BỘ CÂU HỎI PHỎNG VẤN BP 2.0

## Câu 1: Hệ thống các kiến thức đã học ở Module 1

|  |  |
| --- | --- |
| 1. Học cách học | * Ghi chép hiệu quả bằng Cornell Notes * Ngộ nhận về cách học * Hiểu về não bộ * Dùng Pomodoro để tăng năng suất làm việc * Nguyên lý của việc học * Các kĩ thuật học tập hàng đầu   + Sức mạnh của thái độ   + Các câu hỏi Socrates   + Hướng dẫn viết Phản tư (Reflection) * Đọc sách hiệu quả |
| 2. Hoàn thành mọi việc với Kanban | * Tạo lập thói quen Scrum Life để làm việc vui vẻ và hiệu quả * Đặt mục tiêu tốt cho ngắn hạn và dài hạn * Lên kế hoạch linh hoạt và quản lý công việc với bảng Kanban * Giới hạn công việc đang làm để tập trung hoàn thành từng việc một * Sớm nhận biết các khó khăn và thay đổi nhằm thích ứng và giữ cho công việc luôn trôi chảy * Nâng cao chất lượng công việc dựa trên Định nghĩa Hoàn thành * Cải tiến liên tục để tiến bộ không ngừng |
| 3. Thuật toán/Giải thuật | * Thuật toán và các bước giải quyết vấn đề * Hai phương pháp mô tả thuật toán: Pseudo-code & Flowchart |
| 4. Git & HTML | * Tìm hiểu về Git * Một số thao tác cơ bản với Git * Tìm hiểu về một trang web cơ bản * Các thẻ HTML cơ bản và thuộc tính của chúng * Comment trong HTML * HTML Form & Table * Hai phương thức POST và GET trong HTML Form |
| 5. JavaScript Overview | * Nhúng JavaScript vào trang web * Debug trong JS * Một số sự kiện trong JS * Cú pháp JS: Biến, hằng, kiểu dữ liệu, toán tử, … |
| 6. Cấu trúc trong JS | * Cấu trúc rẽ nhánh (Conditions)   + if, if – else, if – else – if,…   + switch – case   + Toán tử ba ngôi * Cấu trúc lặp (Loops)   + For   + While, Do – while   + Vòng lặp lồng nhau |
| 7. Mảng (Arrays) | * Mảng trong JS * Mảng một chiều, mảng nhiều chiều * Các thao tác với mảng * Các hàm có sẵn của mảng |
| 8. Hàm (Functions) | * Hàm trong JS * Đối số của hàm * Giá trị trả về của hàm * Gọi hàm * Coding Convention |
| 9. Lập trình hướng đối tượng - OOP | * Tìm hiểu về lớp và đối tượng * Ba cách định nghĩa lớp trong JS * Vẽ trên Canvas * Khai báo và khởi tạo đối tượng |

## Câu 2: Giải thuật/Thuật toán là gì?

Thuật toán, còn gọi là Giải thuật, là một bộ các quy tắc hay quy trình cụ thể nhằm giải quyết một vấn đề nào đó trong một số bước hữu hạn, hoặc nhằm cung cấp một kết quả từ một tập hợp của các dữ kiện đưa vào.

## Câu 3: So sánh việc trình bày giải thuật bằng pseudo-code và flowchart?

|  |  |  |
| --- | --- | --- |
| **Cách thực hiện** | **Ưu điểm** | **Nhược điểm** |
| Mã giả (Pseudo-code) sử dụng ngôn ngữ tự nhiên để biểu diễn thuật toán nhưng nó không thể được biên dịch và thực thi như mã thật | Đơn giản, không cần kiến thức về cách biểu diễn | Dài dòng, không cấu trúc  Đôi lúc khó hiểu, không diễn đạt được thuật toán |
| Lưu đồ (Flowchart) Một lưu đồ là một hình ảnh minh họa cho giải thuật.  Lưu đồ mô tả giải thuật bằng các sơ đồ hình khối, mỗi khối quy định một hành động. | Quy định chặt chẽ về từng kí hiệu | Cần biết về cách biểu diễn |

## Câu 4: Tại sao lại cần trình bày giải thuật trước khi bắt tay vào code?

Tại vì giải thuật giúp chúng ta phân tích, xác định rõ ràng các bước để giải quyết vấn đề, tìm ra giải pháp tối ưu nhất, ít tốn thời gian và công sức nhất trước khi bắt tay vào code.

