技术时代的生活与工作

Tech among Life and Work

作者： Qingsong Liao

组织： Free Club

时间： November 21, 2025

邮箱： llqingsong@qq.com

目录

序章 技术何为？

内容提要

- ❑ 任务编号：以名为“廖青松”的男婴为身份出生在重庆普通家庭
- ❑ 开始时间：2002-11-04
- ❑ 结束时间：总有一死
- ❑ 完成难度：简单到极难，活着也没 BOSS 啊？
- ❑ 需要小时：一生，大约三万天
- ❑ 本章要点：我对于技术与人生的看法

(如果你进入了 HTML 版本，请点此处进入 PDF 版本。)

世界嘈杂、瘾品横流、随机与即时构成新的枷锁，而真正的自由来自主动的剥离。当目光离开诱惑、离开广告、离开无意义的瞬息刺激，心才开始变得干净。生活越简，能量越纯；越慢，感受越深；越少，越能看见真正的自己。

身体是意志的第一块土地。空腹的清明、弱光的安静、低温的醒觉、缓慢进食的耐性，都在一点点重塑我们早已被工业习惯磨钝的感官。行走、奔跑、提举、拉起、俯卧撑、壶铃、农夫行走——这些最朴素的动作让人重新理解力量的意义：力量不是爆发，而是日复一日不受伤、不懈怠、让心跳稳、让精神长的那种沉稳。

饮食亦是自律的延伸。避免加工、远离高糖盐油，回到豆果菜、坚果与发酵的本味，让身体习惯真实的能量，而不是被化学甜味与工业脂肪驱使的假饱与假快乐。克制不是苦行，而是温和地恢复本能。

至于技术，真正的价值不在追逐流行，而在深入底层、理解根本。用 Emacs[1]，不是为了高效，而是为了与世界拉开距离，与自己靠得更近；不随机、不即时、无广告、无噪声，是一种长期的心性训练。读 LFS、LKD 与 SOC 文档，写驱动、调性能、跑 QEMU、玩内核、读源码，是为了获得一种“我真的懂了”的安静感。而这种懂，不是为了炫耀，不是为了沉迷，而是为了让工作成为谋生技能，让生活成为真正的生活。

真正的智慧是：学会技术，然后把技术放下；拥有力量，然后让力量变得温柔。生活是吃饭、睡觉、读书、编程、走路、壶铃；生命是健康、乐观、会意、精进、闲适与稳稳的力量。

世界广袤、事物繁多，而心若清澈，幸福忽然变得极小，也极近——不来自外界，只来自自身安静而坚定的内心。

Not the Money || the Passion || the Fame || the Rage || the Art || the Madness
True Programmers write code to { FREE } their SOUL
gnu.org

图 1: free-your-soul

[1]本书的牛“图片”均出自于 GNU.org 的自由艺术。GNU 计划，又称革奴计划，是由 Richard Stallman 在 1983 年 9 月 27 日公开发起的，目标是创建一套完全自由的操作系统，将牛作为 gnu 和 emacs 的象征是因为 gnu 在英文中本身就有“牛羚，角马”的意思，有趣的是 gnu 本意为 Gnu is Not Unix⋯再问里面的 gnu 什么意思？还是 Gnu is Not Unix，就像是中国的俗语“山上有座庙”，无穷尽也。

第一章 TODO 被技术改变的生活

亲情/友情/工作/身体/学习/心理这些通通被技术所改变。亲情因为。
认不到字

第二章 学技术的哲学

内容提要

- ❏ 开始时间：2025-11-19
- ❏ 结束时间: 2026 年前
- ❏ 完成难度：难难难
- ❏ 需要小时：100+
- ❏ 本章要点：linux/c/zig/rust/rtos

人没有梦想那和咸鱼有什么区别呢？——周星驰

2.1 学习什么技术

表 2.1: path-to-know-tech

procress	url	for
fag	flag/cintro	c, pretent iam tsoding
embedded C book	read book	c, how compiler/os/c works
Yeetcode	leetgo/book	cpp[rust], algo ds, solve task
paperlike-c/el	paperlike-go	elisp,paperlike emacs controler
ziglings	ziglings	zig, basic ziglangs speed run
zag	fag	zig, I have language erotic
paperlike-zig	paperlike-el	zig, make cli/tary
nixos r2s	github-repos	nix, for network addiction
TsurgizOS	os.phil-opp	zig[rust], make general os for cv
nixos rasberry-pi	github-repos	nix, for embedded os/screen/driver
clings	ziglings	c, lings but clang
freertos emulator	rtos	c, use general rtos
lvgl eink rtos	lvgl	c, embedded ui/driver/sdl
lvgl lora loc	graphic	c, embedded openstress/sdl/lvgl
RZOS	rtos	zig, make general rtos
Celest	game	zig, embedded game/sdl
Safephone	electronic	lvgl/eink/openstress/lora/3Dprint
		stm32/nix/electronic/network
		nsfw image/text detected

1. All in all, it’s for 技术哲学
   (a). 网络/低速/时间/域名/ai 过滤/linux 内核构建-r2s
   (b). 屏幕/护眼/低成瘾/驱动设计/eink 屏幕算法-paperlike
   (c). 通信/安全/无依赖/去平台/lora 远距离通信-safephone
   (d). 交通/电工/电子电路/openstress-去五佰[1]那里学修车

2.2 怎么学习底层技术

1. 第一阶段：打基础（C / Linux / GCC / Emacs）
   - C 语言：学会指针、内存管理、结构体、函数指针；刷一些小项目，比如实现 malloc、shell、http server。
   - Linux 基础：熟悉命令行、文件系统、进程/线程、信号、管道、套接字。
   - GCC：学会编译流程 gcc -E/-S/-c/-o，理解预处理、汇编、链接，玩一下 objdump 和 nm。
   - Emacs：把它当 IDE 来用，掌握基本编辑、调试、补全、LSP 支持。

[1]我父亲那边的车行亲戚

2. 第二阶段：系统调试与逆向（GDB / QEMU）
    - GDB：练习断点、单步、查看寄存器/内存、调试多线程/远程调试。
    - QEMU：
        - 用它运行 Linux kernel 或裸机程序。
        - 学会 qemu -S -s + gdb remote 调试，体验调试内核的感觉。
        - 研究 QEMU 的设备模拟（比如 VirtIO、PCI），理解虚拟化和硬件抽象。
3. 第三阶段：进阶编程语言与系统（C++ / Zig / RTOS）
    - Zig：Zig 是现代系统编程语言，学习它的构建系统、内存模型，可以和 C/C++ 混合编程。
    - RTOS：从 FreeRTOS 入手，学任务调度、中断、任务间通信（队列/信号量）。可以用 QEMU 模拟 Cortex-M 板子跑 RTOS。
4. 第四阶段：融会贯通（大型项目 / 内核 / 编译器）
    - QEMU + GDB：调试内核启动、写内核模块。
    - 编译器开发：研究 GCC 或 Clang 的前端/后端；或者用 Zig 写一个简化编译器。
    - 个人项目：比如写一个简易 RTOS，或者在 QEMU 里跑自己写的内核。
5. 操作系统构建与升级（Yocto/Android/内核）在树莓派上交叉编译 Linux 内核，修改驱动或设备树（Device Tree）进行硬件适配。尝试用 Yocto 或 Buildroot 构建自定义 Linux 镜像，加入自己编写或修改的驱动模块。安装和编译 LineageOS（Android for Pi）或类似 Android 系统，修改系统服务或 HAL 层。实践 OTA（Over-The-Air）升级机制，模拟系统升级和回滚操作。硬件 Bringup（CPU/GPU/Memory/Peripherals）利用树莓派的 GPIO、SPI、I2C、CAN、PWM 等接口，练习外设 Bringup 和驱动调试。连接摄像头模块、显示屏（LCD 或 E-Ink）或音频模块，编写驱动和控制程序。使用 perf/ftrace/gprof 分析 CPU/GPU 性能瓶颈，优化程序调度。
6. 系统稳定性与性能优化构建多线程/多进程网络服务，使用 socket 编程实现客户端/服务器通信，模拟并发场景。在树莓派上测试高负载条件下的系统稳定性，分析内核日志、内存占用和 CPU/GPU 使用率。实践内核参数调优，如调节调度器、内存缓存策略，观察性能变化。
7. 客户功能定制与基线升级自己设计一个树莓派应用（如小型智能家居控制器或信息显示终端），从硬件 Bringup 到应用功能定制完整流程。模拟不同版本的系统镜像管理，练习分支合并、基线升级和版本回退。
8. 加分技能训练
    (a). 低功耗优化：通过关闭不必要的外设、调节 CPU/GPU 频率或使用 E-Ink 显示屏练习功耗控制。
    (b). 虚拟化/容器：在树莓派上安装 QEMU/KVM 或 Docker，运行多系统虚拟环境，模拟嵌入式应用部署。
    (c). 芯片平台经验：如果有 MTK 或其他 ARM 板卡，可以对比树莓派练习移植和平台适配经验。

第三章 编辑器哲学

This is my blog and p(pdf)log, and emacs/nixos config, code snippest and even more, all in just one repo. 这是仓库是我的博客也是我写的书，还是我是 emacs 和 nixos 配置，代码模板等等等。

```
(org-babel-do-load-languages
'org-babel-load-languages '((emacs-lisp . t)(shell . t)))
```

```
if [[ -f "/home/$USER/.emacs.d" ]] then
   git clone --depth 1 https://github.com/jamescherti/minimal-emacs.d ~/.emacs.d
fi
ln -sf $PWD/nixos/ ~/.config/;
ln -sf $PWD/snippets ~/.config/;
ln -sf $PWD/nixos/hmdz.pyim ~/.config/;
ln -sf $PWD/nixos/post-init.el ~/.emacs.d/;
if [[ $XAPIAN_CJK_NGRAM != "true" ]] then
  sudo cp /etc/hardware-configuration.nix $PWD/nixos/hardware-configuration.nix
  chmod 777 $PWD/nixos/hardware-configuration.nix
  mkdir -p $HOME/.config/sops/age
  cp .keys.txt $HOME/.config/sops/age/keys.txt
  sudo nix-channel --add https://nixos.org/channels/nixos-unstable
  sudo nix-channel --update
  sudo nixos-rebuild switch --flake /home/$USER/.config/nixos#$hostname --option substituters 'https
      ://mirrors.ustc.edu.cn/nix-channels/store https://cache.nixos.org' --cores 6 -j 12;
fi
```

Listing 3.1: 我能用这代码块重现我的电脑，怎么样？帅吧！

这是 org 的文学编程能力最简单的体现，从 c 到 bash，在一个 org 文件中能一起运行，神奇吧！为什么 emacs 这么强大？大概就是因为它使用的 elisp 相当 “灵” 吧，再说为什么 emacs 这么小众，也大概是它的 elisp 太 “难” 了吧。有些人不敢学，怕自己投入又得不到钞票，有些人不能学，他们把自己的心给了别的教派如 “vim” “vscode” 于是固步自封，自高自大，vim 模态天下第一！我曾同样如此，抱着网上破烂配凑出的 neovim，对那些大神的操作流口水但是又没法认同 emacs。其实犹豫不定很正常，进入一个陌生的环境，很多怪人怪事，emacs 更是如此，你常常不知道那些人口中说的 eglot/flycheck/eww 是什么？常常写错一些代码。当然这同样也是学习 emacs 的乐趣，也正是它的陌生带来的新鲜，而且有了 chatgpt 的帮助，学习起来相当方便。

1. 如果人生还有一次，我还选 emacs

   minimal based on the clean and fast minimal-emacs.

   purezen pure white and black theme written by me called ”purezen”.

   reproducible package manager is using **nixos**.

   simple just a 1500+ lines of post-init.el.

   keyboard for my **ZMK** conifg, the only one soul.

   effcient selecting elegantly like queen and editting fastly like king.

   zen the world is tremendously large, the items are enormously rich, only me along writting with leere feeling, true happiness come from nothing but within.

