For each of the prompts, write the answer to the prompt in Python on the left side, write the answer to the prompt in Javascript on the right side. The best way to do this is to write a small program in the appropriate IDE, then copy/paste the *tested and working* code into the appropriate box.

1. Declare a Variable with a value. Reassign a variable’s value

|  |  |
| --- | --- |
| Y = 7 | Var y = 7 |

1. Assign the result of an expression to a variable.

|  |  |
| --- | --- |
| Y = 5 + 8 | Var y = 5 + 8 |

1. Change the data type of a variable’s value and assign it to the original value
   1. Convert to string
   2. Convert to number (integer or float)

|  |  |
| --- | --- |
| x=int(x)  x=str(x) | var x=int(x)  var x=str(x) |

1. Create a conditional statement that branches depending on if a variable’s value is greater than or equal to 7, greater than 4, or neither

|  |  |
| --- | --- |
| if x >= 7:    pass  elif x > 4:    pass  else:    print("boo") | if (x >= 7) {          console.log("boo")      } else if (x > 4){          console.log("cherry")      }else {          console.log("bananana")      } |

1. Create a conditional statement that only branches if the variable’s value is ‘blue’ OR ‘green’

|  |  |
| --- | --- |
| if x = "blue" or x = "green":        pass | If (x = “blue” || x = “green”) {          Break          } |

1. Create a conditional statement that only branches if the variable’s value is ‘black’ AND ‘yellow’

|  |  |
| --- | --- |
| if x = "black" and x = "yellow":  pass | If (x= “blue” && x = “yellow”) {      Break      } |

1. Create a function that outputs “Hello” to the console

|  |  |
| --- | --- |
| def function ():              print("hello!") | function sayHello() {                  console.log("Hello!")    } |

1. Create a function that returns “Hello”

|  |  |
| --- | --- |
| def function ():    return "hello" | function printHello() {      return "Hello"  } |

1. Create a function that takes two arguments, adds their values together, assigns the new value to a local variable, and returns that variable

|  |  |
| --- | --- |
| def printHello(fish, snail):  animal = fish + snail      return animal | function printHello(fish, snail) {      let animal = fish + snail      return animal  }  printHello(5, 6) |

1. Create a loop that repeats while a variable’s value is true

|  |  |
| --- | --- |
| while x == true:      continue | while(x == true) {              console.log ("continue")          } |

1. Create a loop that repeats while a variable’s value is true, but will break mid-loop if another variable equals false

|  |  |
| --- | --- |
| while x == true:           continue          if y == false:          break | while (x == true) {              console.log("continue")             if (y == false) {              break;           }           } |

1. Create an array/list that contains three strings
2. Using the array from #12, remove the last element in the array

|  |  |
| --- | --- |
| list = ["Macadamia", "Pistachio", "Hazelnut"] | let list = ["Macadamia", "Pistachio", "Hazelnut"] |

1. Using the array from #12, remove the first element in the array

|  |  |
| --- | --- |
| list.pop(0) | list.shift() |

1. Using the array from #12, remove the element in the middle (index 1)

|  |  |
| --- | --- |
| list.pop(1) | list.splice(1,1) |

1. Using the array from #12, output the first element

|  |  |
| --- | --- |
| print(list[0]) | console.log(list[0]) |

1. Create an empty array. Then add a string element to it

|  |  |
| --- | --- |
| list = []  list.append("pecan") | var list = []  list.push("pecan") |

1. Using the array from #12, use a loop to output each element

|  |  |
| --- | --- |
| for item in list:  print(item) | for (item in list){      console.log (item)   } |

1. Using the array from #12, use a loop to output the index of each element

|  |  |
| --- | --- |
| for index, val in enumerate(list):    print(index, val) | for (var i = 0; i < list.length; i++) {          console.log(i);      } |

1. Create an object with one property and one method. The property value will be a string, and the method will output the value of this object’s property.

|  |  |
| --- | --- |
| class Restaurant:  def \_\_init\_\_(self):  self.menuItem1 = "Sushi"  def sayMenuItem1(self):  print(self.menuItem1) | const object = {    property: "Sushi",    method: function() {      console.log(this.property)    };  }; |

