

Instituto Tecnológico Superior del Occidente del Estado de Hidalgo

ITSOEH

“Programación orientada a objetos”

Tecnologías de la información y comunicaciones
(TIC's)

Maria Guadalupe Tolentino Cruz

Kleyder Murillo Martínez (230110626)

2 “C”

1

Competencia

Comprende y aplica la estructura de clases para la creación de objetos y utiliza clases predefinidas para facilitar el desarrollo de aplicaciones.

Comprende y aplica los diferentes tipos de métodos, tomando en cuenta el ámbito y tiempo de vida de los datos durante la ejecución de un programa.

Introducción

La programación orientada a objetos (POO) es un paradigma que organiza el diseño del software alrededor de objetos y datos, en lugar de funciones y lógica. Java, siendo un lenguaje de programación orientado a objetos, se utiliza ampliamente debido a su robustez, portabilidad y escalabilidad.

Clases y Objetos:

Clase: Es una plantilla o modelo que define las propiedades y comportamientos comunes de un grupo de objetos. En Java, una clase se define usando la palabra clave class.

Objeto: Es una instancia de una clase. Representa una entidad en el mundo real con características (atributos) y comportamientos (métodos).

Atributos y Métodos:

Atributos: Son las variables que almacenan el estado o características de un objeto.

Métodos: Son las funciones o procedimientos que definen el comportamiento de un objeto.

Encapsulamiento:

Es el principio de agrupar datos (atributos) y métodos que operan sobre esos datos en una sola unidad, la clase. En Java, se logra mediante modificadores de acceso como `private`, `protected` y `public`.

Herencia:

Es un mecanismo que permite crear nuevas clases (subclases) a partir de clases existentes (superclases). La subclase hereda atributos y métodos de la superclase, promoviendo la reutilización del código. En Java, se usa la palabra clave `extends` para la herencia.

Polimorfismo:

Permite que diferentes clases se traten como instancias de una misma clase a través de una interfaz común. Esto se logra mediante el uso de métodos sobrecargados y sobrescritos. En Java, se implementa con interfaces y la palabra clave `implements`.

Abstracción:

Es el proceso de simplificar sistemas complejos ocultando los detalles innecesarios y exponiendo solo la funcionalidad esencial. En Java, se logra utilizando clases abstractas (palabra clave `abstract`) e interfaces.

Ventajas de la POO en Java

Modularidad: Los programas se dividen en módulos o clases, lo que facilita la lectura, el mantenimiento y la actualización del código.

Reutilización de Código: Las clases y métodos pueden reutilizarse en diferentes partes del programa o en diferentes proyectos.

Facilidad de Mantenimiento: Los errores y cambios se pueden manejar más fácilmente debido a la estructura modular del código.

Claridad y Comprensibilidad: El código orientado a objetos tiende a ser más intuitivo y fácil de entender, especialmente para representar sistemas complejos.

Índice

Portada.....	1
Competencia.....	2
Introducción.....	3
Índice.....	6
Practica 1.....	7
Practica 2.....	9
Practica 3.....	11
Practica 4.....	13
Practica 5.....	15
Practica 6.....	18
Practica 7.....	20
Practica 8.....	22
Practica 9.....	24
Practica 10.....	25
Practica 11.....	26
Practica 12.....	27
Practica 13.....	28
Practica 14.....	29
Practica 15.....	30
Practica 16.....	32
Conclusión.....	34

Práctica no. 1

Doctor

En esta práctica se declaran los atributos y los métodos que son el ID, el nombre y la especialidad, se crea un método para mostrar el nombre en la clase main principal, y en la clase main principal se declara el nombre del doctor y se muestra el nombre de dicho doctor.

```
1 package com.mycompany.doctor;
2
3 public class Doctor {
4     // atributos
5     int id;
6     String name;
7     String speciality;
8
9     // comportamientos-metodos
10
11     public void showName(){ //nombre del objeto es showName
12         System.out.println(name); // funcion para mostrar el nombre
13     }
14 }
15
```

```
1 package com.mycompany.doctor;
2
3 class Main { //clase main
4
5     public static void main (String[] args){ // metodo main principal
6
7         Doctor myDoctor = new Doctor(); //creacion de un objeto cuyo nombre es my Doctor
8
9         myDoctor.name = "Alejandro Perez"; // asigna el valor de alejandro perez al nombre
10        myDoctor.showName( ); // llama al comportamiento showName
11    }
12
13 }
14
```

Output - Run (Main)

```
-----< com.mycompany:Doctor >-----
Building Doctor 1.0-SNAPSHOT
  from pom.xml
-----[ jar ]-----

--- resources:3.3.1:resources (default-resources) @ Doctor ---
skip non existing resourceDirectory D:\Documents\NetBeansProjects\Doctor\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ Doctor ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ Doctor ---
Alejandro Perez

BUILD SUCCESS

Total time: 0.895 s
Finished at: 2024-06-30T13:51:45-06:00
-----
```


Práctica no. 2

Persona

En esta práctica se declaran los atributos de la persona como el nombre, apellido, escolaridad y el año de nacimiento, también se crean dos métodos, uno para mostrar el nombre, y otro para calcular la edad de dicha persona.

