



Universidad Nacional Autónoma de México

Facultad de Estudios Superiores Aragón

Ingeniería en Computación

Asignatura: Estructura de datos

TAREA 3: Prácticas con clase Nodo

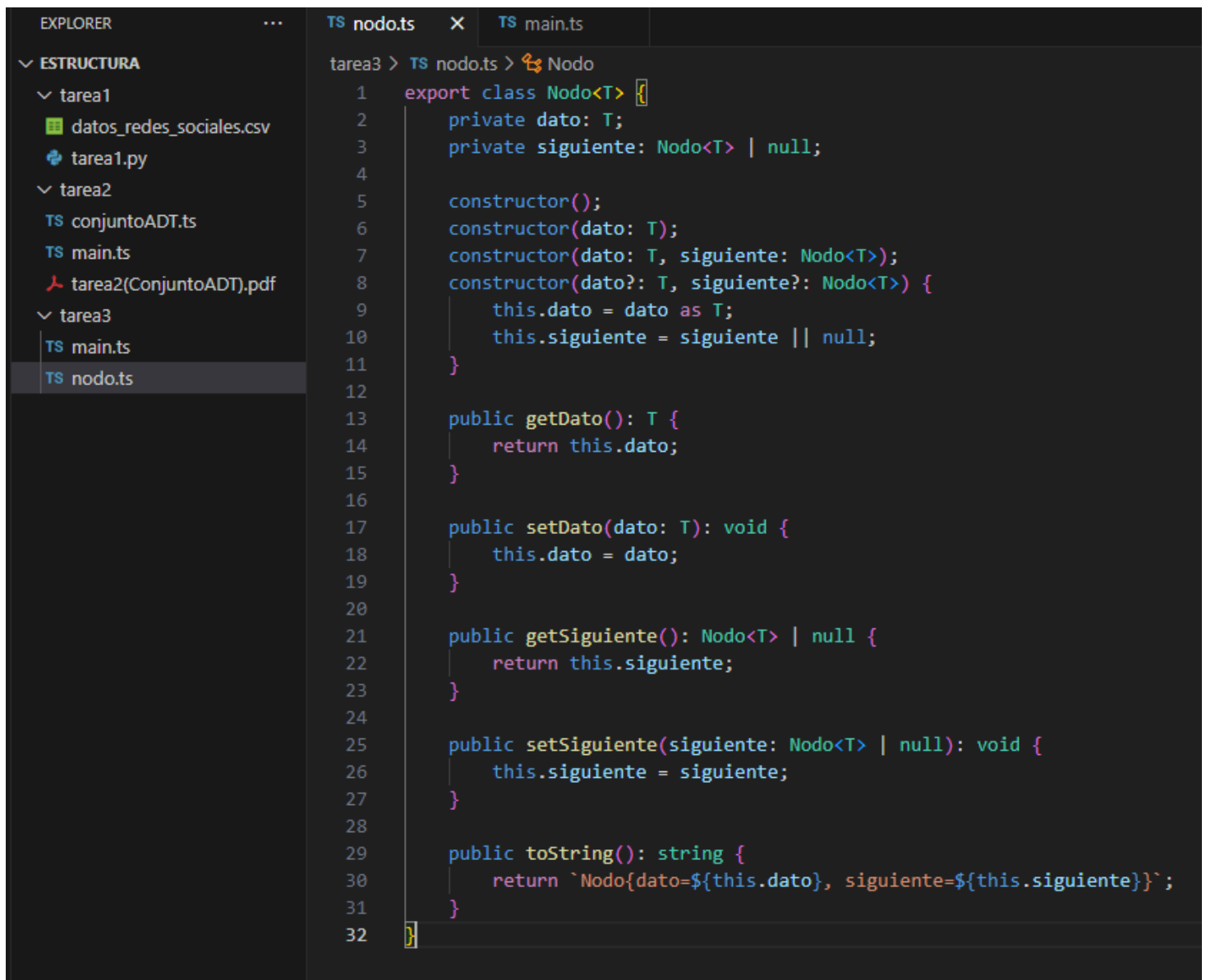
Profesor: Jesús Hernández Cabrera

Alumno: Juan Diego Ortiz Cruz

Grupo: 1360

Fecha: 27/08/2024

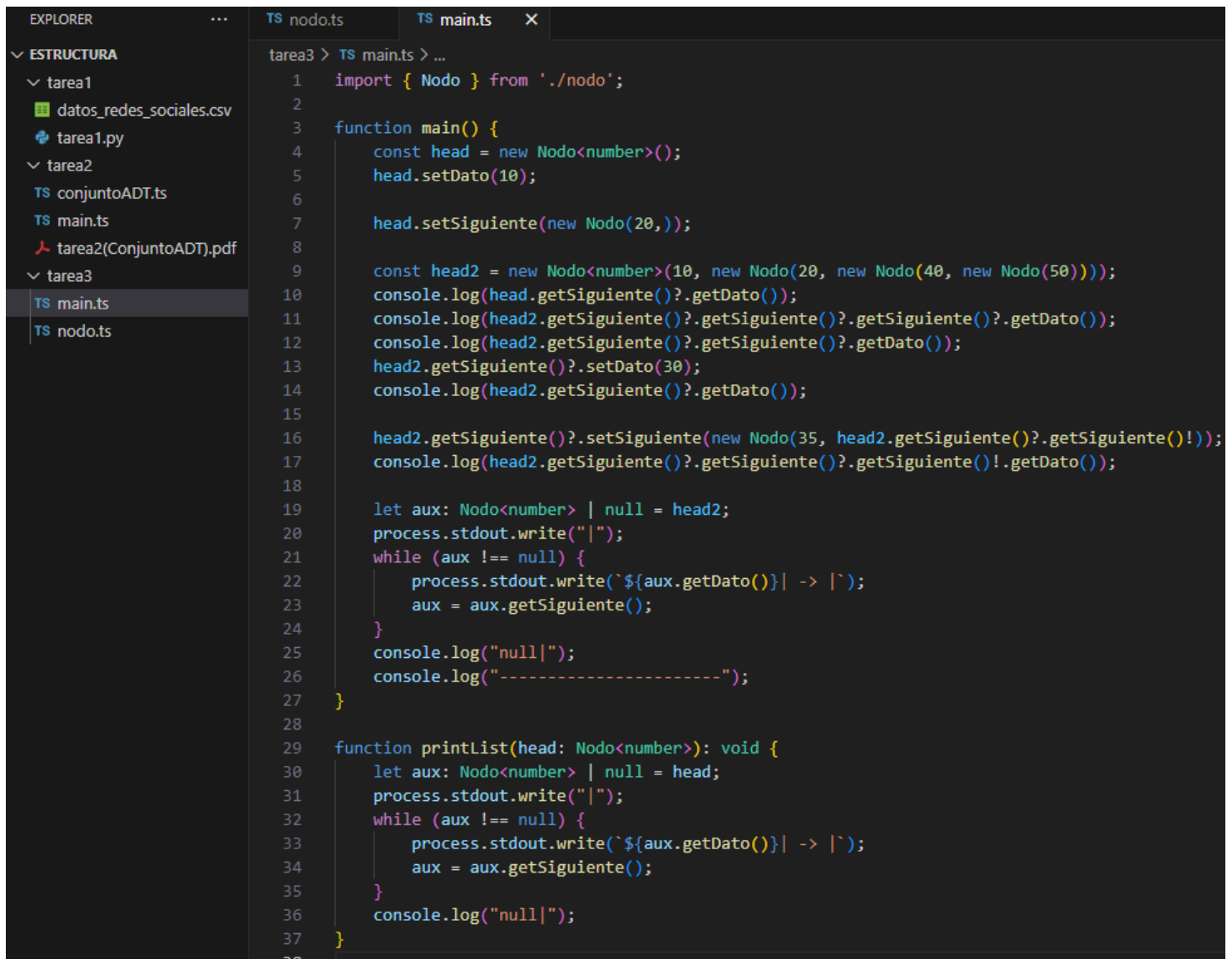
Capturas Clase Nodo:



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders 'tarea1', 'tarea2', and 'tarea3'. The 'tarea3' folder is expanded, showing files 'main.ts' and 'nodo.ts'. The 'nodo.ts' file is selected. The code editor shows the implementation of the 'Nodo' class in TypeScript. The class is generic, taking a type 'T' as a parameter. It has two private attributes: 'dato' of type 'T' and 'siguiente' of type 'Nodo<T> | null'. The class has four constructors: an empty constructor, a constructor for 'dato', a constructor for both 'dato' and 'siguiente', and a constructor with optional parameters. It also has four public methods: 'getDato()', 'setDato()', 'getSiguiente()', and 'setSiguiente()'. Finally, it has a 'toString()' method that returns a string representation of the node.

```
1 export class Nodo<T> {
2     private dato: T;
3     private siguiente: Nodo<T> | null;
4
5     constructor();
6     constructor(dato: T);
7     constructor(dato: T, siguiente: Nodo<T>);
8     constructor(dato?: T, siguiente?: Nodo<T>) {
9         this.dato = dato as T;
10        this.siguiente = siguiente || null;
11    }
12
13    public getDato(): T {
14        return this.dato;
15    }
16
17    public setDato(dato: T): void {
18        this.dato = dato;
19    }
20
21    public getSiguiente(): Nodo<T> | null {
22        return this.siguiente;
23    }
24
25    public setSiguiente(siguiente: Nodo<T> | null): void {
26        this.siguiente = siguiente;
27    }
28
29    public toString(): string {
30        return `Nodo{dato=${this.dato}, siguiente=${this.siguiente}}`;
31    }
32 }
```