2. 我的编辑器邪教之旅：

   浏览器里的 vimium C，因为 wsl 学习的 neovim，因为换上 nixos 后抄的 helix 配置，因为大四没找工作太闲学的 emacs。我每用一个就大呼 “我草，早知道，还得是 xx!”，然后给同学老师家人都安利安利，给同学说

“我有一宝”，给老师说 “看我操作”，给家人嘻嘻傻笑像是捡到五百钱。虽然这代表着我非常的闲，但是折腾 emacs 确实让我不是很闲到以至于抑郁，我还是在用 Vimium C 和 emacs，赋予我闪电切换页面的神奇体验，当然，代价是它们也让我得了一种名为 “换页面换 buffer” 的 ADHD，再者，其实不用 vim/emacs 也会得，只不过叫作 “懒得换页面就直接玩手机” ADHD，“懒得折腾电脑就去玩 CSgo 成了换弹瘾” 的 ADHD。在这个时代，玩 csgo 和玩 emacs 有什么区别呢？一个能炒皮肤，另一个能折腾皮肤，如是而矣。

3. 我使用 emacs 没有太多原因就只是:
   (a). 依赖少，之前用 vim，我就要关心 vim 键位的 app 适合用来看 pdf，看 epub 看文档，整 vimium 和 vim 一个键位/什么 lazygit/tmux/fcitx/alacritty，还生怕它们之间有什么矛盾，关键 vimscript 也不适合作为灵活的语言，整天很困扰很憋屈，我还折腾 colemak、虎码和 nixos，我的天，对于任何一个大脑快定型的成年人来说，这简直是噩梦！幸好有 emacs 解救我于快捷键地狱中，装包能 list-package 再一点就安装了，pyim 配置虎码就一行的事，那里还要 fcitx5 怎么怎么样！自从用了 emacs，我再也没有折改过 firefox，一是我没时间，二是我就只用 emacs 的 eww，我也就不再在乎什么 vimium 的键位和 vim 不一样了，**能用就行**才是真理！再说用上了 elisp，那才叫美梦成真，想写一个工具简直就比玩手机打游等更有意思多了。什么 tmux/lazygit/nixos，现在我只想笑笑，我用 vterm/magit/任何平台都能跑 emacs 不是爽多了吗？只要给我 emacs/编译器，写代码就是轻轻松松的事！emacs 改好了我的强迫症和极多主义，现在我心宽又极简，感谢 emacs！
   (b). 功能多，之前用 helix，虽是开箱即用但是很明显它的功能只限于写代码，快是快但是对我这样的学习困难型人来说，快和慢没有多大的区别，只会让我再难受于 “我好慢它好快，我是不是 x 痿了”，helix 是个自带丰富功能工具，但是对于习惯 homerow 的人来说其实用 vim mode 没有什么增益，只会让我更迷茫 “我能直接不动手就能上下了，干啥还要这个 hjkl??”。有插件和服务器模式这点就足以胜过 helix 了。当然我从 helix 的文档也学到了很多编辑知识，如果没有它的 emacs 主题和打算用 scheme 写插件系统，和让我折腾很久的 helix eink 主题，我也不会有潜移默化现在的编辑技术和极简主题审美了。
   (c). 分心小，之前用 vimium 那就是一个换换换的感觉，总是想去换页面，看看新闻看看新工具，上下跳转，左右腾挪，helix 功能太少不能在里面呆太久，而 vimium 是个浏览器插件在里面呆着又太容易去娱乐，~~用 vim 主要是去 youtube 参加编辑器神战~~，helix 也好 vimium 也好。其实我学习 emacs，到能快速掌握 avy 靠得是之前用过 vimium，学习 surrond/expand 靠的是 helix，能熟悉 info/man/tldr 靠得是之前在终端里学习 vim。总之 emacs 必然是编辑技术的一个终点站。
   (d). 定制化，没有一个 term emulator 支持比例字体的，只有 emacs，没有一个 editor 的主题是可以自己完全定制的，vscode 那样的使用 javascript 还难以控制还过于复杂资源又大的就算了。elisp/fsf/clean/simple，学 emacs 之前根本不能理解这话！有些事情得是自己做了才能得之心而应之于手！
   (e). 历史，50 年历史的编辑器，计算机史的一页，还能写 100 年左右。
   (f). 装 b，(linus/stallman) 开源巨人们/(rust/python/java/ruby/curl/各种函数式⋯) 之父们钦定编辑器。

第四章 操作系统哲学

1. network, 471mb hosts to block internet out of my pc with dnsmasq and dae proxy tool based on epbf… you know I am ill about internet? because I am in china, and the nsfw/trump/4chan dudes, I love they so much, so I add they into my hosts.
2. emacs editor, + nixos, they both ruin my early adult life.
3. wm: sway, minimal config, pure white look, although ~~miku cursor~~, now, I use plain mouse instead.
4. shell: fish+foot, with all I need in, direnv/many alias/zoxide/git/and thefxxk or thefxxk updated to ”chatgpt”'s codex nowadays?
5. browser: firefox [default uninstall], I use eww because I hate browser.
6. fully reproducible, even with dict/font/anime wallpaper!
7. zig/linux/ccpp doc in ngnix, all kind texinfo can read in emacs.
8. with sops, to store my 0.1 dollar's deepseek account apikey, and chat with my ai girlfriend with my payment passward.
9. I am tired of nixos, but I can't leave it, because after using nixos eveything is hard to do in other distor, like show off to other distor users ”BTW, I use NixOS”.

4.1 用 linux 只为戒

翻开中国帝王史，有多少帝王是沉迷女色后国家衰亡，多少帝王是皇太后控制后抑郁早亡。野史里选王要的是看他不好色否？当皇帝得看他有无抢夺妇女。翻开网络史发展，有多少人民群众是沉迷情色后国家衰亡，多少人是被色情控制后抑郁早亡，看人得看他好色否？好不好色，得看他懂得怎么预防。oppo 的天气预报能跳到擦边短剧，bing 的结果第一个 BMI 测试广告中跳到黄色导航，百度误输成 baidu.co 中进入黄色应用下载，网易看见过全露的黄色直播，B 站中处处是擦边广告和直播，微博中卖到各种片，知乎里表露情色隐私更引人浏览，我的世界游戏里用色情披风，淘宝里随便看情趣用品，openai 支持成人内容，twitter 全是福利姬，就连 QQ 空间连上擦边广告商，onlyfans 比所有 ai 公司利润更高，pornhub 全球流量前 10，整个 00 后的网络被“这到提醒我了”占据，整代女人被商业文化营造黑丝和大胸的“性感”，整代男人被情色文化成为无脑动物与被情欲“冲动”控制，男人就该好色？女人就该这样美吗？天下皆知美之为美，斯恶也。互联网 33% 的流量，70% 的男性，30% 的女性，90% 男性沉迷其中···数字不会说谎，一切已经到来，生存与繁衍不再重要，存在不过是不断刺激感官。不只是色情···网络带来的问题似乎总是和成瘾有关，赌博/购物/社交/游戏···就连互联网最自豪的正面形象所谓的“利于学习”也成为了信息过载的社会巨大问题，自由的代价是什么？自由即是强迫。

论我怎么戒色？答：学习技术、理解技术、控制技术，学习自由软件，远离商业软件，拒绝即是自由。

- 成瘾源于：随机 | 不必 | 即时 | 匿名 | 易得 | 免费 | 广告 | 失范
- 通过 **路由器 +NixOS** ，不使用手机与笔记本

  时间 限时，2-5PM，足够一天专注

  速度 网速，1Mbps，足够文本传输

  空间 域名，非编程网站，471MB

  强制 禁 cache.nixos.org 以止系统更新
- 软件使用 **emacs** ，不使用浏览器与 IDE

  google tavily

  chatgpt gptel

  github consult-gh

  leetcode leetgo

  firefox eww

  vpn dae/dnsmasq

obsidian org

wiki nginx/info/tldr/devdocs/man

- 硬件使用 **eink**，不使用音响与普通键盘

  音乐 无音响

  游游 无显卡

  动漫 无色彩

  社媒 无软件

  购物 无网络

  视频 竖立屏

第五章 键盘流哲学

5.1 设计哲学

- 与原生相合 (win)
- 不冲突 (special prefix)
- 合乎习惯 (stay base)
- home row 效率 (home row effiction)
- general in os(browser, editor)

5.2 分层

1. base 层，colemak-dh，最快速与舒适的英文布局
2. nav 层，上下左右，快速移动，C-c C-~ 为前缀，emacs 快速选择
3. num 层，数字键，控制 sway 的窗口，与系统剪切板和应用菜单
4. mos 层，鼠标上下左右，C-c C-; 为前缀，一些 emacs 功能。
5. sym 层，全部符号，gpt prompt 与个人信息快速输入层
6. fun 层，function 1/2../12 层
7. med 层，提供蓝牙/媒体切换

5.3 本地刷录

```
  #!/usr/bin/env bash
# ===== CONFIGURATION =====
GITHUB_OWNER="NestorLiao"
GITHUB_REPO="zmk-config"
GITHUB_TOKEN="$(cat␣~/.github_token)"
ZIP_DEST="./zmk_build.zip"
UNZIP_DIR="./zmk_build"

LEFT_FW="corne_left-nice_nano_v2-zmk.uf2"
RIGHT_FW="corne_right-nice_nano_v2-zmk.uf2"
SETTINGS_RESET="settings_reset-nice_nano_v2-zmk.uf2"

# ===== FUNCTIONS =====

flash_settings_reset() {
    echo "Flashing␣settings␣reset␣firmware..."

    for i in 1 2; do
        echo "Reset␣device␣$i/2␣into␣bootloader␣mode..."

        # Wait for mount
        while [ ! -d "/media/NICENANO" ]; do
            sleep 0.5
        done

        echo "NICENANO␣detected.␣Flashing␣settings␣reset..."
        if [ -f "$SETTINGS_RESET" ]; then
            cp "$SETTINGS_RESET" /media/NICENANO/
            echo "Settings␣reset␣flashed␣$i/2."
        else
            echo "␣$SETTINGS_RESET␣not␣found␣in␣current␣directory."
            exit 1
        fi

        echo "Waiting␣for␣NICENANO␣to␣unmount..."
        while [ -d "/media/NICENANO" ]; do
            sleep 0.5
        done
    done
    echo "␣Settings␣reset␣completed␣twice."
}

download_latest_artifact() {
    echo "Fetching␣latest␣artifact..."

    run_id=$(curl -s -H "Authorization:␣token␣$GITHUB_TOKEN" \
        "https://api.github.com/repos/$GITHUB_OWNER/$GITHUB_REPO/actions/runs?per_page=1&status=success
             " \
        | jq -r '.workflow_runs[0].id')

    if [ "$run_id" == "null" ] || [ -z "$run_id" ]; then
        echo "␣No␣successful␣workflow␣run␣found."
        exit 1
    fi

    echo "Found␣workflow␣run␣ID:␣$run_id"

    artifact_url=$(curl -s -H "Authorization:␣token␣$GITHUB_TOKEN" \
        "https://api.github.com/repos/$GITHUB_OWNER/$GITHUB_REPO/actions/runs/$run_id/artifacts" \
        | jq -r '.artifacts[0].archive_download_url')

    if [ "$artifact_url" == "null" ] || [ -z "$artifact_url" ]; then
        echo "␣No␣artifacts␣found␣for␣run␣$run_id."
        exit 1
    fi

    echo "Downloading␣artifact␣zip..."
    curl -L -H "Authorization:␣token␣$GITHUB_TOKEN" \
        "$artifact_url" -o "$ZIP_DEST"

    echo "Unzipping..."
    rm -rf "$UNZIP_DIR"
    unzip -q "$ZIP_DEST" -d "$UNZIP_DIR"
}

flash_from_local() {
    if [ ! -f "$ZIP_DEST" ]; then
        echo "␣$ZIP_DEST␣not␣found␣in␣current␣directory."
        exit 1
    fi

    echo "Using␣local␣$ZIP_DEST..."
    rm -rf "$UNZIP_DIR"
    unzip -q "$ZIP_DEST" -d "$UNZIP_DIR"
}

flash_firmware() {
    local fw_path="$1"
    echo "Please␣reset␣the␣device␣into␣bootloader␣mode..."

    while [ ! -d "/media/NICENANO" ]; do
        sleep 0.5
    done

    echo "NICENANO␣detected.␣Flashing␣$fw_path..."
    cp "$fw_path" /media/NICENANO/
    echo "Done."

    echo "Waiting␣for␣NICENANO␣to␣unmount..."
    while [ -d "/media/NICENANO" ]; do
        sleep 0.5
    done
}

flash_left_right() {
    flash_firmware "$UNZIP_DIR/$LEFT_FW"
    flash_firmware "$UNZIP_DIR/$RIGHT_FW"
}

show_usage() {
    echo "Usage:␣$0␣<mode>"
    echo "Modes:"
    echo "␣␣1␣-␣Flash␣settings␣reset␣twice"
    echo "␣␣2␣-␣Download␣from␣GitHub␣and␣flash␣left/right"
    echo "␣␣3␣-␣Use␣local␣zmk_build.zip␣to␣flash␣left/right"
}

# ===== MAIN =====

if [ $# -ne 1 ]; then
    show_usage
    exit 1
fi

case $1 in
    1)
        echo "==␣Mode␣1:␣Flash␣Settings␣Reset␣=="
        flash_settings_reset
        ;;
    2)
        echo "==␣Mode␣2:␣Download␣and␣Flash␣=="
        download_latest_artifact
        flash_left_right
        ;;
    3)
        echo "==␣Mode␣3:␣Local␣Flash␣=="
        flash_from_local
        flash_left_right
        ;;
    *)
        echo " ␣Invalid␣mode:␣$1"
        show_usage
        exit 1
        ;;
esac

echo " ␣All␣done."
```

