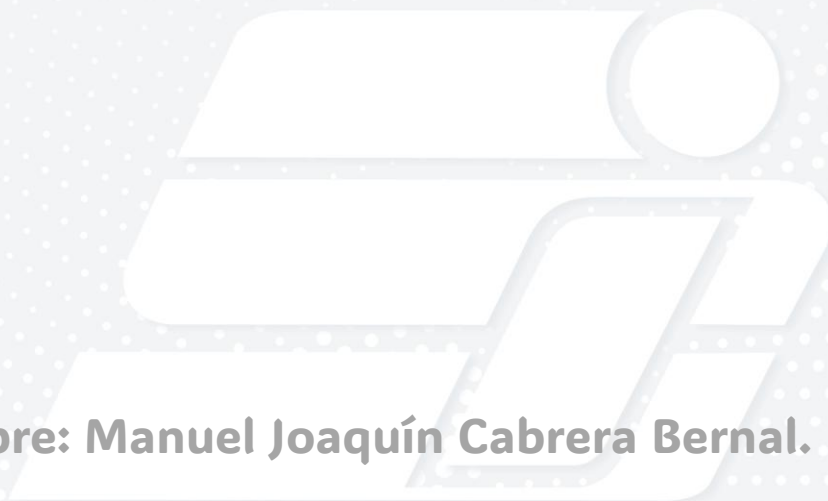


INSTITUTO SUPERIOR PARTICULAR SUDAMERICANO



Nombre: Manuel Joaquín Cabrera Bernal.

Carrera: Desarrollo de Software.

**Materia: METODOLOGÍA PARA RESOLVER
PROBLEMAS INFORMÁTICOS.**

Clase: Semana 8

Fecha: 30/11/2025



1. OBJETIVOS DE LA CLASE:

Aprender cómo se ocupa el Typescript.

2. DESARROLLO:

EJERCICIO 1: En la clase Banco se implementan métodos simples para manejar operaciones básicas: depositar, retirar y transferir. Cada método trabaja con números recibidos por parámetros y retorna el nuevo saldo. depositar suma el monto al saldo, retirar valida que no se retire más de lo disponible usando una comparación y lanza un error con throw si no hay fondos suficientes, mientras que transferir funciona igual que retirar, pero representando el envío de dinero desde una cuenta. La clase no almacena datos, simplemente calcula y devuelve los nuevos saldos según lo solicitado.

EJERCICIO 2: En la clase Estudiante se crean tres métodos que ayudan a manejar el rendimiento de un alumno. El método agregarNota recibe el promedio actual y una nueva nota, y calcula un promedio simple entre ambos valores. El método aprobar compara la nota final con la nota mínima requerida y devuelve true o false según si alcanza el puntaje. Finalmente, calcularFaltas suma las faltas previas con las nuevas para devolver el total acumulado. La clase solo hace cálculos directos sin almacenamiento interno, cumpliendo las reglas dadas.

EJERCICIO 3: La clase Auto simula acciones básicas de un vehículo mediante tres métodos. acelerar incrementa la velocidad actual sumando los km/h indicados. frenar resta un valor a la velocidad, pero utiliza una condición para evitar que el resultado sea negativo, devolviendo 0 cuando la resta queda por debajo de ese límite. El método recorrer calcula la velocidad media dividiendo la distancia entre el tiempo. Cada operación sigue un cálculo matemático simple y retorna el resultado directamente.

