

Tecnicatura Universitaria en Programación

Materia: Programación I

Profesor: Ariel Betancoud

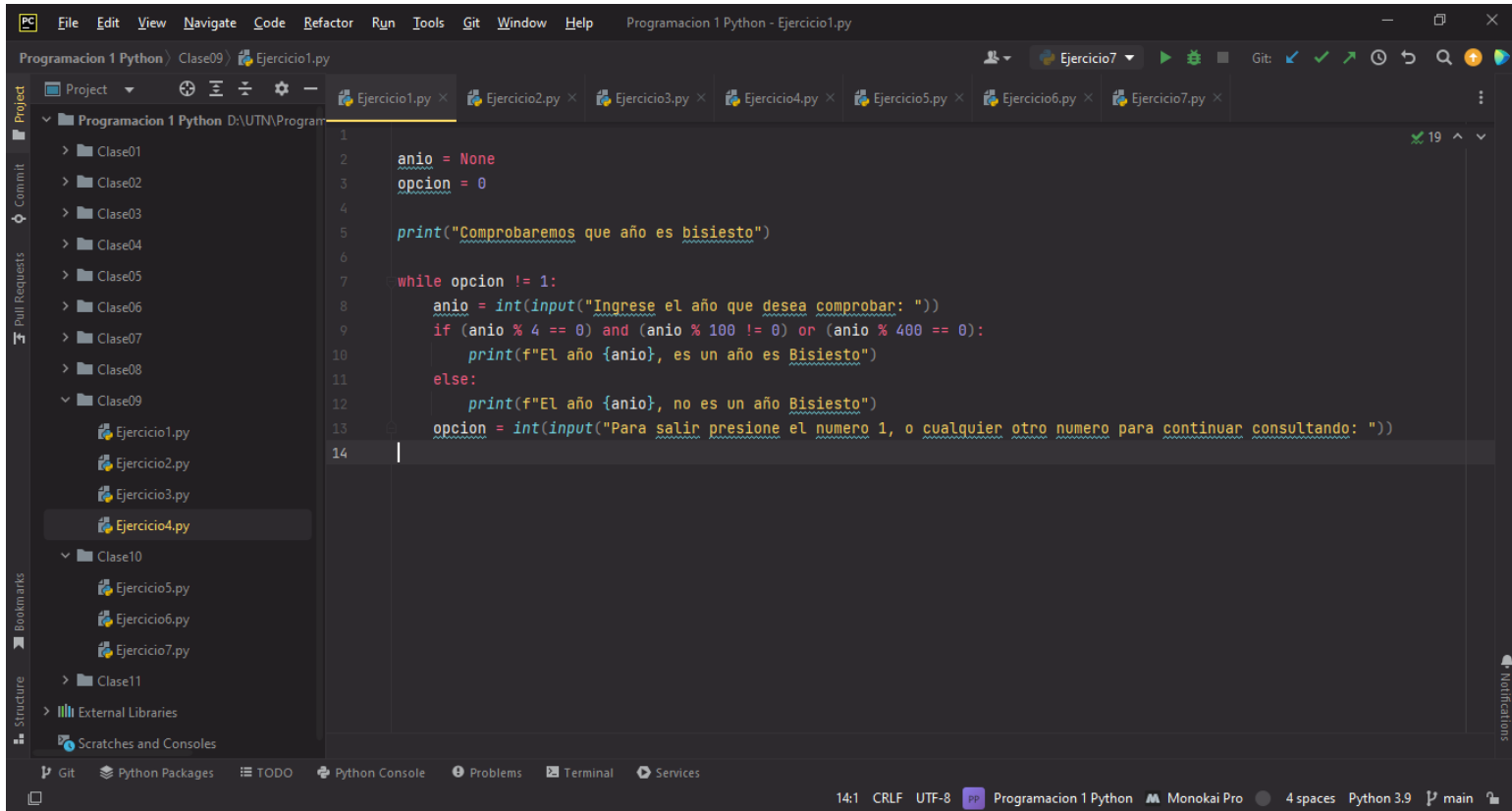
Alumno: Ignacio Villarraza

Grupo: BeeCoders

Capturas Ejercicios

Programación I - Python

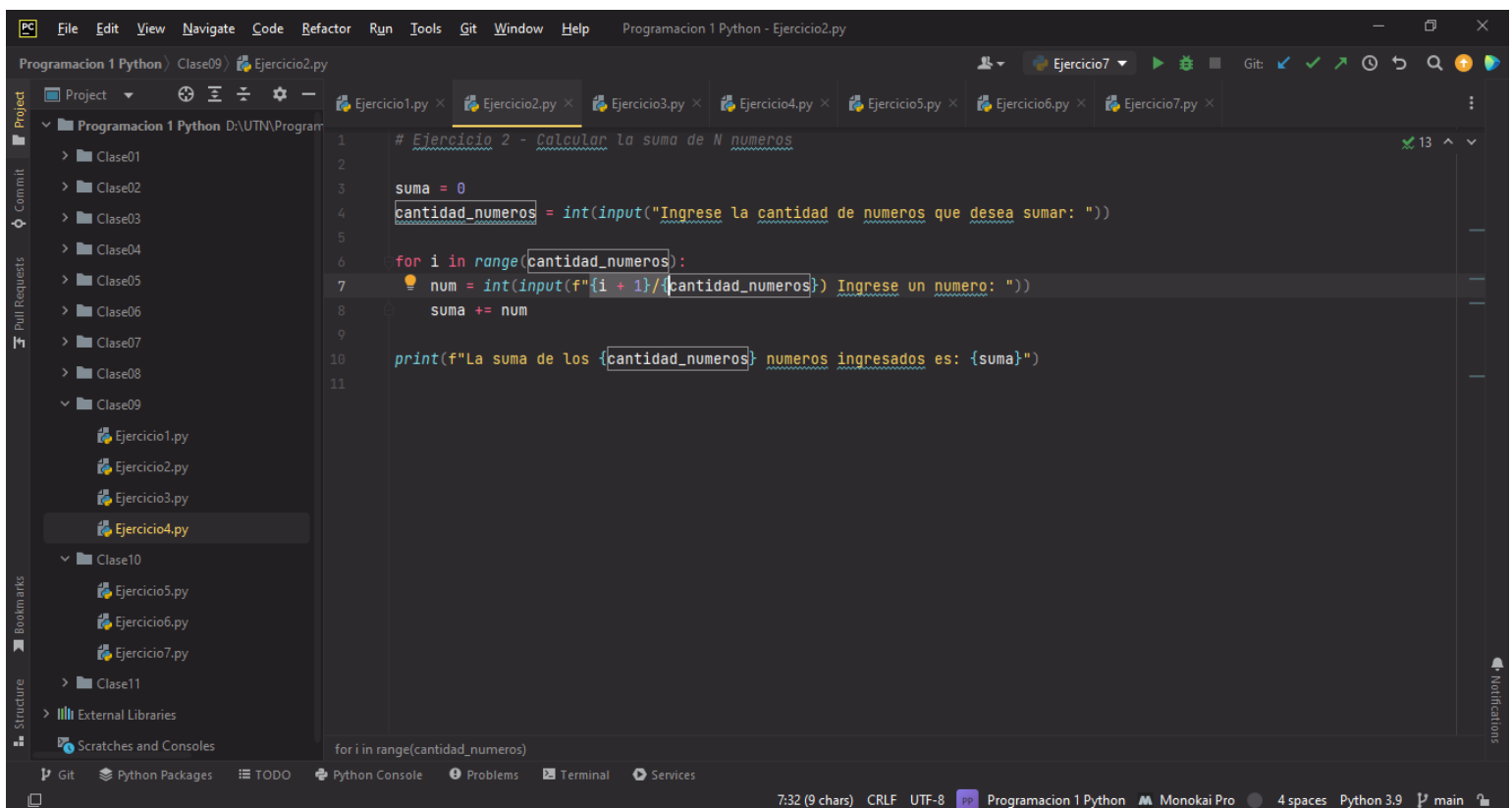
Ejercicio 1



The screenshot shows a code editor with the file 'Ejercicio1.py' open. The code is a Python script for checking if a year is a leap year. It uses a while loop to repeatedly prompt the user for a year until they press '1' to exit. The code is as follows:

```
1 anio = None
2 opcion = 0
3
4 print("Comprobaremos que año es bisiestro")
5
6 while opcion != 1:
7     anio = int(input("Ingrese el año que desea comprobar: "))
8     if (anio % 4 == 0) and (anio % 100 != 0) or (anio % 400 == 0):
9         print(f"El año {anio}, es un año es Bisiestro")
10    else:
11        print(f"El año {anio}, no es un año Bisiestro")
12    opcion = int(input("Para salir presione el numero 1, o cualquier otro numero para continuar consultando: "))
13
14
```

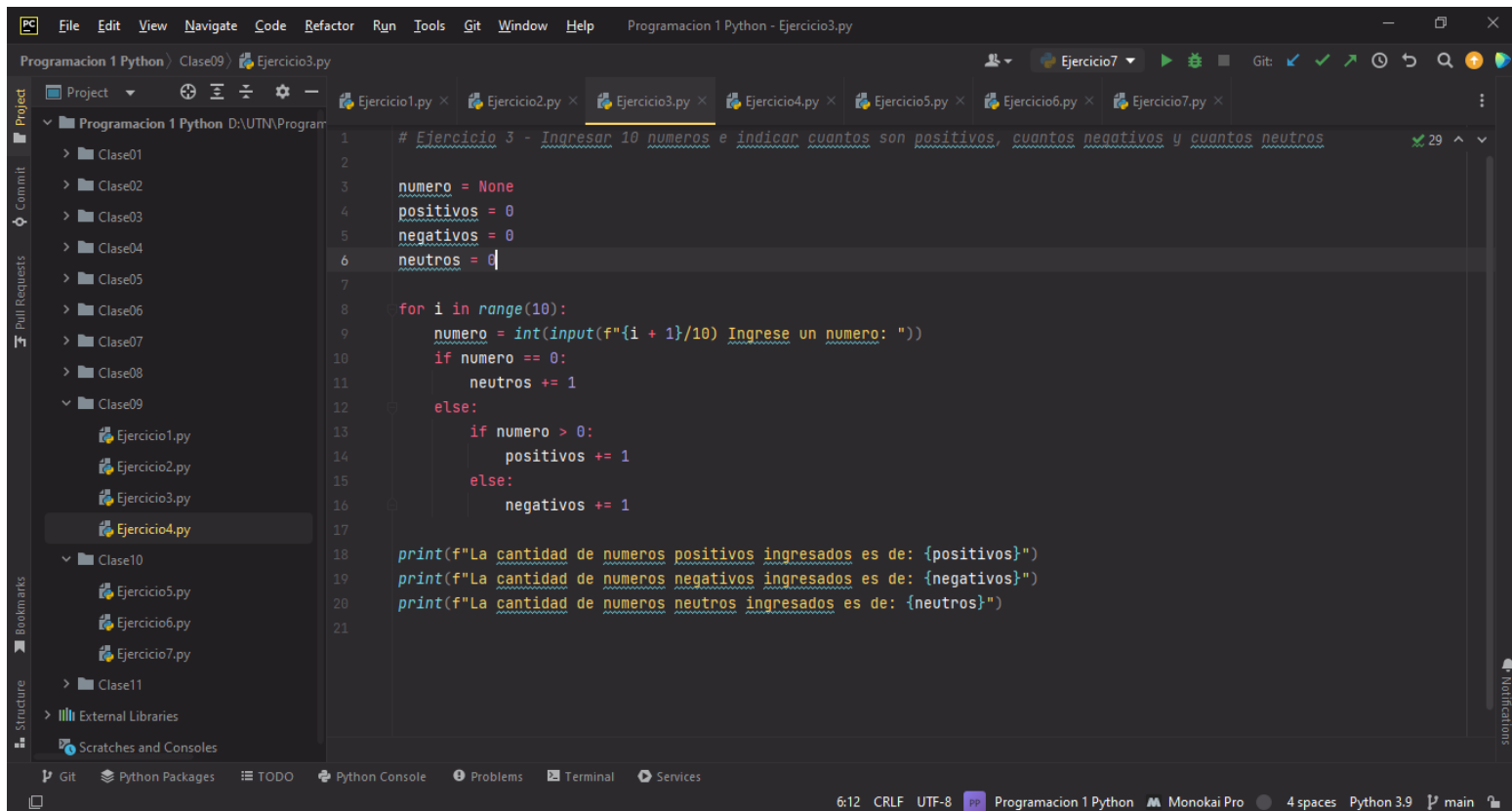
Ejercicio 2



The screenshot shows a code editor with the file 'Ejercicio2.py' open. The code is a Python script for calculating the sum of N numbers. It prompts the user for the quantity of numbers, then uses a for loop to sum them. The code is as follows:

