

“一生一芯” 太理工作室见习学员第二次学习路线

古之立大事者，
不惟有超世之才，
亦必有坚韧不拔之志。

——苏轼《晁错论》

注意：本周作业重在Linux环境学习和C语言学习，因此我们没有出很难的编程题，希望大家这周也能好好学习，认真对待；否则下周作业我们出的编程题大家可能就应付不了了，以及最后的大作业如果没有这些基础，是绝对做不出来的，希望大家可以持之以恒，不中途放弃！

预防针

交作业一定要看提交作业部分的具体方式！确认提交作业的格式正确

小朋友（许鹏远）学长的碎碎念

本次作业看上去很多，实际上完成起来很快，必做基础任务除了C语言部分，其他部分都可以很快完成，只需要动动手，学完即可，更多的是大家之后一直接触从而熟络的这个过程。因此实际上我们的作业**集中在C语言部分的学习**，大家会在上一次作业学完循环之后这次学完函数和数组部分知识，这是我们的**大头任务**；希望大家可以**合理安排时间，每天都抽出两三个小时时间学习技术**，那么作业是绝对可以顺利完成的，大家的晚自习时间或是空课时间都可以拿出电脑学习技术；另外，注意劳逸结合哦！这个作业量只要合理安排时间并且不要摆，那么是不会给大家太大压力的。学完这些内容后我们的编程环境就变得高端大气上档次了（-v-），之后可以做很多有趣的事情，希望大家加油，诸君共勉！

未来大家的编程环境：

```
Zellij (rusty-musktrat) npc nemo
npc
Press ? for help
.. (up a dir)
</sysx-workbench/npc/
> build/
> csrc/
> include/
> sdb/
> utils/
device.cpp
diffest.cpp
init.cpp
main.cpp
mem.cpp
reg.cpp
tags
time.cpp
> vsrc/
delay.v
lfsr.v
top.sdc
top.v
sysx_23060271_ALU.v
sysx_23060271_arbiter.v
sysx_23060271_CLINT.v
sysx_23060271_Define.v
sysx_23060271_DRC.v
sysx_23060271_EXU.v
sysx_23060271_IDU.v
sysx_23060271_IFU.v
sysx_23060271_LSU.v
sysx_23060271_Rf.v
sysx_23060271_SRAM.v
sysx_23060271_UART.v
sysx_23060271_WBU.v
Makefile
19 .rst(rst),
2 .csr(csr),
3 .rs1(rs1),
4 .imm(imm),
5 .mpc(mpc),
6 .inst(inst),
7 .pc(pc)
8 );
9 //instruct the code
10 //always@(*)
11 //display("instruction begining");
12 sysx_23060271_IDU Idu (
13 .rst(rst),
14 .clk(clk),
15 .inst(inst),
16 .imm(imm),
17 .fun7(fun7),
18 .opcode(opcode),
19 .rs1_ad(rs1_ad),
20 .rs2_ad(rs2_ad),
21 .rd_ad(rd_ad),
22 .fun3(fun3),
23 .ALU_operator(ALU_operator),
24 .ALU_object(ALU_object),
25 .LSU_sign(LSU_sign),
26 .jalr(jalr),
27 .jalr(jalr),
28 .branch(branch),
29 .csr(csr),
30 .wen(wen)
31 );
32
33 //execute the code
34 //always@(*)
35 //display("executon begining");
36 sysx_23060271_EXU Exe (
37 .in_valid(LSU_valid),
38 .clk(clk),
39 .opcode(opcode),
40 .EXU_rd(EXU_rd),
41 .rs1(rs1),
42 .rs2(rs2),
43 .ALU_operator(ALU_operator),
44 .ALU_object(ALU_object),
45
test
[ if-else] PASS
[ bit] PASS
[ mul-longlong] PASS
[ recursion] PASS
[ string] PASS
[ min3] PASS
[ pascal] PASS
[ select-sort] PASS
[ add-longlong] PASS
[ prime] PASS
[ shuixianhua] PASS
[ mov-c] PASS
[ unalign] PASS
[ div] PASS
[ switch] PASS
[ max] PASS
[ hello-str] PASS
[ load-store] PASS
[ crc32] PASS
[ fact] PASS
[ mersenne] PASS
[ leap-year] PASS
[ add] PASS
[ shift] PASS
[ movsx] PASS
[ sub-longlong] PASS
[xpy69 ~/.ssh/sysx-workbench/am-kernels/tests/cpu-tests] >

debug
7 0x80000278: ff 41 01 13 addi sp, sp, -12
1 0x8000027c: 00 00 05 17 auipc a0, 0
2 0x80000280: 0d 05 05 13 addi a0, a0, 208
3 0x80000284: 00 11 24 23 sw ra, 8(sp)
4 0x80000288: f6 5f f0 ef jal ra, 0x800001ec
5 0x800001ec: ff 41 01 13 addi sp, sp, -12
6 0x800001f0: 00 00 45 37 lui a0, 4
7 0x800001f4: 00 11 24 23 sw ra, 8(sp)
8 0x800001f8: 00 81 22 23 sw s0, 4(sp)
9 0x800001fc: 00 00 05 93 addi a1, zero, 0
10 0x80000200: 00 00 07 97 auipc a5, 0
11 0x80000204: 15 c7 a7 83 lw a5, 348(a5)
12 0x80000208: 82 35 05 13 addi a0, a0, -2013
13 0x8000020c: 00 07 80 e7 jalr ra, 0(a5)
14 0x80000010: 00 00 07 17 auipc a4, 0
15 0x80000014: 35 c7 07 13 addi a4, a4, 860

NORMAL npc-log.txt unix | utf-8 | text 0% 7:40

Ctrl + <LOCK> <PANE> <TAB> <RESIZE> <MOVE> <SEARCH> <SESSION> <QUIT>
Tip: Alt + <n> => open new pane. Alt + <+/-> or Alt + <h/jk/l> => navigate between panes. Alt + <+/-> => increase/decrease pane size.
```

硅基生物（李振杰）学长的碎碎念

学习呢，是一个持久的过程。提问是这个过程很重要的一个环节，学会科学的提问是很必要的。与此同时也要有一定的自己解决问题的能力，也需要一定的翻译能力。上周的学习大家也体会到了英语的重要性，很多信息都是英文的，总是得走出舒适圈去试着翻译的。

