



JONAS.IO
SCHMEDTMANN

Thử thách viết mã cho tất cả các phần

Hoàn chỉnh
Khóa học JavaScript



@jonasschmedtman

JS

Mục lục

Hướng dẫn.....	4
Khái niệm cơ bản về JavaScript - Phần 1.....	5
Thử thách viết mã số 1.....	5
Thử thách viết mã số 2.....	6
Thử thách viết mã số 3	7
Thử thách viết mã số 4	8
Khái niệm cơ bản về JavaScript - Phần 2.....	9
Thử thách viết mã số 1.....	9
Thử thách viết mã số 2.....	10
Thử thách viết mã số 3	11
Thử thách viết mã số 4	12
Kỹ năng dành cho nhà phát triển và thiết lập trình chỉnh sửa.....	13
Thử thách viết mã số 1.....	13
JavaScript trong trình duyệt: DOM và sự kiện.....	14
Thử thách viết mã số 1.....	14
Cấu trúc dữ liệu, toán tử hiện đại và chuỗi.....	15
Thử thách viết mã số 1.....	15
Thử thách viết mã số 2.....	17
Thử thách viết mã số 3	18
Thử thách viết mã số 4	19
Xem xét kỹ hơn về chức năng.....	20
Thử thách viết mã số 1.....	20
Thử thách viết mã số 2.....	22

Làm việc với mảng.....	23
Thử thách viết mã số 1.....	23
Thử thách viết mã số 2.....	24
Thử thách viết mã số 3	24
Thử thách viết mã số 4	25
Lập trình hướng đối tượng (OOP)	27
Thử thách viết mã số 1.....	27
Thử thách viết mã số 2.....	28
Thử thách viết mã số 3	28
Thử thách viết mã số 4	29
JavaScript không đồng bộ.....	30
Thử thách viết mã số 1.....	30
Thử thách viết mã số 2.....	32
Thử thách viết mã số 3	33

Hướng dẫn

§ Khóa học JavaScript hoàn chỉnh chứa hơn 25 thử thách viết mã để bạn tự mình thực hành các khái niệm chúng tôi học trong mỗi phần. Đây là một phần cơ bản trong hành trình học tập của bạn 🧐

§ Tài liệu này chứa tất cả các thử thách viết mã trong khóa học này để bạn có thể đọc văn bản thử thách mà không xem video (tôi vẫn khuyên bạn nên xem video vì tôi có thể cung cấp thêm một số manh mối)

§ Một số thử thách yêu cầu một số mã khởi đầu mà bạn có thể lấy từ tài liệu này thay vì sao chép nó từ video

§ Giải pháp cho từng thử thách đều có trong video nên bạn vẫn phải đi xem chúng 😊

§ Một số thử thách sau này có mục đích khá khó khăn (dù sao thì chúng cũng được gọi là thử thách phải không?). 😓 Nếu một trong số đó quá khó, đừng lo lắng, điều này là hoàn toàn bình thường! Chỉ cần bắt đầu xem giải pháp, sau đó tạm dừng video và tiếp tục tự làm việc. Đấu tranh là một phần của cuộc hành trình, nhưng bạn có thể làm được! 🚀

Nguyên tắc cơ bản về JavaScript - Phần 1

Thử thách viết mã số 1

Mark và John đang cố gắng so sánh chỉ số BMI (Chỉ số khối cơ thể) của họ, được tính bằng công thức:

$$\text{BMI} = \text{khối lượng} / \text{chiều cao}^2 = \text{khối lượng} / (\text{chiều cao} * \text{chiều cao})$$
 (khối lượng tính bằng kg và chiều cao tính bằng mét).

Nhiệm vụ của bạn:

1. Lưu trữ khối lượng và chiều cao của Mark và John theo các biến
2. Tính chỉ số BMI của cả hai bằng công thức (thậm chí bạn có thể tính cả hai phiên bản)
3. Tạo biến Boolean 'markHigherBMI' chứa thông tin về liệu Mark có chỉ số BMI cao hơn John hay không.

Dữ liệu thử nghiệm:

§ Dữ liệu 1: Mark nặng 78 kg và cao 1,69 m. John nặng 92 kg và cao 1,95 m.

§ Dữ liệu 2: Mark nặng 95 kg và cao 1,88 m. John nặng 85 kg và cao 1,76 m.

CHÚC MAY MẮN 😊

Thử thách viết mã số 2

Sử dụng ví dụ về BMI từ Thử thách số 1 và đoạn mã bạn đã viết rồi cải thiện nó.

Nhiệm vụ của bạn:

1. In một kết quả đẹp ra bằng điều khiển, cho biết ai có chỉ số BMI cao hơn. Thông báo có nội dung "Chỉ số BMI của Mark cao hơn John!" hoặc "Chỉ số BMI của John cao hơn Mark!"
2. Sử dụng chữ mẫu để bao gồm các giá trị BMI trong kết quả đầu ra. Ví dụ: "Mark BMI (28,3) cao hơn John (23,9)!"


Gợi ý: Sử dụng câu lệnh if/else 🤔

CHÚC MAY MẮN 😊

Thử thách viết mã số 3

Có hai đội thể dục dụng cụ là Dolphins và Koalas. Họ thi đấu với nhau 3 lần. Người chiến thắng có số điểm trung bình cao nhất sẽ giành được một chiếc cúp!