En la clase main se crea un objeto llamado "OPersona", ahí mismo se le asigna el nombre de la persona y el año de nacimiento, también se obtiene el nombre y se calcula la edad.

```
1 package com.mycompany.persona;
2
3 public class Persona {
4
5     // atributos-propiedades
6     String Nombre;
7     String Apellido;
8     String Escolaridad;
9     int AñoNacimiento;
10
11     // comportamientos-metodos
12     public void getNombre() { //metodo sin retorno para obtener el nombre
13         System.out.println(Nombre); // se imprime el nombre
14     }
15
16     public void calcularEdad() { //metodo sin retorno para calcular la edad
17         int Edad = 2024 - AñoNacimiento; // se calcula la edad restando los años
18         System.out.println(Edad); // se imprime el resultado de la resta
19     }
20 }
```

```
1 package com.mycompany.persona;
2
3 class Main {
4
5     public static void main(String[] args) {
6         Persona OPersona = new Persona(); //creacion de un objeto cuyo nombre es Opersona
7
8         OPersona.Nombre = "Kleyder"; // se le asigna un nombre al objeto persona
9         OPersona.AñoNacimiento = 2005; // se le asigna un nombre al año de nacimiento
10
11         OPersona.getNombre(); // llama al comportamiento getNombre
12         OPersona.calcularEdad(); // llama al comportamiento calcularEdad
13     }
14
15 }
```

Output - Run (Main)

```
from pom.xml
-----[ jar ]-----
--- resources:3.3.1:resources (default-resources) @ Persona ---
skip non existing resourceDirectory D:\Documents\NetBeansProjects\Persona\src\main\resources
--- compiler:3.11.0:compile (default-compile) @ Persona ---
Changes detected - recompiling the module! :source
Compiling 2 source files with javac [debug target 21] to target\classes
--- exec:3.1.0:exec (default-cli) @ Persona ---
Kleyder
19
-----
BUILD SUCCESS
-----
Total time: 1.461 s
Finished at: 2024-06-30T16:04:17-06:00
-----
```

Práctica no. 3

Estudiante

En este código se declaran los atributos que son el nombre, edad y la carrera, después se usan métodos set y get para asignar los valores correspondientes a los datos ingresado, después en el método main principal se le piden al usuario los datos anteriormente mencionados y mediante la librería “Scanner” se obtienen los datos ingresados por el usuario y al final se imprimen los datos ingresados.

```
verano > Estudiante > J Estudiante.java > Estudiante > setEdad(int)
1  package Estudiante;
2
3  class Estudiante{
4
5      // Atributos - Propiedades
6      private String Nombre;
7      private int Edad;
8      private String Carrera;
9
10     public void setNombre(String Nombre){ // Metodo para asignar el nombre
11         this.Nombre = Nombre;
12     }
13
14     public String getNombre(){ // Metodo para obtener el nombre
15         return Nombre;
16     }
17
18     public void setEdad(int Edad){ // Metodo para asignar la edad
19         this.Edad = Edad;
20     }
21
22     public int getEdad(){ // Metodo para obtener la edad
23         return Edad;
24     }
25
26     public void setCarrera(String Carrera){ // Metodo para asignar la carrera
27         this.Carrera = Carrera;
28     }
29
30     public String getCarrera(){ // Metodo para obtener la carrera
31         return Carrera;
32     }
33
34 }
```

```

verano > Estudiante > J Main.java > Main
1 package Estudiante;
2
3 import java.util.Scanner; // importamos la libreria escaner
4
5 public class Main {
6     public static void main(String[] args) { // Declaracion de la clase principal
7         Scanner entrada = new Scanner(System.in); // Inicializar el escaner Resource leak: 'entrada' is never closed
8         Estudiante OEstudiante = new Estudiante(); // Creacion del objeto OEstudiante a partir de la clase Estudiante
9
10        System.out.println(x:"ingresa tu nombre"); // Entrada del primer dato
11        String Nombre = entrada.nextLine(); // si el dato es String se usa nextLine
12        OEstudiante.setNombre(Nombre); //se le asigna el valor al nombre
13
14        System.out.println(x:"Ingresa tu edad"); // Entrada del segundo dato
15        int Edad = entrada.nextInt(); // si el dato es int se usa nextInt
16        OEstudiante.setEdad(Edad); // se le asigna el valor a la edad
17
18        // Añadimos esta línea para consumir el carácter de nueva línea dejado por nextInt()
19        entrada.nextLine();
20
21        System.out.println(x:"Ingresa la carrera"); // Entrada al tercer dato
22        String Carrera = entrada.nextLine(); // si el dato es String se usa nextLine
23        OEstudiante.setCarrera(Carrera); // se le asigna el valor a la carrera
24
25        // Se imprimen los datos
26        System.out.println("El nombre de la persona es: " + OEstudiante.getNombre() + " Su edad es de: "
27        + OEstudiante.getEdad() + " años " + " Su carrera es: " + OEstudiante.getCarrera());
28    }
29
30