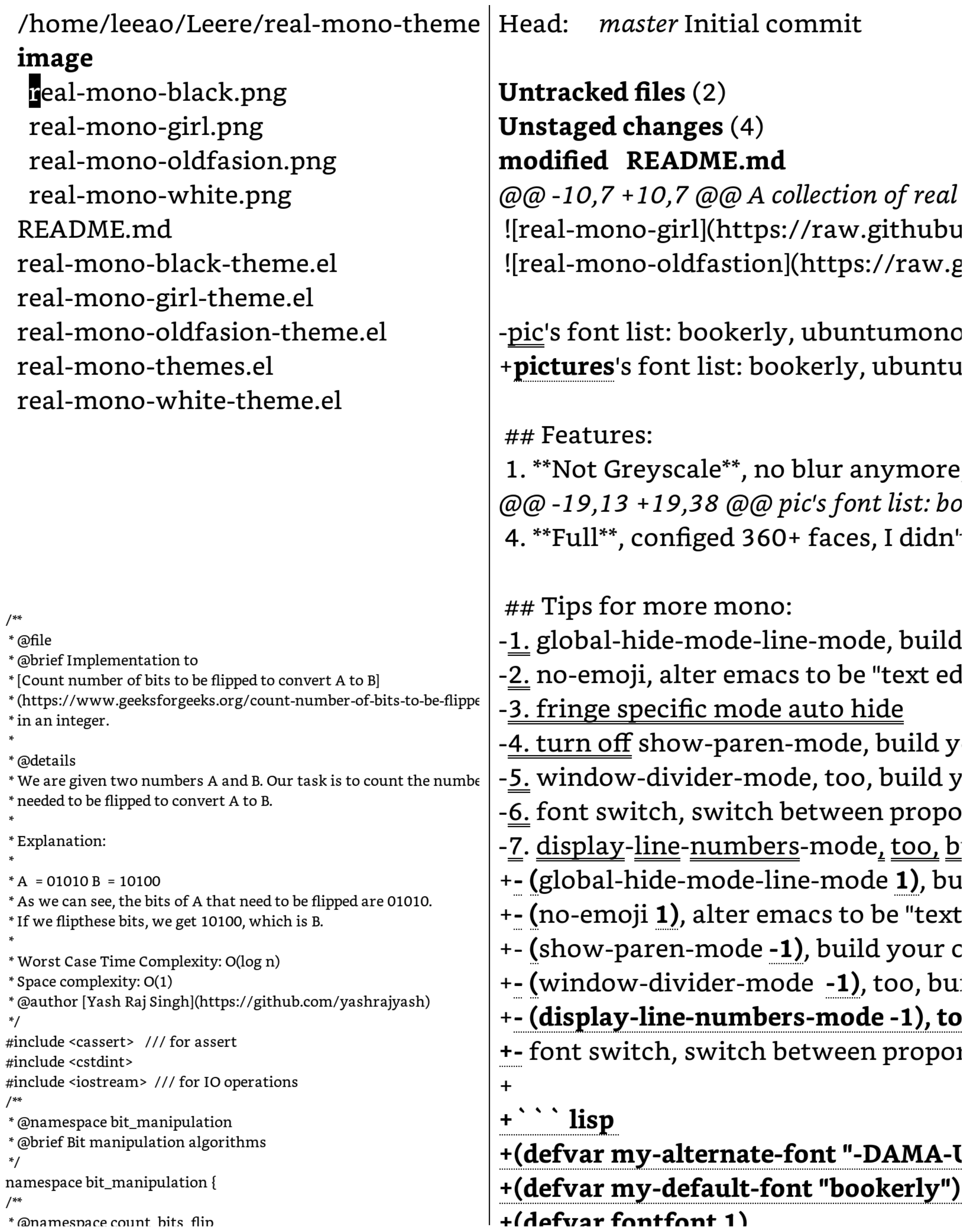
Listing 5.1: 1 刷重置，2 刷 github action，3 刷本地 zip 文件

第六章 屏幕哲学

Real Mono Theme, two colors are enough for emacs.

6.1 Feels:

A collection of real monochrome emacs themes in a couple of variants.



```
/home/leeao/Leere/real-mono-theme
image
  real-mono-black.png
  real-mono-girl.png
  real-mono-oldfasion.png
  real-mono-white.png
README.md
real-mono-black-theme.el
real-mono-girl-theme.el
real-mono-oldfasion-theme.el
real-mono-themes.el
real-mono-white-theme.el
```

```
/**
 * @file
 * @brief Implementation to
 * [Count number of bits to be flipped to convert A to B]
 * (https://www.geeksforgeeks.org/count-number-of-bits-to-be-flippe
 * in an integer.
 *
 * @details
 * We are given two numbers A and B. Our task is to count the numbe
 * needed to be flipped to convert A to B.
 *
 * Explanation:
 *
 * A  = 01010 B  = 10100
 * As we can see, the bits of A that need to be flipped are 01010.
 * If we flipthese bits, we get 10100, which is B.
 *
 * Worst Case Time Complexity: O(log n)
 * Space complexity: O(1)
 * @author [Yash Raj Singh](https://github.com/yashrajyash)
 */
#include <cassert>   /// for assert
#include <cstdint>
#include <iostream>  /// for IO operations
/**
 * @namespace bit_manipulation
 * @brief Bit manipulation algorithms
 */
namespace bit_manipulation {
/**
 * @namespace count_bits_flip
```

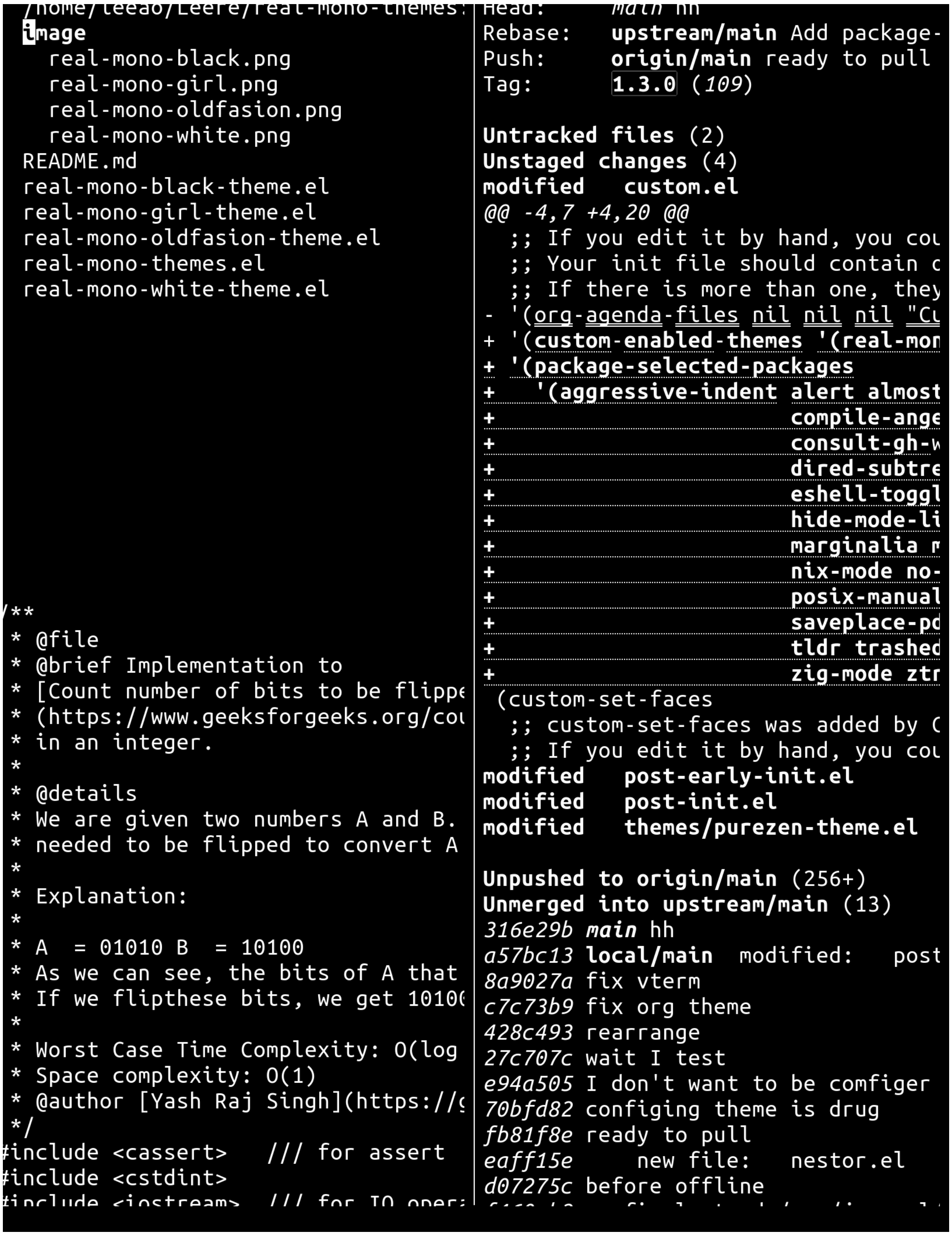
```
Head:     master Initial commit

Untracked files (2)
Unstaged changes (4)
modified   README.md
@@ -10,7 +10,7 @@ A collection of real
 ![real-mono-girl](https://raw.githubu
 ![real-mono-oldfastion](https://raw.g

-pic's font list: bookerly, ubuntumono
+pictures's font list: bookerly, ubuntu

 ## Features:
 1. **Not Greyscale**, no blur anymore
@@ -19,13 +19,38 @@ pic's font list: bo
 4. **Full**, configed 360+ faces, I didn'

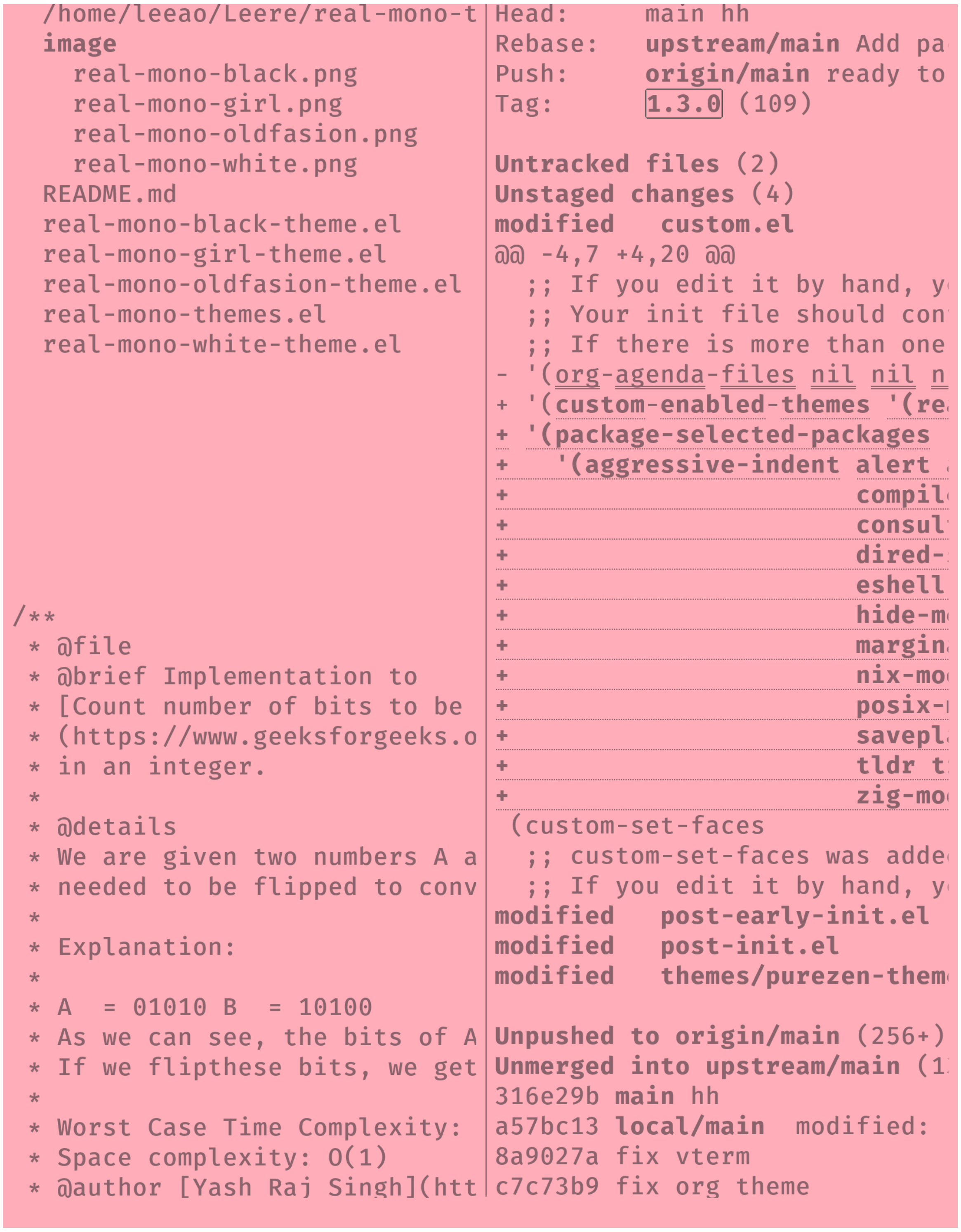
 ## Tips for more mono:
-1. global-hide-mode-line-mode, build
-2. no-emoji, alter emacs to be "text ed
-3. fringe specific mode auto hide
-4. turn off show-paren-mode, build y
-5. window-divider-mode, too, build y
-6. font switch, switch between propo
-7. display-line-numbers-mode, too, b
+- (global-hide-mode-line-mode 1), bu
+- (no-emoji 1), alter emacs to be "text
+- (show-paren-mode -1), build your c
+- (window-divider-mode  -1), too, bu
+- (display-line-numbers-mode -1), to
+- font switch, switch between propo
+
+```lisp
+(defvar my-alternate-font "-DAMA-U
+(defvar my-default-font "bookerly")
+(defvar fontfont 1)
```



```
 /home/leeao/Leere/real-mono-themes:
 image
   real-mono-black.png
   real-mono-girl.png
   real-mono-oldfasion.png
   real-mono-white.png
 README.md
 real-mono-black-theme.el
 real-mono-girl-theme.el
 real-mono-oldfasion-theme.el
 real-mono-themes.el
 real-mono-white-theme.el
```

