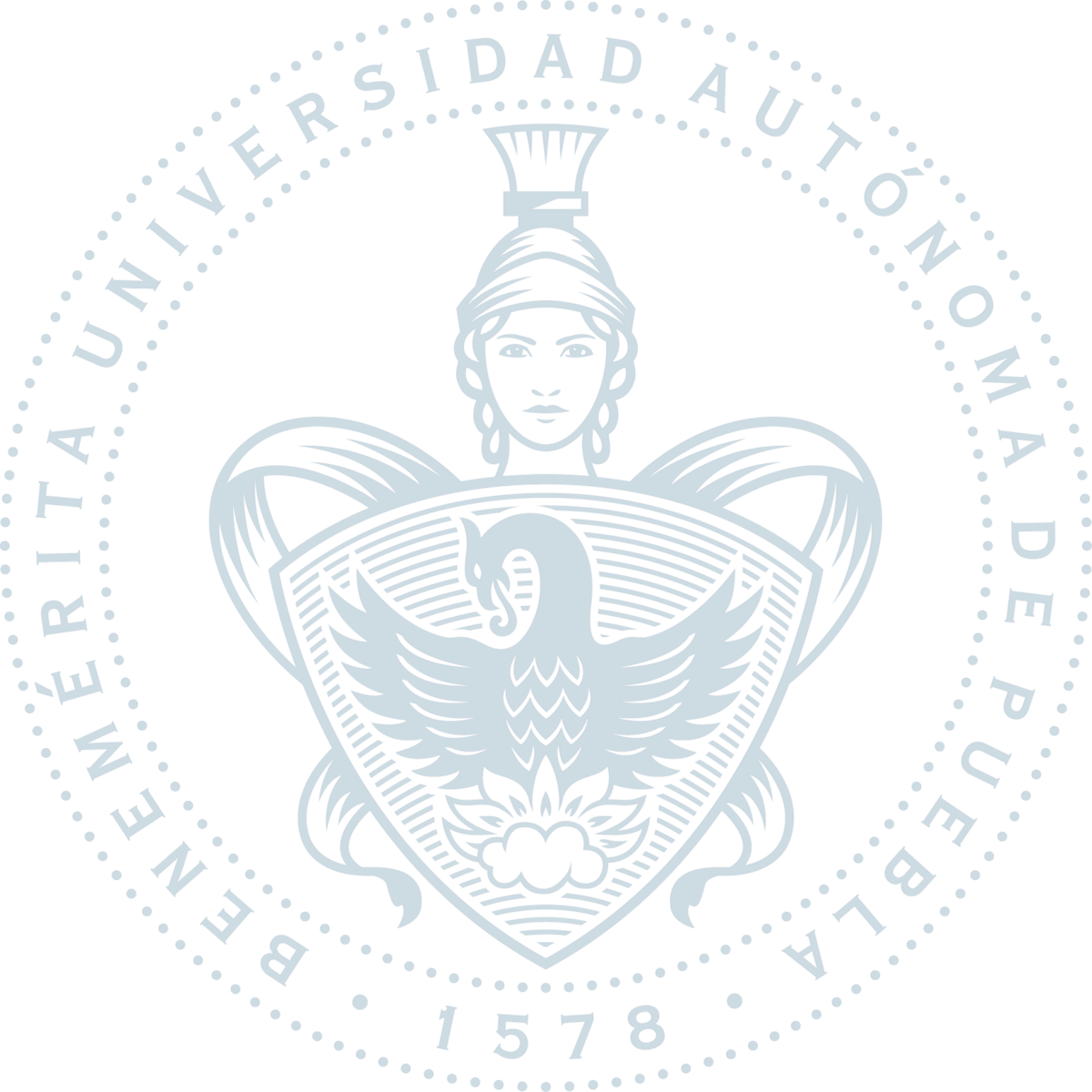
**BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA**

BUAP

** FACULTAD DE CIENCIAS DE LA COMPUTACIÓN**

**INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN**

**PROGRAMACIÓN 2**

**“PRACTICA 16B. REPORTE RECURSIVIDAD”**

**PROFESOR:**

**DRA. CARMEN CERÓN GARNICA.**

**EQUIPO:  
BRENDA LIZETH COCA GARCIA**

**MATRICULA:201917154**

**BRENDA AILED RODRIGUEZ COLIS**

**MATRICULA:201910848**

**MIGUEL CARREON VAZQUEZ**

**MATRICULA: 201915389**

**IRVYN XICALE CARRERA**

**MATRICULA: 201963582**

**INTRODUCCIÓN**En java podemos crear y leer archivos ya sean archivos binarios, archivos de txt o archivos de objetos.

**PROGRAMA EJERCICIO 1**

**DIAGRAMA UML**

**CLASE EJER1**

// Copia el contenido de un archive en otro.

import java.io.\*;

class CopyFile {

    public static void main(String args[]) throws IOException {

        int i;

        FileInputStream fin;

        FileOutputStream fout;

        try {

            fin = new FileInputStream("ejem1.txt");

            fout = new FileOutputStream("ejem2.txt");

        } catch (FileNotFoundException e) {

            System.out.println("Archivo no encontrado.....");

            return;

        } catch (IOException e) {

            System.out.println("Error al abrir el archivo de salida");

            return;

        } catch (ArrayIndexOutOfBoundsException e) {

            System.out.println("Formato: CopyFile falta alguno Origen Destino");

            return;

        }

        // copya el archivo

        try {

            do {

                i = fin.read();

                if (i != -1)

                    fout.write(i);

            } while (i != -1);

        } catch (IOException e) {

            System.out.println("Error...");

        }

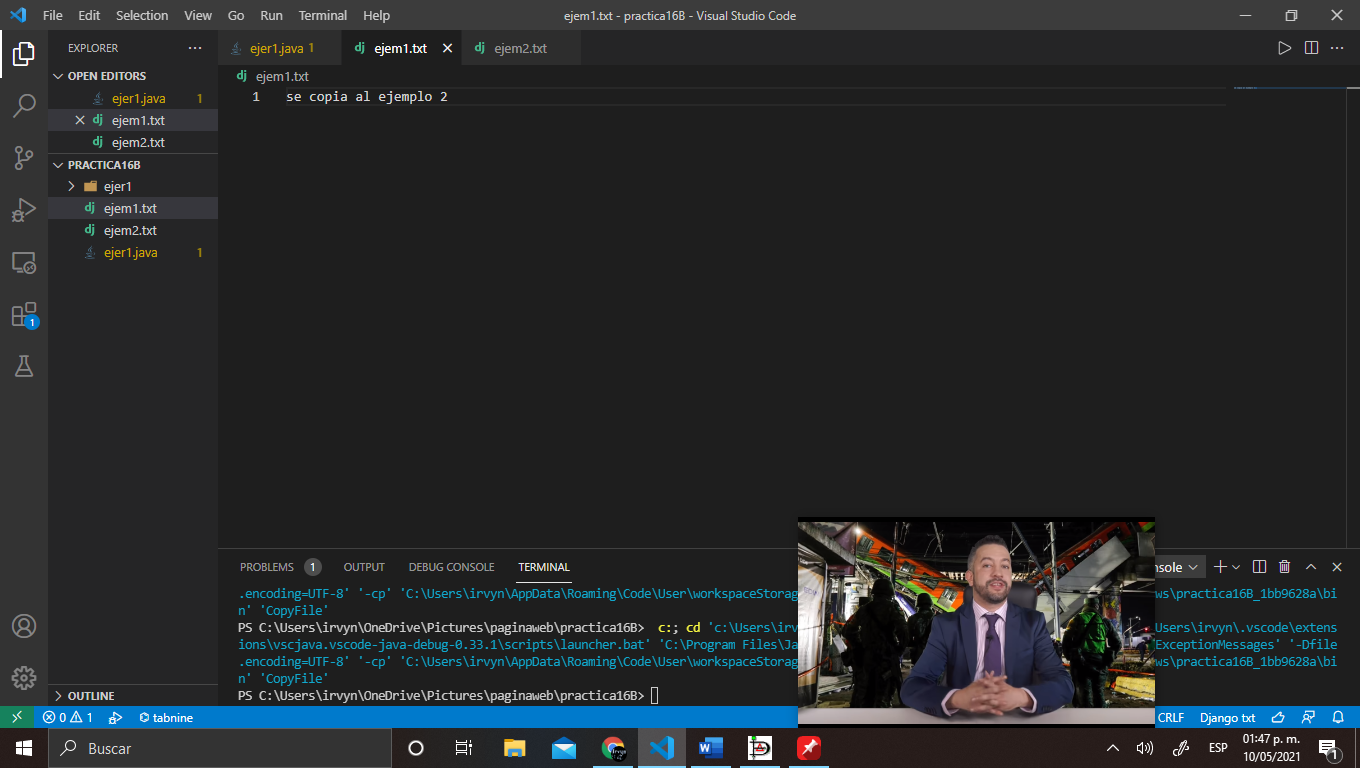
        fin.close();