其次就是代码风格，好的代码可读性很高，看着很舒服，是能做到不言自明的。可以自行STFW然后确定自己的代码风格，该缩进的地方缩进，该对齐的地方对齐，变量名也是有意义的，而不是abcd...。那么说了这么多，好的代码风格是怎样的：

```
vim
15
16 #include "macro.h"
17 #include <memory/host.h>
18 #include <memory/paddr.h>
19 #include <device/mmio.h>
20 #include <isa.h>
21
22 #if defined(CONFIG_PMEM_MALLOCC)
23 static uint8_t *pmem = NULL;
24 #else // CONFIG_PMEM_GARRAY
25 static uint8_t pmem[CONFIG_MSIZ] PG_ALIGN = {};
26 #endif
27
28 uint8_t* guest_to_host(paddr_t paddr) { return pmem + paddr - CONFIG_MBASE; }
29 paddr_t host_to_guest(uint8_t *haddr) { return haddr - pmem + CONFIG_MBASE; }
30
31 static word_t pmem_read(paddr_t addr, int len) {
32     word_t ret = host_read(addr: guest_to_host(paddr: addr), len);
33     return ret;
34 }
35
36 static void pmem_write(paddr_t addr, int len, word_t data) {
37     host_write(addr: guest_to_host(paddr: addr), len, data);
38 }
39
40 static void out_of_bound(paddr_t addr) {
41     panic("address = " FMT_PADDR " is out of bound of pmem [" FMT_PADDR ", " FMT_PADDR "] at pc = " FMT_WORD,
42         addr, PMEM_LEFT, PMEM_RIGHT, cpu.pc);
43 }
44
45 void init_mem() {
46     #if defined(CONFIG_PMEM_MALLOCC)
47     pmem = malloc(CONFIG_MSIZ);
48     assert(pmem);
49     #endif
50     #ifdef CONFIG_MEM_RANDOM, memset(s: pmem, c: rand(), n: CONFIG_MSIZ);
51     Log("physical memory area [" FMT_PADDR ", " FMT_PADDR "], PMEM_LEFT, PMEM_RIGHT);
52 }
53
NORMAL pa2 nemo > c paddr.c
17 46% 35:1 11:38
```

希望大家之后coding的时候能写出健全美观的代码。

旅鼠（章廷宽）学长的碎碎念

相信大家真的接触到Linux时，会先被它完全不同于Windows等日常直接使用的操作系统的黑窗口吓到——所有的操作都要通过敲入一个个字母组合来执行。诚然，这么多的指令，是都需要我们熟练使用，就跟C语言的那些函数、将来马上要学的Git一样。有人可能要问，是不是应该一个个抄写、背下来。但你要知道的是，计算机世界中，我们做这些实践的成本是很低的，只要动动手指就行。也就是，你大可以**多多实践、多多试错**——做好备份/快照就行。所以也完全没有必要纠结在某一些东西记不住——当你用到时，再次查阅、学习。不断的实践、熟悉中，这一切也就变得游刃有余了。

必做基础学习内容及任务（按顺序完成）

0. 字符编码与乱码原因（学习即可）

根据以下链接学习字符编码与乱码原因相关内容，**之后提交的源码都要使用UTF-8编码（Linux默认就是，Devcpp默认是GB2312/GBK编码）**。如果使用不匹配的编码打开文件会乱码（比如说使用UTF-8编码方式打开GBK编码的文件）。

<https://www.paicoding.com/article/detail/305>

https://www.hello-algo.com/chapter_data_structure/character_encoding/

<https://blog.csdn.net/xuan196/article/details/115127416>

1. Markdown（码字神器）

Markdown 是一种轻量级的标记语言，可用于在纯文本文档中添加格式化元素。Markdown 由 John Gruber 于 2004 年创建，如今已成为世界上最受欢迎的标记语言之一。它有以下几个优点：

1. 专注于文字内容；
2. 纯文本，易读易写，可以方便地纳入版本控制；
3. 语法简单，没有什么学习成本，能轻松在码字的同时做出美观大方的排版。

有了Markdown，就可以轻松记录笔记。你可以自己STFW寻找Markdown教程，这里我们给几个推荐的教程：

- <https://markdown.com.cn/>
- （交互式）[Markdown Tutorial](#)

Markdown需要渲染，**这里我们推荐使用Typora（群文件有破解版）**或者VSCode下的Markdown Preview Enhance插件或者Office Viewer插件（这个类似Typora，所见即所得）。之后的笔记就用Markdown写。

效果展示：

elf二进制执行文件.md - Typora

文件(F) 编辑(E) 段落(P) 格式(O) 视图(V) 主题(T) 帮助(H)

文件大纲

elf二进制执行文件.md

elf二进制执行文件 [TOC] 官方文档的链接: Tool Interface

make语法.md

make语法 [TOC] 核心 target ... prerequisites依赖 ... recipe方法

vim学习及配置.md

vim学习及配置 相关模式 正常模式: 在文件中四处移动光标

ysyx学习杂项.md

ysyx学习杂项 readline函数库简单使用 (达到仿照终端控制行

elf二进制执行文件

elf二进制执行文件

0.相关使用命令

1.文件头Ehdr

2.文件节Shdr

3.文件段Phdr

4.解析其中的函数——符号表Sym和str

5.elf形成与c, 编译器之间的联系

官方文档的链接: [Tool Interface Standard \(TIS\) Executable and Linking Format \(ELF\) Specification Version 1.2](#)

ELF 源码文件位置: `/usr/include/elf.h`

0.相关使用命令

readelf——riscv64-linux-gnu-readelf -a+文件名使用, ysyx中推荐使用

objdump——objdump, linux的反汇编elf文件命令

hexdump——hd+文件名使用, 以十六进制输出elf内容同时对比相应的内容, 推荐与readelf对照使用

1.文件头Ehdr

这里的数据结构非常简单, 但初学理解容易混乱, 文件头是由一个E1f64_Ehdr的数据结构组成的, 简单说明该结构体中的数据

e_ident作为开头主要用装一些标示信息, 固定值来判断ELF文件的合法性, 其开头4个字节固定为0x7f, ELF三个字符。

e_type是用来标志文件类型的比如有1.可重定位文件, 2.可执行文件, 3.共享目标文件。

sh_type	描述
SHT_NULL	无对应节区, 该节其他字段取值无意义

< </>12365 字符

注意：接下来的所有作业都会在Linux（你的虚拟机）上进行

2.APT包管理器（学习了解即可）

APT是Debian及其衍生（如Ubuntu）的Linux软件包管理器。APT可以自动下载，配置，安装二进制或者源代码格式的软件包，因此简化了Linux系统上管理软件的过程。

APT需要root权限，如果提示Permission denied，看看自己是不是没有给权限（sudo）。

常用的一些指令

- 更新系统中的软件包列表：`apt update`
- 升级软件包：`apt upgrade`
- 安装指定的软件命令：`apt install <package_name>`
- 安装多个软件包：`apt install <package_1> <package_2> <package_3>`
- 更新指定的软件命令：`apt update <package_name>`
- 显示软件包具体信息,例如：版本号，安装大小，依赖关系等等：`apt show <package_name>`
- 删除软件包命令：`apt remove <package_name>`
- 清理不再使用的依赖和库文件：`apt autoremove`
- 移除软件包及配置文件：`apt purge <package_name>`
- 查找软件包命令：`apt search <keyword>`