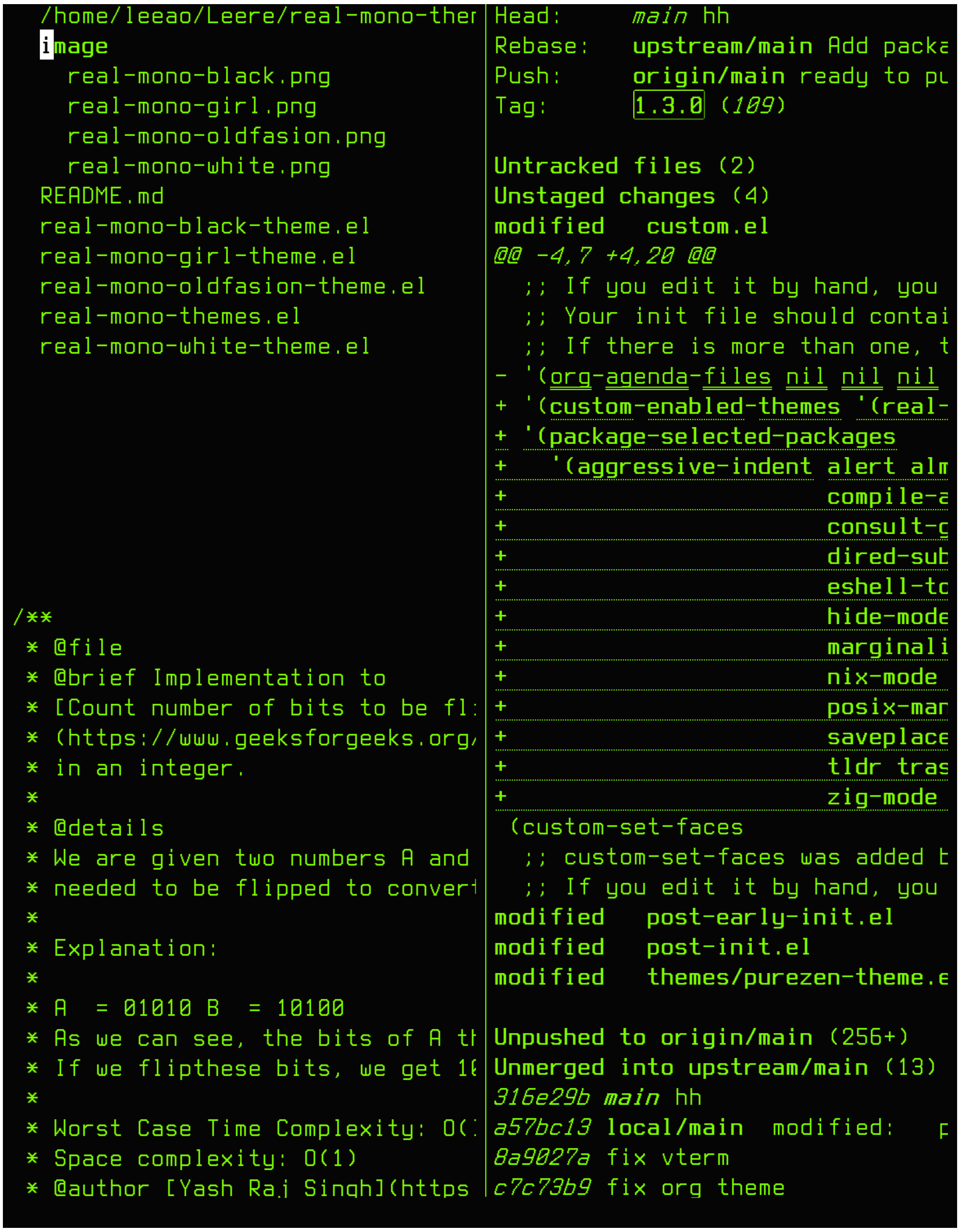
```
/**
 * @file
 * @brief Implementation to
 * [Count number of bits to be flippe
 * (https://www.geeksforgeeks.org/cou
 * in an integer.
 *
 * @details
 * We are given two numbers A and B.
 * needed to be flipped to convert A
 *
 * Explanation:
 *
 * A  = 01010 B  = 10100
 * As we can see, the bits of A that
 * If we flipthese bits, we get 1010(
 *
 * Worst Case Time Complexity: O(log
 * Space complexity: O(1)
 * @author [Yash Raj Singh](https://g
 */
#include <cassert>   /// for assert
#include <cstdint>
#include <iostream>  /// for IO opera
```

```
Head:      main hh
Rebase:    upstream/main Add package-
Push:      origin/main ready to pull
Tag:       1.3.0 (109)

Untracked files (2)
Unstaged changes (4)
modified   custom.el
@@ -4,7 +4,20 @@
  ;; If you edit it by hand, you cou
  ;; Your init file should contain c
  ;; If there is more than one, they
- '(org-agenda-files nil nil nil "Cu
+ '(custom-enabled-themes '(real-mon
+ '(package-selected-packages
+   '(aggressive-indent alert almost
+                        compile-ange
+                        consult-gh-w
+                        dired-subtre
+                        eshell-toggl
+                        hide-mode-li
+                        marginalia m
+                        nix-mode no-
+                        posix-manual
+                        saveplace-pd
+                        tldr trashed
+                        zig-mode ztr
 (custom-set-faces
  ;; custom-set-faces was added by C
  ;; If you edit it by hand, you cou
modified   post-early-init.el
modified   post-init.el
modified   themes/purezen-theme.el

Unpushed to origin/main (256+)
Unmerged into upstream/main (13)
316e29b main hh
a57bc13 local/main  modified:   post
8a9027a fix vterm
c7c73b9 fix org theme
428c493 rearrange
27c707c wait I test
e94a505 I don't want to be comfiger
70bfd82 configing theme is drug
fb81f8e ready to pull
eaff15e     new file:   nestor.el
d07275c before offline
```



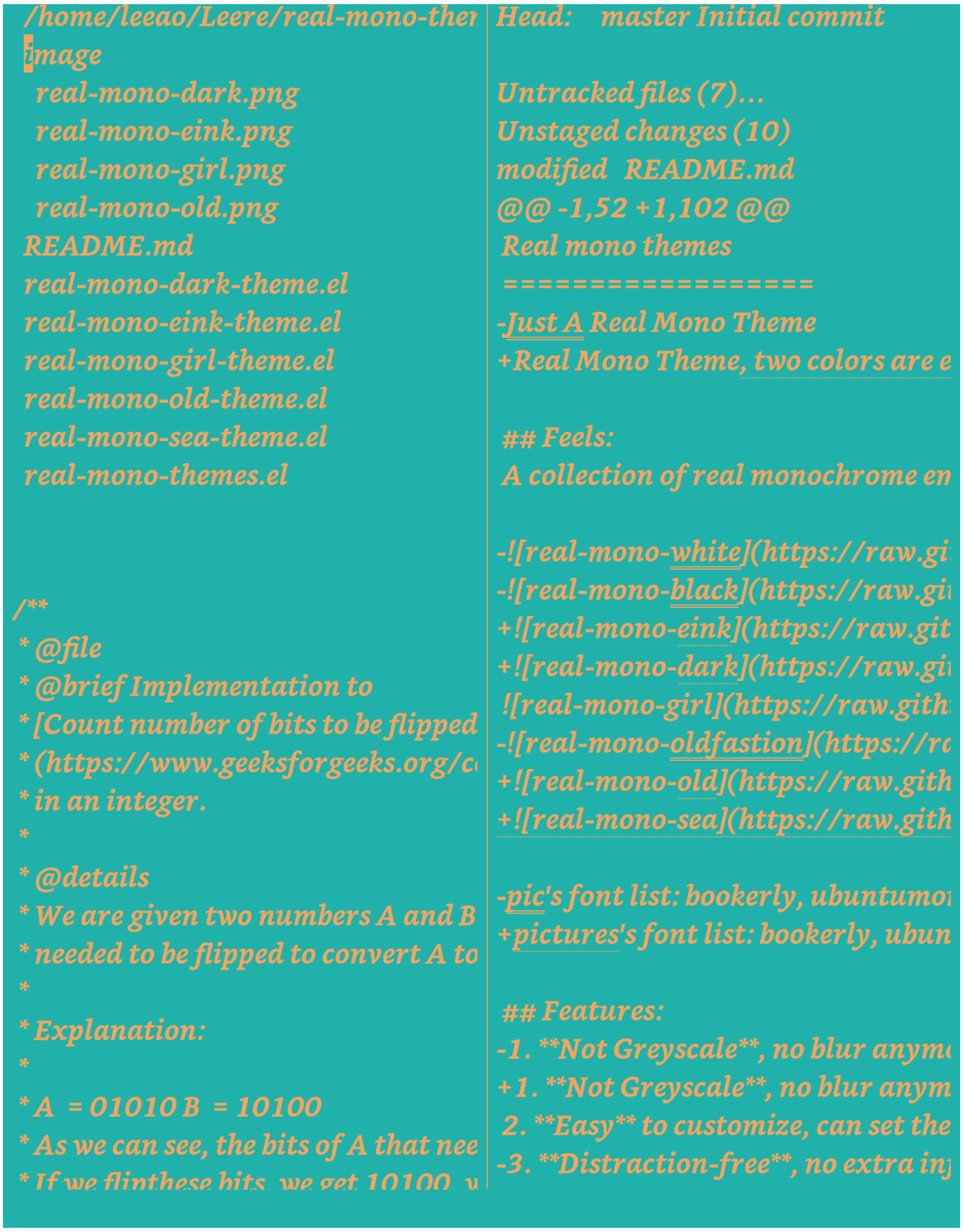
```
  /home/leeao/Leere/real-mono-t
  image
    real-mono-black.png
    real-mono-girl.png
    real-mono-oldfasion.png
    real-mono-white.png
  README.md
  real-mono-black-theme.el
  real-mono-girl-theme.el
  real-mono-oldfasion-theme.el
  real-mono-themes.el
  real-mono-white-theme.el








/**
 * @file
 * @brief Implementation to
 * [Count number of bits to be
 * (https://www.geeksforgeeks.o
 * in an integer.
 *
 * @details
 * We are given two numbers A a
 * needed to be flipped to conv
 *
 * Explanation:
 *
 * A  = 01010 B  = 10100
 * As we can see, the bits of A
 * If we flipthese bits, we get
 *
 * Worst Case Time Complexity:
 * Space complexity: O(1)
 * @author [Yash Raj Singh](htt
```

```
Head:     main hh
Rebase:   upstream/main Add pa
Push:     origin/main ready to
Tag:      1.3.0 (109)

Untracked files (2)
Unstaged changes (4)
modified   custom.el
@@ -4,7 +4,20 @@
  ;; If you edit it by hand, y
  ;; Your init file should con
  ;; If there is more than one
- '(org-agenda-files nil nil n
+ '(custom-enabled-themes '(re
+ '(package-selected-packages
+   '(aggressive-indent alert
+                        compil
+                        consul
+                        dired-
+                        eshell
+                        hide-m
+                        margin
+                        nix-mo
+                        posix-
+                        savepl
+                        tldr t
+                        zig-mo
 (custom-set-faces
  ;; custom-set-faces was adde
  ;; If you edit it by hand, y
modified   post-early-init.el
modified   post-init.el
modified   themes/purezen-them

Unpushed to origin/main (256+)
Unmerged into upstream/main (1
316e29b main hh
a57bc13 local/main  modified:
8a9027a fix vterm
c7c73b9 fix org theme
```



```
  /home/leeao/Leere/real-mono-ther
  image
    real-mono-black.png
    real-mono-girl.png
    real-mono-oldfasion.png
    real-mono-white.png
  README.md
  real-mono-black-theme.el
  real-mono-girl-theme.el
  real-mono-oldfasion-theme.el
  real-mono-themes.el
  real-mono-white-theme.el









/**
 * @file
 * @brief Implementation to
 * [Count number of bits to be fl:
 * (https://www.geeksforgeeks.org/
 * in an integer.
 *
 * @details
 * We are given two numbers A and
 * needed to be flipped to convert
 *
 * Explanation:
 *
 * A  = 01010 B  = 10100
 * As we can see, the bits of A th
 * If we flipthese bits, we get 1(
 *
 * Worst Case Time Complexity: O(:
 * Space complexity: O(1)
 * @author [Yash Raj Singh](https
```

```
Head:      main hh
Rebase:    upstream/main Add packa
Push:      origin/main ready to pu
Tag:       1.3.0 (109)

Untracked files (2)
Unstaged changes (4)
modified   custom.el
@@ -4,7 +4,20 @@
  ;; If you edit it by hand, you
  ;; Your init file should contai
  ;; If there is more than one, t
- '(org-agenda-files nil nil nil
+ '(custom-enabled-themes '(real-
+ '(package-selected-packages
+   '(aggressive-indent alert alm
+                       compile-a
+                       consult-g
+                       dired-sub
+                       eshell-tc
+                       hide-mode
+                       marginali
+                       nix-mode
+                       posix-mar
+                       saveplace
+                       tldr tras
+                       zig-mode
 (custom-set-faces
  ;; custom-set-faces was added b
  ;; If you edit it by hand, you
modified   post-early-init.el
modified   post-init.el
modified   themes/purezen-theme.e

Unpushed to origin/main (256+)
Unmerged into upstream/main (13)
316e29b main hh
a57bc13 local/main  modified:   p
8a9027a fix vterm
c7c73b9 fix org theme
```



```
/home/leeao/Leere/real-mono-the
image
  real-mono-dark.png
  real-mono-eink.png
  real-mono-girl.png
  real-mono-old.png
README.md
real-mono-dark-theme.el
real-mono-eink-theme.el
real-mono-girl-theme.el
real-mono-old-theme.el
real-mono-sea-theme.el
real-mono-themes.el



/**
 * @file
 * @brief Implementation to
 * [Count number of bits to be flipped
 * (https://www.geeksforgeeks.org/c
 * in an integer.
 *
 * @details
 * We are given two numbers A and B
 * needed to be flipped to convert A to
 *
 * Explanation:
 *
 * A  = 01010 B  = 10100
 * As we can see, the bits of A that nee
 * If we flipthese bits, we get 10100, 
```

```
Head:    master Initial commit

Untracked files (7)...
Unstaged changes (10)
modified   README.md
@@ -1,52 +1,102 @@
 Real mono themes
 ==================
-Just A Real Mono Theme
+Real Mono Theme, two colors are e

 ## Feels:
 A collection of real monochrome en

-![real-mono-white](https://raw.gi
-![real-mono-black](https://raw.gi
+![real-mono-eink](https://raw.git
+![real-mono-dark](https://raw.gi
 ![real-mono-girl](https://raw.gith
-![real-mono-oldfastion](https://r
+![real-mono-old](https://raw.gith
+![real-mono-sea](https://raw.gith

-pic's font list: bookerly, ubuntumo
+pictures's font list: bookerly, ubun

 ## Features:
-1. **Not Greyscale**, no blur anym
+1. **Not Greyscale**, no blur anym
 2. **Easy** to customize, can set the
-3. **Distraction-free**, no extra in
```