```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS D:\Documents\Visual Studio Code>
PS D:\Documents\Visual Studio Code> d:; cd 'd:\Documents\Visual Studio Code'; &
ExceptionMessages' '-cp' 'C:\Users\Kleyder\AppData\Roaming\Code\User\workspaceSto
dio Code_91cb6507\bin' 'Estudiante.Main'
ingresa tu nombre
Kleyder
Ingresa tu edad
19
Ingresa la carrera
TICS
El nombre de la persona es: Kleyder Su edad es de: 19 años Su carrera es: TICS
PS D:\Documents\Visual Studio Code>

```

Práctica no. 4

Doctor 2.0

En este código se implementaron más cosas, entre ellas los métodos get y set para asignar el tipo de dato ingresado, y en el método main principal, se agregó la librería scanner para que el usuario ingrese los datos que en este caso son el ID, el nombre y la especialidad, ahí mismo se crea el objeto ODoctor y al final se imprimen todos los datos.

```
verano > Doctor > J Doctor.java > Doctor
1  package Doctor;
2
3  class Doctor {
4
5      // Atributos - Propiedades
6
7      int ID;
8      String Nombre;
9      String Especialidad;
10
11     public void setID(int ID){ // Metodo para asignar el ID
12         this.ID = ID;
13     }
14
15     public int getID(){ // Metodo para obtener el ID
16         return ID;
17     }
18
19     public void setNombre(String Nombre){ // Metodo para asignar el nombre
20         this.Nombre = Nombre;
21     }
22
23     public String getNombre(){ // Metodo para obtener el nombre
24         return Nombre;
25     }
26
27     public void setEspecialidad(String Especialidad){ // Metodo para asignar la especialidad
28         this.Especialidad = Especialidad;
29     }
30
31     public String getEspecialidad(){ // Metodo para obtener la especialidad
32         return Especialidad;
33     }
34 }
```

```

verano > Doctor > J Main.java > Main > main(String[])
1  package Doctor;
2
3  import java.util.Scanner; // importamos la libreria escaner
4
5  public class Main{
6      Run | Debug
      public static void main(String[] args) { // Declaracion de la clase principal
7
8          Scanner entrada = new Scanner(System.in); // Inicializar el escanner Resource leak: 'entrada' is never closed
9          Doctor ODoctor = new Doctor(); // creamos el objeto "ODoctor"
10
11          System.out.println(x:"ingresa el ID del doctor"); // Entrada del primer dato
12          int ID = entrada.nextInt(); // si el dato es String se usa nextLine si no, se usa nextInt
13          ODoctor.setID(ID); //se le asigna el valor al ID
14
15          // Añadimos esta línea para consumir el carácter de nueva línea dejado por nextInt()
16          entrada.nextLine();
17
18          System.out.println(x:"ingresa el Nombre del doctor"); // Entrada del segundo dato
19          String Nombre = entrada.nextLine(); // si el dato es String se usa nextLine si no, se usa nextInt
20          ODoctor.setNombre(Nombre); //se le asigna el valor al nombre
21
22          System.out.println(x:"ingresa la especialidad del doctor"); // Entrada del tercer dato
23          String Especialidad = entrada.nextLine(); // si el dato es String se usa nextLine si no, se usa nextInt
24          ODoctor.setEspecialidad(Especialidad); //se le asigna el valor a la especialidad
25
26          // Se imprimen los datos
27          System.out.println("El ID del doctor es: " + ODoctor.getID() + " Su nombre es: " +
28          ODoctor.getNombre() + " Su especialidad es: " + ODoctor.getEspecialidad());
29      }
30  }

```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS D:\Documents\Visual Studio Code> ^C
PS D:\Documents\Visual Studio Code>
PS D:\Documents\Visual Studio Code> d:; cd 'd:\Documents\Visual Studio Code'; &
ExceptionMessages' '-cp' 'C:\Users\Kleyder\AppData\Roaming\Code\User\workspaceSt
dio Code_91cb6507\bin' 'Doctor.Main'
ingresa el ID del doctor
230110626
ingresa el Nombre del doctor
Juan
ingresa la especialidad del doctor
Cirujano
El ID del doctor es: 230110626 Su nombre es: Juan Su especialidad es: Cirujano
PS D:\Documents\Visual Studio Code>

