Week 2 Assignment 2 JavaScript Practice

Data Types and Variables:

a. What are the different data types used in JavaScript variables in the provided code?

String - used to handle sequence of characters such numbers , letters , white spaces , symbols

Enclosed in single of double quotations ie “ ” or ‘ ’ to store and manipulate text

number - Used to handle numeric values (real numbers)

Bigint - used handle numbers which are greater than 2^53 -1

Array - it is a data structure that holds multiple values, indexed starting from 0. It can store elements of different types (e.g., numbers, strings) and allows accessing them via their index.
Boolean - It is a data type that can have only two values: true or false. It is often used in conditional statements and logical operations to control the flow of a program.
Object - is a data structure used to store collections of key-value pairs. Keys are called properties, and each property has a value that can be of any data type (e.g., string, number, array, function).
symbol - A symbol in JavaScript is a unique and immutable data type that is primarily used to create unique identifiers for object properties. Each symbol value is unique, even if symbols have the same description.
Undefined - In JavaScript, undefined occurs when a variable is declared but not assigned a value. It signifies the absence of an initial value.
Null - signifies the intentional absence of a value in JavaScript.

b. Explain the difference between var, let, and const in JavaScript.

var: An older keyword used to declare variables that can be re-assigned and have function scope.

let: A modern keyword for declaring variables with block scope that can be re-assigned.

const: A keyword for declaring variables whose values cannot be changed after initialization.

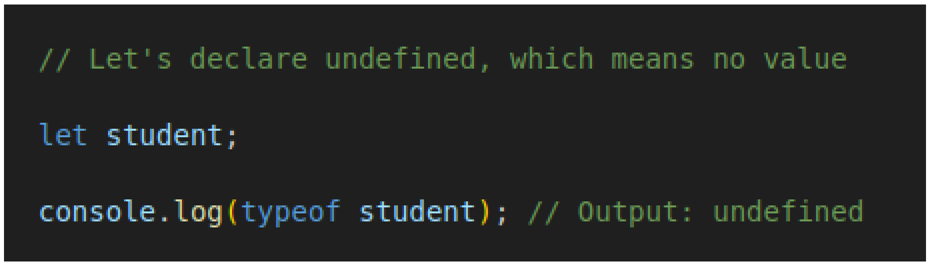
c. Why does JavaScript allow assigning different data types to the same variable?

JavaScript allows assigning different data types to the same variable due to its dynamic typing system, which means variables are not bound to a specific type and can change type as needed.

d. How does JavaScript handle variables declared but not initialized? Illustrate with an example from the code.

JavaScript handles variables declared but not initialized by assigning them the value undefined. This indicates that the variable exists but has not been assigned a specific value. When such a variable is logged to the console, it will display undefined

Example



```
// Let's declare undefined, which means no value

let student;

console.log(typeof student); // Output: undefined
```

e. Discuss the significance of variable names in programming and how they are used in JavaScript.

Variable names in programming are crucial as they act as identifiers for storing and accessing data in memory. When a variable is declared and assigned a value, it is allocated a specific space in memory, and the value can only be accessed using the variable's name.

In JavaScript, variable names are declared using keywords like var, let, or const, and must follow specific naming conventions:

They must start with a letter, underscore (_), or dollar sign ($).

They can include letters, numbers, underscores, and dollar signs after the initial character.

They are case-sensitive and should be descriptive to enhance code readability.

Numeric DataTypes

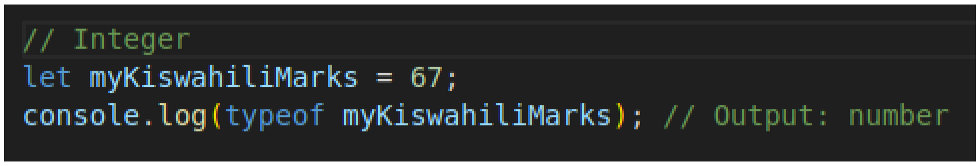
a. What are the various numeric data types used in JavaScript, as shown in the code?

In JavaScript, the primary numeric data type is:

Number: Represents both integer and floating-point numbers, including special values like Infinity and NaN.

