TECNOLÓGICO DE ESTUDIOS SUPERIORES DE JOCOTITLÁN

Normas ISO en compiladores

Alumnos: López Piña Andrés (2022150480607)

Grupo: IC-603

Materia: Lenguajes y Autómatas I

Profesor: Erika López González

Normas ISO en Compiladores

◆ GCC (GNU Compiler Collection)

• Supported languages: C, C++, Fortran, Ada, Objective-C, among others.
• Standards it follows:

- ISO/IEC 9899 (C)
- ISO/IEC 14882 (C++)
- ISO/IEC 1539 (Fortran)
- ISO/IEC 8652 (Ada, using GNAT)
  • Advantages:
- Highly portable (Linux, Windows, macOS).
- Supports multiple standard versions (C89, C99, C11, C17, C++98, C++11, C++20, etc.).
  • Example:

```
gcc -std=c99 program.c
```

This tells the compiler to follow the ISO C99 standard.

◆ Clang/LLVM

• Supported languages: C, C++, Objective-C, Swift (via LLVM).
• Standards it follows:

- ISO/IEC 9899 (C)
- ISO/IEC 14882 (C++)
  • Features:
- Compatible with GCC.
- Focused on analysis tools, clear error messages, and fast compilation.
- Support for modern standards (C++20, C++23).
  • Usage:

```
clang -std=c11 main.c
```

◆ Microsoft Visual C++ (MSVC)

• Language: C, C++
• Standards it follows:

- ISO/IEC 14882 (C++)
- Partial compatibility with ISO/IEC 9899 (C), though it uses proprietary extensions.
  • Note: Mainly covers modern C++ versions, such as C++17 and C++20.
  • Advantage: Widely used for development on Windows and Microsoft platforms.

◆ Intel C++ Compiler (ICC / ICX)
• Language: C, C++
• Standards:

- Fully compatible with ISO/IEC 9899 and 14882.
  • Focus: High performance computing (HPC) and scientific computation.
  • Uses LLVM in its latest versions (ICX).

◆ GFortran
• Language: Fortran
• Standard:

- ISO/IEC 1539
  • Supports: Fortran 77, 90, 95, 2003, 2008, and partially 2018.
  • Application: Simulation, science, engineering.

◆ GNAT (GNU Ada Translator)
• Language: Ada
• Standard:

- ISO/IEC 8652
  • Purpose: Embedded, critical, military, and aerospace systems.
  • Example of use: Real-time development with strong safety verification.

◆ Roslyn (Microsoft C# Compiler)
• Language: C#
• Standard:

- ISO/IEC 23270
  • Features:
- Open source.
- Compiles C# according to official standards and supports static code analysis.

🔹 JRuby / Ruby MRI
• Language: Ruby
• Standard:

- ISO/IEC 30170
  • Note: Not all interpreters fully comply, but MRI (Matz's Ruby Interpreter) is the reference aiming for full compliance.[1]

Summary Table

Compiler	Language	ISO Standards Followed	Notable Points
GCC	C, C++, Fortran, Ada	ISO 9899, 14882, 1539, 8652	Widely used in Linux systems
Clang	C, C++	ISO 9899, 14882	GCC compatible, very modern
MSVC	C, C++	ISO 14882	Better at C++ than standard C
Intel ICX	C, C++	ISO 9899, 14882	Optimized performance
GFortran	Fortran	ISO 1539	Science and engineering
GNAT	Ada	ISO 8652	High reliability, critical systems
Roslyn	C#	ISO 23270	Modern language, cross-platform
MRI / JRuby	Ruby	ISO 30170	Reference implementations

How to Know if a Compiler Complies with ISO

1. Official documentation: Check the compiler's official page for ISO/IEC compliance.
2. Command-line options: Many compilers let you specify the standard using flags like -std=c11, -std=c++17, etc.
3. Formal certifications: Some compilers (especially in critical fields like Ada or embedded systems) are formally certified for ISO compliance.

Why Are ISO Standards Important in Compilers?

A compiler transforms source code into machine code. Without a global standard, each compiler might interpret the language differently. ISO standards prevent this by setting official rules that compilers must follow.

Practical Example: ISO/IEC 9899 (C Language)

Suppose you have this C code:

```
int main() {

   int a = 5 / 2;

   printf("%d\n", a);

   return 0;

}
```

- According to ISO 9899, integer division must also result in an integer.
- A compliant compiler must return 2 (not 2.5).
- If you use float, the compiler may apply IEEE 754 (if defined in the C standard).

The standard also defines:

- How printf() should behave.
- How main() should be interpreted.
- What constitutes undefined behavior (e.g., division by zero, out-of-bounds access).[2]

Referencias

[1]. P. Svoboda, How Exploitable is Insecure C Code?, IEEE Security Development Conference, 2021. [Online]. Available: https://secdev.ieee.org/wp-content/uploads/2021/10/tutorial-A4-svoboda.pdf

[2]. IEEE, POSIX® Product Standards: Realtime Controller 1003.13 PSE54, IEEE, 2003. [Online]. Available: https://get.posixcertified.ieee.org/docs/pse54-2003-1.1.html