2021 Spring

Advanced Data Structure

Zhengwei QI，Mingkai Dong

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Teachers & TAs

- Zhengwei QI (戚正伟)
  - qizhwei@sjtu.edu.cn
- Mingkai Dong（董明凯)
  - dmkaplony@sjtu.edu.cn
- TAs
  - 贾兴国
  - 张孝东
  - 江学强
  - 项羽心

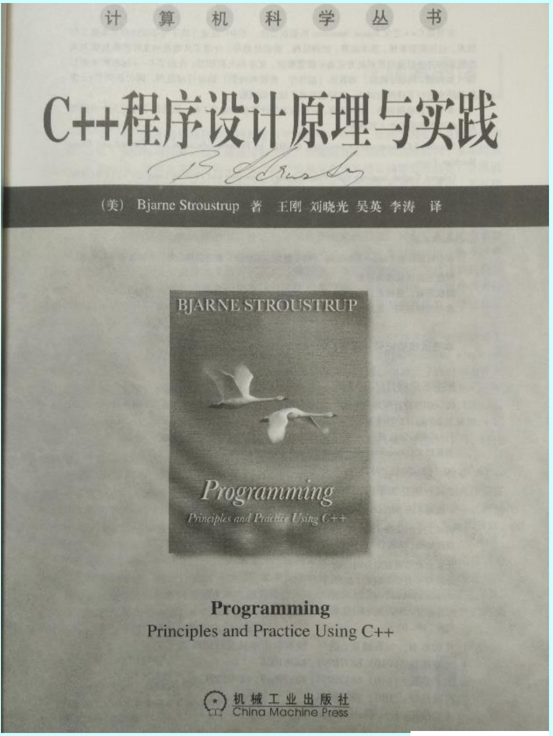
Prerequisites & Textbooks

- Textbooks
  - 数据结构（C++语言版）第三版, 邓俊辉, 清华大学出版社, 2013年9月
  - 讲义 待出版
- Reference books
  - Roberts, Eric S. Programming Abstractions in C++. ISBN 978-0133454840.

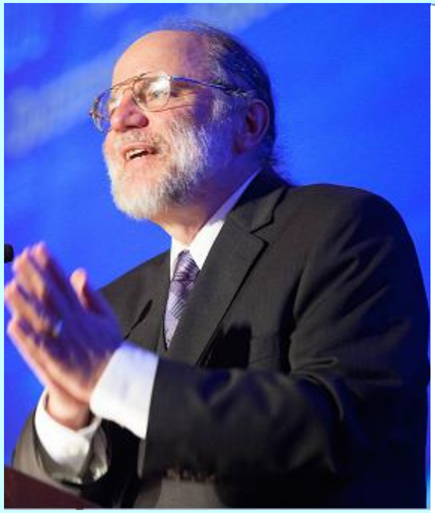
Rough course outline

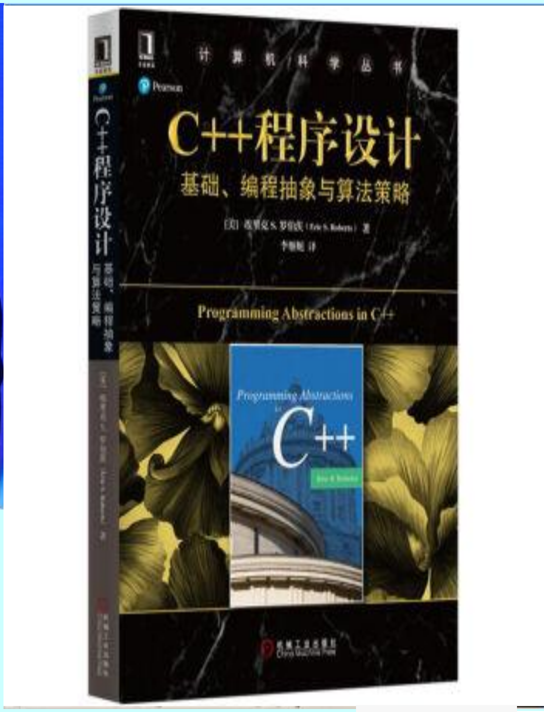
- Advanced Data Structure
  - Advanced data structures, including advanced search trees, graphs, collections, heaps, etc.. And through designing an integrated software system as several projects, training the ability to use a variety of data structures.

数据结构-相关书籍


计 算 机 科 学 丛 书
C++程序设计原理与实践
（美） Bjarne Stroustrup 著 王刚 刘晓光 吴英 李涛 译
BJARNE STROUSTRUP
Programming
Principles and Practice Using C++
Programming
Principles and Practice Using C++
机械工业出版社
China Machine Press