pictures's font list: bookerly, ubuntumono, firacode, terminess, bookerly blod ltalic.

6.2 Features:

1. **Not Greyscale**, no blur anymore, it's much better to use eink screen for pure black and white!
2. **Easy** to customize, can set the only two colors by config the default face's foreground/background color.
3. **Distraction-free**, no extra info-overwhelming causing by font-lock, only few font diversity in magit/dired etc for better function recognize.

4. **Full**, configed 370+ faces, I didn't see any monochrome theme can reach that much, as my daily driver, it's good enough.

6.3 Tips for mono:

- (global-hide-mode-line-mode 1), build your own brain memory
- (no-emoji 1), alter emacs to be "text editor" instead of discord
- (show-paren-mode -1), build your own eye insight
- (window-divider-mode -1), too, build your memory
- (display-line-numbers-mode -1), too, build your own eye insight

```
(defvar my-alternate-font "-DAMA-UbuntuMono␣Nerd␣Font-regular-normal-normal-*-13-*-*-*-m-0-iso10646-1"
    )
(defvar my-default-font "bookerly")
(defvar fontfont 1)
(defun my-toggle-font ()
  "Toggle␣between␣UbuntuMono␣and␣bookerly␣fonts."
  (interactive)
  (if (= fontfont 1)
      (progn (set-face-attribute 'default nil :font my-default-font :height 160) (setq fontfont 0))
    (progn (set-face-attribute 'default nil :font my-alternate-font :height 210) (setq fontfont 1))))
```

- fringe specific mode auto hide

```
(defun my-set-fringe-face ()
  "auto␣hide␣fringe␣face␣depending␣on␣major␣mode."
  (if (derived-mode-p '(occur-mode gud-mode))
      (set-face-attribute 'fringe nil
                          :background (face-attribute 'default :background)
                          :foreground (face-attribute 'default :foreground))
    (set-face-attribute 'fringe nil
                        :background (face-attribute 'default :background)
                        :foreground (face-attribute 'default :background))))
(add-hook 'after-change-major-mode-hook #'my-set-fringe-face)
```

- elisp for toggling paperlike-hd to swtich between read and watch.

```
(defvar monitor-state 0
  "Current␣monitor␣state,␣either␣0␣for␣read␣or␣␣1␣for␣watch.")
(defun monitor ()
  "swtich␣monitor␣from␣read␣mode␣to␣watch␣mode"
  (interactive)
  (let((monitorpath "-i2c␣␣/dev/i2c-4")
       (monitorcli "paperlike-cli␣")
       (monitorarg '("␣-contrast␣" "␣-speed␣" "␣-mode␣" "␣-clear"))
       (mode-state '(("9" "5" "1") ("2" "1" "3"))))
    (split-window-below)
    (other-window 1)
    (switch-to-buffer "*Shell␣Command␣Output*")
    (split-window-below)
(other-window 1)
(switch-to-buffer "*Async␣Shell␣Command*")
(progn
  (dotimes (number 3)
    (shell-command (concat
                    monitorcli
                    monitorpath
                    (car (nthcdr number monitorarg))
                    (car (nthcdr number (car (nthcdr monitor-state mode-state)))))
                   ))
  (sleep-for 1))
(setq monitor-state (if (= 0 monitor-state) 1 0 ))
(sleep-for 1.5)
(sleep-for 0.5)
(async-shell-command (concat monitorcli monitorpath "␣-clear"))
(let ((async (get-buffer-window "*Async␣Shell␣Command*"))
      (shell (get-buffer-window "*Shell␣Command␣Output*")))
  (when async (delete-window async))
  (when shell (delete-window shell)))
(sleep-for 0.5)
(kill-buffer "*Async␣Shell␣Command*")
(kill-buffer "*Shell␣Command␣Output*")
(donothing)))
```

第七章 TODO fag.h

Copied from tsoding's flag module: https://github.com/tsoding/flag.h

7.1 Quick Start

Check example.c

```
cc -o example example.c
./example -help
```

第八章 不用 AI 完成 Leetcode 刷题

内容提要

- ❏ 开始时间：2025-11-20
- ❏ 结束时间：2025-12 月前
- ❏ 完成难度：difficult
- ❏ 需要小时：100+
- ❏ 本章要点：把力扣算法题热题都刷完一遍

不用 AI 完成 Leetcode 刷题是一件很爽的事，虽然产生的经济效应为零，虽然永远写代码写不过 AI，但是 ⋯ 人类还需要人类? 不知资本家怎么想，但有搜索引擎了，就不用背书了吗，有炒菜机和机车就不用做菜和走路了吗? 得之心而应于手，理解一件事情本身有它的价值，虽然我更想去种田/木工/画画/乐器/写作，但⋯我得先有工作啊 ~

8.1 1. Two Sum (Easy)

Given an array of integers `nums` and an integer `target`, return indices of the two numbers such that they add up to `target`.

You may assume that each input would have **exactly one solution**, and you may not use the same element twice.

You can return the answer in any order.

Example 1:

```
Input: nums = [2,7,11,15], target = 9
Output: [0,1]
Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].
```

Example 2:

```
Input: nums = [3,2,4], target = 6
Output: [1,2]
```

Example 3:

```
Input: nums = [3,3], target = 6
Output: [0,1]
```

Constraints:

- `2` < nums.length <= 10^4=
- `-10` < nums[i] <= 10^9=
- `-10` < target <= 10^9=
- **Only one valid answer exists.**

Follow-up: Can you come up with an algorithm that is less than `O(n²)` time complexity?

1. Solution

```
// -*- compile-command: "make -f ../Makefile submit" -*-
// Created by NestorLiao at 2025/11/21 14:58
// leetgo: 1.4.15
// https://leetcode.com/problems/two-sum/
```

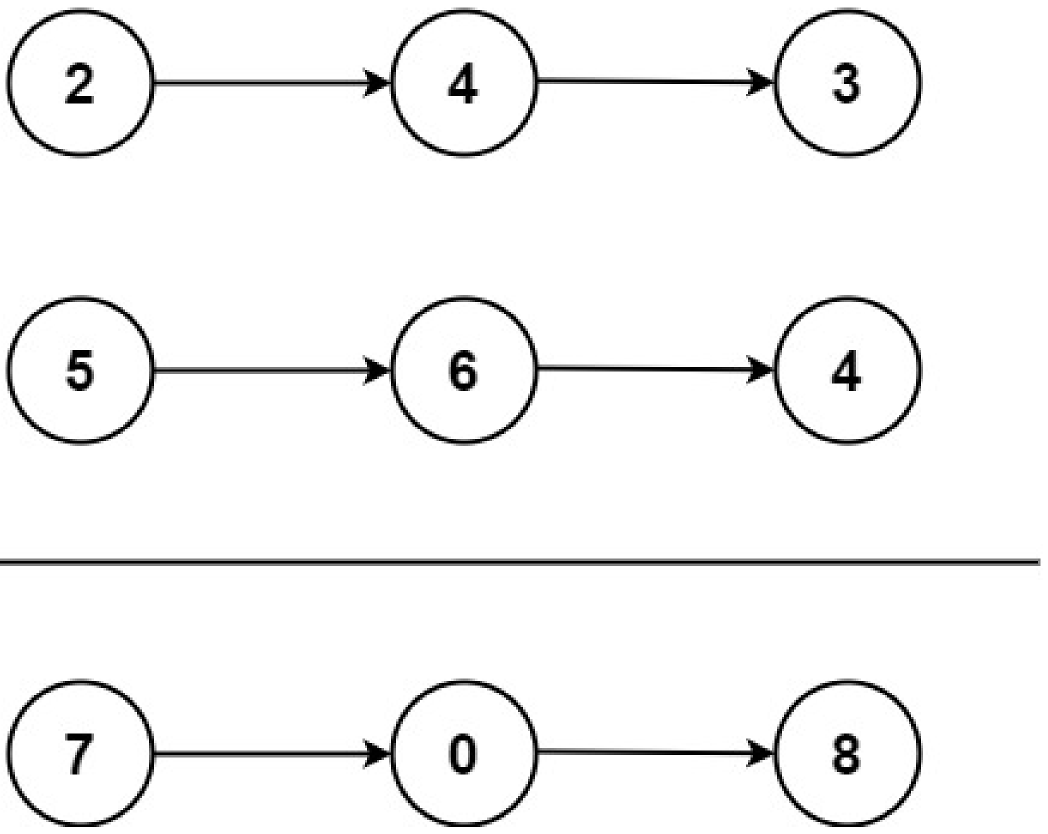
```
#include <bits/stdc++.h>
#include "LC_IO.h"
using namespace std;

// @lc code=begin

class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {

    }
};

// @lc code=end

int main() {
    ios_base::sync_with_stdio(false);
    stringstream out_stream;

    vector<int> nums;
    LeetCodeIO::scan(cin, nums);
    int target;
    LeetCodeIO::scan(cin, target);

    Solution *obj = new Solution();
    auto res = obj->twoSum(nums, target);
    LeetCodeIO::print(out_stream, res);
    cout << "\noutput: " << out_stream.rdbuf() << endl;

    delete obj;
    return 0;
}
```

8.2 2. Add Two Numbers (Medium)

You are given two **non-empty** linked lists representing two non-negative integers. The digits are stored in **reverse order**, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Example 1:



```
Input: l1 = [2,4,3], l2 = [5,6,4]
Output: [7,0,8]
Explanation: 342 + 465 = 807.
```

Example 2:

```
Input: l1 = [0], l2 = [0]
Output: [0]
```

Example 3:

```
Input: l1 = [9,9,9,9,9,9,9], l2 = [9,9,9,9]
Output: [8,9,9,9,0,0,0,1]
```

Constraints:

- The number of nodes in each linked list is in the range `[1, 100]`.
- `0` < Node.val <= 9=
- It is guaranteed that the list represents a number that does not have leading zeros.

