

# CLASE 01

---

30-04-2024

## Comandos básicos en Visual Estudio Code

Todos los comandos a continuacion son ejecutables en windows:

- **CTRL + SHIFT + P**: Abre la paleta de comandos.
  - **CTRL + P**: Abre el buscador de archivos.
  - **CTRL + B**: Abre y cierra la barra lateral.
  - **CTRL + D**: Activa el multi-cursor.
  - **SHIFT + ALT + UP / SHIFT + ALT + DOWN**: Seleccionar toda la linea de codigo.
  - **SHIFT + ALT + A**: Comentar varias lineas de codigo.
  - **CTRL + K + C**: Comentar una linea de codigo.
  - **ALT + ---->**: Retroceder o avanzar.
  - **CTRL + T**: Mostrar comandos.
  - **CTRL + ESPACIO**: Sugerencias de activacion.
- 

# CLASE 02

---

01-05-2024

## Markdown

Es un lenguaje de marcado, su extension es ".md", se usara para anotar los apuntes de la clases y se puede transformar a pdf. Tiene varias funcionalidades para resaltar de diversas formas las partes del texto.

Titulo #                      ##                      ###                      ####                      #####

**\*\*palabras en negrita\*\***

*\*palabras en cursiva\**

***\*\*\*palabras en negrita y cursiva\*\*\****

==Texto resaltado==

~~Texto tachado~~

## Colocar imagenes

![Nombre alternativo] (ruta de acceso de la imagen)

## Colocar lineas de codigo

```
int main (){  
    cout << "Esto es un ejemplo"<<endl;  
}
```

## GIT

Herramienta con la cual podremos subir nuestros codigos a repositorios en linea, que facilita el compartido de los mismos, a continuacion los comandos mas usados:

### Identidad del usuario

```
git config --global user.name "pat_mic"
```

```
git config --global user.email pat_mic@hotmail.com
```

```
git config user.email
```

```
git config user.name
```

### Control de versiones

```
git init
```

```
git status
```

```
git add .
```

```
git add NombreCarpeta/NombreArchivo.ext
```

```
git commit -m 'mensaje : initial project version'
```

## Control de versiones - clonando

```
git clone https://github.com/xyyyy/abc
```

```
$ git clone https://github.com/xyyyy/abc miPropioNombre  
Touch (crear archivos)
```

## Quitar archivos del control

```
echo "nombreArchivo.ext" >> .gitignore
```

```
echo "*.txt" >> .gitignore
```

## Ver archivos ignorados

```
cat .gitignore
```

## Ver archivos ignorados

```
git add -f NombreArchivo.log
```

## Crear y clonar

```
git init
```

```
git clone /ruta del repositorio
```

# CLASE 03

06/05/2024

## JAVA

- Origen

### Como funciona:

1. Codigo
2. Compilacion
3. Bytecode
4. JVM
5. Multiplataforma.

### Sabores de JAVA

- Java Micro Edition.
- Java Standar Edition.
- Java enterprise Edition.

### Sintaxis de java

The diagram shows a Java code editor with the following code:

```
1 package team.ed.course;
2 import java.lang.*;
3 public class Person {
4     private String name;
5     public static void main(String args[]){
6         Person friend = new Person();
7         friend.name = "Peter";
8         System.out.println("Hola " +
9             friend.name);
10    }
11 }
```

Annotations and explanations:

- Todos los archivos pertenecen a un paquete** (All files belong to a package) - points to line 1.
- Importa los paquetes para el proyecto** (Imports packages for the project) - points to line 2.
- Java usa clases para ejecutar el código** (Java uses classes to execute the code) - points to line 3.
- Se debe indicar el tipo de dato** (The data type must be indicated) - points to line 5.
- Modificadores de acceso: private, public, protected o por defecto ninguno** (Access modifiers: private, public, protected or by default none) - points to line 4.
- El método principal en Java es el método main** (The main method in Java is the main method) - points to line 5.
- La palabra reservada new crea un objeto del tipo de dato especificado** (The reserved word new creates an object of the specified data type) - points to line 6.
- Se utilizan ; para cada sentencia** (Semicolons are used for each statement) - points to the semicolon at the end of line 11.
- Se usan {} para identificar el bloque de código** (Braces are used to identify the code block) - points to the curly braces at the end of line 11.

## Tipos de lenguaje

- **Compilado:** Convierte el código a binarios que lee el S.O.
- **Interpretado:** Requiere de un programa que lea la instrucción del código en tiempo real.
- **Intermedio:** Se compila el código fuente a un lenguaje intermedio y este último se ejecuta en una máquina virtual.

## Estructurado vs O.O

### Programación estructurada

- La programación estructurada tiene **funciones o procedimientos**.
- La programación estructurada se maneja con **estructuras**.
- La programación estructurada tiene **variables**.

### Programación O.O

- La programación orientada a objetos tiene **métodos**.
- La programación orientada a objetos se maneja con **clases**.
- La programación orientada a objetos tiene **propiedades**.
- Las variables si existen, solo **dentro de los métodos**.

## CLASE 07

---

13-05-2024

### P.O.O

#### 1. Conceptualización

Se refiere a la idea que se quiere realizar, se puede dibujar, prototipar, etc.

- Se necesita ponerle un nombre y concretar un significado.
- A los objetos se les puede atribuir dos cosas: características o acciones:

**Características:** propiedades, se debe poder almacenar, y debe almacenar información (descripción precisa del objeto).

**Acciones:** métodos, son acciones, verbo, las acciones deben estar ligadas a las propiedades. Tienen parámetros (información necesaria para que se cumpla la acción de manera exitosa).

#### **Ámbitos:**

- Public (+)
- Protect (-)

- Friendly (~)
- Protect

## 2. UML

(Lenguaje de modelado unificado) se usan las clases, todo lo conceptualizado se ubica dentro de una clase

### **Clase**

---

- edad: float

+Tipocabello: string | ~ bailar (cancion:string, tiempomin:int, ritmo:string) : string | ~ tocar (Objeto:string, tiempoMin:int): boolean +Saltar (AlturaMts: int, cantidad:int):void

Se puede pedir que las acciones devuelvan un valor, que puede ser:

- void: No regresa nada
- String: que regrese una palabra o dato
- Boolean: regrese un valor verdadero o falso.

**Eventos:** Situaciones en la que interactua el objeto con otros objetos.

## 3.Codigo

```
public class mujer {  
    private float edad;  
    public boolean tieneojos;  
  
    protected bailar (String cancion, TiempoMin int, String ritmo)  
        return "sddsdsds";  
  
}
```