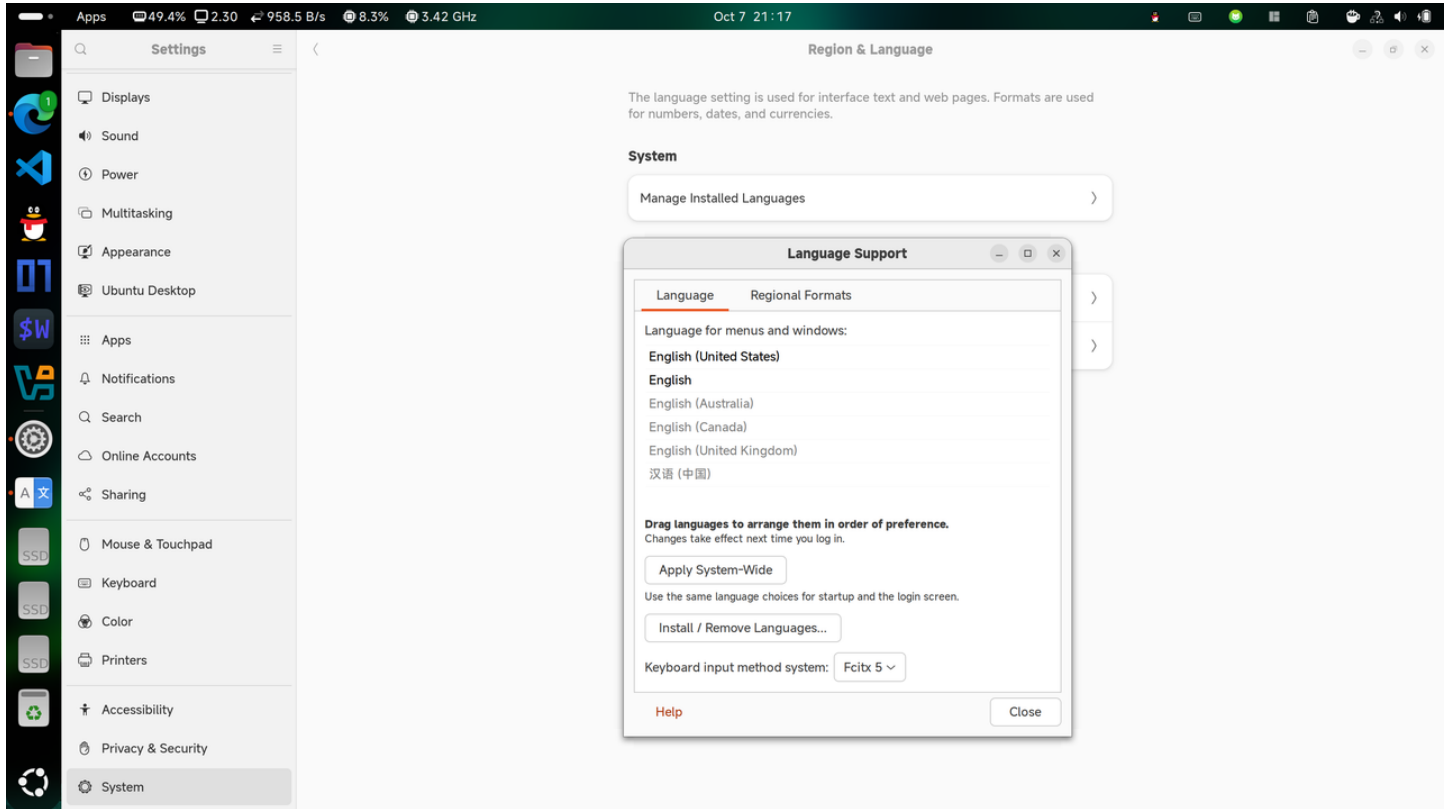
3.中文输入法（使用fcitx5后端）（安装使用即可）

众所周知，在计算机的世界里要输入中文，需要中文输入法，我们来引导大家安装中文输入法

- Step 1: 检查系统中文环境

在系统设置（Setting）——区域与语言（Region & Language）——管理安装语言（Manage Installed Languages）——Install/Remove Languages里找到Chinese（simplified）并选中，最后Apply System Wide

- 安装fcitx5



- 1 `$ sudo apt install fcitx5`
- 2 `$ sudo apt install fcitx5-chinese-addons`
- 3 `$ sudo apt install fcitx5-frontend-gtk3 fcitx5-frontend-gtk2`
- 4 `$ sudo apt install fcitx5-frontend-qt5 kde-config-fcitx5`

