**MARTIN MUTINDA**

**JS-ASSIGNMENT 2**

**1.Data Types and Variables**

a).The data types used in the code are:

* Strings e.g var first\_name = 'Eluid Murithi';
* Numbers e.g const phoneNumber = 254789567364;
* Objects e.g let countryInfo = { citizenShip: 'Kenyan', idNumber: 44455567 };
* Arrays e.g let marks = [34, 56, 67, 78];

b). var, let and const are all keywords used to declare variables in Javascript.

var is function-scoped, meaning that it is accessible within the function where it is declared, or globally if declared outside of a function. Var can both be redeclared and reassigned eg:

var x=20

var x=20

let is global-scoped, meaning that it is only accessible within the block where it is defined. let cannot be redeclared but can be reassigned eg:

let x=10

x=15

const is also global scoped and it is used when no redeclaration and reassigning is required.

c). The reason as to why Javascript allows assigning different data types to the same variable is because Javascript is a dynamic programming language and allows dynamic typing.

d). In Javascript, when a variable is declared but not initialized, it is automatically set to undefined.

Example from the code:

let student;

console.log(typeof student);

e). In programming, variable names are crucial for writing readable, maintainable and understandable code. They serve as reference to stored data and are used to label values and objects so that they can be reused and manipulated throughout the program. Clear and descriptive variable names makes the code easier to understand for others or even the owner when revisiting. A meaningful name helps to understand the purpose of the variable at glance.

Javascript variable names are case-sensitive, meaning that for example, firstName and firstname are two different variable names. You cannot not also use reserved keywords like let and function as a variable name.

**2. Numeric Data Types**

a). The various numeric data types used in Javascript as used in the provided code are:

1. Integers e.g let myKiswahiliMarks = 67
2. Float e.g let bankBalance = 23.78

b). Integers are whole numbers numbers without decimals e.g:

let integer = 42 // This is treated as an integer

let negativeInteger = -100 // Negative integer

console.log(integer) // Outputs: 42

Doubles refers to numbers that have decimal points e.g:

let double = 3.14 // This is a double (floating-point number)

let anotherDouble = -123.456 // Another double

console.log(double) // Outputs: 3.14

Infinity is a special numeric value in Javascript that represents numbers that exceed the upper limit of the Number type(a number that is greater than the maximum representable number) e.g:

let positiveInfinity = 1 / 0 // Outputs: Infinity

let negativeInfinity = -1 / 0 // Outputs: -Infinity

console.log(positiveInfinity, negativeInfinity);

c). Javascript performs arithmetic operations on different numeric data types by automatically converting them to a common type. Below is a breakdown of how Javascript handles these conversions:

1. **Arithmetic operations between integers and floating-point numbers are straightforward. The result will always be a floating-point number e.g:**

let intNum = 5 // Integer

let floatNum = 2.5 // Floating-point number

let result = intNum + floatNum; // 7.5 (result is a floating-point number)

1. **Any number added or subtracted from infinity or -infinity results in infinity or -infinity. Multiplying a +ve number with infinity whereas multiplying a -ve number with infinity results in -ve infinity. Division by infinity results to 0, but diving by 0 results in infinity:**

let inf = Infinity;

console.log(inf + 10) // Infinity

console.log(-inf + 5) // -Infinity

console.log(inf \* -1) // -Infinity

console.log(5 / Infinity) // 0

console.log(5 / 0) // Infinity

1. **Any arithmetic involving NaN results in NaN e.g:**

let nanValue = NaN

console.log(nanValue + 5) // NaN

console.log(0 / 0) // NaN

console.log(Math.sqrt(-1)) // NaN

1. **When performing arithmetic operations between numbers and strings that represent numeric values, Javascript attempts to coerce the string into a number. If the string can be converted to a valid number, the operation proceeds; otherwise the result is NaN. Addition between a number and a string results in string concatenation while for other operators (-, \*, /, %), Javascript attempts to coerce the string into a number before performing the operation. Below is an illustration:**

let number = 10

let stringNumber = "5"

let result1 = number + stringNumber; // "105" (string concatenation)

let result2 = number - stringNumber; // 5 (string "5" is coerced to number 5)

let result3 = number \* stringNumber; // 50 (multiplication coerces the string)

let result4 = number / stringNumber; // 2 (division coerces the string)

1. If one operand is Bignit, the other operand will be converted to a BigInt before operation.

**3. String Data Type**

a). Strings in Javascript are represented as a sequence of characters enclosed in quotes, which can be either single, double or literal quotes.

b).There is no functional difference between strings declared with single or double quotes in Javascript. However, there becomes a need to choose between single and double quotes in declaring strings in various situations, for example, when using single quotes inside a string or when using double quotes inside a string as illustrated below:

let str1 = "It's a beautiful day."

let str2 = 'It\'s a beautiful day.'

let str1 = 'She said, "Hi"'

let str2 = "She said, \"Hi\""

c). Characters in Javascript are automatically treated as strings because Javascript does not have a separate data type for characters as some other programming languages like C do.

**4. Boolean and Undefined Data Types**

a). Boolean variables are essential for controlling the flow of a program,making decisions and evaluating conditions. They are used to represent one of two possible values: true or false.

b). Undefined in Javascript occurs when you declare a variable and you don’t assign a value to it. An example in the code is:

let student

c). They help to determine which blocks of code should be executed based on whether certain conditions are met.

**5. Null Types**

a). Null is used to indicate that a variable has been intentionally set to an empty.

b). A null represents an intentional absence of a value while undefined represents a variable that has been declared but not yet assigned a value.

c). An example of null from the code:

let age = null;

1. **Object Data Type**

a). Objects in Javascript are composed of keys where each key(property) is a string or a symbol, and the value can be any data type, including another object. This is the structure of an object: keyword variableName{}

b). The object is declared using the let keyword and assigned to the variable countryInfo. citizenship and idNumber are keys (properties) while ‘Kenyan’ and 44455567 are values assigned to the keys respectively. The object is used to represent information related to an individual’s country of citizenship.

c). You can assign an object in another object by simply assigning a Key inside a property of another property containing keys and values.

1. **Array Data Type**

a). Arrays are used to store multiple values in a single variable. The structure of an array is:  
 keyword variableName = []

b). Examples of arrays containing different data types from the code:

I could not find any.

c). An array of arrays (or multidimensional array) is an array where each element is itself an array e.g

let matrix = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9]

]

They are important in creating and managing complex data structures like matrices.

1. **Variable Naming Conventions**

a). Use camelCase for naming variables eg firstName,

Start variable names with a letter, underscore or a dollar sign

Be descriptive but concise

Avoid reserved keywords

b).Meaning and descriptive variable names are important because:

They make the code easy to read and understand

They are easily identified in case of debugging

They improve collaboration s everyone can easily understand what they are for

They enable easy documentation avoiding too much explaining comments

c). camelCase naming has been honoured e.g let myName, let myKiswahiliMarks e.t.c

Use of underscore has been honoured e.g var first\_name

cameCase naming has also been violated e.g in let snake instead of let sName. The variable name is also not descriptive.

1. **Constants in Javascript**

a). const keyword is used to declare block-scoped variables that do not need reassigning after initialization.

b). Reassigning const variables results in an error because const enforces immutability of the reference to ensure code consistency and prevent unintended side effects.

c). Examples from the code demonstrating the use of const:

const phoneNumber = 254789567364