Universidad Distrital Francisco José de Caldas

Facultad de Ciencias y Educación

# Resumen MADO-17_FPv5-1

Autor: Felipe Vanegas

Docente: Karen De Los Ríos
Asignatura: Algoritmos en C/C++

Bogotá D.C.
16/09/2025

# Índice general

# Capítulo 1

# Introducción

MADO-17_FPv5-1 que trata el diseño de algoritmos y la resolución de problemas, brindando las bases conceptuales requeridas para la capacitación en programación. En este se abordan componentes fundamentales, como el pseudocódigo, las pruebas de escritorio y el ciclo de vida del software. Estas son herramientas cruciales para estructurar, verificar y optimizar soluciones antes de que sean ejecutadas en un lenguaje de programación. La guía, además, presenta ejemplos concretos que colaboran con la comprensión práctica de estos conceptos, mostrando cómo cada uno tiene el potencial de resolver problemas reales.

Asimismo, el documento presenta la idea de repositorio como un lugar fundamental para manejar, guardar y controlar las versiones de proyectos de software. En este contexto, se ofrece una guía práctica para la creación de un repositorio en GitHub, que complementa el aprendizaje teórico con una herramienta que se usa mucho en ámbitos académicos y profesionales.

# Capítulo 2

# Resumen

## 2.1. Control de versiones

Un controlador de versiones es un software que registra los cambios en archivos a lo largo del tiempo. Permite regresar a versiones anteriores, comparar y revertir modificaciones, identificar quién las hizo y proteger contra errores. Facilita el trabajo colaborativo y el respaldo de información, siendo útil no solo para programadores, sino también para diseñadores, escritores y cualquier persona que requiera llevar un control de sus archivos.

### 2.1.1. Tipos:

**- Local:**

El registro de cambios se almacena de forma local (Figura 2.1).

**- Centralizado:**

En estos sistemas se trabaja de manera colaborativa, por lo que un servidor central lleva el control de las versiones y cada usuario descarga los archivos desde ese servidor y sube sus cambios al mismo (Figura 2.2).

**- Distribuido:**

Los usuarios tienen una copia exacta del proyecto, así como todo el registro de las versiones, de esta manera si el servidor remoto falla o se corrompe, los usuarios pueden restablecer el servidor con sus propias copias y obtener los cambios en los directamente del equipo de otros usuarios, favorece evitar perder datos por fallas del sistema (Figura 2.3).

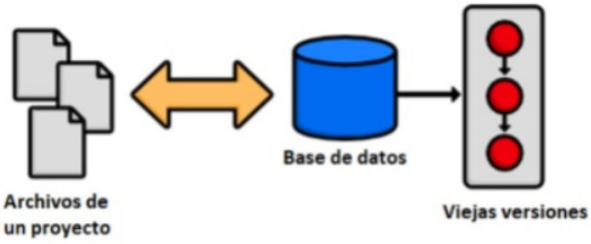


Figura 2.1: Local

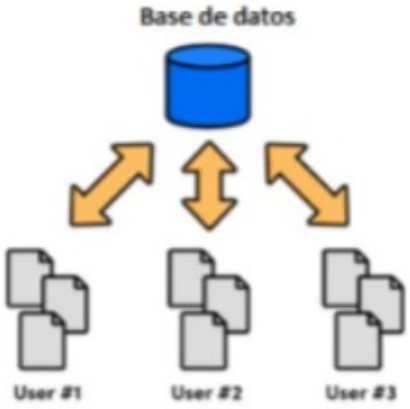


Figura 2.2: Centralizado

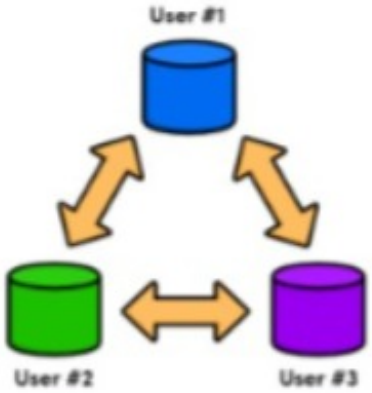


Figura 2.3: Distribuido

### 2.1.2. Repositorio

Es el directorio de trabajo usado para organizar un proyecto, aquí se encuentran todos los archivos que integran nuestro proyecto.