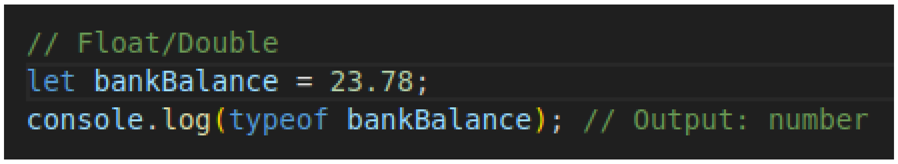
b. Explain the difference between integers, doubles, and Infinity in JavaScript with examples.

Integers - Whole numbers without a decimal point.



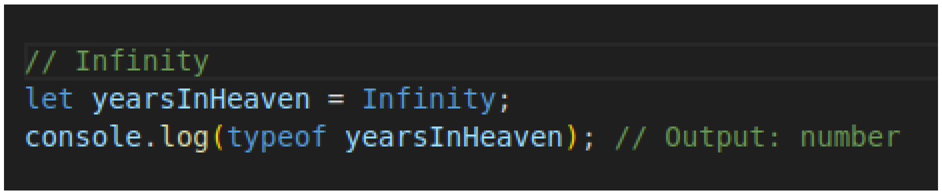
```
// Integer
let myKiswahiliMarks = 67;
console.log(typeof myKiswahiliMarks); // Output: number
```

Doubles - Floating-point numbers with a decimal point, representing real numbers.



```
// Float/Double
let bankBalance = 23.78;
console.log(typeof bankBalance); // Output: number
```

Infinity - A special value representing positive or negative infinity, often used to signify values beyond the range of the Number type

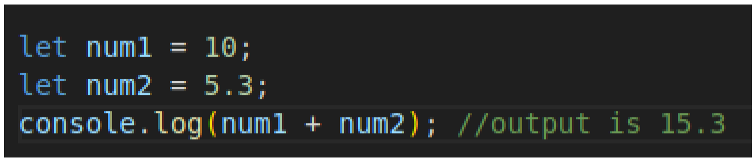


```javascript
// Infinity
let yearsInHeaven = Infinity;
console.log(typeof yearsInHeaven); // Output: number
```

c. How does JavaScript handle arithmetic operations involving different numeric data types?

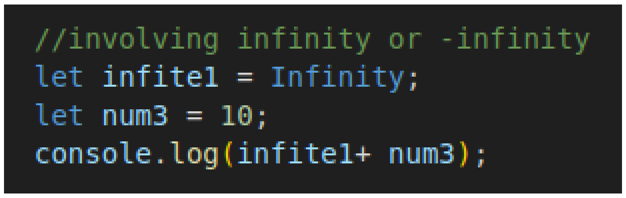
In JavaScript, arithmetic operations involving different numeric data types are handled with type coercion:

Mixed Integers and Doubles: When performing operations between integers and floating-point numbers, JavaScript converts integers to floating-point numbers to ensure precise calculations.



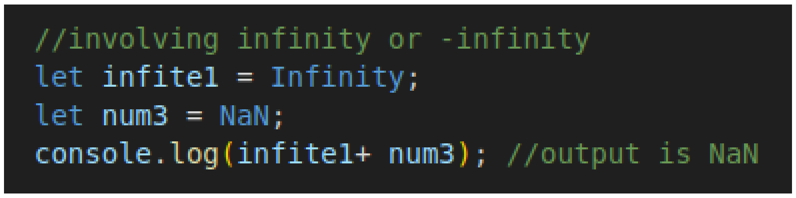
```javascript
let num1 = 10;
let num2 = 5.3;
console.log(num1 + num2); //output is 15.3
```

Infinity: Any arithmetic operation involving Infinity (or -Infinity) results in Infinity (or -Infinity), except when involving NaN.



```javascript
//involving infinity or -infinity
let infite1 = Infinity;
let num3 = 10;
console.log(infite1+ num3);
```

NaN: Operations involving NaN generally result in NaN.



```javascript
//involving infinity or -infinity
let infite1 = Infinity;
let num3 = NaN;
console.log(infite1+ num3); //output is NaN
```

String Data Type:

a. How are strings represented in JavaScript?

In JavaScript, strings are represented as sequences of characters enclosed in single quotes ('), double quotes ("), or backticks (`). They are immutable, meaning once created, their content cannot be changed.**Single and Double Quotes**: Used for standard string values.

