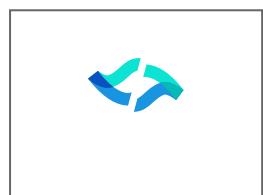


# NekoBytes-TheMissing

讲师-hewo

HDU-计算机科学与技术协会





# Agenda

---

- 为什么学习 C 语言
- 怎么学习 C 语言
- 用什么写 C 语言
- 数制与码制
- C 语言基础语法
- 命令行工具
- VsCode 和 vim

# C 语言介绍

---

“C不是一种“非常高级”的语言，也不是一种“体量大”的语言，并且不专门针对任何特定的应用领域。但它没有限制且具有通用性，这使得它比所谓更强大的语言更方便、更有效地完成许多任务。”——K&R

- 实现了第一个非汇编语言编写的操作系统
- 为什么要学习C？
  - 我们可以编写程序来利用计算机体系结构的底层功能
    - 内存管理！
  - 然而，这也意味着在C语言中事情更容易出错

# C 的优点

---

- 贴近硬件
- 功能强大
- 语法简介
- 学习轻松
- 应用广泛
- 关注底层
- 以及...



# C 的优点

---

- 极致的速度与节能!

**Table 4.** Normalized global results for Energy, Time, and Memory

Total				
	Energy	Time	Mb	
(c) C	1.00	1.00	(c) Pascal	1.00
(c) Rust	1.03	1.04	(c) Go	1.05
(c) C++	1.34	1.56	(c) C	1.17
(c) Ada	1.70	1.85	(c) Fortran	1.24
(v) Java	1.98	1.89	(c) C++	1.34
(c) Pascal	2.14	2.14	(c) Ada	1.47
(c) Chapel	2.18	2.83	(c) Rust	1.54
(v) Lisp	2.27	3.02	(v) Lisp	1.92
(c) Ocaml	2.40	3.09	(c) Haskell	2.45
(c) Fortran	2.52	3.14	(i) PHP	2.57
(c) Swift	2.79	3.40	(c) Swift	2.71
(c) Haskell	3.10	3.55	(i) Python	2.80
(v) C#	3.14	4.20	(c) Ocaml	2.82
(c) Go	3.23	4.20	(v) C#	2.85
(i) Dart	3.83	6.30	(i) Hack	3.34
(v) F#	4.13	6.52	(v) Racket	3.52
(i) JavaScript	4.45	6.67	(i) Ruby	3.97
(v) Racket	7.91	11.27	(c) Chapel	4.00
(i) TypeScript	21.50	26.99	(v) F#	4.25
(i) Hack	24.02	27.64	(i) JavaScript	4.59
(i) PHP	29.30	36.71	(i) TypeScript	4.69
(v) Erlang	42.23	43.44	(v) Java	6.01
(i) Lua	45.98	46.20	(i) Perl	6.62
(i) Jruby	46.54	59.34	(i) Lua	6.72
(i) Ruby	69.91	65.79	(v) Erlang	7.20
(i) Python	75.88	71.90	(i) Dart	8.64
(i) Perl	79.58	82.91	(i) Jruby	19.84

## Languages



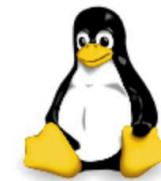
- C 98.4%
- Assembly 0.9%
- Shell 0.3%
- Makefile 0.2%
- Python 0.1%
- Perl 0.1%

# Linux 系统内核

---

📄	bpf_local_storage.c
📄	bpf_lru_list.c
📄	bpf_lru_list.h
📄	bpf_lsm.c
📄	bpf_struct_ops.c
📄	bpf_struct_ops_types.h
📄	bpf_task_storage.c
📄	btf.c
📄	cgroup.c
📄	core.c
📄	cpumap.c
📄	devmap.c
📄	disasm.c
📄	disasm.h
📄	dispatcher.c
📄	hashtab.c
📄	helpers.c
📄	inode.c
📄	local_storage.c

# The Linux Kernel Archives



[About](#)   [Contact us](#)   [FAQ](#)   [Releases](#)   [Signatures](#)   [Site news](#)

Protocol	Location
HTTP	<a href="https://www.kernel.org/pub/">https://www.kernel.org/pub/</a>
GIT	<a href="https://git.kernel.org/">https://git.kernel.org/</a>
RSYNC	<a href="rsync://rsync.kernel.org/pub/">rsync://rsync.kernel.org/pub/</a>

Latest Release

**5.14.10**



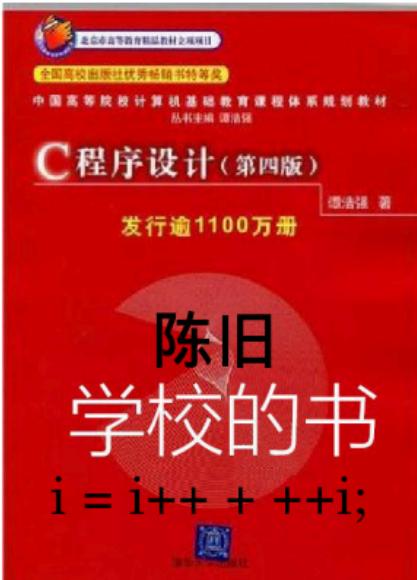
mainline:	<b>5.15-rc4</b>	2021-10-03	<a href="#">[tarball]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>	<a href="#">[browse]</a>
stable:	<b>5.14.10</b>	2021-10-07	<a href="#">[tarball]</a>	<a href="#">[pgp]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>
stable:	<b>5.13.19 [EOL]</b>	2021-09-18	<a href="#">[tarball]</a>	<a href="#">[pgp]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>
longterm:	<b>5.10.71</b>	2021-10-06	<a href="#">[tarball]</a>	<a href="#">[pgp]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>
longterm:	<b>5.4.151</b>	2021-10-06	<a href="#">[tarball]</a>	<a href="#">[pgp]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>
longterm:	<b>4.19.209</b>	2021-10-06	<a href="#">[tarball]</a>	<a href="#">[pgp]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>
longterm:	<b>4.14.249</b>	2021-10-06	<a href="#">[tarball]</a>	<a href="#">[pgp]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>
longterm:	<b>4.9.285</b>	2021-10-06	<a href="#">[tarball]</a>	<a href="#">[pgp]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>
longterm:	<b>4.4.287</b>	2021-10-07	<a href="#">[tarball]</a>	<a href="#">[pgp]</a>	<a href="#">[patch]</a>	<a href="#">[inc. patch]</a>	<a href="#">[view diff]</a>
linux-next:	<b>next-20211007</b>	2021-10-07					<a href="#">[browse]</a>