```

Práctica no. 5

Persona 2.0

En esta práctica se hizo lo mismo que la práctica “Doctor 2.0” cambiando ciertos datos, que en este caso se usan los datos de nombre, apellido, escolaridad y año de nacimiento con su respectivo método set y get; en el método main principal se usa la librería scanner para la entrada de datos que el usuario ingrese, se crea el objeto Opersona y se imprimen los datos que se ingresaron.

```
verano > Persona > Persona.java > Persona > setNombre(String)
1  package Persona;
2
3  class Persona {
4
5      // atributos-propiedades
6      String Nombre;
7      String Apellido;
8      String Escolaridad;
9      int AñoNacimiento;
10
11     // comportamientos-metodos
12
13     public void setNombre (String Nombre){ // Metodo para asignar el nombre
14         this.Nombre = Nombre;
15     }
16
17     public String getNombre(){ // Metodo para obtener el nombre
18         return Nombre;
19     }
20
21     public void setApellido(String Apellido){ // Metodo para asignar el apellido
22         this.Apellido = Apellido;
23     }
24
25     public String getApellido(){ // Metodo para obtener el apellido
26         return Apellido;
27     }
28
29     public void setEscolaridad(String Escolaridad){ // Metodo para asignar la escolaridad
30         this.Escolaridad = Escolaridad;
31     }
32
33     public String getEscolaridad(){ // Metodo para obtener la escolaridad
34         return Escolaridad;
35     }
}
```

```
3 class Persona {
13     public void setNombre (String Nombre){ // Metodo para asignar el nombre
14         this.Nombre = Nombre;
15     }
16
17     public String getNombre(){ // Metodo para obtener el nombre
18         return Nombre;
19     }
20
21     public void setApellido(String Apellido){ // Metodo para asignar el apellido
22         this.Apellido = Apellido;
23     }
24
25     public String getApellido(){ // Metodo para obtener el apellido
26         return Apellido;
27     }
28
29     public void setEscolaridad(String Escolaridad){ // Metodo para asignar la escolaridad
30         this.Escolaridad = Escolaridad;
31     }
32
33     public String getEscolaridad(){ // Metodo para obtener la escolaridad
34         return Escolaridad;
35     }
36
37     public void setAñoNacimiento(int AñoNacimiento){ // Metodo para asignar el año de nacimiento
38         this.AñoNacimiento = AñoNacimiento;
39     }
40
41     public int getAñoNacimiento(){ // Metodo para obtener el año de nacimiento
42         return AñoNacimiento;
43     }
44
45 }
```

```
verano > Persona > J Main.java > Main > main(String[])
1 package Persona;
2
3 import java.util.Scanner; // importamos la libreria escaner
4
5 public class Main{
6     public static void main(String[] args) { // Declaracion de la clase principal
7
8         Scanner entrada = new Scanner(System.in); // Inicializar el escanner Resource leak: 'entrada' is never closed
9         Persona OPersona = new Persona(); // creamos el objeto "OPersona"
10
11         System.out.println(x:"Ingresa tu nombre"); // Entrada del primer dato
12         String Nombre = entrada.nextLine(); // si el dato es String se usa nextLine si no, se usa nextInt
13         OPersona.setNombre(Nombre); //se le asigna el valor al Nombre
14
15         System.out.println(x:"Ingresa tu apellido"); // Entrada del segundo dato
16         String Apellido = entrada.nextLine(); // si el dato es String se usa nextLine si no, se usa nextInt
17         OPersona.setApellido(Apellido); //se le asigna el valor al Apellido
18
19         System.out.println(x:"Ingresa la escolaridad"); // Entrada del tercer dato
20         String Escolaridad = entrada.nextLine(); // si el dato es String se usa nextLine si no, se usa nextInt
21         OPersona.setEscolaridad(Escolaridad); //se le asigna el valor a la escolaridad
22
23         System.out.println(x:"Ingresa tu año de nacimiento"); // Entrada del cuarto dato
24         int AñoNacimiento = entrada.nextInt(); // si el dato es String se usa nextLine si no, se usa nextInt
25         OPersona.setAñoNacimiento(AñoNacimiento); //se le asigna el valor al AñoNacimiento
26
27         // Añadimos esta línea para consumir el carácter de nueva línea dejado por nextInt()
28         entrada.nextLine();
29
30         // Se imprimen los datos
31         System.out.println("Tu nombre es: " + OPersona.getNombre() + " y tu apellido es: " + OPersona.getApellido() +
32         " tu escolaridad es: " + OPersona.getEscolaridad() + " y tu año de nacimiento es: " + OPersona.getAñoNacimiento());
33     }
34 }
```


PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\Documents\Visual Studio Code> d:; cd 'd:\Documents\Visual Studio Code'; & 'C:\Program Files\Java\jdk-17\
ExceptionMessages' '-cp' 'C:\Users\Kleyder\AppData\Roaming\Code\User\workspaceStorage\2c72c63e25dd056c3b0833772
dio Code_91cb6507\bin' 'Persona.Main'
Ingresa tu nombre
Kleyder
Ingresa tu apellido
Murillo
Ingresa la escolaridad
Universidad
Ingresa tu año de nacimiento
2005
Tu nombre es: Kleyder y tu apellido es: Murillo tu escolaridad es: Universidad y tu año de nacimiento es: 2005
PS D:\Documents\Visual Studio Code>
```

Práctica no. 6

Día Anyo

En esta práctica se le pide al usuario su fecha de nacimiento (día y año) y también se le pide la fecha actual (día y año) estos datos se inicializan con un constructor y se usa un método tipo booleano que compara ambas fechas para ver si las fechas son las mismas, de ser así, imprime un mensaje de feliz cumpleaños, si no, solo imprime un mensaje de “feliz día”.