- 卸载ibus（ibus与fcitx5之间可能有点冲突）

- 1 `$ sudo apt remove ibus`

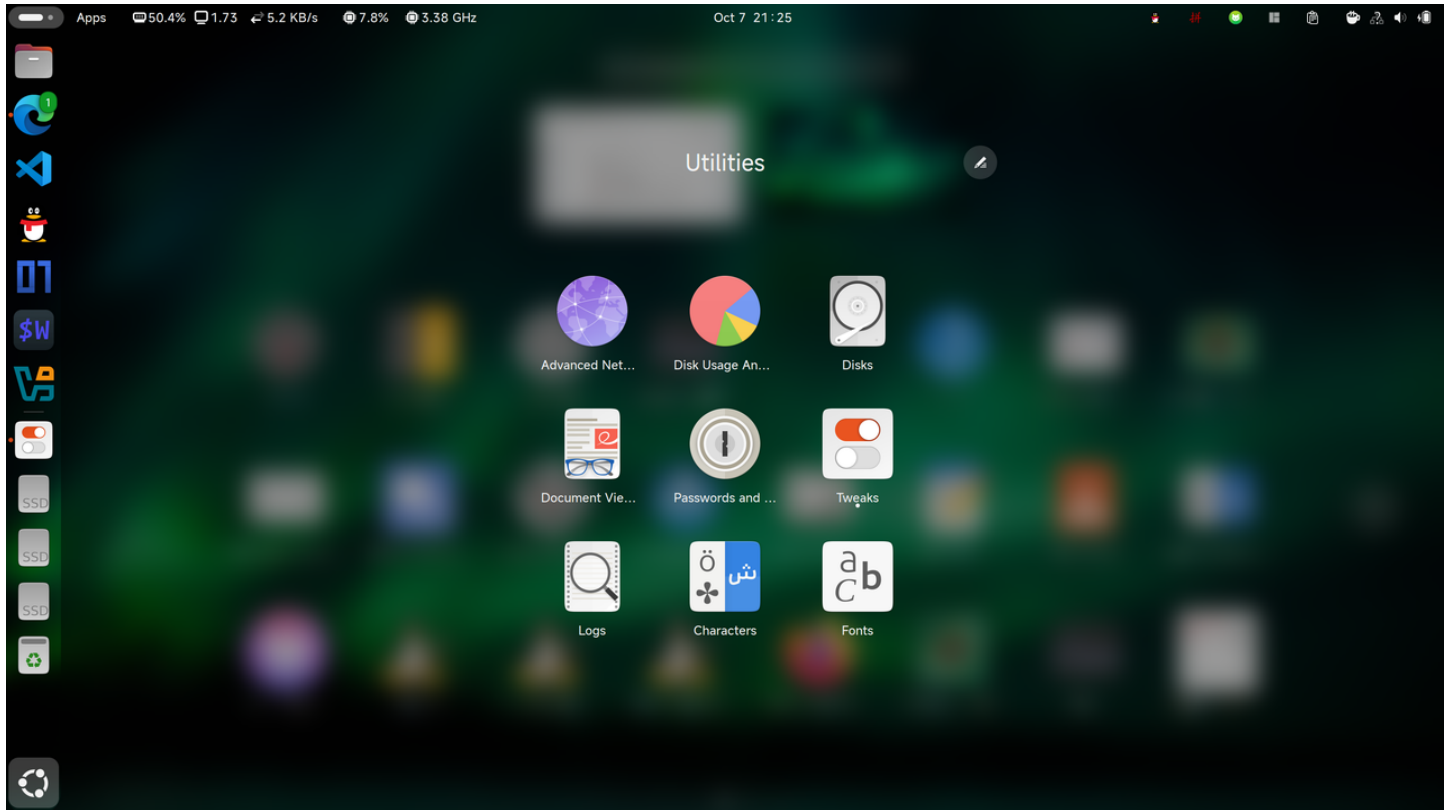
- 配置默认输入法，在终端中输入然后选择fcitx5

- 1 `$ im-config`

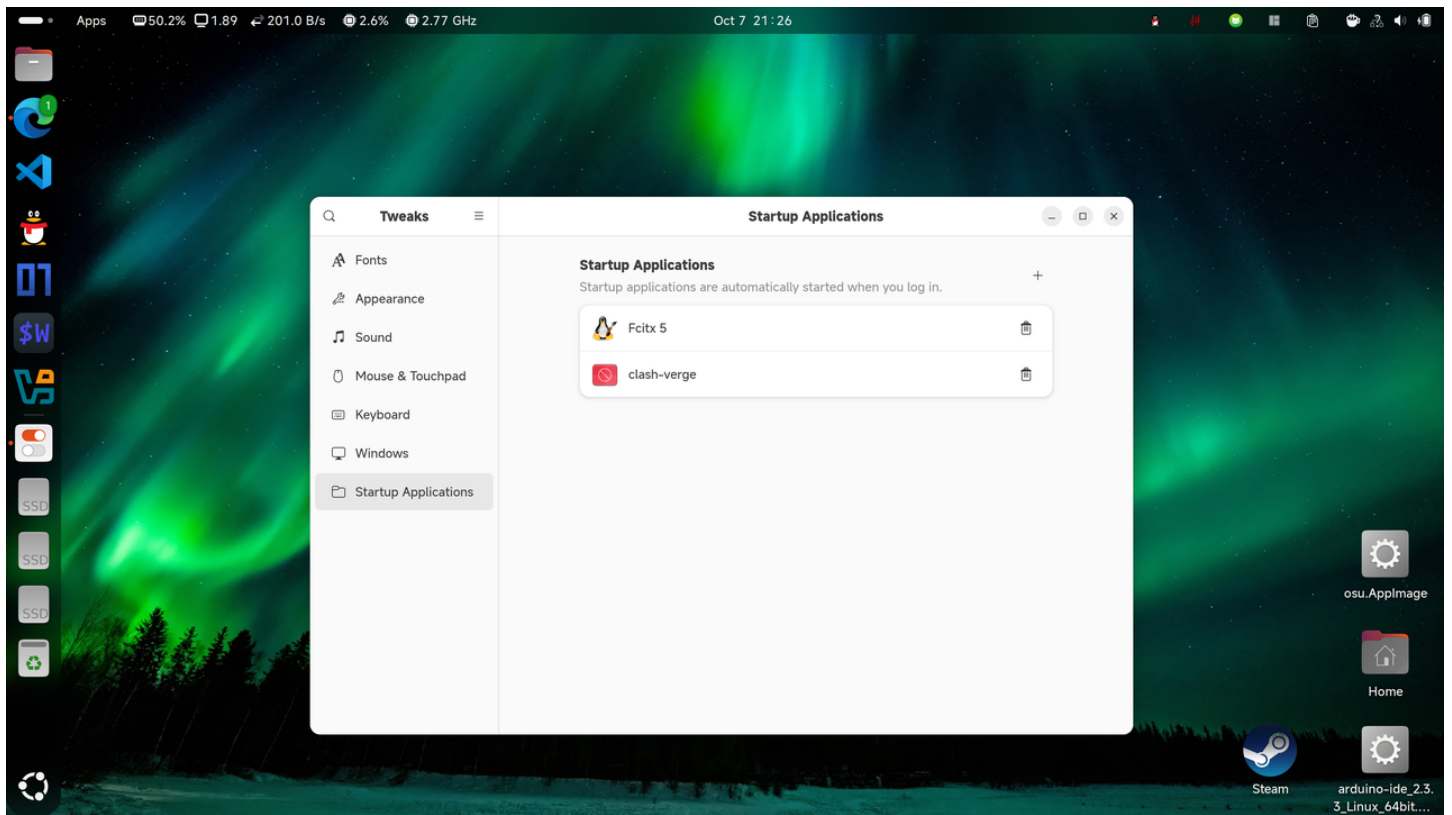
- 使用gnome-tweaks来设置fcitx5为开机自启动

首先安装gnome-tweaks

```
1 $ sudo apt install gnome-tweaks
```

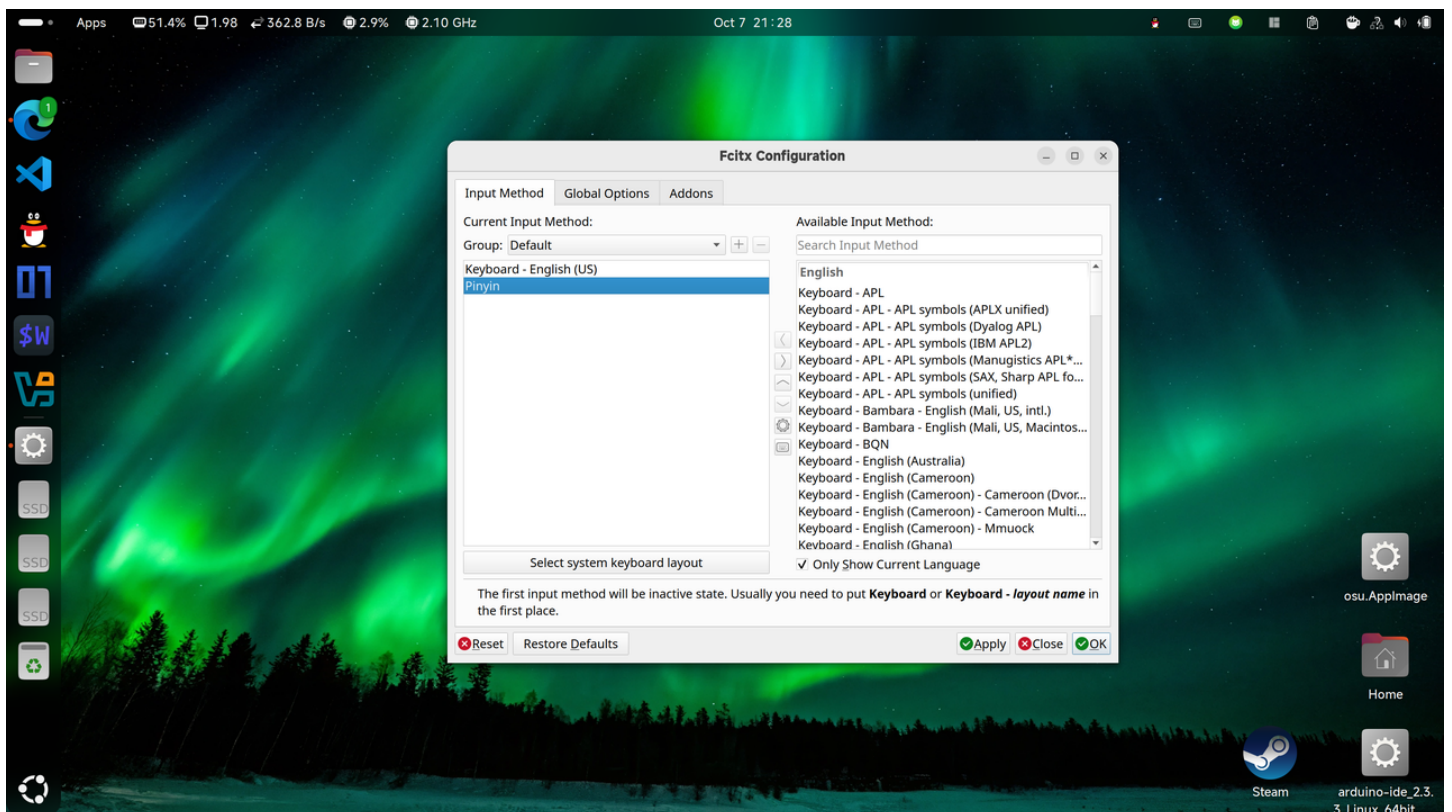


从应用菜单中找到tweaks（在Utilities文件夹里）并打开，将fcitx5设置为开机自启动



- 设置拼音输入法

在应用菜单中找到Fcitx5 Configuration，从右侧菜单中找到“Pinyin”添加到左侧，然后Apply，就大功告成啦！



PS: fcitx5的其他配置请自行STFW；安装也可以参照网上的其他教程。

PSS: fcitx5中英文切换默认是Ctrl+Space

4.Vim的安装与使用（真正的编程环境）

Vim被称为**编辑器之神**。对于大多数人来说，Vim有着一个比较陡峭的学习曲线。这意味着开始学习的时候可能会进展缓慢，但是一旦掌握一些基本操作之后，能大幅度提高编辑效率。