Template Literals: Enclosed in backticks, allowing for multi-line strings and embedded expressions using ${}.

b. Discuss the difference between declaring strings with single quotes ('') and double quotes ("") in JavaScript.

Use single or double quotes in JavaScript to avoid escaping the other type, such as using single quotes when double quotes are needed inside the string, with no functional difference between them.

c. Explain why characters are automatically treated as strings in JavaScript.

In JavaScript, characters are treated as strings because JavaScript doesn't have a separate character type, so all text is represented as string data.

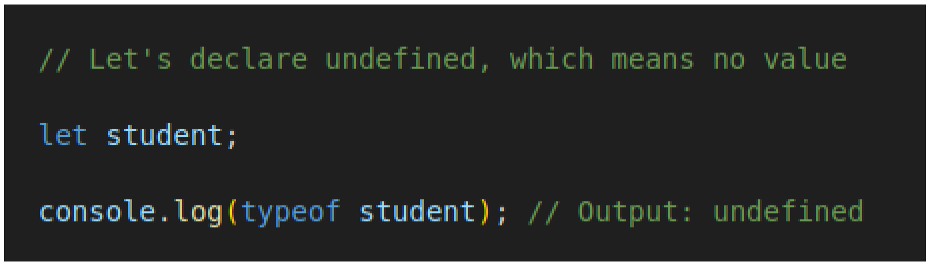
Boolean and Undefined Data Types

a. Explain the purpose of boolean variables in JavaScript.

- Boolean variables in JavaScript represent true or false values, used to determine conditions and control program execution flow.

b. Discuss the concept of undefined in JavaScript variables and provide examples from the code.

- In JavaScript, undefined is the default value of a variable that has been declared but not assigned a value.



```
// Let's declare undefined, which means no value

let student;

console.log(typeof student); // Output: undefined
```

This will log **undefined** in the console

c. How are boolean variables useful in conditional statements and control flow in JavaScript?

Boolean variables in JavaScript control the flow of execution in conditional statements by determining if a specific code block should run, based on whether the condition evaluates to true or false.

Null Data Type:

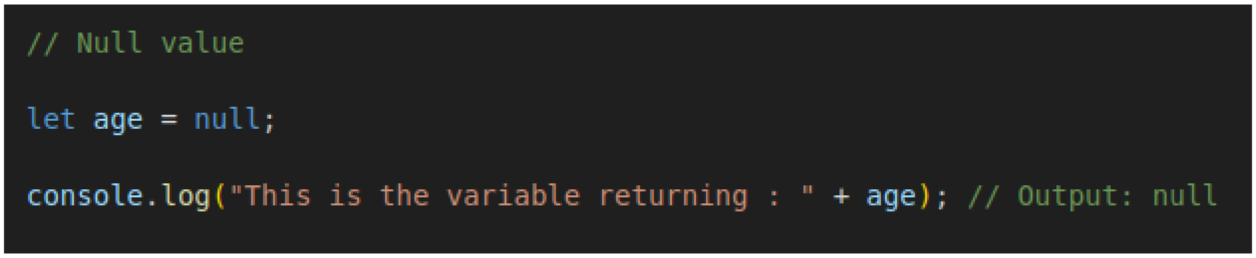
a. Describe the significance of the null value in JavaScript.

In JavaScript, null represents the intentional absence of any value and is an assignment value that signifies "no value" or "no object." It's often used to explicitly indicate that a variable or object reference is empty or non-existent.

b. Differentiate between null and undefined in JavaScript.

In JavaScript, undefined means a variable has been declared but not assigned a value, while null is an explicit assignment that indicates the absence of any value or object. undefined is set by the JavaScript engine, whereas null is set intentionally by the programmer.

b. Provide an example from the code illustrating the use of null.



```javascript
// Null value

let age = null;

console.log("This is the variable returning : " + age); // Output: null
```

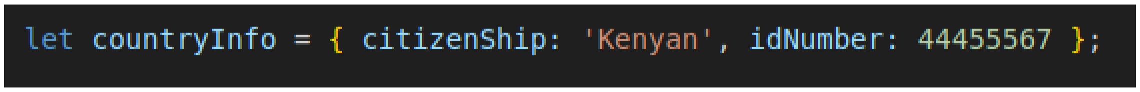
Object Data Type:

a. Explain how objects are represented in JavaScript.