# Agenda

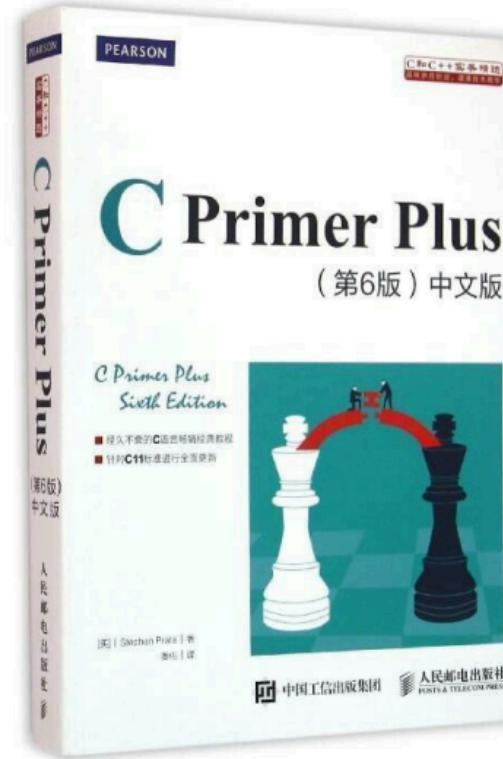
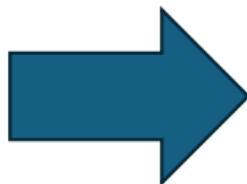
---

- 为什么学习C语言
- 怎么学习C语言
- 用什么写C语言
- 数制与码制
- C语言基础语法
- 命令行工具
- VsCode和vim

# 学习C语言看什么书？



浅薄



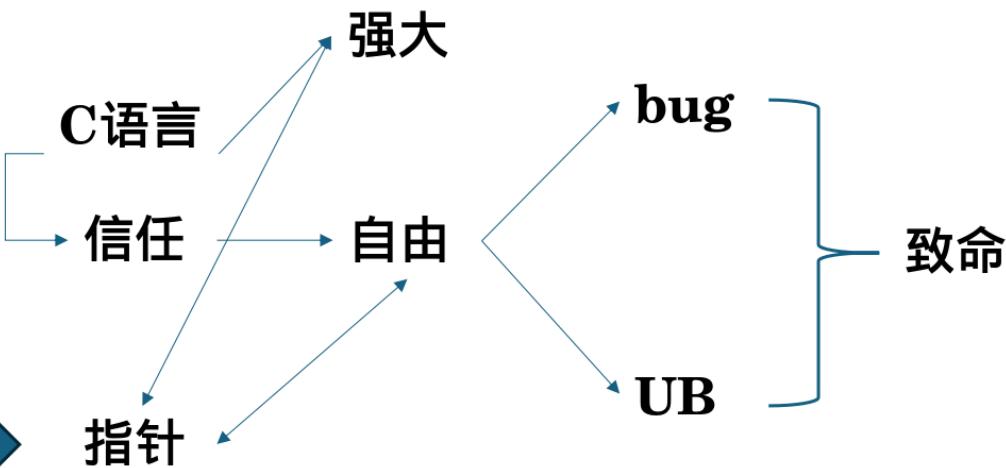
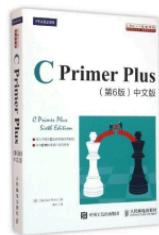
Stephen Prata. C Primer Plus. ISBN 9787115390592.

# C语言的推荐资源

---

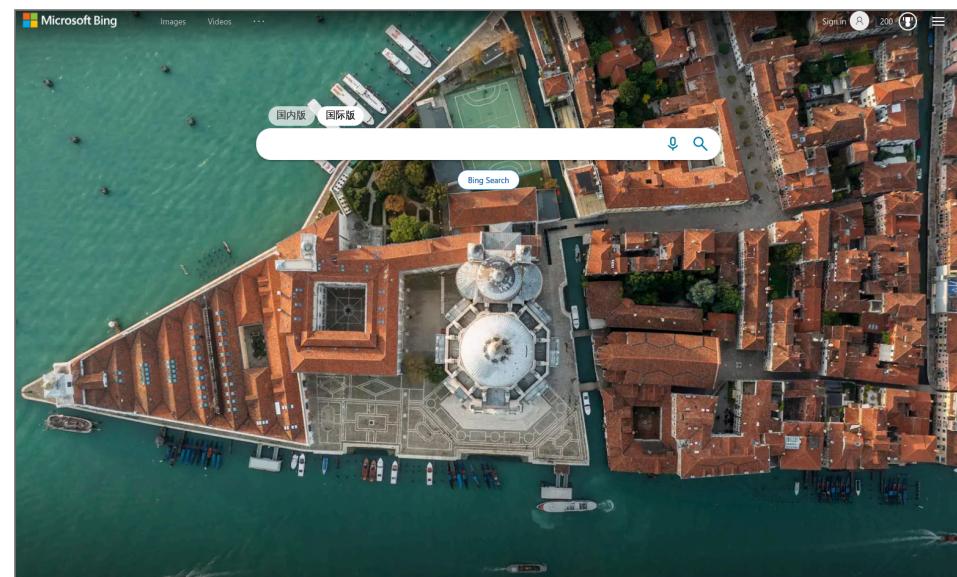
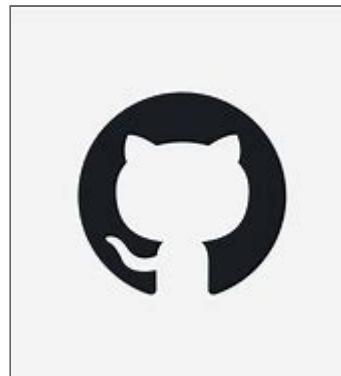
- 杜克大学 Introductory C Programming 专项课程
- Linux C 编程一站式学习
- The C programming language (2nd Edition): 真正的 C 语言之父是这本书的作者 Dennis M. Ritchie, 而不是XXX
- C Coding Standard
- SEI CERT C Coding Standard
- Learn C the hard way 中文版
- C参考手册