Nhiệm vụ của bạn:


1. Tính điểm trung bình của mỗi đội, sử dụng dữ liệu kiểm tra bên dưới
2. So sánh điểm trung bình của đội để xác định đội chiến thắng trong cuộc thi và in nó ra bảng điều khiển. Đừng quên rằng có thể có kết quả hòa, vì vậy hãy kiểm tra điều đó (hòa nghĩa là họ có cùng điểm trung bình)
3. Phần thưởng 1: Bao gồm yêu cầu đạt điểm tối thiểu là 100. Với quy tắc này, đội chỉ thắng nếu có số điểm cao hơn đội kia, đồng thời đạt ít nhất 100 điểm. Gợi ý: Sử dụng toán tử logic để kiểm tra điểm tối thiểu cũng như nhiều khối khác nếu 
4. Phần thưởng 2: Điểm tối thiểu cũng áp dụng cho trận hòa! Vì vậy kết quả hòa chỉ xảy ra khi cả hai đội có cùng số điểm và đều có số điểm lớn hơn hoặc bằng 100 điểm. Nếu không thì không có đội nào giành được cúp

Dữ liệu thử nghiệm:

§ Dữ liệu 1: Cá heo đạt điểm 96, 108 và 89. Gấu túi đạt điểm 88, 91 và 110

§ Dữ liệu thử nghiệm 1: Cá heo đạt điểm 97, 112 và 101. Gấu túi đạt điểm 109, 95 và 123

§ Dữ liệu thử nghiệm 2: Cá heo đạt điểm 97, 112 và 101. Gấu túi đạt điểm 109, 95 và 106

CHÚC MAY MẮN 

Thử thách viết mã số 4

Steven muốn xây dựng một công cụ tính tiền boa rất đơn giản mỗi khi anh ấy đi ăn ở nhà hàng. Ở đất nước của anh ấy, người ta thường boa 15% nếu giá trị hóa đơn nằm trong khoảng từ 50 đến 300. Nếu giá trị khác nhau, tiền boa là 20%.

Nhiệm vụ của bạn:

1. Tính tiền boa tùy vào giá trị hóa đơn. Tạo một biến gọi là 'tip' cho việc này. Không được phép sử dụng câu lệnh if/else (Nếu dễ hơn cho bạn, bạn có thể bắt đầu bằng câu lệnh if/else, sau đó thử chuyển đổi nó thành toán tử ba ngôi!)
2. In một chuỗi ra bảng điều khiển chứa giá trị hóa đơn, tiền boa và giá trị cuối cùng (hóa đơn + tiền boa). Ví dụ: "Hóa đơn là 275, tiền boa là 41,25 và tổng giá trị là 316,25"

Dữ liệu thử nghiệm:

§ Dữ liệu 1: Kiểm tra giá trị hóa đơn 275, 40 và 430

Gợi ý:

§ Để tính 20% của một giá trị, chỉ cần nhân nó với $20/100 = 0,2$

§ Giá trị X nằm trong khoảng từ 50 đến 300, nếu nó $\geq 50 \ \&\& \leq 300$



CHÚC MAY MẮN 😊

Nguyên tắc cơ bản về JavaScript - Phần 2

Thử thách viết mã số 1

Trở lại với hai đội thể dục dụng cụ Cá heo và Gấu túi! Có một bộ môn thể dục dụng cụ mới hoạt động khác hẳn.

Mỗi đội thi đấu 3 lần, sau đó tính trung bình cộng của 3 điểm (do đó mỗi đội sẽ có một điểm trung bình).

Một đội chỉ thắng nếu có ít nhất gấp đôi số điểm trung bình của đội kia.

Nếu không, không có đội nào thắng!

Nhiệm vụ của bạn:

1. Tạo hàm mũi tên "calcAverage" để tính trung bình cộng của 3 điểm
2. Sử dụng hàm tính điểm trung bình cho cả 2 đội
3. Tạo hàm 'checkWinner' lấy điểm trung bình của mỗi đội làm tham số ('avgDolphins' và 'avgKoalas'), sau đó ghi ngư ời chiến thắng vào bảng điều khiển, cùng với điểm chiến thắng, theo quy tắc ở trên.

Ví dụ: "Koala thắng (30 vs. 13)"

4. Sử dụng chức năng 'checkWinner' để xác định ngư ời chiến thắng cho cả Dữ liệu 1 và Dữ liệu 2
5. Bỏ qua lần rút thăm lần này

Dữ liệu thử nghiệm:

§ Dữ liệu 1: Cá heo đạt điểm 44, 23 và 71. Gấu túi đạt điểm 65, 54 và 49

§ Dữ liệu 2: Cá heo đạt điểm 85, 54 và 41. Gấu túi đạt điểm 23, 34 và 27

Gợi ý:

§ Để tính trung bình cộng của 3 giá trị, cộng tất cả lại với nhau rồi chia cho 3

§ Để kiểm tra xem số A có ít nhất gấp đôi số B hay không, hãy kiểm tra $A \geq 2 * B$.

Áp dụng điều này cho điểm trung bình của đội 🤔

CHÚC MAY MẮN 😊

Thử thách viết mã số 2

Steven vẫn đang xây dựng công cụ tính tiền boa của mình, sử dụng các quy tắc tư duy tự như trước: Tiền boa 15% hóa đơn nếu giá trị hóa đơn nằm trong khoảng từ 50 đến 300 và nếu giá trị khác nhau thì tiền boa là 20%.

Nhiệm vụ của bạn:

1. Viết hàm 'calcTip' lấy bất kỳ giá trị hóa đơn nào làm đầu vào và trả về tiền boa tương ứng, được tính toán dựa trên các quy tắc ở trên (bạn có thể kiểm tra mã tử thử thách tính tiền boa đầu tiên nếu cần). Sử dụng loại chức năng bạn thích nhất. Kiểm tra hàm bằng cách sử dụng giá trị hóa đơn là 100
2. Và bây giờ hãy sử dụng mảng! Vì vậy hãy tạo một mảng "hóa đơn" chứa dữ liệu thử nghiệm dư ới
3. Tạo mảng 'tips' chứa giá trị tip cho mỗi hóa đơn, được tính từ hàm bạn đã tạo trước đó
4. Phần thứ 3: Tạo một mảng 'total' chứa các giá trị tổng, do đó hóa đơn + tiền boa

Dữ liệu thử nghiệm: 125, 555 và 44

Gợi ý: Hãy nhớ rằng một mảng cần một giá trị ở mỗi vị trí và giá trị đó thực sự có thể là giá trị được trả về của hàm! Vì vậy, bạn chỉ có thể gọi một hàm dư ới dạng giá trị mảng (vì vậy trước tiên đừng lưu trữ các giá trị tiền boa trong các biến riêng biệt mà ngay trong mảng mới)