（PS：用习惯Vim了之后不管什么软件都想上Vim键位，我的VSCode已经换Vim模式了——LZJ，不用鼠标只用键盘的编程模式是真舒服——XPY）

Vim的安装

```
1  $ sudo apt install vim
```

Vim的使用及配置

以下是两个必做任务：

1. 用 Vim自带的 vimtutor 进行练习（**基础要求Lesson1-Lesson3**，学有余力的可以做完，整篇做完需要20min左右），安装完Vim之后直接在**命令行里输入vimtutor**即可进入练习程序。

（PS：全英文大家要努力看懂哦！实在看不懂的话可以对照这个：

https://blog.csdn.net/qq_40395874/article/details/116047253）

2. 计算机教育中缺失的一课1/15：编辑器（Vim）

<https://missing-semester-cn.github.io/2020/editors/>

课后题只需要做2，5，6，7即可

另外，一些推荐了解性选做教程如下：

- 菜鸟教程Linux vi/vim

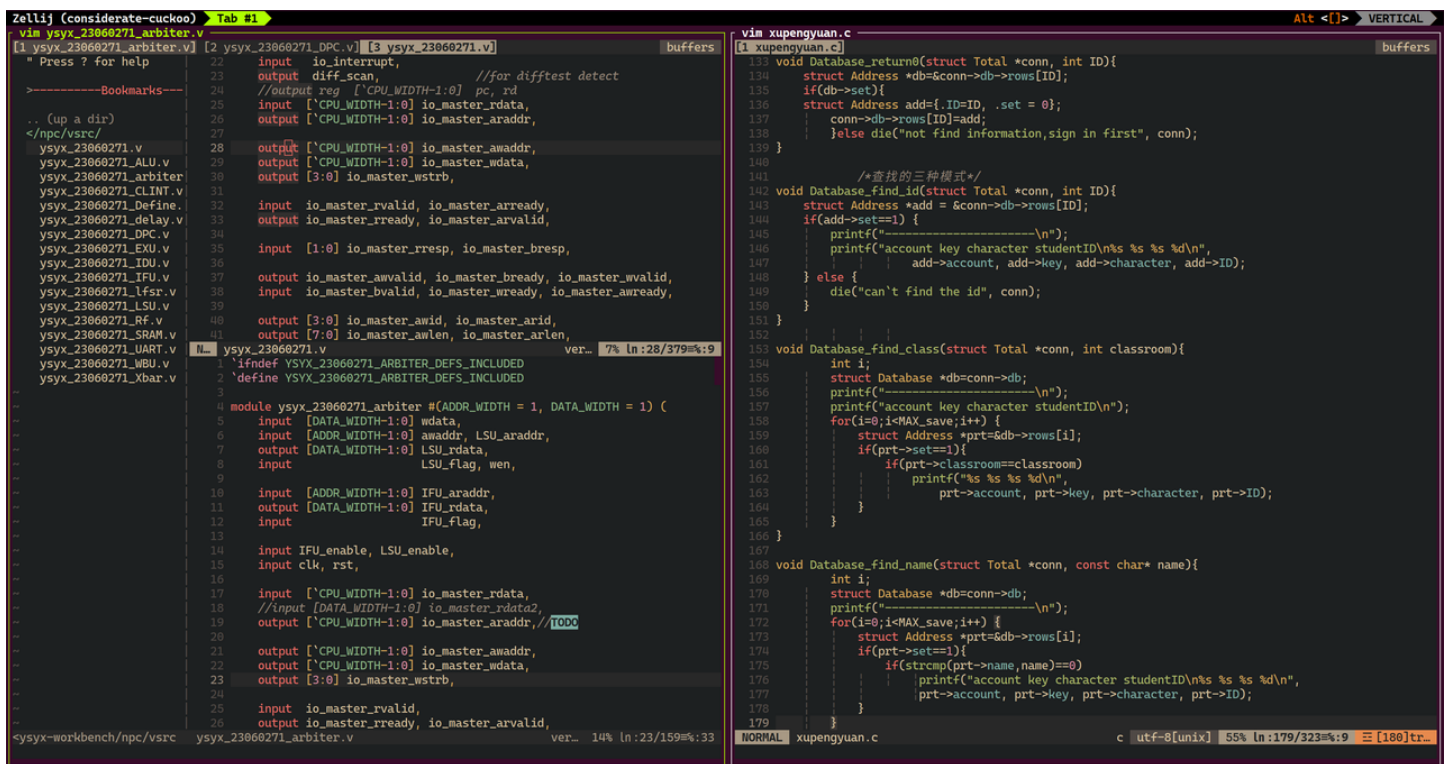
<https://www.runoob.com/linux/linux-vim.html>