```
verano > DiaAnyo > J DiaAnyo.java > DiaAnyo > visualizar()
1  package DiaAnyo;
2
3  class DiaAnyo {
4
5      // atributos-propiedades
6      private int mes;
7      private int dia;
8
9      public DiaAnyo(int d, int m){ // constructor
10
11          dia = d; //se declara el dia con la letra d
12          mes = m; // se declara el mes con la letra m
13
14      }
15
16      public boolean igual (DiaAnyo d){ // metodo tipo booleano
17          if ((dia == d.dia) && (mes == d.mes)){ // compara si el dia y el mes ingresados son los mismos
18              return true;
19          }
20          else{
21              return false;
22          }
23      }
24
25      public void visualizar(){ // metodo sin retorno
26          System.out.println("mes = " + mes + " , dia = " + dia); // se imprimen los dias y meses ingresados
27      }
28
29  }
```

```

verano > DiaAnyo > J Main.java > Main > main(String[])
1  package DiaAnyo;
2
3  import java.io.IOException; // se importan librerias de excepcion
4  import java.util.Scanner; // se importa la libreria scanner
5
6  public class Main {
    Run | Debug
7      public static void main(String[] ar) throws IOException { // metodo main principal
8
9          DiaAnyo hoy;
10         DiaAnyo cumpleanyos;
11
12         int d, m; // se declaran los tipo de dato
13
14         Scanner entrada = new Scanner(System.in); Resource leak: 'entrada' is never closed
15         System.out.print(s:"Introduzca fecha de hoy, dia: ");
16         d = entrada.nextInt(); // entrada de datos
17         System.out.print(s:"Introduzca el número de mes: ");
18         m = entrada.nextInt(); // entrada de datos
19         hoy = new DiaAnyo(d,m);
20         System.out.print(s:"Introduzca su fecha de nacimiento, dia: ");
21         d = entrada.nextInt(); // entrada de datos
22         System.out.print(s:"Introduzca el número de mes: ");
23         m = entrada.nextInt(); // entrada de datos
24         cumpleanyos = new DiaAnyo(d,m);
25         System.out.print( s:" La fecha de hoy es ");
26         hoy.visualizar(); // se llama al metodo hoy.visualizar
27         System.out.print( s:" Su fecha de nacimiento es ");
28         cumpleanyos.visualizar();
29         if (hoy.igual(cumpleanyos)){ // se comparan las fechas para saber si son iguales
30             System.out.println( x:"¡Feliz cumpleaños! ");
31         } else {
32             System.out.println( x:"¡Feliz dia! ");
33         }
34     }

```

```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL
PS D:\Documents\Visual Studio Code> ^C
PS D:\Documents\Visual Studio Code>
PS D:\Documents\Visual Studio Code> d:; cd 'd:\
ExceptionMessages' '-cp' 'C:\Users\Kleyder\AppData
dio Code_91cb6507\bin' 'DiaAnyo.Main'
Introduzca fecha de hoy, dia: 30
Introduzca el número de mes: 06
Introduzca su fecha de nacimiento, dia: 10
Introduzca el número de mes: 05
La fecha de hoy es mes = 6 , dia = 30
Su fecha de nacimiento es mes = 5 , dia = 10
¡Feliz dia!
PS D:\Documents\Visual Studio Code>

```

Práctica no. 7

Cumpleaños

En esta práctica se obtiene la fecha actual para poder determinar la edad de la persona, para esto se necesita una librería para obtener el año actual y así poder hacer la resta, también se muestra el nombre de esta persona.

```
verano > Cumpleaños > J Cumpleaños.java > ...
1 package Cumpleaños;
2
3 import java.time.LocalDate; // importamos la libreria para obtener la fecha y hora actuales
4
5 class Cumpleaños {
6
7     // atributos-propiedades
8     String Nombre;
9     String Apellido;
10    String Escolaridad;
11    int AñoNacimiento;
12
13    // comportamientos-metodos
14    public void getNombre() { // metodo sin retorno para imprimir el nombre
15        System.out.println(Nombre);
16    }
17
18    public void calcularEdad() { // metodo sin retorno
19        LocalDate fecha_actual = LocalDate.now(); // se obtiene la fecha actual
20        int año_actual = fecha_actual.getYear(); // de la fecha actual se obtiene el año
21        int edad = año_actual - AñoNacimiento; // se resta el año actual con la edad
22        System.out.println(edad); // se imprime el resultado
23    }
24 }
```

verano > Cumpleaños > J Main.java > ...

```
1 package Cumpleaños;
2
3 public class Main {
4
5     Run | Debug
6     public static void main(String[] args) { // clase main principal
7         Cumpleaños OCumpleaños = new Cumpleaños(); // se crea el objeto OCumpleaños
8
9         OCumpleaños.Nombre = "Kleyder"; // se asigna el nombre al objeto
10        OCumpleaños.AñoNacimiento = 2005; // se asigna el año de nacimiento al objeto
11
12        OCumpleaños.getNombre(); // se obtiene el nombre
13        OCumpleaños.calcularEdad(); // se obtiene la edad de la persona
14    }
15 }
16
```

PROBLEMS

6

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

```
PS D:\Documents\Visual Studio Code> & 'C:\Program File
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be4
Kleyder
19
PS D:\Documents\Visual Studio Code>
```

Práctica no. 8

Libro

En esta práctica se crean objetos para poder registrar los datos de diferentes libros, se crean los métodos de set y get para obtener los datos de dichos libros, y al final se imprimen los datos de los libros registrados.