## Câu 5: Tại sao lại nên sử dụng Flowchart hơn Pseudo-code khi thể hiện giải thuật của bài toán phức tạp?

Với bài toán phức tạp, sử dụng Flowchart rất hữu ích trong việc truyền đạt. Ngoài việc thể hiện được thuật toán, nó còn thể hiện trực quan luồng đi của chương trình. Giúp người đọc có thể dễ dàng thực hiện dựa trên đó.

Với Pseudo-code, tuy cũng thể hiện được thuật toán, nhưng nó không thực hiện được luồng đi của chương trình nên ta sẽ khó hình dung ra các bước giải quyết bài toán hơn Flowchart.

## Câu 6: Tại sao giải thuật lại cần hữu hạn các bước thực thi?

Nếu không có hữu hạn các bước thực thi (vô hạn) thì chương trình sẽ không bao giờ kết thúc; đồng nghĩa với việc bài toán sẽ không bao giờ được giải quyết.

## Câu 7: Trình bày ý nghĩa các ký hiệu sử dụng khi vẽ flowchart?

* Ký hiệu hình eclipse: Bắt đầu hay kết thúc lưu đồ
* Ký hiệu hình bình hành: Các lệnh xuất/nhập
* Ký hiệu hình chữ nhật: Các bước tính toán, xử lý
* Ký hiệu hình thoi: Các bước cần rẽ nhánh
* Ký hiệu mũi tên: Dòng chảy của chương trình
* Ký hiệu mũi tên + hình tròn nối: Nối hai trang kế tiếp nhau

## Câu 8: Vẽ giải thuật giải phương trình bậc nhất?

Giải thuật phương trình bậc nhất dạng ax + b = 0 như sau:

BEGIN

INPUT a, b

IF a = 0 THEN

IF b = 0 THEN

DISPLAY “Phương trình vô số nghiệm”

ELSE

DISPLAY “Phương trình vô nghiệm”

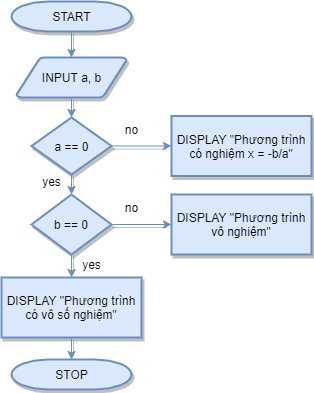
END IF

ELSE

DISPLAY “Phương trình có nghiệm x = -b/a

END IF

END



## Câu 9: Bạn biết gì về VCS – Version Control System (hệ thống quản lý phiên bản)?

Version Control System (VCS) – Hệ thống quản lý phiên bản mã nguồn là một phần mềm hỗ trợ:

* Khôi phục lại phiên bản cũ của các file
* Khôi phục lại phiên bản cũ của toàn bộ dự án
* Xem lại các thay đổi đã được thực hiện theo thời gian
* Xem ai là người thực hiện thay đổi cuối cùng có thể gây ra sự cố
* Khôi phục lại các file vô tình xóa mất

## Câu 10: Phân biệt hệ thống quản lý mã nguồn tập trung và hệ thống quản lý mã nguồn phân tán?

Dựa theo cách thiết kế thì VCS được chia thành hai kiểu khác nhau đó là VCS tập trung (centralized) và VCS phân tán (distributed).

Với các VCS dạng tập trung thì source code của dự án sẽ được lưu trữ trên một kho tập trụng (hay kho trung tâm) trên một máy chủ. Mỗi lập trình viên muốn tạo ra sự thay đổi cho source code lưu trữ trong kho trung tâm trung (centralized repository) thì họ cần phải thực hiện một công việc trước đó là update source code trên máy tính của họ với kho tập trung trước sau đó mới được thay đổi.