- vim配置学习

<https://zhuanlan.zhihu.com/p/145793963>

从中选择你需要的安装使用即可

效果展示：（不要求一样，大家自己用的舒服就可以了）

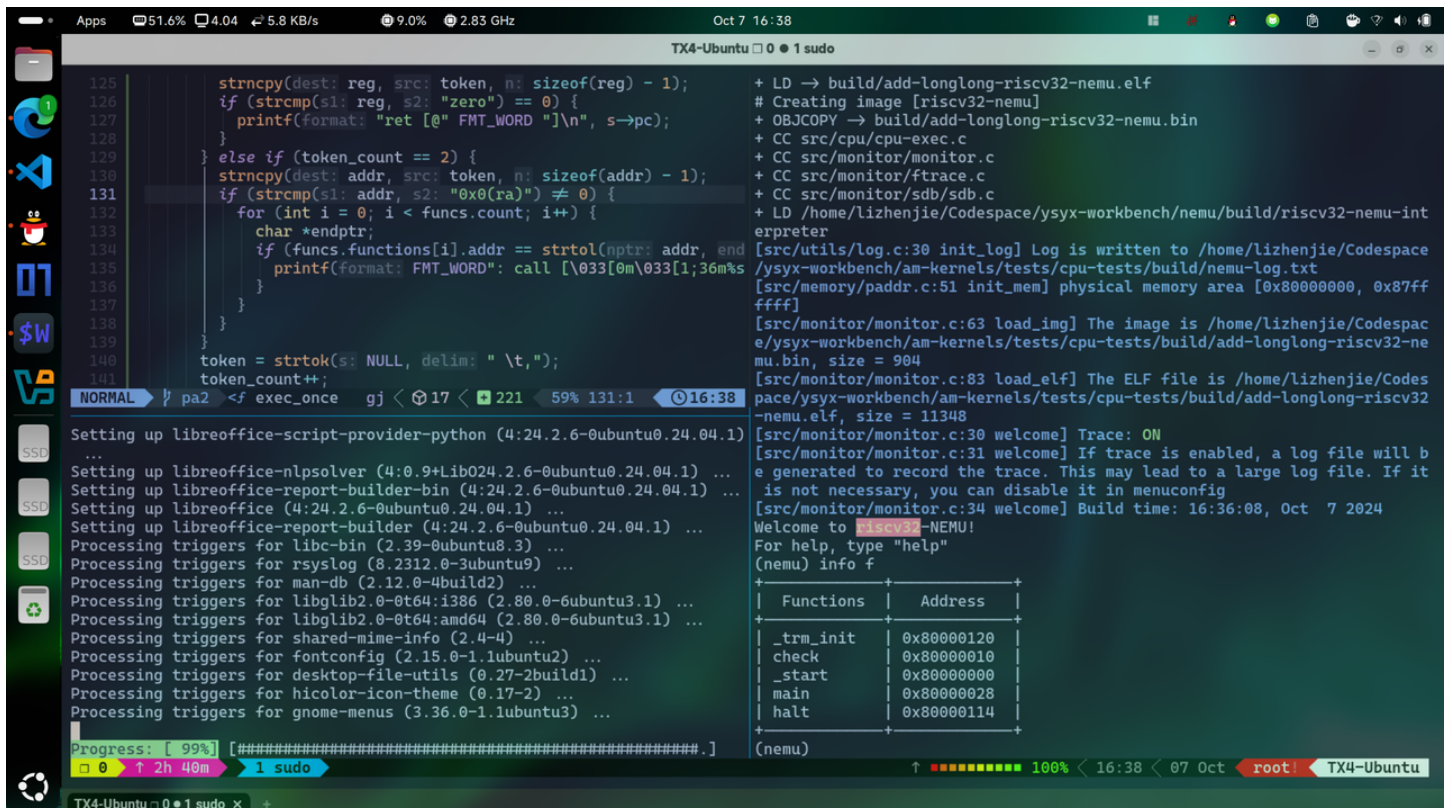


总结：vim部分学习你需要做的工作是配置自己的编程生产环境，因此自己用的舒服即可，后续学到更多也可以继续配置你自己的vim。

5.终端复用器——Tmux（安装使用即可）

你可能会很疑惑，终端能不能也“分屏”，这样只需要打开一个终端就能干很多活了，不需要vim开一个，apt开一个，gcc开一个.....

