**Basic JavaScript**

**1)Comments** //abcde

/\*ab  
 cd  
 d\*/

**2)Declare javascript variable** var myName;

**3)Storing values using assignment operator** a = 7;

**4)Assigning value of one variable to another** b = a;

**5)Initializing variables with assignment operator** var a = 9;

**6)Declare string variables** var myFirstName = “Krishna”;  
 var myLastName = “Adhia”;

**7)Understanding uninitialized variables** var a = 5;  
 var b = 10;  
 var c = “I am a”;  
 //only change code above this line  
 a = a + 1;  
 b = b + 5;  
 c = c + “String!”;

**8)Understanding case sensitive in variables** // Variable declarations  
 var studlyCapVar;  
 var properCamelCase;  
 var titleCaseOver;

// Variable assignments  
 studlyCapVar = 10;  
 properCamelCase = "A String";  
 titleCaseOver = 9000;

**9)let vs var** let catName = “Oliver”;  
 let catSound = “Meow!”;

**10)Declare read only variables with const** const FCC = "freeCodeCamp"; // Change this line  
 let fact = "is cool!"; // Change this line  
 fact = "is awesome!";  
 console.log(FCC, fact); // Change this line

**11)Add two number with javascript** //changed right side number from 0 to 10const sum = 10 + 10;

**12)Subtract** //changed right side number from 0 to 33const sum = 45 - 33;

**13)Multiplication** //changed right side number from 0 to10  
 const product = 8 \* 10;

**14) Division** //changed right side number from 0 to 33 so that quotient will be 2  
 const product = 66 / 33;

**15)Incrementing number** //incrementing using ++let myVar = 87;

//only change code below this line  
 myVar++;

**16)Decrementing number** //decrementing using --let myVar = 11;

//only change code below this line  
 myVar--;

**17)Creating decimal number** const ourDecimal = 5.7;

// Only change code below this line  
 let myDecimal = 5.7;

**18)Multiply two decimal** //change right number from 0 to something such that product = 5.0  
 const product = 2.0 \* 2.5;

**19)Divide two decimal** const quotient = 4.4 / 2.0; // Change this line

**20)Modulo** //use 11 divide by 3, reminder should be 3  
 const remainder = 11%3;

**21)Compound assignment with augmented addition** let a = 3;  
 let b = 17;  
 let c = 12;

// Only change code below this line

a+=12;  
 b+=9;  
 c+=7;

**22) Compound assignment with augmented subtraction** let a = 11;  
 let b = 9;  
 let c = 3;

// Only change code below this line  
 a-=6;  
 b-=15;  
 c-=1;

**23) Compound assignment with augmented multiplication** let a = 5;  
 let b = 12;  
 let c = 4.6;

// Only change code below this line  
 a\*=5;  
 b\*=3;  
 c\*=10;

**24) Compound assignment with augmented division**

let a = 48;  
let b = 108;  
let c = 33;

// Only change code below this line  
a/=12;  
b/=4;7  
c/=11;

**25)Escaping literal quotes in string** const myStr = "I am a \"double quoted\" string inside \"double quotes.\""; // Change  
 this line

console.log(myStr);

**26)Quoting strings with single quotes** const myStr = '<a href="http://www.example.com" target="\_blank">Link</a>';

**27)Escape sequence in string** const myStr = "FirstLine\n\t\\SecondLine\nThirdLine"; // Change this line

**28)String Concatenating**

const myStr = "This is the start." + " This is the end."; // Change this line

**29)Concatenate string with +=** let myStr = "This is the first sentence.";  
 myStr += " This is the second sentence.";

**30)String with variables** // Only change code below this line  
 const myName = "Krishna";  
 const myStr = "My name is " + myName + " and I am well!";

**31)Append variables to string** // Change code below this line  
 const someAdjective = "Awesome";  
 let myStr = "Learning to code is ";  
 myStr += someAdjective;

**32)Find Length** // Setup  
 let lastNameLength = 0;  
 const lastName = "Lovelace";

// Only change code below this line  
 lastNameLength = lastName.length;  
 console.log(lastNameLength);

**33)Bracket notation to find first character in string** // Setup  
 let firstLetterOfLastName = "";  
 const lastName = "Lovelace";