CHÚC MAY MẮN



Thử thách viết mã số 3

Hãy quay lại với Mark và John so sánh chỉ số BMI của họ! Lần này, hãy sử dụng các đối tượng để thực hiện các phép tính! Ghi nhớ: $BMI = \text{khối lượng} / \text{chiều cao}^2 = \text{khối lượng} / (\text{chiều cao} * \text{chiều cao})$ (khối lượng tính bằng kg và chiều cao tính bằng mét)

Nhiệm vụ của bạn:

1. Đối với mỗi người trong số họ, hãy tạo một đối tượng có các thuộc tính về tên đầy đủ, khối lượng và chiều cao của chúng (Mark Miller và John Smith)
2. Tạo phương pháp 'calcBMI' trên từng đối tượng để tính chỉ số BMI (phương pháp tư duy tự trên cả hai đối tượng). Lưu trữ giá trị BMI vào một thuộc tính và cũng trả về giá trị đó từ phương thức
3. Đăng nhập vào bảng điều khiển của người có chỉ số BMI cao hơn, kèm theo tên đầy đủ và chỉ số BMI tương ứng. Ví dụ: "Chỉ số BMI của John (28,3) cao hơn của Mark (23,9)!"

Dữ liệu thử nghiệm: Mark nặng 78 kg và cao 1,69 m. John nặng 92 kg và cao 1,95 m.

CHÚC MAY MẮN 😊

Thử thách viết mã số 4

Hãy cải tiến công cụ tính tiền boa của Steven hơn nữa, lần này bằng cách sử dụng vòng lặp!

Nhiệm vụ của bạn:

1. Tạo một mảng "hóa đơn" chứa tất cả 10 giá trị hóa đơn thử nghiệm
2. Tạo các mảng trống cho số tiền boa và tổng số ('mẹo' và 'tổng số')
3. Sử dụng hàm 'calcTip' mà chúng tôi đã viết trước đây (không cần lặp lại) để tính tiền boa và tổng giá trị (hóa đơn + tiền boa) cho mỗi giá trị hóa đơn trong mảng hóa đơn. Sử dụng cho vòng lặp để thực hiện 10 phép tính!

Dữ liệu thử nghiệm: 22, 295, 176, 440, 37, 105, 10, 1100, 86 và 52

Gợi ý: Gọi 'calcTip' trong vòng lặp và sử dụng phương thức đẩy để thêm giá trị vào mảng mẹo và tổng 😊

Thứ ờng:

4. Phần thứ ờng: Viết hàm 'calcAverage' lấy một mảng có tên là 'arr' làm đối số. Hàm này tính trung bình cộng của tất cả các số trong mảng đã cho. Đây là một thử thách khó khăn (chúng tôi chưa từng làm điều này trước đây)! Đây là cách giải quyết nó:

4.1. Đầu tiên, bạn sẽ cần cộng tất cả các giá trị trong mảng. Để thực hiện phép cộng, hãy bắt đầu bằng cách tạo một biến 'tổng' bắt đầu từ 0. Sau đó lặp qua mảng bằng vòng lặp for. Trong mỗi lần lặp, hãy thêm giá trị hiện tại vào biến 'tổng'. Bằng cách này, đến cuối vòng lặp, bạn có tất cả các giá trị được cộng lại với nhau

4.2. Để tính giá trị trung bình, hãy chia tổng bạn đã tính trước cho độ dài của mảng (vì đó là số phần tử)

4.3. Gọi hàm với mảng 'tổng'

CHÚC MAY MẮN 😊

Kỹ năng dành cho nhà phát triển và thiết lập biên tập viên

Thử thách viết mã số 1

Với một loạt các nhiệt độ tối đa được dự báo, nhiệt kế sẽ hiển thị một chuỗi có nhiệt độ nhất định.

Ví dụ: [17, 21, 23] sẽ in "... 17°C trong 1 ngày ... 21°C trong 2 ngày ... 23°C trong 3 ngày ..."

Nhiệm vụ của bạn:

1. Tạo hàm 'printForecast' nhận một mảng 'arr' và ghi lại một chuỗi như trên vào bảng điều khiển. Hãy thử nó với cả hai tập dữ liệu thử nghiệm.
2. Sử dụng khuôn khổ giải quyết vấn đề: Hiểu vấn đề và chia nhỏ nó vào các vấn đề phụ!

Dữ liệu thử nghiệm:

§ Dữ liệu 1: [17, 21, 23]

§ Dữ liệu 2: [12, 5, -5, 0, 4]

CHÚC MAY MẮN 😊

JavaScript trong trình duyệt: DOM và sự kiện

Thử thách viết mã số 1

Triển khai chức năng tạm dừng trò chơi để người chơi có thể đưa ra dự đoán mới!

Nhiệm vụ của bạn:

1. Chọn phần tử có lớp 'lại' và đính kèm trình xử lý sự kiện nhấp chuột
2. Trong hàm xử lý, khôi phục các giá trị ban đầu của 'điểm' và biến 'số bí mật'
3. Khôi phục các điều kiện ban đầu của tin nhắn, số, điểm và đoán lĩnh vực
4. Đồng thời khôi phục màu nền ban đầu (#222) và chiều rộng số (15rem)

CHÚC MAY MẮN 😊

Cấu trúc dữ liệu, toán tử hiện đại và chuỗi

Thử thách viết mã số 1

Chúng tôi đang xây dựng một ứng dụng cá cược bóng đá (bóng đá cho những người bạn Mỹ của tôi 😊)!

Giả sử chúng ta lấy dữ liệu từ một dịch vụ web về một trò chơi nhất định (biến 'trò chơi' ở trang tiếp theo). Trong thử thách này, chúng ta sẽ làm việc với dữ liệu đó.

