Chuẩn đầu ra PRF192x

1. PRF192x\_o1 - Tạo được file HTML chứa JavaScript và chạy trên máy tính cá nhân
2. PRF192x\_o2 - Tạo được file HTML chứa JavaScript trên bộ soạn thảo online và chạy thử.
3. PRF192x\_o3 - Hiểu cú pháp của JavaScript
   1. Inline:
      1. <button type="button" onclick="document.getElementById('demo').style.display = 'block'">
      2. <button type="button" onclick="document.getElementById('demo').innerHTML = 'My First JavaScript' "> Change Me </button>
   2. External: <script src="myJavaScript.js"> </script>
   3. Body:
      1. <script>  
          function myFunction(){  
          document.getElementById('demo').innerHTML = "Hello Javascript";  
          }  
         </script>
   4. window.alert(5 + 6);
4. PRF192x\_o4 - Hiểu về biến và các toán tử trong JavaScript
   1. Variables
      1. document.write(5 + 6); // 11
      2. document.write(‘5 + 6’); // 5 + 6
      3. JS Var:
         1. Variables declared with the var keyword can NOT have block scope.
         2. Variables declared inside a { } block can be accessed from outside the block.
            1. {  
                var x = 2;  
               }  
               // x CAN be used here
         3. Redeclaring a variable inside a block will also redeclare the variable outside the block:
            1. var x = 10;  
               // Here x is 10  
               {  
               var x = 2;  
               // Here x is 2  
               }  
               // Here x is 2
         4. Redeclaring a JavaScript variable with var is allowed anywhere in a program:
            1. var x = 2;  
               // Now x is 2  
               var x = 3;  
               // Now x is 3
      4. JS Let:
         1. Variables defined with let cannot be Redeclared.
         2. Variables defined with let must be Declared before use.
         3. Variables defined with let have Block Scope.
         4. Variables declared inside a { } block cannot be accessed from outside the block:
            1. {  
                let x = 2;  
               }  
               // x can NOT be used here
         5. Redeclaring a variable inside a block will not redeclare the variable outside the block:
            1. let x = 10;  
               // Here x is 10  
               {  
               let x = 2;  
               // Here x is 2  
               }  
               // Here x is 10
         6. With let, redeclaring a variable in the same block is NOT allowed:
            1. var x = 2; // Allowed  
               let x = 3; // Not allowed  
               {  
               let x = 2; // Allowed  
               let x = 3 // Not allowed  
               }  
               {  
               let x = 2; // Allowed  
               var x = 3 // Not allowed  
               }
         7. Redeclaring a variable with let, in another block, IS allowed:
            1. let x = 2; // Allowed  
               {  
               let x = 3; // Allowed  
               }  
               {  
               let x = 4; // Allowed  
               }
      5. JS Const
         1. Variables defined with const cannot be Redeclared.
            1. const PI = 3.141592653589793;  
               PI = 3.14; // This will give an error  
               PI = PI + 10; // This will also give an error
         2. Variables defined with const cannot be Reassigned.
            1. JavaScript const variables must be assigned a value when they are declared:

const PI;  
PI = 3.14159265359;  
// Wrong

const PI = 3.14159265359;  
// Correct

* + - 1. Variables defined with const have Block Scope.
      2. Constant Arrays
         1. You can change the elements of a constant array:

// You can create a constant array:  
const cars = ["Saab", "Volvo", "BMW"];  
// You can change an element:  
cars[0] = "Toyota";  
// You can add an element:  
cars.push("Audi");

* + - * 1. But you can NOT reassign the array:

const cars = ["Saab", "Volvo", "BMW"];  
cars = ["Toyota", "Volvo", "Audi"]; // ERROR

* + - 1. Constant Objects
         1. You can change the properties of a constant object:

// You can create a const object:  
const car = {type:"Fiat", model:"500", color:"white"};  
// You can change a property:  
car.color = "red";  
// You can add a property:  
car.owner = "Johnson";

* + - * 1. But you can NOT reassign the object:

const car = {type:"Fiat", model:"500", color:"white"};  
car = {type:"Volvo", model:"EX60", color:"red"}; // ERROR

* + - 1. Block Scope
         1. Declaring a variable with const is similar to let when it comes to Block Scope.
         2. The x declared in the block, in this example, is not the same as the x declared outside the block:

const x = 10;  
// Here x is 10  
{  
const x = 2;  
// Here x is 2  
}  
// Here x is 10