// Only change code below this line

firstLetterOfLastName = lastName[0]; // Change this line  
 console.log(firstLetterOfLastName);

**34)Understanding String immutability** // Setup  
 let myStr = "Jello World";

// Only change code below this line  
 myStr= "Hello World"; // Change this line

// Only change code above this line

**35)Use bracket notation to find the Nth character in the string** // Setup

const lastName = "Lovelace";

// Only change code below this line

const thirdLetterOfLastName = lastName[2]; // Change this line  
 console.log(thirdLetterOfLastName);

**36) Use bracket notation to find the last character in the string** // Setup

const lastName = "Lovelace";

// Only change code below this line

const lastLetterOfLastName = lastName[lastName.length - 1]; // Change this line

**37) Use bracket notation to find the Nth-to-last character in the string** // Setup

const lastName = "Lovelace";

// Only change code below this line

const secondToLastLetterOfLastName = lastName[lastName.length - 2]; // Change this line

console.log(secondToLastLetterOfLastName);

**38)Word Blanks** const myNoun = "dog";  
 const myAdjective = "big";  
 const myVerb = "ran";  
 const myAdverb = "quickly";

// Only change code below this line  
 let wordBlanks ="A " + myAdjective + " " + myNoun + " " + myVerb + " " + myAdverb + "."; // Change this line

// Only change code above this line

**39)JavaScript array – first element should be string and second be a number** // Only change code below this line  
 const myArray = ["One",1];

**40)Nested Array** // Only change code below this line  
 const myArray = [["Red","Yellow"]];

**41)Accessing array elements – myData should be equal to the first element of array** const myArray = [50, 60, 70];  
 let myData = myArray[0];

**42)Modify array using index – change the 1st element to 45** // Setup  
 const myArray = [18, 64, 99];  
 myArray[0] = 45;  
 // Only change code below this line

**43)Accessing multidimensional array using index – ans should be 8** const myArray = [  
 [1, 2, 3],  
 [4, 5, 6],  
 [7, 8, 9],  
 [[10, 11, 12], 13, 14],  
 ];

const myData = myArray[2][1];

**44)Manipulate array using push** // Setup  
 const myArray = [["John", 23], ["cat", 2]];

// Only change code below this line  
 myArray.push(["dog",3]);

**45)Manipulate array using pop** // Setup  
 const myArray = [["John", 23], ["cat", 2]];

// Only change code below this line  
 let removedFromMyArray = myArray.pop();

**46)Manipulate array using shift**  // Setup  
 const myArray = [["John", 23], ["dog", 3]];

// Only change code below this line  
 let removedFromMyArray = myArray.shift();

**47)Manipulate array using unshift** // Setup  
 const myArray = [["John", 23], ["dog", 3]];  
 myArray.shift();

// Only change code below this line  
 myArray.unshift(["Paul",35]);

**48)Shopping list – add at least 5 elements as subarray which contains first string name of item and second should be the number of products** const myList = [["Choclate",10],["Strawberries",10],["Mango",25],["Ice Cream",2],["Sweets",20]];

**49)Write reusable javascript with function** function reusableFunction()  
 {  
 console.log("Hi World");  
 }

reusableFunction();

**50)Passing values to functions with arguments** function functionWithArgs(n1,n2)  
 {  
 console.log(n1+n2);  
 }

functionWithArgs(1,2);

functionWithArgs(7,9);

**51)Return a value from a function with return** function timesFive(n1)  
 {  
 return 5\*n1;  
 }

timesFive(5);

timesFive(2);

timesFive(0);

**52)Global Scope and Functions** // Declare the myGlobal variable below this line  
 let myGlobal = 10;  
 function fun1() {  
 // Assign 5 to oopsGlobal Here  
 oopsGlobal = 5;

}

// Only change code above this line  
function fun2()

{  
 var output = "";  
 if (typeof myGlobal != "undefined") {  
 output += "myGlobal: " + myGlobal;  
 }

if (typeof oopsGlobal != "undefined") {  
 output += " oopsGlobal: " + oopsGlobal;  
 }

console.log(output);  
}

**53)Local scope and functions** function myLocalScope()

{   
 // Only change code below this line  
 var myVar = 10;  
 console.log('inside myLocalScope', myVar);  
 }

myLocalScope();