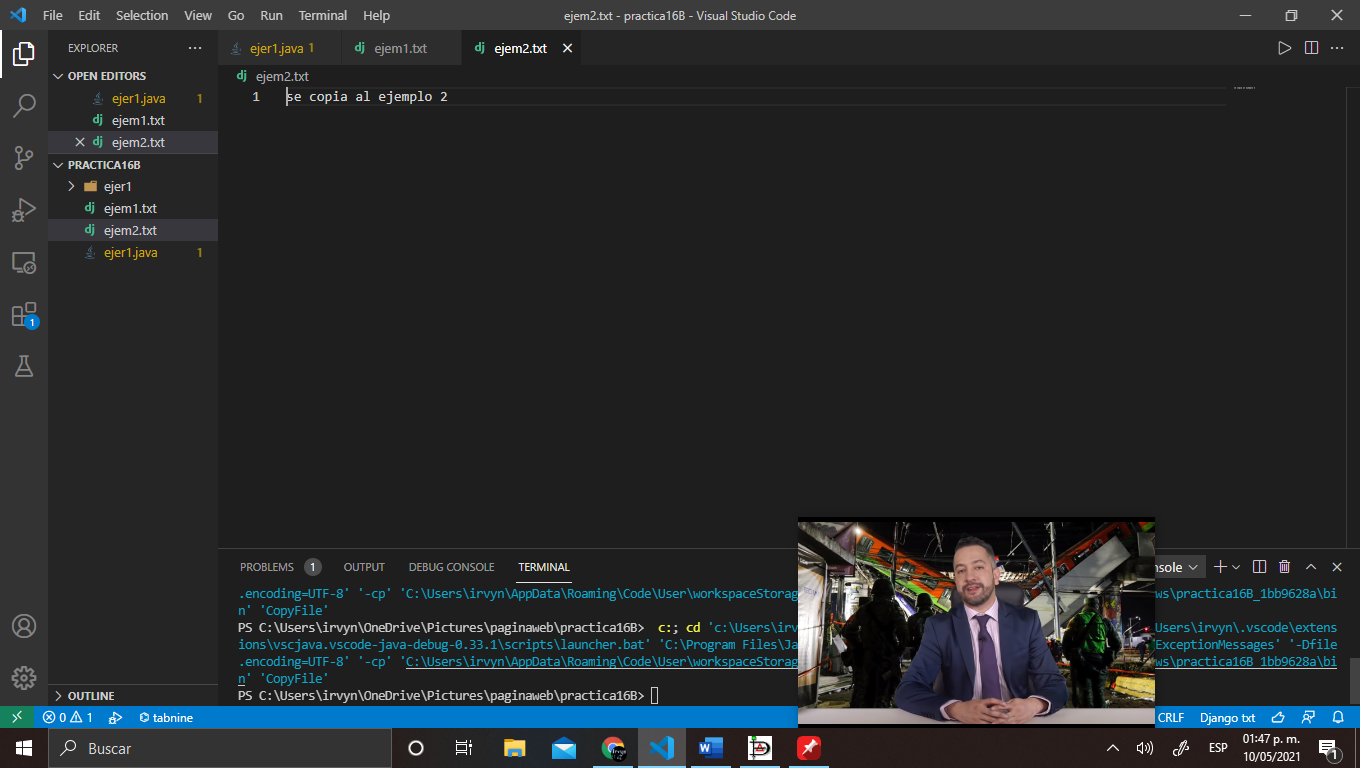
        fout.close();

    }

}

**PRUEBA DE ESCRITORIO**





**PROGRAMA EJERCICIO 2**

**CLASE FILEDEMO**

import java.io.File;

class FileDemo {

    static void p(String s) {

        System.out.println(s);

    }

    public static void main(String args[]) {

        File f1 = new File("ejer1/ejer1.java");

        p("Archivo: " + f1.getName());

        p("Directorio: " + f1.getPath());

        p("Directorio Absoluto: " + f1.getAbsolutePath());

        p("Padre: " + f1.getParent());

        p(f1.exists() ? "Existe" : "No Existe");

        p(f1.canWrite() ? "Se Puede Escribir" : "No Se Puede Escribir");

        p(f1.canRead() ? "Se Puede Leer" : "No Se Puede Leer");

        p((f1.isDirectory() ? " " : "No") + "Es Un Directorio");

        p(f1.isFile() ? "Es Un Archivo Normal" : "Podria Ser Un Enlace Con Nombre");

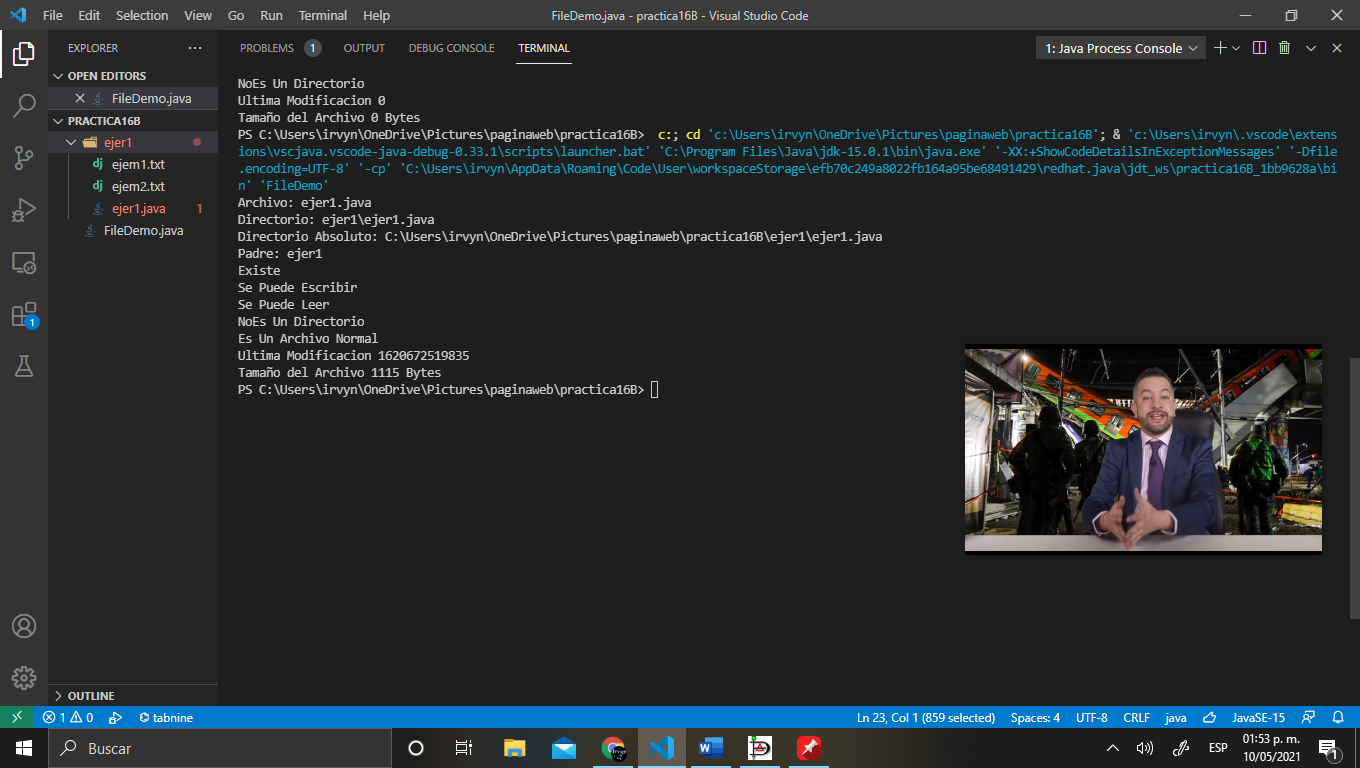
        p("Ultima Modificacion " + f1.lastModified());

        p("Tamaño del Archivo " + f1.length() + " Bytes");

    }

}

**PRUEBA DE ESCRITORIO**



**PROGRAMA EJERCICIO 3**

**CLASE FILEOUTPUTSTREAMDEMO**

import java.io.\*;

class FileOutputStreamDemo {

    public static void main(String args[]) throws Exception {

        String source = "Ahora es el momento de que los hombres buenos\n" + "vengan a ayuda a su país\n"

                + "y paguen sus impuestos";

        byte buf[] = new byte[source.length()];

        source.getBytes(0, buf.length, buf, 0);