# 快乐 C 之旅



# 遇到问题怎么解决？

---





|



Google 搜索

手气不错

Google 提供: English

# 遇到问题怎么解决？



# RTFM: Read The Friendly Manual

---

为什么要阅读手册？

- 准确、迅速地给你答案

## STFW: Search The Friendly Web

---

RTFM 和 STFW：如何知道你已完全搞砸了

有一个古老而神圣的传统：如果你收到RTFM的回应，回答者认为你应该去读他妈的手册。当然，基本上他是对的，你应该去读一读。

RTFM 有一个年轻的亲戚。如果你收到STFW的回应，回答者认为你应该到他妈的网上搜索。那人多半也是对的，去搜索一下吧。（更温和一点的说法是 Google 是你的朋友！）

在论坛，你也可能被要求去爬爬论坛的旧文。事实上，有人甚至可能热心地为你提供以前解决此问题的讨论串。但不要依赖这种关照，提问前应该先搜索一下旧文。

## RTFM: Read The Friendly Manual

---

为什么要阅读手册？

- 准确、迅速地给你答案

## STFW: Search The Friendly Web

---

通常，用这两句之一回答你的人会给你一份包含你需要内容的手册或者一个网址，而且他们打这些字的时候也正在读着。这些答复意味着回答者认为：

- 你需要的信息非常容易获得；
- 你自己去搜索这些信息比灌给你，能让你学到更多。

你不应该因此不爽；依照黑客的标准，他已经表示了对你一定程度的关注，而没有对你的要求视而不见。你应该对他祖母般的慈祥表示感谢。

# Agenda

---

- 为什么学习C语言
- 怎么学习C语言
- **用什么写C语言**
- 数制与码制
- C语言基础语法
- 命令行工具
- VsCode 和 vim

## IDE与编辑器

---

IDE（集成开发环境）和编辑器是软件开发中常用的工具。它们都用于编写、编辑和管理代码，但在功能和用途上有一些区别。

IDE是一种集成了多个工具和功能的软件，旨在提供全面的开发环境。它通常包括代码编辑器、调试器、编译器、构建工具、版本控制系统等。IDE提供了一个统一的界面，使开发人员可以在一个应用程序中完成多个开发任务。

- Visual Studio
- Clion

编辑器则更加轻量级，专注于提供代码编辑功能。它通常具有语法高亮、自动完成、代码折叠等基本功能，但不包含其他开发工具。编辑器通常更加灵活和可定制，适合开发人员根据自己的需求选择插件和扩展。

- Visual Studio Code

# IDE与编辑器

IDE



Visual Studio®



Code::Blocks



编辑器



Visual Studio Code



vim.org



GCC



编译器

# Agenda

---

- 为什么学习C语言
- 怎么学习C语言
- 用什么写C语言
- **数制与码制**
- C语言基础语法
- 命令行工具
- VsCode和vim

## 二进制

---

使用0和1字符串表示数字的方法

为什么计算机使用二进制？

- 计算机的基本构建模块是只能表示两个值的晶体管。 我们决定将这两个值标记为0和1

通过在字符串前面添加0b或添加下标2来表示

# 十六进制

---

- 更易于人类阅读的二进制表示方法
- 基数：16
- 一位十六进制数字可以表示16个数字
- 一位十六进制数字 = 半个字节
- 通过前置“0x”或附加下标16来表示

Decimal	Binary(0b)	Hex(0x)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

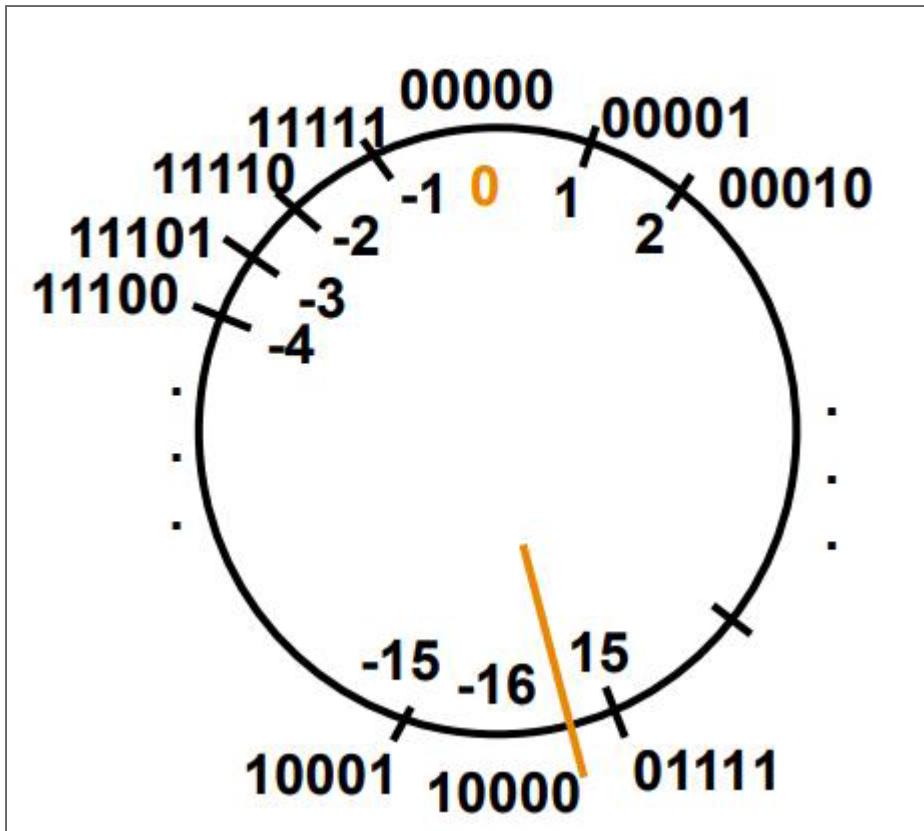
## 补码 (了解即可)

