**TRẢ LỜI CÂU HỎI AUDIT MODULE 1**

|  |  |
| --- | --- |
| **1** | **Các câu lệnh để submit 1 file README.MD lên github là gì? Giải thích tác dụng của từng câu lệnh** |
| |  |  | | --- | --- | | git add . |  | | git commit – m “comment” |  | | git push origin main |  | |
| **2** | **Nêu các thẻ để tạo danh sách ? Sự khác nhau giữa <ul> và <ol> ?** |
| Thẻ tạo danh sách: <ul>,<ol>,<dl> (**Definition list** - danh sách tương tự  từ điển) |
| |  |  |  | | --- | --- | --- | | <ul><li></li</ul> | danh sách không có thứ tự | (mặc định là dấu chấm tròn phía trước mỗi mục) | | <ol><li></li</ol> | danh sách sắp xếp theo thứ tự | (mặc định là số phía trước mỗi mục) | |
| **3** | **Các thẻ tạo nên 1 table? Phân biệt th và td?** |
| Thẻ tạo table:  bắt đàu bằng thẻ mở đóng <table>  tiếp theo là <tr>  tiếp theo là <th> <td>, 2 thẻ này đều nằm trong thẻ <tr> |
| |  |  | | --- | --- | | <th> | table head: thông tin tiêu đề chung cho một nhóm các đơn vị dữ liệu.) | | <td> | table data: (chứa các đơn vị dữ liệu cả bảng) | |
| **4** | **Phân biệt innerHTML và innerText?** |
| |  |  | | --- | --- | | **innerText** | **innerHTML** | | đều dùng để hiển thị dữ liệu đầu ra | | | innerText chỉ dùng để chèn Text (nội dung văn bản, ko thể chèn HTML). | innerHTML có thể chèn text giống như innerText, đồng thời có thể chèn thẻ HTML (khác với innerText) | |  |  | |
| **5** | **Phân biệt get và post.** |
| |  |  | | --- | --- | | **GET** | **POST** | | dữ liệu được hiển thị trên URL | dữ liệu không được hiển thị trên URL | | không bảo mật. | bảo mật hơn. | | có giới hạn dung lượng data khi gởi (thường được dùng với màn hình tìm kiếm hiển thị chi tiết sản phẩm,...). | không giới hạn dung lượng dât khi gởi (thường dùng với màn hình có dữ liệu nhạy cảm như password, thông tin cá nhân người dùng,...). | |
| **6** | **Mô tả thuật toán tìm kiếm 1 phần tử trong mảng bằng mã giả?** |
|  |  |
| **7** | **Mô tả thuật toán sắp xếp mảng số nguyên bằng mã giả?** |
|  |  |
| **8** | **Cách khai báo biến bằng từ khóa let và var khác nhau như thế nào ?. Phạm vi của biến?** |
| |  |  | | --- | --- | | **var** | **let** | | cả 2 đều dùng để khai báo biến | | | -Phạm vi sử dụng biến var là phạm vi toàn cục.. | -Phạm vi sử dụng biến var là phạm vi cục bộ  mỗi block được xác định bởi dấu {} | | -có thể khai báo cùng 1 tên biến | -không được phép khai báo cùng 1 tên biến. | | -cho phép dùng biến var trước khi khai báo. | -bắt buộc khai báo biến trước khi sử dụng (khuyên dùng). | |
| **9** | **Trong Javascript có bao nhiêu loại kiểu dữ liệu ?. Làm thế nào để xác định được biến có kiểu dữ liệu gì?** |
| |  |  | | --- | --- | | 5 kiểu dữ liệu: | string: chuỗi kí tự và giá trị được khai báo trong cặp dấu nháy đôi hoặc đơn. | | number: số nguyên, số thực. | | boolean: giá trị T or F | | object | | undefined: không xác định - khi mình khai báo biến mà không gán giá trị cụ thể, hoặc gán giá trị cho biến là "undefined" | | để xác định được biến có kiểu dữ liệu gì: | dùng từ khóa typeof | |
| **10** | **Các cách ép sang kiểu String trong Javascript?** |
| join()  String()  **số sang chuỗi:** toString() hoặc dùng đối tượng string let abs =new String;  số + ‘ ‘ => chuỗi |
| **11** | **Các cách tạo chuỗi chứa dấu nháy?** |
| sử dụng nháy khác nháy 2 đầu chuỗi  sử dụng ký tự xuyệt \’ -> tự động giữ lại dấu nháy sau dấu \ |
| **12** | **NaN là gì? NaN === NaN có đúng không?Vì sao?** |
|  | **NaN (not a number)**  **nó không phải là giá trị** **nên nó không so sánh được.** |
| **13** | **Toán tử ba ngôi là gì? Cú pháp?** |
|  | **Toán tử ba** : cho phép gán giá trị cho một biến dựa theo điều kiện mà không cần dùng câu lệnh if...else  **Cú pháp: condition ? expression true case : expression false case;** |
| **14** | **Toán tử == và === khác và giống nhau như thế nào** |
