

The Missing Course of Your Computer Science Education

王慧妍

why@nju.edu.cn

南京大学



计算机科学与技术系



计算机软件研究所



本讲概述

- 关于《计算机系统基础》习题课
- 关于传说中的PA实验
- 学术诚信
- 选讲PA0

关于《计算机系统基础》习题课

课程网站: http://www.why.ink:8080/ICS/2022/Main_Page

实验网站: <https://nju-projectn.github.io/ics-pa-gitbook/ics2022/>

习题课的内容

- 《计算机系统基础》理论课中遗失的实践细节
- 对你们从Lab/PA中生存下来是至关重要的
- 对你们未来XX年作为“码农”的生涯都是至关重要的
- 本讲标题来自于[The Missing Semester of Your CS Education](#)
 - jyy墙裂推荐!
 - 我们miss的比这门课多
- 习题课（和课程网站）会布置所有作业
 - PA（编程大实验）
 - Lab（编程小实验）
 - Homework（书后习题）——以理论课为准

老师/助教的使用

- 联系/求助：使用邮件

- 能使你更好地整理问题。也许在整理问题的过程中你就发现答案了

- 老师：

- 王慧妍：why@nju.edu.cn;

- 助教：

- 刘瀚之 jm233333@outlook.com

- 陈楚阳 chuyangchen2018@outlook.com

- QQ群：

- 匡院：733678133

- AI学院：577944061

- Ask

- 对上课内容没有理解的地方、对课程/学习的疑惑、.....

- Do not ask

- 安装XXX错了怎么办？ Segmentation Fault怎么办？



关于传说中地狱难度PA的一些真相

- 它很难，的确很难
 - 确切地说，对最优异的同学来说依然有一些挑战性
 - 如果你感到异常困难，你更需要的其实是C/C++编程的训练
 - （并且我们知道这一点！）
- 往年一些同学都因为不诚信的举动获得了成绩
 - 他们得到了相应的报应（例如在《操作系统》中惨挂）
 - （并且我们知道这一点！）
- 你总是可以耍一些小聪明，从别人那里得到帮助
 - 但越是独立完成，受到的训练就越好
 - （并且我们知道这一点！）

你们可能比较关心的

- 分数占比 (期末只剩下30%)
 - PA: 40%、Lab: 15%、Homework: 15%

- 评分标准

- DDL
 - 5% bonus
 - 100%封顶
- Hard DDL
 - 80%分数

PA0~4

3.2.1 PA: 几乎完全客观评分

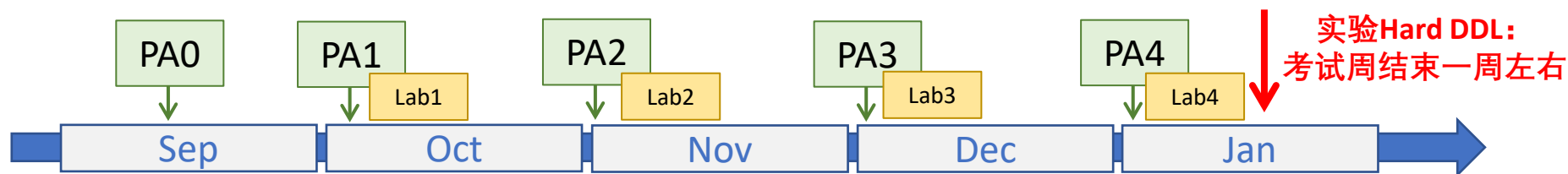
- Rejected, 编译错误或没有通过任何测试用例: 10% (诚信分)
- Partial Accepted, 部分 easy 测试通过 (此时不运行 hard 用例): 30%
- Partial Accepted, 全部 easy 测试通过: $\geq 50\%$, 剩余部分由人工评价给出
- Accepted, 通过全部 easy/hard 测试: $\geq 75\%$, 剩余部分由人工评价给出
- 没有通过全部 easy 测试用例的作业将没有人工评分的机会 (即意味着实验报告不得分。但我们会阅读你的反馈)。

3.2.2 Labs: 完全客观评分

- Rejected, 编译错误或没有通过任何测试用例: 10% (诚信分)
- Partial Accepted, 部分 easy 测试通过 (此时不运行 hard 用例): 50%
- Partial Accepted, 全部 easy 测试通过 (hard 用例没有全部通过): 75%
- Accepted, 通过全部 easy/hard 测试: 100%