* + - * 1. Redeclaring an existing var or let variable to const, in the same scope, is not allowed:

var x = 2; // Allowed  
const x = 2; // Not allowed  
  
{  
let x = 2; // Allowed  
const x = 2; // Not allowed  
}  
  
{  
const x = 2; // Allowed  
const x = 2; // Not allowed  
}

* + - * 1. Reassigning an existing const variable, in the same scope, is not allowed:

const x = 2; // Allowed  
x = 2; // Not allowed  
var x = 2; // Not allowed  
let x = 2; // Not allowed  
const x = 2; // Not allowed  
  
{  
 const x = 2; // Allowed  
 x = 2; // Not allowed  
 var x = 2; // Not allowed  
 let x = 2; // Not allowed  
 const x = 2; // Not allowed  
}

* + - * 1. Redeclaring a variable with const, in another scope, or in another block, is allowed:

const x = 2; // Allowed  
  
{  
 const x = 3; // Allowed  
}  
  
{  
 const x = 4; // Allowed  
}

* 1. Maths:
     1. Adding Strings and Numbers
        1. var x = "5" - 2 + 3;  
           document.getElementById(‘demo’).innerHTML = x; // 6
        2. var x = "5" - 2 - 3;  
           document.getElementById(‘demo’). innerHTML = x; // 0
        3. var x = "5" + 2 + 3;  
           document.getElementById(‘demo’). innerHTML = x; // 523
        4. var x = 2 + 3 + “5”;  
           document.getElementById(‘demo’). innerHTML = x; // 55
     2. JavaScript Comparison Operators