$47.85



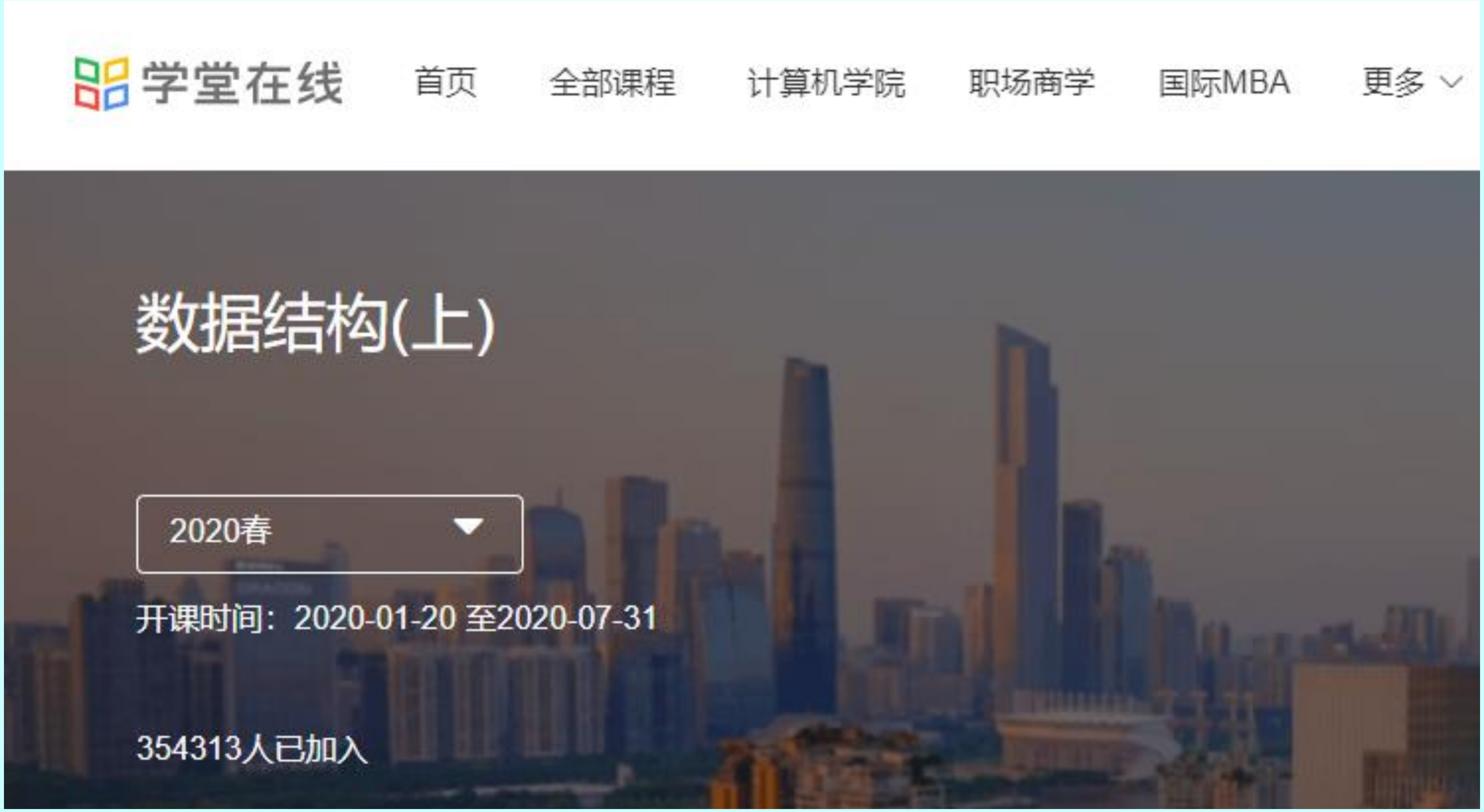


¥117.40

PPP：Programming : principles and practice using C++

PA：Programming Abstractions in C++.

MOOC



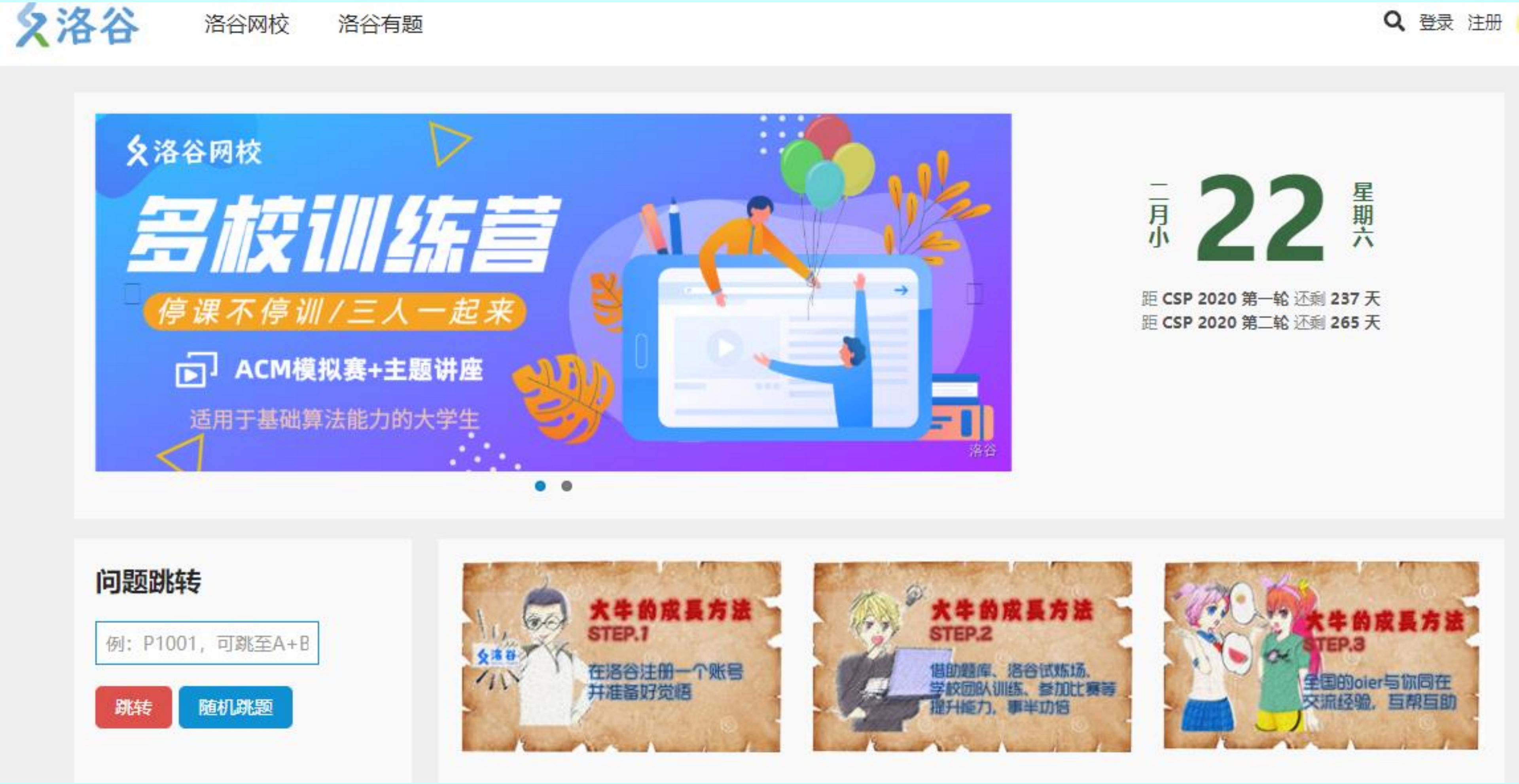
数据结构(上)