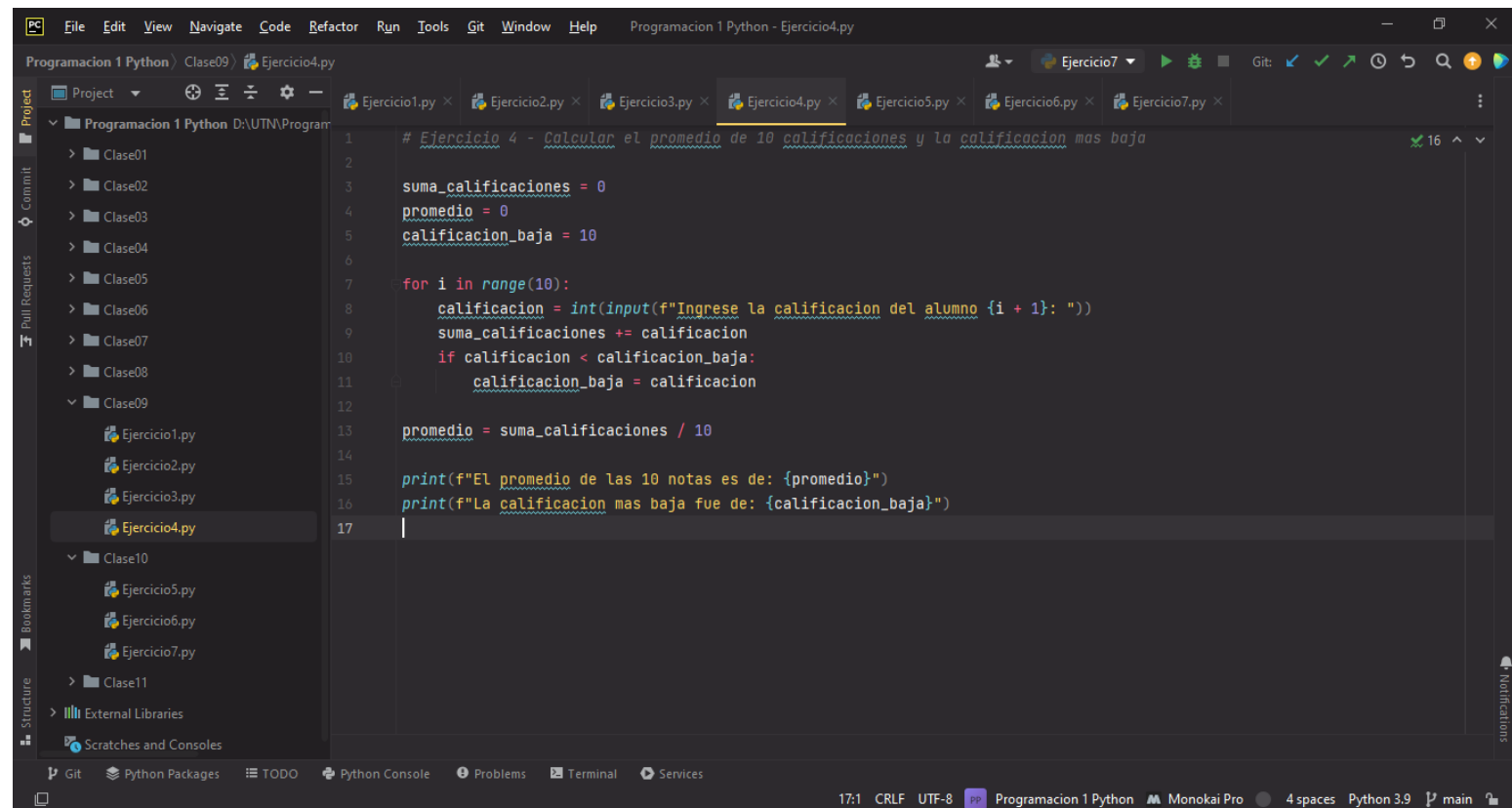
```
1 # Ejercicio 2 - Calcular la suma de N numeros
2
3 suma = 0
4 cantidad_numeros = int(input("Ingrese la cantidad de numeros que desea sumar: "))
5
6 for i in range(cantidad_numeros):
7     num = int(input(f"{i + 1}/{cantidad_numeros} Ingrese un numero: "))
8     suma += num
9
10 print(f"La suma de los {cantidad_numeros} numeros ingresados es: {suma}")
11
```

Ejercicio 3



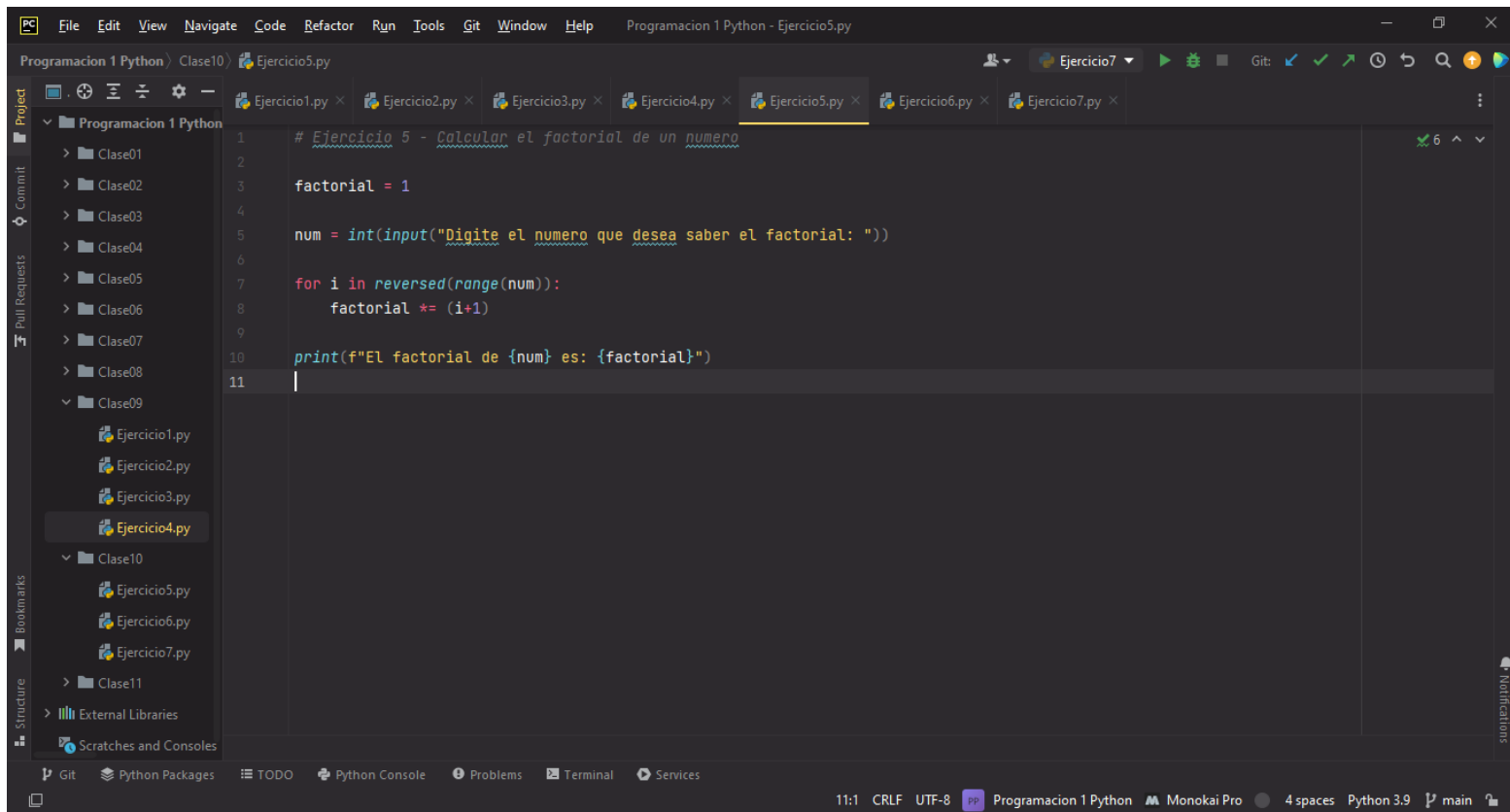
```
1 # Ejercicio 3 - Ingresar 10 numeros e indicar cuantos son positivos, cuantos negativos y cuantos neutros
2
3 numero = None
4 positivos = 0
5 negativos = 0
6 neutros = 0
7
8 for i in range(10):
9     numero = int(input(f"{i + 1}/10 Ingrese un numero: "))
10    if numero == 0:
11        neutros += 1
12    else:
13        if numero > 0:
14            positivos += 1
15        else:
16            negativos += 1
17
18 print(f"La cantidad de numeros positivos ingresados es de: {positivos}")
19 print(f"La cantidad de numeros negativos ingresados es de: {negativos}")
20 print(f"La cantidad de numeros neutros ingresados es de: {neutros}")
21
```

Ejercicio 4



```
1 # Ejercicio 4 - Calcular el promedio de 10 calificaciones y la calificacion mas baja
2
3 suma_calificaciones = 0
4 promedio = 0
5 calificacion_baja = 10
6
7 for i in range(10):
8     calificacion = int(input(f"Ingrese la calificacion del alumno {i + 1}: "))
9     suma_calificaciones += calificacion
10    if calificacion < calificacion_baja:
11        calificacion_baja = calificacion
12
13 promedio = suma_calificaciones / 10
14
15 print(f"El promedio de las 10 notas es de: {promedio}")
16 print(f"La calificacion mas baja fue de: {calificacion_baja}")
17
```

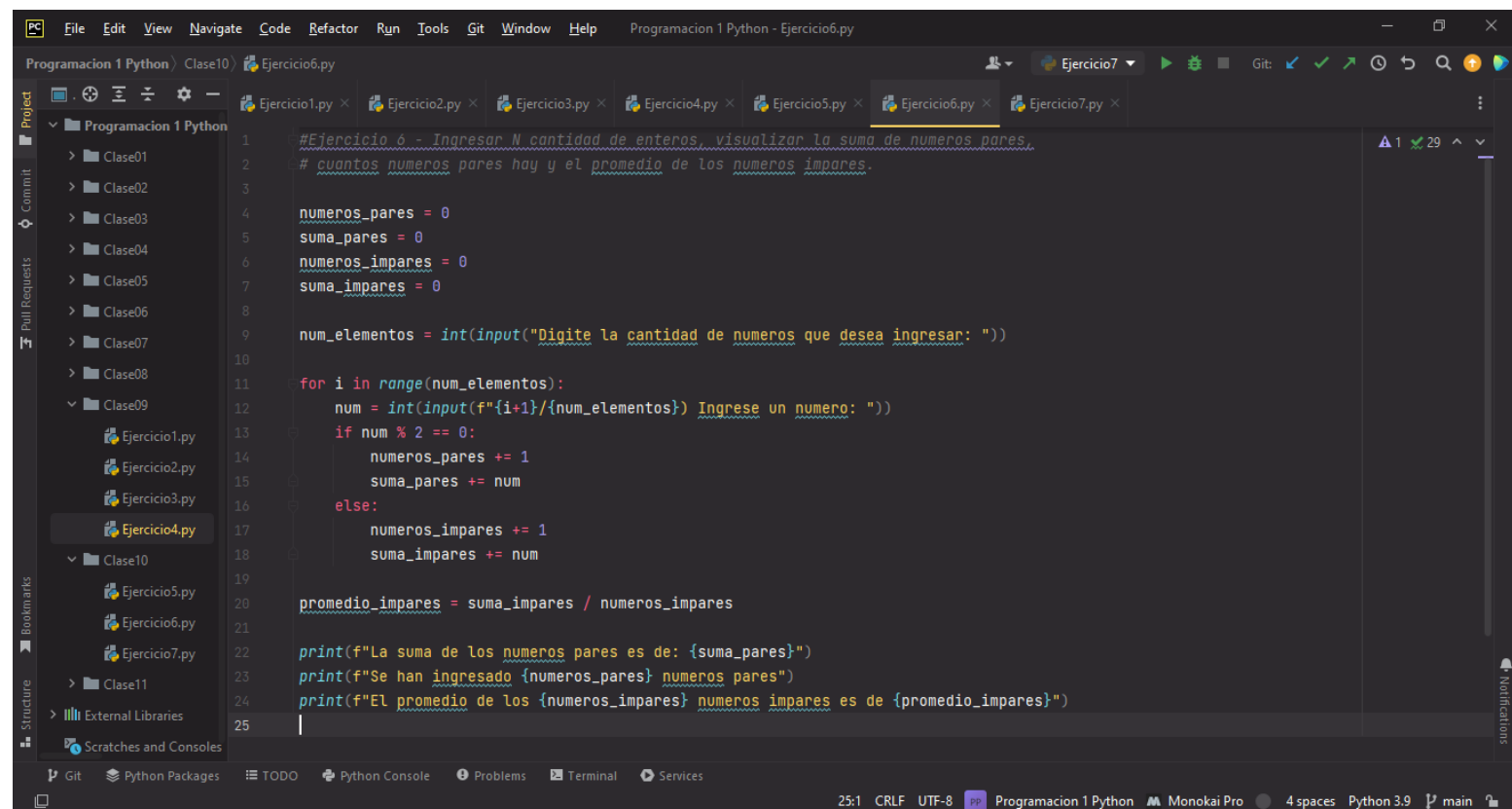
Ejercicio 5



The screenshot shows the VS Code editor with the file 'Ejercicio5.py' open. The code calculates the factorial of a number entered by the user. The left sidebar shows the project structure with folders 'Clase01' through 'Clase11' and files 'Ejercicio1.py' through 'Ejercicio7.py'. The bottom status bar shows '11:1 CRLF UTF-8 Programacion 1 Python Monokai Pro 4 spaces Python 3.9 main'.

```
1 # Ejercicio 5 - Calcular el factorial de un numero
2
3 factorial = 1
4
5 num = int(input("Digite el numero que desea saber el factorial: "))
6
7 for i in reversed(range(num)):
8     factorial *= (i+1)
9
10 print(f"El factorial de {num} es: {factorial}")
11
```

