

TECNOLOGICO NACIONAL DE MEXICO

HISTORIA DE LOS COMPILEDORES

INTRODUCCION

La historia de los compiladores nos sumerge en las bases de la informática moderna. Estas herramientas que en principio parecen simples han sido los pilares sobre los cuales se ha construido la capacidad de los humanos para comunicarse con las maquinas. A través de la presente línea del tiempo se planea explorar el desarrollo y evolución de los compiladores.

1952

Aparece el primer compilador, el A-0,
desarrollado por Grace Hopper



— 1954 —

IBM crea el primer lenguaje de programación FORTRAN y su primer compilador



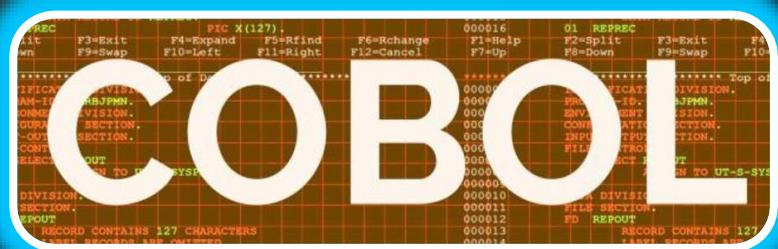
— 1957 —

John Backus y su equipo desarrollan el lenguaje de programación ALGOL, junto con un compilador



— 1960 —

COBOL (Common Business-Oriented Language) es desarrollado por un comité dirigido por Grace Hopper, con su primer compilador disponible poco después



—1964

IBM lanza el lenguaje de
programación PL/I con su propio
compilador



1969

Ken Thompson crea el primer compilador para el lenguaje de programación B, precursor de C, en los laboratorios Bell



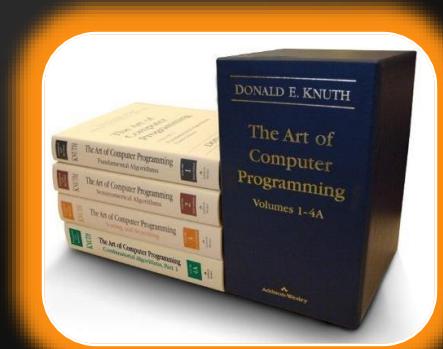
1972

Dennis Ritchie desarrolla el lenguaje de programación C y crea su primer compilador



— 1978 —

Donald Knuth publica su obra “El arte de programar computadoras”, donde aborda temas relacionados con el desarrollo de compiladores



1983

Richard Stallman inicia el proyecto GNU, que eventualmente llevara al desarrollo del compilador GCC (GNU Compiler Collection)



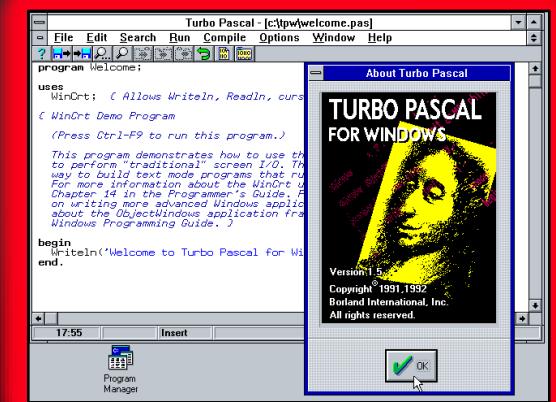
—1987

La International Organization
for Standardization (ISO)
publica el estándar ANSI C,
formalizando el lenguaje C y su
compilador



— 1990 —

Anders Hejlsberg lidera el equipo de desarrollo de Borland para crear el compilador Turbo Pascal



1991

Java es lanzado junto con su compilador, marcando el inicio de la máquina virtual. Desarrollado por James Gosling



1995

Microsoft lanza Visual C++, un entorno de desarrollo integrado que incluye un compilador para C++



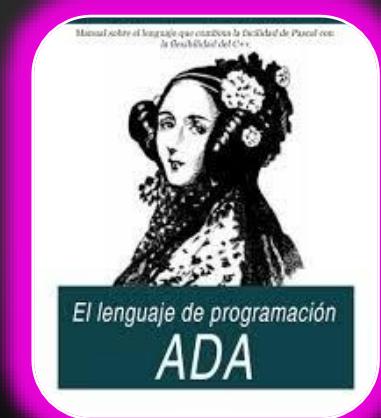
2000

Se realizan avances en la optimización de código de LLVM (Low Level Virtual Machine), un proyecto de investigación que busca proporcionar una infraestructura moderna y flexible para el desarrollo de compiladores y herramientas de análisis estático



2003

La versión 3.4 de GCC introduce
soporte para el lenguaje de
programación Ada



2005

LLVM (Low Level Virtual Machine) se lanza como un proyecto de código abierto, proporcionando una infraestructura para la construcción de compiladores. Ha sido parte importante de las herramientas de desarrollo Xcode de Apple



2008

Se anuncia el lanzamiento de Clang, un nuevo compilador de C/C++ basado en LLVM. Destacado por su rápido tiempo de compilación, diseño modular y mensajes de error amigables, ganando aprobación en la comunidad de desarrolladores



2010

Se lanza GCC 4.5, que incluye mejoras significativas en la optimización del código y la compatibilidad con los estándares de lenguaje de programación más recientes, como C++0x. Introduce el soporte inicial para el lenguaje de programación Go



2011

Se lanza el compilador Clang el cual es compilador de código abierto con énfasis en la velocidad y el soporte para C, C++, y Objective-C. Clang utiliza LLVM como su back-end y ha sido parte del ciclo de lanzamiento de LLVM



— 2012 —

Microsoft anuncia Roslyn, un nuevo compilador de C# y Visual Basic .NET, escrito en C# y expuesto como una API para que los desarrolladores puedan acceder y modificar el código fuente durante la compilación



2014

Se publica la especificación inicial del lenguaje Rust, tiene como objetivo proporcionar un sistema de tipado seguro y un alto nivel de concurrencia. Utiliza LLVM como back-end para su compilador



2017

Se anuncia la versión inicial de Kotlin/Native, un compilador que permite compilar código Kotlin directamente a código máquina, eliminando la necesidad de una máquina virtual Java.



2019

Se anuncia la versión estable de Kotlin 1.3, que incluye el soporte completo para la compilación a código nativo a través de Kotlin/Native



2021

Se anuncia la versión estable de Rust 1.5, incluye mejoras en la sintaxis del lenguaje, el rendimiento del compilador y la seguridad del sistema. Rust continúa ganando popularidad como opción para el desarrollo de sistemas de alto rendimiento y seguridad



2023

Se continúa el desarrollo y la evolución de los compiladores con enfoques hacia la optimización de rendimiento, la parallelización automática y la adaptación a nuevas arquitecturas de hardware



CONCLUSION

En conclusión, para los estudiantes de sistemas, entender la historia de los compiladores es esencial. Esta comprensión proporciona una base sólida en conceptos fundamentales, permite apreciar la evolución tecnológica, mejora las habilidades de resolución de problemas y ofrece un contexto valioso para el desarrollo actual en la construcción de software.