}
 * Đếm số lần xuất hiện của mỗi phần tử trong mảng, nếu là chuỗi thì
không phân biệt chữ hoa chữ thường
 * @param {any[]} arr - Mång chứa giá trị bất kỳ
 * @return {Record<string, number>} object chứa kết quả, với key là giá
trị trong mảng và value là số lần xuất hiện trong mảng
 * @example
 * let result = countElement([1, true, 1, 1, true]);
 * console.log(result); // {1: 3, true: 2}
 * @example
 * let result = countElement(['Ba', 'Béo', 'Ba']);
 * console.log(result); // {ba: 2, béo: 1}
```

```
function countElement(arr) {}
 * Object chứa thông tin về sinh viên
 * @typedef {Object} Student
 * @property {number} id - ID duy nhất của sinh viên
 * @property {string} name - Tên sinh viên
 * @property {string} dateOfBirth - Ngày sinh
 * @property {number} grade - Điểm tốt nghiệp
 * Sắp xếp danh sách sinh viên dưa theo:
 * - Điểm tốt nghiệp theo thứ tự giảm dần
 * - Nếu sinh viên có điểm tốt nghiệp bằng nhau thì sắp xếp theo thứ tự
ngày sinh tăng dần
 * - Nếu ngày sinh cũng giống nhau thì sắp xếp theo tên (bảng chữ cái)
 * Kết quả trả về phải là một mảng mới, không thay đổi mảng cũ
 * @param {Student[]} students - Mång chứa thông tin sinh viên
 * @return {Student[]} Danh sách sinh viên sau khi sắp xếp
 * @example
 * let students = [
 * { id: 1, name: "Ba Nguyễn", dateOfBirth: '1992-05-24', point: 10.0
 * { id: 2, name: "Tuấn Anh", dateOfBirth: '1993-10-29', point: 4.5 },
 * { id: 3, name: "Tuấn Vũ", dateOfBirth: '1996-03-13', point: 8.2 },
   { id: 4, name: "Minh Vân", dateOfBirth: '1996-03-31', point: 7.0 },
 * { id: 5, name: "Phương Thảo", dateOfBirth: '1999-05-15', point: 9.0
* { id: 6, name: "Mai Anh", dateOfBirth: '1994-08-22', point: 4.5 },
 * { id: 7, name: "Thanh Hà", dateOfBirth: '1996-03-31', point: 7.0 },
```

```
* { id: 8, name: "Hoàng Linh", dateOfBirth: '1991-07-01', point: 7.2
 * { id: 9, name: "Lan Nhi", dateOfBirth: '1990-09-10', point: 5.5 },
 * sortStudents(students);
 * // Kết quả
 * // sắp xếp theo điểm
 * { id: 1, name: "Ba Nguyễn", dateOfBirth: '1992-05-24', point: 10.0
},
* { id: 5, name: "Phương Thảo", dateOfBirth: '1999-05-15', point: 9.0
},
 * { id: 3, name: "Tuấn Vũ", dateOfBirth: '1996-03-13', point: 8.2 },
 * { id: 8, name: "Hoàng Linh", dateOfBirth: '1991-07-01', point: 7.2
 * // điểm bằng nhau, sắp xếp theo ngày sinh
 * { id: 7, name: "Thanh Hà", dateOfBirth: '1996-02-16', point: 7.0 },
 * { id: 4, name: "Minh Vân", dateOfBirth: '1993-03-31', point: 7.0 },
 * // sắp xếp theo điểm
 * { id: 9, name: "Lan Nhi", dateOfBirth: '1990-09-10', point: 5.5 },
 * // điểm bằng nhau, ngày sinh giống nhau, sắp xếp theo tên
 * { id: 6, name: "Mai Anh", dateOfBirth: '1996-03-31', point: 4.5 },
* { id: 2, name: "Tuấn Anh", dateOfBirth: '1996-03-31', point: 4.5 },
function sortStudents(students) {}
```

