Academia Techno Artist

Temario Oficial de C++

Desde Fundamentos hasta Programación Avanzada e IA

Este temario abarca desde fundamentos hasta programación avanzada, optimización de código, concurrencia, C++20 e integración de Inteligencia Artificial.

Primer Nivel: Fundamentos de C++ y Programación

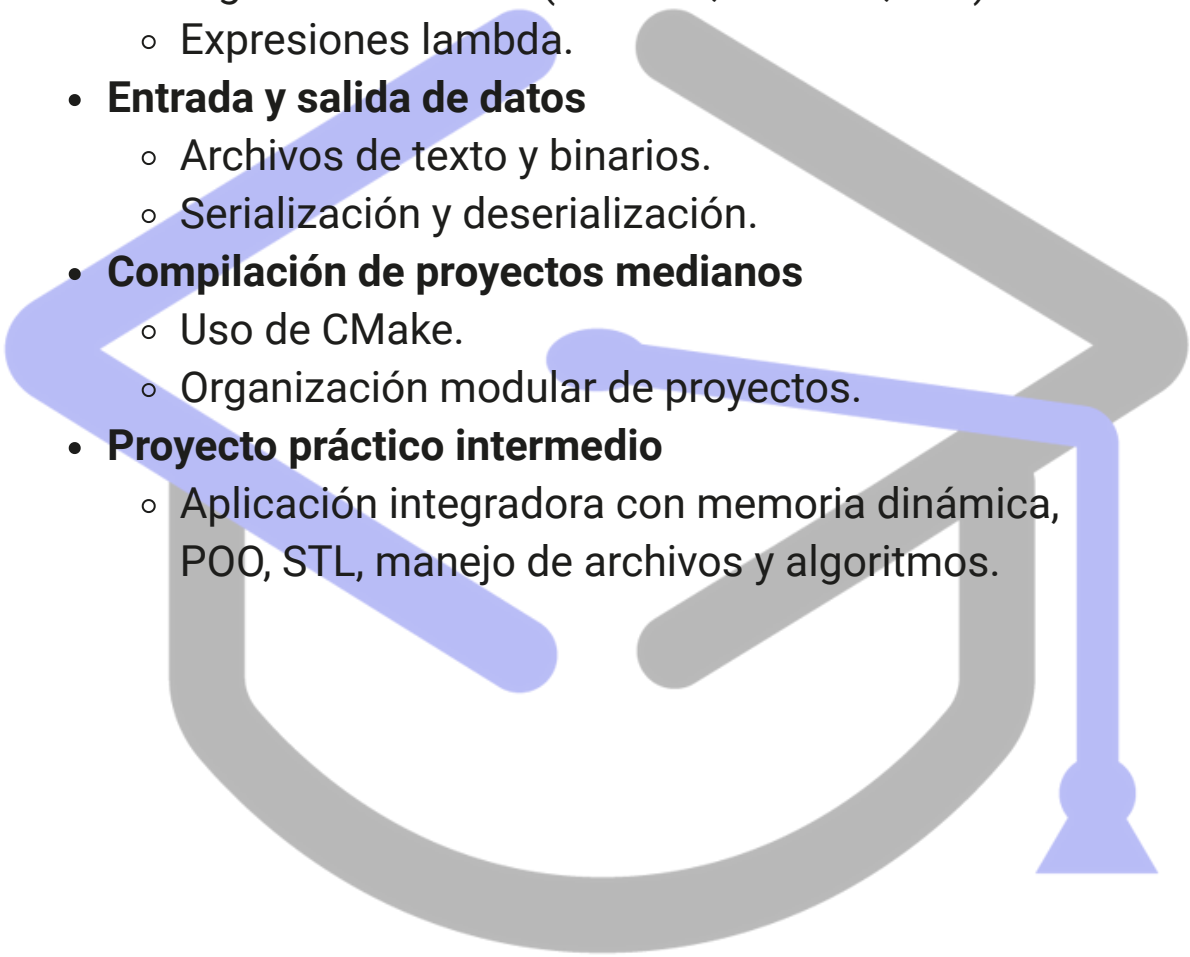
- **Conceptos básicos de programación**
  - Variables y tipos de datos.
  - Operadores aritméticos, lógicos, relacionales y de asignación.
  - Estructuras de control condicional (if, else, switch).
  - Estructuras de bucle (for, while, do-while).
  - Enumeraciones (enum).
  - Espacios de nombres (namespace).
  - Introducción a std::string y manipulación de cadenas.
- **Funciones**
  - Definición y declaración.
  - Parámetros, argumentos y valor de retorno.
  - Modularización y reutilización.
  - Sobrecarga de funciones.
- **Arreglos y vectores**
  - Arreglos unidimensionales y multidimensionales.
  - Introducción y uso de std::vector.
  - Manejo de colecciones de datos dinámicas.
- **Estructuras (struct) y clases básicas**
  - Definición de estructuras.
  - Introducción a clases y objetos.
  - Encapsulación y abstracción.
  - Diferencias entre struct y class.
  - Matrices (uso y manipulación).