```
verano > Libro > Libro.java > Libro > Libro(int, String, String, int)
1  package Libro;
2
3  class Libro {
4
5      // atributos - propiedades
6
7      int ISBN;
8      String Titulo;
9      String Autor;
10     int NumPaginas;
11
12     public Libro (int ID, String T, String A, int P){ /* constructor que declara que
13         tipo de dato se usa y que variable se le asigna */
14         ISBN = ID;
15         Titulo = T;
16         Autor = A;
17         NumPaginas = P;
18     }
19
20     public void setISBN(int ISBN){ // metodo para asignar el ISBN
21         this.ISBN = ISBN;
22     }
23
24     public int getISBN (){ // metodo para obtener el ISBN
25         return ISBN;
26     }
27
28     public void setTitulo(String Titulo){ // metodo para asignar el titulo
29         this.Titulo = Titulo;
30     }
31
32     public String getTitulo(){ // metodo para obtener el titulo
33         return Titulo;
34     }
35 }
```

```

verano > Libro > Libro.java > Libro > Libro(int, String, String, int)
3  class Libro {
22  }
23
24  public int getISBN () { // metodo para obtener el ISBN
25      return ISBN;
26  }
27
28  public void setTitulo(String Titulo) { // metodo para asignar el titulo
29      this.Titulo = Titulo;
30  }
31
32  public String getTitulo() { // metodo para obtener el titulo
33      return Titulo;
34  }
35
36  public void setAutor(String Autor) { // metodo para asignar el autor
37      this.Autor = Autor;
38  }
39
40  public String getAutor() { // metodo para obtener el autor
41      return Autor;
42  }
43
44  public void setNumPaginas(int NumPaginas) { // metodo para asignar el numero de paginas
45      this.NumPaginas = NumPaginas;
46  }
47
48  public int getNumPaginas() { // metodo para obtener el numero de paginas
49      return NumPaginas;
50  }
51
52  public String toString() { // metodo string para imprimir los datos
53      return "El libro " + Titulo + " con ISBN de: " + ISBN + " creado por " + Autor + " tiene " + NumPaginas + " paginas";
54  }
55  }
  
```

```

verano > Libro > Main.java > Main > main(String[])
1  package Libro;
2
3  public class Main {
4
5      Run | Debug
6      public static void main(String[] args) { // clase main principal
7          Libro OLibro1 = new Libro(ID:19284, T:"Programacion", A:"Leonardo", P:300); // se crea un objeto y se asignan los datos
8          Libro OLibro2 = new Libro(ID:579165, T:"Electricidad y magnetismo", A:"Tesla", P:247); // se crea un objeto y se asignan los datos
9
10         System.out.println(OLibro1.toString()); // se imprimen los datos del objeto 1
11         System.out.println(OLibro2.toString()); // se imprimen los datos del objeto 2
12
13     }
  
```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS D:\Documents\Visual Studio Code> & 'C:\Program Files\Java\jdk-17\bin\java.exe' '-XX:+S
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be4110e\redhat.java\jdt_ws\Visual Stud
El libro Programacion con ISBN de: 19284 creado por Leonardo tiene 300 paginas
El libro Electricidad y magnetismo con ISBN de: 579165 creado por Tesla tiene 247 paginas
PS D:\Documents\Visual Studio Code>
  
```


Práctica no. 10

Cuadrado

En esta práctica se realizó un código para poder sacar el área de un cuadrado, para lograr esto se asigna el valor de uno de los lados del cuadrado y se multiplica por sí mismo.

```
verano > Cuadrado > J Cuadrado.java > IT Cuadrado > U calcularArea()
1  package Cuadrado;
2
3  class Cuadrado{
4      private double Lado; // se le asigna el tipo de dato al lado del cuadrado
5      public Cuadrado(double L){ // constructor que declara el tipo de dato del lado
6          Lado = L;
7      }
8      public double calcularArea(){ // metodo tipo double para calcular la edad
9          return Math.pow(Lado, 2); // operacion matematica para sacar el area del cuadrado
10     }
11 }
```





```
verano > Cuadrado > J Main.java > IT Main > U main(String[])
1  package Cuadrado;
2
3  public class Main{
4      Run | Debug
5      public static void main(String[] args) { // metodo main principal
6          Cuadrado c = new Cuadrado(L:15); // se crea el objeto "c" para asignarle el valor del lado
7          System.out.println(" El área del cuadrado es " + c.calcularArea()); // se imprime el area del cuadrado
8      }
9  }
```






```
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\Documents\Visual Studio Code> & 'C:\Program Files
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be41
El área del cuadrado es 225.0
PS D:\Documents\Visual Studio Code>
```