1. Solution

```
// -*- compile-command: "make -f ../Makefile submit" -*-
// Created by NestorLiao at 2025/11/21 14:58
// leetgo: 1.4.15
// https://leetcode.com/problems/add-two-numbers/

#include <bits/stdc++.h>
#include "LC_IO.h"
using namespace std;
```

```
// @lc code=begin

class Solution {
public:
    ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {

    }
};

// @lc code=end

int main() {
    ios_base::sync_with_stdio(false);
    stringstream out_stream;

    ListNode* l1;
    LeetCodeIO::scan(cin, l1);
    ListNode* l2;
    LeetCodeIO::scan(cin, l2);

    Solution *obj = new Solution();
    auto res = obj->addTwoNumbers(l1, l2);
    LeetCodeIO::print(out_stream, res);
    cout << "\noutput: " << out_stream.rdbuf() << endl;

    delete obj;
    return 0;
}
```

8.3 3. Longest Substring Without Repeating Characters (Medium)

Given a string s, find the length of the **longest substring** without duplicate characters.

Example 1:

```
Input: s = "abcabcbb"
Output: 3
Explanation: The answer is "abc", with the length of 3. Note that "bca" and "cab" are also correct
answers.
```

Example 2:

```
Input: s = "bbbbb"
Output: 1
Explanation: The answer is "b", with the length of 1.
```

Example 3:

```
Input: s = "pwwkew"
Output: 3
Explanation: The answer is "wke", with the length of 3.
Notice that the answer must be a substring, "pwke" is a subsequence and not a substring.
```

Constraints:

- 0 < s.length <= 5 * 10^4=
- s consists of English letters, digits, symbols and spaces.

1. Solution

```cpp
// -*- compile-command: "make -f ../Makefile submit" -*-
// Created by NestorLiao at 2025/11/21 14:58
// leetgo: 1.4.15
// https://leetcode.com/problems/longest-substring-without-repeating-characters/

#include <bits/stdc++.h>
#include "LC_IO.h"
using namespace std;

// @lc code=begin

class Solution {
public:
   int lengthOfLongestSubstring(string s) {

   }
};

// @lc code=end

int main() {
   ios_base::sync_with_stdio(false);
   stringstream out_stream;

   string s;
   LeetCodeIO::scan(cin, s);

   Solution *obj = new Solution();
   auto res = obj->lengthOfLongestSubstring(s);
   LeetCodeIO::print(out_stream, res);
   cout << "\noutput: " << out_stream.rdbuf() << endl;

   delete obj;
   return 0;
}
```

8.4 设计刷题环境

8.4.1 leetgo

本来打算刷多门语言的，现在看了还是太高估自己了，把 cpp 刷好就可以了。

```
# Leetgo configuration file, see more at https://github.com/j178/leetgo

# Your name
author: NestorLiao
# Language of the question description: 'zh' (Simplified Chinese) or 'en' (English).
language: en
code:
  # Language of code generated for questions: go, cpp, python, java...
  # (will be overridden by command line flag -l/--lang).
  lang: cpp
  # The default template to generate filename (without extension), e.g. {{.Id}}.{{.Slug}}
  # Available attributes: Id, Slug, Title, Difficulty, Lang, SlugIsMeaningful
  # (Most questions have descriptive slugs, but some consist of random characters. The
      SlugIsMeaningful boolean indicates whether a slug is meaningful.)
  # Available functions: lower, upper, trim, padWithZero, toUnderscore, group.
  filename_template: '{{ .Id | padWithZero 4 }}{{ if .SlugIsMeaningful }}.{{ .Slug }}{{ end }}'
  # Generate question description into a separate question.md file, otherwise it will be embed in the
      code file.
  separate_description_file: true
  # Default modifiers for all languages.
  modifiers:
    - name: removeUselessComments
  go:
    # Base directory to put generated questions, defaults to the language slug, e.g. go, python, cpp.
    out_dir: go
    # Functions that modify the generated code.
    modifiers:
      - name: removeUselessComments
      - name: changeReceiverName
      - name: addNamedReturn
      - name: addMod
  python3:
    # Base directory to put generated questions, defaults to the language slug, e.g. go, python, cpp.
    out_dir: python
    # Path to the python executable that creates the venv.
    executable: python3
  cpp:
    # Base directory to put generated questions, defaults to the language slug, e.g. go, python, cpp.
    out_dir: src
    # C++ compiler
    cxx: g++
    # C++ compiler flags (our Leetcode I/O library implementation requires C++17).
    cxxflags: -O2 -std=c++17
  rust:
    # Base directory to put generated questions, defaults to the language slug, e.g. go, python, cpp.
    out_dir: .
  java:
    # Base directory to put generated questions, defaults to the language slug, e.g. go, python, cpp.
    out_dir: java
leetcode:
  # LeetCode site, https://leetcode.com or https://leetcode.cn
  site: https://leetcode.com
  # Credentials to access LeetCode.
  credentials:
    # How to provide credentials: browser, cookies, password or none.
    from:
      - browser
    # Browsers to get cookies from: chrome, safari, edge or firefox. If empty, all browsers will be
        tried. Only used when 'from' is 'browser'.
    browsers: [firefox]
contest:
  # Base directory to put generated contest questions.
  out_dir: contest
  # Template to generate filename of the question.
  filename_template: '{{ .ContestShortSlug }}/{{ .Id }}{{ if .SlugIsMeaningful }}.{{ .Slug }}{{ end
      }}'
  # Open the contest page in browser after generating.
  open_in_browser: true
# Editor settings to open generated files.
editor:
  # Use a predefined editor: vim, vscode, goland
  # Set to 'none' to disable, set to 'custom' to provide your own command and args.
  use: none
  # Custom command to open files.
  command: "emacsclient"
  # Arguments to your custom command.
  # String contains {{.CodeFile}}, {{.TestFile}}, {{.DescriptionFile}}, {{.TestCasesFile}} will be
      replaced with corresponding file path.
  # {{.Folder}} will be substituted with the output directory.
  # {{.Files}} will be substituted with the list of all file paths.
  args: "-n -s server {{.CodeFile}}"
```

默认的模板就行，没多少需要修改的，如果需要多语言的可以改 out$_{dir}$，在浏览器面登录就行。

8.4.2 emacs

```
(defvar my/leetcode-root "~/leetcode/src/"
  "Root␣directory␣of␣leetgo-generated␣problems.")

(defun my/leetcode-format-number (n)
  "Return␣N␣formatted␣like␣1␣→␣\"0001\"."
  (format "%04d" n))
```

```
(defun my/leetcode--find-problem-dir (n)
  "Return␣directory␣path␣for␣problem␣number␣N."
  (let* ((prefix (my/leetcode-format-number n))
         (dirs (directory-files my/leetcode-root t
                                (concat "^" prefix "\\."))))
    (car dirs)))

(defun my/leetcode--pandoc-md-to-org (md-file org-file)
  "Convert␣MD-FILE␣→␣ORG-FILE␣using␣pandoc."
  (call-process "pandoc" nil nil nil md-file "-o" org-file "--wrap=none"))

(defun my/leetcode--download-image (url dest)
  "Download␣image␣URL␣to␣DEST␣file␣path."
  (url-copy-file url dest t))

(defun my/leetcode--process-org-images (org-file problem-dir)
  "Download␣remote␣images␣using␣wget␣and␣replace␣links␣with␣local␣filenames."
  (with-temp-buffer
    (insert-file-contents org-file)
    (goto-char (point-min))

    ;; 匹配所有 http 图片链接
    (while (re-search-forward "\\[\\[\\(https?://[^]]+\\.\\(png\\|jpg\\|jpeg\\|gif\\)\\)\\]\\]" nil t)
      (let* ((url (match-string 1))
             (filename (file-name-nondirectory url)) ;; 保留原始图片名
             (local-path (expand-file-name filename problem-dir)))

        ;; 如果图片不存在 -> 用 wget 下载
        (unless (file-exists-p local-path)
          (message "Downloading␣image␣via␣wget:␣%s" url)
          (call-process "wget" nil nil nil "-q" "-O" local-path url))

        ;; 替换 org 链接为相对路径
        (replace-match (format "[[./%s]]" filename) t t)))

    (write-region (point-min) (point-max) org-file nil 'quiet)))

(defun my/leetcode-open (n)
  "Open␣Leetcode␣problem␣N.␣If␣not␣found,␣auto-fetch␣using␣`leetgo␣pick␣-l␣cpp␣N`."
  (interactive "nProblem␣number:␣")
  (let* ((dir (my/leetcode--find-problem-dir n)))

    ;; If not found → auto call: leetgo pick -l cpp n
    (unless dir
      (message "Problem␣%d␣not␣found.␣Fetching␣via␣`leetgo␣pick␣-l␣cpp␣%d`..." n n)
      (let ((default-directory (expand-file-name "../" my/leetcode-root)))
        (call-process "leetgo" nil "*leetgo-pick*" t
                      "pick" "-l" "cpp" (number-to-string n)))
      ;; Re-scan after fetch
      (setq dir (my/leetcode--find-problem-dir n))
      (unless dir
        (error "After␣running␣`leetgo␣pick`,␣problem␣%d␣still␣not␣found.␣Check␣leetgo␣login." n)))

    ;; ---- If directory exists, continue normal code ----
    (let* ((md (expand-file-name "question.md" dir))
           (org (expand-file-name "question.org" dir))
           (cpp (expand-file-name "solution.cpp" dir)))

      ;; 1. Convert MD → ORG
      (my/leetcode--pandoc-md-to-org md org)

      ;; 2. Replace remote images → local images
      (my/leetcode--process-org-images org dir)

      ;; 3. Add include block to org
      (with-temp-buffer
        (insert-file-contents org)
        (goto-char (point-max))
        (insert "\n#+INCLUDE:␣\"./solution.cpp\"␣src␣cpp\n\n\\newpage")
        (write-region (point-min) (point-max) org nil 'quiet))

      ;; 4. Add compile-command to solution.cpp
      (with-temp-buffer
        (insert-file-contents cpp)
        (goto-char (point-min))
        (unless (looking-at "//␣-*-␣compile-command:")
          (insert "//␣-*-␣compile-command:␣\"make␣-f␣../Makefile␣submit\"␣-*-\n"))
        (write-region (point-min) (point-max) cpp nil 'quiet))

      ;; 5. Open org and cpp buffers
      (find-file org)
      (save-window-excursion
        (find-file cpp))
      (message "Loaded␣LeetCode␣%d␣from␣%s" n dir))))

(defun my/leetcode-generate-includes ()
  "Generate␣#+INCLUDE␣lines␣for␣all␣LeetCode␣question␣org␣files."
  (let* ((root (expand-file-name "src" default-directory))
         (dirs (directory-files root t "^[0-9]+\\..+"))
         results)
    (dolist (d dirs (nreverse results))
      (let ((q (expand-file-name "question.org" d)))
        (when (file-exists-p q)
          (push (format "#+INCLUDE:␣\"%s\"␣:minlevel␣1"
                        (file-relative-name q default-directory))
                results))))))
```

```
(defun my/leetcode-fetch-one-silent (n)
  "Fetch␣problem␣N␣silently␣(no␣opening␣buffers),␣generating␣question.org␣and␣downloading␣images."
  (let* ((dir (my/leetcode--find-problem-dir n)))

    ;; pick if missing
    (unless dir
      (message "Fetching␣%d␣via␣leetgo␣pick␣..." n)
      (let ((default-directory (expand-file-name "../" my/leetcode-root)))
        (call-process "leetgo" nil nil nil "pick" "-l" "cpp" (number-to-string n)))
      (setq dir (my/leetcode--find-problem-dir n)))

    (unless dir
      (message "Failed␣to␣fetch␣problem␣%d" n)
      (cl-return-from my/leetcode-fetch-one-silent nil))

    (let* ((md (expand-file-name "question.md" dir))
           (org (expand-file-name "question.org" dir))
           (cpp (expand-file-name "solution.cpp" dir)))

      ;; convert
      (my/leetcode--pandoc-md-to-org md org)
      (my/leetcode--process-org-images org dir)

      ;; add include to org
      (with-temp-buffer
        (insert-file-contents org)
        (goto-char (point-max))
        (insert "\n****␣Solution\n#+INCLUDE:␣\"./solution.cpp\"␣src␣cpp\n")
        (write-region (point-min) (point-max) org nil 'quiet))

      ;; add compile-command
      (with-temp-buffer
        (insert-file-contents cpp)
        (goto-char (point-min))
        (unless (looking-at "//␣-*-␣compile-command:")
          (insert "//␣-*-␣compile-command:␣\"make␣-f␣../Makefile␣submit\"␣-*-\n"))
        (write-region (point-min) (point-max) cpp nil 'quiet)))

    (message "Fetched␣%d␣OK" n)))

(defun my/leetcode-fetch-batch (numbers)
  "Fetch␣multiple␣problems␣silently.
NUMBERS␣is␣a␣string␣like␣\"1␣2␣3␣11␣17␣19\"."
  (interactive "sProblem␣numbers␣(e.g.␣\"1␣2␣3␣11\"):␣")
  (let ((nums (mapcar #'string-to-number (split-string numbers))))
    (dolist (n nums)
      (ignore-errors
        (my/leetcode-fetch-one-silent n))))
  (message "Batch␣fetch␣done."))
```

```
(defun my/leetcode-update-includes ()
  "Find␣marker␣'#␣begin␣of␣leetcode␣include',␣erase␣old␣includes,␣insert␣new␣ones."
  (interactive)
  (save-excursion
    (goto-char (point-min))
    ;; 找到标记行
    (if (search-forward "#␣begin␣of␣leetcode␣include" nil t)
        (progn
          ;; 到下一行（开始删除旧的内容）
          (forward-line 1)

          ;; 记录起点
          (let ((start (point)))

            ;; 删除所有旧的 #+INCLUDE 行
            (while (looking-at "^#\\+INCLUDE:")
              (forward-line 1))
            (delete-region start (point))

            ;; 生成新的 include 内容
            (let* ((root (expand-file-name "src" default-directory))
                   (dirs (directory-files root t "^[0-9]+\\..+")))
              (dolist (d dirs)
                (let ((q (expand-file-name "question.org" d)))
                  (when (file-exists-p q)
                    (insert
                     (format "#+INCLUDE:␣\"%s\"␣:minlevel␣2\n"
                             (file-relative-name q default-directory)))))))))
      (message "Marker␣'#␣begin␣of␣leetcode␣include'␣not␣found!"))))
```

