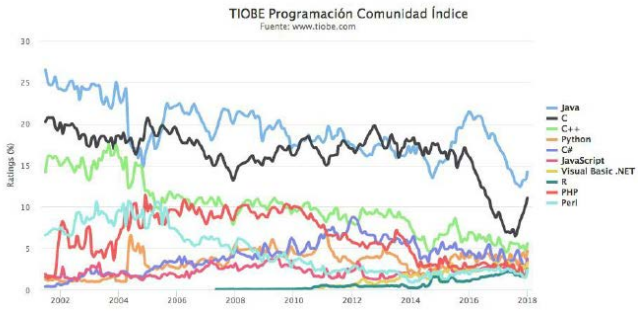
GUIÓN

UNIDAD 3: DESARROLLO DE SOFTWARE.

PROGRAMA INFORMÁTICO Y SU EJECUCIÓN. TIPOS DE SOFTWARE: SO, APLICACIONES Y DRIVERS.

CÓDIGO FUENTE, CÓDIGO OBJETO Y CÓDIGO EJECUTABLE.

LENGUAJES DE PROGRAMACIÓN. CLASIFICACIÓN POR NIVELES – GENERACIONES – PARADIGMA.



INGENIERÍA DE SOFTWARE: CONCEPTOS & CICLO DE VIDA: PROCESOS GENERALES, SOPORTE Y GENERALES.

FASES DE DESARROLLO & MODELOS DE DESARROLLO & REUTILIZACIÓN DE CÓDIGO.

ACTIVIDAD: DIFERENTES CICLOS DE VIDA – VENTAJAS & INCONVENIENTES.

UNIDAD 4: INSTALACIÓN Y USO DE ENTORNOS DE DESARROLLO.

ENTORNOS DE DESARROLLO (IDE): FUNCIONES. EJEMPLOS. CARACTERÍSTICAS DE UN BUEN IDE.

INSTALACIÓN DE UN ENTORNO DE DESARROLLO: ECLIPSE & NETBEANS.

ACTIVIDAD: Instalar Eclipse. IMPORTAR: Gradle, UML Generator y Window Builder.

UNIDAD 5: DISEÑO Y REALIZACIÓN DE PRUEBAS.

PRUEBAS EN EL DESARROLLO DE SOFTWARE. PROCESO.

TIPOS DE PRUEBAS.

HERRAMIENTAS DE TESTEO: JUNIT.

PRUEBAS UNITARIAS CON JUNIT.

ACTIVIDAD: Crear una clase Calculadora y una clase con JUnit para probar las funcionalidades de la clase Calculadora.

UNIDAD6: OPTIMIZACIÓN Y DOCUMENTACIÓN: Optimización, mantenimiento y evolución.

REFACTORIZACIÓN.

Restructuración del código fuente: Cambiar lo escrito por nuevo código, sin cambiar ninguna funcionabilidad. Este cambio solo busca que el código resultante sea mejor, más legible y menos complejo. Resolver errores ocultos y vulnerabilidades en el sistema y simplificación de algoritmos.

“Y, nada más lejos de la realidad, el código para ser óptimo también debe estar bien estructurado, ha de ser fácil de leer y además estar correctamente documentado.”

CONTROL DE VERSIONES. FORMAS. GESTIÓN Y ADMINISTRACIÓN. HERRAMIENTAS & FUNCIONAMIENTO.

TRABAJAR CON **GIT**: INSTALACIÓN, USO DE CONSOLA. USO FRAMEWOK: EGIT (Eclipse) o HERRAMIENTAS DE TERCEROS.

ACTIVIDAD: APLICACIÓN DE REFACTORIZACIÓN. CONTROL DE VERESIONES CON GIT. DOCUMENTAR CON **JAVADOC.**

UNIDAD 7: INTRODUCCIÓN AL LENGUAJE UNIFICADO DE MODELADO (UML).

INTRODUCCIÓN A DIAGRAMAS UML.

DIFERENTES DIAGRAMAS EN FUNCIÓN DE SU UTILIDAD.

+ DIAGRAMADA DE CLASES (UML)

* EXPLICACIÓN DIAGRAMAS DE FLUJO. VISIBILIDAD DE LOS RECURSOS.

ACTIVIDAD: Identificar tipos de diagramas de UML

UNIDAD 8: Elaboración de diagramas de comportamiento y clases

ELABORACIÓN DE DIAGRAMAS DE COMPARTAMIENTO Y CLASES. Casos de uso. Relación entre clases.

DEFINICIÓN DE CLASES: ATRIBUTOS Y MÉTODOS.

RELACIÓN ENTRE CLASES DIVERSAS RELACIONES ENTRE LOS ELEMENTOS.

INGENIERÍA INVERSA Y GENERACIÓN DE CÓDIGO.

ACTIVIDAD: Realizar en un programa de modelado UML el diagrama de clases y casos dado un enunciado.

ACTIVIDAD2: Paso a código. Realiza el diagrama de clases para el programa solicitado y mediante una herramienta CASE de tu elección genera el código automáticamente partiendo del diagrama realizado

FINAL:

El alumno deberá ir creando y completando los diferentes apartados que se soliciten, como si de una memoria de proyecto se tratase.

- Fase de definición: Añadir especificaciones del programa.
- Fase de diseño: UML – diagrama de clases para implementar el algoritmo.
- Crear boceto de diseño.
- Análisis – Diseño: Lista tecnologías, herramientas y elementos hardware requeridos.
- Fase de implementación: Codificación.
- Fase de pruebas: Pruebas Unitarias JUnit.