Lab1~4

以课程网站为准

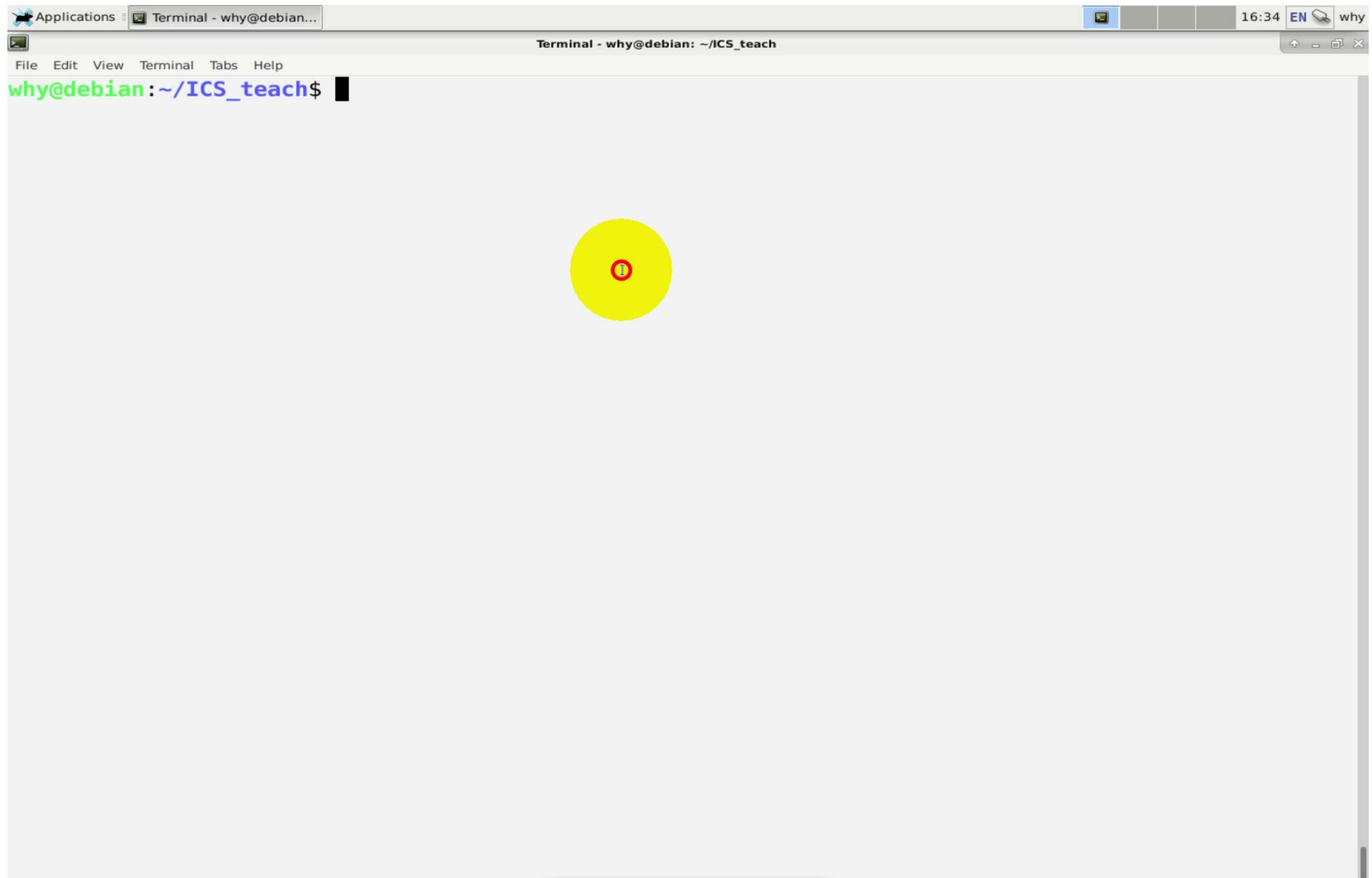


关于传说中的PA实验

PA是什么东西？

回答终极拷问

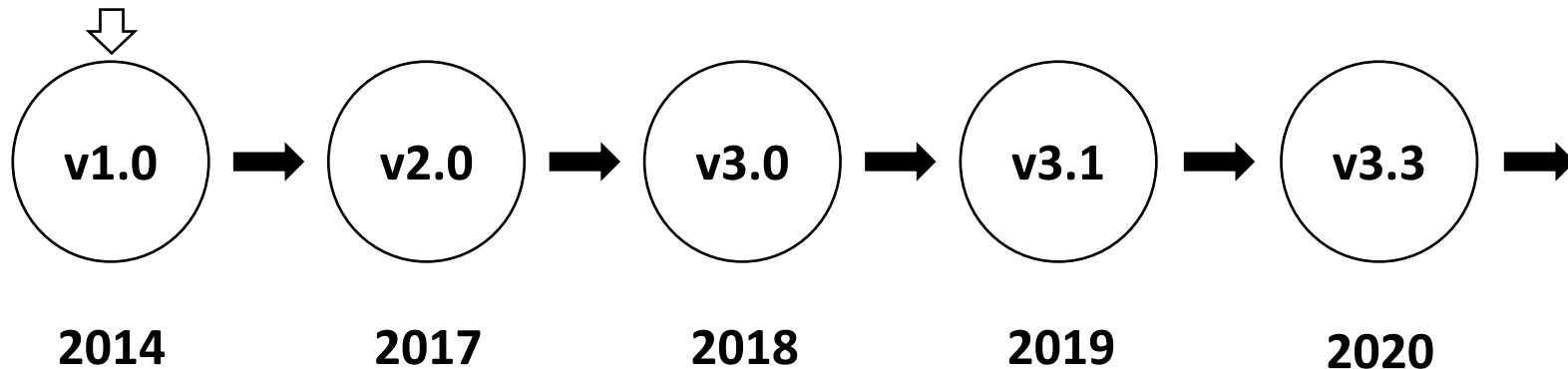
当你运行一个Hello World程序的时候，计算机究竟做了什么？



PA实验

- 功能完备但简化的模拟器NEMU(NJU EMUlator)的实
 - 实验环境配置 (PA0)
 - 四个连贯的实验内容 (PA1~PA4) , 即: 简易调试器、程序执行、cache与存储管理、异常与I/O