Objects in JavaScript are represented as key-value pairs enclosed in curly braces,

e.g., const myObject = { key: value };.

b. Discuss the structure and purpose of the countryInfo object in the provided code.



```javascript
let countryInfo = { citizenShip: 'Kenyan', idNumber: 44455567 };
```

The countryInfo object holds details about a person's citizenship and ID number, structured as key-value pairs for easy access to this information.

b. How can objects be nested within other objects in JavaScript?

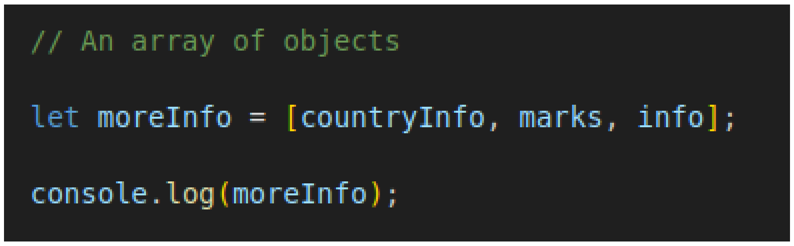
In JavaScript, objects can be nested within other objects by assigning an object as a property value of another object. This allows for hierarchical or complex data structures.

Array Data Type:

a. Describe the purpose and structure of arrays in JavaScript.

Arrays in JavaScript are used to store multiple values in a single variable

b. Provide examples from the code demonstrating arrays containing different data types.



```
// An array of objects

let moreInfo = [countryInfo, marks, info];

console.log(moreInfo);
```

c. Discuss the concept of "array of arrays" and its significance.

An "array of arrays" (also called a 2D array) is a nested structure where each element is itself an array, often used to represent grids, tables, or matrices. This concept is significant as it allows for efficient organization and manipulation of complex data sets in structured formats.

Variable Naming Conventions:

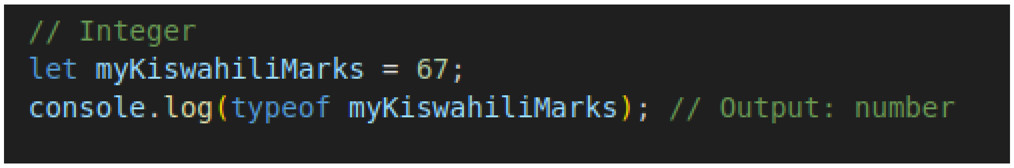
a. What are the conventions for naming variables in JavaScript?

JavaScript variables typically follow camelCase, start with a letter, and cannot contain spaces or special characters (except $ and _).

b. Discuss the importance of choosing meaningful and descriptive variable names.

Using descriptive names improves code readability, making it easier to understand and maintain.

c. Identify any variable naming conventions followed or violated in the provided code.



```
// Integer
let myKiswahiliMarks = 67;
console.log(typeof myKiswahiliMarks); // Output: number
```

The variable name myKiswahiliMarks follows the camelcase Naming convention.

Constants in JavaScript:

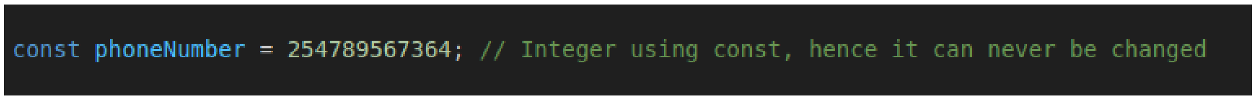
a. Explain the use of const keywords in JavaScript.

The const keyword in JavaScript is used to declare variables that are immutable, meaning their reference cannot be reassigned after initial assignment. However, for objects and arrays, the contents can still be modified, even though the variable reference itself cannot be changed.

b. Discuss why reassigning a value to a constant variable results in an error.

Reassigning a value to a constant variable results in an error because const ensures that the variable's reference is immutable, meaning once it is assigned, it cannot be changed or reassigned to another value. This protects the variable from unintended modifications, ensuring stability in the code.

c. Provide examples from the code demonstrating the declaration and use of constants.



```
const phoneNumber = 254789567364; // Integer using const, hence it can never be changed
```