// Run and check the console

// myVar is not defined outside of myLocalScope

console.log('outside myLocalScope', myVar);

**54)Global vs Local scope in functions** // Setup  
 const outerWear = "T-Shirt";

function myOutfit()

{

// Only change code below this line  
 var outerWear = "sweater";

// Only change code above this line  
 return outerWear;

}

myOutfit();

**55)Understanding undefined value returned from a function** // Setup  
 let sum = 0;

function addThree()

{  
 sum = sum + 3;

}

// Only change code below this line

function addFive()

{  
 sum = sum + 5;

}

// Only change code above this line

addThree();

addFive();

**56)Assignment with a returned value** // Setup  
 let processed = 0;

function processArg(num)   
{  
 return (num + 3) / 5;  
}

processed = processArg(7);

// Only change code below this line

**57)Stand in line** function nextInLine(arr, item)   
 {  
 // Only change code below this line  
 let insertedelement = arr.push(item);  
 let removedelement = arr.shift();  
 return removedelement;  
 // Only change code above this line

}  
 // Setup  
 const testArr = [1, 2, 3, 4, 5];  
 // Display code  
 console.log("Before: " + JSON.stringify(testArr));  
 console.log(nextInLine(testArr, 6));  
 console.log("After: " + JSON.stringify(testArr));  
 nextInLine([],5);  
 nextInLine([],1);  
 nextInLine([2],1);  
 nextInLine([5,6,7,8,9],1);  
 nextInLine(testArr,10);  
 console.log(testArr[4]);

**58)Understanding Boolean values** function welcomeToBooleans()

{

// Only change code below this line

return true; // Change this line

// Only change code above this line

}

**59)Use conditional logic with if statements** function trueOrFalse(wasThatTrue)

{

// Only change code below this line  
 if(wasThatTrue)  
 {  
 return "Yes, that was true";  
 }  
 return "No, that was false";

// Only change code above this line

}

trueOrFalse(true);

trueOrFalse(false);

**60)Comparison with equality operator** // Setup

function testEqual(val)

{

if (val == 12)

{ // Change this line  
return "Equal";

}

return "Not Equal";

}

testEqual(10);

testEqual(12);

**61)Comparison with strict equality operator** // Setup

function testStrict(val)   
{

if (val === 7)  
 { // Change this line  
 return "Equal";  
 }  
 return "Not Equal";  
}

testStrict(10);  
testStrict(7);  
testStrict("7");

**62)Practice comparing different values** // Setup  
 function compareEquality(a, b)  
 {  
 if (a === b)   
 { // Change this line  
 return "Equal";  
 }

return "Not Equal";  
 }

compareEquality(10, "10");  
compareEquality(20, "20");

**63)Comparison with the inequality operator** // Setup

function testNotEqual(val)   
{  
 if (val!=99)   
 { // Change this line  
 return "Not Equal";  
 }  
 return "Equal";  
}

testNotEqual(99);  
testNotEqual("99");  
testNotEqual(12);  
testNotEqual("12");  
testNotEqual("bob");

**64)Comparison with the strict inequality operator  
 // Setup** function testStrictNotEqual(val)  
 {  
 if (val !== 17)   
 { // Change this line  
 return "Not Equal";  
 }  
 return "Equal";  
 }

testStrictNotEqual(17);  
testStrictNotEqual("17");  
testStrictNotEqual(12);  
testStrictNotEqual("bob");

**65)Comparison with the greater than operator** function testGreaterThan(val)   
 {

if (val>100)   
 { // Change this line  
 return "Over 100";  
 }

if (val>10)  
 { // Change this line  
 return "Over 10";  
 }  
 return "10 or Under";  
 }

testGreaterThan(0);  
testGreaterThan(10);  
testGreaterThan(11);  
testGreaterThan(99);  
testGreaterThan(100);  
testGreaterThan(101);  
testGreaterThan(150);