        OutputStream f0 = new FileOutputStream("file.txt");

        for (int i = 0; i < buf.length; i += 2) {

            f0.write(buf[i]);

        }

        f0.close();

        OutputStream f1 = new FileOutputStream("file2.txt");

        f1.write(buf);

        f1.close();

        OutputStream f2 = new FileOutputStream("file3.txt");

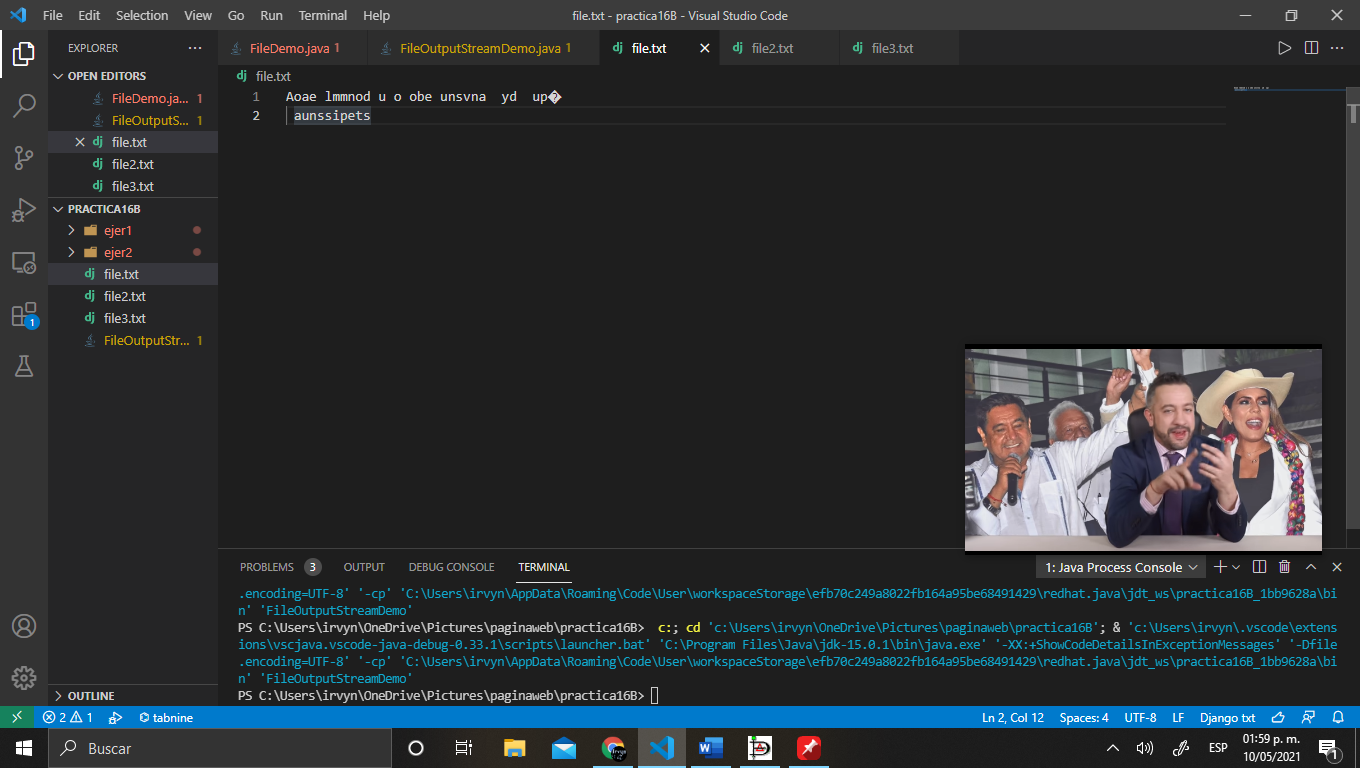
        f2.write(buf, buf.length - buf.length / 4, buf.length / 4);

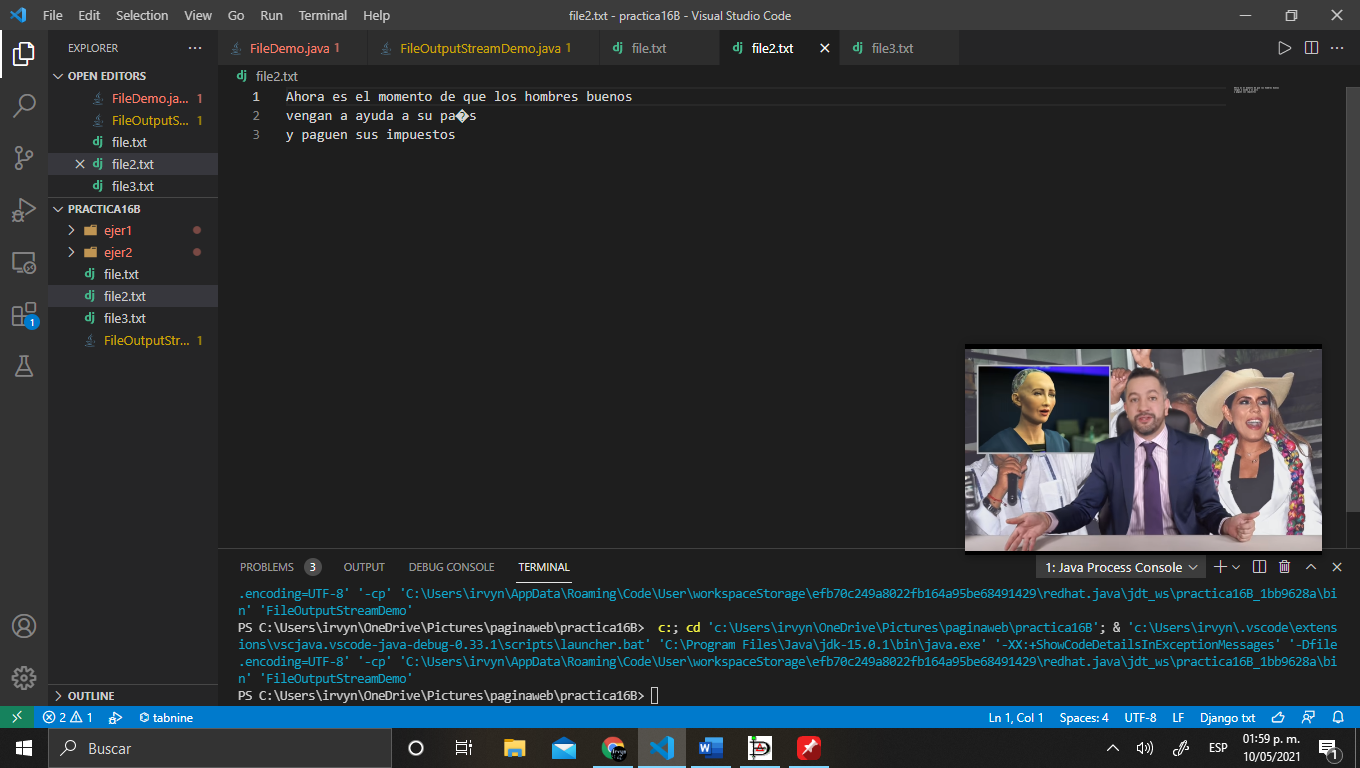
        f2.close();