**Local:** Se encuentra en nuestro propio equipo y solo el dueño tiene acceso a él.

**Remoto:** Está alojado en la nube, esto quiere decir, que se encuentra en un servidor externo, el cual puede ser accedido desde Internet y que nos va a permitir tener siempre a la mano nuestros archivos. Algunos ejemplos de este son:

- github.com Es una plataforma de almacenamiento para control de versiones y colaboración. que permite almacenar nuestros repositorios de una forma fácil y rápida, además nos da herramientas para el mejor control del proyecto, posibilidad de agregar colaboradores, notificaciones, herramientas gráficas y mucho más. Actualmente Github es la plataforma más grande de almacenamiento de código en el mundo. Tiene como ventaja su facil manejo y que contiene IA (Inteligencia Artificial) que ayuda con el proceso de creación y/o subida de archivos a nuestros repositorios.

- bitbucket.org

- gitlab.com

## 2.2. Almacenamiento en la nube

### 2.2.1. ¿Qué es?

Es un servicio que permite guardar, administrar y respaldar datos en servidores remotos a los cuales es posible acceder por Internet. Plataformas como Google Drive, OneDrive, iCloud o Dropbox ofrecen este servicio; y algunas incluyen herramientas para crear documentos, hojas de cálculo y presentaciones en línea. Estas aplicaciones permiten editar y compartir archivos en tiempo real con varias personas, facilitando el trabajo en grupo, cuando se necesita colaboración al mismo tiempo con otras personas. Estos documentos pueden verse, editarse, compartirse y descargarse desde cualquier dispositivo con acceso a Internet (celular, computador, tablet, etc...).

### 2.2.2. Ejemplos

- Google Forms

- OneNote

- Dropbox

## 2.3. Buscadores académicos

### 2.3.1. ¿Què es un buscador de internet?

Es una herramienta en línea que permite encontrar información en la web mediante la introducción de palbras clave o consultas. En el ámbito académico, los motores de búsqueda especializados son esenciales para encontrar información que sea confiable, creada por expertos y ademas de facil de usar que sea pertienente y util para todo lo que tiene que ver con el ambito academico (investigaciones, tareas, etc...).

**Ejemplos**

**- Google Scholar (Google Académico):** Es la herramienta de Google, que se especializa en encontrar revistas cientificas, tesis, libros y otros materiales academicos.

**- Microsoft Academic:** Brinda acceso a literatura académica y datos sobre el impactor de las publicaciones.

**- ScienceDirect y SpringerLink:** Repositorio de revistas y libros editoriales.

**- ResearchGate:** Plataforma que reune plataformas de diferentes áreas.

**- BASE (Bielefeld Academic Search Engine):** Es de los buscadores academicos más completos que dispone de biblioecas y repositorios.

**- Repositorio UNAM:** Plataforma que guarda trabajos academicos, tesis, articulos y recursos de investigacion de la UNAM.

## 2.4. Inteligencia Artificial(IA)

### 2.4.1. Definición

La Inteligencia Artificial (IA) ha transformado la forma de manejar y generar información, ya que automatiza procesos, facilita análisis predictivos y optimiza decisiones basadas en grandes volúmenes de datos. En ingeniería, se ha vuelto clave para aumentar la productividad y la innovación. Sus herramientas permiten crear de forma automática texto, imágenes, audio y video, apoyando en tareas como redacción, diseño y generación de ideas. Plataformas como Google Colab, OpenAI y otros servicios en la nube facilitan su integración en proyectos académicos y profesionales.

### 2.4.2. Herramientas de IA

**ChatGPT (OpenAI):** Generación de texto para resúmenes, explicaciones, propuestas, etc... Es la IA más conocida y más usada; en especial en su versión gratuita.

**Ideogram:** Generación de imágenes a partir de un texto.

**Google Docs con complementos de IA:** Ayuda para la redacción y edición de documentos.

**Gamma:** Diseño gráfico asistido por IA para crear presentaciones y gráficos (incluyendo presentaciones tipo Power Point (diapositivas)).

### 2.4.3. Prompt

Básicamente es darle bien las indicaciones a la IA para que te dé bien lo que le pides, desde imágenes a textos, si eres lo más específico posible, le das contexto, especificas el público y le das ejemplos; vas a mejorar el contenido que te dé, se acercara más a lo que pides, o a esa imagen mental que tienes y no tendrás que durar tanto para una tarea.