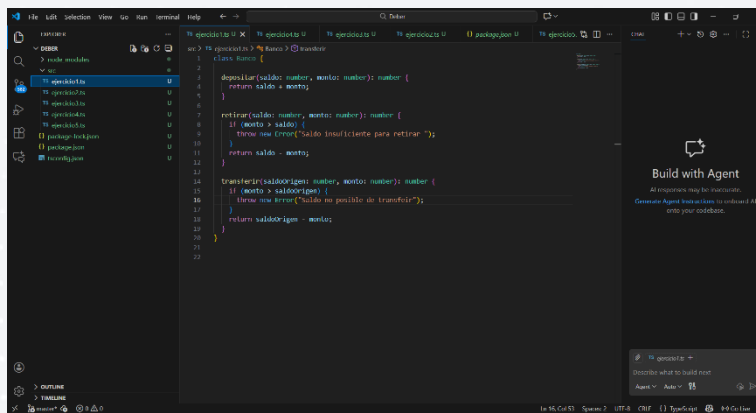
EJERCICIO 4: La clase Restaurante calcula el costo de un pedido, aplica descuentos y estima propinas. El método calcularTotal multiplica precio por cantidad para saber cuánto cuesta el producto. aplicarDescuento reduce el total según el porcentaje ingresado usando regla de tres ($\text{total} * \text{descuento} / 100$). calcularPropina hace un cálculo similar, pero solo devuelve el valor de la propina sin sumarlo al total. Cada método devuelve un número y está pensado para trabajar de forma independiente sin almacenar datos.



EJERCICIO 5: En la clase Hospital se incluyen operaciones útiles para cálculos médicos básicos. calcularDosis multiplica el peso del paciente por los miligramos por kilogramo para obtener la dosis total del medicamento. calcularIMC usa la fórmula estándar del índice de masa corporal dividiendo el peso para la altura al cuadrado. registrarVisitas suma las visitas nuevas al total actual. La clase funciona únicamente como calculadora, devolviendo valores correctos según las fórmulas indicadas.

3. CAPTURAS:

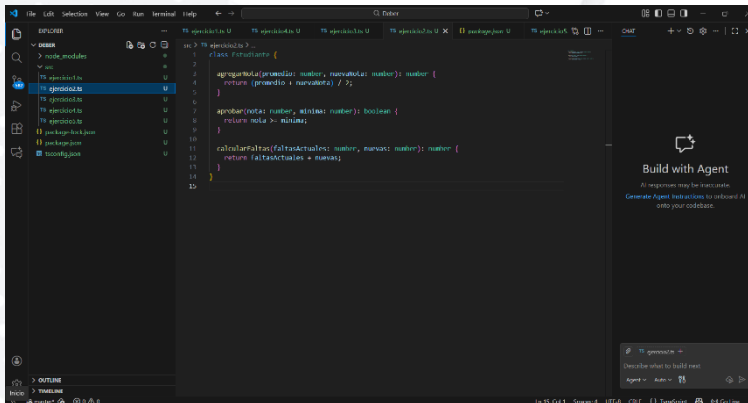
EJERCICIO 1:



```

1  class Hospital {
2      depositar(saldo: number, monto: number): number {
3          return saldo + monto;
4      }
5      retirar(saldo: number, monto: number): number {
6          if (monto > saldo) {
7              throw new Error("saldo insuficiente para retirar.");
8          }
9          return saldo - monto;
10     }
11     transfer(saldoOrigen: number, monto: number): number {
12         if (monto > saldoOrigen) {
13             throw new Error("saldo no posible de transferir.");
14         }
15         return saldoOrigen - monto;
16     }
17 }
    
```

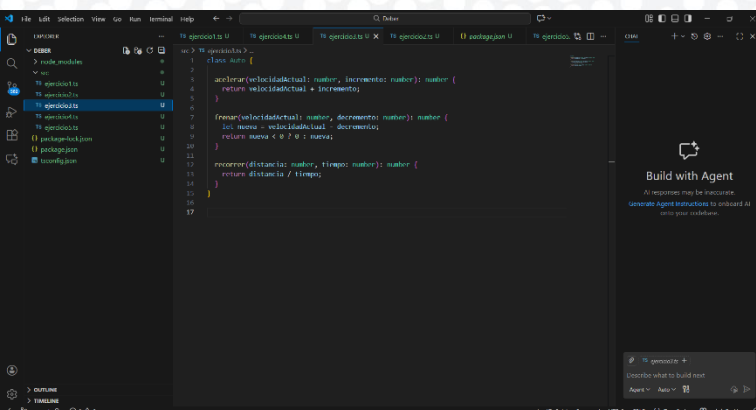
EJERCICIO2:



```

1  class Faltas {
2      agregarFaltas(promedio: number, nuevaFalta: number): number {
3          return (promedio + nuevaFalta) / 2;
4      }
5      aprobarNotas(numero: number, notas: number): boolean {
6          return nota > minimo;
7      }
8      calcularFaltas(faltasActuales: number, nuevas: number): number {
9          return faltasActuales + nuevas;
10     }
11 }
    
```