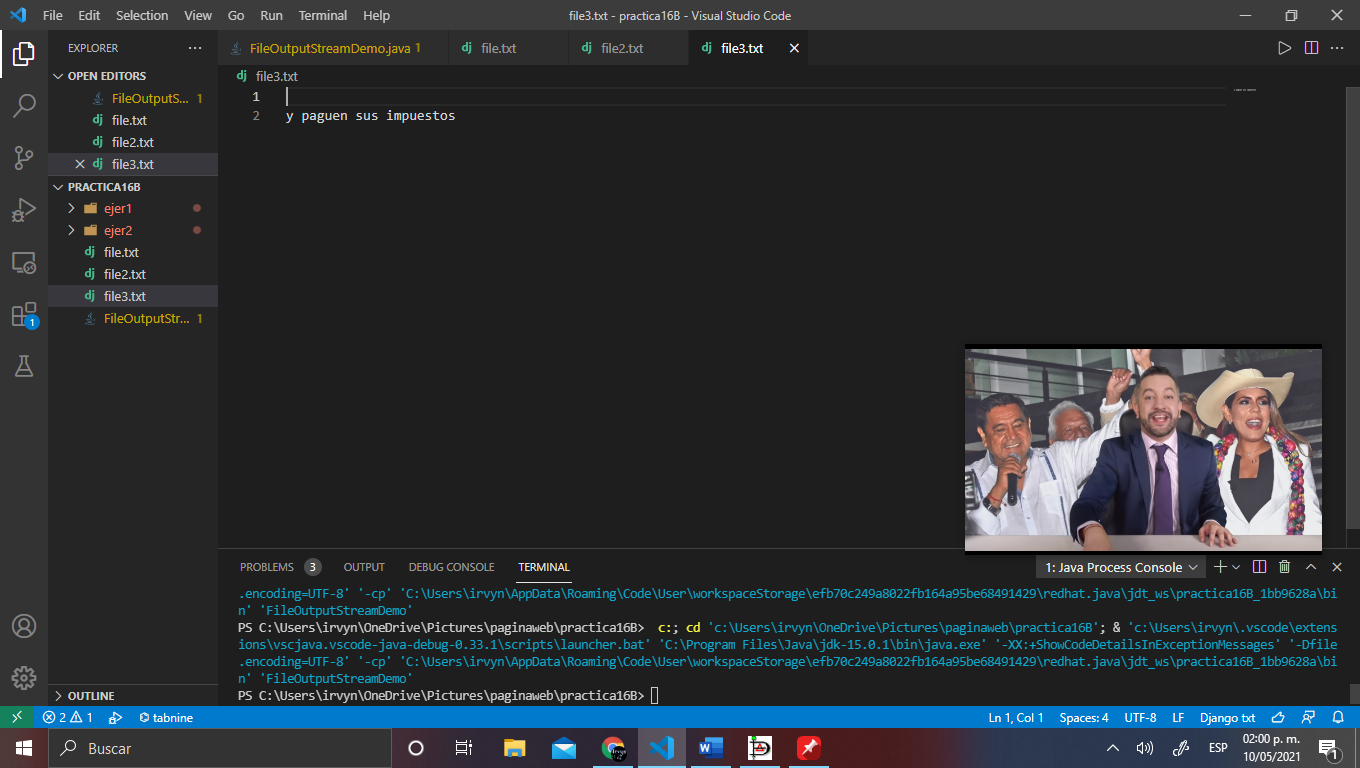
    }

}

**PRUEBA DE ESCRITORIO**







**PROGRAMA EJERCICIO 4**

**CLASE ONLYEXT**

import java.io.\*;

public class OnlyExt implements FilenameFilter{

    String ext;

    public OnlyExt(String ext){

        this.ext="."+ext;

    }

    public boolean accept (File dir, String name){

        return name.endsWith(ext);

    }

}

class DirListOnly {

    public static void main(String args[]) {

        String dirname = "ejer1/";

        File f1 = new File(dirname);

        FilenameFilter only = new OnlyExt("java");

        String s[] = f1.list(only);

        for (int i = 0; i < s.length; i++) {

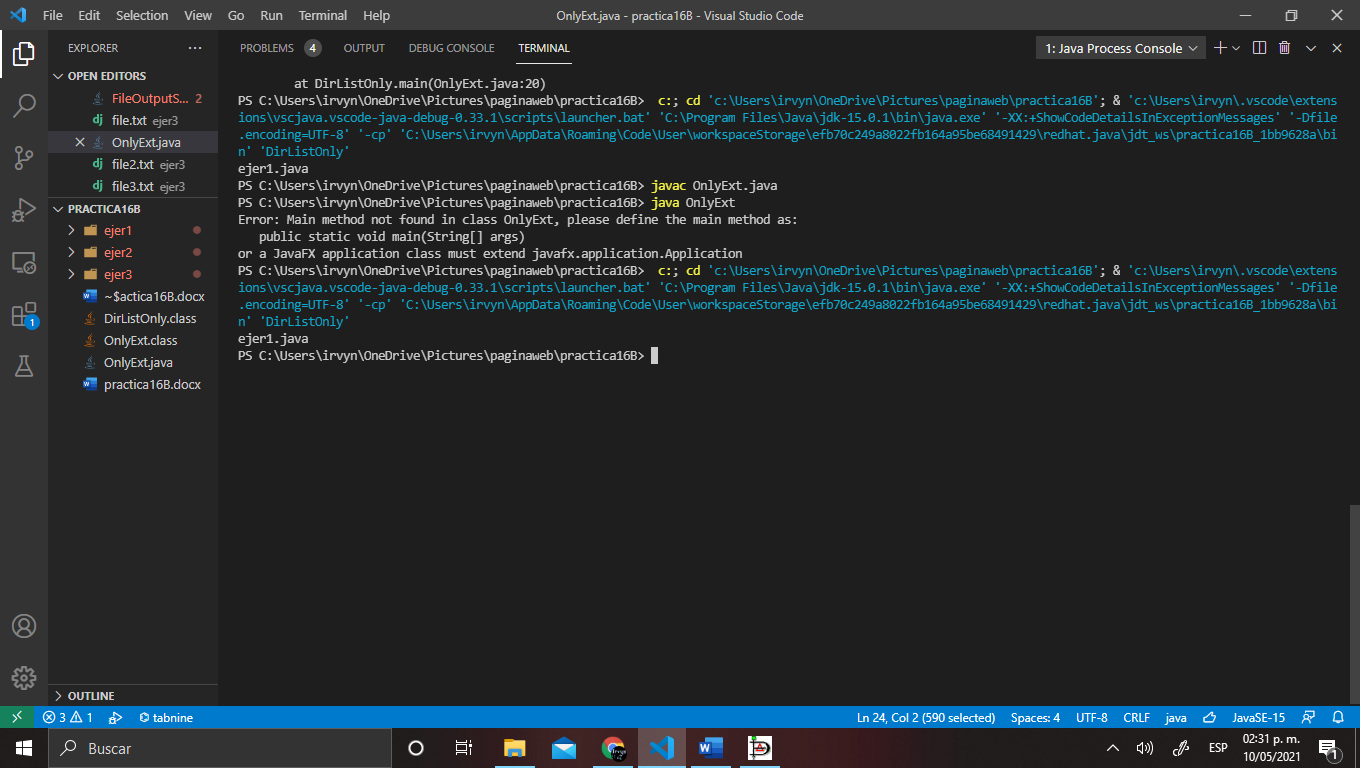
            System.out.println(s[i]);

        }

    }

}

**PRUEBA DE ESCRITORIO**



**PROGRAMA EJERCICIO 5**

**CLASE DIRLISTMOD**

import java.io.\*;

class DirListMod {

    public static void main(String args[]) {

        String dirname = "ejer1";

        File f1 = new File(dirname);

        if (f1.isDirectory()) {

            System.out.println("Directorio de " + dirname);

            String s[] = f1.list();

            for (int i = 0; i < s.length; i++) {

                File f = new File(dirname + "/" + s[i]);

                if (f.isDirectory()) {

                    System.out.println(s[i] + "\tEs un directorio");

                } else {

                    System.out.println(s[i] + "\tEs un archivo");

                }

            }

        } else {

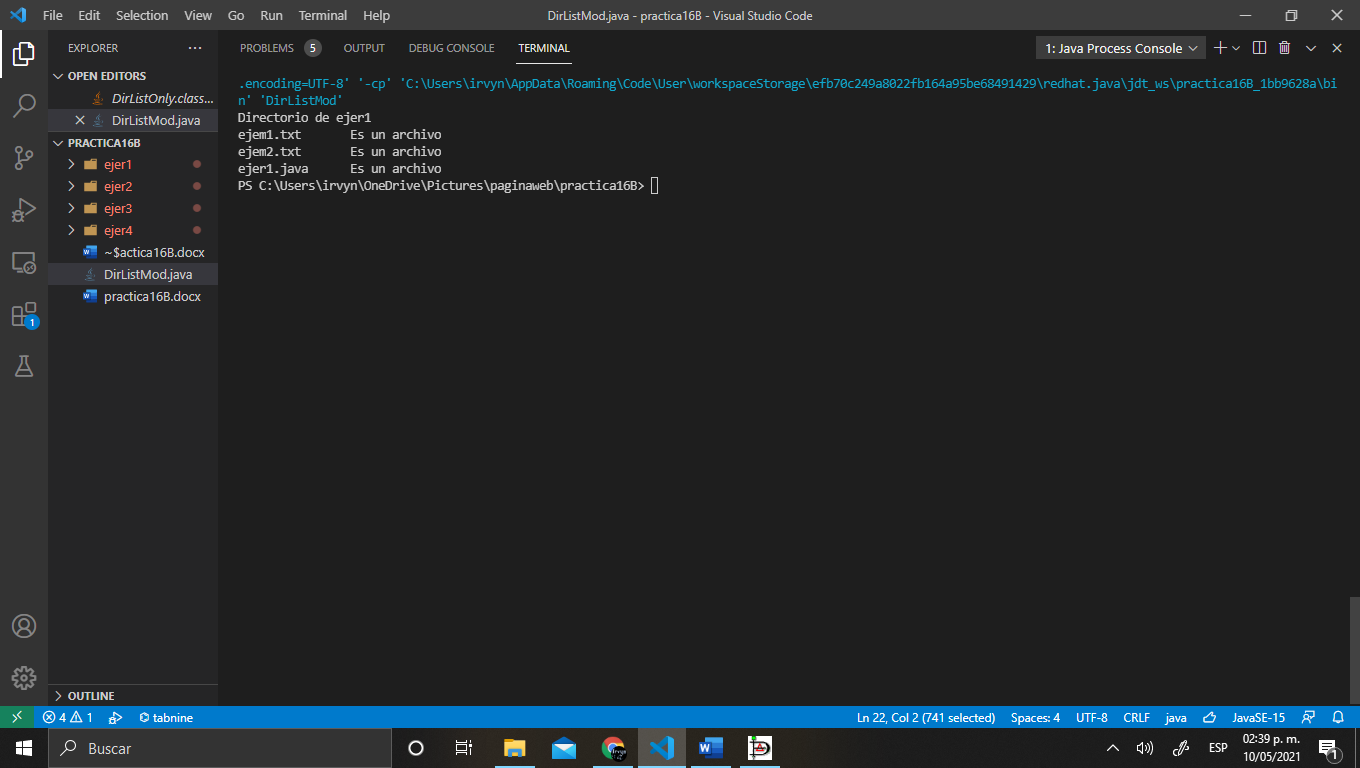
            System.out.println(dirname + "\tNo es un directorio");