8.4.3 本地刷题测试

由于本人的网络不是 24 小时都有，所以有一个能本地刷题的方式就好了，用测试例测好，明天再提交到网页也不失也一种方法。

```
  BEGIN { testcase = 0; input_lines = ""; expected = ""; fail = 0 }

/^input:/ {
    testcase++;
    input_lines = "";
    expected = "";
    next
}

/^output:/ {
    getline expected;

    tmp = "/tmp/input_" testcase ".txt";
    print input_lines > tmp;
    close(tmp);

    cmd = prog "␣<␣" tmp;
    result = "";
    while ((cmd | getline line) > 0) {
        if (result != "") result = result "\n";
        result = result line;
    }
    close(cmd);

    sub(/^output:[ \t]*/, "", result);
    sub(/[ \t\r\n]+$/, "", expected);

    if (result == expected) {
        print "Testcase␣" testcase "␣PASSED";
    } else {
        print "Testcase␣" testcase "␣FAILED";
        print "␣␣Input:␣" input_lines;
        print "␣␣Expected:␣" expected;
        print "␣␣Got:␣" result;
        fail = 1  # 标记失败
    }
    next
}

{
    if (input_lines == "")
        input_lines = $0;
    else
        input_lines = input_lines "\n" $0;
}

END {
    if (fail) exit 1
}
```

Listing 8.1: 利用本地测试 testcase.txt，先过本地才能上传网页

```
# Current directory (e.g. 0001.two-sum)
CURDIR_NAME := $(notdir $(CURDIR))
# Extract part after first dot, e.g. "0001.two-sum" -> "two-sum"
SPLIT_DIRNAME := $(word 2, $(subst ., ,$(CURDIR_NAME)))

a.out: solution.cpp
    $(CXX) solution.cpp -I.. -O2 -std=c++17 -Wall -o a.out

submit: testc
    leetgo submit $(SPLIT_DIRNAME) -l cpp
```

```
testc: a.out
	@awk -v prog="./a.out" -f ../../tester.awk testcases.txt

# Default target
testr:
	cargo build --bin $(SPLIT_DIRNAME)
	awk -v prog="./../../target/debug/$(SPLIT_DIRNAME)" -f ../../tester.awk testcases.txt
```

Listing 8.2: 用 makefile 测试，makefile 知道是从那一个 subdir 传了的 make 指令

[-] Leetcode

|–[-] src

| |–[+] 0001.two-sum

| |–[+] 0002.add-two-numbers

| |–[+] 0004.median-of-two-sorted-arrays

| |–[+] 0005.longest-palindromic-substring

| |–[+] 0006.zigzag-conversion

| |–[+] bits

| |—– LC$_{IO.h}$

| '—– Makefile

|–[+] target

|—– Cargo.lock

|—– Cargo.toml

|—– README.org

|—– hh.elc

|—– leetgo.yaml

'—– tester.awk

第九章 版本更新历史

2025-11-21 更新：晴

① **C 语言**: 完成了对 c 语言的复习。

② **域名**: 为 columndeeply/hosts 提交了 17868 行新域名，这个 hosts 是我最近在网上看到的最大的 host 文件，达到了近 410mb，共 12.576.671 行，专门用于 Porn，今天比较闲就把之前使用 tempermonkey 写的自动根据关键词用 ublacklist 自动 block 和从 fackads/和 github u3m8 collection 提出来的 cdn 网站和以前又闲又愤青时期屏蔽的各种网站，进行比较有二万三千行的不同 host，用后缀比如 gov/org/cn 过滤了一些自己动手又删了些，发现像很多云很多热门网站像 cloud.tencent.com./taobao/nvidia/xx.gov.cn 但用了这么久好像没有任何冲突，可能是我实在是没有怎么上网，只是用用 github 就够了。看起来很多，17868 行，几天检查不完，但是也就只是它的 703 分之一，我看了下很多是 blogspot/tumblr，互联网的域名真得是故意弄成这样难管？如果说 porn 的域名长成这样：028b2c9ad2a24433ab97b8e5dbf69597.mediatailor.us-east-1.amazonaws.com(这还真是 porn)，这能管？设立 whitelist 比 blacklist 更容易，但网络的架构 (tcp/ip-dns-domain) 就不允许这种事，我也好久没有更新我的 host 了，用 ai 比用搜索引索好用多了，ai 的用处就是过滤黄色网站，当然 openai 要开放成人内容这事说明：其实也没什么一定得搜的事。用 emacs 看文档，把记忆留在自己的脑子里，比起什么网页要重要的多。

2025-11-20 更新：晴

① **leetcode**: 完成对 leetgo 工具的 elisp 改造。chatgpt 很给力。明天一定刷 leecode。

② **real-mono-themes**: 今日 maintiner 还没有动，看来工作很忙啊。

③ **博客/pdf**: 加入了新的 org snippest，更深入学习了 org 转入 latex 更多方法，更完美的使用了本 latex 模板。

2025-11-19 更新：晴

① **天气好，玩了一下午**: 结果踩在沥水槽腿掉下去了，还好只是擦伤。

② **leetcode**: 让 chatgpt 写 leetgo 的 elisp 工具，写到最后凌晨没网了！

③ **后悔的一天**: reddit 里的 china$\backslash_{\text{irl 我常常去看}}$，实在是没意思，但总有一种引力让我去刷，莫名的又想看片，于是乎又成了看片加键政，什么高市什么献忠，吃瓜吃到 10 年的兽兽门，再看现在的其他事件，感觉 10 年代清新多了，那时候的主角还是 87 年“小妹”，现在则是 0 几年的“小妹”，真是世风日下，历史重是重复又重复，技术重是加强又如强啊，献忠机器人，何时到来？假韩炳哲的话改下：“互联网让道德世风内涵成了空气，一切除了炫/爽/性没有他用”。

④ **折腾 nixos/emacs**: 精简所有文件到博客这一个仓库。

nixos 只用一个 flake.nix 和 secrets.yaml 再把一串 key 放到一个位置就能用了，简直是极简完了。emacs 只用一个 post-init.init，再 git clone 下 minimal-emacs 到位，org 再 src 运行一下，就能用了。这两都是 2000 行的配置。总之我有了一个 blog/pdf/emacs/nixos/snippest/hosts 集于一体的仓库。完全可重现，从配置到经历，其它的代码也可以从其它 repo 里 clone 下来，但总之目前这些就是我大学大概折腾的所有玩具了吧。越说越感自己的脑残了，明明有女生喜欢我，我只有折腾这些 sb 玩意去了！现在想起了，后悔得我都成了反科技主义了。

- **越说越后悔**: 总之，刷题刷项目，早点找工作，早点重新联系她。别再做 reddit 上的支人了···别再看什么瓜片了···反技术的后果就是“深山老林里的怪人”!!! 第一，你还有家人要养，第二，你也没生存能力，第三，我还是渴望家庭，第四，你都花了二十多年在技术上，现在去深山，那你学的拼音/加法/英文/编程都是屁吗？

2025-11-18 更新：阴

① **沉迷看书**[1] 错过了我设定的上网时间二点到五点[2]…明天再测试吧[3]

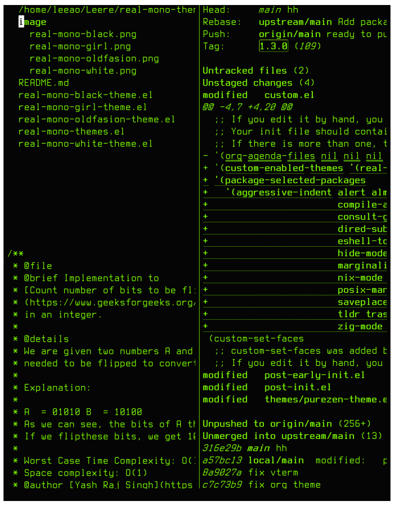
② **C 语言**: 刷一遍 flag 代码，发现自己还是不太懂 C 语言。今天又没刷力扣，感觉自己很废，就这样还想搞什么?

2025/11/17 更新： 雨

① **开始刷 Leetcode**: 完成 0 题，折腾 leetgo 去了，使用 async-shell-command 与 emacs 不全好用，两者的时间不一致问题，shell-command 又卡 emacs 本身，两者同样不能管理弹出的 shell 输出。只能用完整的 shell 命令和在 async 里 eval emacs function 套娃看看。

② **将本 "博客" 的 html 和 pdf 上传了 github.io**: 只是不只为何几十分钟了 qingsongliao.github.io 还没有，明天再看吧。

③ **完成了 real-mono-themes 的 emac 主题包**: 从此 emacs 又多了个别具一格的主题，不过也许 70 年代的 emacs 就长这样吧，oldfasion never die。



```
  /home/leeao/Leere/real-mono-ther | Head:     main hh
  image                            | Rebase:   upstream/main Add packa
    real-mono-black.png            | Push:     origin/main ready to pu
    real-mono-girl.png             | Tag:      1.3.0 (109)
    real-mono-oldfasion.png        |
    real-mono-white.png            | Untracked files (2)
  README.md                        | Unstaged changes (4)
  real-mono-black-theme.el         | modified   custom.el
  real-mono-girl-theme.el          | @@ -4,7 +4,20 @@
  real-mono-oldfasion-theme.el     |   ;; If you edit it by hand, you
  real-mono-themes.el              |   ;; Your init file should contai
  real-mono-white-theme.el         |   ;; If there is more than one, t
                                   | - '(org-agenda-files nil nil nil
                                   | + '(custom-enabled-themes '(real-
                                   | + '(package-selected-packages
                                   | +   '(aggressive-indent alert alm
                                   | +                        compile-a
                                   | +                        consult-c
                                   | +                        dired-sub
                                   | +                        eshell-tc
                                   | +                        hide-mode
/**                                | +                        marginali
 * @file                           | +                        nix-mode
 * @brief Implementation to        | +                        posix-man
 * [Count number of bits to be fl: | +                        saveplace
 * (https://www.geeksforgeeks.org/ | +                        tldr tras
 * in an integer.                  | +                        zig-mode
 *                                 |  (custom-set-faces
 * @details                        |   ;; custom-set-faces was added t
 * We are given two numbers A and  |   ;; If you edit it by hand, you
 * needed to be flipped to convert | modified   post-early-init.el
 *                                 | modified   post-init.el
 * Explanation:                    | modified   themes/purezen-theme.e
 *                                 |
 * A  = 01010 B  = 10100           | Unpushed to origin/main (256+)
 * As we can see, the bits of A th | Unmerged into upstream/main (13)
 * If we flipthese bits, we get 1( | 316e29b main hh
 *                                 | a57bc13 local/main  modified:   p
 * Worst Case Time Complexity: O(: | 8a9027a fix vterm
 * Space complexity: O(1)          | c7c73b9 fix org theme
 * @author [Yash Raj Singh](https  |
```

图 9.1: real-mono-old

2025/11/16 更新： 阴

- **开始写作**
- **使用 elegent 模板**: 搭建自己博客? 写书还是写博客? 干脆一起写吧。

[1]把什么现代诗选和中国皇帝传都丢了吧，再看下去这辈子就毁了! 只留下两本书，互联网浅薄与雅思 7 天词汇，一是提醒我互联网对大脑的 "伤害" 另一个是提升下我的大脑，最近几年或十年特别是高考后，感觉自己的脑子雾雾的。

[2]我有点网瘾，所以通过家里的路由器限制一下。速度 200kps，限制热门社交媒体的域名，时间上只有下午二点到五点能用。每次无限制上网都有一种沉迷的感觉，看黄片刷新闻作为瘾症生意在互联网真是完美载体，总之，戒断的痛苦真是难受啊!!!

