

Table of contents

Bienvenido a Fundamentos de Programación	2
Algoritmos	4

Bienvenido a Fundamentos de Programación

Este curso está diseñado para estudiantes sin experiencia previa en programación. Aprenderás los conceptos básicos de programación y cómo aplicarlos en la práctica. Al final del curso, podrás escribir programas simples y resolver problemas de programación.

Objetivos del curso

- Aprender los conceptos básicos de programación.
- Aplicar los conceptos aprendidos en la práctica.
- Escribir programas simples.
- Resolver problemas de programación.
- Prepararte para cursos más avanzados de programación.
- Desarrollar habilidades de resolución de problemas.
- Aprender a trabajar en equipo.
- Aprender a comunicarte con otros programadores.
- Aprender a buscar información y recursos en línea.
- Aprender a utilizar herramientas de programación.

Requisitos previos

No necesitas experiencia previa en programación para tomar este curso. Sin embargo, es útil tener conocimientos básicos de matemáticas y lógica.

Estructura del curso

El curso consta de varias lecciones, cada una con una serie de temas. Cada tema incluye una explicación teórica y ejemplos prácticos. Al final de cada lección, hay ejercicios para practicar lo aprendido.


Evaluación

La evaluación se basa en los ejercicios realizados durante el curso. Al final del curso, tendrás que completar un proyecto final para demostrar tus habilidades de programación.

Recursos


IDE (Entorno de Desarrollo Integrado)


- CLion (<https://www.jetbrains.com/clion/>)
- Visual Studio Code (<https://code.visualstudio.com/>)
- Dev-C++ (https://filehippo.com/es/download_dev-c/)

 **Nota:** Puedes utilizar cualquier IDE que te resulte cómodo. Sin embargo, se recomienda utilizar **CLion** para este curso.

Herramientas de Diseño

- Draw.io (<https://app.diagrams.net/>)

 **Nota:** Puedes utilizar cualquier herramienta de diseño que te resulte cómoda. Sin embargo, se recomienda utilizar **Draw.io** para este curso.

 **Nota:** Queda prohibido el uso de herramientas como **PSeInt**, ya que no permiten la escritura de código real. Así como también queda prohibido el uso de herramientas de diseño como **PowerPoint** o **Word**, porque no son adecuadas para la creación de diagramas de flujo.

Algoritmos

Introducción

En este tema, aprenderás qué es un algoritmo, por qué es importante y cómo diseñar algoritmos eficientes. También aprenderás a implementar algoritmos en pseudocódigo y diagramas de flujo. Finalmente, resolverás problemas de programación utilizando algoritmos y desarrollarás habilidades de resolución de problemas.

Objetivos

- Comprender qué es un algoritmo y por qué es importante.
- Aprender a diseñar algoritmos eficientes.
- Aprender a implementar algoritmos en pseudocódigo y diagramas de flujo.
- Resolver problemas de programación utilizando algoritmos.
- Desarrollar habilidades de resolución de problemas.

Contenido

1. ¿Qué es un algoritmo?
2. Características de un algoritmo.

¿Qué es un algoritmo?

Un algoritmo es un conjunto de instrucciones que describe cómo resolver un problema. Los algoritmos son la base de la programación y se utilizan para diseñar programas informáticos. Un buen algoritmo es eficiente, preciso y fácil de entender.

Características de un algoritmo

Un algoritmo debe tener las siguientes características:

1. Precisión (Definición exacta de cada paso):

- Cada paso del algoritmo debe estar claramente definido, sin ambigüedades. Esto significa que las instrucciones deben ser entendibles y ejecutables por cualquiera que siga el algoritmo, ya sea una persona o una máquina.

2. Finito (Debe tener un final):

- Un algoritmo debe tener un número finito de pasos. No puede ser infinito; debe llegar a una conclusión en algún momento, resolviendo el problema planteado.

3. Entrada (Datos de entrada):

- Un algoritmo puede requerir cero o más entradas para funcionar. Estas entradas son los datos iniciales necesarios para que el algoritmo comience su proceso.

4. Salida (Resultados generados):

- Como resultado del proceso, el algoritmo debe producir una o más salidas. Estas son las soluciones o los resultados obtenidos tras la ejecución de los pasos del algoritmo.

5. Efectividad (Operaciones básicas ejecutables):

- Los pasos de un algoritmo deben ser simples y realizables en un tiempo razonable. Cada operación debe ser lo suficientemente básica como para ser realizada sin necesidad de un análisis adicional complejo.

6. Claridad (Debe ser comprensible):

- Un algoritmo debe ser claro y comprensible. Esto significa que cualquier persona con el conocimiento adecuado debe poder leer y entender el algoritmo sin necesidad de interpretaciones adicionales.

7. Independencia (Independencia del lenguaje de programación):

- Un algoritmo no depende de un lenguaje de programación específico. Se puede escribir en lenguaje natural, diagramas de flujo, pseudocódigo, etc., y luego implementarse en cualquier lenguaje de programación.

Ejemplo

Imagina que estás creando un algoritmo para preparar un sándwich. Los pasos serían algo así:

1. Entrada:

- Pan, jamón, queso, lechuga, tomate, etc.

2. Proceso:

- Colocar una rebanada de pan en la mesa.
- Colocar una rebanada de jamón sobre el pan.
- Agregar una rebanada de queso.
- Colocar lechuga y tomate.
- Cubrir con la otra rebanada de pan.

3. Salida:

- Un sándwich listo para comer.

Este ejemplo cumple con todas las características mencionadas: es preciso, finito, tiene entradas y salidas, es efectivo, claro, y puede ser interpretado sin importar el idioma en el que se describa.

Estas características aseguran que un algoritmo sea confiable y utilizable para resolver problemas de manera sistemática y eficiente.

Conclusión

Los algoritmos son la base de la programación y la resolución de problemas. Al comprender qué es un algoritmo y cómo diseñarlo de manera eficiente, podrás mejorar tus habilidades de programación y resolver problemas de manera más eficiente. ¡Sigue practicando y desarrollando tus habilidades algorítmicas!