        }

    }

}

**PRUEBA DE ESCRITORIO**



**PROGRAMA EJERCICIO 6**

**CLASE ARCHIVOS**

import java.io.\*;

public class Archivos {

    public static void main(String[] args){

        File in = new File("entrada.txt");

        File out = new File("salida.txt");

        FileReader fr;

        FileWriter fw;

        BufferedReader br;

        PrintWriter pr;

        try{

            fr=new FileReader(in);

            fw=new FileWriter(out);

            br=new BufferedReader(fr);

            pr=new PrintWriter(fw);

            String linea=br.readLine();

            while(linea!=null){

                int num=Integer.parseInt(linea);

                int fac=factorial(num);

                pr.println(fac);

                linea=br.readLine();

            }

            fr.close();

            fw.close();

        }catch (IOException e){

            System.err.println("Error");

        }

    }

    public static int factorial(int n){

        int f=1;

        for(int i=1;i<=n;i++){

            f=f\*i;

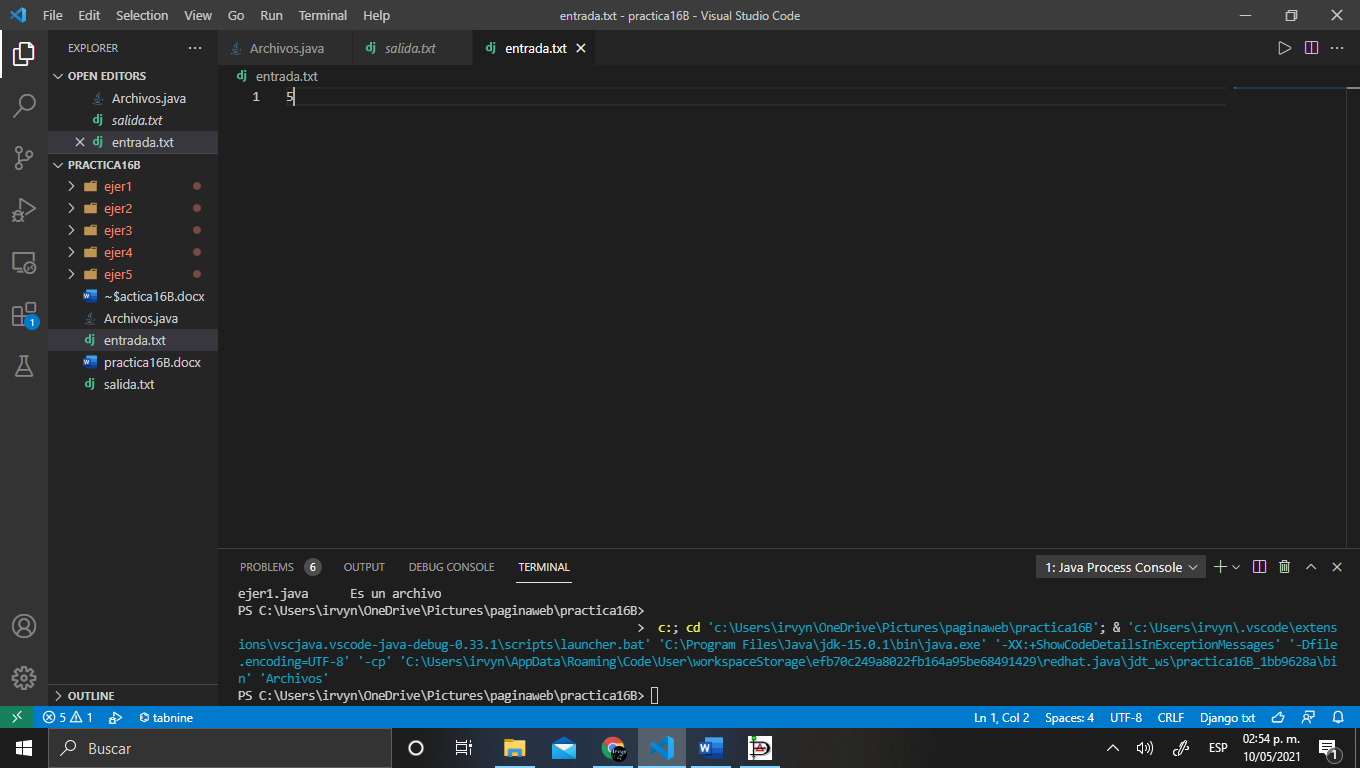
        }

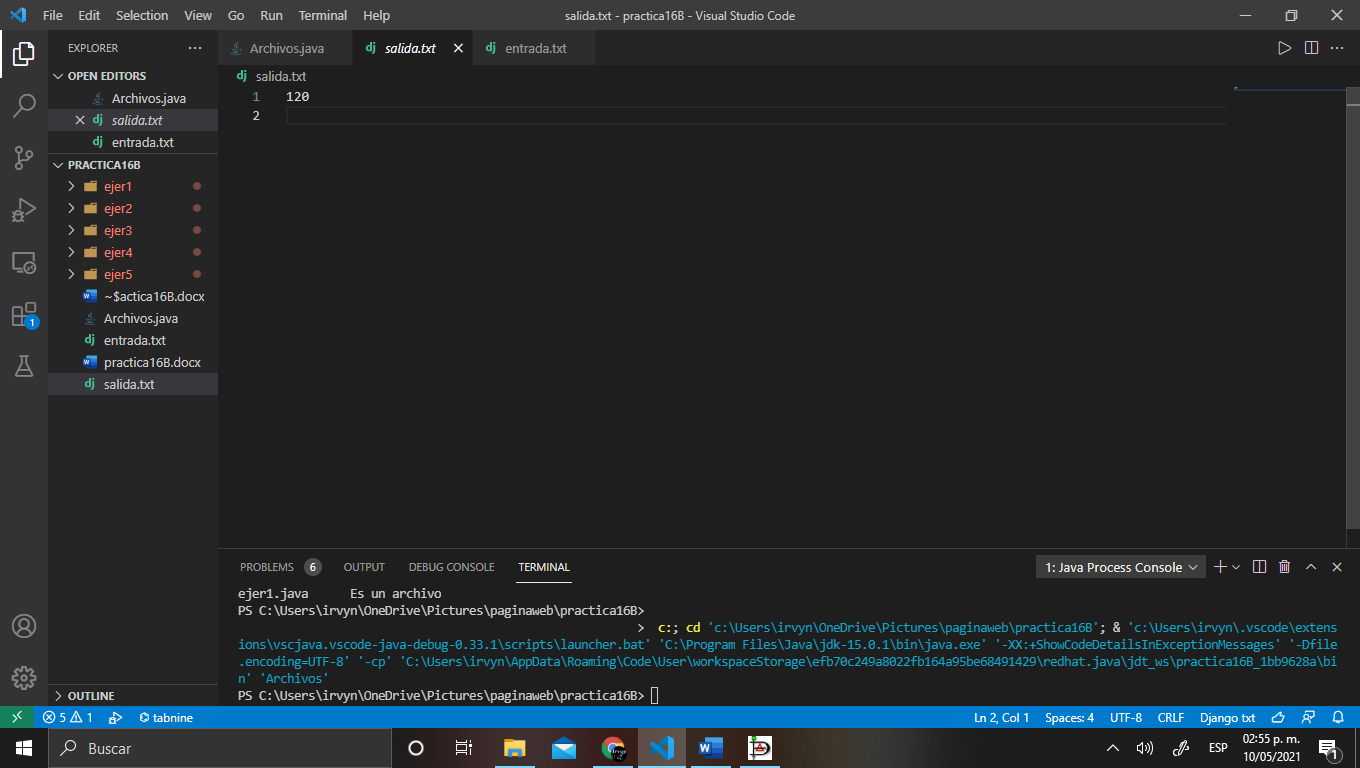
        return f;