[3]晚上又用母亲的手机上网了，怎么试都发现这个 github.io 是 404，下载下来好像是 html 本身的问题。切换了一下账号发现看不了那个号了，问了下 chatgpt 发现 github 有新号防 bot 功能，没法只能重新用回我的 NestorLiao 账号https://github.com/orgs/community/discussions/55609?utm_source=chatgpt.com。

参考文献

[1] LI Q, CHEN L, ZENG Y. The Mechanism and Effectiveness of Credit Scoring of P2P Lending Platform: Evidence from Renrendai.com. China Finance Review International, 2018, 8(3): 256-274.

[2] CARLSTROM C T FUERST T S. Agency Costs, Net Worth, and Business Fluctuations: A Computable General Equilibrium Analysis. The American Economic Review, 1997: 893-910.

[3] QUADRINI V. Financial Frictions in Macroeconomic Fluctuations. FRB Richmond Economic Quarterly, 2011, 97(3): 209-254.

[4] 方军雄. 所有制、制度环境与信贷资金配置. 经济研究, 2007(12): 82-92.

[5] 刘凤良, 章潇萌, 于泽. 高投资、结构失衡与价格指数二元分化. 金融研究, 2017(02): 54-69.

[6] 吕捷 王高望. CPI 与 PPI “背离”的结构性解释. 经济研究, 2015, 50(04): 136-149.

[7] Ted Kazynski. 工业革命与人类命运. 社会学研究, 2017(02): 54-69.

附录 A 开发中遇见的各种软件问题

A.0.1 General Contributions

内容提要

- ☐ org-mode，提交 pr 增加 INCLUDE 的行末倒数行数的行为代码
- ☐ melpa，提交 pr 增加自己的 leetgo
- ☒ consult-gh，提交 pr 修改 readme 中的 consult-gh-search-code 的使用
- ☒ melpa，提交 pr 增加自己的 real-mono-themes
- ☒ nixpkg，提交 issue 增加 biospy 等 nix 生理信号包
- ☒ columndeeply/hosts，提交 pr 增加 17868 行新域名，修改脚本为 0.0.0.0
- ☒ pyim，提交 issue 修改% 报错行为

A.0.2 TODO package leego elisp package as emacs package

- description: do leetcode in emacs with leetgo
- require: wget/pandoc/leetgo/
- wget for fetch image
- pandoc for markdown to org
- leetgo for pick/commit questions
- simple init, use leetgo
- batch fetch, (leetcode-fetch-batch "1 2 3") to fetch 1/2/3 questions
- also you can fetch contest,
- auto detected browser cookie, no need to fill cookies by yourself
- offline test, use pre fetched testcase/makefile/awk to local test tasks.
- org format, local image, and generally you can easily export all questions.

A.0.3 FIXME pyim % 字符触发 Not enough arguments for format string

使用虎码码表https://github.com/welandx/huma-danzi.pyim readme 发现输入 l 时报错: Debugger entered–Lisp error: (error "Not enough arguments for format string") message("[l]: 1. 而 2.% (1/1) $ ") #f(compiled-function (string position tooltip) " 使用 minibuffer 来显示 STRING。" #<bytecode -0x26f1e7f81176ed3>)("[l]: 1. 而 2.% (1/1) $ " 620 minibuffer) apply(#f(compiled-function (string position tooltip) " 使用 minibuffer 来显示 STRING。" #<bytecode -0x26f1e7f81176ed3>) "[l]: 1. 而 2.% (1/1) $ " 620 minibuffer nil) pyim-page-show("[l]: 1. 而 2.% (1/1) $ " 620 minibuffer) pyim-page–refresh(nil) pyim-process-ui-refresh() pyim-process-run() pyim-self-insert-command() funcall-interactively(pyim-self-insert-command) call-interactively(pyim-self-insert-command) pyim-process-input-method(108) pyim-input-method(108)

删除 hmdz.pyim 中的% 或使用中文的％使用时不会报错，发现 pyim 有% 就报错: Debugger entered–Lisp error: (error "Not enough arguments for format string") message("[l]: 1.% (1/1) $ ") #f(compiled-function (string position tooltip) " 使用 minibuffer 来显示 STRING。" #<bytecode 0x1d8cdaf8ba1c112a>)("[l]: 1.% (1/1) $ " 624 minibuffer) apply(#f(compiled-function (string position tooltip) " 使用 minibuffer 来显示 STRING。" #<bytecode 0x1d8cdaf8ba1c112a>) "[l]: 1.% (1/1) $ " 624 minibuffer nil) pyim-page-show("[l]: 1.% (1/1) $ " 624 minibuffer) pyim-page–refresh(nil) pyim-process-ui-refresh() pyim-process-run() pyim-self-insert-command() funcall-interactively(pyim-self-insert-command) call-interactively(pyim-self-insert-command) pyim-process-input-method(108) pyim-input-method(108)

A.0.4 FIXME How to modify org-mode's #+INCLUDE:

to let it to exclude top and buttom? it have: top specific line to end specific line line to top specific top specific line to end contribute to the org-mode's include function to let it

exclude from top specific line to end specific line but count from buttom

- study org-mode, try to **contribute** to it.

A.0.5 DONE pull request to the porn site list project

fix the columndeeply/hosts 's 127.0.0.0 to 0.0.0.0

by clone and compare the problem is that "I include a lost non-porn site in to it". not just "non-porn" sites, I also get a lot cdn server to block, which fetch from a lot of m3u8 streaming server. it may also block some movies which using the same server as porn sites… so, there is only KLUDGE for thing like blocking porn site, first question: what is porn? really? to be, the social media like twitter which spread sexual clip? the pornhub teaching mathmatic? the reddit/4chan/zhihu/weibo/… even the taobao can sell sex toys with sexual pictures, so blocking internet is blocking porn really? I don't know, it's a question for everyone.

finished, by

```
 #!/usr/bin/env bash
set -euo pipefail

HOST_URLS=(
  "https://raw.githubusercontent.com/columndeeply/hosts/main/hosts00"
  "https://raw.githubusercontent.com/columndeeply/hosts/main/hosts01"
  "https://raw.githubusercontent.com/columndeeply/hosts/main/hosts02"
  "https://raw.githubusercontent.com/columndeeply/hosts/main/hosts03"
  "https://raw.githubusercontent.com/columndeeply/hosts/main/hosts04"
  "https://raw.githubusercontent.com/columndeeply/hosts/main/hosts05"
)

TMPDIR="$(mktemp␣-d)"
REMOTE_DOMAINS="$TMPDIR/remote_domains"
YOUR_DOMAINS="$TMPDIR/your_domains"
OUTPUT="hosts6"

echo "[*]␣Downloading␣hostlists..."
> "$REMOTE_DOMAINS"
for url in "${HOST_URLS[@]}"; do
    echo "␣␣␣␣->␣$url"
    curl -sL "$url" \
        | grep -E "^[0-9:\.]+" \
        | awk '{print␣$2}' \
        >> "$REMOTE_DOMAINS"
done

echo "[*]␣Normalizing␣remote␣domains..."
sort -u "$REMOTE_DOMAINS" > "$TMPDIR/remote_sorted"

echo "[*]␣Extracting␣your␣domains..."
```

```
grep -E "^[0-9:\.]+" /etc/hosts \
    | awk '{print␣$2}' \
    | sort -u \
    > "$YOUR_DOMAINS"

echo "[*]␣Generating␣unique␣domains␣(hosts6)..."
comm -23 "$YOUR_DOMAINS" "$TMPDIR/remote_sorted" \
    | awk '{print␣"0.0.0.0",␣$1}' \
    > "$OUTPUT"

echo "[+]␣Done.␣Unique␣domains␣saved␣in␣$OUTPUT"
echo "␣␣␣␣Total:␣$(wc␣-l␣<␣"$OUTPUT")"
```

get uniq hosts, and update to github. it's I only get 23240 unique lines.

add 17868 new hosts

Used to maintain my hostslist, write a tempermonkey script with ublacklist addon to block chinese keyword search result automatically.

mainly it blocked sexual model gallary sites/Pirate JAV/and chinese porn m3u8 CDN which I filter from a github m3u8 collection repos/or anything non-programming like socialmedia/news/shopping…, occationally the script blocks org/edu/gov sites too.

I remove duplicate hosts in yours and delete non-porn sites, but there still too many which I can't checkout they all.

BTW, I didn't update my hostslist for a long time, because I just find blocking search engine and "hot" social-media instead porn sites is way more easier, I now use tavily/chatgpt/github in emacs only.

BTW, seems like use 0.0.0.0 instaed of 127.0.0.1 is faster.

A.0.6 DONE make Leetcode fresh tasks list

- have already?
- yes/no, yes, edit that file, no, get the files(testcase/question.md/solution.xxx)
- edit the files question to include src in end, solution.xxx to have mode line in header.
- download image to local, turn md in org format, and delete md one.
- add a allorg, to include all under src dir's question.org
- 支持无网本地测试，在源代码文件头部中加入 mode line，在题目文件尾部中加入 include 源代码
- 根据题号在相同的目录下抓取不同题目和几个指定语言，并下载至本地图片，再转化 md 为 org 格式
- 支持在根目录中得到 src 的所有 org file 的 include
- md to org 的 == 问题
- based on leetgo to creat a leetcode elisp pkg
- test leetcode emacs pkg instead

A.0.7 DONE learn how to reference in org

<empty citation> org-cite-insert unable to insert problem **M-RET** to enter cite list.

A.0.8 DONE upload large files into the github repo

use sed to filter, use what to split into small files.

A.0.9 DONE try to package real-mono-theme to melpa

minic almost-mono-theme, creat recipe, make pr wait for mainter check. I am familaring with github, feeling awesome!

A.0.10 DONE creat new account's github repos for github.io

have a usrname.github.io public repo, have index.html in it, in url

A.0.11 DONE upload Purezen to github

- description: theme that really monochrome A collection of real monochrome emacs themes in a couple of variants.
- clone a monochrome and study it https://github.com/cryon/almost-mono-themes
- minic a theme to creat a repo https://github.com/qingsongliao/real-mono-themes

附录 B 嵌入式软件招聘常见要求

- 本科及以上学历，嵌入式软件工作经验，电子、通信或计算机相关专业；
- 有 ARM Linux 软件的开发经验，熟练使用 C++ 和 C 语言；
- 熟悉 Linux 常用设备操作，有 spi、can、WiFi、audio、video 相关驱动开发经验优先；
- 熟悉多线程、多进程和 socket 网络编程，有并发程序开发经验和良好的设计思路；
- 有机器人相关嵌入式设计或者实时系统经验优先。
- 负责 yocto & android 的构建（Build）与升级（OS Upgrade），确保系统的稳定性和兼容性。
- 参与新硬件平台的 Bringup，包括 CPU、GPU、Memory 等核心组件的初始化和调试。
- 分析和解决系统稳定性问题，优化系统架构设计，提升整体性能和可靠性。
- 研究客户系统功能需求，定制和优化系统功能，满足客户特定场景的需求。
- 解决系统性能问题，包括 CPU、GPU、Memory 等资源的优化与调度。
- 主导基线升级（Baseline Upgrade），确保系统与最新技术和安全标准同步。
- 具备操作系统（如 Android、Linux）的构建与升级经验，熟悉系统启动流程和内核开发。
- 有硬件平台 Bringup 经验，熟悉 CPU、GPU、Memory 等核心组件的初始化和调试。
- 具备系统架构设计能力，能够优化系统性能并解决稳定性问题。
- 有基线升级经验，熟悉版本管理和代码合并流程。
- 具备客户需求分析和功能定制能力，能够根据客户需求优化系统功能。
- 熟悉性能优化工具（如 perf, ftrace, gprof 等），能够解决 CPU、GPU、Memory 相关的性能问题。
- 有嵌入式系统开发经验，熟悉低功耗设计和优化。
- 熟悉虚拟化技术（如 KVM, QEMU）和容器化技术（如 Docker, Kubernetes）。
- 有 MTK 芯片平台开发经验

附录 C 嵌入式招聘常见笔试问题

附录 D 嵌入式招聘常见面试问题