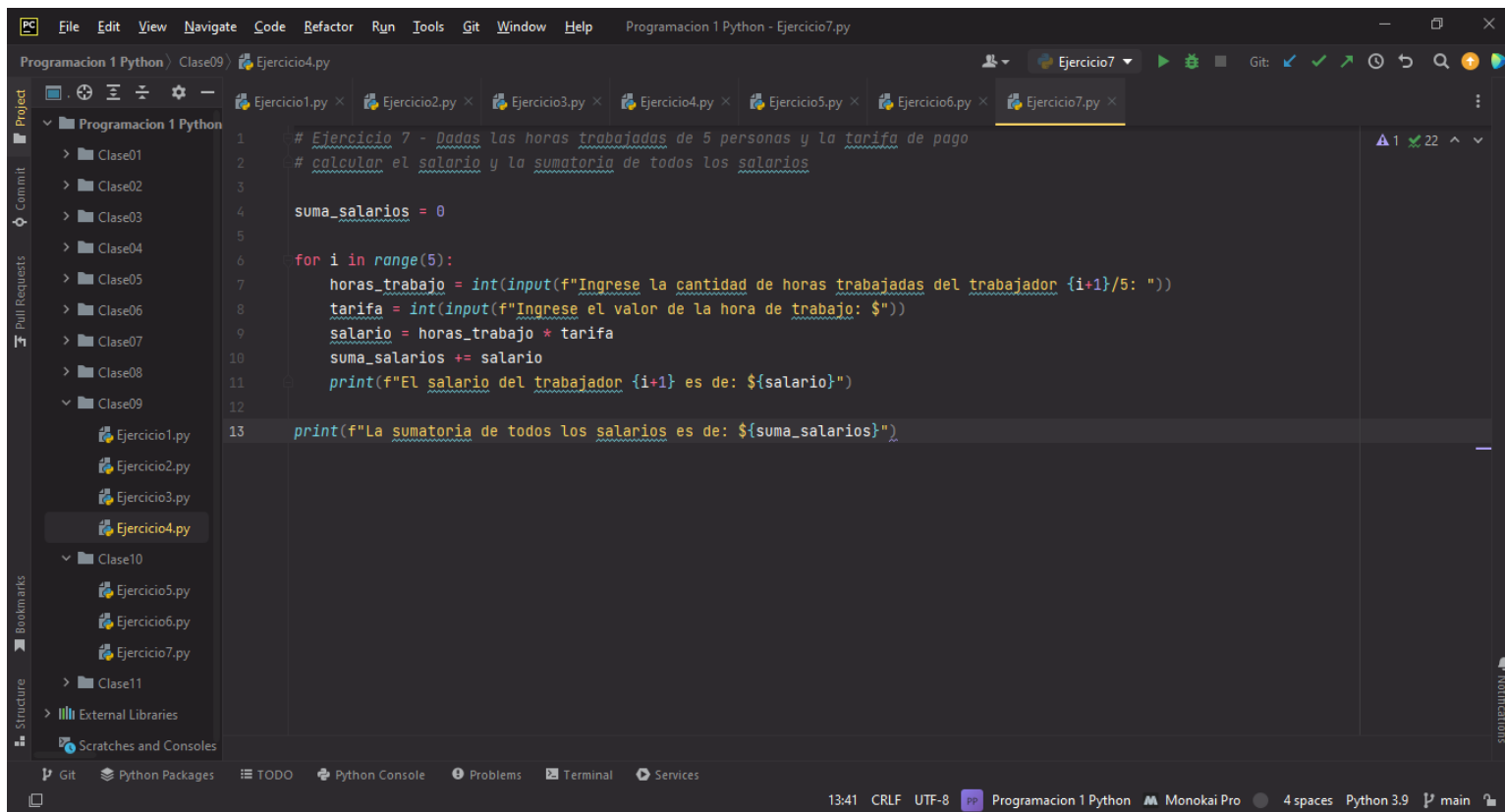
Ejercicio 6



The screenshot shows the VS Code editor with the file 'Ejercicio6.py' open. The code calculates the sum of even numbers and the average of odd numbers from a list of numbers entered by the user. The left sidebar shows the project structure with folders 'Clase01' through 'Clase11' and files 'Ejercicio1.py' through 'Ejercicio7.py'. The bottom status bar shows '25:1 CRLF UTF-8 Programacion 1 Python Monokai Pro 4 spaces Python 3.9 main'.

```
1 #Ejercicio 6 - Ingresar N cantidad de enteros, visualizar la suma de numeros pares,
2 # cuantos numeros pares hay y el promedio de los numeros impares.
3
4 numeros_pares = 0
5 suma_pares = 0
6 numeros_impares = 0
7 suma_impares = 0
8
9 num_elementos = int(input("Digite la cantidad de numeros que desea ingresar: "))
10
11 for i in range(num_elementos):
12     num = int(input(f"{i+1}/{num_elementos} Ingrese un numero: "))
13     if num % 2 == 0:
14         numeros_pares += 1
15         suma_pares += num
16     else:
17         numeros_impares += 1
18         suma_impares += num
19
20 promedio_impares = suma_impares / numeros_impares
21
22 print(f"La suma de los numeros pares es de: {suma_pares}")
23 print(f"Se han ingresado {numeros_pares} numeros pares")
24 print(f"El promedio de los {numeros_impares} numeros impares es de {promedio_impares}")
25
```

Ejercicio 7



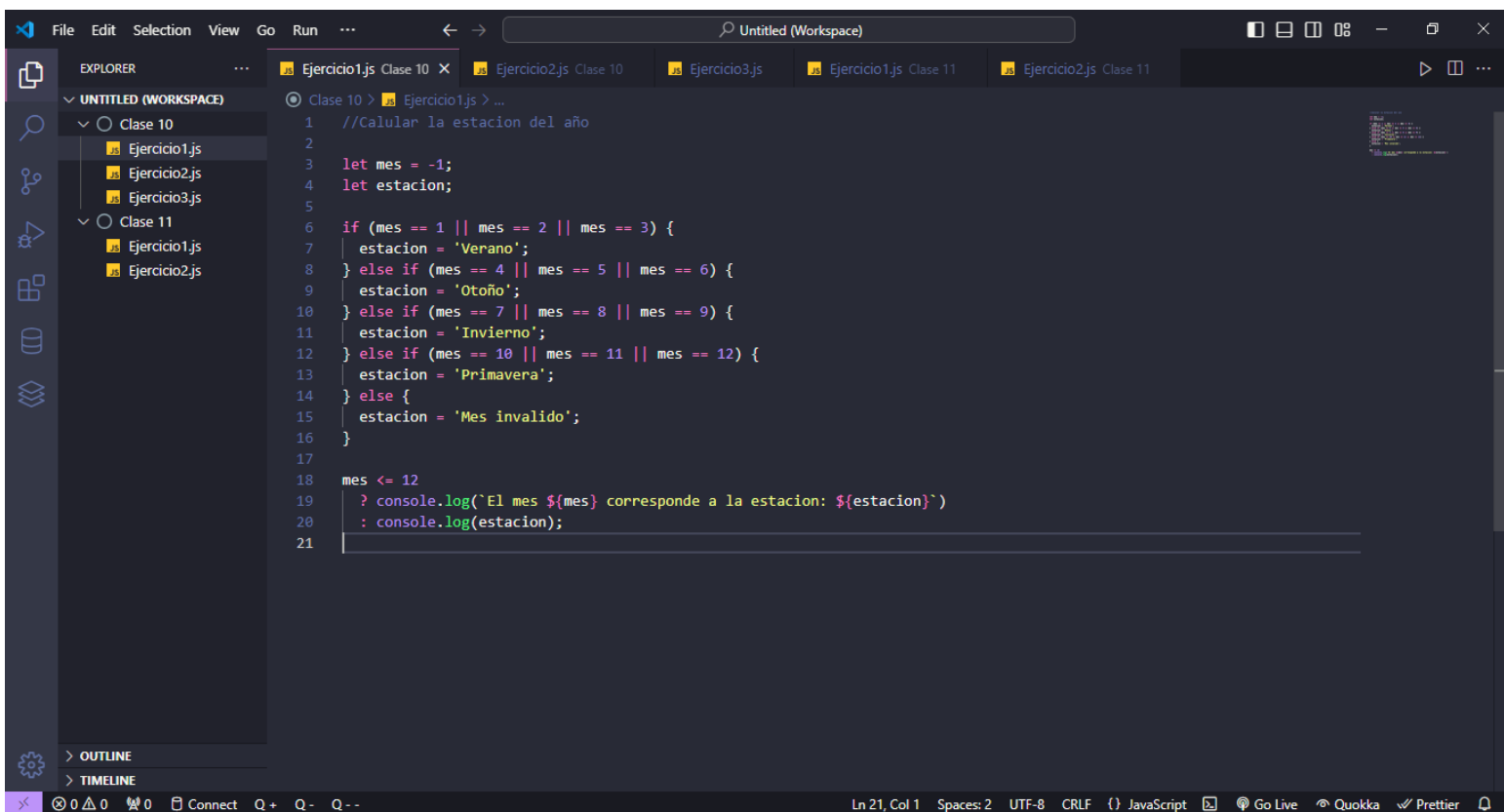
The screenshot shows a code editor with a dark theme. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, Git, Window, and Help. The title bar reads 'Programacion 1 Python - Ejercicio7.py'. The left sidebar shows a project structure with folders 'Clase01' through 'Clase11' and files 'Ejercicio1.py' through 'Ejercicio7.py'. The main editor area displays the following Python code:

```
1  # Ejercicio 7 - Dadas las horas trabajadas de 5 personas y la tarifa de pago
2  # calcular el salario y la sumatoria de todos los salarios
3
4  suma_salarios = 0
5
6  for i in range(5):
7      horas_trabajo = int(input(f"Ingrese la cantidad de horas trabajadas del trabajador {i+1}/5: "))
8      tarifa = int(input(f"Ingrese el valor de la hora de trabajo: $"))
9      salario = horas_trabajo * tarifa
10     suma_salarios += salario
11     print(f"El salario del trabajador {i+1} es de: ${salario}")
12
13 print(f"La sumatoria de todos los salarios es de: ${suma_salarios}")
```

The status bar at the bottom shows the time 13:41, encoding CRLF and UTF-8, a Python icon, the file name 'Programacion 1 Python', the theme 'Monokai Pro', 4 spaces, Python 3.9, and the branch 'main'.

Programación I - Javascript

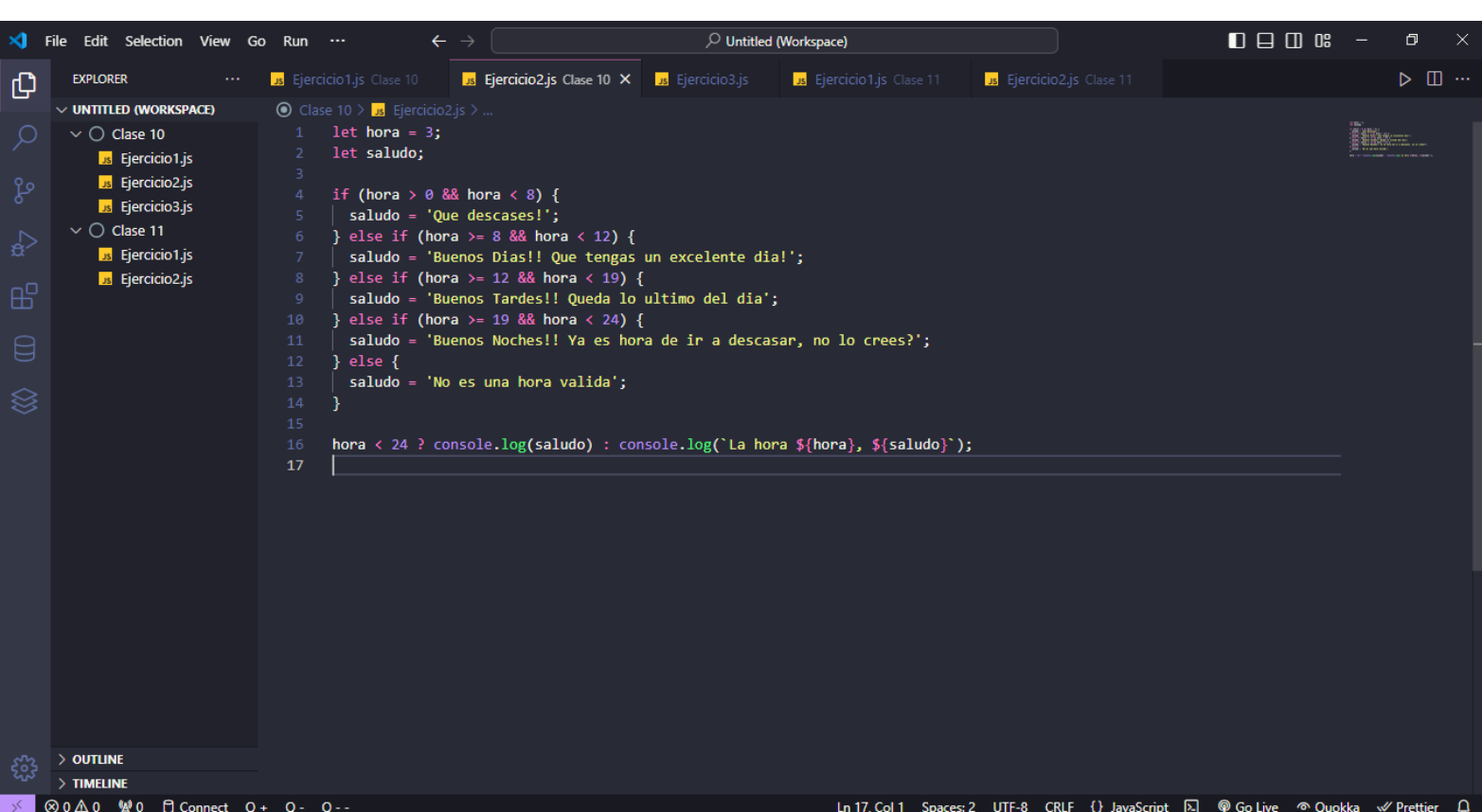
Clase 10 - Ejercicio 1



The screenshot shows the VS Code editor with a workspace named 'Untitled (Workspace)'. The Explorer panel on the left shows a project structure with 'Clase 10' and 'Clase 11' folders, each containing 'Ejercicio1.js' and 'Ejercicio2.js'. The main editor area displays the code for 'Ejercicio1.js' in 'Clase 10'. The code is as follows:

```
1 //Calular la estacion del año
2
3 let mes = -1;
4 let estacion;
5
6 if (mes == 1 || mes == 2 || mes == 3) {
7   estacion = 'Verano';
8 } else if (mes == 4 || mes == 5 || mes == 6) {
9   estacion = 'Otoño';
10 } else if (mes == 7 || mes == 8 || mes == 9) {
11   estacion = 'Invierno';
12 } else if (mes == 10 || mes == 11 || mes == 12) {
13   estacion = 'Primavera';
14 } else {
15   estacion = 'Mes invalido';
16 }
17
18 mes <= 12
19 ? console.log('El mes ${mes} corresponde a la estacion: ${estacion}')
20 : console.log(estacion);
21
```