Nhiệm vụ của bạn:

1. Tạo một mảng người chơi cho mỗi đội (biến 'players1' và 'players2')
2. Cầu thủ đầu tiên trong dàn cầu thủ là thủ môn, những người còn lại là cầu thủ trên sân.
Đối với Bayern Munich (đội 1), tạo một biến ('gk') với tên thủ môn và một mảng ('fieldPlayers') với tất cả 10 cầu thủ còn lại
3. Tạo một mảng 'allPlayers' chứa tất cả người chơi của cả hai đội (22 người chơi)
4. Trong trận đấu, Bayern Munich (đội 1) sử dụng 3 cầu thủ dự bị. Vì vậy, hãy tạo một mảng mới ('players1Final') chứa tất cả các cầu thủ ban đầu của team1 cộng với 'Thiago', 'Coutinho' và 'Perisic'
5. Dựa trên đối tượng game.odds, tạo một biến cho mỗi số lẻ (được gọi là 'team1', 'draw' và 'team2')
6. Viết hàm ('printGoals') nhận số lượng tên cầu thủ tùy ý (không phải mảng) và in từng tên đó ra bảng điều khiển, cùng với tổng số bàn thắng được ghi (số tên cầu thủ được ghi vào)
7. Đội nào có tỷ lệ cược thấp hơn sẽ có nhiều khả năng thắng hơn. In ra bảng điều khiển đội có nhiều khả năng giành chiến thắng hơn mà không cần sử dụng câu lệnh if/else hoặc toán tử ba ngôi.

Dữ liệu thử nghiệm cho 6.: Đầu tiên, sử dụng các cầu thủ 'Davies', 'Muller', 'Lewandowski' và 'Kimmich'.

Sau đó, gọi lại hàm với người chơi từ game.scored

CHÚC MAY MẮN 😊

```

const game =
  { team1: 'Bayern Munich', team2:
    'Borussia Dortmund', cầu thủ: [ [

      'Neuer',
      'Pavard',
      'Martinez',
      'Alaba',
      'Davies',
      'Kimmich',
      'Goretzka',
      'Coman',
      'Muller',
      'Gnarby',
      'Lewandowski', ],

    [

      'Burki',
      'Schulz',
      'Hummels',
      'Akanji',
      'Hakimi',
      'Weigl',
      'Witsel',
      'Hazard',
      'Brandt',
      'Sancho',

      'Gotze', ], ],
  điểm: '4:0', ghi bàn: ['Lewandowski', 'Gnarby', 'Lewandowski',
    'Hummels'],
  ngày: '9/11/2037', tỷ lệ cược:
  { team1:
    1.33, x: 3.25,
    team2:
    6.5, }, };

```


Thử thách viết mã số 2

Hãy tiếp tục với ứng dụng cá cược bóng đá của chúng tôi! Tiếp tục sử dụng biến 'trò chơi' từ trước.

Nhiệm vụ của bạn:

1. Lặp lại mảng `game.scored` và in tên từng người chơi lên bảng điều khiển, cùng với số bàn thắng (Ví dụ: "Bàn thắng 1: Lewandowski")
2. Sử dụng vòng lặp để tính số lẻ trung bình và ghi nó vào bảng điều khiển (Chúng tôi đã nghiên cứu cách tính số trung bình, bạn có thể kiểm tra nếu không nhớ)
3. In 3 tỷ lệ cược ra bảng điều khiển, nhưng theo cách có định dạng đẹp mắt, chính xác như thế này:

Tỷ lệ thắng Bayern Munich: 1.33

Tỷ lệ hòa: 3.25

Tỷ lệ thắng Borussia Dortmund: 6.5

Lấy tên đội trực tiếp từ đối tượng trò chơi, không mã hóa chúng (ngoại trừ "hòa"). Gợi ý: Lưu ý tỷ lệ cược và đối tượng trò chơi có tác dụng như thế nào cùng tên thuộc tính 4. 🤔

Phần thưởng: Tạo một đối tượng gọi là 'người ghi điểm' chứa tên của cầu thủ ghi bàn là thuộc tính và số bàn thắng là giá trị. Trong trò chơi này, nó sẽ trông như thế này:


```
{
  Gnarby: 1,
  Hummel: 1,
  Lewandowski: 2
}
```

CHÚC MAY MẮN 😊






Thử thách viết mã số 3

Hãy tiếp tục với ứng dụng cá cược bóng đá của chúng tôi! Lần này, chúng ta có một bản đồ tên là 'gameEvents' (xem bên dưới) với nhật ký các sự kiện đã xảy ra trong trò chơi. Các giá trị chính là các sự kiện và khóa là số phút mà mỗi sự kiện xảy ra (một trận bóng đá có 90 phút cộng thêm một số thời gian bù giờ).

Nhiệm vụ của bạn:

1. Tạo một mảng 'sự kiện' gồm các sự kiện trò chơi khác nhau đã xảy ra (không có trùng lặp)
2. Sau khi trận đấu kết thúc, người ta nhận thấy thẻ vàng từ phút 64 thật không công bằng. Vì vậy hãy xóa sự kiện này khỏi nhật ký sự kiện trò chơi.
3. Tính toán và ghi chuỗi sau vào bảng điều khiển: "Trung bình cứ 9 phút lại có một sự kiện xảy ra" (hãy nhớ rằng một trò chơi có 90 phút)
4. Lặp lại 'gameEvents' và ghi từng thành phần vào bảng điều khiển, đánh dấu cho dù đó là hiệp một hay hiệp hai (sau 45 phút) của trận đấu, như thế này:
[NỬA ĐẦU] 17:  MỤC TIÊU

CHÚC MAY MẮN 😊

```
const gameEvents = Bản đồ mới ([
  [17,  MỤC TIÊU],
  [36, 'Tuyển thế'],
  [47,  MỤC TIÊU],
  [61, 'Tuyển thế'],
  [64, 'Thẻ vàng'],
  [69,  Thẻ đỏ],
  [70, 'Tuyển thế'],
  [72, 'Tuyển thế'],
  [76,  MỤC TIÊU],
  [80,  MỤC TIÊU],
  [92, 'Thẻ vàng'],
]);
```

Thử thách viết mã số 4

Viết chương trình nhận danh sách các tên biến được viết bằng gạch dưới_case và chuyển đổi chúng thành CamelCase.

Đầu vào sẽ đến từ một vùng văn bản được chèn vào DOM (xem mã bên dưới để chèn các phần tử) và quá trình chuyển đổi sẽ xảy ra khi nhấn nút.