余子濠 开发完成



PA的简要历史

NEMU

- NEMU是一个用来执行其它程序的程序（怎么理解？）
 - 支付宝：用软件模拟硬件ATM
 - 取款、存款、转账、汇款等等



- NEMU: 用软件模拟出 “计算机”

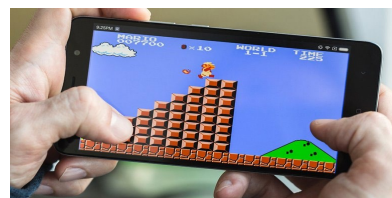


✓模拟PC机



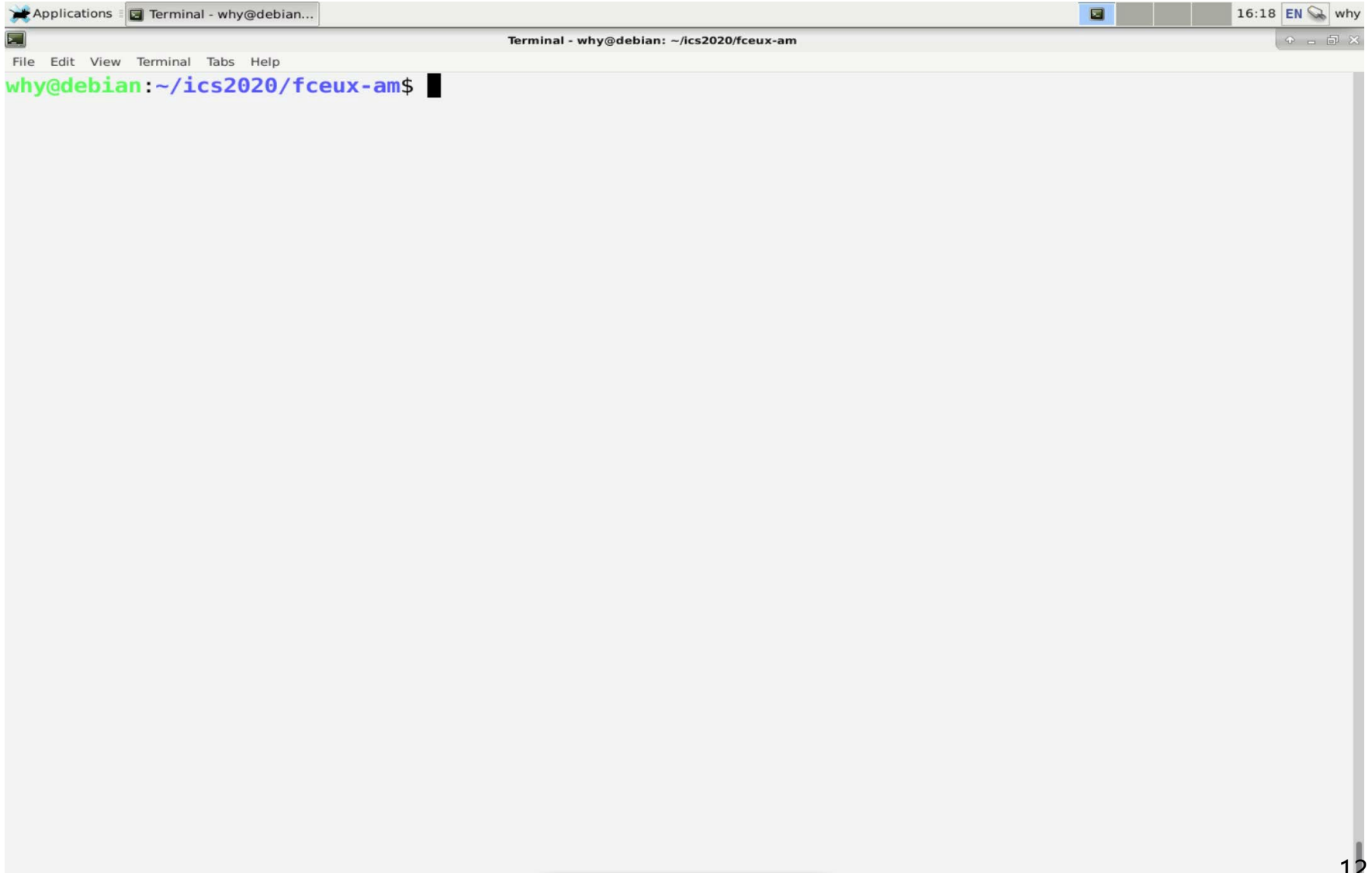
ANDROID

✓模拟手机

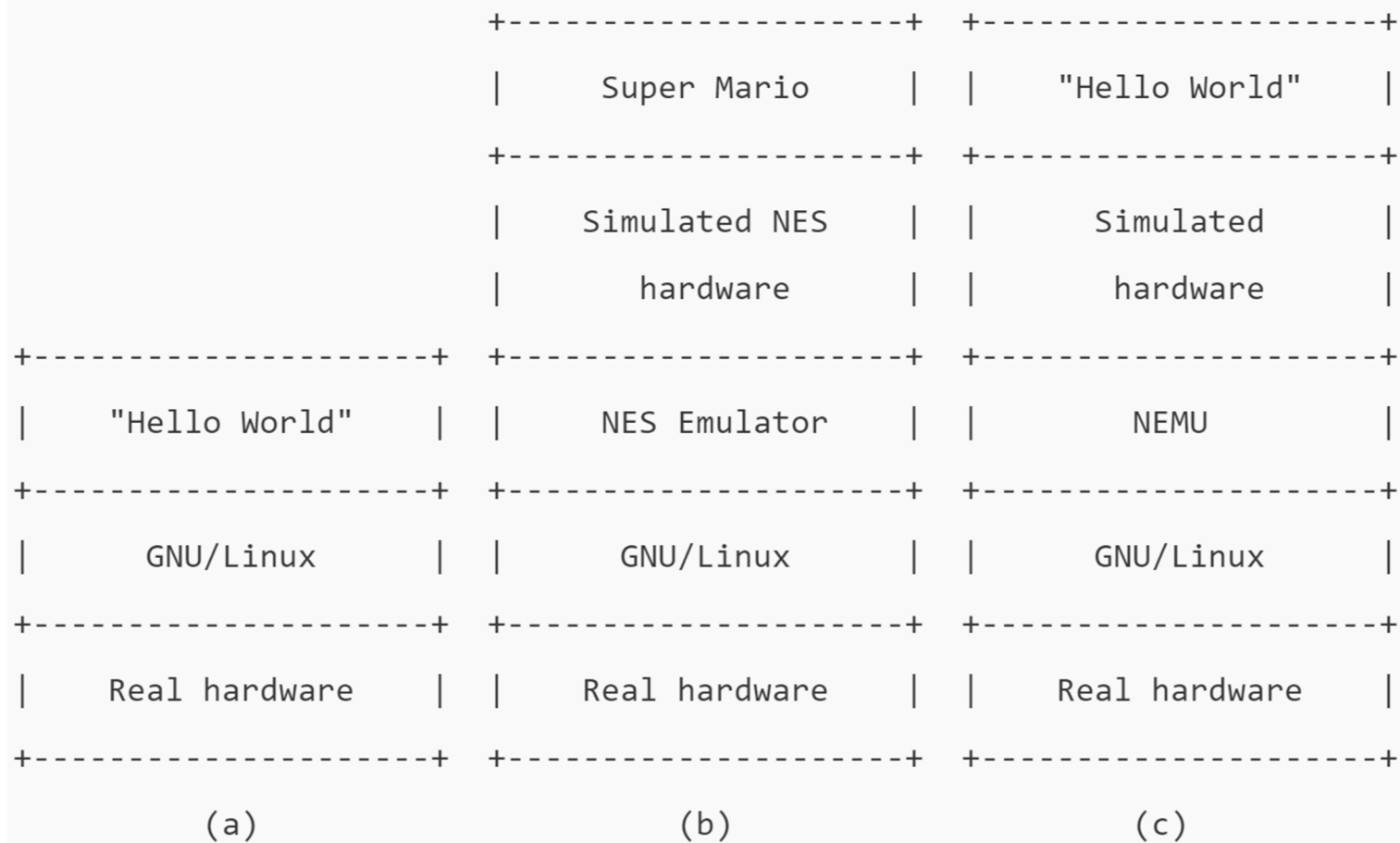


✓模拟游戏机

用红白机模拟器NES Emulator玩Mario



NEMU模拟器



学术诚信 (Academic Integrity)

学生手册：不能抄作业

- What is academic integrity?
 - [What is Academic Integrity? | Academic Integrity at MIT](#)

Plagiarism	Cheating		
Do:	Do:	Don't:	
Trust the	Demonstrate your own achievement.	Don't copy answers from another student; don't ask another student to do your work for you. Don't fabricate results. Don't use electronic or other devices during exams.	e a paper
Undertake work.	Accept corrections from the instructor as part of the learning process.	Don't alter graded exams and submit them for re-grading.	out citing
	Do original work for each class.	Don't submit projects or papers that have been done for a previous class.	
Unautho	Facilitating Academic Dishonesty		
Do:	Do:	Don't:	
Trust the	Showcase your own abilities.	Don't allow another student to copy your answers on assignments or exams. Don't take an exam or complete an assignment for another student.	d the

学生手册：不能抄作业

- What is academic integrity?
 - 简单概况：独立完成
- 针对作业的独立完成
 - 自己完成作业
 - 对使用的已有资料作出明确的标示
 - [ACM Policy on Plagiarism, Misrepresentation, and Falsification](#)
- 针对代码的独立完成
 - 自己完成代码的编写
 - 自己完成测试用例
 - 在允许的范围内使用他人的成果

具体案例：MIT 6.005 Elements of Software Construction

- Individual work
 - Problem sets in this class are intended to be primarily individual efforts. You are encouraged to discuss approaches with other students but *your code and your write-up must be your own*.
 - You *may not use materials produced as course work by other students*, whether in this term or previous terms, nor may you provide work for other students to use.
 - It's good to help other students. But as a general rule, during the time that you are helping another student, *your own solution should not be visible*, either to you or to them. Make a habit of closing your laptop while you're helping.