---

如何形成负数？

- 翻转位并加一  
最高位代表值的符号
- 0 表示其正数
- 1 表示其负数

## 补码 (了解即可)



思考

- 有无符号位对表示范围的影响
- 为什么补码可以表示正负？
- 考虑忽略溢出

## 实数（了解即可）

---

定点数表示法

浮点数表示法

- IEEE 754 浮点数标准
- IEEE 754 模拟器
  - <https://www.h-schmidt.net/FloatConverter/IEEE754.html>
  - <https://godbolt.org/z/Yf4Edbjee>

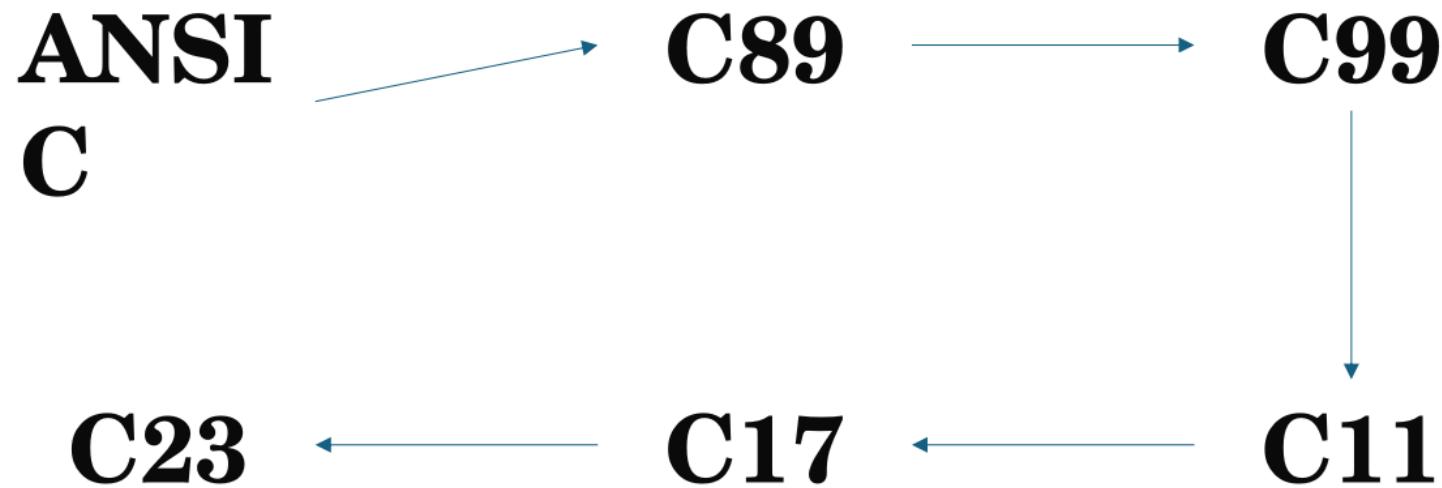
# Agenda

---

- 如何科学的提问
- 为什么学习C语言
- 怎么学习C语言
- 用什么写C语言
- 数制与码制
- **C语言基础语法**
- 命令行工具
- VsCode和vim

# C 语言语法标准

---



## 编译 vs 解释

---

我们在某种程度上用英语编写代码，但系统实际上只看到0和1。我们该如何解决这个问题？

- 如果有人不懂英语，但懂法语，我们会翻译文字！
- 系统中的处理类似，但我们翻译成非人类可读的语言  
翻译以两种方式进行
- 编译（事前翻译）
- 解释（在线翻译）
- 有些语言同时使用这两种方式！

## 编译

---

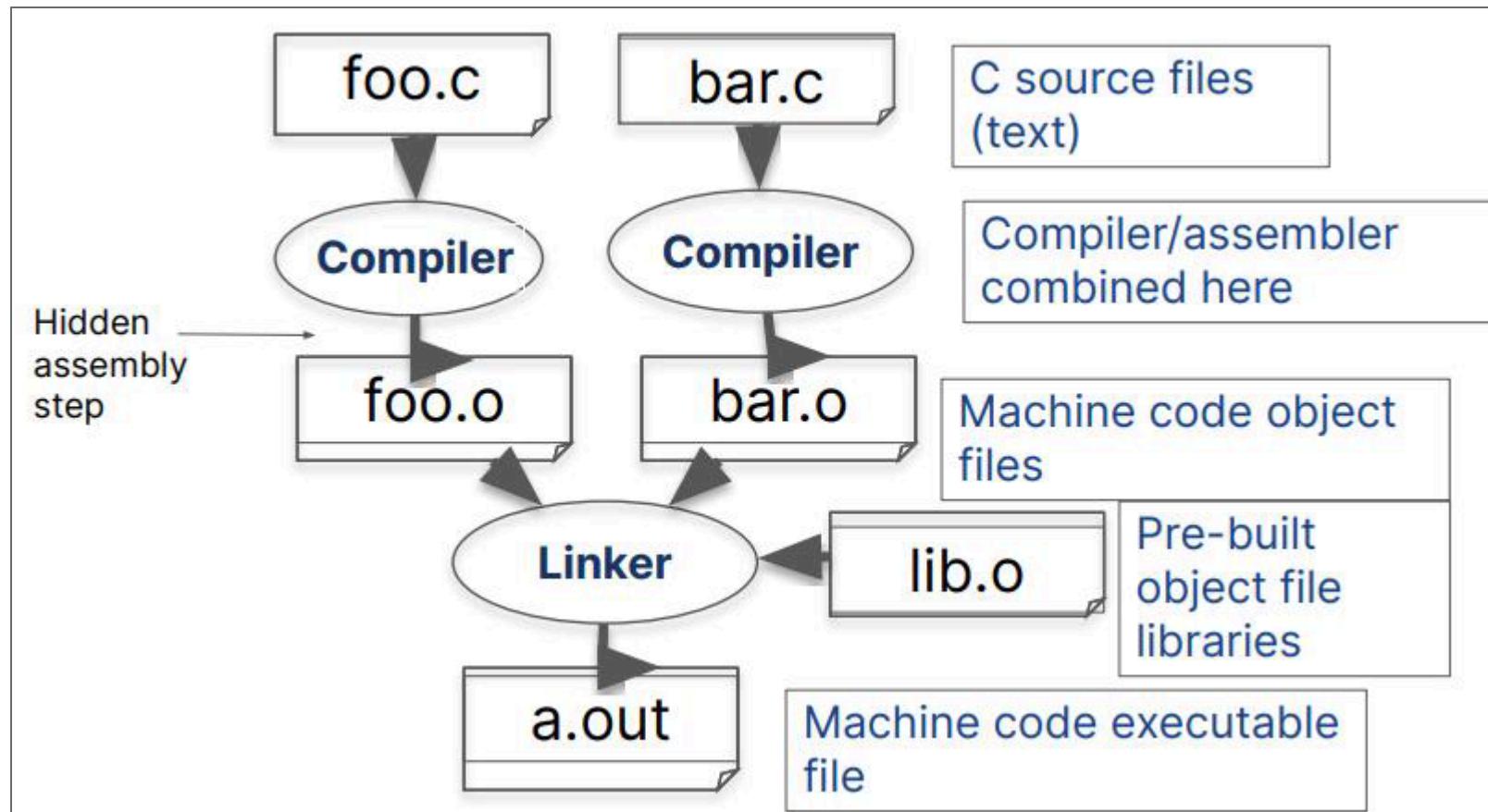
C 编译器将 C 程序直接映射为特定于体系结构的机器代码（1 和 0 的数值串）。