开课时间：2020-01-20 至2020-07-31

354313人已加入

https://next.xuetangx.com/course/THU08091000384/1516243

OJ-https://www.luogu.org/



洛谷 洛谷网校 洛谷有题 登录 注册

洛谷网校

多校训练营

停课不停训/三人一起来

ACM模拟赛+主题讲座

适用于基础算法能力的大学生

二月小 22 星期六

距 CSP 2020 第一轮 还剩 237 天

距 CSP 2020 第二轮 还剩 265 天

问题跳转

例：P1001，可跳至A+B

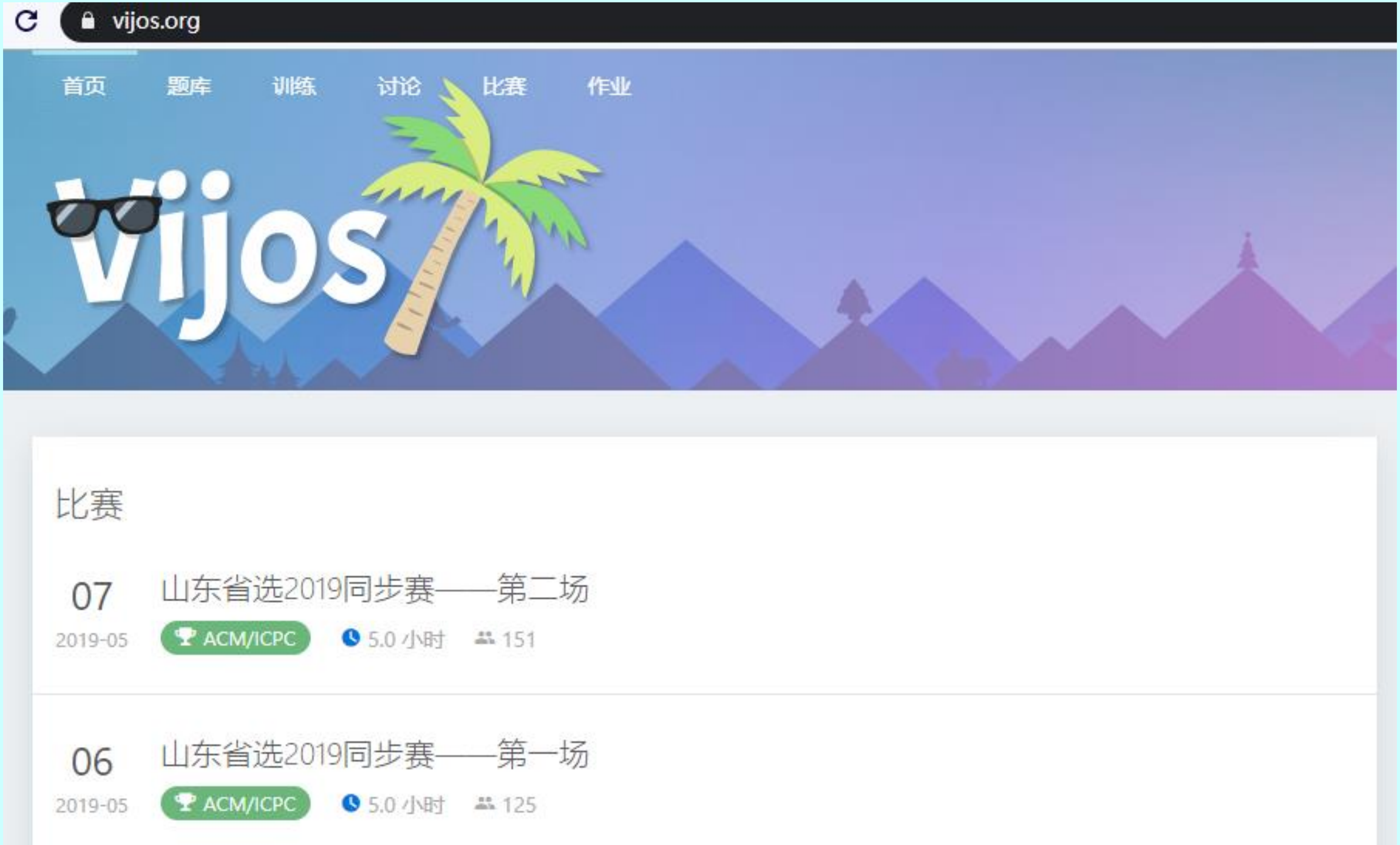
跳转 随机跳题

大牛的成長方法 STEP.1 在洛谷注册一个账号并准备好觉悟

大牛的成長方法 STEP.2 借助题库、洛谷试炼场、学校团队训练、参加比赛等提升能力，事半功倍

大牛的成長方法 STEP.3 全国的oier与你同在 交流经验，互帮互助

OJ- https://vijos.org/



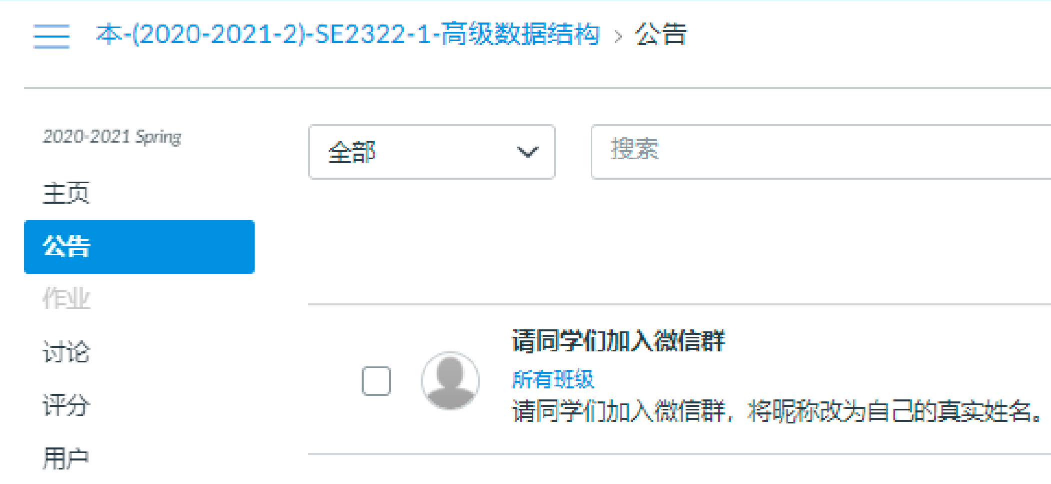
比赛

07
2019-05
山东省选2019同步赛——第二场
ACM/ICPC 5.0 小时 151

06
2019-05
山东省选2019同步赛——第一场