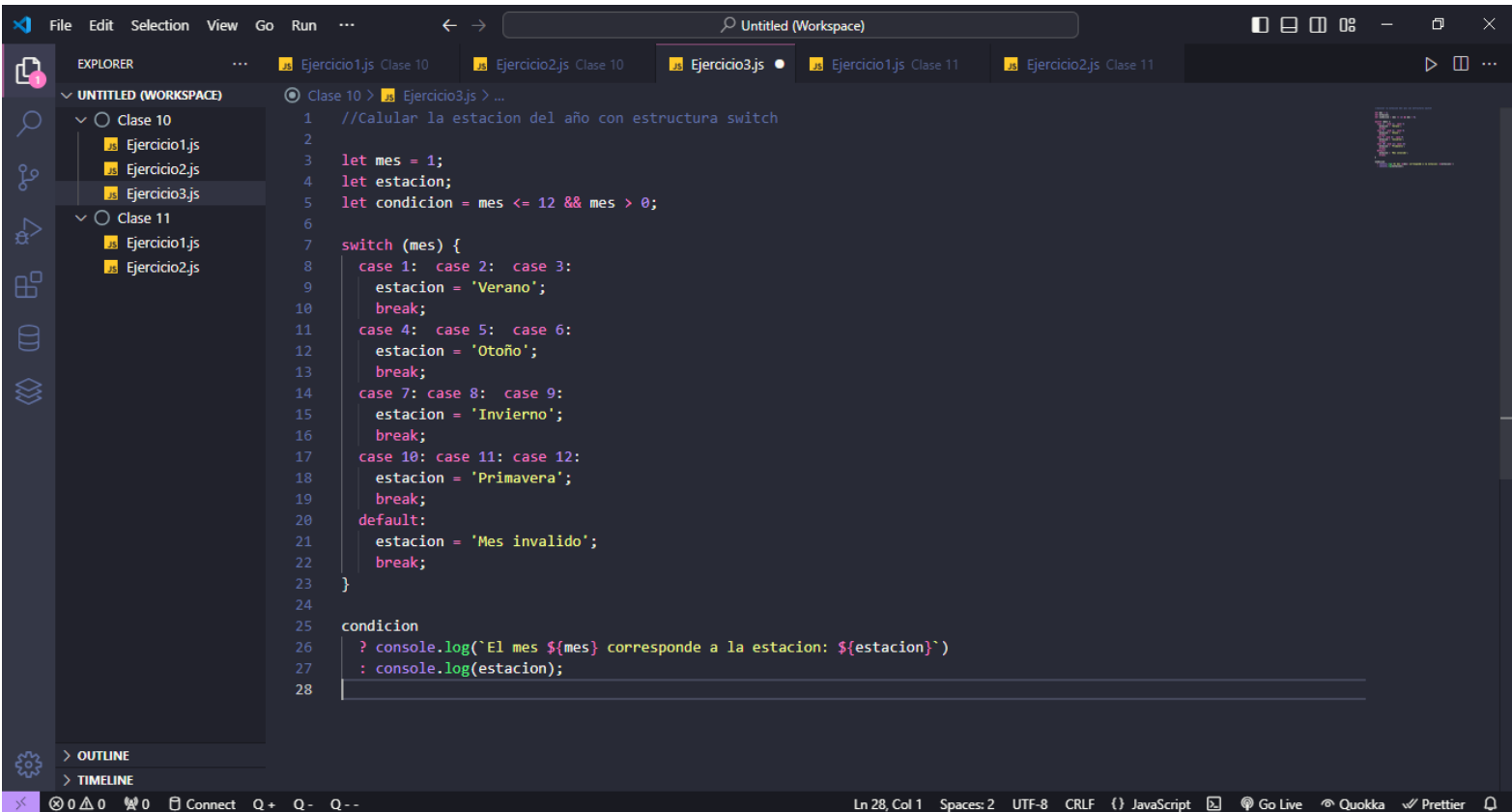
Clase 10 - Ejercicio 2



The screenshot shows the VS Code editor with a workspace named 'Untitled (Workspace)'. The Explorer panel on the left shows a project structure with 'Clase 10' and 'Clase 11' folders, each containing 'Ejercicio1.js' and 'Ejercicio2.js'. The main editor area displays the code for 'Ejercicio2.js' in 'Clase 10'. The code is as follows:

```
1 let hora = 3;
2 let saludo;
3
4 if (hora > 0 && hora < 8) {
5   saludo = 'Que descases!';
6 } else if (hora >= 8 && hora < 12) {
7   saludo = 'Buenos Días!! Que tengas un excelente dia!';
8 } else if (hora >= 12 && hora < 19) {
9   saludo = 'Buenos Tardes!! Queda lo ultimo del dia';
10 } else if (hora >= 19 && hora < 24) {
11   saludo = 'Buenos Noches!! Ya es hora de ir a descasar, no lo crees?';
12 } else {
13   saludo = 'No es una hora valida';
14 }
15
16 hora < 24 ? console.log(saludo) : console.log('La hora ${hora}, ${saludo}');
17
```

Clase 10 - Ejercicio 3



```
1 //Calular la estacion del año con estructura switch
2
3 let mes = 1;
4 let estacion;
5 let condicion = mes <= 12 && mes > 0;
6
7 switch (mes) {
8   case 1: case 2: case 3:
9     estacion = 'Verano';
10    break;
11   case 4: case 5: case 6:
12     estacion = 'Otoño';
13    break;
14   case 7: case 8: case 9:
15     estacion = 'Invierno';
16    break;
17   case 10: case 11: case 12:
18     estacion = 'Primavera';
19    break;
20   default:
21     estacion = 'Mes invalido';
22    break;
23 }
24
25 condicion
26 ? console.log(`El mes ${mes} corresponde a la estacion: ${estacion}`)
27 : console.log(estacion);
28
```

Clase 11 - Ejercicio 1

The image displays two screenshots of the Visual Studio Code editor, showing the development of a JavaScript file named `Ejercicio1.js`.

Top Screenshot: The file `Ejercicio1.js` is open, showing a switch statement that logs the day of the week based on a variable `dia`. The code is as follows:

```
1 // DRY => Dont Repeat yourself
2
3 let dia = 'Sabado';
4
5 switch (dia) {
6   case 'Lunes':
7     console.log('Hoy es ' + dia);
8     break;
9   case 'Martes':
10    console.log('Hoy es ' + dia);
11    break;
12   case 'Miercoles':
13    console.log('Hoy es ' + dia);
14    break;
15   case 'Jueves':
16    console.log('Hoy es ' + dia);
17    break;
18   case 'Viernes':
19    console.log('Hoy es ' + dia);
20    break;
21   case 'Sabado':
22    console.log('Hoy es ' + dia);
23    break;
24   case 'Domingo':
25    console.log('Hoy es ' + dia);
26    break;
27   default:
28    console.log('Dia no Valido!');
29    break;
30 }
```

Bottom Screenshot: The same file `Ejercicio1.js` is shown, but with a different implementation. It uses an array `dias` to store the days of the week and a function `getDias` to log the day of the week based on a number `num`. The code is as follows:

```
30 }
31 let dias = [
32   'Lunes',
33   'Martes',
34   'Miercoles',
35   'Jueves',
36   'Viernes',
37   'Sabado',
38   'Domingo',
39 ];
40 function getDias(num) {
41   if (num < 1 || num > 7 || num == undefined) {
42     throw new Error('Numero No Valido o Fuera de Rango');
43   } else {
44     console.log('Hoy es ' + dias[num - 1]);
45   }
46 }
47
48 getDias();
49
```

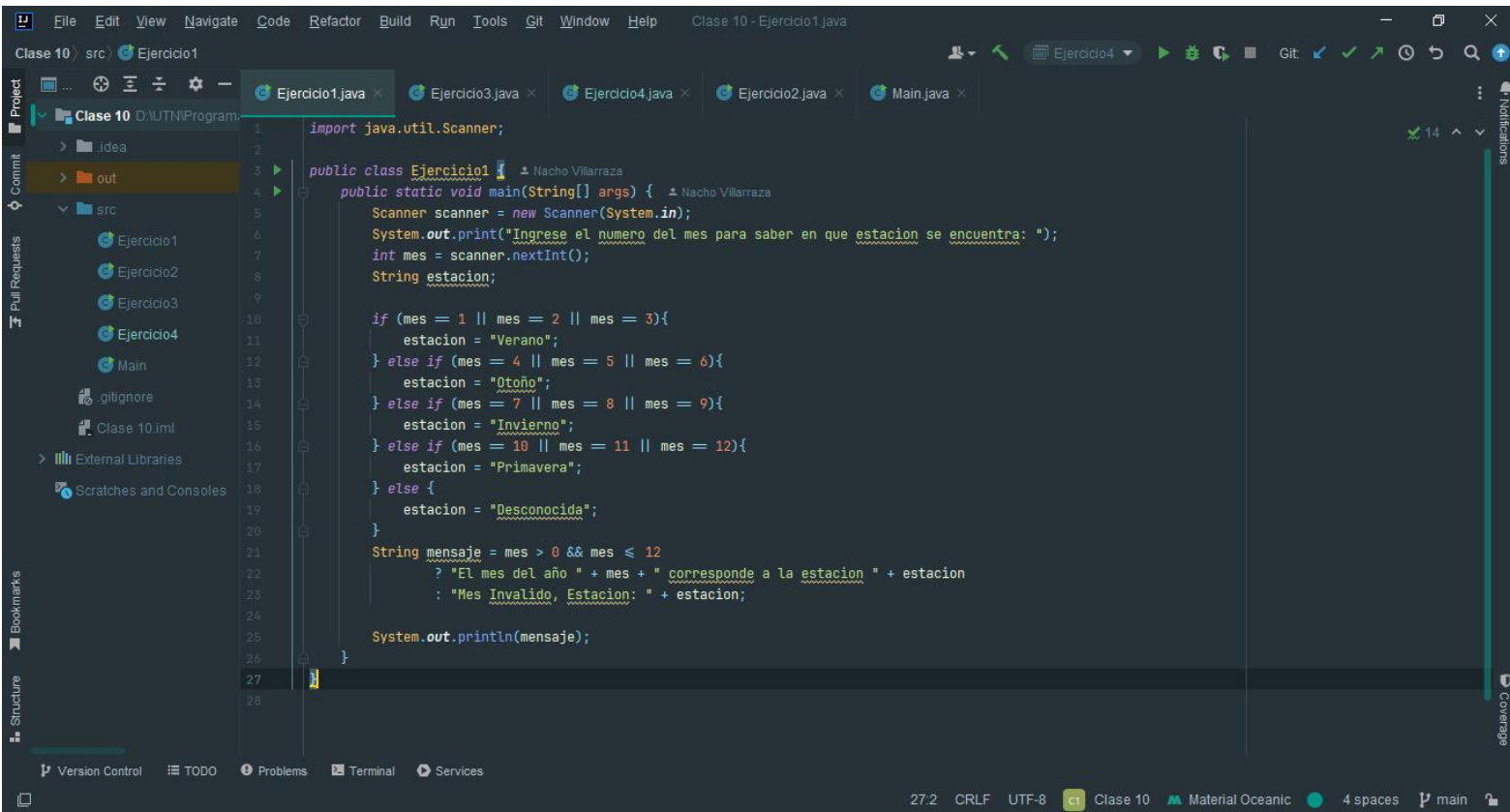

Clase 11 - Ejercicio 2

The image shows two screenshots of a VS Code editor. The top screenshot shows the first part of a JavaScript file, 'Ejercicio2.js', where a switch statement is used to log the month name based on a variable 'mes'. The switch statement has cases for 'Enero' through 'Octubre'. The bottom screenshot shows the continuation of the same file, starting from line 33. It includes a 'default' case for 'Diciembre', a list of all months in an array 'meses', a 'getMeses' function that validates the input number and logs the corresponding month, and a final call to 'getMeses(1)'.

```
1 // Ejercicio Meses del año con Switch y con Funcion Mejorada
2 let mes = 'Enero';
3 switch (mes) {
4   case 'Enero':
5     console.log('Estamos en el mes de ' + mes);
6     break;
7   case 'Febrero':
8     console.log('Estamos en el mes de ' + mes);
9     break;
10  case 'Marzo':
11    console.log('Estamos en el mes de ' + mes);
12    break;
13  case 'Abril':
14    console.log('Estamos en el mes de ' + mes);
15    break;
16  case 'Mayo':
17    console.log('Estamos en el mes de ' + mes);
18    break;
19  case 'Junio':
20    console.log('Estamos en el mes de ' + mes);
21    break;
22  case 'Julio':
23    console.log('Estamos en el mes de ' + mes);
24    break;
25  case 'Agosto':
26    console.log('Estamos en el mes de ' + mes);
27    break;
28  case 'Septiembre':
29    console.log('Estamos en el mes de ' + mes);
30    break;
31  case 'Octubre':
32    console.log('Estamos en el mes de ' + mes);
33    break;
34  case 'Noviembre':
35    console.log('Estamos en el mes de ' + mes);
36    break;
37  case 'Diciembre':
38    console.log('Estamos en el mes de ' + mes);
39    break;
40  default:
41    console.log('Mes no Valido!');
42    break;
43 }
44 let meses = ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio', 'Julio', 'Agosto', 'Septiembre', 'Octubre', 'Noviembre', 'Diciembre'];
45
46 function getMeses(num) {
47   if (num < 1 || num > 12 || num == undefined) {
48     throw new Error('Numero No Valido o Fuera de Rango');
49   } else {
50     console.log('Estamos en el mes de ' + meses[num - 1]);
51   }
52 }
53
54 getMeses(1);
55
```