具体案例：MIT 6.005 Elements of Software Construction (cont'd)

- Using external resources
 - It's fine to use material from external sources like [StackOverflow](#), but only with proper attribution, and only if the assignment allows it. In particular, if the assignment says “implement X,” then you must create your own X, not reuse one from an external source.
 - It's also fine to use any code provided by this semester's 6.031 staff (in class, readings, or problem sets), without need for attribution. Staff-provided code may not be publicly shared without permission, however, as discussed later in this document.

具体的案例 (1)

- Alyssa and Ben sit next to each other with their laptops while working on a problem set. They talk in general terms about different approaches to doing the problem set. They draw diagrams on the whiteboard. When Alyssa **discovers a useful class in the Java library**, she mentions it to Ben. When Ben finds a StackOverflow answer that helps, he sends the URL to Alyssa. **OK**.
- As they type lines of code, they speak the code aloud to the other person, to make sure both people have the right code. **INAPPROPRIATE**.
- In a tricky part of the problem set, Alyssa and Ben look at each other's screens and compare them so that they can get their code right. **INAPPROPRIATE**.

具体的案例 (2)

- Jerry already finished the problem set, but his friend Ben is now struggling with a nasty bug. Jerry sits next to Ben, looks at his code, and helps him **debug**. **OK**.
- Jerry opens his own laptop, finds his solution to the problem set, and refers to it while he's helping Ben correct his code. **INAPPROPRIATE**.

具体的案例 (3)

- Louis had three problem sets and two quizzes this week, was away from campus for several days for a track meet, and then got sick. Ben feels sorry for Louis and wants to help, so he sits down with Louis and talks with him about how to do the problem set while Louis is working on it. Ben already handed in his own solution, but he doesn't open his own laptop to look at it while he's helping Louis. **OK.**
- Ben opens his laptop and reads his own code while he's helping Louis. **INAPPROPRIATE.**
- Ben has by now spent a couple hours with Louis, and Louis still needs help, but Ben really needs to get back to his own work. He puts his code in a Dropbox and shares it with Louis, after Louis promises only to look at it when he really has to. **INAPPROPRIATE.**

具体的案例 (4)

- John and Ellen both worked on their problem sets separately. They exchange their test cases with each other to check their work. **INAPPROPRIATE**. **Test cases** are part of the material for the problem set, and part of the learning experience of the course. You *are copying if you use somebody else's test cases, even if temporarily*.

具体的案例：PA0

PA0: 安装 Linux 系统，并提交空的文件。允许向互联网/同学求助。

- 遇到问题（如安装错误）找同学询问/解决 **OK**
 - 但你可能就失去了这门课原本的训练
 - 尽可能先自己解决
 - 帮其他同学解决问题的人
 - 一起还原解决问题的过程
- 请别人安装系统，或使用他人的虚拟机镜像 **INAPPROPRIATE**

Academic integrity

- 感到三观尽毁？

- 原来拿个测试用例也违反academic integrity？
- 拿个大腿的作业来改改不香吗？我还读懂了呢！
- 老师压根就没精力管，对他来说吃力不讨好
 - 有些事情是“天然”被禁止的
 - 但是我们的教育里缺失了“这是不对的”

- 那些痛苦是对你的训练（training）

- “看懂”和“自己设计测试用例、自己做出来”天差地别
 - PA难度无形剧增←过去没有academic integrity欠的债
- 不要看不起美国人
 - （大部分）学生真的明白并自发地执行这个标准

知乎贴：作业抄袭中的人生百态



农夫山泉
NONGFU SPRING

我们不生产~~水~~,我们只是~~大自然的搬运工~~

代码

互联网



代码抄袭：那些让985学

```
1 void P1() {
2 ...
3 puts("Game_Over");
4 ...
5 }
6 void P2() {
7 ...
8 puts("G"); puts("a")
9 puts("e"); puts("_")
10 puts("v"); puts("e")
11 ...
12 }
```

```
1 cur->lineno = temp->lineno;
2 strcpy(cur->type, type);
3 cur->isLexical = 0;
4 cur->children = temp;
```

```
1 $$->is_root=1;
2 $$->no_leaves=1;
3 $$->leaves[0]=(Node*)$1;
4 if(exit_error==0)
5 {print_tree($$,0);}
```

```
1 temp->line = a->line;
2 temp->lChild = a;
3 while(num > 1){
4 a->rChilds = va_arg(list,
5 node*);
6 a = a->rChilds;
7 num--;
```

```
1 head->number_signal = 0;
2 head->line = temp->line;
3 strcpy(head->type,type);
4 head->child_left = temp;
```

```
1 $$->final=0;
2 $$->num_children=1;
3 $$->children=(Node**)malloc
4 (sizeof(Node*)*$$->
5 num_children);
6 $$->children[0]=(Node*)$1;
7 if(!wrong)
printNode($$,0);
```

```
1 p_node->left_child = temp;
2 p_node->line = temp->line;
3 for(int i=0;i < num-1;++i){
4 temp->right_child =
5 va_arg(valist,struct
6 Node*);
7 temp = temp->right_child;
```