1. PRF192x\_o5 - Thao tác với câu lệnh rẽ nhánh.
   1. if (new Date().getHours() < 18){  
       document.getElementById('demo').innerHTML = "Good Day!";  
       } else {  
       document.getElementById('demo').innerHTML = "Good Evening";  
       }
2. PRF192x\_o6 - Hiểu cách thao tác với String và Number
3. PRF192x\_o7 - Thao tác với vòng lặp
   1. for - loops through a block of code a number of times
      1. const cars = ["BMW", "Volvo", "Saab", "Ford", "Fiat", "Audi"];  
          let text = "";  
          for (let i = 0; i < cars.length; i++){  
          text += [i + 1] + ". " + cars[i] + "<br>";   
          };  
          document.getElementById('demo').innerHTML = text;
      2. for (let i = 0, len = cars.length, text = ""; i < len; i++) {  
          text += cars[i] + "<br>";  
         }
   2. for/in - loops through the properties of an object
      1. const person = {  
          fname:"John",   
          lname:"Doe",   
          age:25  
          };  
          let txt = "";  
          for (let x in person) {  
          txt += person[x] + " ";  
          };  
          document.getElementById('demo').innerHTML = txt;
      2. const numbers = [45, 4, 9, 16, 25];  
          let txt = "";  
          numbers.forEach(myFunction);  
          document.getElementById('demo').innerHTML = txt;  
          function myFunction(value, index, array) {  
          txt += value + "<br>";  
          }
   3. for/of - loops through the values of an iterable object
      1. const car = ["BMW", "Volvo", "Mini"];  
          let txt = "";  
          for (let x of car) {  
          txt += x + "<br>";  
          };  
          document.getElementById('demo').innerHTML = txt;
   4. while - loops through a block of code while a specified condition is true
      1. let txt = "";  
          let i = 0;  
          while (i < 10){  
          txt += "<br>The number is " + i;  
          i++;  
          };  
          document.getElementById('demo').innerHTML = txt;
   5. do/while - also loops through a block of code while a specified condition is true
      1. let txt = "";  
          let i = 1;  
          do {  
          txt += "<br>The number is " + i;  
          i++;   
          }  
          while (i < 10);  
          document.getElementById('demo').innerHTML = txt;
4. PRF192x\_o8 - Hiểu và vận dụng được về functions trong JavaScript
   1. Function Return
      1. When JavaScript reaches a return statement, the function will stop executing.
      2. If the function was invoked from a statement, JavaScript will "return" to execute the code after the invoking statement.
      3. Functions often compute a return value. The return value is "returned" back to the "caller":
         1. var x = myFunction(4, 3); document.getElementById('demo').innerHTML = x;  
             function myFunction(a, b){  
             return a \* b;  
             }
         2. function toCelsius(f){  
             return (5 / 9) \* (f - 32);  
             }  
             document.getElementById('demo').innerHTML = "The temperature is " + toCelsius(77) + " Celsius";
5. PRF192x\_o9 - Hiểu và vận dụng được các loại sự kiện (event) trong JavaScript.
   1. onchange An HTML element has been changed
   2. onclick The user clicks an HTML element
   3. onmouseover The user moves the mouse over an HTML element
   4. onmouseout The user moves the mouse away from an HTML element
   5. onkeydown The user pushes a keyboard key
   6. onload The browser has finished loading the page
6. PRF192x\_o10 - Hiểu và vận dụng được mảng (array) trong JavaScript.
   1. Creating an Array
      1. const cars = [];  
          cars[0] = "Volvo";  
          cars[1] = "BMW";  
          document.getElementById('demo').innerHTML = cars;
   2. Accessing Array Elements
      1. let x = cars[0];
   3. Accessing the Last Array Element
      1. const fruits = ["Banana", "Orange", "Apple", "Mango"];  
         fruits[fruits.length - 1]; // Returns "Mango"
   4. Looping Array Elements
      1. const fruits = ["Banana", "Orange", "Apple", "Mango"];  
          let txt = "<ul>";  
          for (let i = 0; i < fruits.length; i++){  
          txt += "<li>" + fruits[i] + "</li>";  
          }  
          txt += "</ul>";  
          document.getElementById('demo').innerHTML = txt;
      2. const fruits = ["Banana", "Orange", "Apple", "Mango"];  
          let txt = "<ul>";  
          fruits.forEach(myFunction);  
          txt += "</ul>";  
          document.getElementById('demo').innerHTML = txt;  
          function myFunction(value){  
          txt += "<li>" + value + "</li>";  
          }
   5. Adding Array Elements
      1. const fruits = ["Banana", "Orange", "Apple"];  
         fruits.push("Lemon"); // Adds a new element (Lemon) to fruits
      2. const fruits = ["Banana", "Orange", "Apple"];  
         fruits[fruits.length] = "Lemon"; // Adds "Lemon" to fruits
7. PRF192x\_o11 - Hiểu và vận dụng được đối tượng (object) trong JavaScript.
   1. const car = {type:"Fiat", model:"500", color:"white"};
   2. Accessing Object Properties
      1. objectName.propertyName
      2. objectName["propertyName"]
   3. Object Methods
      1. const person ={  
          firstName: "James",  
          lastName: "High",  
          id: 5566,  
          fullname: function(){  
          return this.firstName + " " + this.lastName;  
          }  
          }  
          document.getElementById('demo').innerHTML = person.fullname();
8. PRF192x\_o12 - Hiểu và vận dụng được AJAX trong JavaScript.
   1. function loadDoc() {  
      // Define a callback function  
       const xhttp = new XMLHttpRequest();  
      // Define a callback function  
       xhttp.onload = function() {  
       document.getElementById("demo").innerHTML = this.responseText;  
       }  
      // Send a request   
       xhttp.open("GET", "ajax\_info.txt", true);  
       xhttp.send();  
      }
9. PRF192x\_o13 - Hiểu và sử dụng được cấu trúc jQuery.
   1. Finding HTML Element by Id
      1. $("#id01"); == document.getElementById("id01");
   2. Finding HTML Elements by Tag Name
      1. $("p"); == document.getElementsByTagName("p");
   3. Finding HTML Elements by Class Name
      1. $(".intro"); == document.getElementsByClassName("intro");
   4. $(document).ready(function(){  
       $("p").click(function(){  
       $(this).hide();  
       });  
      });
10. PRF192x\_o14 - Sử dụng được Selector và Functions của jQuery