ACM/ICPC 5.0 小时 125

Course Web

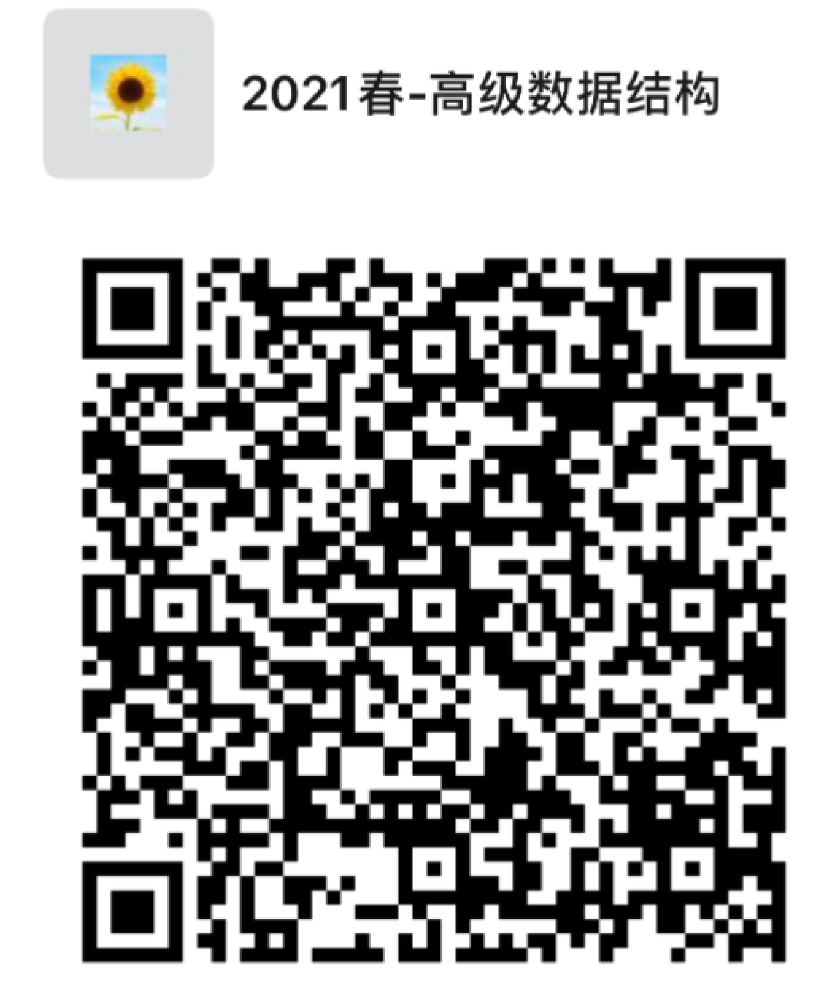
- https://oc.sjtu.edu.cn/courses/30733


本-(2020-2021-2)-SE2322-1-高级数据结构 › 公告

□ 请同学们加入微信群

所有班级

请同学们加入微信群，将昵称改为自己的真实姓名。

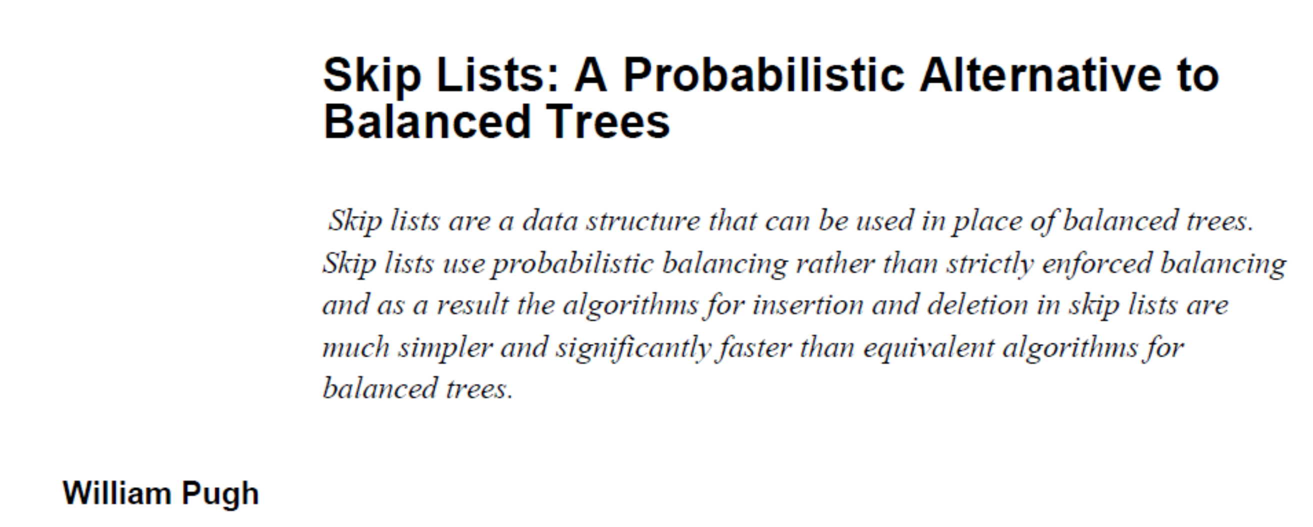

2021春-高级数据结构

Related Top Conferences

- POPL - Symposium on Principles of Programming Languages
- PLDI - SIGPLAN Conference on Programming Language Design and Implementation
- ECOOP - European Conference on Object-Oriented Programming
- CAV - Computer Aided Verification
- OOPSLA - Conference on Object-Oriented Programming Systems, Languages, and Applications
