

CURSO DE FUNDAMENTOS DE PROGRAMACIÓN DESDE CERO



EL PROCESO DE CREACIÓN DE UN PROGRAMA: DE LA IDEA AL SOFTWARE





DE LA IDEA AL SOFTWARE



Crear un programa no consiste solo en escribir código. Es un proceso planificado y estructurado que va desde identificar una necesidad hasta obtener un software que funcione correctamente.





DE LA IDEA AL SOFTWARE



Cada etapa es importante para asegurar que el programa resuelva el problema real de manera eficiente.





DE LA IDEA AL SOFTWARE



1. IDENTIFICACIÓN DEL PROBLEMA O NECESIDAD

3. DISEÑO DEL ALGORITMO O SOLUCIÓN

5. PRUEBAS Y DEPURACIÓN



7. IMPLEMENTACIÓN O ENTREGA DEL SOFTWARE

2. ANÁLISIS DEL PROBLEMA

4. CODIFICACIÓN



6. DOCUMENTACIÓN Y MANTENIMIENTO



1. IDENTIFICACIÓN DEL PROBLEMA O NECESIDAD

Todo programa nace de una idea o necesidad.

“¿QUÉ PROBLEMA QUIERO RESOLVER CON MI PROGRAMA?”

EJEMPLO: UNA TIENDA QUIERE UN SISTEMA PARA REGISTRAR SUS VENTAS.



1. IDENTIFICACIÓN DEL PROBLEMA O NECESIDAD

En esta etapa se:

1. Define el propósito del software.
2. Identifican los usuarios finales.
3. Determinan las funciones principales.



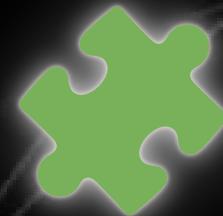


2. ANÁLISIS DEL PROBLEMA

Consiste en entender a fondo lo que se debe hacer.

El programador analiza los datos de entrada, los procesos necesarios y los resultados esperados.



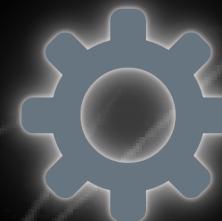


2. ANÁLISIS DEL PROBLEMA

Ejemplo:

- Entradas: productos vendidos, precios, cantidad.
- Procesos: cálculo del total, actualización de stock.
- Salidas: factura o ticket de venta.

SE DETERMINA QUÉ DEBE HACER EL PROGRAMA (NO AÚN CÓMO HACERLO).



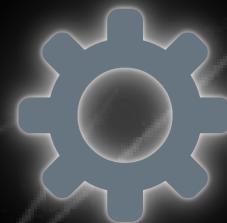
3. DISEÑO DEL ALGORITMO O SOLUCIÓN

Aquí se planifica la lógica del programa antes de escribir código.

Se pueden usar herramientas como:

Diagramas de flujo.

Pseudocódigo.



3. DISEÑO DEL ALGORITMO O SOLUCIÓN

El objetivo es definir los pasos que seguirá el programa para resolver el problema.

Ejemplo:

1. Leer los datos del producto.
2. Calcular el total.
3. Mostrar el resultado.



4. CODIFICACIÓN

En esta etapa se traduce el algoritmo a un lenguaje de programación (Python, Java, C++, etc.).





4. CODIFICACIÓN

El programador escribe el código fuente siguiendo las reglas del lenguaje y las buenas prácticas.

Ejemplo: usar un if para tomar decisiones o un for para repetir acciones.



5. PRUEBAS Y DEPURACIÓN

Una vez escrito el código, se debe probar el programa para detectar errores (bugs) y corregirlos.





5. PRUEBAS Y DEPURACIÓN

Tipos de pruebas:

Pruebas de funcionamiento: comprobar si el programa hace lo esperado.

Pruebas de errores: verificar qué pasa ante datos incorrectos o situaciones imprevistas.



5. PRUEBAS Y DEPURACIÓN

La depuración (debugging) es una parte esencial del trabajo del programador.





6. DOCUMENTACIÓN Y MANTENIMIENTO

Después de que el programa funciona, se debe documentar cómo está hecho y cómo se usa.





6. DOCUMENTACIÓN Y MANTENIMIENTO

Además, con el tiempo se realizan mejoras o actualizaciones (mantenimiento).





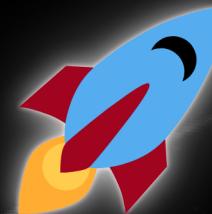
6. DOCUMENTACIÓN Y MANTENIMIENTO

Ejemplos:

Corregir errores encontrados por los usuarios.

Agregar nuevas funciones.

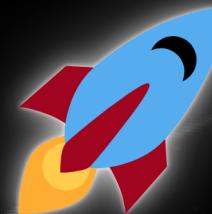
Adaptarlo a nuevas tecnologías.



7. IMPLEMENTACIÓN O ENTREGA DEL SOFTWARE

Finalmente, el programa se pone en uso. Puede instalarse en computadoras, servidores o publicarse en la web.





7. IMPLEMENTACIÓN O ENTREGA DEL SOFTWARE

Aquí se capacita a los usuarios y se recopila retroalimentación para futuras versiones.





CONCLUSIÓN

El desarrollo de software es un proceso ordenado y cíclico, no una sola actividad.





CONCLUSIÓN

Cada etapa —desde la idea hasta la implementación— garantiza que el resultado final sea un programa útil, eficiente y confiable.





CONCLUSIÓN

Comprender este proceso ayuda a pensar como un verdadero programador y a crear soluciones tecnológicas de calidad.



**NOS VEMOS EN UN PRÓXIMO
VIDEO DE ESTE CURSO,
SALUDOS**