|  | |  |  | | --- | --- | | **==** | **===** | | so sánh về mặt giá trị | so sánh về giá trị và kiểu dữ liệu | |  |  | |
| **15** | **Phân biệt giữa 2 toán tử && và || trong JavaScript?** |
|  | |  |  | | --- | --- | | **&&** | **||** | | cả 2 vế đều T đúng thì trả về T | 1 trong 2 vế T thì nó T | |  |  | |
| **16** | **Các hàm dùng để tạo thông báo trong Javascript ?** |
|  | alert(): thông báo/ cung cấp thông tin đến người dùng.  prompt(): thông báo cho người dùng, đồng thời yêu cầu người dùng cung cấp thông tin. (kết quả luôn trả về là chuỗi)  confirm(); thông báo xác nhận đúng hoặc sai (true or false), thường dùng để hỏi yes/no. |
| **17** | **Casting data type trong JS** |
|  | (ép kdl này sang kdl khác: parseInt…)   |  |  | | --- | --- | | chuyển từ chuỗi qua số | parseInt() chuyển sang số nguyên | | parseFloat() chuyển sang số thập phân | | số sang chuỗi | String(số);String(true);String([1,2,3]) | | chuỗi sang số | number(‘12’) -> 12 | |  | number(true) -> 1 number(false) ->0 | | ko thể ép mảng thành chuỗi | number(’adadsvdvdv’) -> NaN | | boolean | boolean(1) -> true; boolean(0) -> false; boolean(‘true’) -> true | |
| **18** | **Có bao nhiêu Statement control trong Java Script ?** |
|  | Câu lệnh điều khiển (control follow statement)  **Có 3 Statement control trong JS:**  **- câu lệnh điều kiện (conditional statement)**  **- câu lệnh lặp (loop statement)**  **- câu lệnh nhảy (jump statement)** |
| **19** | **So sánh sự khác nhau giữa if và switch case?.** |
|  | |  |  | | --- | --- | | If | **switch case** | | có thể so sánh >,<,=,!= | chỉ ung cho so sánh = hoặc != | | biểu thức có giá trị trả về là kiểu ungan. | Biểu thức có giá trị trả về là kiểu number, string, ungan. | | Chỉ có 1 khối lệnh được thực thi nếu điều kiện đúng. | Nếu không có break; thì tất cả khối lệnh đằng sau đều được thực thi. | | Nên dung tối đa 3 if lồng nhau. | Hỗ trợ check nhiều điều kiện | |  |  | |
| **20** | **Switch case so sánh = = hay = = =. Đặt ra trường hợp là so sánh bằng thì khi nào sử dụng if bậc thang? Khi nào sử dụng switch case** |
|  | |  |  |  | | --- | --- | --- | | **so sánh = = =** | vì xét cả giá trị và kdl |  | | **trường hợp là so sánh bằng thì** | **sử dụng if bậc thang khi** | **ít hơn 3 thì dùng bậc thang** | | **sử dụng switch casekhi** | **nhiều hơn 3 khối đkien** | |
| **21** | **Các biểu thức và luồng thực thi của for. Nếu thiếu 1 hoặc tất cả các biểu thức thì vòng for sẽ chạy như thế nào?** |
|  | |  |  | | --- | --- | | biểu thức | for (initial-action (1); loop-condition (2); action-after each interaction (4)){  //body (3)  }   1. khối lệnh khời tạo biểu lặp, chỉ chạy 1 lần 2. điều kiện lặp 3. nội dung cần thực thi 4. khối lệnh tăng hoặc giảm biểu lặp | | luồng thực thi: | (1)->(2)->(3)->(4)->(2)->(3)-(4) … chạy đến khi không thỏa được điều kiện thì thoát vòng lặp. | | Nếu thiếu 1 hoặc tất cả các biểu thức | khối khởi tạo để bên trên, khối lệnh tăng giảm để cuối body  ko nên để khuyết khối lệnh đk, để nó tường minh | |
| **22** | Đặt ra 1 bài toán. Xác định bài toán cần sử dụng vòng lặp nào? |
|  |  |
| **23** | **So sánh giống và khác nhau giữa for, while và do..while** |
|  | |  |  |  | | --- | --- | --- | | **for** | **while** | **do..while** | | sử dụng for khi biết trước số lần lặp | sử dụng while khi không biết trước số lần lặp | thông thường sử dụng **do..while** khi không biết trước số lần lặp | |  | xét điều kiện trước rồi mới lặp | lặp ít nhất 1 lần rồi mới xét điều kiện | |
| **24** | So sánh null và rỗng |
|  | |  |  |  | | --- | --- | --- | | **null** | **rỗng** | ***undefined*** | | **là giá trị rỗng hoặc giá trị không tồn tại, nó có thể được sử dụng để gán cho một biến như là một đại diện không có giá trị.**  **có tồn tại chưa cấp ô nhớ, là kiểu object** | **là biến đó đã được gán giá trị rỗng vào đó hoặc một mảng không chứa giá trị nào nhưng đã được gán dấu [] thì độ dài của nó là 0 (mảng rỗng, không có phần tử trong mảng.)**  **đc cấp ô nhớ nhưng ko có giá trị** | ***là không xác định. Trong javascript, khi bạn khai báo một biến nhưng chưa gán giá trị cho nó.*** | |