1. Using the object in #20, print the name of each property/method key in the object

|  |  |
| --- | --- |
| # Use nested loops to iterate through the array  for i in range(rows):      for j in range(cols):          print(two\_dim\_array[i][j], end=" ")  # Print each element      print()  # Move to the next row      class MyClass:      def \_\_init\_\_(self):          self.property1 = "Hello"          self.property2 = 42      def method1(self):          return "Method 1 called"      def method2(self):          return "Method 2 called"  # Instantiate an object  my\_object = MyClass()  # Get the list of attributes and methods using dir()  attributes\_and\_methods = dir(my\_object)  # Filter and print custom attributes and methods  for item in attributes\_and\_methods:      if not item.startswith("\_\_"):  # Exclude built-in attributes/methods          print(item) | # Filter and print custom attributes and methods  for item in attributes\_and\_methods:      if not item.startswith("\_\_"):  # Exclude built-in attributes/methods          print(item)  class MyClass {              constructor() {                  this.property1 = "Hello";                  this.property2 = 42;              }              method1() {                  return "Method 1 called";              }              method2() {                  return "Method 2 called";              }          }          // Instantiate an object          const myObject = new MyClass();          // Get the keys (property/method names) using a for...in loop          for (const key in myObject) {              if (myObject.hasOwnProperty(key)) {                  console.log(key);              }          } |

1. Create a variable whose value is a string equal to the name (key) of the property of the object in #20. Output the property value of the object by using the variable (not the property name)

|  |  |
| --- | --- |
| rest = Restaurant()  objProp = ‘menuItem1’  print(rest[objProp]) | Const rest = restaurant();  Objprop = ‘menuItem1’  Console.log(rest[objProp]) |

1. Create a two-dimensional array
   1. Create an array/list with three elements
   2. Each element is another array/list with three elements whose values are strings

|  |  |
| --- | --- |
| # Access an element in the array  element = apples[row\_index][bananas] | // Create a 3x3 two-dimensional array filled with values  const rows = 3;  const cols = 3;  const twoDimArray = [];  // Populate the array with values  for (let i = 0; i < rows; i++) {      const row = [];      for (let j = 0; j < cols; j++) {          row.push(i \* cols + j + 1);  // Filling with incrementing numbers      }      twoDimArray.push(row);  }  // Print the array  for (let i = 0; i < rows; i++) {      console.log(twoDimArray[i]);  } |

1. Using the array in #23, use two loops (one inside the other) to output all 9 elements in the multi-dimensional array.

|  |  |
| --- | --- |
| # Create a 3x3 two-dimensional array filled with numbers  rows = 3  cols = 3  two\_dim\_array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]  # Use nested loops to iterate through the array  for i in range(rows):  for j in range(cols):  print(two\_dim\_array[i][j], end=" ") # Print each element  print() # Move to the next row | // Create a 3x3 two-dimensional array filled with values  const rows = 3;  const cols = 3;  const twoDimArray = [      [1, 2, 3],      [4, 5, 6],      [7, 8, 9]  ];  // Use nested loops to iterate through the array  for (let i = 0; i < rows; i++) {      for (let j = 0; j < cols; j++) {          console.log(twoDimArray[i][j]);      }  } |

1. Add a new property to the object used in #20 with a number value

|  |  |
| --- | --- |
| class MyClass:  def \_\_init\_\_(self, initial\_value):  self.existing\_property = initial\_value  # Instantiate an object  my\_object = MyClass(42)  # Add a new property with a number value  my\_object.new\_number\_property = 100  # Access and print the properties  print("Existing Property:", my\_object.existing\_property)  print("New Number Property:", my\_object.new\_number\_property) | // Create an object  const myObject = {      existingProperty: "Hello"  };  // Add a new property with a number value  myObject.newNumberProperty = 100;  // Access and print the properties  console.log("Existing Property:", myObject.existingProperty);  console.log("New Number Property:", myObject.newNumberProperty); |