- ICSE - International Conference on Software Engineering

Paper Example

William Pugh. 1990. Skip lists: a probabilistic alternative to balanced trees. Commun. ACM 33, 6 (June 1990), 668–676.

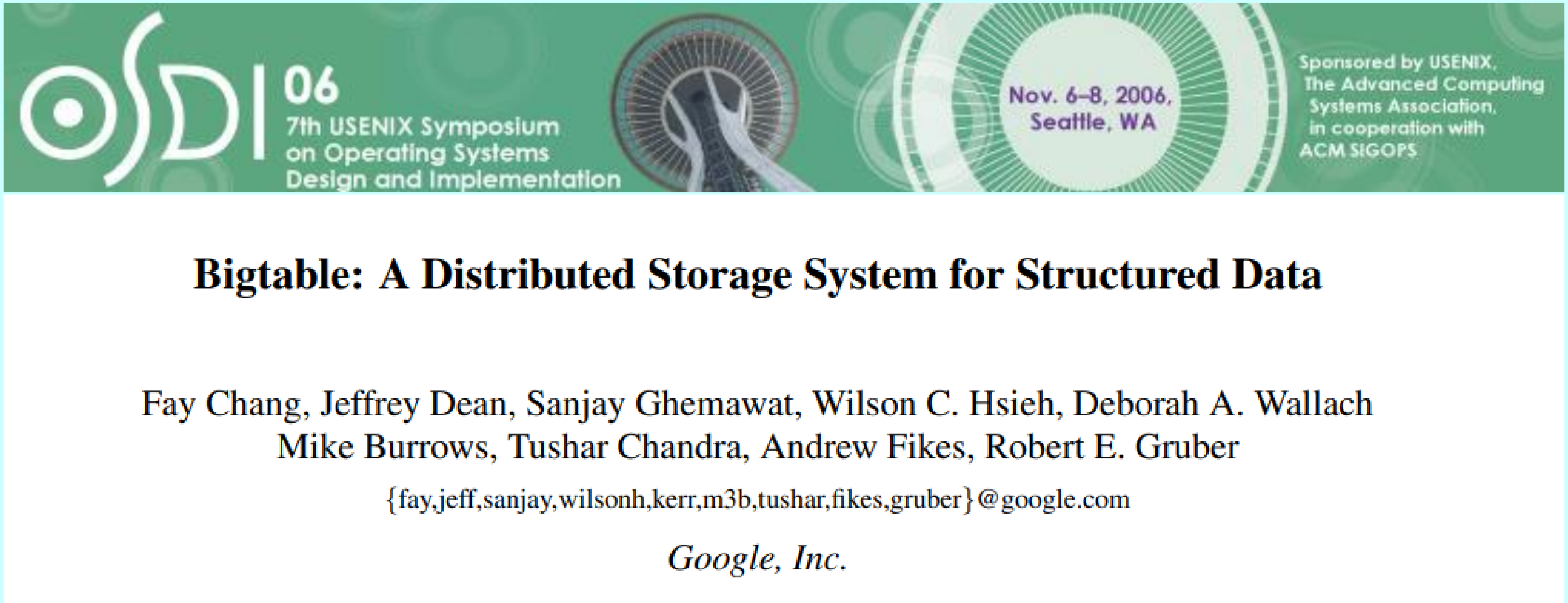


Skip Lists: A Probabilistic Alternative to Balanced Trees

Skip lists are a data structure that can be used in place of balanced trees. Skip lists use probabilistic balancing rather than strictly enforced balancing and as a result the algorithms for insertion and deletion in skip lists are much simpler and significantly faster than equivalent algorithms for balanced trees.