Ngược lại với VCS dạng phân tán thì mỗi lập trình viên sẽ có riêng một kho (repository) của mình và có thể tự do phát triển khác feature và tạo version mới (hay đúng hơn là một commit) mà không cần phải đồng bộ với kho trên server. Điều này sẽ giúp lập trình viên tập trung vào việc hoàn thành feature hay fix bug mà không cần lo về việc phải liên tục update code trên máy tính của họ với code trên server để giảm thiểu xung đột. Việc xử lý xung đột (nếu có) sẽ được thực hiện một lần duy nhất.

## Câu 11: Git là gì? Tại sao cần sử dụng git?

Git là một hệ thống điều khiển phiên bản (version control system) theo hình thức phân tán.

Git được sử dụng để quản lý mã nguồn (source code) và ghi nhận các thay đổi

**Git sẽ giúp bạn:**

* Lưu trữ mã nguồn tập trung
* Chia sẻ mã nguồn giữa các bên
* Cộng tác giữa các thành viên trong nhóm phát triển
* Khôi phục mã nguồn về các phiên bản khác nhau
* Dễ dàng chỉnh sửa mã nguồn
* Tránh trùng lặp, xung đột mã nguồn

## Câu 12: Git thuộc hệ thống quản lý mã nguồn tập trung hay phân tán?

Git thuộc hệ thống quản lý mã nguồn phân tán.

## Câu 13: Repository là gì? Phân biệt LocalRepository và RemoteRepository?

Repository thường được gọi ngắn gọn là Repo là nơi chứa toàn bộ các file mã nguồn và lịch sử của các file đó.

Có 2 loại Repository là Local Repository và Remote Repository.

**Local Repository:** Ở trên máy của lập trình viên.

**Remote Repository**: Ở trên một máy chủ chia sẻ (chẳng hạn như GitHub).

## Câu 14: Đồng bộ giữa các repository (local và remote) để làm gì?

Đồng bộ giữa các Repository để giúp cho mã nguồn luôn được cập nhật. Làm cho tất cả các Repo là những bản sao của nhau.

## Câu 15: Thao tác push của git để làm gì? Cú pháp câu lệnh?

git push: Đẩy mã nguồn từ Local Repository lên Remote Repository.

Cú pháp: git push -u origin master

## Câu 16: Thao tác pull của git để làm gì?

git pull: Cập nhật mã nguồn từ một Remote Repository về Local Repository.

## Câu 17: Thao tác clone của git để làm gì? Cú pháp câu lệnh?

git clone: Sao chép một Remote Repository về máy của lập trình viên.

Cú pháp: git clone https://github.com/USERNAME/REPOSITORY.git

## Câu 18: Nêu tên và ý nghĩa các câu lệnh cơ bản đã được học để làm việc với git?

git clone Sao chép một Repository

git init Khởi tạo một Repository.

git remote add origin https://github.com/user/repo.git Kết nối tới Remote Repository

git add <file\_name> Đưa một file vào quản lý trong chỉ mục của GIT (staged)

git commit -m "First commit" Commit phiên bản kèm theo Message

git push -u origin master Đồng bộ từ Local Repository tới Remote Repository

## Câu 19: www là gì?

www viết tắt của cụm từ World Wide Web là không gian nơi chứa thông tin, tài liệu và tài nguyên của mọi website trên toàn cầu. www được xác định bởi các siêu liên kết URL và truy cập thông qua Internet.

## Câu 20: HTML là gì?

HTML (Hypertext Markup Language) là mã được dùng để xây dựng nên cấu trúc và nội dung của trang web. HTML không phải là một ngôn ngữ lập trình; nó là một ngôn ngữ đánh dấu (markup language) định hình cấu trúc nội dung.

## Câu 21: Nêu tên các thẻ làm việc với văn bản trong tài liệu html?

Tiêu đề (thẻ h1–h6): Thường chỉ sử dụng tối đa là <h1>–<h4>:

<h1>My main title</h1>

<h2>My top level heading</h2>

<h3>My subheading</h3>

<h4>My sub-subheading</h4>

Đoạn văn (thẻ p): <p>This is a single paragraph</p>

Danh sách (thẻ ol, ul): Các loại danh sách phổ biến nhất có thứ tự <ol> và danh sách không có thứ tự <ul>.