Dữ liệu thử nghiệm (được dán vào vùng văn bản, bao gồm cả dấu cách):

gạch dưới_case

tên_đầu tiên

Một số_Biến

tính_AGE

bị trì hoãn_khởi hành

Nên tạo đầu ra này (5 đầu ra console.log riêng biệt):

gạch dướiCase



tên đầu tiên



một



tính Tuổi bị trì



hoãnKhởi hành



Gợi ý:

§ Ghi nhớ ký tự nào định nghĩa một dòng mới trong vùng văn bản



§ Giải pháp chỉ cần áp dụng cho biến có 2 từ, như a_b

§ Bắt đầu mà không cần lo lắng về Chỉ giải quyết vấn đề đó sau khi bạn có biến chuyển đổi tên đang hoạt động 🤔

§ Thử thách này có mục đích khó khăn, vì vậy hãy bắt đầu xem giải pháp trong trường hợp bạn đang bị mắc kẹt. Sau đó tạm dừng và tiếp tục!

Sau đó, hãy kiểm tra bằng dữ liệu kiểm tra của riêng bạn!

CHÚC MAY MẮN



```
document.body.append(document.createElement('textarea'));
document.body.append(document.createElement('button'));
```

Một cái nhìn cận cảnh hơn về các chức năng

Thử thách viết mã số 1

Hãy xây dựng một ứng dụng thăm dò đơn giản!

Cuộc thăm dò ý kiến có một câu hỏi, một loạt các tùy chọn mà mọi người có thể chọn và một mảng có số lượng câu trả lời cho mỗi tùy chọn. Dữ liệu này được lưu trữ trong đối tượng 'thăm dò ý kiến' ban đầu bên dưới.

Nhiệm vụ của bạn:

- Tạo một phương thức có tên 'registerNewAnswer' trên đối tượng 'thăm dò ý kiến'. Phương pháp này thực hiện 2 việc:
 - Hiển thị cửa sổ nhắc người dùng nhập số tùy chọn đã chọn. Lời nhắc sẽ trông như thế này:


```
Ngôn ngữ lập trình yêu thích của bạn là gì?
0: JavaScript
1: Trăn
2: Rì sét
3: C++
(Viết số tùy chọn)
```
 - Dựa trên số đầu vào, hãy cập nhật thuộc tính mảng 'câu trả lời'. Ví dụ: nếu tùy chọn là 3, hãy tăng giá trị ở vị trí 3 của mảng lên 1. Đảm bảo kiểm tra xem đầu vào có phải là số không và số đó có hợp lý không (ví dụ: câu trả lời 52 sẽ không có ý nghĩa, phải không?)
- Gọi phương thức này bất cứ khi nào người dùng nhấp vào nút "Trả lời cuộc thăm dò ý kiến".
- Tạo phương thức 'displayResults' để hiển thị kết quả thăm dò ý kiến. các phương thức lấy một chuỗi làm đầu vào (được gọi là 'loại'), có thể là 'chuỗi' hoặc 'mảng'. Nếu loại là 'mảng', chỉ cần hiển thị mảng kết quả như cũ bằng cách sử dụng console.log(). Đây phải là tùy chọn mặc định. Nếu loại là 'chuỗi', hãy hiển thị chuỗi như "Kết quả thăm dò ý kiến là 13, 2, 4, 1".
- Chạy phương thức 'displayResults' ở cuối mỗi cuộc gọi phương thức 'registerNewAnswer'.
- Phần thử nghiệm: Sử dụng phương thức 'displayResults' để hiển thị 2 mảng trong dữ liệu thử nghiệm. Sử dụng cả tùy chọn 'mảng' và 'chuỗi'. Đừng đặt các mảng vào đối tượng thăm dò ý kiến! Vậy từ khóa this sẽ trông như thế nào trong tình huống này?

Dữ liệu thử nghiệm để nhận tiền thưởng:

§ Dữ liệu 1: [5, 2, 3]

§ Dữ liệu 2: [1, 5, 3, 9, 6, 1]

Gợi ý: Sử dụng nhiều công cụ bạn đã học trong phần này và phần trước 😊

CHÚC MAY MẮN 😊

```
cuộc thăm dò const = {  
  câu hỏi: "Ngôn ngữ lập trình yêu thích của bạn là gì?",  
  tùy chọn: ["0: JavaScript", "1: Python", "2: Rust", "3: C++"],  
  
  // Điều này tạo ra [0, 0, 0, 0]. Thêm trong phần tiếp theo!  
  câu trả lời: Mảng mới (4).fill(0),  
};
```

Thử thách viết mã số 2

Đây là một thử thách về tư duy hơn là một thử thách viết mã



Nhiệm vụ của bạn:

1. Lấy IIFE bên dưới và ở cuối hàm, đính kèm một trình lắng nghe sự kiện để thay đổi màu của phần tử h1 đã chọn ('tiêu đề') thành màu xanh lam, mỗi khi phần tử nội dung được nhấp vào. Đừng chọn lại phần tử h1!
2. Và bây giờ hãy giải thích cho chính bạn (hoặc ai đó xung quanh bạn) tại sao điều này lại hiệu quả! Hãy dành tất cả thời gian bạn cần. Hãy suy nghĩ về thời điểm chính xác hàm gọi lại được thực thi và điều đó có ý nghĩa gì đối với các biến liên quan đến ví dụ này.

```
(chức năng () {  
    tiêu đề const = document.querySelector('h1');  
    header.style.color = 'đỏ';  
})();
```

CHÚC MAY MẮN



Làm việc với mảng

Thử thách viết mã số 1

Julia và Kate đang nghiên cứu về loài chó. Vì vậy, mỗi người trong số họ đã hỏi 5 người nuôi chó về tuổi con chó của họ và lưu trữ dữ liệu vào một mảng (mỗi mảng một mảng). Hiện tại, họ chỉ quan tâm đến việc biết một con chó là chó trưởng thành hay chó con.

Một con chó được coi là trưởng thành nếu nó ít nhất 3 tuổi và là chó con nếu nó dưới 3 tuổi.