| **25** | **Đặc điểm mảng một chiều trong Javascript** |
|  | Đặc điểm:  -truy xuất mảng thông qua index  -mảng ko cố định, kích thước có thể thay đổi |
| **26** | Các cách khởi tạo một mảng kiểu String trong JavaScript |
|  | let str = [‘’abc”;”22424”];  let arr = new Array(“abc”,”2133”)  dùng split |
| **27** | **1 số hàm thao tác với mảng?  - Phân biệt push(thêm phần tử cuối mảng) và unshift(thâm phần tử đầu mảng) - Phân biệt push(them phần tử đầu mảng) và pop(xóa phần tử cuối mảng) - Phân biệt shift(xóa phần tử đầu khỏi mảng) và unshift(them phần tử vào đầu mảng)** |
| **28** | **Phân biệt tham trị và tham chiếu trong Javascript** |
|  | Truyền tham trị áp dụng kiểu nguyên thủy(số,Boolean,string) giá trị của biến trước khi gọi funcition và sau khi gọi là không thay đổi.  Truyền tham chiếu áp dụng cho kiểu đối tượng (obiject,arr) giá trị có thể bị thay đổi. |
| **29** | **Phân biệt giữa Hàm có return và hàm không có return** |
|  | Hàm có return là có giá trị trả về còn ko có thì ko có giá trị trả về |
| **30** | **So sánh break, continue và return** |
|  | break, dùng trong vòng lặp và switch-case  return, trả về giá trị + thoát hàm |
| **31** | **Lập trình hướng đối tượng là gì** |
|  | LT OOP(obiject oriented programming) là kĩ thuật lập trình cho phép lập rình viên tạo ra các đối tượng trong code ánh xạ các đối tượng thực tế trong ngôn ngữ LT  Đối tượng bao gốm thuộc tính và hành vi |
| **32** | **Các đặc điểm trong Lập trình hướng đối tượng. Đưa ra 1 ví dụ 1 trong 4 tính chất.** |
|  | **Tính đóng gói (Encapsulation)** Tính đóng gói cho phép che giấu thông tin và những tính chất xử lý bên trong của đối tượng. Các đối tượng khác không thể tác động trực tiếp đến dữ liệu bên trong và làm thay đổi trạng thái của đối tượng mà bắt buộc phải thông qua các phương thức công khai do đối tượng đó cung cấp. **Tính kế thừa (Inheritance)** Đây là tính chất được sử dụng khá nhiều. Tính kế thừa cho phép xây dựng một lớp mới (lớp Con), kế thừa và tái sử dụng các thuộc tính, phương thức dựa trên lớp cũ (lớp Cha) đã có trước đó. Các lớp Con kế thừa toàn bộ thành phần của lớp Cha và không cần phải định  nghĩa lại. Lớp Con có thể mở rộng các thành phần kế thừa hoặc bổ sung những thành phần mới. **Tính đa hình (Polymorphism)** Tính đa hình trong lập trình OOP cho phép các đối tượng khác nhau thực thi chức năng giống nhau theo những cách khác nhau. **Tính trừu tượng (Abstraction)** Tính trừu tượng giúp loại bỏ những thứ phức tạp, không cần thiết của đối tượng và chỉ tập trung vào những gì cốt lõi, quan trọng.  LƯU Ý : MỌI NGƯỜI PHẢI TỰ ĐƯA RA ĐƯỢC VÍ DỤ VỀ CÁC TÍNH CHẤT ! |
| **33** | **Constructor là gì? Trong 1 class có nhiều hơn 1 contrustor được hay không?** |
|  | Là hàm khởi tạo mảng,chỉ có duy nhất 1 constructor  Dùng từ khóa new |
| **34** | **Những phương thức nào cho phép tương tác với chuỗi** |
|  | Sring(), toString(), split(); substring() cắt chuỗi, substr(), slice(), toLowerCase(), toUpperCase(), lấy kí tự trong chuỗi charAt() |
| **35** | **Các câu lệnh để vẽ 1 hình tròn. Giải thích từng câu lệnh?** |
|  | Tạo thẻ canvas  Cho số đo  Dùng ctx.beginPath  Ctx.arc(circle.x,circle.y circle.radius, 0, 2\*math.PI  Ctx.fill |
| **36** | **Trình bày ý tưởng của 1 thuật toán bất kì?** |
|  |  |
| **37** | **Cho 1 bài toán liên quan đến ++ trước và ++ sau. Yêu cầu dự đoán kết quả, và tại sao lại có kết quả như vậy?** |
|  | ++a: tăng lên 1 giá trị và trả về sau khi đc cộng 1 giá trị  a++ : tăng lên 1 giá trị khi thực thi và trả về giá trị cũ |
| **38** | **Các loại toán tử trong js? Cho biết độ ưu tiên của các toán tử trong một biểu thức?** |
|  | TT gán  TT số học  TT so sánh  TT logic |