    }

}

**PRUEBA DE ESCRITORIO**





**PROGRAMA EJERCICIO 7**

**CLASE LIBRO**

import java.io.\*;

class Libro implements Serializable {

    private String titulo;

    private String autor;

    private String editorial;

    private int pagina;

    public Libro() {

        titulo = autor = editorial = null;

    }

    public Libro(String t, String a, String e, int pg) {

        titulo = t;

        autor = a;

        editorial = e;

        pagina = pg;

    }

    public void entrada(BufferedReader ent) throws IOException {

        System.out.print("Titulo: ");

        titulo = ent.readLine();

        System.out.print("Autor: ");

        autor = ent.readLine();

        System.out.print("Editorial: ");

        editorial = ent.readLine();

        System.out.print("Páginas: ");

        pagina = Integer.parseInt(ent.readLine());

    }

}

class Disco implements Serializable {

    private String artista;

    private String titulo;

    private int numCancion, duracion;

    private transient double precio;

    public Disco() {

        artista = titulo = null;

    }

    public Disco(String a, String t, int n, int d, double p) {

        titulo = t;

        artista = a;

        numCancion = n;

        duracion = d;

        precio = p;

    }

    public void entrada(BufferedReader ent) throws IOException {

        System.out.print("Cantante: ");

        artista = ent.readLine();

        System.out.print("Titulo: ");

        titulo = ent.readLine();

        System.out.print("Canciones: ");

        numCancion = Integer.parseInt(ent.readLine());

        System.out.print("Duración(minutos): ");

        duracion = Integer.parseInt(ent.readLine());

        System.out.print("Precio: ");

        precio = Double.valueOf(ent.readLine()).doubleValue();

    }

}

**CLASE REVISTA**

import java.io.\*;

public class Revista implements Serializable {

    private String titulo;

    private String autor;

    private String editorial;

    private int pagina;

    public Revista() {

        titulo = autor = editorial = null;

    }

    public Revista(String t, String a, String e, int pg) {

        titulo = t;

        autor = a;

        editorial = e;

        pagina = pg;

    }

    public void entrada(BufferedReader ent) throws IOException {

        System.out.print("Titulo: ");

        titulo = ent.readLine();

        System.out.print("Autor: ");

        autor = ent.readLine();

        System.out.print("Editorial: ");

        editorial = ent.readLine();

        System.out.print("Páginas: ");

        pagina = Integer.parseInt(ent.readLine());

    }

}

**CLASE LIBRERÍA**

import java.io.\*;

public class Libreria {

    public static void main(String[] a) {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        String tipo = "";

        File mf = new File("libreria.dat");

        ObjectOutputStream fobj = null;

        Libro libro = new Libro();

        Disco disco = new Disco();

        Revista revista = new Revista();

        boolean mas = true;

        try {

            fobj = new ObjectOutputStream(new FileOutputStream(mf));

            do {

                System.out.println("Seleccione la opcion:");

                System.out.println("L-libro");

                System.out.println("D-disco");

                System.out.println("R-revista");

                System.out.println("F-finalizar)");

                tipo = br.readLine();

                switch (tipo) {

                    case "L":

                    case "l":

                        libro = new Libro();

                        libro.entrada(br);

                        fobj.writeObject(libro);

                        break;

                    case "D":

                    case "d":

                        disco = new Disco();

                        disco.entrada(br);

                        fobj.writeObject(disco);

                        break;

                        case "R":

                        case "r":

                            revista = new Revista();

                            revista.entrada(br);

                            fobj.writeObject(revista);

                            break;

                    case "F":

                    case "f":

                        fobj.close();

                        mas = false;

                }

            } while (mas != false);

        } catch (IOException e) {

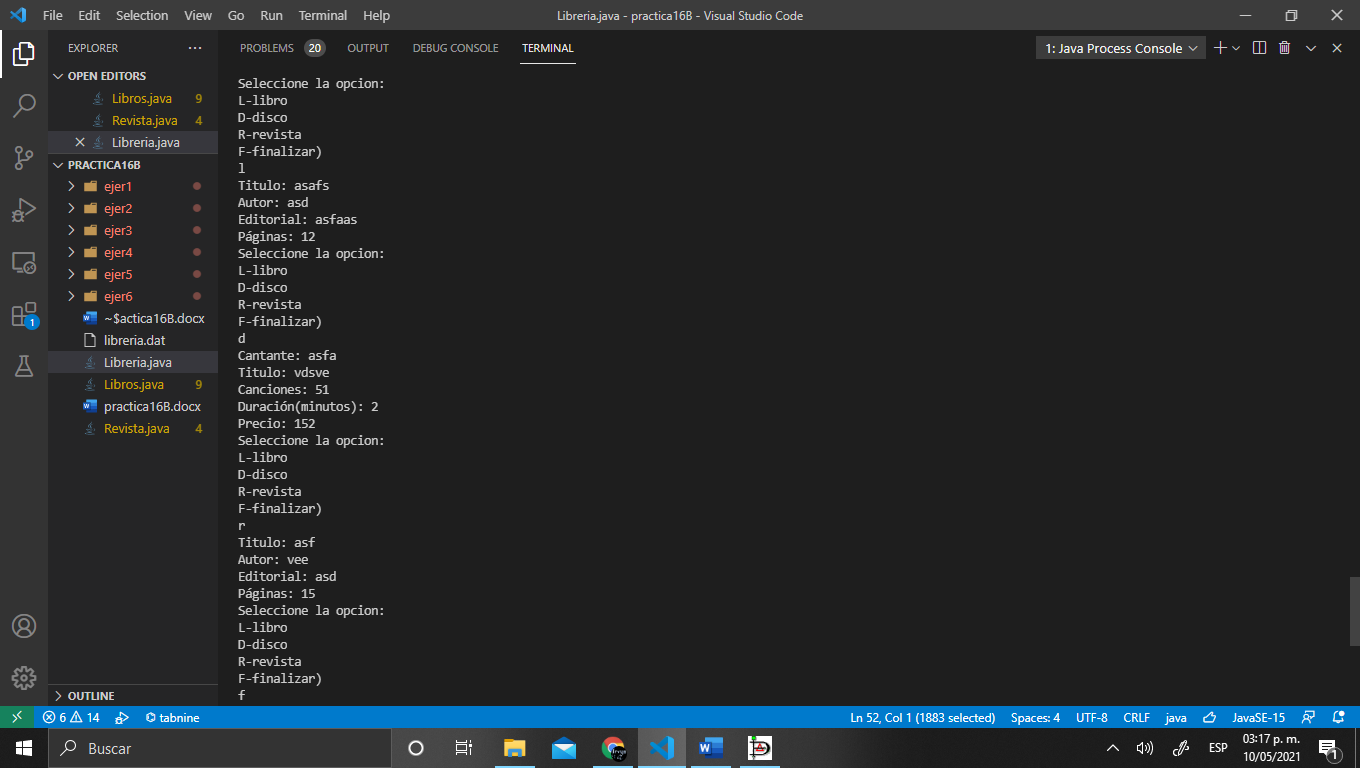
            e.printStackTrace();