Capturas Main:



The image shows a screenshot of the Visual Studio Code editor. On the left, the 'EXPLORER' sidebar displays the project structure under 'ESTRUCTURA'. The files listed are: 'tarea1' (containing 'datos_redes_sociales.csv' and 'tarea1.py'), 'tarea2' (containing 'conjuntoADT.ts' and 'main.ts'), and 'tarea3' (containing 'main.ts' and 'nodo.ts'). The 'main.ts' file in 'tarea3' is selected and open in the editor. The editor shows the following TypeScript code:

```
1 import { Nodo } from './nodo';
2
3 function main() {
4     const head = new Nodo<number>();
5     head.setDato(10);
6
7     head.setSiguiente(new Nodo(20,));
8
9     const head2 = new Nodo<number>(10, new Nodo(20, new Nodo(40, new Nodo(50))));
10    console.log(head.getSiguiente()?.getDato());
11    console.log(head2.getSiguiente()?.getSiguiente()?.getSiguiente()?.getDato());
12    console.log(head2.getSiguiente()?.getSiguiente()?.getDato());
13    head2.getSiguiente()?.setDato(30);
14    console.log(head2.getSiguiente()?.getDato());
15
16    head2.getSiguiente()?.setSiguiente(new Nodo(35, head2.getSiguiente()?.getSiguiente(!));
17    console.log(head2.getSiguiente()?.getSiguiente()?.getSiguiente(!).getDato());
18
19    let aux: Nodo<number> | null = head2;
20    process.stdout.write("|");
21    while (aux !== null) {
22        process.stdout.write(`${aux.getDato()}| -> |`);
23        aux = aux.getSiguiente();
24    }
25    console.log("null|");
26    console.log("-----");
27 }
28
29 function printList(head: Nodo<number>): void {
30     let aux: Nodo<number> | null = head;
31     process.stdout.write("|");
32     while (aux !== null) {
33         process.stdout.write(`${aux.getDato()}| -> |`);
34         aux = aux.getSiguiente();
35     }
36     console.log("null|");
37 }
```

ESTRUCTURA

▼ tarea1

datos_redes_sociales.csv

tarea1.py

▼ tarea2

TS conjuntoADT.ts

TS main.ts

tarea2(ConjuntoADT).pdf

▼ tarea3

TS main.ts

TS nodo.ts

tarea3 > TS main.ts > main

```

38
39 function practicas() {
40     // Crear la estructura de la imagen anexa
41     let head = new Nodo<number>(100);
42     head.setSiguiente(new Nodo(200, new Nodo(300, new Nodo(400, new Nodo(600))));
43
44     // Imprimir todo
45     printList(head);
46
47     // Cambiar el valor del 3er nodo de 300 a 333
48     head.getSiguiente()?.getSiguiente()?.setDato(333);
49
50     // Imprimir todo nuevamente
51     console.log("\nDespués de cambiar 300 a 333:");
52     printList(head);
53
54     // Insertar un nodo 700 después del nodo 600 (Al final)
55     let aux: Nodo<number> | null = head;
56     while (aux.getSiguiente() !== null) {
57         aux = aux.getSiguiente()!;
58     }
59     aux.setSiguiente(new Nodo(700));
60
61     // Imprimir todo nuevamente
62     console.log("\nDespués de insertar 700 al final:");
63     printList(head);
64
65     // Insertar un nodo con valor 50 al principio (antes del nodo 100)
66     head = new Nodo(50, head);
67
68     // Imprimir todo nuevamente
69     console.log("\nDespués de insertar 50 al principio:");
70     printList(head);
71 }
72
73 // Ejecutar la función principal
74 main();
75
76 // Ejecutar las funciones de prácticas
77 practicas();

```

Capturas consola (Ejecucion):

```

Node.js v20.16.0
PS C:\Users\juani\Documents\Estructura> cd .\tarea3\
PS C:\Users\juani\Documents\Estructura\tarea3> ts-node main.ts
20
50
40
30
40
|10| -> |30| -> |35| -> |40| -> |50| -> |null|
-----
|100| -> |200| -> |300| -> |400| -> |600| -> |null|

Después de cambiar 300 a 333:
|100| -> |200| -> |333| -> |400| -> |600| -> |null|

Después de insertar 700 al final:
|100| -> |200| -> |333| -> |400| -> |600| -> |700| -> |null|

Después de insertar 50 al principio:
|50| -> |100| -> |200| -> |333| -> |400| -> |600| -> |700| -> |null|
PS C:\Users\juani\Documents\Estructura\tarea3>

```