Introduction to the C Language

Md. Aminul Islam Shazid

Outline

1. History
2. Features of C
3. Getting Started with C
4. Our First C Program

History

Creation of C

- Created by Dennis Ritchie in close collaboration with Ken Thompson at the Bell Labs in the early 1970s
- The first operating system written in C is Unix which ran on the 16bit PDP-11 computer

Evolution of C

- In 1978, Dennis Ritchie and Brian Kernighan released the book titled “The C Programming Language”
- C was standardized by ANSI (known as C89) in 1989, this was adopted by ISO later in 1990 (known as C90)
- The latest version of the standard is C23
- Since 2000, consistently ranked top four in the TIOBE index

Features of C

Basic Features

- High level (compared to assembly or machine code)
- Also provides low level access
  - Allows writing inline assembly code
  - Provides direct access to memory management
- Fewer keywords compared to other contemporary languages
- Not object oriented

Basic Features (cont.)

- Compiles to native machine code – runs very fast
- Statically typed, supports user defined data types
- Compund data types – structs and unions
- Standard library as well as third party libraries

Shortcomings

- Lacks exceptions
- No garbage collection
- Can be unsafe if not careful
  - No range-checking
  - Limited type checking at compile time
  - No type checking at runtime

Use Cases

- Operating systems, such as Linux
- Many software
  - Database: PostgreSQL, SQLite, MySQL
  - VLC Player
  - Apache web server, Nginx
- Embedded systems (microcontrollers)
- Other higher level programming languages
  - CPython
  - Ruby
  - PHP
  - JVM (Java, Kotlin etc.)

Getting Started with C

Ingredients

- C source file (plain text file with .c extension)
- Compiler (ideally, also a debugger)
- Text editor
- Terminal or the command line

Familiarity with the command line (PowerShell, BASH etc.) and environment variables (for example, PATH) is highly recommended.

Compilers and Debuggers

- Compiler turns the C code into machine code
- Debugger helps in debugging a compiled program
- Most popular C compilers include
  - GNU C Compiler and GNU Debuger (GCC and GDB)
  - Clang
  - Microsoft Visual C Compiler (MSVC)

IDE

To make life easier, use an IDE (integrated development environment), it combines:

- Source editing
- Invoking compiler
- Debugging
- Version control system (VCS), for example git or subversion

Most modern IDEs also provide “intellisense” as well as perform static analysis on source codes.

Environrment Setup on Windows

- Easiest option: download CodeBlocks with MinGW which includes GCC and GDB
- Microsoft Visual Studio with MSVC (much bigger download size)

Installing Only GCC on Windows

To use an IDE that does not bundle a compiler, the compiler needs to be installed separately. A few options:

- Installing via Cygwin
- Download MinGW-W64 using MSYS2
- Download MinGW-W64 standalone from WinLibs
- Install WinLibs MinGW-W64 via WinGet by running:
  `winget install BrechtSanders.WinLibs.POSIX.UCRT`

Installing via WinGet updates the PATH environment variable automatically.

Environment Setup on Linux

- Install the `gcc` package using the package manager
- Then install the `codeblocks` package or any other suitable IDE

Environment Setup on MacOS

- Install XCode, comes with the Clang compiler
- Can be installed from the App Store
- If only the compilers are wanted, then run the follwing in the terminal: `xcode-select --install`

Crossplatform IDEs and Code Editors

- Codeblocks
- Visual Studio Code (requires seperate extension for C/C++)
- Clion (free for non-commercial use only)
- Eclipse IDE

Our First C Program

Hello World in C

```c
#include <stdio.h>

int main(){
    printf("Hellow, world!");
    return 0;
}
```

Thank you.