Liên kết (thẻ a): <a> — "a" viết tắt của "anchor". Để làm cho văn bản trong đoạn văn của bạn thành một liên kết

## Câu 22: Thẻ div dùng để làm gì? <div> khác gì <span>

Thẻ <div> hay thẻ div trong HTML được dùng để nhóm nhiều phần tử HTML lại với nhau thành một khối.

<div> có kiểu hiển thị là Block (khối), trong khi <span> là một thẻ có kiểu hiển thị là Inline (một dòng) để nhóm các văn bản.

## Câu 23: URL là gì? Phân biệt url tuyệt đối và url tương đối?

URL là từ viết tắt của cụm từ tiếng Anh: “Uniform Resource Locator” có nghĩa tiếng Việt là “Tham chiếu tài nguyên internet”. Nó là một phương tiện để người dùng sử dụng, truy cập đến tài nguyên trên mạng máy tính. Mỗi tài nguyên trên máy tính khi ta truy cập tìm kiếm hàng ngày đều được gán một địa chỉ, hay còn gọi là đường link cố định (thay vì một dãy địa chỉ IP dài). Chỉ cần nhập địa chỉ (đường link) đó và search, bạn có thể đến được địa chỉ website mình tìm kiếm.

URL tuyệt đối (là một chuỗi đầy đủ bao gồm http://, tên miền của trang web, đường dẫn đến tập tin).

URL tương đối là một phần nhỏ của đường dẫn tuyệt đối (thông thường đường dẫn tương đối là phần đường dẫn đến tập tin).

Ví dụ:

URL tuyệt đối: https://www.google.com.vn/search?q=URL

URL tương đối: search?q=URL

## Câu 24: Siêu liên kết là gì?

Hyperlink có nghĩa là một siêu liên kết, là nguồn dẫn tài liệu để khi người dùng click vào nó sẽ được dẫn đến một trang khác hoặc một vị trí nào đó trên trang web.

## Câu 25: Muốn liên kết đến một phần trên cùng trang thì làm thế nào?

Sử dụng thẻ <a>, trong thuộc tính href thì chèn #id của phần mà bạn muốn đi tới trên trang web.

Ví dụ:

<div id="top"></div>

<a href="#top">Go to top</a>

## Câu 26: Muốn liên kết đến một phần trên trang khác thì làm thế nào?

Sử dụng thẻ <a>, trong thuộc tính href thì chèn url của trang web mà bạn muốn tới

Ví dụ: <a href="https://www.google.com.vn/">Go to Google</a>

## Câu 27: Phân biệt internal link và external link?

Internal Link là một liên kết nội bộ từ trang này sang trang khác trên cùng một tên miền.

External link là liên kết ra bên ngoài trang web khác.

## Câu 28: Có mấy loại danh sách?

Trong HTML có 3 kiểu danh sách (HTML Lists) đó là có danh sách thứ tự (ordered list), danh sách không có thứ tự (Unordered list) và danh sách mô tả (description list). Cụ thể:

1. **An ordered HTML list**: Một danh sách có thứ tự có thể là số hoặc theo thứ tự bảng chữ cái.

<ol>

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ol>

1. **An unordered HTML list**: Danh sách không có thứ tự dành cho các danh sách có thứ tự của các vật phẩm, chẳng hạn như công thức

<ul>

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ul>

1. **Description Lists**:

<dl>

<dt>Coffee</dt>

<dd>- black hot drink</dd>

<dt>Milk</dt>

<dd>- white cold drink</dd>

</dl>

## Câu 29: Muốn hiển thị danh sách có thứ tự bắt đầu C. D. E… thì làm thế nào?

Dùng thẻ <ol> với thuộc tính type="A" và start="3"

<ol start="3" type="A">

<li>Coffee</li>

<li>Tea</li>

<li>Milk</li>

</ol>

## Câu 30: Form hay biểu mẫu dùng để làm gì?

Form hay biểu mẫu dùng để thu thập thông tin từ phía người dùng.

## Câu 31: Thuộc tính action của form dùng để làm gì?

Thuộc tính action dùng để định nghĩa hành động sẽ được thực hiện khi nhấn nút gửi dữ liệu, hành động này có thể là gửi dữ liệu lên web server, gửi dữ liệu đến một mail server.