知乎 @蒋炎岩7


[About FYTGS](#)
[Curricula](#)
[Regulations](#)
[Academic Requirements](#)
[Awards & Grants](#)
[Resources](#)
[Aims and Objectives of RPg Education](#)
[Strategic Framework for TPg Education](#)
[HKUST Fok Ying Tung Graduate School](#)
[News](#)
[抄袭](#)
[学术](#)
[论文](#)
[ICCV](#)
[科研热点](#)

关注者

3,449

被浏览

6,642,782

如何看待 ICCV21 接收的某港科大学生为一作的论文被指抄袭 ICML21 发表的论文？

ICCV21接收论文m-RevNet: Deep Reversible Neural Networks with Momentum被指出与 ICML21接收论文Momentum residual neural networks在核心思路、实验和图表上有多处雷同，疑似抄袭

ICML21论文作者的声明和详细的抄袭证据：

<https://michaelsdr.github.io/momentumnet/plagiarism/>

[🔗 michaelsdr.github.io/momentumnet/plagiarism/](https://michaelsdr.github.io/momentumnet/plagiarism/)

被指抄袭的论文的一作个人主页网页快照：

Paper A (Duo Li and Shang-Hua Gao, Peyré)

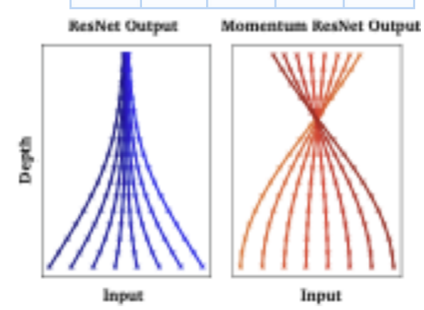
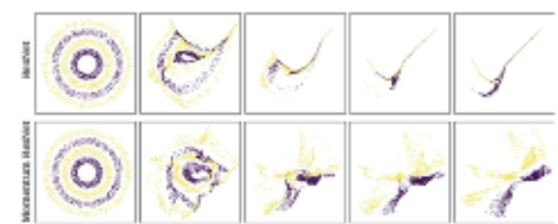
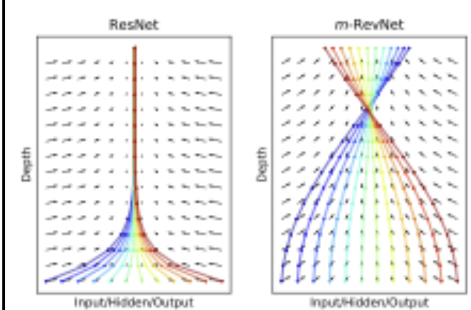


Table 1. Comparison of reversible residual architectures

	NeurODE	i-ResNet	i-RevNet	RevNet	MomNet
Closed-form inversion	✓	✗	✓	✓	✓
Same parameters	✗	✓	✗	✗	✓
Unconstrained training	✓	✗	✓	✓	✓



Paper B (Duo Li and Shang-Hua Gao)



Method	ResNet	RevNet/i-RevNet	i-ResNet	NODE (and variants)	m-RevNet (ours)
Analytical Reversal	N/A	✓	✗	✗	✓
Architectural Preservation	N/A	✗	✓	✗	✓
End-to-End Optimization	✓	✗	✓	✓	✓

Table 1. Comparing m-RevNet and ResNet to RevNet, i-RevNet, i-ResNet, NODE (and its variants). Architectural preservation refers to

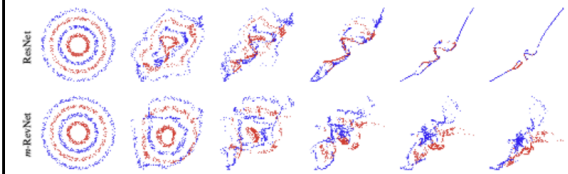


Figure 2. Comparison of learning dynamics between ResNet (top) and m-RevNet (bottom) in a 2D example. The leftmost distribution of

Notes

Same figure, added vector field and changed colors.

The three rows have the same meaning: closed form inversion = analytical reversal, same parameters = architecture preservation and unconstrained training = end-to-en optimization. The columns are shuffled, and Paper B added the ResNet baseline.

Choice of the exact same initial dataset, four nested rings. The usual experiments as found e.g. in <https://arxiv.org/abs/1904.01681> as cited by both papers rather use 2 nested rings. To the best of our knowledge, using four nested rings as an illustration had never been done before.

Installing GNU/Linux

First Exploration with GNU/Linux

Installing Tools

Configuring vim

More Exploration

Getting Source Code for PAs

PA0 选讲：进入Linux世界

学会用Linux工作吧

让时间回到1980s

- 就算回到1980s, 该做的事情还得干啊
 - 管理文件; 编代码; 写作业; 排版杂志; 上网.....