**66) Comparison with the greater than or equal to operator** function testGreaterOrEqual(val)   
 {  
 if (val>=20)   
 { // Change this line  
 return "20 or Over";  
 }  
 if (val>=10)   
 { // Change this line  
 return "10 or Over";  
 }  
 return "Less than 10";  
 }

testGreaterOrEqual(0);  
testGreaterOrEqual(9);  
testGreaterOrEqual(10);  
testGreaterOrEqual(11);  
testGreaterOrEqual(19);  
testGreaterOrEqual(100);  
testGreaterOrEqual(21);

**67) Comparison with the less than or equal to operator** function testLessThan(val)   
 {  
 if (val<25)   
 { // Change this line  
 return "Under 25";  
 }

if (val<55)   
 { // Change this line  
 return "Under 55";  
 }  
 return "55 or Over";  
 }

testLessThan(0);  
testLessThan(24);  
testLessThan(25);  
testLessThan(54);  
testLessThan(55);  
testLessThan(99);

**68) Comparison with the less than or equal to operator** function testLessOrEqual(val)   
 {  
 if (val<=12)   
 { // Change this line  
 return "Smaller Than or Equal to 12";  
 }  
 if (val<=24)   
 { // Change this line  
 return "Smaller Than or Equal to 24";  
 }  
 return "More Than 24";  
 }

testLessOrEqual(0);  
testLessOrEqual(11);  
testLessOrEqual(12);  
testLessOrEqual(23);  
testLessOrEqual(24);  
testLessOrEqual(25);  
testLessOrEqual(55);

**69)Comparison with logical AND operator** function testLogicalAnd(val)   
 {  
 // Only change code below this line  
 if (val <= 50 && val>=25)  
 {  
 return "Yes";  
 }  
 // Only change code above this line  
 return "No";  
 }

testLogicalAnd(0);  
testLogicalAnd(24);  
testLogicalAnd(25);  
testLogicalAnd(30);  
testLogicalAnd(50);  
testLogicalAnd(51);  
testLogicalAnd(75);  
testLogicalAnd(80);

**70)Comparison with logical OR operator** function testLogicalOr(val)   
 {  
 // Only change code below this line  
 if (val>20 || val<10)   
 {  
 return "Outside";  
 }  
 // Only change code above this line  
 return "Inside";  
 }

testLogicalOr(0);  
testLogicalOr(9);  
testLogicalOr(10);  
testLogicalOr(15);  
testLogicalOr(19);  
testLogicalOr(20);  
testLogicalOr(21);  
testLogicalOr(25);

**71)Introduction to else statement** function testElse(val)   
 {  
 let result = "";  
 // Only change code below this line  
 if (val > 5)   
 {  
 result = "Bigger than 5";  
 }  
 else  
 {  
 result = "5 or Smaller";  
 }  
 // Only change code above this line  
 return result;  
 }

testElse(4);  
testElse(5);  
testElse(6);  
testElse(10);

**72)Introducing elseif statement** function testElseIf(val)   
 {  
 if (val > 10)   
 {  
 return "Greater than 10";  
 }  
 else if (val < 5)   
 {  
 return "Smaller than 5";  
 }  
 else  
 {  
 return "Between 5 and 10";  
 }  
 }

testElseIf(0);  
testElseIf(5);  
testElseIf(7);  
testElseIf(10);  
testElseIf(12);

**73)Logical order in if else statements** function orderMyLogic(val)   
 {  
 if (val < 5)   
 {  
 return "Less than 5";

}   
else if (val < 10)   
{  
 return "Less than 10";

}   
else   
{  
 return "Greater than or equal to 10";  
}

}

orderMyLogic(4);  
orderMyLogic(6);  
orderMyLogic(11)**;**

**74)Chaining if else statements** function testSize(num)   
 { // Only change code below this line  
 if(num<5)  
 {  
 return "Tiny";  
 }  
 else if(num<10)  
 {  
 return "Small";  
 }  
 else if(num<15)  
 {  
 return "Medium";  
 }  
 else if(num<20)  
 {  
 return "Large";  
 }  
 else  
 {  
 return "Huge";  
 }  
 return "Change Me";  
 // Only change code above this line  
 }

testSize(0);  
testSize(4);  
testSize(5);  
testSize(8);  
testSize(10);  
testSize(14);  
testSize(15);  
testSize(17);  
testSize(20);  
testSize(25);