Práctica no. 11

Triangulo

En esta práctica se realizó un código para poder sacar el área de un triángulo, para lograr esto se necesita saber las medidas de la altura y la base del triángulo para poder sacar su área.

```
verano > Triangulo >  Triangulo.java >  Triangulo >  calcularArea()
1  package Triangulo;
2
3  class Triangulo{
4      private double Base, Altura; // se asigna el tipo de dato de la base y la altura
5      public Triangulo(double B, double A){ // constructor que declara el tipo de dato de la base y la altura
6          Base = B;
7          Altura = A;
8      }
9      public double calcularArea()  // metodo tipo double para calcular el area
10     return (Base*Altura)/2; // operacion matematica para sacar el area del triangulo
11
12 }
```

```
verano > Triangulo >  Main.java >  Main >  main(String[])
1  package Triangulo;
2
3  public class Main{
4      Run | Debug
5      public static void main(String[] args)  // metodo main principal
6       riangulo Ot = new Triangulo(B:51,A:94); // se crea el objeto Ot y despues se le asignan los valores de la altura y la base
7      System.out.println(" El área del Triangulo es " + Ot.calcularArea()); // se imprime el area del triangulo
8  }
```

```
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\Documents\Visual Studio Code> & 'C:\Program File
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be4
El área del Triangulo es 2397.0
PS D:\Documents\Visual Studio Code>
```

Práctica no. 12

Rectángulo

En esta práctica se realizó un código para poder sacar el área de un rectángulo, para lograr esto se necesita saber las medidas de la altura y la base del rectángulo para poder sacar su área.

```
verano > Rectangulo > J Rectangulo.java > Rectangulo > calcularArea()
1  package Rectangulo;
2
3  class Rectangulo
4  {
5      private double base, altura; // se le asigna el tipo de dato a la base y la altura
6      public Rectangulo(double b, double h) // constructor que asigna el tipo de dato a la base y la altura
7      {
8          base = b;
9          altura = h;
10     }
11     public double calcularArea() // metodo double para sacar el area del circulo
12     {
13         return base*altura; // operacion matematica para obtener la base y la altura del rectangulo
14     }
15 }
```

```
verano > Rectangulo > J Main.java > Main > main(String[])
1  package Rectangulo;
2
3  public class Main
4  {
5      Run | Debug
6      public static void main(String ar[]) // metodo main principal
7      {
8          Rectangulo r = new Rectangulo(b:3.0,h:7.0); // se crea el objeto r y se le asigna el valor de la altura y la base
9          System.out.println(" El área es " + r.calcularArea()); // se imprime el area del rectangulo
10     }
11 }
```

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\Documents\Visual Studio Code> & 'C:\Program File
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be4
El área es 21.0
PS D:\Documents\Visual Studio Code>
```

Práctica no. 13

Exception

En este código se usa una excepción aritmética para saber cómo funcionan dichas excepciones.

```
verano > Excepcion > J Excepcion.java > Excepcion > main(String[])  
1 package Excepcion;  
2  
3 class Excepcion{  
    Run | Debug  
4     public static void main(String[] args) {  
5         try{  
6             // codigo que puede generar una excepcion  
7             int resultado = 10/0; // esto generará una excepción de division  
8         } catch(ArithmeticException e){  
9             // captura de la excepcion especifica  
10            System.out.println("Ocurrio una excepcion: " + e.getMessage());  
11        }  
12    }  
13 }
```

```
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS  
  
PS D:\Documents\Visual Studio Code> & 'C:\Program Files  
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be43  
Ocurrio una excepcion: / by zero  
PS D:\Documents\Visual Studio Code>
```

Práctica no. 14

Exception Array

En esta práctica podemos ver que son las excepciones y cómo funcionan, para este ejemplo se hace un arreglo de 4 caracteres y se intenta generar otro de 5 caracteres generando un error, este error se imprime para saber cómo es que este funciona.

```
verano > ExceptionArray > Exceptionarray.java > Exceptionarray
1 package ExceptionArray;
2
3 public class Exceptionarray {
4     Run | Debug
5     public static void main(String[] args) {
6         try{
7             int [] arreglo = new int[4]; // se crea un arreglo de 4 posiciones
8             System.out.println(x:"intentando ingresar a un arreglo");
9             // genera excepcion tipo ArrayIndexOutOfBoundsException
10            int valor = arreglo[5]; /* se crea un arreglo de 5 posiciones generando un error ya que el arreglo creado es de 4 posiciones, no de 5*/
11        }
12        catch(ArrayIndexOutOfBoundsException e){
13            System.out.println("se a producido un error de tipo " + e.getMessage()); // se obtiene el mensaje de error
14        }
15        finally{
16            System.out.println(x:"el bloque finally se ejecutara siempre exista una excepción o no"); // esta linea se ejecuta siempre
17        }
18    }
19
20 }
```

```
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\Documents\Visual Studio Code> & 'C:\Program Files\Java\jdk-1
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be4110e\redhat.
intentando ingresar a un arreglo
se a producido un error de tipo Index 5 out of bounds for length 4
el bloque finally se ejecutara siempre exista una excepción o no
PS D:\Documents\Visual Studio Code>
```

Práctica no. 15

Comparación numero

En esta práctica se le pide al usuario ingresar un número y el programa debe de ver si este número ingresado por el usuario es mayor o menor a 0.