- Java 转换为独立于体系结构的字节码，然后由即时 (JIT) 编译器进行编译。
- Python 在运行时而不是编译时转换为 Python 字节码。
  - 运行时编译与 JIT 编译的不同之处在于程序转换为低级汇编语言并最终转换为机器代码的时间。

对于 C，处理 .c 源文件的编译通常分为 3 个部分

- .c 文件被**编译**为 .s 文件 => 由编译器编译
- .s 文件被**汇编**为 .o 文件 => 由汇编器汇编（此步骤一般是隐藏的，所以大多数时候我们直接将.c文件转换为.o文件）
- .o 文件**链接**在一起创建可执行文件 => 由链接器链接

# C 编译概述



# 起点：Hello World

---

<https://godbolt.org/z/4esj9rvar>

程序的组成要件

- 预编译指令
- main函数
- 注释
- 限定符
- 花括号、函数体和块
- 声明
- 赋值
- 函数调用
- return语句——返回值

# C Pre-Processor (CPP) --C 预处理器

---

C 源文件在编译器看到代码之前首先经过预处理器(CPP)处理

CPP 用单个空格替换注释

CPP 命令以“#”开头

- `#include "file.h" /* 将 file.h 插入到文件 */`
- `#include /* 在标准位置查找文件，但没有实际区别 */`
- `#define PI (3.14159) /* 定义常量 */`
- `#if/#endif /* 有条件地包含文本 */`  
使用 gcc 的 `-fpreprocessed` 选项查看预处理结果
- 完整文档位于：<http://gcc.gnu.org/onlinedocs/cpp/>

# 标准库——官方DLC

标准库（Standard Library）是一组在编程语言中提供常用功能和工具的软件库。在C语言中，标准库是由C标准委员会定义的，它包含了一系列的头文件和函数，提供了许多常见的操作和功能，如输入输出、字符串处理、内存管理、数学运算等。

## 官方DLC

此游戏的内容	浏览所有(15)
Fjordur - ARK Expansion Map	免费
Lost Island - ARK Expansion Map	免费
ARK: Genesis Season Pass	¥ 58.00
Valguero - ARK Expansion Map	免费
Ragnarok - ARK Expansion Map	免费
ARK: Extinction - Expansion Pack	N/A
ARK: Aberration - Expansion Pack	N/A
ARK: Scorched Earth - Expansion Pack	N/A
Primitive+ ARK Total Conversion	免费
The Center - ARK Expansion Map	免费
Crystal Isles - ARK Expansion Map	免费
ARK: Survival Evolved Original Soundtrack	N/A
ARK: Expansion Packs Original Soundtrack	N/A
ARK: Genesis Part 1 Original Soundtrack	N/A
ARK: Genesis Part 2 Original Soundtrack	N/A

¥ 58.00 [将所有 DLC 添加至购物车](#)

# 标准库——官方DLC

## C 标准库头文件

C 标准库的接口由下列头文件的汇集定义。

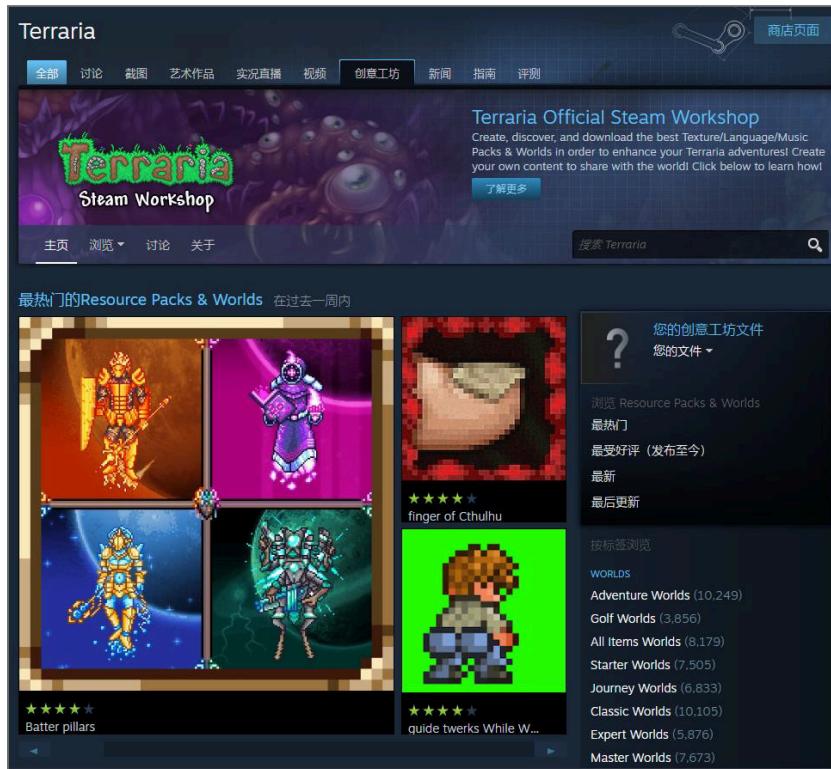
<code>&lt;assert.h&gt;</code>	条件编译宏，将参数与零比较
<code>&lt;complex.h&gt; (C99)</code>	复数运算
<code>&lt;ctype.h&gt;</code>	用来确定包含于字符数据中的类型的函数
<code>&lt;errno.h&gt;</code>	报告错误条件的宏
<code>&lt;fenv.h&gt; (C99)</code>	浮点环境
<code>&lt;float.h&gt;</code>	浮点类型的极限
<code>&lt;inttypes.h&gt; (C99)</code>	整数类型的格式转换
<code>&lt;iso646.h&gt; (C95)</code>	运算符的替代写法
<code>&lt;limits.h&gt;</code>	整数类型的范围
<code>&lt;locale.h&gt;</code>	本地化工具
<code>&lt;math.h&gt;</code>	常用数学函数
<code>&lt;setjmp.h&gt;</code>	非局部跳转
<code>&lt;signal.h&gt;</code>	信号处理
<code>&lt;stdalign.h&gt; (C11)</code>	<code>alignas</code> 与 <code>alignof</code> 便利宏
<code>&lt;stdarg.h&gt;</code>	可变参数