Nhiệm vụ của bạn:

Tạo hàm 'checkDogs', chấp nhận 2 mảng độ tuổi của chó ('dogsJulia' và 'dogsKate') và thực hiện những việc sau:

1. Julia phát hiện ra rằng chủ nhân của con chó đầu tiên và hai con chó cuối cùng thực sự nuôi mèo chứ không phải chó! Vì vậy, hãy tạo một bản sao nông của mảng của Julia và xóa cat age khỏi mảng được sao chép đó (vì việc thay đổi các tham số hàm là một cách làm không tốt)
2. Tạo một mảng có cả dữ liệu của Julia (đã sửa) và của Kate
3. Đối với mỗi con chó còn lại, hãy đăng nhập vào bảng điều khiển dù đó là con trưởng thành ("Con chó số 1 đã trưởng thành và 5 tuổi") hay chó con ("Con chó số 2 vẫn là con chó con 🐶")
4. Chạy hàm cho cả hai tập dữ liệu thử nghiệm

Dữ liệu thử nghiệm:

§ Dữ liệu 1: Dữ liệu của Julia [3, 5, 2, 12, 7], Dữ liệu của Kate [4, 1, 15, 8, 3]

§ Dữ liệu 2: Dữ liệu của Julia [9, 16, 6, 8, 3], Dữ liệu của Kate [10, 5, 6, 1, 4]

Gợi ý: Sử dụng các công cụ từ tất cả các bài giảng trong phần này cho đến nay 🤔

CHÚC MAY MẮN 😊

Thử thách viết mã số 2

Chúng ta hãy quay trở lại nghiên cứu của Julia và Kate về loài chó. Lần này, họ muốn chuyển đổi tuổi của chó sang tuổi của con người và tính tuổi trung bình của những con chó trong nghiên cứu của họ.

Nhiệm vụ của bạn:

Tạo hàm 'calcAverageHumanAge', chấp nhận một mảng độ tuổi của chó ('ages') và thực hiện những việc sau theo thứ tự:

1. Tính tuổi chó theo năm của con người bằng công thức sau: nếu chó ≤ 2 tuổi thì $\text{humanAge} = 2 * \text{dogAge}$. Nếu chó > 2 tuổi, $\text{humanAge} = 16 + \text{dogAge} * 4$
2. Loại trừ tất cả những con chó dưới 18 tuổi (giống như nuôi chó ít nhất 18 tuổi)
3. Tính tuổi trung bình của con người của tất cả những con chó trưởng thành (bạn nên biết từ những thách thức khác về cách chúng tôi tính toán mức trung bình)
4. Chạy hàm cho cả hai tập dữ liệu thử nghiệm

Dữ liệu thử nghiệm:

§ Dữ liệu 1: [5, 2, 4, 1, 15, 8, 3]

§ Dữ liệu 2: [16, 6, 10, 5, 6, 1, 4]

CHÚC MAY MẮN 😊

Thử thách viết mã số 3

Viết lại hàm 'calcAverageHumanAge' từ Thử thách #2, nhưng lần này dư ới dạng hàm mũi tên và sử dụng chuỗi!

Dữ liệu thử nghiệm:

§ Dữ liệu 1: [5, 2, 4, 1, 15, 8, 3]

§ Dữ liệu 2: [16, 6, 10, 5, 6, 1, 4]

CHÚC MAY MẮN 😊

Thử thách viết mã số 4

Julia và Kate vẫn đang nghiên cứu về chó và lần này họ đang nghiên cứu xem chó ăn quá nhiều hay quá ít.

Ăn quá nhiều nghĩa là khẩu phần ăn hiện tại của chó lớn hơn khẩu phần khuyến nghị, còn ăn quá ít thì ngược lại.

Ăn một lượng vừa phải có nghĩa là khẩu phần ăn hiện tại của chó nằm trong khoảng 10% trên và 10% dưới khẩu phần khuyến nghị (xem gợi ý).

Nhiệm vụ của bạn:

- Lập lại mảng 'dogs' chứa các đối tượng chó và đối với mỗi con chó, hãy tính phần thức ăn được đề xuất và thêm nó vào đối tượng làm thuộc tính mới. Không tạo mảng mới, chỉ cần lặp qua mảng đó. Formula: khuyến nghị Thực phẩm = trọng lượng ** 0,75 * 28. (Kết quả tính bằng gam thực phẩm và trọng lượng cần tính bằng kg)
- Tìm con chó của Sarah và đăng nhập vào bảng điều khiển xem nó có ăn quá nhiều hay không nhỏ bé. Gợi ý: Một số con chó có nhiều chủ, vì vậy trước tiên bạn cần tìm Sarah trong mảng chủ sở hữu, và do đó, việc này hơi phức tạp (có mục đích) 🤔
- Tạo một mảng chứa tất cả những người nuôi chó ăn quá nhiều ("ownersEatTooMuch") và một mảng chứa tất cả chủ sở hữu của những con chó ăn quá ít ("ownersEatTooLittle").
- Ghi một chuỗi vào bảng điều khiển cho mỗi mảng được tạo trong 3., như thế này: "Chó của Matilda và Alice và Bob ăn quá nhiều!" và "Chó của Sarah, John và Michael ăn quá ít!"
- Đăng nhập vào bảng điều khiển xem có con chó nào ăn đúng lượng thức ăn không điều đó được khuyến nghị (chỉ đúng hoặc sai)
- Đăng nhập vào bảng điều khiển xem có con chó nào ăn đủ lượng thức ăn hay không (chỉ đúng hoặc sai)
- Tạo một mảng chứa những con chó đang ăn một lượng thức ăn vừa phải (cố gắng sử dụng lại điều kiện được sử dụng trong 6.)
- Tạo một bản sao nông của mảng 'dogs' và sắp xếp nó theo phần thức ăn được đề xuất theo thứ tự tăng dần (hãy nhớ rằng các phần nằm bên trong các đối tượng của mảng)



Gợi ý:

§ Sử dụng nhiều công cụ khác nhau để giải quyết những thách thức này, bạn có thể sử dụng bản tóm tắt bài giảng để lựa chọn giữa chúng 😊

§ Nằm trong phạm vi 10% trên và dư ới phần đư ợc đề xuất có nghĩa là: hiện tại > (đư ợc khuyến nghị * 0,90) && hiện tại < (đư ợc khuyến nghị * 1,10). Về cơ bản, phần hiện tại phải nằm trong khoảng từ 90% đến 110% phần đư ợc đề xuất.