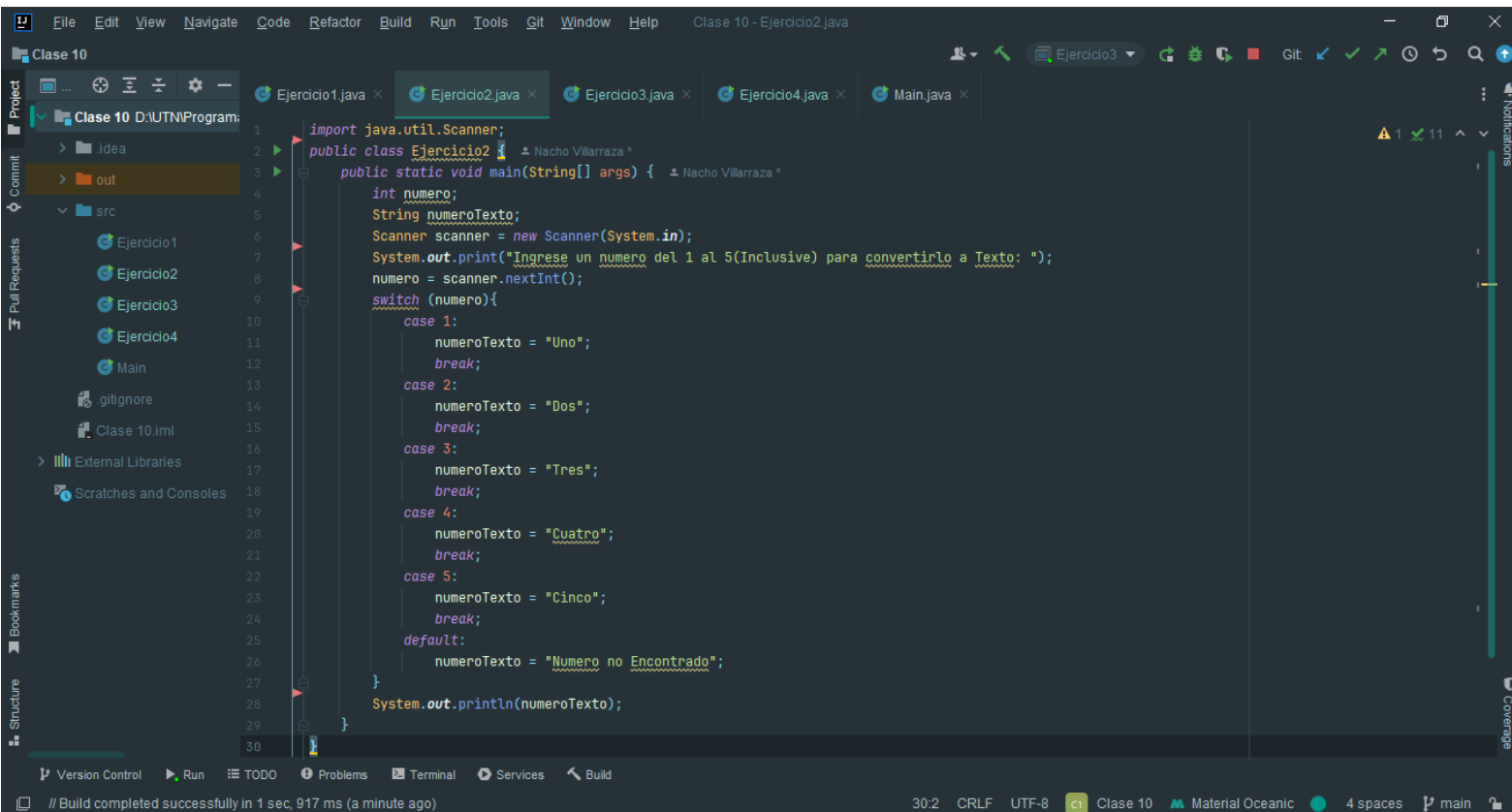
Programación I - Java

Clase 10 - Ejercicio 1



```
1 import java.util.Scanner;
2
3 public class Ejercicio1 {
4     public static void main(String[] args) {
5         Scanner scanner = new Scanner(System.in);
6         System.out.print("Ingrese el número del mes para saber en que estación se encuentra: ");
7         int mes = scanner.nextInt();
8         String estacion;
9
10        if (mes == 1 || mes == 2 || mes == 3){
11            estacion = "Verano";
12        } else if (mes == 4 || mes == 5 || mes == 6){
13            estacion = "Otoño";
14        } else if (mes == 7 || mes == 8 || mes == 9){
15            estacion = "Invierno";
16        } else if (mes == 10 || mes == 11 || mes == 12){
17            estacion = "Primavera";
18        } else {
19            estacion = "Desconocida";
20        }
21
22        String mensaje = mes > 0 && mes <= 12
23            ? "El mes del año " + mes + " corresponde a la estación " + estacion
24            : "Mes Inválido, Estación: " + estacion;
25
26        System.out.println(mensaje);
27    }
28 }
```

Clase 10 - Ejercicio 2



```
1 import java.util.Scanner;
2
3 public class Ejercicio2 {
4     public static void main(String[] args) {
5         int numero;
6         String numeroTexto;
7         Scanner scanner = new Scanner(System.in);
8         System.out.print("Ingrese un número del 1 al 5(Inclusive) para convertirlo a Texto: ");
9         numero = scanner.nextInt();
10        switch (numero){
11            case 1:
12                numeroTexto = "Uno";
13                break;
14            case 2:
15                numeroTexto = "Dos";
16                break;
17            case 3:
18                numeroTexto = "Tres";
19                break;
20            case 4:
21                numeroTexto = "Cuatro";
22                break;
23            case 5:
24                numeroTexto = "Cinco";
25                break;
26            default:
27                numeroTexto = "Número no Encontrado";
28        }
29        System.out.println(numeroTexto);
30    }
31 }
```

// Build completed successfully in 1 sec, 917 ms (a minute ago)

Clase 10 - Ejercicio 3

The screenshot shows an IDE with the following components:

- Project View (Left):** Shows a project named "Clase 10" with a source folder "src" containing files "Ejercicio1", "Ejercicio2", "Ejercicio3", "Ejercicio4", and "Main".
- Editor View (Center):** Displays the code for "Ejercicio3.java". The code is as follows:

```
1 import java.util.Scanner;
2 public class Ejercicio3 {
3     public static void main(String[] args) {
4         Scanner scanner = new Scanner(System.in);
5         System.out.print("Ingrese el numero del mes para saber en que estacion se encuentra: ");
6         int mes = scanner.nextInt();
7         String estacion;
8         switch( mes ) {
9             case 1: case 2: case 3:
10                 estacion = "Verano";
11                 break;
12             case 4: case 5: case 6:
13                 estacion = "Otoño";
14                 break;
15             case 7: case 8: case 9:
16                 estacion = "Invierno";
17                 break;
18             case 10: case 11: case 12:
19                 estacion = "Primavera";
20             default:
21                 estacion = "Desconocida";
22                 break;
23         }
24         String mensaje = mes > 0 && mes <= 12
25             ? "El mes del año " + mes + " corresponde a la estacion " + estacion
26             : "Mes Invalido, Estacion: " + estacion;
27
28         System.out.println(mensaje);
29     }
30 }
```
- Run View (Bottom):** Shows the status "Build completed successfully in 1 sec, 917 ms (a minute ago)".
- Toolbar (Top):** Includes icons for File, Edit, View, Navigate, Code, Refactor, Build, Run, Tools, Git, Window, and Help.
- Toolbar (Bottom):** Includes icons for Version Control, Run, TODO, Problems, Terminal, Services, and Build.

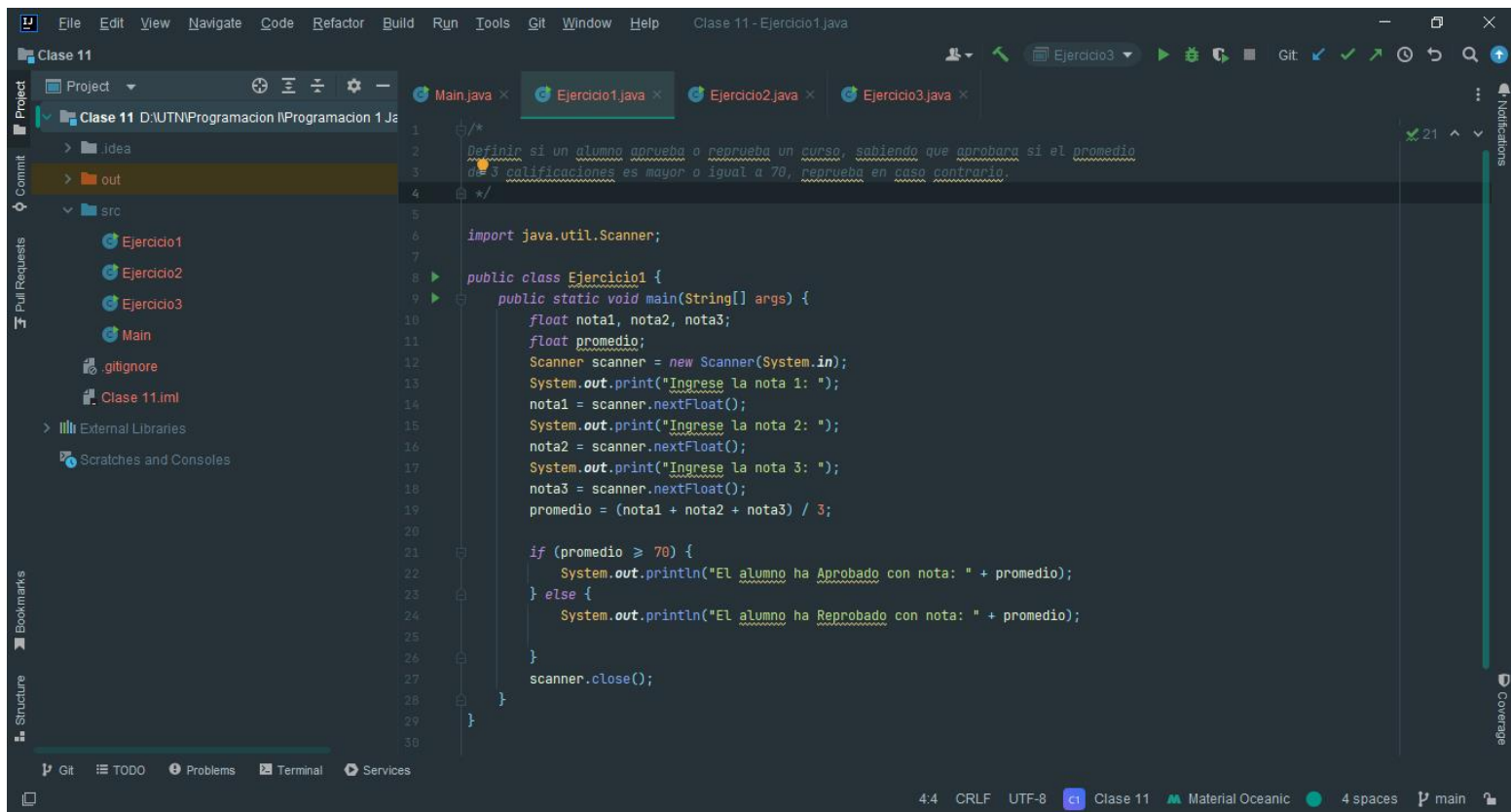
Clase 10 - Ejercicio 4

The screenshot shows an IDE with a project named "Clase 10". The project structure includes a "src" folder with files "Ejercicio1", "Ejercicio2", "Ejercicio3", "Ejercicio4", and "Main". The "Ejercicio4.java" file is open, showing the following code:

```
1 import java.util.Scanner;
2 public class Ejercicio4 {
3     public static void main(String[] args) {
4         Scanner scanner = new Scanner(System.in);
5         System.out.print("Ingrese la calificación obtenida (del 1 al 10): ");
6         int nota = scanner.nextInt();
7         String calificacion;
8         switch (nota){
9             case 10: case 9:
10                calificacion = "A";
11                break;
12             case 8:
13                calificacion = "B";
14                break;
15             case 7:
16                calificacion = "C";
17                break;
18             case 6:
19                calificacion = "D";
20                break;
21             case 5: case 4: case 3: case 2: case 1:
22                calificacion = "F";
23                break;
24             default:
25                calificacion = "Fuera de Rango";
26                break;
27         }
28         String mensaje = nota > 0 && nota <= 10
29             ? "Tu nota " + nota + " corresponde a la calificación: " + calificacion
30             : "Nota " + calificacion;
31         System.out.println(mensaje);
32     }
33 }
34
35
```

