

```
1 "use strict";
2 // Estructura de datos: Lista de tiendas
3 const tiendas = [
4   {
5     id: 1,
6     nombre: "San Juan",
7     categorias: [
8       {
9         nombre: "Electrónica",
10        productos: [
11          { nombre: "TV", precio: 1599.99, stock: 10, estado: true },
12          { nombre: "Radio", precio: 49.99, stock: 25, estado: true },
13          { nombre: "Teléfono", precio: 699.99, stock: 15, estado: false },
14        ],
15      },
16      {
17        nombre: "Ropa",
18        productos: [
19          { nombre: "Camiseta", precio: 19.99, stock: 50, estado: true },
20          { nombre: "Pantalón", precio: 39.99, stock: 30, estado: true },
21          { nombre: "Zapatos", precio: 89.99, stock: 20, estado: false },
22        ],
23      },
24    ],
25  },
26  {
27    id: 2,
28    nombre: "Santa Marta",
29    categorias: [
30      {
31        nombre: "Electrónica",
32        productos: [
```

```
47   },
48 ];
49 /**
50  * Muestra todas las tiendas en forma de tabla organizada.
51  */
52 function mostrarTiendas() {
53   console.log("\n--- 📋 Lista de Tiendas ---\n");
54   // Recorre cada tienda y organiza la información en formato tabular
55   tiendas.forEach(tienda => {
56     console.log(`🏪 Tienda: ${tienda.nombre} (ID: ${tienda.id})`);
57     tienda.categorias.forEach(categoria => {
58       console.log(`📦 Categoría: ${categoria.nombre}`);
59       const tablaProductos = categoria.productos.map(producto => ({
60         Nombre: producto.nombre,
61         Precio: `${producto.precio.toFixed(2)}`,
62         Stock: producto.stock,
63         Estado: producto.estado ? "Activo" : "Inactivo",
64       }));
65       console.table(tablaProductos); // Muestra los productos como tabla
66     });
67   });
68 }
69 /**
70  * Busca un producto por su nombre y muestra los resultados como tabla.
71  * @param nombreProducto - El nombre del producto a buscar.
72  */
73 function buscarProducto(nombreProducto) {
74   console.log(`\n--- 🔍 Buscando Producto: ${nombreProducto} ---\n`);
75   const resultados = [];
76   // Recorre todas las tiendas y categorías para buscar el producto
77   tiendas.forEach(tienda => {
```

This screenshot shows a VS Code editor with a TypeScript file named `app.ts`. The code defines two functions: `mostrarTiendas()` and `buscarProducto(nombreProducto: string): void`. The `mostrarTiendas()` function iterates through a list of stores (`tiendas`), and for each store, it iterates through its categories (`categorias`). For each category, it maps the products (`productos`) into a table structure with columns for name, price (formatted to two decimal places), stock, and status (Active/Inactive). The `buscarProducto` function logs the search criteria and the results, which are structured as an array of objects containing store name, category, price, and stock.

```
67
68
69
70
71 function mostrarTiendas(): void {
72     console.log("\n--- 📋 Lista de Tiendas ---\n");
73
74     // Recorre cada tienda y organiza la información en formato tabular
75     tiendas.forEach(tienda => {
76         console.log(`🏪 Tienda: ${tienda.nombre} (ID: ${tienda.id})`);
77         tienda.categorias.forEach(categoria => {
78             console.log(`📦 Categoría: ${categoria.nombre}`);
79             const tablaProductos = categoria.productos.map(producto => ({
80                 Nombre: producto.nombre,
81                 Precio: `${producto.precio.toFixed(2)}`,
82                 Stock: producto.stock,
83                 Estado: producto.estado ? "Activo" : "Inactivo",
84             }));
85             console.table(tablaProductos); // Muestra los productos como tabla
86         });
87     });
88 }
89
90
91 /**
92  * Busca un producto por su nombre y muestra los resultados como tabla.
93  * @param nombreProducto - El nombre del producto a buscar.
94  */
95 function buscarProducto(nombreProducto: string): void {
96     console.log(`\n--- 🔍 Buscando Producto: ${nombreProducto} ---\n`);
97     const resultados: { Tienda: string; Categoría: string; Precio: string; Stock: number }[] = [];
98
99     // Recorre todas las tiendas y categorías para buscar el producto
```

This screenshot shows the same VS Code editor with the same TypeScript file, but now displaying the data structures defined in the code. It includes three interfaces: `Producto`, `Categoria`, and `Tienda`. The `Producto` interface has properties for name, price, stock, and status. The `Categoria` interface has a name and a list of products. The `Tienda` interface has an ID, name, and a list of categories. Below the interfaces, there is a constant `tiendas` of type `Tienda[]` containing one store named "San Juan". This store has a category named "Electrónica", which contains three products: "TV", "Radio", and "Teléfono", each with specific price, stock, and status values.

```
1 // TypeScript Types for Data Structure
2 interface Producto {
3     nombre: string; // Nombre del producto
4     precio: number; // Precio del producto
5     stock: number; // Cantidad disponible en inventario
6     estado: boolean; // Estado del producto (true: activo, false: inactivo)
7 }
8
9 interface Categoria {
10     nombre: string; // Nombre de la categoría
11     productos: Producto[]; // Lista de productos pertenecientes a esta categoría
12 }
13
14 interface Tienda {
15     id: number; // Identificador único de la tienda
16     nombre: string; // Nombre de la tienda
17     categorias: Categoria[]; // Lista de categorías en la tienda
18 }
19
20 // Estructura de datos: Lista de tiendas
21 const tiendas: Tienda[] = [
22     {
23         id: 1,
24         nombre: "San Juan",
25         categorias: [
26             {
27                 nombre: "Electrónica",
28                 productos: [
29                     { nombre: "TV", precio: 1599.99, stock: 10, estado: true },
30                     { nombre: "Radio", precio: 40.90, stock: 25, estado: true },
31                     { nombre: "Teléfono", precio: 699.99, stock: 15, estado: false },
32                 ]
33             }
34         ]
35     }
36 ]
```