## Câu 32: Phân biệt 2 method là get và post?

Mặc định thuộc tính method của form có giá trị là GET. Khi sử dụng phương thức GET, dữ liệu của form sẽ được đưa vào kèm theo với đường dẫn URL gửi đến máy chủ web.

Khi sử dụng phương thức POST với form nhập liệu, các dữ liệu form sẽ không được hiển thị trong URL mà nó sẽ đóng gói trong header của gói tin gửi đi, do đó các thông tin sẽ được bảo mật hơn. Với phương thức POST, độ lớn của dữ liệu gửi đi là không giới hạn, tuy nhiên gửi dữ liệu kiểu POST không thể đánh dấu để sử dụng lại sau.

## Câu 33: Phân biệt các nút bấm có type là: button, submit, reset?

<input type="button" value="Anonymous button"> <!-- Không có hiệu ứng khi nhấn nhưng có thể được tùy chỉnh bằng JavaScript -->

<input type="submit" value="Submit"> <!-- Gửi dữ liệu form đến máy chủ. -->

<input type="reset" value="Reset"> <!-- Đặt lại tất cả giá trị form về mặc định. -->

## Câu 34: Muốn submit form đến trang CodeGym.vn thì làm thế nào?

<form action="https://codegym.vn" method="get">

<input type="submit" value="Submit">

</form>

## Câu 35: Nếu muốn nối 3 ô trên cùng 1 hàng thì làm thế nào?

Sử dụng thuộc tính colspan

Ví dụ: <td colspan="3"></td>

## Câu 36: Nếu muốn nối 2 ô trên cùng 1 cột thì làm thế nào?

Sử dụng thuộc tính rowspan

Ví dụ: <td rowspan="3"></td>

## Câu 37: Phân biệt thuộc tính cellpadding và thuộc tính cellspacing?

* Thuộc tính cellspacing dùng để tạo khoảng cách giữa các ô trong table.
* Thuộc tính cellpadding dùng để tạo khoảng cách giữa nội dung trong ô so với đường viền.

## Câu 38: Trình bày các cách nhúng JavaScript vào 1 tài liệu html?

Có một số cách khác nhau để nhúng mã JavaScript vào một trang web:

* Sử dụng thẻ <script> bên trong mã HTML
* Sử dụng file JavaScript riêng biệt
* Viết mã JavaScript ngay trong thẻ các thẻ html

## Câu 40: Phát biểu “JavaScript phân biệt chữ hoa chữ thường ” đúng hay sai?

JavaScript phân biệt hoa thường — myVariable và myvariable là hai biến khác nhau

## Câu 41: Trình bày về quy tắc CamelCase và camelCase?

camelCase: Chữ cái đầu tiên của từ đầu tiên viết thường. Các từ còn lại viết hoa chữ cái đầu tiên. Thường sử dụng để đặt tên biến, hàm, phương thức

VD: createCircle, getArea, inputNumber,…

CamelCase: Quy tắc này yêu cầu viết hoa các chữ cái đầu tiên của mọi từ. Thường sử dụng để đặt tên class.

VD: PuzzleGame, FlashLamp,….

## Câu 42: Biến là gì? Phân biệt biến cục bộ (local) và biến toàn cục (global)?

Biến là tên của vùng nhớ dùng để lưu trữ dữ liệu.

Biến cục bộ (local): Một biến được gọi là cục bộ khi bạn khai báo nó nằm bên trong một hàm cụ thể nào đó, lúc này biến đó sẽ không sử dụng được ở bên ngoài hàm. (Sử dụng từ khoá let khai báo trong hàm hoặc trong khoảng dấu {} )

Biến toàn cục (global): là biến mà bạn khai báo bên ngoài và không nằm bên trong một hàm cụ thể nào cả. (Sử dụng từ khoá let hoặc var; khi khai báo trong hàm hoặc khoảng dấu {} mà dùng var thì biến đó vẫn là biến toàn cục)

## Câu 43: Phân biệt: var, let, const?

const được sử dụng để khai báo 1 hằng số, và giá trị của nó không thay đổi trong suốt chương trình.

let khai báo biến chỉ có thể truy cập được trong block bao quanh nó được xác định bằng cặp {}.

var khai báo biến có thể truy cập ở phạm vi hàm số hoặc bên ngoài hàm số, toàn cục.