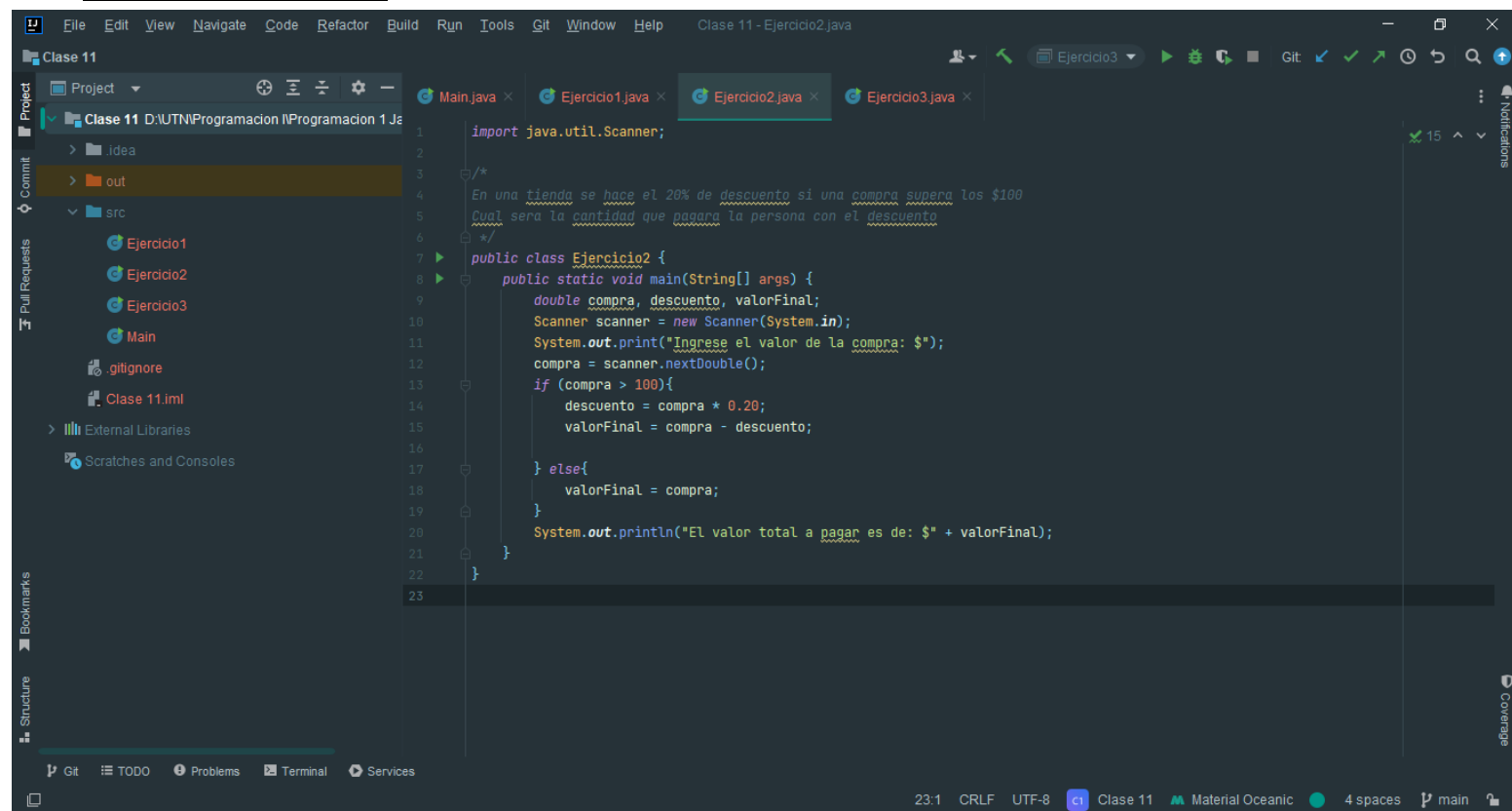
The bottom status bar indicates: "Build completed successfully in 1 sec, 917 ms (4 minutes ago)". The bottom right corner shows settings: "28:1 CRLF UTF-8 Clase 10 Material Oceanic 4 spaces main".

Clase 11 - Ejercicio 1



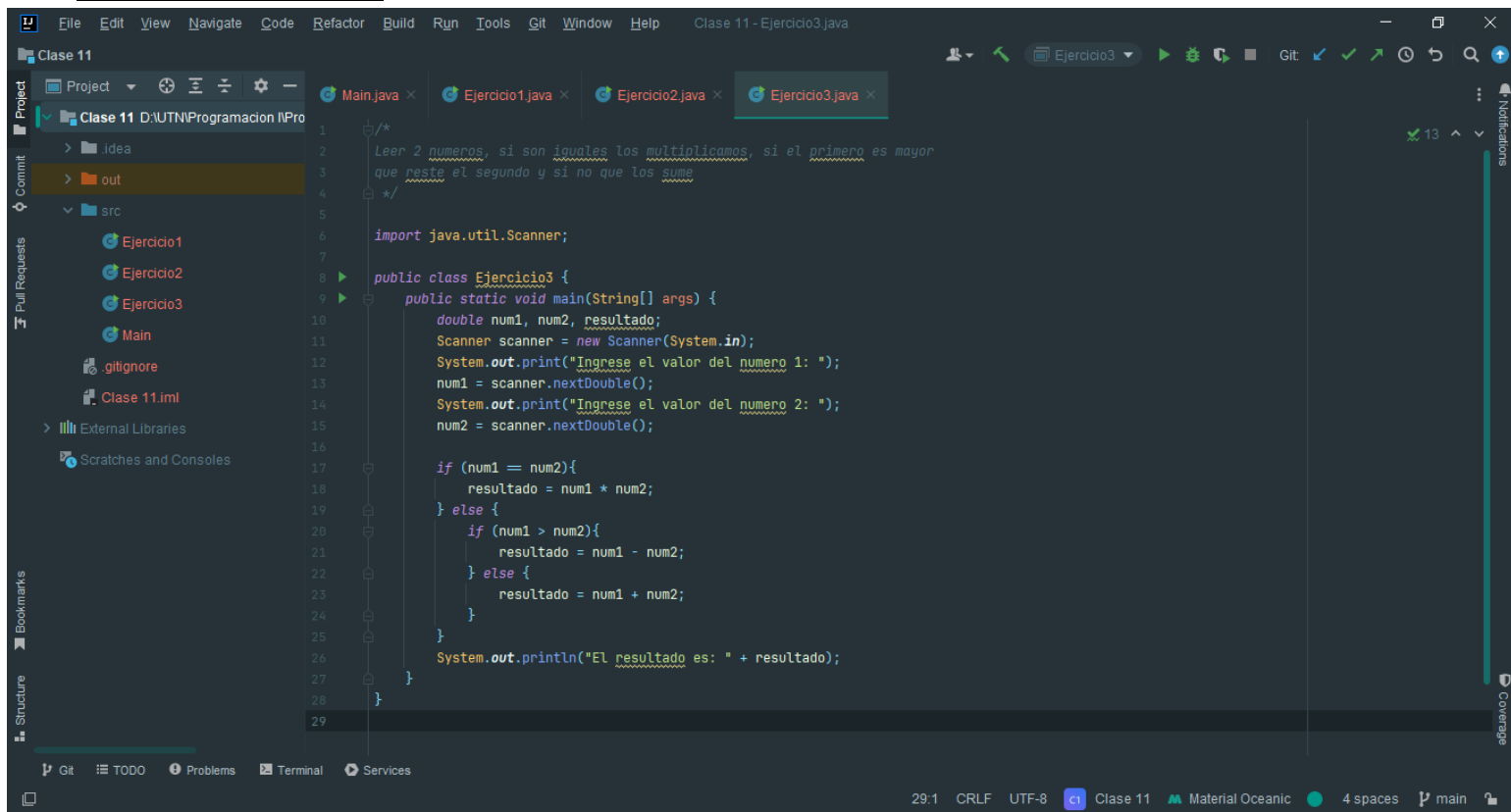
```
1  /*
2  Definir si un alumno aprueba o reprueba un curso, sabiendo que aprobará si el promedio
3  de 3 calificaciones es mayor o igual a 70, reprueba en caso contrario.
4  */
5
6  import java.util.Scanner;
7
8  public class Ejercicio1 {
9      public static void main(String[] args) {
10         float nota1, nota2, nota3;
11         float promedio;
12         Scanner scanner = new Scanner(System.in);
13         System.out.print("Ingrese la nota 1: ");
14         nota1 = scanner.nextFloat();
15         System.out.print("Ingrese la nota 2: ");
16         nota2 = scanner.nextFloat();
17         System.out.print("Ingrese la nota 3: ");
18         nota3 = scanner.nextFloat();
19         promedio = (nota1 + nota2 + nota3) / 3;
20
21         if (promedio >= 70) {
22             System.out.println("El alumno ha Aprobado con nota: " + promedio);
23         } else {
24             System.out.println("El alumno ha Reprobado con nota: " + promedio);
25         }
26         scanner.close();
27     }
28 }
29
30
```

Clase 11 - Ejercicio 2



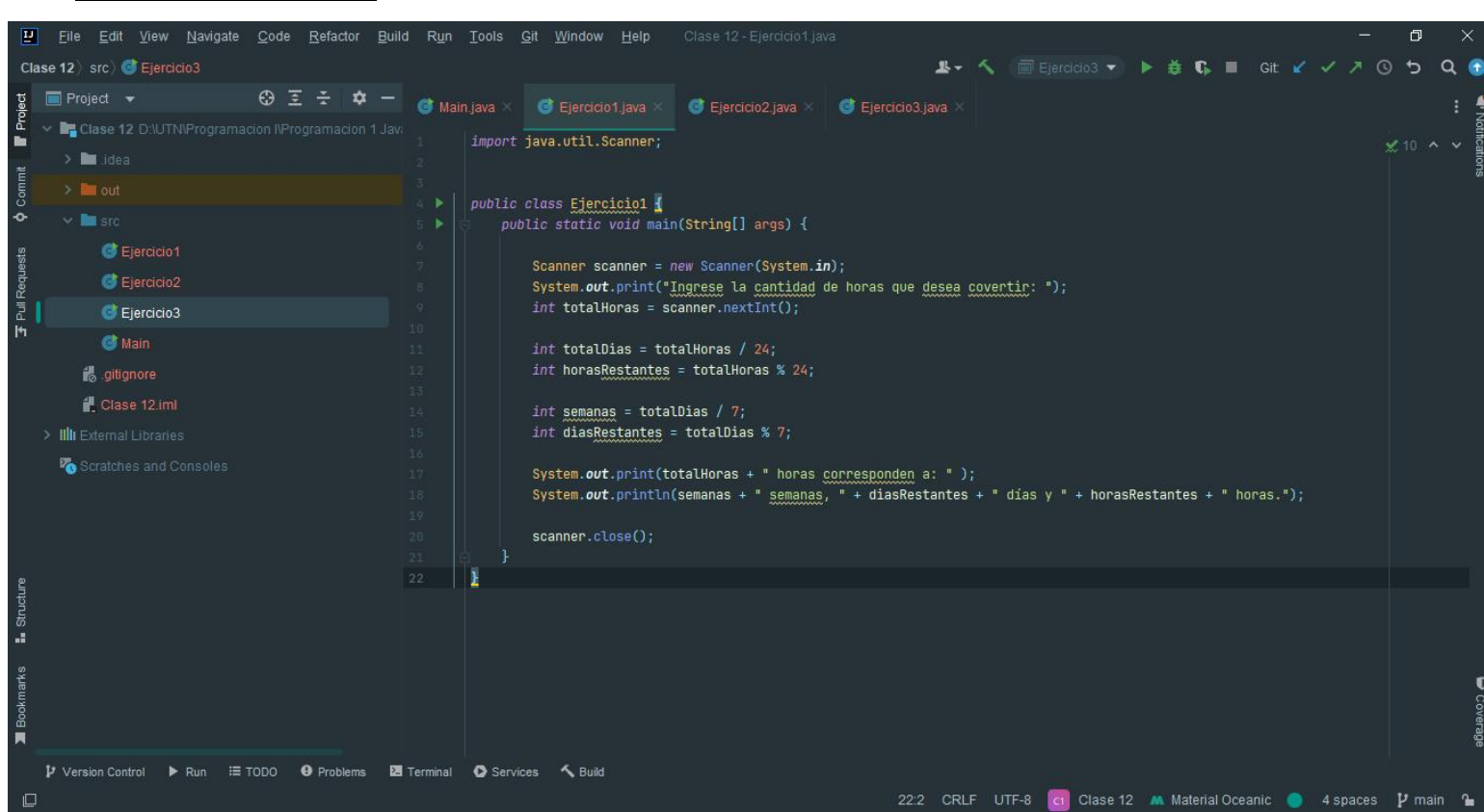
```
1  import java.util.Scanner;
2
3  /*
4  En una tienda se hace el 20% de descuento si una compra supera los $100
5  Cual sera la cantidad que pagara la persona con el descuento
6  */
7
8  public class Ejercicio2 {
9      public static void main(String[] args) {
10         double compra, descuento, valorFinal;
11         Scanner scanner = new Scanner(System.in);
12         System.out.print("Ingrese el valor de la compra: $");
13         compra = scanner.nextDouble();
14         if (compra > 100) {
15             descuento = compra * 0.20;
16             valorFinal = compra - descuento;
17         } else {
18             valorFinal = compra;
19         }
20         System.out.println("El valor total a pagar es de: $" + valorFinal);
21     }
22 }
23
```