Segundo Nivel: Programación Intermedia en C++

- **Gestión de memoria y punteros**
  - Uso de punteros y referencias.
  - Punteros a funciones.
  - Memoria dinámica (new, delete).
  - Arreglos dinámicos.
- **Programación Orientada a Objetos avanzada**
  - Herencia simple y múltiple.
  - Polimorfismo.
  - Constructores y destructores.
  - Clases abstractas y métodos virtuales.
  - Patrones básicos: Singleton, Factory.
- **Plantillas y programación genérica**
  - Plantillas de funciones y clases.
  - Conceptos (concepts) en C++20.
- **Manejo de errores**
  - Excepciones (try, catch, throw).
  - Clases de excepción personalizadas.
- **Algoritmos y estructuras de datos**
  - Algoritmos de ordenamiento: burbuja, selección, inserción.
  - Algoritmos avanzados: quicksort, mergesort.
  - Búsquedas: secuencial y binaria.

Segundo Nivel: Programación Intermedia en C++

- **Standard Template Library (STL)**
  - Contenedores: vector, map, set, queue, stack.
  - Iteradores.
  - Algoritmos de STL (std::sort, std::find, etc.).
  - Expresiones lambda.
- **Entrada y salida de datos**
  - Archivos de texto y binarios.
  - Serialización y deserialización.
- **Compilación de proyectos medianos**
  - Uso de CMake.
  - Organización modular de proyectos.
- **Proyecto práctico intermedio**
  - Aplicación integradora con memoria dinámica, POO, STL, manejo de archivos y algoritmos.