### 2.4.4. Detección

La IA tiene como contra la pereza y la dependencia, en el caso de los estudiantes, por ejemplo, prefieren decirle a una IA que les haga su trabajo o tarea y no se toman si quiera el tiempo de revisar lo que les dice, el profesor o la persona que revisa cualquier contenido tiene herramientas ("detectores") para saber si algo es generado con IA, aunque no son totalmente efectivos ya que si subes contendido ya existente como algo de la biblia lo puede llegar a detectar como IA. La IA lego para quedarse y lo importante es aprender que es una herramienta que ayuda no una herramienta que te haga todo.

# Capítulo 3

# Pseudocódigo y Pruebas de Escritorio

## 3.1. Definición

### 3.1.1. Pseudocódigo

El pseudocódigo es una herramienta utilizada en la programación para describir de forma clara, ordenada y precisa los pasos de un algoritmo, empleando un lenguaje intermedio entre el lenguaje natural y el lenguaje de programación. Su principal función es facilitar la comprensión y diseño de soluciones antes de codificarlas en un lenguaje específico.

No se rige por una sintaxis estricta como los lenguajes de programación, pero sí mantiene reglas básicas de estructura lógica: instrucciones secuenciales, decisiones (si...entonces), repeticiones (mientras, para) y operaciones con variables. Gracias a su simplicidad, permite que tanto programadores como personas sin experiencia en un lenguaje de programación puedan comprender el funcionamiento de un algoritmo.

### 3.1.2. Pruebas de Escritorio

La prueba de escritorio es un procedimiento utilizado para verificar la validez de un algoritmo antes de implementarlo en un lenguaje de programación. Consiste en simular manualmente la ejecución del algoritmo, paso a paso, utilizando valores de prueba como entrada.

En esta simulación, se construye una tabla donde se registran las variables involucradas, los cálculos realizados y los resultados obtenidos en cada iteración o decisión. De esta manera, se observa el flujo del algoritmo y se pueden detectar inconsistencias, errores lógicos o resultados inesperados.

Su importancia radica en que permite validar la corrección y eficiencia de un algoritmo sin necesidad de recurrir aún a la computadora, lo cual ahorra tiempo y evita errores de programación innecesarios.

## 3.2. Análisis de ejemplos

### 3.2.1. Número positivo o negativo

**Análisis:** Se pide determinar si un número ingresado es positivo o negativo (con la restricción de que no puede ser cero).

**Comentario** Ejemplo sencillo que ilustra cómo identificar datos de entrada (el número real), salida (indicar si es positivo o negativo) y restricciones. Sirve como base para introducir condicionales.

### 3.2.2. Factorial de un número

**Análisis:** Calcular el factorial de un número natural o cero. Se definen variables de control (contador, factorial) y un ciclo repetitivo hasta llegar al resultado.

**Comentario** Ejemplo útil para introducir el concepto de iteración. Además, permite practicar la prueba de escritorio, viendo cómo cambian las variables en cada paso hasta obtener el resultado correcto.

### 3.2.3. Mayor de dos números

**Análisis:** El algoritmo debe comparar dos números distintos y determinar cuál es mayor.

**Comentario** Refuerza el uso de comparaciones y condiciones anidadas. Se observa la importancia de validar que los datos cumplan la restricción de ser diferentes.

# Capítulo 4

# Conclusiones

El estudio de los sistemas de control de versiones, junto con plataformas como GitHub, permitió comprender la importancia de llevar un registro ordenado y colaborativo de los proyectos de software, evitando la pérdida de información y favoreciendo el trabajo en equipo. Asimismo, el almacenamiento en la nube y los buscadores académicos se presentan como recursos indispensables para el ámbito académico y profesional, ya que facilitan el acceso, la consulta y el intercambio de información confiable en cualquier momento y lugar.

Por otro lado, el análisis de la Inteligencia Artificial mostró cómo estas herramientas han transformado la forma de crear y analizar contenido, ofreciendo apoyo en la automatización de procesos, generación de ideas y optimización de decisiones. Sin embargo, también se resaltó la necesidad de utilizarlas con criterio, reconociendo que son un complemento y no un sustituto del trabajo humano.

Finalmente, los conceptos de pseudocódigo y pruebas de escritorio, junto con los ejemplos prácticos desarrollados, resultaron esenciales para consolidar las bases de la programación. Estos elementos permiten estructurar y validar algoritmos antes de implementarlos en un lenguaje específico, garantizando claridad, corrección lógica y eficiencia. En conjunto, los temas abordados en la guía integran teoría y práctica, mostrando que el desarrollo de software exige tanto fundamentos conceptuales como el uso adecuado de herramientas tecnológicas modernas.