The screenshot shows the VS Code editor with a file named `app.js` open. The file contains a TypeScript function `buscarProducto(nombreProducto)` that searches for a product in a nested array structure. The function uses `console.log` to display the search results as a table. The execution of the code is shown in the console, displaying the results for searching for "Radio", "Zapatos", and "Laptop".

```
72 function buscarProducto(nombreProducto) {  
73   console.log(`\n--- Buscando Producto: ${nombreProducto} ---\n`);  
74   const resultados = [];  
75   // Recorre todas las tiendas y categorías para buscar el producto  
76   tiendas.forEach(tienda => {  
77     tienda.categorías.forEach(categoría => {  
78       categoría.productos  
79         .filter(producto => producto.estado && producto.nombre.toLowerCase() === nombreProducto.toLowerCase())  
80         .forEach(producto => {  
81           resultados.push({  
82             Tienda: tienda.nombre,  
83             Categoría: categoría.nombre,  
84             Precio: `$$${producto.precio.toFixed(2)}`,  
85             Stock: producto.stock,  
86           });  
87         });  
88       });  
89     });  
90   });  
91   // Muestra los resultados como tabla o indica que no se encontró el producto  
92   if (resultados.length > 0) {  
93     console.table(resultados);  
94   }  
95   else {  
96     console.log(` ✖ No se encontró el producto activo: ${nombreProducto}`);  
97   }  
98 }  
99 // Ejecución del código: Mostrar tiendas y buscar productos  
100 mostrarTiendas(); // Muestra toda la información de las tiendas  
101 buscarProducto("Radio"); // Busca el producto "Radio"  
102 buscarProducto("Zapatos"); // Busca el producto "Zapatos"  
103 buscarProducto("Laptop"); // Busca un producto que no existe
```

The screenshot shows the VS Code editor with the `package-lock.json` file open. The file contains the metadata for the project, including the name, version, lockfile version, and the list of packages and their dependencies.

```
1 {  
2   "name": "node-typescript-project",  
3   "version": "1.0.0",  
4   "lockfileVersion": 3,  
5   "requires": true,  
6   "packages": {  
7     "": {  
8       "name": "node-typescript-project",  
9       "version": "1.0.0",  
10       "devDependencies": {  
11         "typescript": "^5.0.0"  
12       },  
13     },  
14     "node_modules/typescript": {  
15       "version": "5.7.3",  
16       "resolved": "https://registry.npmjs.org/typescript/-/typescript-5.7.3.tgz",  
17       "integrity": "sha512-84MwSjMEHP+FQRPy3pX9sTVV/INIex7Is9TL2Gm5FG/WG15qXkYz0k7/b1V/4Fd0zI12CBY1vGc4og/eus0fw==",  
18       "dev": true,  
19       "license": "Apache-2.0",  
20       "bin": {  
21         "tsc": "bin/tsc",  
22         "tsserver": "bin/tsserver"  
23       },  
24       "engines": {  
25         "node": ">=14.17"  
26       }  
27     }  
28   }  
29 }
```

This screenshot shows the Visual Studio Code editor with the `package.json` file open. The Explorer sidebar on the left shows the project structure with folders like `dist`, `app.js`, `node_modules`, `src`, and `ts`, and files like `package-lock.json`, `package.json`, and `tsconfig.json`. The `package.json` file is selected and its content is displayed in the main editor. The status bar at the bottom indicates the cursor is at line 1, column 1, with 4 spaces, UTF-8 encoding, and CRLF line endings.

```
1 {
2   "name": "node-typescript-project",
3   "version": "1.0.0",
4   "main": "dist/app.js",
5   "scripts": {
6     "build": "tsc",
7     "start": "node dist/app.js"
8   },
9   "dependencies": {},
10  "devDependencies": {
11    "typescript": "^5.0.0"
12  }
13 }
```

This screenshot shows the Visual Studio Code editor with the `tsconfig.json` file open. The Explorer sidebar on the left shows the project structure, and the `tsconfig.json` file is selected. The main editor displays the content of `tsconfig.json`. The status bar at the bottom indicates the cursor is at line 1, column 1, with 4 spaces, UTF-8 encoding, and CRLF line endings, and that the file is JSON with comments.

```
1 {
2   "compilerOptions": {
3     "target": "ES6",
4     "module": "commonjs",
5     "strict": true,
6     "esModuleInterop": true,
7     "outDir": "./dist",
8     "rootDir": "./src"
9   },
10  "include": ["src/**/*.ts"],
11  "exclude": ["node_modules"]
12 }
```