    1. jQuery Selectors
       1. $("\*") Selects all elements
       2. $(this) Selects the current HTML element
       3. $("p.intro") Selects all <p> elements with class="intro"
       4. $("p:first") Selects the first <p> element
       5. $("ul li:first") Selects the first <li> element of the first <ul>
       6. $("ul li:first-child") Selects the first <li> element of every <ul>
       7. $("[href]") Selects all elements with an href attribute
       8. $("a[target='\_blank']") Selects all <a> elements with a target attribute value equal to "\_blank"
       9. $("a[target!='\_blank']") Selects all <a> elements with a target attribute value NOT equal to "\_blank"
       10. $(":button") Selects all <button> elements and <input> elements of type="button"
       11. $("tr:even") Selects all even <tr> elements
       12. $("tr:odd") Selects all odd <tr> elements
    2. Functions:
11. PRF192x\_o15 - Sử dụng được thẻ HTML động với jQuery.
    1. Get Content - text(), html(), and val()
       1. text() - Sets or returns the text content of selected elements
          1. $("#btn1").click(function(){  
              alert("Text: " + $("#test").text());  
             });
       2. html() - Sets or returns the content of selected elements (including HTML markup)
          1. $("#btn2").click(function(){  
              alert("HTML: " + $("#test").html());  
             });
       3. val() - Sets or returns the value of form fields
          1. $("#btn1").click(function(){  
              alert("Value: " + $("#test").val());  
             });
    2. Get Attributes - attr()
       1. $("button").click(function(){  
           alert($("#w3s").attr("href"));  
          });
    3. Set Content - text(), html(), and val()
       1. text() - Sets or returns the text content of selected elements
          1. $("#btn1").click(function(){  
              $("#test1").text("Hello world!");  
             });
       2. html() - Sets or returns the content of selected elements (including HTML markup)
          1. $("#btn2").click(function(){  
              $("#test2").html("<b>Hello world!</b>");  
             });
       3. val() - Sets or returns the value of form fields
          1. $("#btn3").click(function(){  
              $("#test3").val("Dolly Duck");  
             });
    4. Set Attributes - attr()
       1. $("button").click(function(){  
           $("#w3s").attr({  
           "href" : "https://www.w3schools.com/jquery/",  
           "title" : "W3Schools jQuery Tutorial"  
           });  
          });
    5. jQuery - Add Elements
       1. append() - Inserts content at the end of the selected elements
          1. $("p").append("Some appended text.");
       2. prepend() - Inserts content at the beginning of the selected elements
          1. $("p").prepend("Some prepended text.");
       3. after() - Inserts content after the selected elements
          1. $("img").after("Some text after");
       4. before() - Inserts content before the selected elements
          1. $("img").before("Some text before");
    6. jQuery - Remove Elements
       1. remove() - Removes the selected element (and its child elements)
          1. $("#div1").remove();
          2. $("p").remove(".test, .demo");
       2. empty() - Removes the child elements from the selected element
          1. $("#div1").empty();
12. PRF192x\_o16 - Làm việc với Sự kiện (Events) trong jQuery.
    1. Mouse Events
       1. Click
       2. Dbclick
       3. Mouseenter
       4. Mouseleave
       5. hover()
          1. $("#p1").hover(function(){  
              alert("You entered p1!");  
             },
    2. Keyboard Events
       1. Keypress
       2. Keydown
       3. Keyup
    3. Form Events
       1. Submit
       2. Change
       3. Focus
       4. Blur
    4. Document/Window Events
       1. Load
       2. Resize
       3. Scroll
       4. unload
13. PRF192x\_o17 - Làm việc với Hiệu ứng trong jQuery.
    1. jQuery toggle()
       1. $("button").click(function(){  
           $("p").toggle();  
          });
    2. jQuery Fading Methods
       1. fadeIn()
       2. fadeOut()
       3. fadeToggle()
       4. fadeTo()
    3. jQuery Sliding Methods
       1. slideDown()
       2. slideUp()
       3. slideToggle()
    4. jQuery animate()
       1. $("button").click(function(){  
           $("div").animate({  
           left: '250px',  
           height: '+=150px',  
           width: '+=150px'  
           });  
          });
14. PRF192x\_o18 - Vận dụng được cách chuẩn hóa dữ liệu Form của jQuery.
    1. JavaScript Form Validation
       1. function validateForm() {  
           let x = document.forms["myForm"]["fname"].value;  
           if (x == "") {  
           alert("Name must be filled out");  
           return false;  
           }  
          }
    2. JavaScript Validation API
       1. The checkValidity() Method
          1. <input id="id1" type="number" min="100" max="300" required>  
             <button onclick="myFunction()">OK</button>  
               
             <p id="demo"></p>  
               
             <script>  
             function myFunction() {  
              const inpObj = document.getElementById("id1");  
              if (!inpObj.checkValidity()) {  
              document.getElementById("demo").innerHTML = inpObj.validationMessage;  
              }  
             }  
             </script>
       2. The rangeOverflow Property
          1. <input id="id1" type="number" max="100">  
             <button onclick="myFunction()">OK</button>  
               
             <p id="demo"></p>  
               
             <script>  
             function myFunction() {  
              let text = "Value OK";  
              if (document.getElementById("id1").validity.rangeOverflow) {  
              text = "Value too large";  
              }  
             }  
             </script>
       3. The rangeUnderflow Property
          1. <input id="id1" type="number" min="100">  
             <button onclick="myFunction()">OK</button>  
               
             <p id="demo"></p>  
               
             <script>  
             function myFunction() {  
              let text = = "Value OK";  
              if (document.getElementById("id1").validity.rangeUnderflow) {  
              text = "Value too small";  
              }  
             }  
             </script>
15. PRF192x\_o19 - Sử dụng được cấu trúc JSON với jQuery