Dữ liệu thử nghiệm:

```
const chó = [  
  { weight: 22, curFood: 250, chủ sở hữu: ['Alice', 'Bob'] },  
  { weight: 8, curFood: 200, chủ sở hữu: ['Matilda'] },  
  { weight: 13, curFood: 275, chủ sở hữu: ['Sarah', 'John'] },  
  { trọng lượng: 32, curFood: 340, chủ sở hữu: ['Michael'] },  
];
```

CHÚC MAY MẮN 😊

Lập trình hướng đối tượng (OOP)

Thử thách viết mã số 1

Nhiệm vụ của bạn:

1. Sử dụng hàm tạo để triển khai 'Ô tô'. Một chiếc ô tô có thuộc tính 'nhãn hiệu' và 'tốc độ'. Thuộc tính 'tốc độ' là tốc độ hiện tại của ô tô tính bằng km/h
2. Triển khai phương pháp 'tăng tốc' để tăng tốc độ của ô tô lên 10 và ghi tốc độ mới vào bảng điều khiển
3. Thực hiện phương pháp 'phanh' sẽ giảm tốc độ của ô tô xuống 5 và ghi lại tốc độ mới cho bảng điều khiển
4. Tạo 2 đối tượng 'Ô tô' và thử nghiệm cách gọi 'tăng tốc' và 'phanh' nhiều lần trên mỗi chiếc

Dữ liệu thử nghiệm:

§ Xe dữ liệu 1: 'BMW' đang chạy với tốc độ 120 km/h

§ Xe dữ liệu 2: 'Mercedes' đang chạy với tốc độ 95 km/h

CHÚC MAY MẮN



Thử thách viết mã số 2

Nhiệm vụ của bạn:

1. Tạo lại Thử thách số 1, nhưng lần này sử dụng lớp ES6 (gọi nó là 'CarCl')
2. Thêm một getter có tên 'speedUS' để trả về tốc độ hiện tại tính bằng mi/h (chia bằng 1,6)
3. Thêm một trình cài đặt có tên 'speedUS' để đặt tốc độ hiện tại tính bằng mi/h (nhưng chuyển đổi nó thành km/h trước khi lưu giá trị, bằng cách nhân đầu vào với 1,6)
4. Tạo một chiếc ô tô mới và thử nghiệm tính năng 'tăng tốc' và 'phanh' các phương thức và với getter và setter.

Dữ liệu thử nghiệm:

5 Xe dữ liệu 1: 'Ford' đang chạy với tốc độ 120 km/h

CHÚC MAY MẮN



Thử thách viết mã số 3

Nhiệm vụ của bạn:

1. Sử dụng hàm khởi tạo để triển khai Xe điện (được gọi là 'EV') khi còn nhỏ "đăng cấp" của 'Ô tô'. Bên cạnh nhãn hiệu và tốc độ hiện tại, 'EV' còn có mức sạc pin hiện tại tính bằng % (thuộc tính 'charge')
2. Triển khai phương thức 'chargeBattery' lấy đối số 'chargeTo' và đặt mức sạc pin thành 'chargeTo'
3. Thực hiện phương pháp 'tăng tốc' để tăng tốc độ của ô tô lên 20, và giảm phí đi 1%. Sau đó ghi lại một thông báo như thể này: 'Tesla đang đi với tốc độ 140 km/h, với mức sạc 22%'
4. Tạo một đối tượng ô tô điện và thử nghiệm gọi 'tăng tốc', 'phanh' và 'chargeBattery' (sạc tới 90%). Hãy để ý xem điều gì sẽ xảy ra khi bạn "tăng tốc"! Gợi ý: Ôn lại định nghĩa đa hình



Dữ liệu thử nghiệm:

5 Xe dữ liệu 1: 'Tesla' đang chạy với tốc độ 120 km/h, sạc 23%

CHÚC MAY MẮN



Thử thách viết mã số 4

Nhiệm vụ của bạn:

1. Tạo lại Thử thách #3, nhưng lần này sử dụng các lớp ES6: tạo một lớp con 'EVC1' của lớp 'CarCl'.
2. Đặt thuộc tính 'charge' ở chế độ riêng tư 3.

Triển khai khả năng xâu chuỗi 'tăng tốc' và 'chargeBattery'

các phương thức của lớp này, đồng thời cập nhật phương thức 'phanh' trong lớp 'CarCl'. Sau đó thử nghiệm với chuỗi!

Dữ liệu thử nghiệm:

5 Xe dữ liệu 1: 'Rivian' chạy với tốc độ 120 km/h, sạc 23%

CHÚC MAY MẮN 😊

JavaScript không đồng bộ

Thử thách viết mã số 1

Trong thử thách này, bạn sẽ xây dựng một hàm 'whereAmI' để hiển thị một quốc gia chỉ dựa trên tọa độ GPS. Để làm được điều đó, bạn sẽ sử dụng API thứ hai để mã hóa tọa độ địa lý. Vì vậy, trong thử thách này, bạn sẽ tự mình sử dụng API lần đầu tiên 😊

Nhiệm vụ của bạn:

