**Assignment For Day 2**

1. What is lexical structure?

Ans:- A programming language’s lexical structure specifies a set of some basic rules about how code should be written in it. Rules like what variable names should look like, the delimiter characters for comments, and how one program statement is separated from the text. It is the lowest level syntax of a language.

1. What is Unicode?

Ans:- In Javascript, the identifiers and string literals can be expressed in Unicode via a Unicode escape sequence. The general syntax is \uXXXX , where X denotes four hexadecimal digits. For example, the letter o is denoted as '\u006F' in Unicode.

1. Explain all the keywords present in the JavaScript with examples.

Ans:- Keywords are reserved words that are part of the syntax in the programming language. For example, const a = 'hello'; Here, const is a keyword that denotes that a is a constant.

1. What are shorthand operators, explain with a suitable example?

Ans;- JavaScript has the following types of operators.

| **Name** | **Shorthand operator** | **Meaning** |
| --- | --- | --- |
| [Assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Assignment) | x = f() | x = f() |
| [Addition assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Addition_assignment) | x += f() | x = x + f() |
| [Subtraction assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Subtraction_assignment) | x -= f() | x = x - f() |
| [Multiplication assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Multiplication_assignment) | x \*= f() | x = x \* f() |
| [Division assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Division_assignment) | x /= f() | x = x / f() |
| [Remainder assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Remainder_assignment) | x %= f() | x = x % f() |
| [Exponentiation assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Exponentiation_assignment) | x \*\*= f() | x = x \*\* f() |
| [Left shift assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Left_shift_assignment) | x <<= f() | x = x << f() |
| [Right shift assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Right_shift_assignment) | x >>= f() | x = x >> f() |
| [Unsigned right shift assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Unsigned_right_shift_assignment) | x >>>= f() | x = x >>> f() |
| [Bitwise AND assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Bitwise_AND_assignment) | x &= f() | x = x & f() |
| [Bitwise XOR assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Bitwise_XOR_assignment) | x ^= f() | x = x ^ f() |
| [Bitwise OR assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Bitwise_OR_assignment) | x |= f() | x = x | f() |
| [Logical AND assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical_AND_assignment) | x &&= f() | x && (x = f()) |
| [Logical OR assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical_OR_assignment) | x ||= f() | x || (x = f()) |
| [Logical nullish assignment](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical_nullish_assignment) | x ??= f() | x ?? (x = f()) |

1. What is “use Strict” in JavaScript?

Ans:- The purpose of “use strict” is to indicate that the code should be executed in “strict mode”.