- 例子

- `vi a.txt`
- `ip addr / ping baidu.com`
- `df -h /`
- `find . -name "*.cpp"`
- `fdisk /dev/sdb`
- `shutdown -h 0`
- `apt install qemu-system`
- `pdfjoin a.pdf b.pdf`
- `iconv -f gbk -t utf-8 file.txt`



现在都2022年了，还整这些玩意干嘛？

上面这些事情不是点点鼠标就能搞定的吗？

- 被迫接受
 - ~~不学你就挂子~~
- 主动接受
 - 对系统更强的控制力
 - 应用程序通常无法满足 power user 的全部需求
 - 生产系统编程的事实标准
 - Linux, macOS, Windows, ...运维基本都靠命令行工具
 - 来自开源社区的一份礼物
 - 非常丰富、可定制、看得见源码的软件栈
- [流传自远古时代的OS实验课程网站中的Linux入门教程](#)

Linux命令行概述

第一课

- 这是个啥？让我用这个度过余生？

\$ █

Unix is user friendly. It's just selective about who its friends are.

Read the manual.

Search the web.

为什么大家在一开始都感到困难？

- 1980s: 以 MHz 为单位的主频；80 x 24 的字符终端
 - 无法提供丰富的交互界面
 - 但依然要完成各类任务
 - 你会如何设计？

不可避免，需要用户查阅手册记住一些系统里的约定（下面是最重要的一些）。

- 目录结构、文件命名规律、访问权限
- 命令执行的约定
- 常用命令行工具
- Shell 编程语言

常见的命令行工具

```
missing:~$ date  
Fri 10 Jan 2020 11:49:31 AM EST  
missing:~$
```

```
missing:~$ ls  
missing:~$ cd ..  
missing:/home$ ls  
missing  
missing:/home$ cd ..  
missing:/$ ls  
bin  
boot
```

```
missing:~$ ls -l /home  
drwxr-xr-x 1 missing users 4096 Jun 15 2019 missing
```

```
...
```

常见的命令行工具

```
missing:~$ pwd
/home/missing
missing:~$ cd /home
missing:/home$ pwd
/home
missing:/home$ cd ..
missing:/$ pwd
/
missing:/$ cd ./home
missing:/home$ pwd
/home
missing:/home$ cd missing
missing:~$ pwd
/home/missing
missing:~$ ../../bin/echo hello
hello
```

常见的命令行工具

```
missing:~$ echo hello  
hello
```

```
missing:~$ echo $PATH  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin  
missing:~$ which echo  
/bin/echo  
missing:~$ /bin/echo $PATH  
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