PHẦN 1

1. Tạo hàm 'whereAmI' lấy đầu vào là giá trị vĩ độ ('lat') và giá trị kinh độ ('lng') (đây là tọa độ GPS, các ví dụ nằm trong dữ liệu thử nghiệm bên dưới).
2. Thực hiện "mã hóa địa lý ngược" của tọa độ được cung cấp. Mã hóa địa lý ngược có nghĩa là chuyển đổi tọa độ thành một vị trí có ý nghĩa, chẳng hạn như tên thành phố và quốc gia.
Sử dụng API này để thực hiện mã hóa địa lý ngược: <https://geocode.xyz/api>. Lệnh gọi AJAX sẽ được thực hiện với một URL có định dạng sau:
<https://geocode.xyz/52.508,13.381?geoit=json>. Sử dụng API tìm nạp và hứa sẽ lấy dữ liệu.
Không sử dụng chức năng 'getJSON' mà chúng tôi đã tạo, đó là gian lận 😊
3. Sau khi bạn có dữ liệu, hãy xem dữ liệu đó trong bảng điều khiển để xem tất cả các thuộc tính mà bạn nhận được về vị trí được cung cấp. Sau đó, sử dụng dữ liệu này để ghi một thông báo như thế này vào bảng điều khiển: "Bạn đang ở Berlin, Đức"
4. Xâu chuỗi một phương thức .catch vào cuối chuỗi lời hứa và ghi lại các lỗi vào
bảng điều khiển
5. API này cho phép bạn chỉ thực hiện 3 yêu cầu mỗi giây. Nếu tải lại nhanh, bạn sẽ gặp lỗi này với mã 403. Đây là lỗi với yêu cầu. Hãy nhớ rằng, hàm get() không từ chối lời hứa trong trường hợp này. Vì vậy hãy tự tạo một lỗi để từ chối lời hứa, với một thông báo lỗi có ý nghĩa

PHẦN 2

6. Bây giờ là lúc sử dụng dữ liệu nhận được để hiển thị một quốc gia. Vì vậy, hãy lấy thuộc tính có liên quan từ kết quả API mã hóa địa lý và cắm nó vào API quốc gia mà chúng tôi đang sử dụng.
7. Render quốc gia và bắt lỗi giống như chúng ta đã làm ở bài giảng trước (thậm chí có thể copy đoạn code này, không cần gõ lại đoạn code đó)

Dữ liệu thử nghiệm:

§ Tọa độ 1: 52.508, 13.381 (Vĩ độ, Kinh độ)

§ Tọa độ 2: 19.037, 72.873

§ Tọa độ 3: -33.933, 18.474

CHÚC MAY MẮN 😊

Thử thách viết mã số 2

Đối với thử thách này, bạn thực sự sẽ phải xem video! Sau đó, xây dựng chức năng tải hình ảnh mà tôi vừa cho bạn xem trên màn hình.

Nhiệm vụ của bạn:

Nhiệm vụ lần này không quá mang tính mô tả để bạn có thể tự mình tìm ra một số nội dung. Giả vờ như bạn đang làm việc một mình 🤔

PHẦN 1

1. Tạo hàm 'createImage' nhận 'imgPath' làm đầu vào.
Hàm này trả về một lời hứa tạo một hình ảnh mới (sử dụng `document.createElement('img')`) và đặt thuộc tính `.src` cho đường dẫn hình ảnh được cung cấp
2. Khi hình ảnh được tải xong, hãy thêm nó vào phần tử DOM bằng 'hình ảnh' và giải quyết lời hứa. Giá trị đáp ứng phải là chính phần tử hình ảnh. Trừ trường hợp xảy ra lỗi khi tải ảnh (nghe sự kiện 'error'), hãy từ chối lời hứa
3. Nếu phần này quá khó đối với bạn, hãy xem phần đầu tiên của lời giải

PHẦN 2

4. Thực hiện lời hứa bằng cách sử dụng `.then` và thêm trình xử lý lỗi
5. Sau khi hình ảnh được tải xong, hãy tạm dừng thực thi trong 2 giây bằng cách sử dụng nút 'chờ' chức năng chúng tôi đã tạo trước đó
6. Sau 2 giây trôi qua, hãy ẩn hình ảnh hiện tại (đặt thuộc tính CSS hiển thị thành 'none') và tải hình ảnh thứ hai (Gợi ý: Sử dụng thành phần hình ảnh được trả về bởi lời hứa 'createImage' để ẩn hình ảnh hiện tại. Bạn sẽ cần một biến toàn cục cho điều đó)
🤔
7. Sau khi hình ảnh thứ hai được tải, hãy tạm dừng thực hiện lại trong 2 giây
8. Sau 2 giây trôi qua, ẩn hình ảnh hiện tại

Dữ liệu thử nghiệm: Hình ảnh trong thư mục `img`. Kiểm tra trình xử lý lỗi bằng cách chuyển một đường dẫn hình ảnh sai. Đặt tốc độ mạng thành "Fast 3G" trong tab Mạng của công cụ dành cho nhà phát triển, nếu không hình ảnh sẽ tải quá nhanh

CHÚC MAY MẮN 😊

Thử thách viết mã số 3

Nhiệm vụ của bạn:

PHẦN 1

1. Viết hàm async 'loadNPause' để tạo lại Thử thách số 2, lần này sử dụng async/await (chỉ phần mà lời hứa được thực hiện, hãy sử dụng lại hàm 'createImage' từ trước)
2. So sánh hai phiên bản, nghĩ về những điểm khác biệt lớn và xem phiên bản nào bạn thích hơn
3. Đừng quên kiểm tra trình xử lý lỗi và đặt tốc độ mạng thành "Fast 3G" trong tab Mạng của công cụ dành cho nhà phát triển

PHẦN 2

1. Tạo hàm không đồng bộ 'loadAll' nhận một loạt đường dẫn hình ảnh 'imgArr'
2. Sử dụng .map để lặp mảng, tải tất cả hình ảnh bằng hàm 'createImage' (gọi mảng kết quả là 'imgs')
3. Kiểm tra mảng 'imgs' trong bảng điều khiển! Có giống như bạn mong đợi không?
4. Sử dụng hàm tổ hợp lời hứa để thực sự lấy hình ảnh từ mảng 5. Thêm lớp 'song song' cho tất cả các hình ảnh (nó có một số kiểu CSS) 🤔

Dữ liệu thử nghiệm Phần 2: ['img/img-1.jpg', 'img/img-2.jpg', 'img/img-3.jpg']. Để kiểm tra hãy tắt chức năng 'loadNPause'

CHÚC MAY MẮN 😊