**Swami Vivekananda University**

**Indranil Paul**

**Roll No. 002-BCS-2022-174**

**DAY-9 ASSIGNMENT SOLUTIONS**

**Section 1: Data Types & Length**

**1. What will be the output of the following code? Explain why.**

let x = "5";

let y = 5;

console.log(x == y); // true

console.log(x === y); // false

Explanation:

* == (loose equality) converts x into a number before comparing, so "5" and 5 are equal.
* === (strict equality) checks both value and type, and since x is a string and y is a number, it returns false.

**2. Function to find the length of the longest word in an array**

const words = ["JavaScript", "Programming", "Function", "Hoisting"];

function longestWordLength(words) {

return Math.max(...words.map(word => word.length));

}

console.log(longestWordLength(words)); // Output: 11 (Programming)

**Section 2: let, const, var & Scope**

**3. What will be logged in the console? Explain your answer.**

function testScope() {

if (true) {

var a = 10;

let b = 20;

const c = 30;

}

console.log(a); // 10 (var is function-scoped)

console.log(b); // Error (let is block-scoped)

console.log(c); // Error (const is block-scoped)

}

testScope();

**4. Rewrite function using const and fix issues**

const name = "John";

function greet() {

if (true) {

const message = "Hello " + name;

}

console.log(message); // Error: message is block-scoped

}

greet();

**Fixed Version:**

const name = "John";

function greet() {

let message = "Hello " + name;

console.log(message);

}

greet();

**Section 3: Traditional Function vs. Arrow Function**

**5. Convert function to arrow function**

const multiply = (a, b) => a \* b;

**6. Output of the following code**

const obj = {

name: "Alice",

sayHello: function() {

setTimeout(() => {

console.log("Hello, " + this.name);

}, 1000);

}

};

obj.sayHello(); // Output: "Hello, Alice"

Explanation:

* Arrow functions do not have their own this; they inherit from the surrounding scope.

**Section 4: Hoisting**

**7. Output and Explanation**

console.log(a); // undefined (due to hoisting)

var a = 10;

console.log(b); // ReferenceError (let variables are not hoisted)

let b = 20;

**8. Fix hoisting issues**

console.log(square(5)); // 25

function square(n) {

return n \* n;

}

console.log(double(4)); // Error

var double = function(n) {

return n \* 2;

};

**Fixed Version:**

console.log(square(5)); // 25

function square(n) {

return n \* n;

}

const double = (n) => n \* 2;

console.log(double(4)); // 8

**Section 5: Operators**

**9. Output of expressions**

console.log(5 + "5"); // "55"

console.log(5 - "3"); // 2

console.log(5 \* "2"); // 10

console.log("10" / 2); // 5

console.log(10 % "3"); // 1

**10. Rewrite using shorthand assignment operators**

let x = 10;

x += 5;

x \*= 2;

x -= 3;

x /= 2;

**11. Logical operators output**

console.log(5 > 3 && 10 < 20); // true

console.log(5 > 3 || 10 > 20); // true

console.log(!(5 > 3)); // false

**Section 6: Function Parameters vs Arguments**

**12. Output of function calls**

function sum(a, b, c = 5) {

return a + b + c;

}

console.log(sum(2, 3)); // 10 (2 + 3 + 5)

console.log(sum(2, 3, 10)); // 15 (2 + 3 + 10)

**13. Function to take any number of arguments and return their sum**

function sumAll(...numbers) {

return numbers.reduce((acc, num) => acc + num, 0);

}

console.log(sumAll(1, 2, 3, 4)); // 10

**Bonus: Hard Challenge Question**

**14. Fix and explain the issue**

function outer() {

let count = 0;

return function inner() {

count++;

console.log(count);

};

}

const counter1 = outer();

counter1(); // 1

counter1(); // 2

const counter2 = outer();

counter2(); // 1

counter2(); // 2

**Explanation:** Each call to outer() creates a new scope with its own count variable.

**Some Statement Questions**

**1. Reverse a number without built-in methods**

function reverseNumber(num) {

let str = "";

let negative = num < 0;

num = Math.abs(num);

while (num > 0) {

str += num % 10;

num = Math.floor(num / 10);

}

return negative ? -parseInt(str) : parseInt(str);

}

**2. Custom length function**

function customLength(str) {

let count = 0;

for (let char of str) count++;

return count;

}

**3. Fix hoisting bug**

console.log(add(2, 3)); // 5

console.log(multiply(2, 3)); // 6

function add(a, b) {

return a + b;

}

const multiply = (a, b) => a \* b;

**4. Function returning another function (Counter)**

function counter() {

let count = 0;

return function() {

return ++count;

};

}

const count = counter();

console.log(count()); // 1

console.log(count()); // 2

console.log(count()); // 3