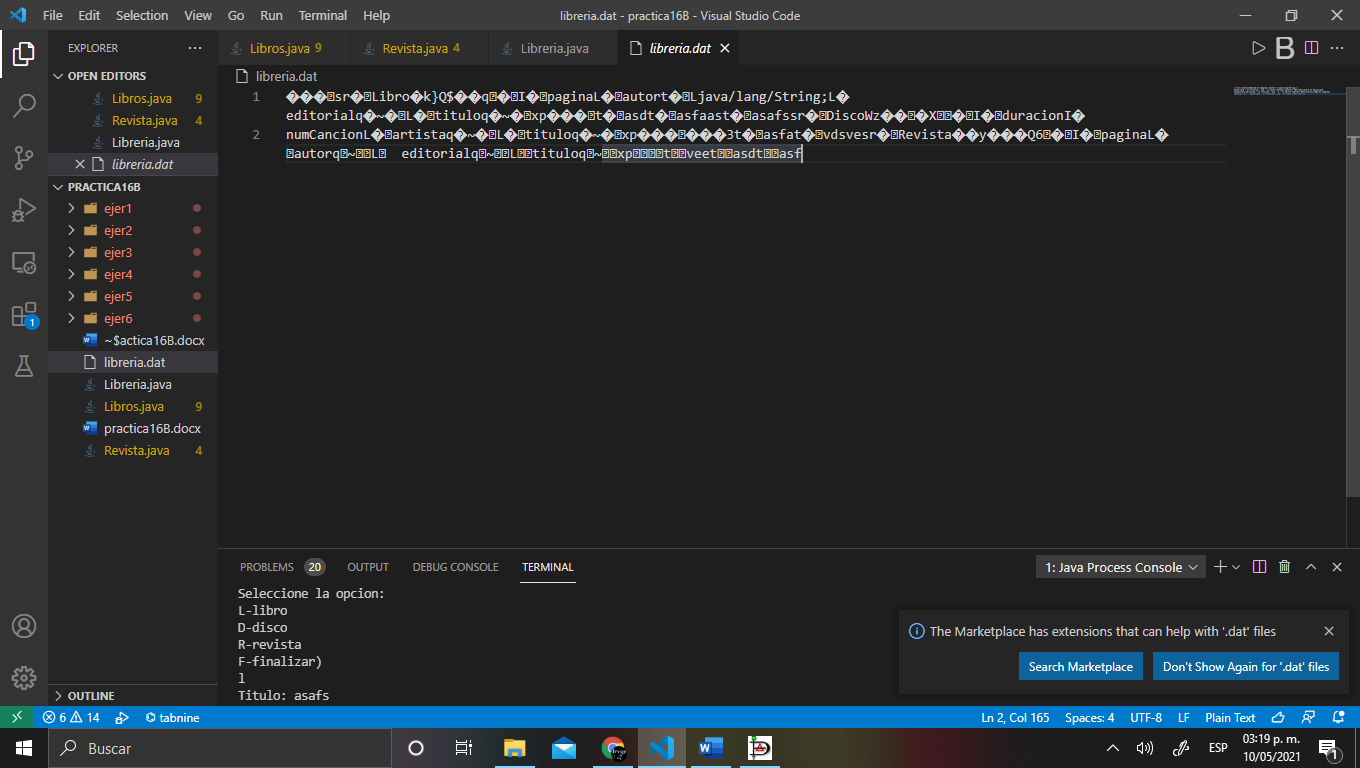
        }

    }

}

**PRUEBA DE ESCRITORIO**





**PROGRAMA EJERCICIO 8**

**CLASE PERSONAS**

import java.io.Serializable;

public class Personas implements Serializable {

    String nombre;

    int edad;

    String curp;

    public Personas() {

    }

    public Personas(String nombre, int edad, String curp) {

        super();

        this.nombre = nombre;

        this.edad = edad;

        this.curp = curp;

    }

    public String getNombre() {

        return nombre;

    }

    public int getEdad() {

        return edad;

    }

    public String getCurp() {

        return curp;

    }

}

**CLASE LECTURAESCRITURA**

import java.io.\*;

//Clase LecturaEscritura main

public class LecturaEscritura {

    // método de escritura de un arreglo a un archivo de objetos

    public static void setEscritura() throws IOException {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        String tipo = "";

        String vrnombre, vrcurp = "";

        int vredad;

        int i = 0;

        File f = new File("datos2.obj");

        FileOutputStream fos = new FileOutputStream(f);

        ObjectOutputStream oos = new ObjectOutputStream(fos);

        System.out.println("dame el número de Personas a ingresar");

        int num = Integer.parseInt(br.readLine());

        Personas arreglo[] = new Personas[num];

        try {

            for (i = 0; i < num; i++) {

                System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

                System.out.println("Dame nombre:");

                vrnombre = br.readLine();

                System.out.println("Dame edad:");

                vredad = Integer.parseInt(br.readLine());

                System.out.println("Dame CURP:");

                vrcurp = br.readLine();

                System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

                // creamos el objeto en el arreglo

                arreglo[i] = new Personas(vrnombre, vredad, vrcurp);

                // mandamos a escribir el primero objeto del arreglo al archivo

                oos.writeObject(arreglo[i]);

            }

        } catch (IOException error) {

            System.err.println("Error " + error.getMessage());

        } finally {

            // cerramos el archivo de objetos

            oos.close();

        }

    }

    // método de lectura de objetos y pasar a un arreglo

    public static void getLectura() throws ClassNotFoundException, IOException {

        int i = 0, num = 1;

        ObjectInputStream ois = null;

        try {

            File f = new File("datos2.obj");

            FileInputStream fis = new FileInputStream(f);

            ois = new ObjectInputStream(fis);

            Personas arreglo1[] = new Personas[10];

            while (true) {

                Personas p = (Personas) ois.readObject();

                // pasamos el objeto al arreglo en la posición 0

                arreglo1[i] = p;

                // mostramos desde el archivo

                System.out.println("\*\*\*\*\*\*Registro\*\*\*\* " + (num++));

                System.out.println("Nombre :" + p.getNombre());

                System.out.println("Edad : " + p.getEdad());

                System.out.println("CURP: " + p.getCurp());

                System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

                // Mostramos desde el arreglo

                System.out.println("\*\*\*\*\*\*Arreglo\*\*\*\* " + i);

                System.out.println("Nombre: " + arreglo1[i].nombre);

                System.out.println("Edad: " + arreglo1[i].edad);

                System.out.println("CURP: " + arreglo1[i].curp);

                // incrementamos posición en el arreglo

                i++;

            }

        } catch (IOException io) {

            System.out.println("\n\*\*\*\*\*\*\*\*\*\*\* Fin\*\*\*\*\*\*\*\*\*\*\*\*");

        } finally {

            // creamos archivo

            ois.close();

        }

    }

    public static void main(String[] args) throws IOException, ClassNotFoundException {

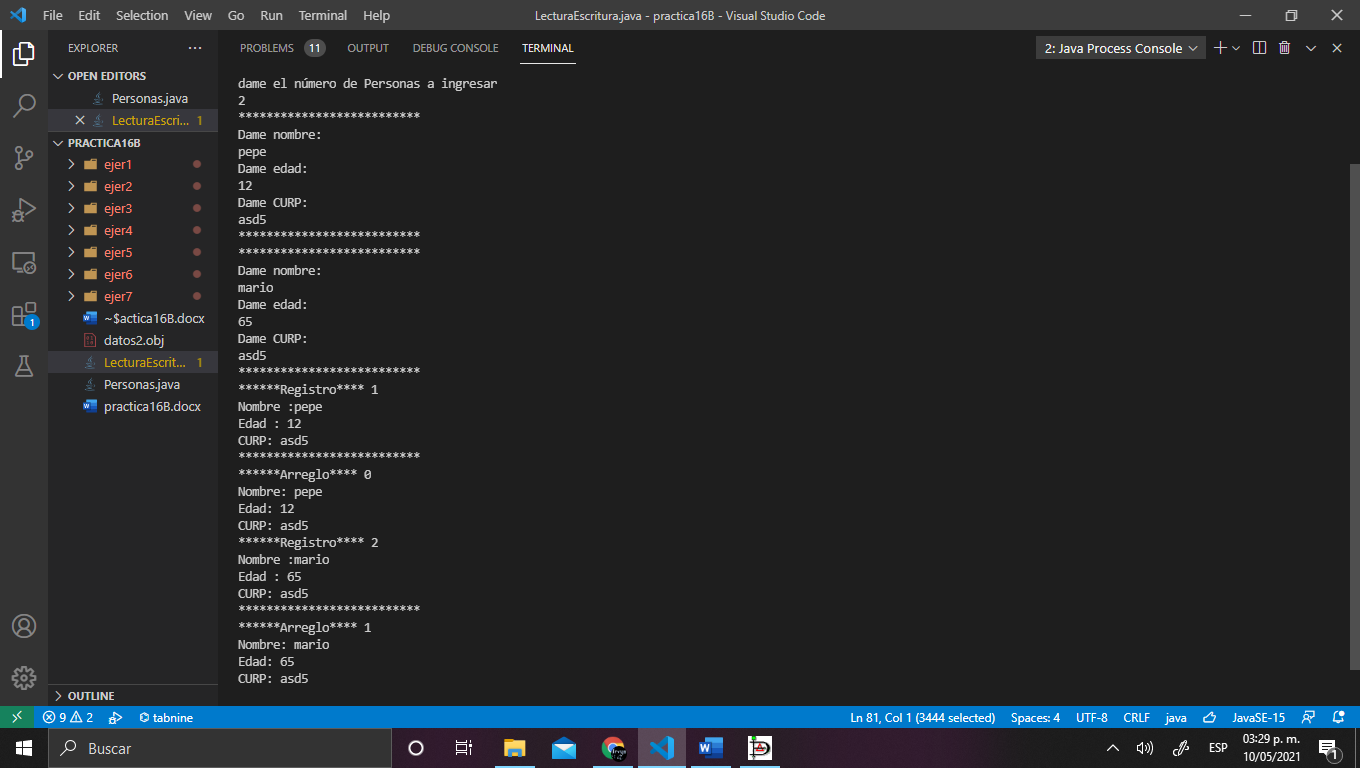
        setEscritura();

