Historia de la programación

Jonathan Morales 4-848-21

Breve línea de tiempo

- 1801 – Telar de Jacquard: primeras tarjetas perforadas para dar instrucciones.
- 1843 – Ada Lovelace escribe el primer algoritmo (considerada la primera programadora).
- 1936 – Alan Turing plantea la "máquina universal" (fundamentos de la computación).
- 1950s – Primeros lenguajes de alto nivel: Fortran, COBOL.
- 1970s-80s – Lenguajes estructurados y orientados a objetos: C, Pascal, Smalltalk.
- 1990s-2000s – Expansión de la web: Java, JavaScript, PHP, Python.
- Hoy – Lenguajes modernos: Go, Rust, Kotlin, con foco en seguridad, velocidad y usabilidad.

Concepto de programación

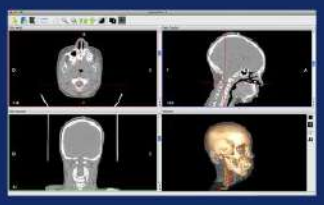
- Proceso de dar instrucciones a una computadora para que realice tareas específicas.
- Se hace a través de lenguajes de programación, que sirven como puente entre el ser humano y la máquina.

Características de la Programación

- Lógica: uso de algoritmos y estructuras.
- Precisión: las instrucciones deben ser exactas.
- Creatividad: permite resolver problemas de múltiples formas.
- Evolutiva: los lenguajes se adaptan a nuevas necesidades.

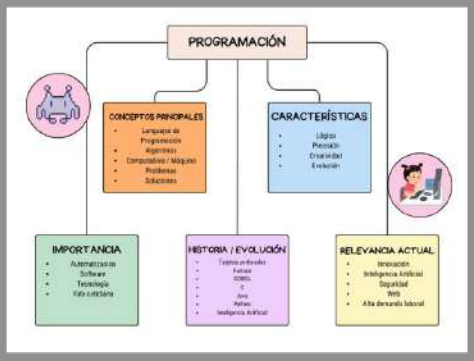
Importancia en Ing. De Sistemas y Computación

- Es la base para el desarrollo software que usamos a diario (apps, redes, juegos, sistemas).
- Permite automatizar procesos, y gestionar casi cualquier cosa imaginable de forma lógica.



Historia
DE LA PROGRAMACIÓN

Jonathan Morales
4-840-21 | 2EG7116



La programación es el proceso de dar instrucciones a las computadoras mediante lenguajes que funcionan como puente entre humanos y máquinas. Su historia inicia con los primeros intentos de automatización, como el telar de Jacquard (1801) y el algoritmo de Ada Lovelace (1843). A lo largo del siglo XX surgieron lenguajes como Fortran, COBOL y C, que dieron paso a paradigmas estructurados y orientados a objetos. En las últimas décadas, lenguajes como Java, Python o Rust han impulsado la expansión de la web, la inteligencia artificial y la seguridad digital. La programación se caracteriza por su precisión, lógica, creatividad y constante evolución, siendo hoy fundamental para la automatización, la tecnología y la vida cotidiana.

¿Qué entiendo por "programación" y qué cosas de mi vida diaria dependen de ella?

¿Por qué la programación sigue siendo relevante en la actualidad y qué impacto puede tener en el futuro cercano?

¿Cuáles han sido los hitos más importantes en la evolución de la programación y cómo han transformado la forma en que usamos la tecnología?

Algoritmo vs Compilación/Ejecución vs Prueba de escritorio vs Codificación

Jonathan Morales
4-840-21 | 2EG7116

	Definición	Relevancia	Ejemplo
Algoritmo	Serie de pasos que concluye con la resolución de un problema	Es importante para resolver un problema en una cantidad finita de pasos, osea más rápido y eficiente.	Instructivos para armar muebles, receta de cocina.
Compilación/Ejecución	La compilación traduce tu código a instrucciones que la computadora entiende (lenguaje máquina) y la ejecución es el proceso de hacer que el programa corra, es decir, que realice esas instrucciones.	Es crucial ya que el compilador transforma lenguaje humano a lenguaje máquina creando un programa ejecutable.	El compilador de PSeint te indica si llevas errores, para que los corrijas, y entonces puedas ejecutar un programa.
Codificación	Proceso de convertir información o datos en un formato específico para diversos propósitos.	En la informática, es fundamental para que las máquinas comprendan y ejecuten instrucciones, impulsando la innovación y la creación de aplicaciones.	Conversión de caracteres alfanuméricos en una representación binaria, como se ve en el código ASCII, donde la letra "A" se convierte en el número 65.
Prueba de escritorio	Una prueba de escritorio, en el contexto de la programación, es un proceso manual para verificar la lógica y el funcionamiento de un algoritmo o programa antes de su ejecución.	Permite identificar errores y fallos lógicos en las etapas tempranas del desarrollo, asegura la correcta funcionalidad del algoritmo antes de su implementación y mejora la calidad del software al prevenir problemas y optimizar su rendimiento.	Laura hizo un programa para sacar promedio, así que imitó la serie de pasos del algoritmo a mano, así revisa si su programa está en correcto funcionamiento.