<code>&lt;stdatomic.h&gt; (C11)</code>	原子操作
<code>&lt;stdbool.h&gt; (C99)</code>	布尔类型的宏
<code>&lt;stddef.h&gt;</code>	常用宏定义
<code>&lt;stdint.h&gt; (C99)</code>	定宽度整数类型
<code>&lt;stdio.h&gt;</code>	输入/输出
<code>&lt;stdlib.h&gt;</code>	基础工具：内存管理、程序工具、字符串转换、随机数、算法
<code>&lt;stdnoreturn.h&gt; (C11)</code>	<code>noreturn</code> 便利宏
<code>&lt;string.h&gt;</code>	字符串处理
<code>&lt;tgmath.h&gt; (C99)</code>	泛型数学（包装 <code>math.h</code> 和 <code>complex.h</code> 的宏）
<code>&lt;threads.h&gt; (C11)</code>	线程库
<code>&lt;time.h&gt;</code>	时间/日期工具
<code>&lt;uchar.h&gt; (C11)</code>	UTF-16 和 UTF-32 字符工具
<code>&lt;wchar.h&gt; (C95)</code>	扩展多字节和宽字符工具
<code>&lt;wctype.h&gt; (C95)</code>	用来确定包含于宽字符数据中的类型的函数

# 第三方库——社区模组

第三方库（Third-party library）是由独立的开发者或组织创建和维护的软件库，它们不是编程语言的标准库的一部分。第三方库通常提供了额外的功能和工具，以扩展编程语言的能力，使开发者能够更快速、更方便地开发应用程序。

## 社区模组



## 主函数：main

---

简单的main函数形式：

- int main(void)

要让 main 函数接受参数，请使用以下语句：

- int main (int argc, char \*argv[])

这是什么意思？

- argc 将包含命令行上的字符串数量（可执行文件计为 1，每个参数加 1）。这里 argc 是 2

- \$ touch a.txt

- argv 是一个指向数组的指针，该数组包含字符串形式的参数。

# C 基本变量类型

必须声明变量的类型

- 强变量类型——类型不能更改。例如. int var = 2;

类型	描述	例子
char	8位，ASCII	'a', 'A', '\n', 12
int	整数值（正、负、0），>= 16位，一般为32位	0, 78, -217, 0x2E
unsigned int	整数值（正、0）	0, 6, 35102
short	整数值（正、负、0），>= 16位，一般为16位	0, -8, 32767
long	整数值（正、负、0），>= 32位，一般为32位	0, 78, -217, 301713123194

# C 基本变量类型

---

类型	描述	例子
long long	整数值 (正、负、0) , >= 32位, 一般为64位	31705192721092512
float	单精度浮点数, 32位, IEEE 754	0.0, 3.14, 6.02e3
double	双精度浮点数, 64位, IEEE 754	^
etc.		