这时候，终端复用器就派上用场了。它可以实现“分屏”效果，大大提升我们干活/Coding的效率，就像这样：



cTmux的安装

```
1 $ sudo apt install tmux
```

Tmux的使用和配置

Tmux使用教程： <https://docs.hpc.sjtu.edu.cn/login/tmux.html>

如果想自定义你的Tmux的话，是完全OK的，上面这张图片就是配置后的Tmux

Tmux配置教程： <https://hamvocke.com/blog/a-guide-to-customizing-your-tmux-conf/>

考核要求：安装使用即可，配置随个人意愿

 终端复用器进阶：如果你对繁琐难看的tmux界面和配置感到厌烦，想了解其他的终端复用器，我们推荐先了解使用一下zellij吧！

还想了解更多？欢迎大家Bing或Google！

6.C语言（请在linux环境下完成）

注意：在做之前确保你安装了vim和gcc

```
1 $ sudo apt install vim gcc
```

基础学习内容及作业（必做）

本周学习内容：

函数，数组（PS：如果你好好做了第一次的作业，那么你应该已经学完了循环，如果没有，Sorry！这周作业对你来说可能有些多了。。。）

如何在Linux下运行你的程序（需要用到gcc，暂时不展开它的其他用法）：

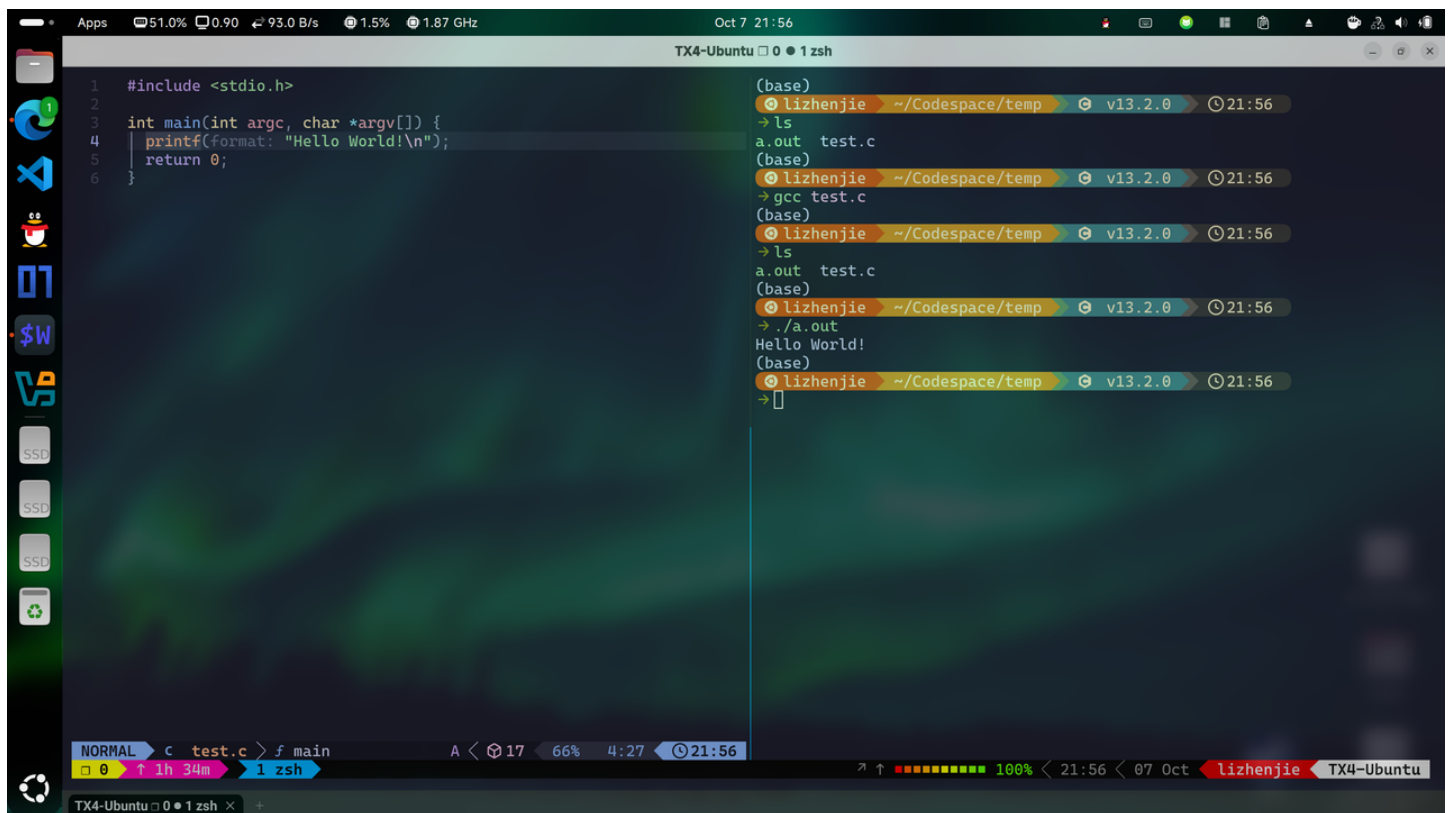
```
1 $ gcc <source_code.c>      #不指定程序名，在当前目录下输出a.out
2 $ gcc <source_code.c> -o <program_name>    #指定程序名
```

例如：

```
1 $ ls
2 program1.c  program2.c
3 $ gcc program1.c -o prog_1
```

```
4 $ ls
5 program1.c program2.c prog_1
6 $ ./prog_1
7 <程序prog_1 (源码program1.c) 的运行输出>
8 $ gcc program2.c
9 $ ls
10 program1.c program2.c prog_1 a.out
11 $ ./a.out
12 <程序a.out (源码program2.c) 的运行输出>
```

LZJ学长的碎碎念：为什么我们要让大家学tmux呢，因为这样就可以：（太方便了不是吗）



（碎碎念：我们知道本周C语言学习任务会有一些多，大家要花时间去消化知识，因此我们不会给出难的编程作业，下面任务是C语言最基础的学习内容，**任选其一做完即可**）

选择1.C语言推荐书籍：

- 《C Primer Plus》（基础）
- 《C与指针》（对指针讲的很透彻）
- 《C陷阱与缺陷》（涵盖了初学到进阶的各种常见错误）
- 《C专家编程》（讲解了一些C的高级用法）
- 《C程序设计语言》（黑皮书，主要是从标准角度讲解，写的很精练，由C语言的创造者所编写，但是不推荐一上来就看，对C有一定了解之后再）
- Linux C编程一站式学习<https://akaedu.github.io/book/>

以上书籍对应范围：结束函数，数组部分学习

选择2.C语言推荐视频：

<https://www.bilibili.com/video/BV1XZ4y1S7e1/?>

[vd_source=4ec31615294fd2510d5fd40f0183648f](https://www.bilibili.com/video/BV1XZ4y1S7e1/?vd_source=4ec31615294fd2510d5fd40f0183648f)，对应范围：看到第80p数组结束（翁恺老师一大特征就是讲课详细，但是带来的代价就是视频长，讲课相当慢；推荐1.5-2倍速且搭配跳转食用，尤其是题目讲解部分，直接上手实践即可，不要一直参考老师的思路）

选择3.C语言推荐练习习题：（基础即可，提高作业不要求做）

基础：慕课<https://www.imooc.com/learn/249>（对应范围：4-6章）（虽然比较简单，但是用于练手还是可以的）

提高：头歌湖南工业大学C语言题目<https://www.educoder.net/paths/klbm7gto>（——某软件学长碎碎念：难度高一些，但是这和咱们一些老师会安排的C语言课后作业大部分重复，所以学校的C语言课和作业会非常轻松）

提高学习作业（必做）

排序

题目解释：

用户需输入10个整数，程序对其进行排序。

要求：

奇数全在前面，偶数全在后面，并且按照从小到大的顺序输出。

例如：

输入：9 96 23 21 6 200 2 28 92 10

输出：9 21 23 2 6 10 28 92 96 200

效果演示：

```
enter your number1 2 50 38 62 49 72 59 83 77 80
1 49 59 77 83 2 38 50 62 72
```