Clase 11 - Ejercicio 3



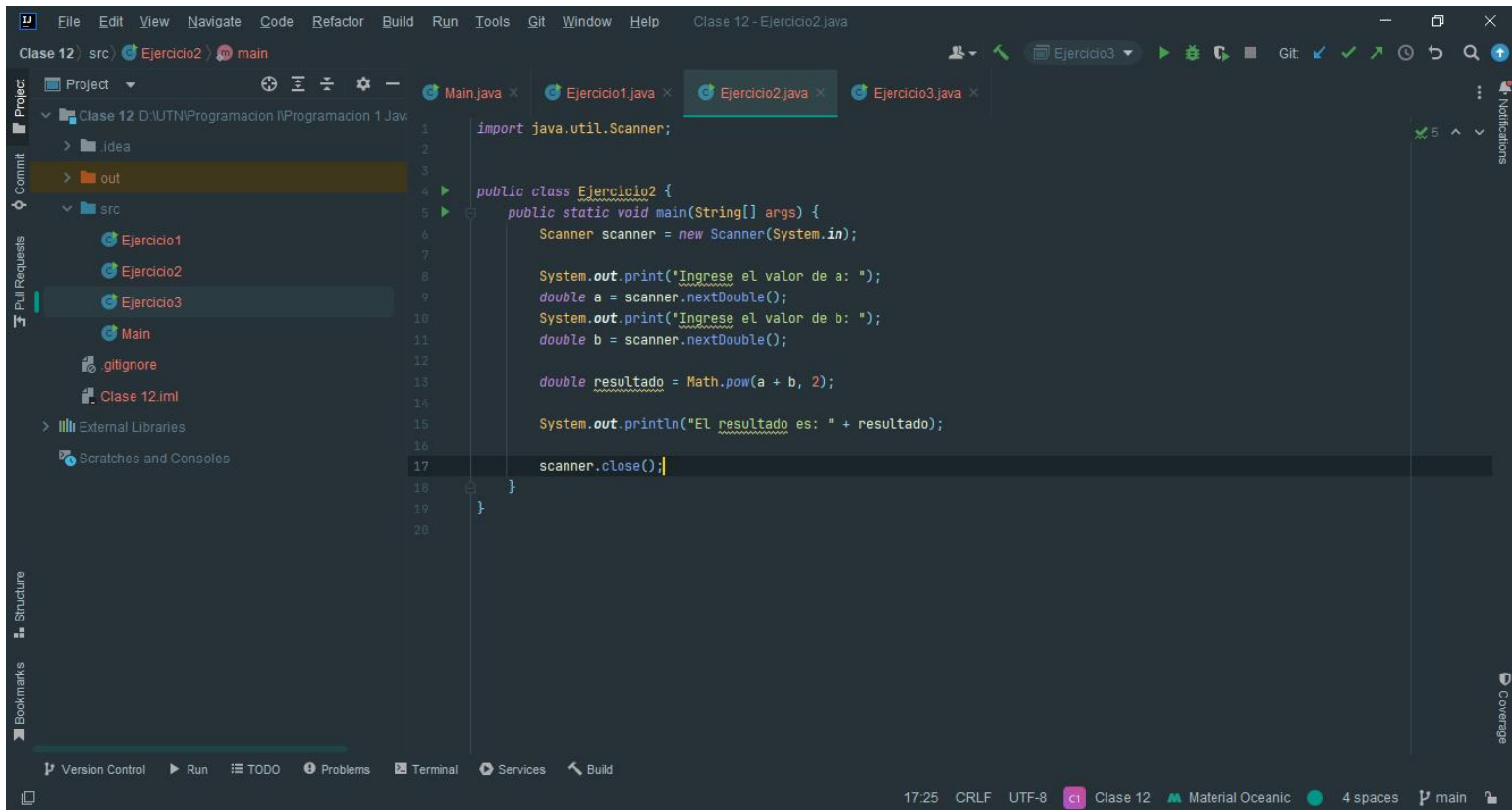
```
1  /*
2  Leer 2 números, si son iguales los multiplicamos, si el primero es mayor
3  que reste el segundo y si no que los sume
4  */
5
6  import java.util.Scanner;
7
8  public class Ejercicio3 {
9      public static void main(String[] args) {
10         double num1, num2, resultado;
11         Scanner scanner = new Scanner(System.in);
12         System.out.print("Ingrese el valor del numero 1: ");
13         num1 = scanner.nextDouble();
14         System.out.print("Ingrese el valor del numero 2: ");
15         num2 = scanner.nextDouble();
16
17         if (num1 == num2){
18             resultado = num1 * num2;
19         } else {
20             if (num1 > num2){
21                 resultado = num1 - num2;
22             } else {
23                 resultado = num1 + num2;
24             }
25         }
26         System.out.println("El resultado es: " + resultado);
27     }
28 }
29
```

Clase 12 - Ejercicio 1



```
1  import java.util.Scanner;
2
3
4  public class Ejercicio1 {
5      public static void main(String[] args) {
6
7          Scanner scanner = new Scanner(System.in);
8          System.out.print("Ingrese la cantidad de horas que desea convertir: ");
9          int totalHoras = scanner.nextInt();
10
11          int totalDias = totalHoras / 24;
12          int horasRestantes = totalHoras % 24;
13
14          int semanas = totalDias / 7;
15          int diasRestantes = totalDias % 7;
16
17          System.out.print(totalHoras + " horas corresponden a: ");
18          System.out.println(semanas + " semanas, " + diasRestantes + " días y " + horasRestantes + " horas.");
19
20          scanner.close();
21      }
22  }
```


Clase 12 - Ejercicio 2



```
import java.util.Scanner;

public class Ejercicio2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

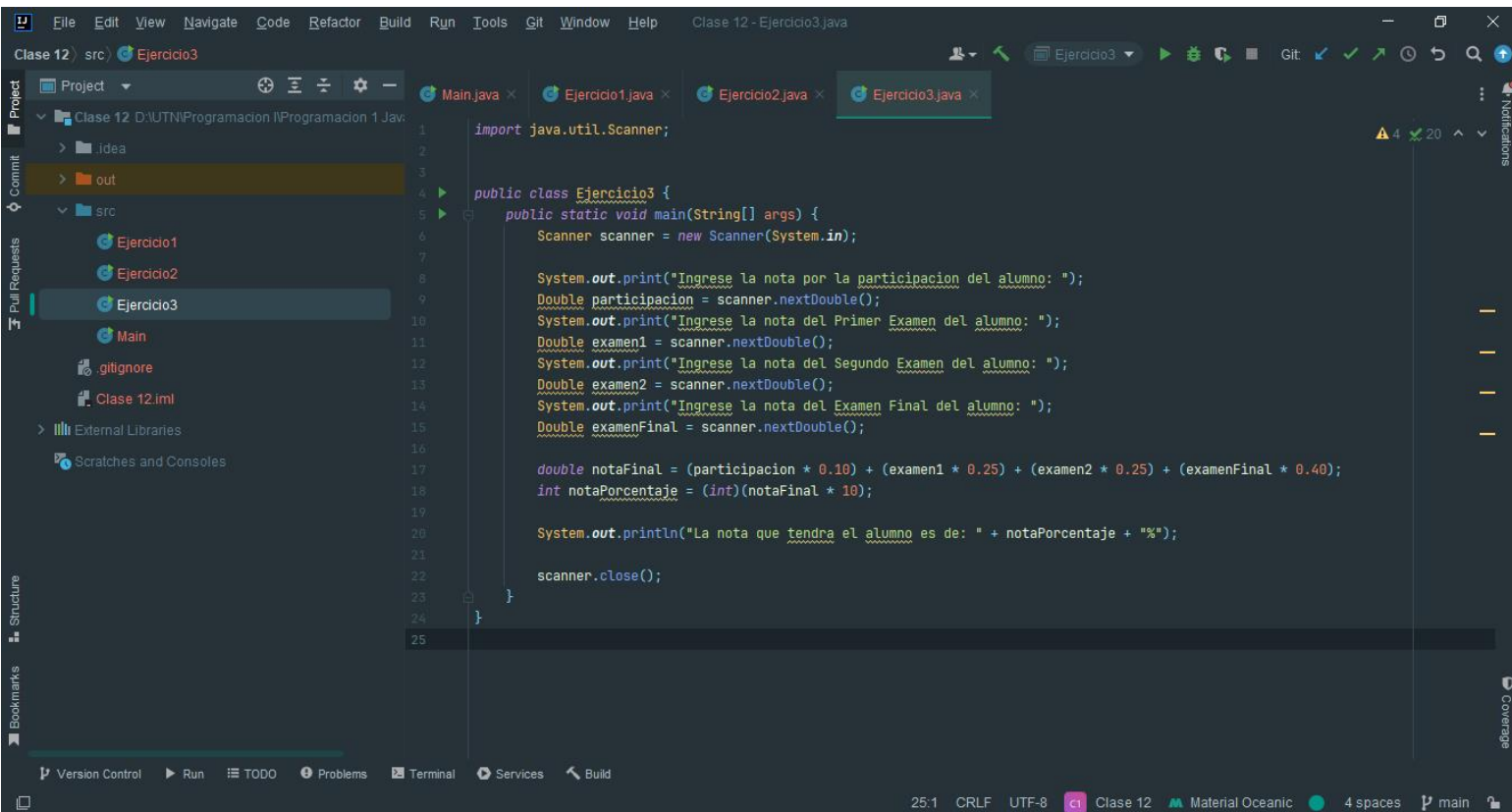
        System.out.print("Ingrese el valor de a: ");
        double a = scanner.nextDouble();
        System.out.print("Ingrese el valor de b: ");
        double b = scanner.nextDouble();

        double resultado = Math.pow(a + b, 2);

        System.out.println("El resultado es: " + resultado);

        scanner.close();
    }
}
```

Clase 12 - Ejercicio 3



```
import java.util.Scanner;

public class Ejercicio3 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Ingrese la nota por la participacion del alumno: ");
        Double participacion = scanner.nextDouble();
        System.out.print("Ingrese la nota del Primer Examen del alumno: ");
        Double examen1 = scanner.nextDouble();
        System.out.print("Ingrese la nota del Segundo Examen del alumno: ");
        Double examen2 = scanner.nextDouble();
        System.out.print("Ingrese la nota del Examen Final del alumno: ");
        Double examenFinal = scanner.nextDouble();

        double notaFinal = (participacion * 0.10) + (examen1 * 0.25) + (examen2 * 0.25) + (examenFinal * 0.40);
        int notaPorcentaje = (int)(notaFinal * 10);

        System.out.println("La nota que tendra el alumno es de: " + notaPorcentaje + "%");

        scanner.close();
    }
}
```

Programación I - Gitbash

Git log

```
MINGW64:/d:/UTN/Programacion I/Tecnicatura/class-git
Date:   Fri Jun 28 09:07:53 2024 -0300

Agregamos archivo css para darle estilo a nuestro portfolio

commit 4c6f98d3c44b85424a1f63734aa10d3bdd31f9cf
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 09:05:52 2024 -0300

Agregamos html para nuestro Portfolio

commit d2b5719dcc72c8d096f98f599ecb496c0fc45d22
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 08:58:31 2024 -0300

Agregada clase 4 al archivo README.md

commit 96269c5f40f0fa38d36eafa3d322650d12ef326f
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 08:53:11 2024 -0300

Agregando README.md

commit 3c44ba5b917e25a3cb5bae709df8df6a2da7bf2b
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Wed Apr 10 21:41:01 2024 -0300

Segundo Commit

commit a4ee389fec5b2b151a21065889bfc1965753103b
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Wed Apr 10 21:40:03 2024 -0300

Primer Commit
(END)

MINGW64:/d:/UTN/Programacion I/Tecnicatura/class-git

commit c8192802302d14f97f07a36eb24e1c29da375fb3
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 09:49:38 2024 -0300

Agregando la clase 12 al archivo README.md

commit 7bd554d394cf59cb05cc8b213b65ce189cda223a
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 09:47:41 2024 -0300

Agregando la clase 11 al archivo README.md

commit f1a13f0a501769b3a71ed183fdeda35eb4d292ba
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 09:42:32 2024 -0300

Agregando la clase 10 al archivo README.md

commit 84f33781bf28daf05d17acff1551cfb15fc89c3
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 09:31:02 2024 -0300

Agregando la clase 9 al archivo README.md

commit ea2c4cc2a718bee24f6fcfd64ac78f4aa464ac32 (segunda)
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 09:17:02 2024 -0300

Agregando la clase 8 al archivo README.MD

commit 0c3e6fb0835200480cd53b59f1bc44c2f59d8939
commit 0c3e6fb0835200480cd53b59f1bc44c2f59d8939
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date:   Fri Jun 28 09:07:53 2024 -0300
```



```
MINGW64:/d/UTN/Programacion I/Tecnicaura/class-git
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 10:00:09 2024 -0300

Agregando la clase 12 al archivo README.md y Modificando el portafolio Personal/Grupal

commit c8192802302d14f97f07a36eb24e1c29da375fb3
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 09:49:38 2024 -0300
:...skipping...
commit e492622c65c9267a8b0e744f6cd30cdcc2635108 (HEAD -> main)
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 13:51:14 2024 -0300

Portafolio Personal Finalizado, con todos las secciones

commit 7429c90d61dea66d856570276bcd9206a09251b
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 13:50:42 2024 -0300

Portafolio Personal Finalizado, con todos las secciones

commit 83d79d7fbce8837b0ce375b49abaf81a4a320aa9
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 10:01:06 2024 -0300

Agregando la clase 12 al archivo README.md y Modificando el portafolio Personal/Grupal

commit e8da7f1d3ed91a5cf1577bd36ccf45f657f3b7ac
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 10:00:09 2024 -0300

Agregando la clase 12 al archivo README.md y Modificando el portafolio Personal/Grupal

commit c8192802302d14f97f07a36eb24e1c29da375fb3
```

```
MINGW64:/d/UTN/Programacion I/Tecnicaura/class-git
Nacho@DESKTOP-Q765IVQ MINGW64 /d/UTN/Programacion I/Tecnicaura/class-git (main)
$ git log
commit e492622c65c9267a8b0e744f6cd30cdcc2635108 (HEAD -> main)
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 13:51:14 2024 -0300

Portafolio Personal Finalizado, con todos las secciones

commit 7429c90d61dea66d856570276bcd9206a09251b
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 13:50:42 2024 -0300

Portafolio Personal Finalizado, con todos las secciones

commit 83d79d7fbce8837b0ce375b49abaf81a4a320aa9
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 10:01:06 2024 -0300

Agregando la clase 12 al archivo README.md y Modificando el portafolio Personal/Grupal

commit e8da7f1d3ed91a5cf1577bd36ccf45f657f3b7ac
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 10:00:09 2024 -0300

Agregando la clase 12 al archivo README.md y Modificando el portafolio Personal/Grupal

commit c8192802302d14f97f07a36eb24e1c29da375fb3
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 09:49:38 2024 -0300
:...skipping...
commit e492622c65c9267a8b0e744f6cd30cdcc2635108 (HEAD -> main)
Author: Nacho Villarraza <nachovillarraza@hotmail.com>
Date: Fri Jun 28 13:51:14 2024 -0300
```