## 类型的本质

---

类型的本质是对二进制串的不同理解方式

- 计算机中本只有0和1，理解的方式多了，便成了数据类型  
<https://godbolt.org/z/bP1rbWTae>

# Agenda

---

- 为什么学习C语言
- 怎么学习C语言
- 用什么写C语言
- 数制与码制
- C语言基础语法
- 命令行工具
- VsCode 和 Vim

## 为什么要用命令行

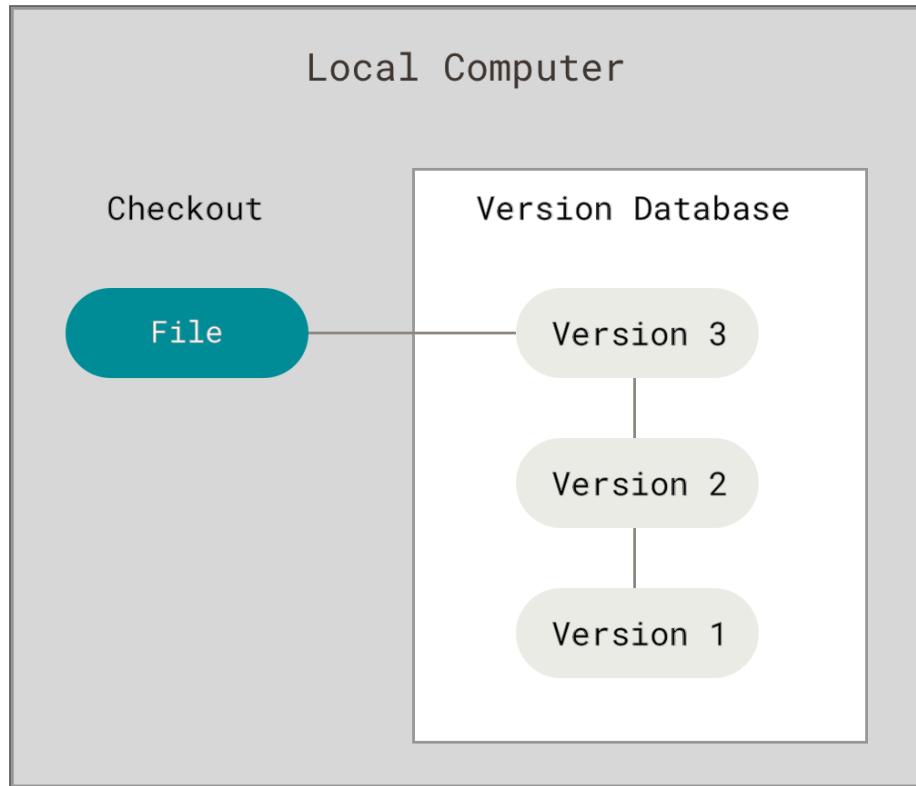
---

作为计算机的直接操控者，CLI 会给你提供更加直接的操作  
想想修改系统某处的时候，你是会选择手忙脚乱的翻 GUI,还是几行命令解决？  
更何况，CLI 其实更加方便。当你熟练使用，并且配置舒服后，你会喜欢上 CLI 的。

