**Class 4**

***LOOPS***

They allow you to execute a block of code repeatedly.

***FOR LOOP***

The for loop is used when you know the number of iterations in advance.

**Syntax**

For (inintialization; condition; iteration)

**for (let i = 0; i < 5; i++) {**

**console.log(i);**

**}**

In the above example, the loop will iterate five times (0 through 4), and **i** will take the values 0, 1, 2, 3, and 4 in each iteration.

* **tsc –watch**

**Modules**

**Default export, curly braces**

modules are a way to organize code into separate files and control the visibility of variables, functions, classes, and other entities between different parts of your codebase. Modules provide a mechanism to encapsulate code and manage its dependencies, making it easier to maintain and scale larger projects.

There are two main module systems in TypeScript:

**ES6 (ES2015) Modules:** ES6 modules are the standardized module system introduced in ECMAScript 2015 (ES6) and later versions of JavaScript. They are designed to work both in browser environments and in Node.js. ES6 modules use the **import** and **export** keywords to define and use modules.

**Defining a Module:**

// mathOperations.ts

export function add(a: number, b: number): number {

return a + b;

}

**Using a Module:**

// main.ts

import { add } from './mathOperations';

let result = add(5, 3);

console.log(result); // Output: 8

**CommonJS Modules:**

CommonJS is a module system primarily used in Node.js environments. It uses the **require** function to import modules and the **exports** object to define what is exported from a module.

// mathOperations.js

exports.add = function(a, b) {

return a + b;

};

**Using a Module:**

// main.js

const mathOperations = require('./mathOperations');

let result = mathOperations.add(5, 3);

console.log(result); // Output: 8

**Default Export:**

A default export allows you to export a single value, function, class, or object from a module. When you import a default export in another module, you can give it any name you want, making it more flexible. There can be only one default export per module.

**Named Exports (Using Curly Braces):**

Named exports allow you to export multiple values, functions, classes, or objects from a module using curly braces. You must import them using the exact same name.

**Modules provide benefits such as:**

* **Encapsulation:** Modules allow you to encapsulate code, reducing global scope pollution and potential naming conflicts.
* **Organization:** Modules help you organize code into manageable files and directories, improving codebase maintainability.
* **Reusability:** You can reuse modules across different parts of your application or in different projects.
* **Dependency Management:** Modules provide a clear way to manage dependencies between different parts of your code.