**75)Golf Code**

const names = ["Hole-in-one!", "Eagle", "Birdie", "Par", "Bogey", "Double Bogey", "Go Home!"];

function golfScore(par, strokes) {

// Only change code below this line

if(strokes == 1)

{

return names[0];

}

else if(strokes <= par-2)

{

return names[1];

}

else if(strokes === par -1)

{

return names[2];

}

else if(strokes === par)

{

return names[3];

}

else if(strokes === par + 1)

{

return names[4];

}

else if(strokes === par+2)

{

return names[5];

}

else if(strokes>=par+3)

{

return names[6];

}

// Only change code above this line

}

golfScore(5, 4);

**76)Selecting from many options with switch statements** function caseInSwitch(val)   
 {  
 let answer = "";  
 // Only change code below this line  
 switch(val)  
 {  
 case 1:  
 answer = "alpha";  
 break;

case 2:  
 answer ="beta";  
 break;

case 3:  
 answer ="gamma";  
 break;  
 case 4:  
 answer ="delta";  
}

// Only change code above this line  
 return answer;

}  
caseInSwitch(1);  
caseInSwitch(2);  
caseInSwitch(3);  
caseInSwitch(4);

**77)Adding a default option in switch statements** function switchOfStuff(val)   
 {   
 let answer = "";  
 // Only change code below this line  
 switch(val)  
 {  
 case "a":  
 answer = "apple";  
 break;

case "b":  
 answer = "bird";  
 break;

case "c":  
 answer = "cat";  
 break;

default:  
 answer = "stuff";  
}  
 // Only change code above this line  
return answer;

}

switchOfStuff("a");  
switchOfStuff("b");  
switchOfStuff("c");  
switchOfStuff("d");  
switchOfStuff(4);

**78)Multiple identical options in switch statement**

function sequentialSizes(val)   
{  
 let answer = "";  
 // Only change code below this line  
 switch(val)  
 {  
 case 1:  
 case 2:  
 case 3:  
 answer = "Low";  
 break;  
 case 4:  
 case 5:  
 case 6:  
 answer = "Mid";  
 break;  
 case 7:  
 case 8:  
 case 9:  
 answer = "High";  
 break;

}

// Only change code above this line  
 return answer;

}

sequentialSizes(1);  
sequentialSizes(2);  
sequentialSizes(3);  
sequentialSizes(4);  
sequentialSizes(5);  
sequentialSizes(6);  
sequentialSizes(7);  
sequentialSizes(8);  
sequentialSizes(9);

**79)Replacing if else chains with switch**

function chainToSwitch(val) {

let answer = "";

// Only change code below this line

switch(val)

{

case "bob":

answer = "Marley";

break;

case 42:

answer = "The Answer";

break;

case 99:

answer = "Missed me by this much!";

break;

case 7:

answer = "Ate Nine";

break;

case 1:

answer = "There is no #1";

break;

default:

answer = "";

}

// Only change code above this line

return answer;

}

chainToSwitch(7);

**80)Returning Boolean values from functions** function isLess(a, b) {

// Only change code below this line

return a<b;

// Only change code above this line

}

isLess(10, 15);

**81)Return early pattern for functions**// Setup

function abTest(a, b) {

// Only change code below this line

if(a<0 ||b<0)

{

return undefined;

}

// Only change code above this line

return Math.round(Math.pow(Math.sqrt(a) + Math.sqrt(b), 2));

}

abTest(2,2);

**82)Counting cards**let count = 0;  
function cc(card) {

// Only change code below this line

switch(card)

{

case 2:

case 3:

case 4:

case 5:

case 6:

count++;

break;

case 7:

case 8:

case 9:

break;

case 10:

case 'J':

case 'Q':

case 'K':

case 'A':

count--;

break;

}

if(count >= 1)

{

return count + " Bet";

}

else if(count<=0)

{

return count + " Hold";

}

// Only change code above this line

}

cc(2); cc(3); cc(7); cc('K'); cc('A');

**83)Build Javascript objects**const myDog = {

// Only change code below this line

"name": "Rocky",

"legs": 4,

"tails": 1,

"friends": ["humans","cows"]

// Only change code above this line

};

**84)Accessing object properties with dot notation**// Setup

const testObj = {

"hat": "ballcap",

"shirt": "jersey",

"shoes": "cleats"