Readme.md

Clase 12

```
File Edit Selection View Go Run ... class-git
index.html README.md X styles.css footer.css
README.md > # Clase 13 > En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local. > ##### 2FA
402
403 # Clase 12
404
405 ---
406
407 ##### Cómo funcionan las llaves públicas y privadas
408
409 Las llaves públicas y privadas, conocidas también como cifrado asimétrico de un solo camino, sirven para mandar mensajes privados entre varios
410 nodos con la lógica de que firmas tu mensaje con una llave pública vinculada con una llave privada que puede leer el mensaje.
411
412 Las llaves públicas y privadas nos ayudan a cifrar y descifrar nuestros archivos de forma que los podamos compartir sin correr el riesgo de
413 que sean interceptados por personas con malas intenciones.
414
415 Cómo funciona un mensaje cifrado con llaves públicas y privadas Ambas personas deben crear su llave pública y privada.
416
417 Ambas personas pueden compartir su llave pública a las otras partes (recuerda que esta llave es pública, no hay problema si la "interceptan").
418
419 La persona que quiere compartir un mensaje puede usar la llave pública de la otra persona para cifrar los archivos y asegurarse que solo
420 puedan ser descifrados con la llave privada de la persona con la que queremos compartir el mensaje.
421
422 El mensaje está cifrado y puede ser enviado a la otra persona sin problemas en caso de que los archivos sean interceptados.
423
424 La persona a la que enviamos el mensaje cifrado puede emplear su llave privada para descifrar el mensaje y ver los archivos.
425
426 Nota: puedes compartir tu llave pública, pero nunca tu llave privada.
427
428 # Clase 13
429 ##### Configura tus llaves SSH en local
430
431 Si usamos GitHub solo con usuario y contraseña, si un día perdemos nuestra PC, perdemos todo, nuestras contraseñas y los proyectos de nuestros
432 clientes están todos en riesgo. Esta es la forma en que muchos sitios web son jackeados, para evitar esto tenemos que agregar una capa de
433 seguridad mucho más fuerte. Es aquí donde podemos comenzar a crear el entorno con llaves públicas y privadas. Esta tiene una ventaja, no solo
434 es que nuestra seguridad será más fuerte, si no que ya no tendrás que poner nunca más tu usuario y contraseña.
```

Clase 13

```
File Edit Selection View Go Run ... class-git
index.html README.md X styles.css footer.css
README.md > # Clase 13 > En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local. > ##### 2FA
424
425 # Clase 13
426
427 ##### Configura tus llaves SSH en local
428
429 Si usamos GitHub solo con usuario y contraseña, si un día perdemos nuestra PC, perdemos todo, nuestras contraseñas y los proyectos de nuestros
430 clientes están todos en riesgo. Esta es la forma en que muchos sitios web son jackeados, para evitar esto tenemos que agregar una capa de
431 seguridad mucho más fuerte. Es aquí donde podemos comenzar a crear el entorno con llaves públicas y privadas. Esta tiene una ventaja, no solo
432 es que nuestra seguridad será más fuerte, si no que ya no tendrás que poner nunca más tu usuario y contraseña.
433
434 En nuestra maquina, debemos crear una llave privada y otra pública, una vez creada la llave pública se la enviamos a GitHub en nuestro
435 repositorio, y le decimos: para este repositorio quiero que uses esta llave pública, de mi llave privada en mi PC, todo esto lo conectamos por
436 un protocolo nuevo, en vez de conectarnos al repositorio por HTTPS, vamos a conectarnos por un protocolo que se llama SSH.
437
438 En la primera conexión GitHub se va a dar cuenta que le mandaste una llave publica que esta relacionada con tu llave privada y nos va a
439 enviar cifrada con nuestra llave pública su propia llave pública de GitHub, porque GitHub también tiene una llave privada, todo esto sucederá
440 automáticamente, a la llave privada que nosotros tenemos, se le puede hacer una contraseña encima, para añadir más seguridad para hacerla mas
441 fuerte y más poderosa.
442
443 Las llaves SSH no son por repositorio o por proyecto, si no que es por persona, ahora vamos a crear unas llaves exclusivamente para nosotros.
444
445 ##### En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local.
446
447 ##### Comandos:
448
449 abrir git bash #Esto en window
450
451 abrir terminal #En ubuntu, y nos quedamos sin entrar a ningun proyecto o carpeta.
452
453 `git config -l` #Recordamos nuestra configuración en Git, podemos hacer esto estando en la ruta de cualquier sitio en nuestro PC
454
455
456
```

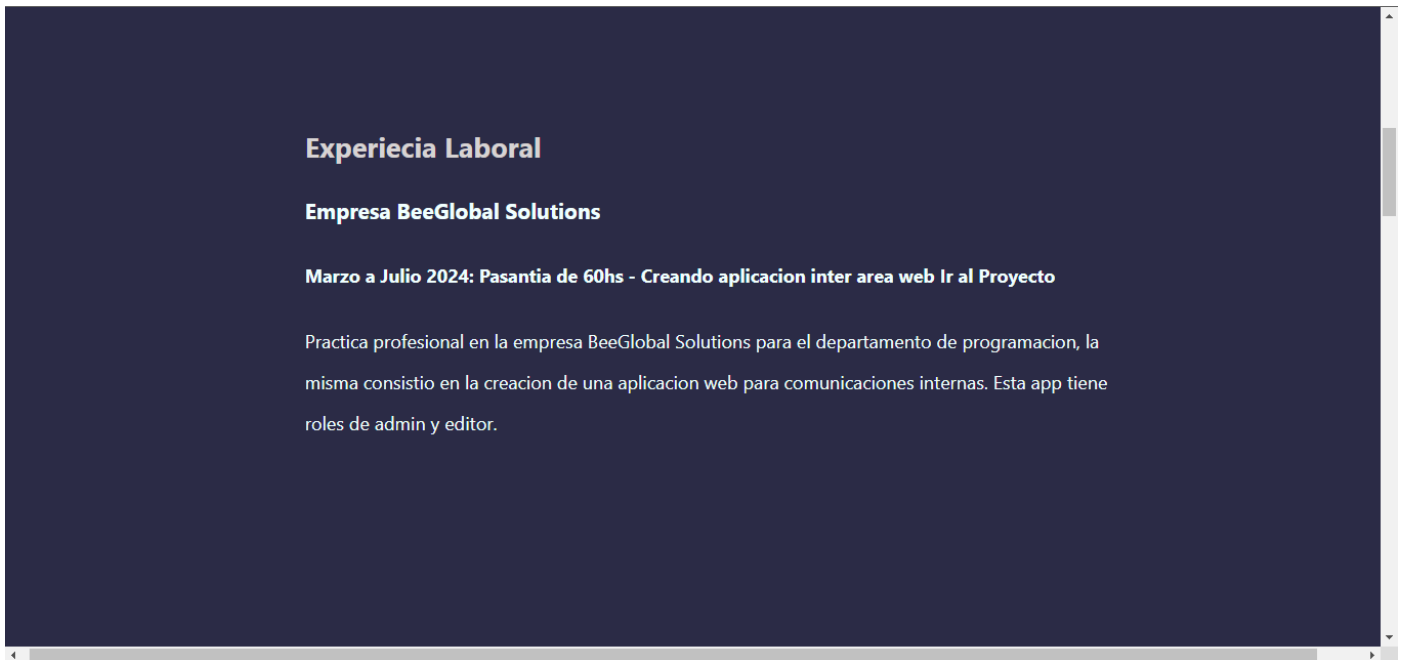
```
File Edit Selection View Go Run ... class-git
index.html README.md X styles.css footer.css
README.md > # Clase 13 > En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local. > ##### 2FA
425 # Clase 13
427 ##### Configura tus llaves SSH en local
436
437 ##### En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local.
438
439 ##### Comandos:
440
441 abrir git bash #Esto en window
442
443 abrir terminal #En ubuntu, y nos quedamos sin entrar a ningun proyecto o carpeta.
444
445 `git config -l` #Recordamos nuestra configuración en Git, podemos hacer esto estando en la ruta de cualquier sitio en nuestro PC
446
447 `git config --global user.email "alumnos@mail.com"` #Actualizamos el correo que usamos en GitHub.
448
449 `ssh-keygen -t rsa -b 4096 -C "alumnos@mail.com"` #Dira que esta generando la llave pública y privada, también nos pregunta donde vamos a
guardar la llave, presionamos enter, nos va a pedir otra contraseña
450
451 `eval $(ssh-agent -s)` #Encendemos el servidor de llaves SSH, ya esta corriendo
452
453 `~` #Se utiliza virgulilla para poner la ruta, es una variable que tiene el nombre de nuestra carpeta home, esto para el siguiente comando
454
455 `ssh-add ~/.ssh/id_ga35745` #Añadimos la llave, no la .pub ponemos la llave privada, recordar que es una ruta, se debe poner el nombre de la
carpeta que contiene la clave privada.
456
457 ##### Cómo generar tus llaves SSH:
458
459 a. Generar tus llaves SSH\**
460
461 Recuerda que es muy buena idea proteger tu llave privada con una contraseña.
462
463 `ssh-keygen -t rsa -b 4096 -C "tu@email.com"
Ln 518, Col 1 Spaces: 2 UTF-8 CRLF Markdown No Environment Go Live Quokka Prettier
```

```
File Edit Selection View Go Run ... class-git
index.html README.md X styles.css footer.css
README.md > # Clase 13 > En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local. > ##### 2FA
425 # Clase 13
427 ##### En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local.
457 ##### Cómo generar tus llaves SSH:
464
465 b. Terminar de configurar nuestro sistema.
466
467 En Windows y Linux:
468
469 Encender el "servidor" de llaves SSH de tu computadora:
470
471 `eval $(ssh-agent -s)`
472
473 Añadir tu llave SSH a este "servidor":
474
475 `ssh-add ruta-donde-guardaste-tu-llave-privada`
476
477 En Mac:
478
479 Encender el "servidor" de llaves SSH de tu computadora:
480
481 `eval "$(ssh-agent -s)"`
482
483 Si usas una versión de OSX superior a Mac Sierra (v10.12), debes crear o modificar un archivo "config" en la carpeta de tu usuario con el
siguiente contenido (ten cuidado con las mayúsculas): vim config
484
485 Host \*
486
487 AddKeysToAgent yes
488
489 UseKeychain yes
490
491 IdentityFile ruta-donde-guardaste-tu-llave-privada
Ln 518, Col 1 Spaces: 2 UTF-8 CRLF Markdown No Environment Go Live Quokka Prettier
```

```
File Edit Selection View Go Run ... class-git
index.html README.md X styles.css footer.css
README.md > # Clase 13 > ### En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local. > ##### 2FA
425 # Clase 13
437 ### En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local.
457 ##### Cómo generar tus llaves SSH:
492
493 Añadir tu llave SSH al "servidor" de llaves SSH de tu computadora (en caso de error puedes ejecutar este mismo comando pero sin el argumento
-K):
494
495 ssh-add -K ruta-donde-guardaste-tu-llave-privada
496
497 ##### 2FA
498
499 Por último les quiero hablar del 2FA: Segundo Factor de Autenticación. Este se puede hacer con varios dispositivos, y deberías hacerlo, ante
el robo o pérdida de un celular o ordenador, deberías tener un respaldo ante esto, este 2FA se puede hacer con diferentes generadores de
códigos de seguridad.
500
501 Para añadir un 2FA:
502
503 1. Clic en nuestro perfil, arriba y a la derecha, seleccionamos...
504
505 2. Settings
506
507 3. Password and Authentication
508
509 4. GitHub Mobile: GitHub Mobile can be used for two-factor authentication by installing the GitHub Mobile app and signing in to your account.
-> GitHub Mobile se puede utilizar para la autenticación de 2FA instalando la aplicación GitHub Mobile e iniciando sesión en su cuenta.
510
511 Esto quiere decir que también se utiliza la app de GitHub donde al iniciar sesión desde cualquier dispositivo nos muestra un número que
debemos ingresar en la app de nuestro dispositivo celular.
512
513 5. Authenticator app: Edit
514
515 Esto para agregar a través de un QR una app que genere cada 1 segundo nuevos códigos numéricos para la autenticación, yo recomiendo la
aplicación: Twilio Authy Authenticator
main 0 0 0 Connect Ln 512, Col 1 Spaces: 2 UTF-8 CRLF Markdown No Environment Go Live Quokka Prettier
```

```
File Edit Selection View Go Run ... class-git
index.html README.md X styles.css footer.css
README.md > # Clase 13 > ### En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local. > ##### 2FA
425 # Clase 13
437 ### En este ejemplo, aprenderemos cómo configurar nuestras llaves SSH en local.
497 ##### 2FA
508
509 4. GitHub Mobile: GitHub Mobile can be used for two-factor authentication by installing the GitHub Mobile app and signing in to your account.
-> GitHub Mobile se puede utilizar para la autenticación de 2FA instalando la aplicación GitHub Mobile e iniciando sesión en su cuenta.
510
511 Esto quiere decir que también se utiliza la app de GitHub donde al iniciar sesión desde cualquier dispositivo nos muestra un número que
debemos ingresar en la app de nuestro dispositivo celular.
512
513 5. Authenticator app: Edit
514
515 Esto para agregar a través de un QR una app que genere cada 1 segundo nuevos códigos numéricos para la autenticación, yo recomiendo la
aplicación: Twilio Authy Authenticator
516
517 Es recomendable iniciar sesión, osea registrarnos y guardar estos datos para que al cambiar un dispositivo sigamos teniendo acceso.
518
```

Programación I – Portfolio Personal



Mis Estudios

Marzo 2024 - Diciembre 2025

Tecnico Universitario en Programacion



Estudiante de la carrera de Tecnico Universitario en Programacion en la Universidad Tecnologica Nacional con habilidades en Resolucion de problemas de procesamiento de datos mediante la seleccion de algoritmos y tecnicas adecuadas. Desarrollo y correccion de programas en Lenguajes Superiores. Colaboracion en tareas especificas definidas por el Analista de Sistemas de Computacion, Analisis, Depuracion y transferencia de Informacion Procesada.

Proyectos



Proyecto 1

Aplicacion Movil para calcular los pesos a utilizar en el Gimnasio

[Ver más](#)[GitHub](#)[Tecnologías](#)

Proyecto 2

Aplicacion Movil mostrando lugares de interes en el mapa

[Ver más](#)[GitHub](#)[Tecnologías](#)

Proyecto 3

Aplicacion Movil del Juego Piedra-Papel-Tijera-Lizard-Spock

[Ver más](#)[GitHub](#)[Tecnologías](#)