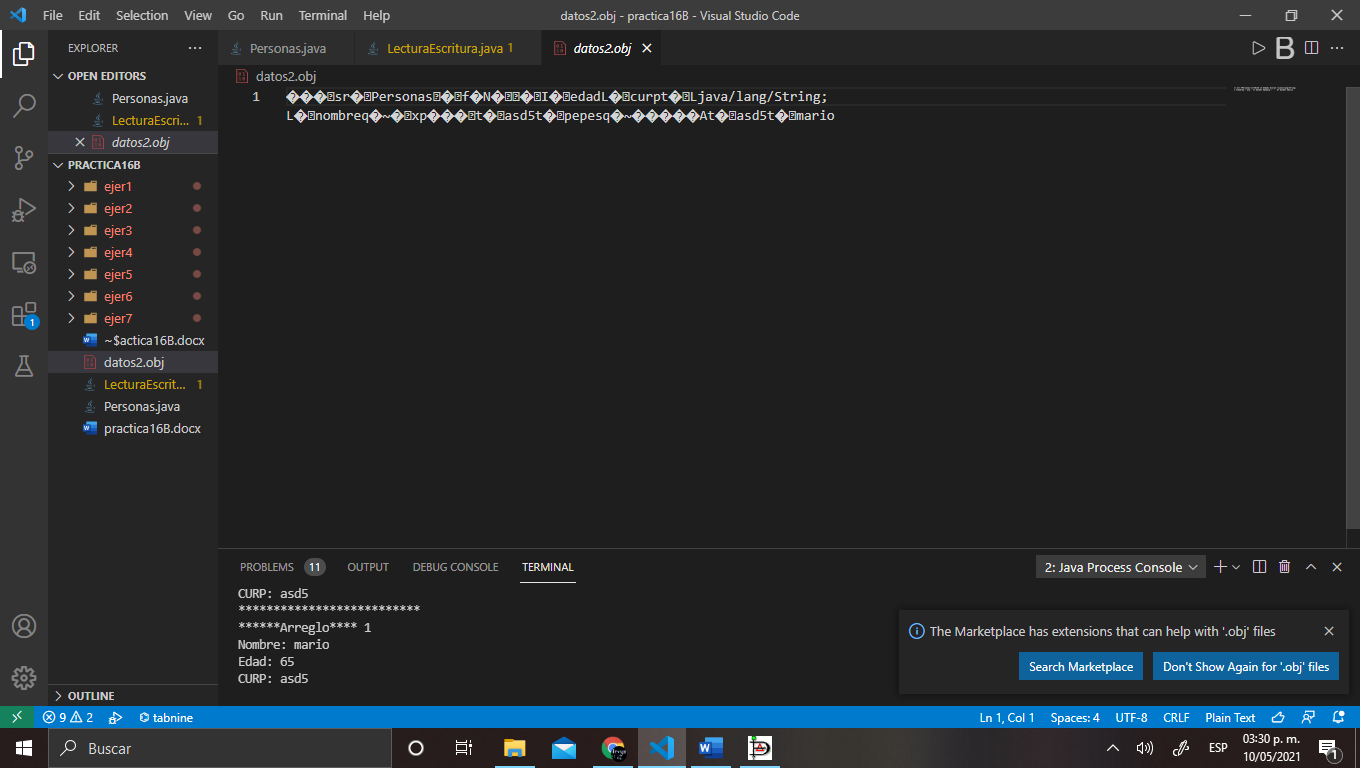
        getLectura();

    }

}

**PRUEBA DE ESCRITORIO**





**CONCLUSIÓN**

Los archivos son muy importantes y un gran recurso que podemos usar en java y si logramos manejar los archivos ampliaremos nuestro conocimientos en java.

**REFERENCIAS BIBLIOGRÁFICAS**

* Fagg, A.H.(2017). GUI Programming: Components. Consultado el día 12 de agosto de 2020 en: <https://www.cs.ou.edu/~fagg/classes/cs2334/lecture/GUI-Components.pdf>
* Horstmann, C. S., & Cornell, G. (2013). Core Java Volumen I Fundamentals. New Jersey: Prentice Hall.
* Joyanes Aguilar, L., & Zahonero Martínez, I. (2011). Programación en Java 6. Algoritmos y programación orientada a objetos. Cd. de México: Mc Graw Hill.
* Poo, D., Kiong, D., & Ashok, S. (2008). Object-Oriented Programming and Java. Singapore: Springer.
* Sznajdleder, P. A. (2016). Java a fondo. Alfaomega.
* Liang, Y. D. ( 2014 ). Introduction to Java Programming, Comprehensive Version. Pearson
* Malik D. S. (2012 ). Java Programming: From Problem Analysis to Program Design, Fifth Edition. Cengage Learning
* GeeksforGeeks. (Última edición: 20 de Julio de 2020) , GeeksforGeeks. https://www.geeksforgeeks.org/types-of-recursions .Consultado de 30 de julio del 2020

**PROGRAMACION II**

**RUBRICA HOJA DE EVALUACIÓN DE PRACTICAS**

**MATRICULA: 201963582 FECHA: 9-MAYO-2021**

**NOMBRE: Irvyn Xicale Cabrera NO. PRACTICA: 16B INDIVIDUAL ( ) COLABORATIVA(x)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Deficiente** | **Suficiente** | **Bueno** | **Excelente** | **Calificación Obtenida** | |
| **CRITERIOS** | **1-5.9** | **6‐7.9** | **8‐9** | **9.1‐10** |
| **%** | **Puntos** |
| **CONOCIMIENTO TEORICO**  **20%** | Conocimiento deficiente de los fundamentos teóricos de POO y no puede aplicarlos en el diseño de clases. | Conocimiento confuso de los fundamentos teóricos de POO y el diseño de las clases y relaciones es incompleto. | Conocimiento claro de los fundamentos teóricos POO, pero requiere mejorar el modelado de las clases, sus métodos  y sus relaciones entre clases. | Dominio del Conocimiento de los fundamentos teóricos POO y puede aplicarlos de forma completa en el modelado de las clases, métodos y todas las relaciones entre clase. |  |  |
| **EJECUCCIÓN DE LA PRACTICA**  **30%** | No puede realizar la práctica ya que desconoce el entorno de trabajo y desarrollo de la práctica en lenguaje UML y Java. | Realiza la práctica de forma incompleta ya que desconoce el entorno de trabajo del lenguaje UML y Java | Realiza la práctica, pero requiere mejorar en el manejo del entorno de trabajo del lenguaje de programación (sintaxis y semántica) | Realiza la práctica de forma correcta y completa, demuestra dominio del entorno de trabajo del lenguaje de programación (sintaxis y semántica). |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOLUCIÓN DE LA PRACTICA**  **40%** | No puede generar las soluciones o programas a los problemas planteados ya que no posee el dominio teórico y práctico del modelado y el lenguaje de programación. | Propone soluciones confusas o programas incompletos a los problemas planteados, ya que carece del dominio del modelado y lenguaje de programación | Genera soluciones con poca profundidad y los programas no están orientados de acuerdo a los problemas solicitados, por lo cual no tiene un dominio profundo de la temática y del Lenguaje. | Genera soluciones con profundidad y los programas son correctos de acuerdo a los problemas planteados, por lo cual demuestra un dominio de la temática y del Lenguaje de Programación. |  |  |
| **ACTITUD DE APRENDER Y COLABORAR EN EQUIPO DE TRABAJO**  **10%** | No posee una actitud proactiva para un aprendizaje autónomo y no le gusta participar y trabajar en equipo. | Posee una actitud propositiva para un aprendizaje autónomo, participa, pero no le gusta trabajar en equipo. | Posee una actitud propositiva logrando un aprendizaje autónomo, colaborativo, le gusta trabajar en equipo, pero requiere mejorar su participación y portaciones de forma profunda. | Posee una actitud proactiva logrando un aprendizaje autónomo, participa con propuestas concretas y profundas, le gusta trabajar en equipo y asume su responsabilidad dentro para lograr éxito del equipo |  |  |
| **Total** | | | | |  |  |  |