# Git

---

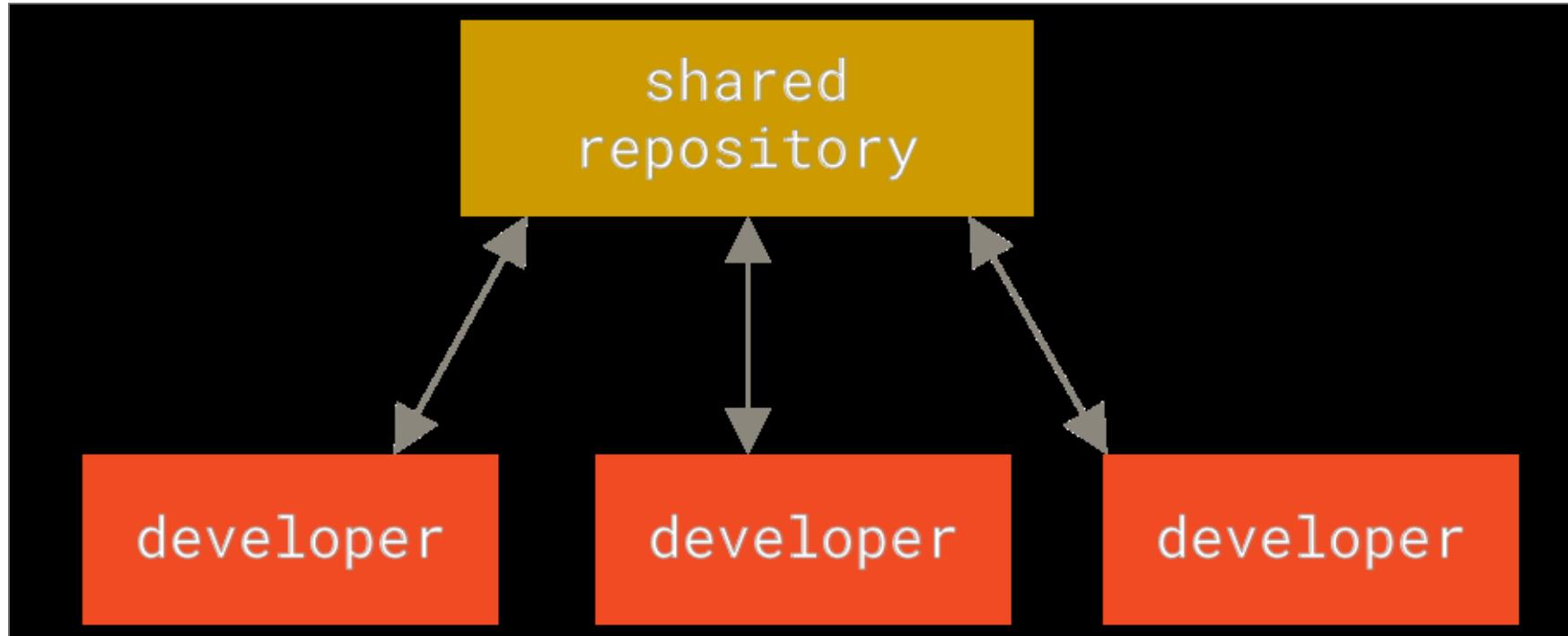
Git是分布式版本控制系统的一大代表  
本地版本控制系统



# Git

---

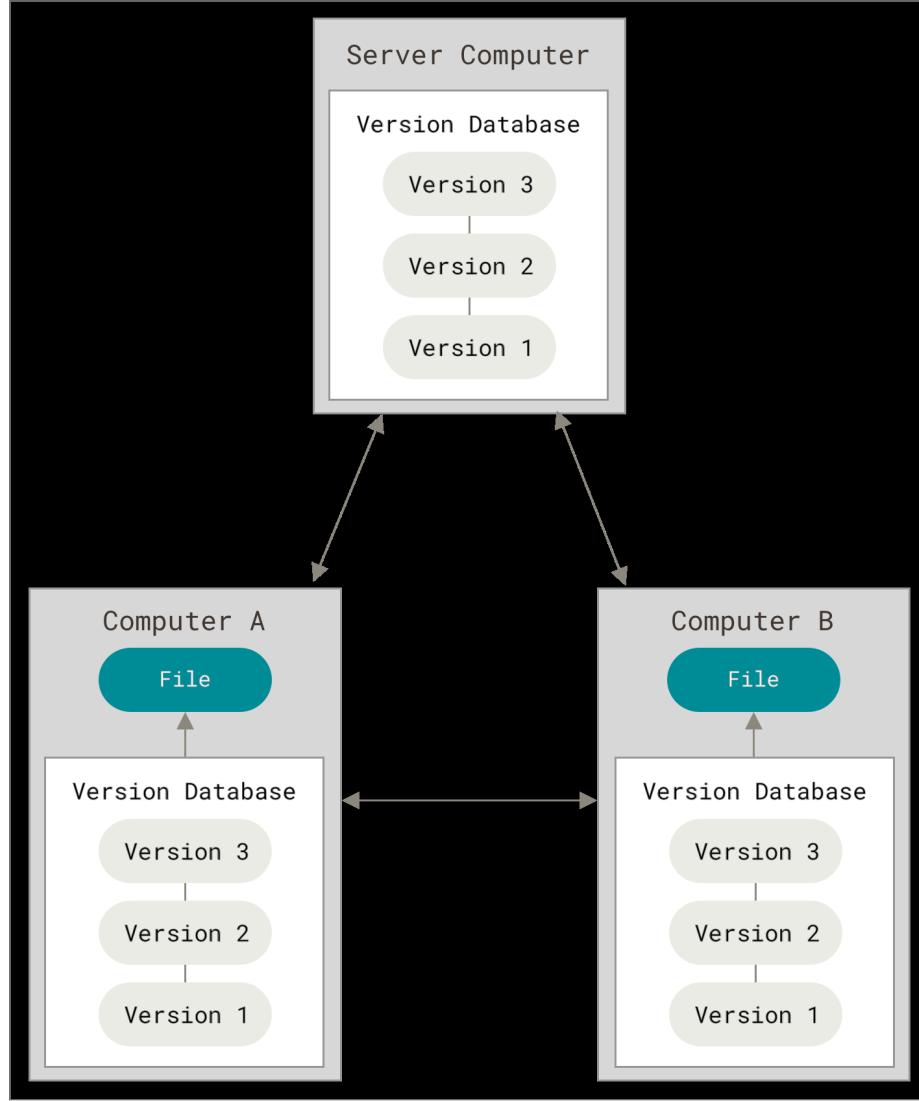
集中化的版本控制系统



# Git

---

## 分布式版本控制系统



# Git的命令行接口

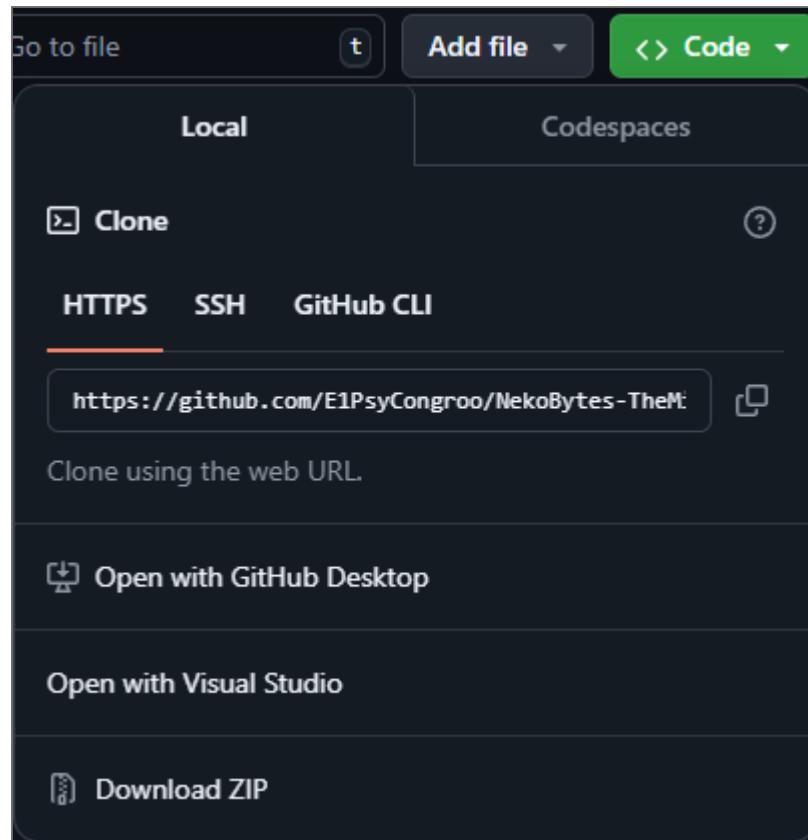
---

- sudo apt install git
- git help <command>
- git init
- git status
- git add <filename>
- git commit
- git log
- git log --all --graph --decorate: 可视化历史记录 (有向无环图)

# Git的命令行接口

---

- git clone <url>



# 怎么学git

---

自己动手实操

- Learn Git Branching (没有项目也能实操)

推荐书籍：

- Pro Git

# tar

---

## 文件后缀 解压命令

.tar tar -xvf <filename>

.tar.gz tar -xzvf <filename>

.tar.bz2 tar -xjvf <filename>

.tar.xz tar -xJvf <filename>

怎么来的？RTFM！

# GCC

---

gcc 全称GNU project C and C++ compiler

gcc是C语言的一种编译器

易混淆的三样东西

- Visual Studio——IDE
- Visual Studio Code——文本编辑器
- GCC——编译器

# GCC的命令行接口

---

## 功能参数

- -o 指定输出文件名
- -Wall 开启警告
- -O(2/3, ...) 优化

## 完整命令

- gcc -o xxx xxx.c

## 运行

- ./xxx

## 项目构建

- Make 与 Makefile

# Agenda

---

- 如何科学地提问
- 为什么学习C语言
- 怎么学习C语言
- 用什么写C语言
- 数制与码制
- C语言基础语法
- 命令行工具
- VsCode和vim

# VSCODE环境配置

---

检查你的 week0  
适合自己的才是最好的

## Vim的哲学

---

在编程的时候，你会把大量时间花在阅读/编辑而不是在写代码上。所以，Vim 是一个多模式编辑器：它对于插入文字和操纵文字有不同的模式。Vim 是可编程的（可以使用 Vimscript 或者像 Python 一样的其他程序语言），Vim 的接口本身也是一个程序语言：键入操作（以及其助记名）是命令，这些命令也是可组合的。Vim 避免了使用鼠标，因为那样太慢了；Vim 甚至避免用上下左右键因为那样需要太多的手指移动。

这样的设计哲学使得 Vim 成为了一个能跟上你思维速度的编辑器。