William Pugh

Paper Example

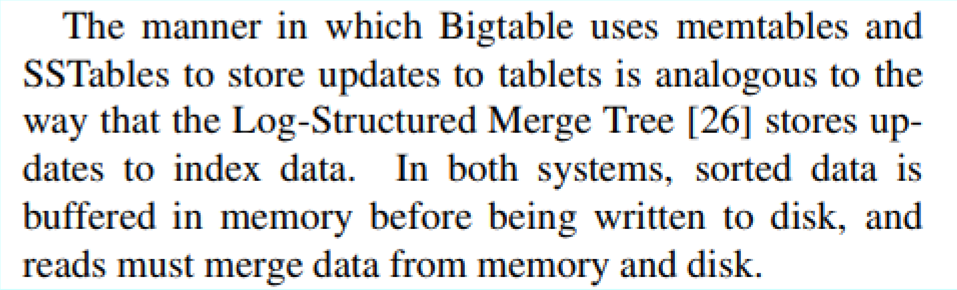

OSDI 06
7th USENIX Symposium
on Operating Systems
Design and Implementation
Nov. 6–8, 2006,
Seattle, WA
Sponsored by USENIX,
The Advanced Computing
Systems Association,
in cooperation with
ACM SIGOPS

Bigtable: A Distributed Storage System for Structured Data

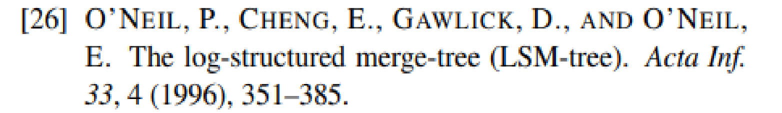
Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach
Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber

{fay,jeff,sanjay,wilsonh,kerr,m3b,tushar,fikes,gruber}@google.com

Google, Inc.



The manner in which Bigtable uses memtables and SSTables to store updates to tablets is analogous to the way that the Log-Structured Merge Tree [26] stores updates to index data. In both systems, sorted data is buffered in memory before being written to disk, and reads must merge data from memory and disk.



[26] O'NEIL, P., CHENG, E., GAWLICK, D., AND O'NEIL, E. The log-structured merge-tree (LSM-tree). *Acta Inf. 33*, 4 (1996), 351–385.

Assignments

- Homework： Every week.
- Projects: ~ Every month
- Exercise Course ： Labs

Survey

- Which kind of C++ IDE do you use?
  - A. Visual Studio
  - B. Visual Studio Code
  - C. IDE for Windows (i.e., Qt Creator)
  - D. IDE for Linux (i.e., Code::Blocks)
  - E. IDE for Mac (i.e., Xcode)
  - F. no GUI IDE (i.e., vim/emacs)

Labs

- LSM
  - Log Structured Merge Trees
- graph
  - Shortest Path Algorithm
- tree
  - balanced tree
- Mini Basic