## Câu 44: Phát biểu “JavaScript là ngôn ngữ định kiểu yếu hay động” nghĩa là gì? Trình bày?

JavaScript là một ngôn ngữ định kiểu yếu hay động. Điều đó nghĩa là không cần phải khai báo kiểu của các biến trước khi dùng. Kiểu sẽ được xác định tự động trong khi chương trình được thực thi.

let foo = 42; // foo có kiểu dữ liệu number

foo = 'bar'; // foo bây giờ có kiểu dữ liệu string

foo = true; // foo bây giờ có kiểu dữ liệu boolean

## Câu 45: Nêu tên các kiểu dữ liệu nguyên thủy trong JavaScript?

Trong JS có 5 kiểu dữ liệu nguyên thuỷ (Primitive Data):

* *Number*
* *Boolean*
* *String*
* *Null*
* *Undefined*

## Câu 46: Trình bày về toán tử typeof? Khi khai báo “let x=10;” thì x có kiểu là gì?

Toán tử typeof trả về kiểu dữ liệu của đối tượng nào đó.

let x = 10; // x có kiểu dữ liệu là number

console.log(typeof(x)) // "number"

## Câu 47: Trình bày các cách đưa thông báo trong JavaScript?

Có 3 cách đưa thông báo trong JS:

1. alert(): Dùng để hiển thị một hộp thoại thông báo đến người dùng
2. confirm(): Dùng để hiển thị một hộp thoại thông báo và yêu cầu người dùng phải xác nhận là đồng ý hoặc quay lại (OK hoặc Cancel)
3. prompt(): Dùng để hiển thị một hộp thoại thông báo và yêu cầu người dùng nhập dữ liệu vào.

## Câu 48: Trình bày về các loại toán tử trong JavaScript?

Toán tử cộng/nối chuỗi

Dùng để cộng hai số, hoặc nối hai chuỗi lại với nhau, ký hiệu +.

### Toán tử số học

Toán tử xuất hiện ở giữa hai toán hạng, cho phép thực hiện các phép tính với giá trị số và chuỗi và trả về một giá trị duy nhất. Các toán tử bao gồm: + (cộng),- (trừ), \* (nhân), / (chia), % (chia lấy dư).

### Toán tử tăng, và giảm

Chúng chỉ thực hiện được trên một toán hạng duy nhất. Toán tử tăng làm tăng giá trị lên 1, trong khi toán tử giảm làm giảm giá trị xuống 1; các toán tử có thể đặt trước hoặc sau toán hạng. Các toán tử bao gồm: ++ (tăng), -- (giảm).

### Toán tử quan hệ

Là các toán tử dùng để so sánh giữa hai toán hạng. Sau khi thực hiện một so sánh, chúng trả lại một giá trị true (đúng) hay false (sai). Các toán tử bao gồm: == (bằng nhau), != (khác nhau), === (bằng nhau và cùng kiểu dữ liệu), !== (khác nhau và khác kiểu dữ liệu), > (lớn hơn, < (nhỏ hơn), >= (lớn hơn hoặc bằng), <= (nhỏ hơn hoặc bằng).

### Toán tử luận lý (logic)

Thực hiện các phép toán logic trên hai toán hạng. Chúng thuộc loại toán tử quan hệ, vì chúng trả về một giá trị boolean. Các toán tử bao gồm: && (và), || (hoặc), ! (phủ định).

### Toán tử đặc biệt (Toán tử ba ngôi)

Là toán tử điều kiện còn được biết đến với toán tử 3 ngôi. Cú pháp của toán tử này như sau.

**Syntax (cú pháp)**:

variable\_name = (condition) ? exprIfTrue : exprIfFalse

Toán tử này sẽ trả lại giá trị là kết quả của exprIfTruenếu điều kiện có giá trị boolean bằng true, ngược lại nó sẽ trả lại giá trị bằng biểu exprIfFalse.

## Câu 49: Phân biệt giữa 2 toán tử == và ===?

==: So sánh hai giá trị với nhau.

===: So sánh hai giá trị và hai kiểu dữ liệu với nhau.