EJERCICIO 3:



```

1  class Velocidad {
2      acelerar(velocidadActual: number, incremento: number): number {
3          return velocidadActual + incremento;
4      }
5      frenar(velocidadActual: number, decremento: number): number {
6          let nueva = velocidadActual - decremento;
7          return nueva < 0 ? 0 : nueva;
8      }
9      recorrer(distancia: number, tiempo: number): number {
10         return distancia / tiempo;
11     }
12 }
    
```



www

EJERCICIO 4:

```

class Restaurante {
    calcularTotal(precio: number, cantidad: number): number {
        return precio * cantidad;
    }
    aplicarDescuento(total: number, descuento: number): number {
        return total - (total * descuento / 100);
    }
    calcularPorcentaje(total: number, porcentaje: number): number {
        return total * (porcentaje / 100);
    }
}

```

EJERCICIO 5:

```

class Banco {
    depositar(saldo: number, monto: number): number {
        return saldo + monto;
    }
    retirar(saldo: number, monto: number): number {
        if (monto > saldo) {
            throw new Error("Saldo insuficiente para retirar");
        }
    }
    registrarVisitas(visitasAnteriores: number, nuevas: number): number {
        return visitasAnteriores + nuevas;
    }
}

```

4. Códigos usados:

Código 1:

class Banco {

```

    depositar(saldo: number, monto: number): number {
        return saldo + monto;
    }

```

```

    retirar(saldo: number, monto: number): number {
        if (monto > saldo) {
            throw new Error("Saldo insuficiente para retirar");
        }
    }

```




```
return saldo - monto;
}

transferir(saldoOrigen: number, monto: number): number {
  if (monto > saldoOrigen) {
    throw new Error("Saldo no posible de transfeir");
  }
  return saldoOrigen - monto;
}
}
```

Código 2:

```
class Estudiante {
```

```
  agregarNota(promedio: number, nuevaNota: number): number {
    return (promedio + nuevaNota) / 2;
  }

  aprobar(nota: number, minima: number): boolean {
    return nota >= minima;
  }

  calcularFaltas(faltasActuales: number, nuevas: number): number {
    return faltasActuales + nuevas;
  }
}
```

Código 3:

```
class Auto {
```

```
  acelerar(velocidadActual: number, incremento: number): number {
    return velocidadActual + incremento;
  }

  frenar(velocidadActual: number, decremento: number): number {
    let nueva = velocidadActual - decremento;
    return nueva < 0 ? 0 : nueva;
  }
}
```



```
recorrer(distancia: number, tiempo: number): number {  
    return distancia / tiempo;  
}  
}
```

Código 4:

```
class Restaurante {  
  
    calcularTotal(precio: number, cantidad: number): number {  
        return precio * cantidad;  
    }  
  
    aplicarDescuento(total: number, descuento: number): number {  
        return total - (total * descuento / 100);  
    }  
  
    calcularPropina(total: number, porcentaje: number): number {  
        return total * (porcentaje / 100);  
    }  
}
```

Código 5:

```
class Hospital {  
  
    calcularDosis(peso: number, mgPorKg: number): number {  
        return peso * mgPorKg;  
    }  
  
    calcularIMC(peso: number, altura: number): number {  
        return peso / (altura * altura);  
    }  
  
    registrarVisitas(visitasActuales: number, nuevas: number): number {  
        return visitasActuales + nuevas;  
    }  
}
```



5. Opinión personal:

Al mi parecer me gusto aprender como se ocupa el Typescript ya que con ello pude ver cómo se ocupa otro lenguaje de programación.