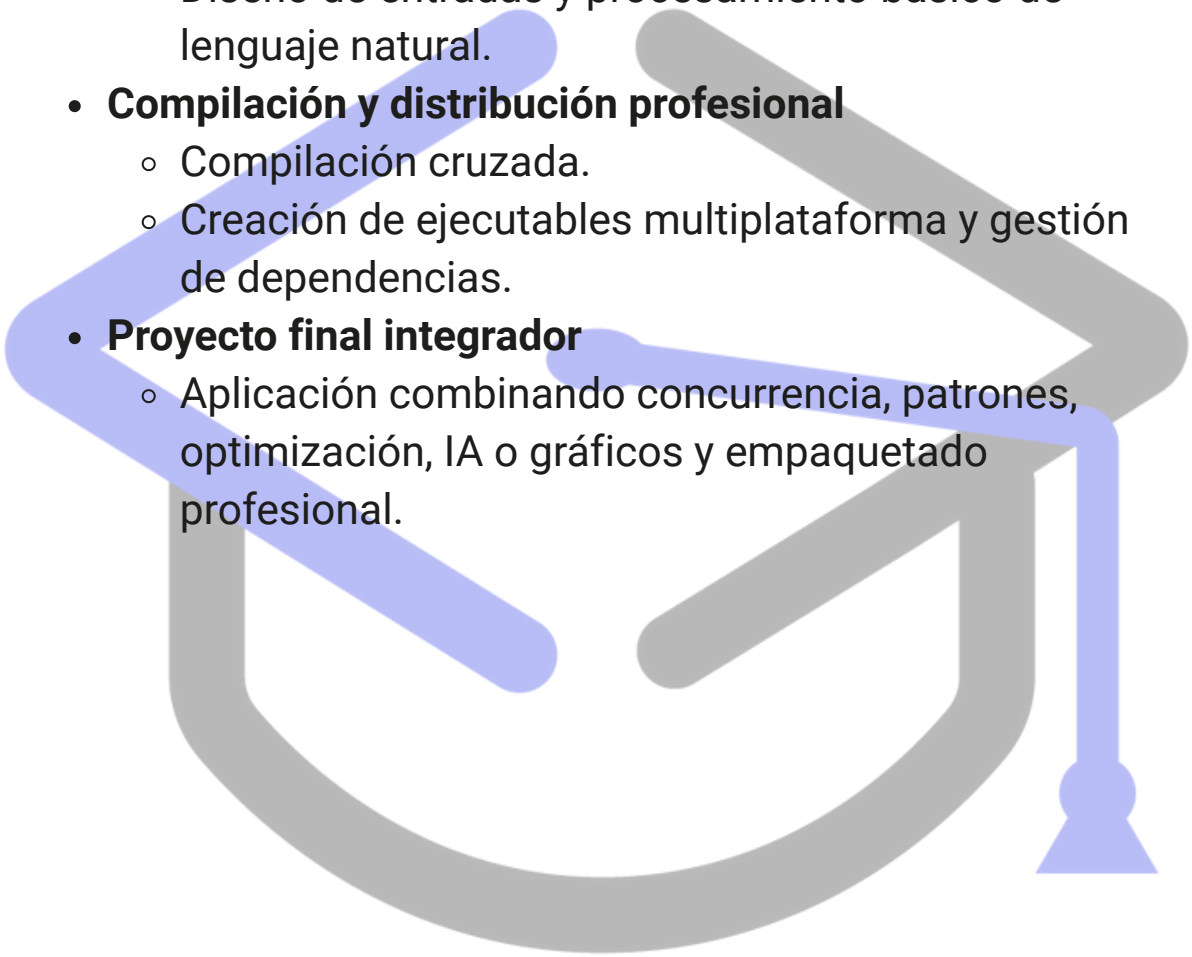


Tercer Nivel: Programación Avanzada, Concurrencia y Diseño de Software

- **Patrones de diseño y arquitecturas**
  - MVC, Singleton, Factory, Observer, Adapter, Command.
  - Arquitecturas escalables y mantenibles.
- **Programación concurrente**
  - std::thread, std::mutex, std::lock_guard.
  - std::future, std::promise, std::async.
  - Coroutines y std::jthread en C++20.
- **Optimización de código**
  - Uso eficiente de memoria.
  - Técnicas de optimización algorítmica.
  - Benchmarking y profiling (Valgrind, gprof, Visual Studio Profiler).
- **Redes y comunicación**
  - Introducción a sockets.
  - Boost.Asio y programación en red.
- **Integración de IA en C++**
  - Fundamentos de algoritmos de búsqueda, toma de decisiones y machine learning.
  - Integración con TensorFlow C++ API y ONNX Runtime.
  - Wrappers C++ ↔ Python con Pybind11.

Tercer Nivel: Programación Avanzada, Concurrencia y Diseño de Software

- **Manejo de errores en aplicaciones de IA**
  - Debugging de modelos y validación de resultados.
- **Creación de prompts efectivos**
  - Diseño de entradas y procesamiento básico de lenguaje natural.
- **Compilación y distribución profesional**
  - Compilación cruzada.
  - Creación de ejecutables multiplataforma y gestión de dependencias.
- **Proyecto final integrador**
  - Aplicación combinando concurrencia, patrones, optimización, IA o gráficos y empaquetado profesional.



Cuarto Nivel: Especialización

- **Interfaces gráficas**
  - Introducción a Qt.
  - Uso básico de SDL y SFML para videojuegos o interfaces.
- **Unit Testing**
  - Fundamentos de pruebas unitarias.
  - Uso de Google Test.
- **Seguridad en C++**
  - Prevención de buffer overflows.
  - Buenas prácticas con punteros.
  - Sanitizers en compilación.
- **Serialización avanzada**
  - Uso de JSON con nlohmann::json.
  - Manejo de XML.
- **Automatización de procesos**
  - Creación de scripts y herramientas utilitarias en C++.
- **Diseño de APIs en C++**
  - Exposición de funciones y clases a través de APIs.
  - Manejo de versiones y compatibilidad.