};

// Only change code below this line

const hatValue = testObj.hat; // Change this line

const shirtValue = testObj.shirt; // Change this line

**85)Accessing object properties with square bracket**// Setup

const testObj = {

"an entree": "hamburger",

"my side": "veggies",

"the drink": "water"

};

// Only change code below this line

const entreeValue = testObj["an entree"]; // Change this line

const drinkValue = testObj["the drink"]; // Change this line

**86)Accessing object properties with variables**// Setup

const testObj = {

12: "Namath",

16: "Montana",

19: "Unitas"

};

// Only change code below this line

const playerNumber = 16; // Change this line

const player = testObj[playerNumber]; // Change this line

console.log(player);

**87)Updating object properties**// Setup

const myDog = {

"name": "Coder",

"legs": 4,

"tails": 1,

"friends": ["freeCodeCamp Campers"]

};

// Only change code below this line

myDog["name"] = "Happy Coder";

**88)Add new properties to a javascript object**const myDog = {

"name": "Happy Coder",

"legs": 4,

"tails": 1,

"friends": ["freeCodeCamp Campers"]

};

myDog["bark"] = "woof";

**89)Delete properties from a javascript object**// Setup

const myDog = {

"name": "Happy Coder",

"legs": 4,

"tails": 1,

"friends": ["freeCodeCamp Campers"],

"bark": "woof"

};

// Only change code below this line

delete myDog.tails;

**90)Using objects for lookup**// Setup

function phoneticLookup(val) {

var result = "";

// Only change code below this line

var lookup ={

"alpha":"Adams",

"bravo": "Boston",

"charlie":"Chicago",

"delta": "Denver",

"echo":"Easy",

"foxtrot":"Frank"

}

// Only change code above this line

result = lookup[val];

return result;

}

phoneticLookup("charlie");

**91)Testing objects for properties**function checkObj(obj, checkProp) {

// Only change code below this line

if(obj.hasOwnProperty(checkProp))

{

return obj[checkProp];

}

else

{

return "Not Found";

}

// Only change code above this line

}

**92)Manipulating complex objects**const myMusic = [

{

"artist": "Billy Joel",

"title": "Piano Man",

"release\_year": 1973,

"formats": [

"CD",

"8T",

"LP"

],

"gold": true

},

{

"artist" : "ABC",

"title" : "BCD",

"release\_year" :2001,

"formats" :[

"mp3",

"mp4"

]

}

];

**93)Accessing nested objects**const myStorage = {

"car": {

"inside": {

"glove box": "maps",

"passenger seat": "crumbs"

},

"outside": {

"trunk": "jack"

}

}

};

const gloveBoxContents = myStorage.car.inside["glove box"];

**94)Accessing nested arrays**const myPlants = [

{

type: "flowers",

list: [

"rose",

"tulip",

"dandelion"

]

},

{

type: "trees",

list: [

"fir",

"pine",

"birch"

]

}

];

const secondTree = myPlants[1].list[1];

**95)Record Collection** // Setup

const recordCollection = {

2548: {

albumTitle: 'Slippery When Wet',

artist: 'Bon Jovi',

tracks: ['Let It Rock', 'You Give Love a Bad Name']

},

2468: {

albumTitle: '1999',

artist: 'Prince',

tracks: ['1999', 'Little Red Corvette']

},

1245: {

artist: 'Robert Palmer',

tracks: []

},

5439: {

albumTitle: 'ABBA Gold'

}

};

// Only change code below this line

function updateRecords(records, id, prop, value)

{

if(prop !== 'tracks' && value !== '')

{

records[id][prop] = value;

}

else if(prop === 'tracks' && records[id].hasOwnProperty('tracks') == false)

{

records[id][prop] = [value];

}

else if(prop === 'tracks' && value !== '')

{

records[id][prop].push(value);

}

else if(value === '')

{

delete records[id][prop];

}

return records;

}

updateRecords(recordCollection, 5439, 'artist', 'ABBA');

**96)Iterate with javascript while loop** // Setup

const myArray = [];

var i = 5;

while(i>=0)

{

myArray.push(i);

i--;