```
verano > ComparacionNumero > J ComparacionNumero.java > ComparacionNumero
1 package ComparacionNumero;
2
3 public class ComparacionNumero { // Nombre de la clase
4
5     public int numero; //Atributo de la clase
6
7     public ComparacionNumero(int n){//Metodo constructor
8         this.numero = n;
9     }
10
11     public void Imprimir(){ //Metodo que compara un numero y imprime una leyenda
12         if (numero > 0) { // Se compara si el valor del numero es mayor a cero
13             System.out.println(x:"EL NUMERO ES MAYOR QUE CERO");//Si la condicion se cumple, imprime esta leyenda
14         }
15         else {
16             System.out.println(x:"EL NUMERO ES MENOR QUE CERO");//Si la condicion no se cumple, imprime esta leyenda
17         }
18     }
19 }
```

```
verano > ComparacionNumero > J Main.java > Main
1 package ComparacionNumero;
2
3 import java.util.Scanner;//Importacion de la clase Scanner
4
5 public class Main {
6     Run | Debug
7     public static void main(String[]args){//Argumento por defecto de la clase principal
8         Scanner entrada = new Scanner(System.in);//Declarar objeto Scanner Resource leak: 'entrada' is never
9
10         System.out.println(x:"INGRESA UN NUMERO: ");//Imprimir instrucciones al usuario
11         int num = entrada.nextInt();//Guardar valor introducido por el usuario en variable
12         ComparacionNumero ONum = new ComparacionNumero(num); //Creacion del objeto a partir de la clase molde
13         ONum.Imprimir();//LLamar el metodo que hace la comparacion del numero y imprime una leyenda
14     }
15 }
```

```
PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\Documents\Visual Studio Code> & 'C:\Program Files\
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be43
INGRESA UN NUMERO:
1205
EL NUMERO ES MAYOR QUE CERO
PS D:\Documents\Visual Studio Code> 
```

Práctica no. 16

Números divisibles

En esta práctica se comprueba si un número es divisible por otro número, esto se comprueba si dividiendo el número 1 y el número 2 se obtiene como resultado 0, significa que estos números si son divisibles entre sí.

```
verano > NumerosDivisibles > J NumerosDivisibles.java > NumerosDivisibles > ComproDivi()
1  package NumerosDivisibles;
2
3  public class NumerosDivisibles {
4
5      private int numero1, numero2 ;
6      // La clase tiene dos atributos privados, numero1 y numero2, que se utilizarán para almacenar los números que se van a verificar
7
8      public NumerosDivisibles (int num1, int num2){
9          //El constructor de la clase recibe dos parámetros (num1 y num2) y los asigna a los atributos numero1 y numero2, respectivamente
10         this.numero1 = num1;
11         this.numero2 = num2;
12     }
13     public void ComproDivi(){
14         /*verifica si numero1 es divisible por numero2 usando el operador de
15         módulo (%). Si numero1 % numero2 es igual a 0*/
16         if (numero1 % numero2 == 0) {
17             System.out.println(x:"VERDADERO");
18         }
19         else{
20             System.out.println(x:"FALSO");
21         }
22     }
23 }
24 }
```

```
verano > NumerosDivisibles > J Main.java > Main > main(String[])
1  package NumerosDivisibles;
2
3  import java.util.Scanner; // se importa la libreria scanner
4
5  public class Main {
6      Run | Debug
7      public static void main(String[]args){ // metodo main principal
8          Scanner entrada = new Scanner(System.in); Resource leak: 'entrada' is never closed
9          int num1 , num2;
10
11          System.out.println(x:"INGRESA EL PRIMER NUMERO: "); // se le pide al usuario el primer numero
12          num1 = entrada.nextInt(); // entrada del primer numero
13          System.out.println(x:"INGRESA EL SEGUNDO NUMERO: "); // se le pide al usuario el segundo numero
14          num2 = entrada.nextInt(); // entrada del segundo numero
15
16          NumerosDivisibles ONums = new NumerosDivisibles(num1,num2); // se crea el objeto Onums
17          ONums.ComproDivi(); // se comprueba si el numero es divisible y se imprime el resultado
18
19
20
21 }
```



```
PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\Documents\Visual Studio Code> & 'C:\Program Files
Code\User\workspaceStorage\2c72c63e25dd056c3b0833772be4
INGRESA EL PRIMER NUMERO:
60
INGRESA EL SEGUNDO NUMERO:
5
VERDADERO
PS D:\Documents\Visual Studio Code> |
```

Conclusión

A lo largo de estas prácticas me he dado cuenta que la programación orientada a objetos es muy útil al momento de realizar tareas específicas, ya que se puede programar en bloques, esto facilita mucho la programación, la organización y se controlan mejor los errores que puedan llegar a suceder en el programa, también he aprendido que java tiene muchas librerías útiles para que el programador no tenga que estar instalando las librerías por sí mismo o incluso, programarlas el mismo.

Java es un lenguaje muy amplio, y tiene muchas herramientas que ayudan al programador a no programar todo en una sola línea de código, esto es muy útil para programar.