```
missing:~$ echo hello > hello.txt  
missing:~$ cat hello.txt  
hello  
missing:~$ cat < hello.txt  
hello  
missing:~$ cat < hello.txt > hello2.txt  
missing:~$ cat hello2.txt  
hello
```

常见的命令行工具

- 查看文件
 - ls (list), cd (change directory), find, tree, ...
- 阅读手册
 - man (man man); apropos; info
- 文本处理
 - cat (concatenate), wc (word count), grep (g/re/p)
 - tr (translate), cut, awk, sed (stream editor)

命令行工具多到什么程度呢.....

- unzip, strip, touch, finger, grep, mount, fsck, more, yes, unmount, sleep.....

UNIX哲学

- *Keep it simple, stupid.* (KISS)
 - *Everything* is a file and *pipeline* programs to work together
- 一个命令只做 “一件事”
 - 从stdin输入 (`printf`)
 - 向stdout输出 (`scanf`)
 - 使用参数控制行为 (`int main (int argc, char * argv[]);`)
- 命令的输入和输出都是人类 + 机器均可读的文本
 - `find .`
 - `wc -l a.txt b.txt`
- 把命令的输入/输出连接起来 (管道) 协作完成任务
 - `find . | grep '\.cpp$' | xargs cat | wc -l`

The Shell Scripting Language

祝贺！刚才其实是Shell执行了一段Shell语言编写的程序。

- Shell 是一门基于字符串和命令的编程语言
 - `a=hello` - 赋值 (注意 = 左右没有空格)
 - `$a` - 将变量的值 “粘贴”
 - `$(cmd)` - 将 `cmd` 运行的 stdout “粘贴”
 - `if cmd; then; ... ; fi` - 根据 `cmd` 运行结果执行分支
 - `cmd > file` - 把 `cmd` 的 stdout 重定向到 `file`
 - `cmd1 | cmd2` - 把 `cmd1` 的 stdout 作为 `cmd2` 的 stdin
- 有趣的小问题
 - 如何用 `if` 比较存储了整数字符串的大小？
 - (`if 1 > 2` 会发生什么？)

开始编程吧！

- 输出当前用户是不是root
 - `[ $UID -eq 0 ] && echo "is root!"`
- 查看磁盘引导扇区 (Master Boot Record)
 - `cat /dev/sdb | head -c 512 | ndisasm -b 16 -o 0x7c00 -`
- 统计所有cpp文件的行数
 - `find . | grep '\.cpp$' | xargs cat | wc -l`
- 统计命令行命令的频率
 - `history | tr -s ' ' | cut -d ' ' -f3 | sort | uniq -c | sort -nr`
- 以上都是命令行里的命令，但同时也是 bash script 的程序
 - (这就是程序员的宿命吧)

自动化工具：程序员福利系列

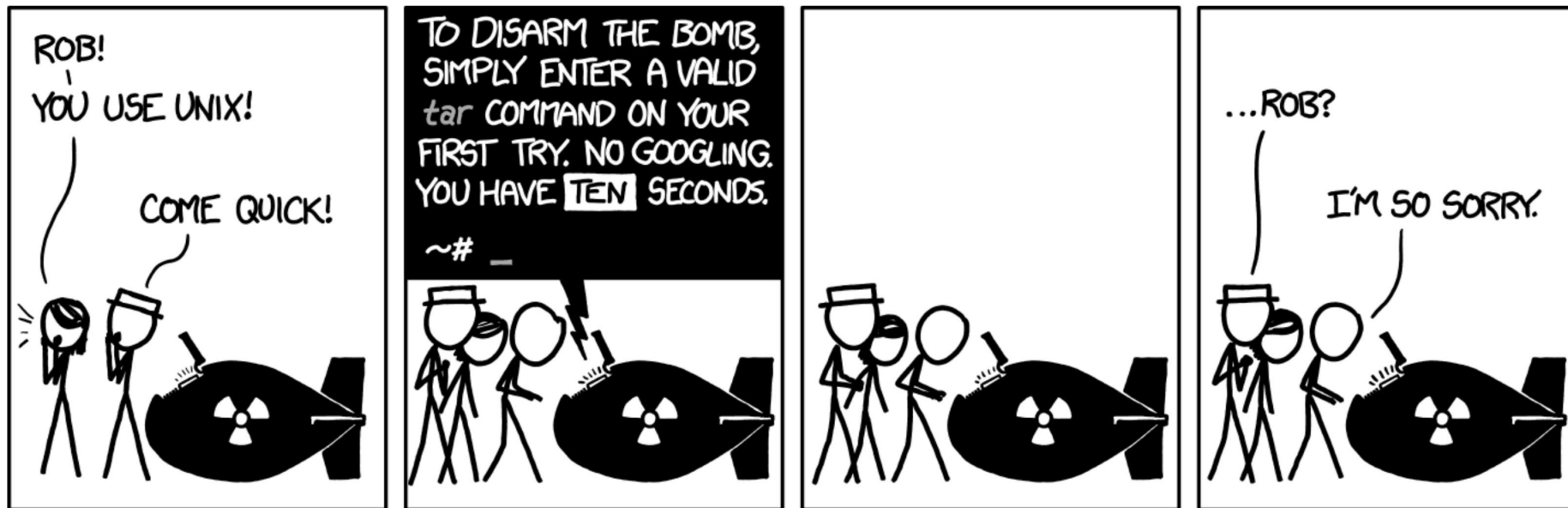
- 重构福利
 - (ex) 对一个目录里的所有.cpp文件执行同样的vim动作
- LaTeX用户福利
 - (pdftocrop) 把LibreOffice的演示文稿导出为PDF，然后将每一页的白边部分裁去，分别命名为fig-1.pdf, fig-2.pdf, ...
- 视频制作福利
 - (ffmpeg) 为视频添加水印和字幕

生存指南

拥抱变化

- 很烦躁：一下就碰壁？
 - 胡乱尝试一通？对了就对了，不对就抓瞎？
 - 有时候不知道该怎么STFW，RTFW？
- 静下心来，从头开始
 - [The Missing Semester of Your CS Education](#)
 - [流传自远古时代的OS实验课程网站中的Linux入门教程](#)
 - RTFM (slides), STFW, RTFSC
 - 不要觉得Makefile、提交脚本都是和你无关的
- 试图理解一切事情是如何发生的
 - 从读得懂开始，逐渐过渡到会写

现代化你的命令行工具



- 例子:

- tldr (替代man)
- zsh-z (替代pushd/popd)
- fzf

拥抱开源社区

- 用好 Github的 “[Awesome](#)” 系列
 - 例如[The art of command line](#)
- 用好 Stack Overflow / Stack Exchange
- 禁用百度和中文关键字（强烈不推荐大家使用中文系统）
 - Linux/macOS
 - /etc/hosts
 - Windows
 - C:\Windows\System32\drivers\etc\hosts
 - 增加一行127.0.0.1 www.baidu.com

选讲PA0

总结

本次课程最重要的内容

- 静下心来，从头开始
 - [The Missing Semester of Your CS Education](#)
 - RTFM (slides), STFW, RTFSC
- 用好 Github的 “[Awesome](#)” 系列
 - 例如[The art of command line](#)
- 用好 Stack Overflow / Stack Exchange
- 禁用百度和中文关键字

总结

- 你不会感到学习这门课很舒适
 - 不要用 “我们学得比较理论……” 来骗自己
 - 就是不扎实
 - Academic integrity 可能让你感到三观尽毁
 - 你会理解到南大还不是 “世界一流大学”
- 但请不要放弃/躺倒
- 你们未来是要承担大事业的

[PAO](#)已发布，暂时未上线OJ，请关注群通知

一些福利建议

- 用好Git
- 关注DDL

机器永远是对的 (and RTFM)

RTFM: Read The Friendly Manual

STFW: Search The Friendly Web

自动化工具：程序员福利系列（cont'd）

```
#自动登陆p.nju.edu.cn不香吗？  
curl -d "username=学号&password=密码" \  
http://p.nju.edu.cn/portal_io/login
```

- 这有什么用？
 - 校园网内某台机器的长久连接（brasd）
 - 作为你的校内代理服务器
 - 实现内网穿透.....
- 一个有趣的问题
 - 密码是明文，被舍友偷窥了怎么办？？？
 - 文件系统有权管理：chmod -r
 - 但是shell script必须读权限？