}

// Only change code below this line

**97)Iterate with javascript for loop**// Setup

const myArray = [];

for(let i=1;i<=5;i++)

{

myArray.push(i);

}

// Only change code below this line

**98)Iterate odd number with for loop**// Setup

const myArray = [];

for(let i=1;i<=9;i+=2)

{

myArray.push(i);

}

// Only change code below this line

**99)Count backward with a for loop**// Setup

const myArray = [];

for(let i=9;i>0;i-=2)

{

myArray.push(i);

}

// Only change code below this line

**100)Iterate through an array with a for loop**// Setup

const myArr = [2, 3, 4, 5, 6];

var total = 0;

for(let i=0;i<myArr.length;i++)

{

total = total + myArr[i];

}

// Only change code below this line

**101)Nesting for loop**function multiplyAll(arr) {

let product = 1;

// Only change code below this line

for(let i = 0;i<arr.length;i++)

{

for(let j=0;j<arr[i].length;j++)

{

product = product\*arr[i][j];

}

}

// Only change code above this line

return product;

}

multiplyAll([[1, 2], [3, 4], [5, 6, 7]]);

**102)Iterate with javascript do..while loop**// Setup

const myArray = [];

let i = 10;

// Only change code below this line

do

{

myArray.push(i);

i++;

}while(i<11);

console.log(i);

**103)Replace loops using recursion**function sum(arr, n) {

// Only change code below this line

if(n<=0)

{

return 0;

}

else{

return sum(arr,n-1) + arr[n-1];

}

// Only change code above this line

}

sum([1],0);

**104)Profile lookup**// Setup

const contacts = [

{

firstName: "Akira",

lastName: "Laine",

number: "0543236543",

likes: ["Pizza", "Coding", "Brownie Points"],

},

{

firstName: "Harry",

lastName: "Potter",

number: "0994372684",

likes: ["Hogwarts", "Magic", "Hagrid"],

},

{

firstName: "Sherlock",

lastName: "Holmes",

number: "0487345643",

likes: ["Intriguing Cases", "Violin"],

},

{

firstName: "Kristian",

lastName: "Vos",

number: "unknown",

likes: ["JavaScript", "Gaming", "Foxes"],

},

];

function lookUpProfile(name, prop) {

// Only change code below this line

for(let i=0;i<contacts.length;i++)

{

if(contacts[i].firstName == name)

{

if(contacts[i].hasOwnProperty(prop))

{

return contacts[i][prop];

}

else

{

return "No such property";

}

}

}

return "No such contact";

// Only change code above this line

}

lookUpProfile("Akira", "likes");

**105)Generate random fractions with javascipt**function randomFraction() {

// Only change code below this line

return Math.random()

// Only change code above this line

}

**106)Generate random whole number with javascript**function randomWholeNum() {

// Only change code below this line

return Math.floor(Math.random()\*10);

}

**107)Generate random whole numbers within a range**function randomRange(myMin, myMax) {

// Only change code below this line

return Math.floor(Math.random() \* (myMax - myMin + 1)) + myMin

// Only change code above this line

}

**108)Use the parseint function**function convertToInteger(str) {

return parseInt(str);

}

convertToInteger("56");

**109)Use the parseint function with radix**function convertToInteger(str) {

return (parseInt(str,2));

}

convertToInteger("10011");

**110)Use the conditional (ternary) operator**function checkEqual(a, b) {

return a == b? "Equal" : "Not Equal";

}

checkEqual(1, 2);

**111)Use multiple conditional(ternary) operator**function checkSign(num) {

return (num==0)?"zero":(num>0)?"positive":"negative";

}

checkSign(10);

**112)Use recursion to create a countdown**// Only change code below this line

function countdown(n){

if(n<=0)

{

return [];

}

else

{

const arr = countdown(n-1);

arr.unshift(n);

return arr;

}

}

console.log(countdown(5));

console.log(countdown(10));

// Only change code above this line

**113)Use recursion to create a range of numbers**function rangeOfNumbers(startNum, endNum) {

if(endNum == startNum)

{

return [startNum];

}

else

{

const arr = rangeOfNumbers(startNum,endNum-1);

arr.push(endNum);

return arr;

}

};

console.log(rangeOfNumbers(1,5));