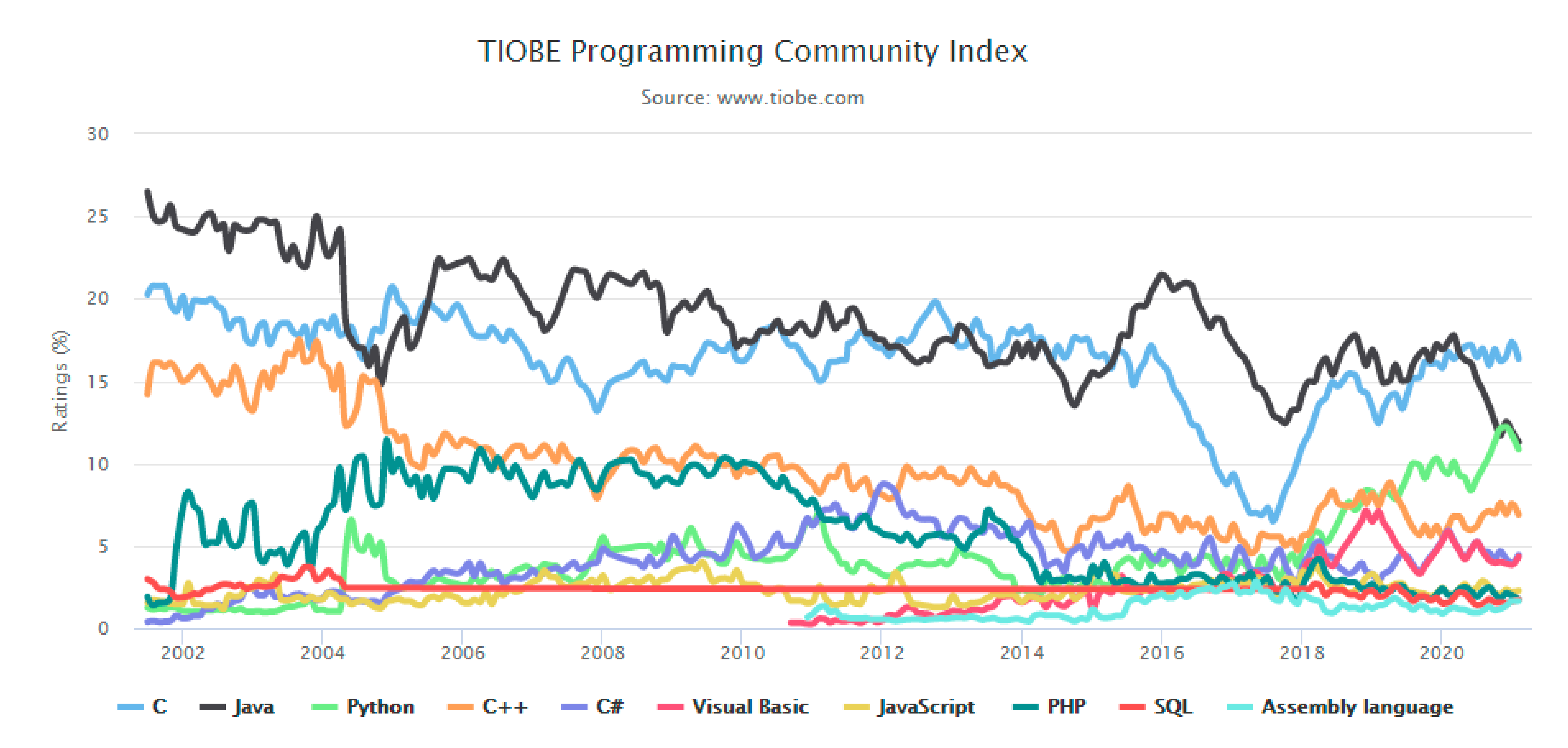
Grading

- Grades
  - Homework/Test and class participation: 20%
  - Labs: 30%
  - Final exam: 50%
- Late and re-grade policies
  - Late submissions of homework will receive partial or no credit.
  - Re-grade must be submitted within ONE WEEK
- Cheating
  - NOT tolerated!
  - Zero for the assignment and other possible repercussions

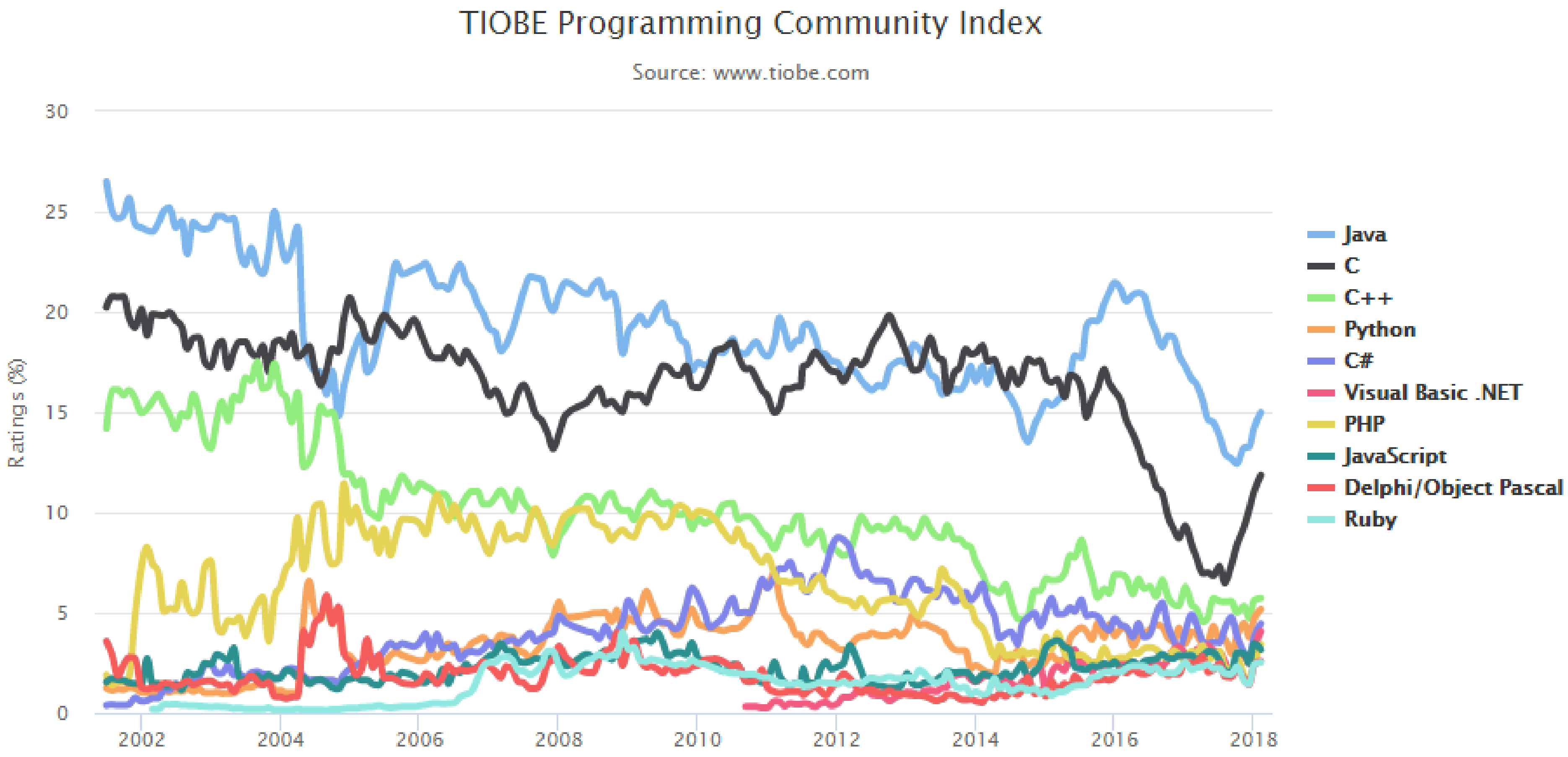
Related Courses

- MIT 6.851: Advanced Data Structures
- Stanford: **cs106 A/B/L** Standard C++ Programming Laboratory (*)
- Berkeley CS 61B: Data Structure
- CMU CS 15-121: Introduction to Data Structures
- CMU CS 15-211: Fundamental Data Structures and Algorithms
- UIUC CS 225: Data Structures and Programming Principles

TIOBE Index for 2021



Compare: TIOBE Index for 2018



So what is programming?