## Câu 50: Phân biệt giữa 2 toán tử && và || trong JavaScript?

&& (và): Trả về giá trị truekhi tất cả các toán hạng đều truevà falsekhi ít nhất một toán hạng false.

|| (hoặc): Trả về giá trị truekhi ít nhất một toán hạng truevà falsekhi tất cả các toán hạng false.

## Câu 51: Kết quả biểu thức +0===-0 là true hay false?

+0 === -0 // trả về true vì số đối của 0 là 0 và cùng là kiểu dữ liệu number

## Câu 52: Trình bày về toán tử ++, -- và – trong JavaScript?

Toán tử ++ dùng để tăng giá trị thêm 1 đơn vị.

Toán tử -- dùng để giảm giá trị đi 1 đơn vị.

Toán tử - dùng để thao tác với số đối.

## Câu 53: Phân biệt ++a và a++?

a++ trả về giá trị của a trước khi tăng.

++a tăng giá trị của a trước rồi mới trả về giá trị sau khi tăng.

## Câu 54: Câu lệnh điều kiện/rẽ nhánh là gì?

Câu lệnh điều kiện là cấu trúc mã nguồn cho phép bạn kiểm tra xem liệu một biểu thức có trả về true hay không, rồi chạy các đoạn mã nguồn kế tiếp dựa theo kết quả ấy.

## Câu 55: Có những loại câu lệnh điều kiện nào?

Trong JavaScript, chúng ta có những lệnh điều kiện sau đây:

1. Sử dụng if để xác định một khối mã được thực thi, nếu một biểu thức điều kiện được đánh giá là true.
2. Sử dụng else để xác định một khối mã được thực thi, nếu điều kiện được chỉ ra trong if là false.
3. Sử dụng else if nhằm xác định một điều kiện mới để kiểm tra, nếu các điều kiện trước đó là false.
4. Sử dụng switch để xác định nhiều khối mã luân phiên được thực thi.

## Câu 56: Trình bày về câu lệnh if?

Sử dụng if nếu điều kiện có giá trị đúng (true) thì khối mã bên trong if ({ ... }) sẽ được thực thi.

Nếu điều kiện có giá trị sai (false) thì khối mã bên trong if sẽ được bỏ qua. Luồng thực thi sẽ chuyển xuống ngay sau khối lệnh if.

**Syntax (cú pháp):**

if (condition) {

// block of code to be executed if the condition is true

}

Trong đó:

• condition: Là biểu thức trả về giá trị kiểu boolean.

• statements: Các câu lệnh sẽ được thực thi nếu điều kiện trả về true.

## Câu 57: Trình bày về câu lệnh switch?

switch-case là một cấu trúc điều kiện cho phép lựa chọn thực thi các khối lệnh khác nhau dựa trên kết quả của việc so sánh.

switch-case so sánh giá trị của một biến với lần lượt từng giá trị một, nếu có giá trị phù hợp với biến thì khối lệnh tương ứng sẽ được thực thi.

switch-case không thể thay thế if-else trong tất cả các trường hợp.

switch (expression) {

case choice1:

// run this code

break;

case choice2:

// run this code instead

break;

// include as many cases as you like

default:

// actually, just run this code

}

## Câu 58: Tại sao lại nói else là một tùy chọn (optional) của câu lệnh if?

Vì nếu thiếu else thì chương trình vẫn hoạt động bình thường mà không hề phát sinh lỗi.

## Câu 59: Khi nào nên sử dụng câu lệnh switch?

Khi biểu thức điều kiện bắt buộc là so sánh bằng và số điều kiện nhiều. Việc sử dụng switch-case sẽ giúp cấu trúc chương trình rõ ràng, dễ đọc, dễ hiểu hơn.

## Câu 60: Câu lệnh break dùng để làm gì? Nếu một case mà không có break thì chương trình chạy thế nào?

Câu lệnh break sẽ thoát ra khỏi khối mã trong switch hoặc trong vòng lặp.

Điều này sẽ làm ngừng thực thi thêm các mã và các case bên trong khối lệnh.

Nếu không có câu lệnh break, chương trình sẽ thực thi từ case đúng cho đến hết khối lệnh theo tuần tự từ trên xuống dưới.