BDD:

1. Escribe la instrucción para crear una tabla “clientes”. Esta tabla tiene los siguientes campos:
    a. id_cliente: entero (este campo es el campo clave)
    b. nombre: cadena de caracteres de tamaño 30
    c. dirección: cadena de caracteres de tamaño 50
    d. teléfono: cadena de caracteres de tamaño 10

```
CREATE TABLE clientes(

id_cliente INT PRIMARY KEY,

id_interno INT NOT NULL AUTO_INCREMENT,

nombre VARCHAR(30),

direccion VARCHAR(50),

telefono VARCHAR(10)

);
```

/*id_interno (se va incrementando) se ha utilizado por si se muere el propietario del DNI para que no se quede con el mismo DNI*/

2. SDEscribe la instrucción para inserter 2 clientes en esa tabla (invéntate los valores).

```
INSERT INTO clientes VALUES

(1, “David”, “Calle x”, “1234567890”),

(2, “Pepe”, “Calle y”, “0987654321”);
```

3. Escribe una instrucción para visualizar todos los registros y toda la información de la tabla clientes.

```
SELECT * FROM clientes;
```

4. Escribe una instrucción para visualizar el nif de un cliente llamado “pepe”.

```
SELECT id_cliente FROM clientes WHERE nombre = “Pepe”;
```

5. Escribe una instrucción para visualizar el nombre de cliente y el teléfono de todos los clientes de la tabla.

```
SELECT nombre, telefono FROM clientes;
```

PROGRAMACIÓN:

1. Escribe el Código necesario para mostrar por pantalla el contenido del siguiente array: int números={5,3,1,0,7}

```
int numeros[] = {5,3,1,0,7};

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

       System.out.println("Posición: " + i + " , Número: " + numeros[i]);

    }
```

2. Crea una clase “Cliente” que tenga las propiedades:
    a. id_cliente: entero
    b. nombre: cadena de caracteres
    c. dirección: cadena de caracteres
    d. teléfono: cadena de caracteres

(Creación de la clase primero)

```
int id_cliente;

String nombre, dirección, telefono;
```

3. En la clase “Cliente”, añade un método para visualizar la información de un cliente.

Clase cliente:

```
@Override

   public String toString() {

      return "cliente{" + "id_cliente=" + id_cliente + ", nombre=" + nombre + ", direcci\u00f3n="
+ dirección + ", telefono=" + telefono + '}';

   }
```

4. En la clase cliente, crea un étodo void pedirDatos() que pida por pantalla todos los datos de un cliente. No hay que hacer control de excepciones.

Clase cliente:

```
void pedirDatos(){


    Scanner sc = new Scanner(System.in);

    System.out.print("Introduce el ID del cliente: ");

    id_cliente = sc.nextInt();
```

```
        System.out.print("Introduce el Nombre del cliente: ");
        nombre = sc.next();
        sc.nextLine();

        System.out.print("Introduce la Dirección del cliente: ");
        direccion = sc.nextLine();

        System.out.print("Introduce el Teléfono del cliente: ");
        telefono = sc.next();
    }
```

Clase lanzadora:

```
cliente miCliente = new cliente();
        miCliente.pedirDatos();
        String datos = miCliente.toString();
        System.out.println(datos);
```

5. En la clase cliente, crea un método void guardarDatos() que escriba los datos de un cliente en un archivo misclientes.txt.

```
    void guardarDatos(){
        String creaFichero = "./archivos/misClientes.txt";
        File fichero = new File(creaFichero);

        if (fichero.exists())
            System.out.println("El fichero ya existe.");
        try (BufferedWriter escritura = new BufferedWriter(new FileWriter(creaFichero))){
//try-with-resources.
            //se cierra automáticamente al final del bloque sin necesidad de escribir un bloque
'finally'.
            escritura.write(toString());
```

```
    }catch(Exception e){
       e.printStackTrace();
    }
  }
```

PARTE II:

1. Crea un programa en el que pidas dos números enteros por pantalla y calcules su división, para mostrar el resultado por pantalla.

```
Scanner sc = new Scanner(System.in);
    int n1, n2, resultado;

    System.out.println("Introduce 2 números:");
    System.out.print("Número 1: ");
    n1 = sc.nextInt();
    System.out.print("Número 2: ");
    n2 = sc.nextInt();

    resultado = n1 / n2;
```

2. Repite el programa con control de Excepciones: controla la excepción de que el usuario escriba algo que no sea un número para que el programa no aborte mostrando un mensaje que diga “No se ha podido realizar la operación”.

```
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n1, n2;
    double resultado;

    try {
    System.out.println("Introduce 2 números:");
    System.out.print("Número 1: ");
    n1 = sc.nextInt();
```

```
        System.out.print("Número 2: ");
        n2 = sc.nextInt();

        resultado = (double) n1 / n2;

        System.out.println("El resultado es: " + resultado);
        } catch (java.util.InputMismatchException e){
            System.out.println("No se ha podido realizar la operación");
        }
        }
    }
```

3. En el código anterior, controla la excepción que se produce si el segundo número es un 0. El control debe hacer que se vuelva a pedir otra vez el número.

Como “resultado” es double, no salta ninguna excepción, salta el mensaje “Infinity”.

El resultado es: Infinity