拔高作业（挑战自己）

PS：以下三个任务，全部完成就是非常优秀了，全部完成难度比较高，完成多少都可以，交回即可。这三个题大家不仅要有对应的基础知识，还需要多加思考，比较锻炼思维，大家加油！

1.Shell模拟器

继续改进你的Shell模拟器，想必你选择做拔高作业已经学完了函数部分，那么请你用函数编程的思想将以下两个程序做进你的Shell内，让你的Shell可以用1, 2启用这两个程序（函数）。

类似这样：

```
61 //副函数1 九九乘法表
62 void program1(){
63
64
65
66
67
68
69
70 }
71 //副函数2闰年判断
72 void program2(){
73
74
75
76
77
78 }
79 //副函数3 排序
80 int program3() {
81
82
83
84
85
```

效果示例：（这样的形式即可，输出大家自定）

```

[xpy69 ~/Desktop] > ./shell

(xpy69) > help
(search how to go on the order)
do:(do the program what you enter)
list:(all program you can do)
exit:(exit the terminal)
help:(search how to go on the order)
(xpy69) > list
(all program you can do)
do
list
help
exit
(xpy69) > do 1
(do the program what you enter)
1*1= 1  1*2= 2  1*3= 3  1*4= 4  1*5= 5  1*6= 6  1*7= 7  1*8= 8  1*9= 9
2*2= 4  2*3= 6  2*4= 8  2*5=10  2*6=12  2*7=14  2*8=16  2*9=18
3*3= 9  3*4=12  3*5=15  3*6=18  3*7=21  3*8=24  3*9=27
4*4=16  4*5=20  4*6=24  4*7=28  4*8=32  4*9=36
5*5=25  5*6=30  5*7=35  5*8=40  5*9=45
6*6=36  6*7=42  6*8=48  6*9=54
7*7=49  7*8=56  7*9=63
8*8=64  8*9=72
9*9=81

(xpy69) > do 2
(do the program what you enter)
you have enter the leap year judgment
2000
2000 is leap year

(xpy69) > do 3
(do the program what you enter)
enter your number1 2 50 38 62 49 72 59 83 77 80
1 49 59 77 83 2 38 50 62 72
now you are adding nums,enter 0 to break: 28
0
1 49 59 77 83 2 28 38 50 62 72 80

```

2.两数之和

给定一个整数数组nums和一个整数目标值target（自己出一个），请你在该数组中找出和为目标值target的那两个整数，并返回它们的数组下标。

你可以假设每种输入只会对应一个答案，并且你不能使用两次相同的元素。

你可以按任意顺序返回答案，**只会存在一个有效答案**。

PS：不要面向结果编程！！

输出效果演示（输出格式按照这个来）：

```
lizhenjie ~/Codespace/temp v13.2.0 20:57
→ ./a.out
nums = 2 7 11 15 3 4
target = 13
index: [0, 2]
2 + 11 = 13
```

思考题（给进度超前的同学）：如何把这个封装成一个函数，然后接受任意一个整数数组呢？

3.排序改进

将提高作业的程序进行改进

完成排序并输出后增加选择，输入1可以添加新数字，仍要求奇数全在前面，偶数全在后面，并且按照从小到大的顺序输出。输入0退出。

效果展示：

```
enter your number1 2 50 38 62 49 72 59 83 77 80
1 49 59 77 83 2 38 50 62 72
now you are adding nums,enter 0 to break: 28
0
1 49 59 77 83 2 28 38 50 62 72 80
```

作业提交

将图片放在一个文件夹里，文件夹命名为 **你的姓名-专业班级**，（PS：我们第一次作业提交还是有一些命名不规范的，我们也会进行记录，细节决定成败）

必做基础任务：

1. **Markdown**部分作业，请提交你的**md文档**（里面内容为名字-专业班级），参考文件包内范例即可。
2. **APT**为必用工具，因此这里不用提交相关凭证（此部分不用提交）
3. **Vim**也是必用工具，这里提交的作业为你的**~/vimrc**，以及你打开一个C语言文件编程时的**编程环境截图**，具体参考文件包内的范例。
4. **Tmux**，提交作业提交你的Shell截图即可，具体参考文件包内的范例。
5. **C语言基础作业**：
 - a. **选择1或2**：请提交你用md做的学习笔记（键位实在是不熟可以纸质），过程中你实践做的程序也需要放在文件夹里命名为C语言提交。
 - b. **选择3**：请提交慕课完成**4-6章或者头歌**（难度较高，建议有基础的人去尝试）完成**5，6部分编程作业的截图**。命名为C语言。

示例截图如下：

第5部分 函数



5-1 函数

实验数 6

5-2 递归函数、嵌套函数

实验数 4

第6部分 数组



6-1 一维数组和二维数组

实验数 7

6-2 字符数组

实验数 6

6. C语言提高作业：C原文件命名为排序放入C语言文件夹即可。

7. C语言拔高作业：如果你完成了Shell任务，那么只应该有一个C文件——Shell，反之给到程序对应的名字提交即可。

(XPY学长留言：有一部分同学C语言已经几乎学完了，那么本周作业对他来说很轻松，请来找我，我会布置其他的拔高作业)

最后将以上你的姓名-专业班级文件夹压缩为一个压缩文件（右键压缩/Commpress即可）。

范例请参考文件夹内另一个文件

虚拟机发邮件：

- 浏览器打开邮箱网站发送；（就和实体机一样，最方便快捷）
- 小蓝鸟thunderbird亦可，具体方式请上网查询（提示：现代邮箱需要使用POP3/IMAP/SMTP/Exchange/CardDAV/CalDAV服务，不能使用密码）；
- 或者建立共享文件夹，将作业打包到win上，然后发送邮件。具体方法也请自己查询。

请严格按照上述要求发送到邮箱：yunding_ysyx@163.com

拓展视野

学累了吗？不如来练练打字？<https://dazidazi.com/>

编译流程与GCC编译器

C程序是怎么从源代码变成可执行文件的？这就需要编译器了。

在你的DevCpp或者Visual Studio上，可能就只是一个按键的事，但是在这之后又发生了什么呢？

对这个过程感兴趣的同学可以去搜索关于C语言编译运行相关的知识，了解即可。

芯片是怎么制造的

<https://www.bilibili.com/video/BV1kxLeBEDD>

本作品《"太理工一生一芯工作室前置讲义见习学员培养篇"》由 许鹏远 创作，并采用 CC BY-SA 4.0 协议进行授权。

遵循CC BY-SA 4.0开源协议：<https://creativecommons.org/licenses/by-nc-sa/4.0/deed.en>

转载或使用请标注所有者：许鹏远，太理“一生一芯”工作室