- Conventional definitions
  - Telling a **very** fast moron *exactly* what to do
  - A plan for solving a problem on a computer
  - Specifying the order of a program execution
    - But modern programs often involve millions of lines of code
    - And manipulation of data is central
- Definition from another domain (academia)
  - A … program is an organized and directed accumulation of resources to accomplish specific … objectives …
    - Good, but no mention of actually doing anything
- The definition we'll use
  - Specifying the structure and behavior of a program, and testing that the program performs its task correctly and with acceptable performance
    - Never forget to check that "it" works
- Software == one or more programs

Programming

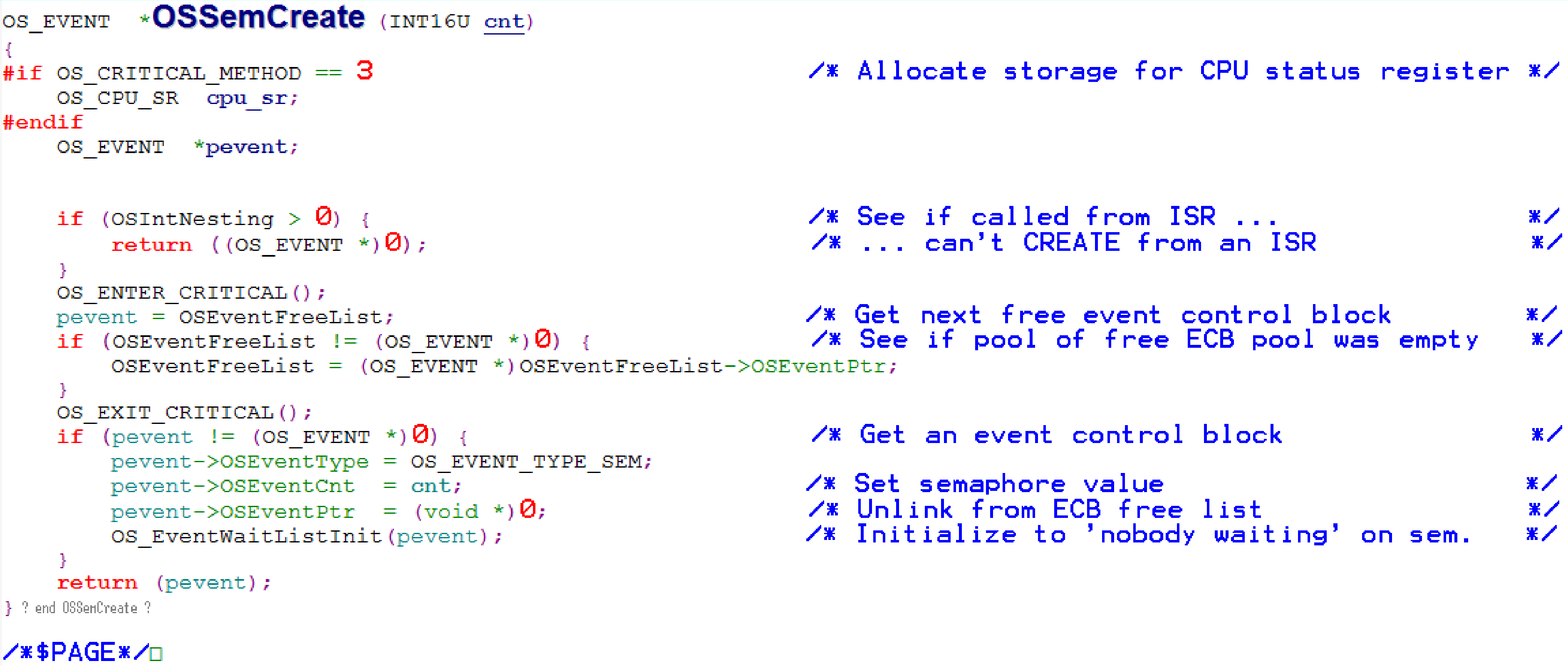
- Programming is fundamentally simple
  - Just state what the machine is to do
- So why is programming hard?
  - We want "the machine" to do complex things
    - And computers are nitpicking, unforgiving, dumb beasts
  - The world is more complex than we'd like to believe
    - So we don' t always know the implications of what we want
  - "Programming is understanding"
    - When you can program a task, you understand it
    - When you program, you spend significant time trying to understand the task you want to automate
  - Programming is part practical, part theory
    - If you are just practical, you produce non-scalable unmaintainable hacks
    - If you are just theoretical, you produce toys

现代软件是复杂工程

- Linux 内核是迄今为止最大的协同软件项目。
  - 在 2016 年，超过 450 家不同公司的 4000 多名开发者对该项目做出了贡献。
  - 该项目共有 6 个版本，每个版本都包含 12000 到 16000 项不同的更改。
  - 在 2016 年底，Linux 内核的规模刚好超过 56000 个文件，其中包括 2200 万行代码、编译脚本和文档（内核版本 4.9）
- 虽然 Linux 内核包含其支持的所有不同芯片架构和硬件驱动程序的代码，但各个系统仅运行一小部分代码库。
  - 一台普通的笔记本电脑需要使用来自 5000 个文件的大约 200 万行内核代码才能正常运行；
  - 而 Pixel 手机需要使用来自 6000 个文件的 320 万行内核代码才能正常运行（因为 SoC 的复杂性有所增加）。

Source: https://source.android.google.cn/devices/architecture/kernel?hl=zh-cn

Maintainability

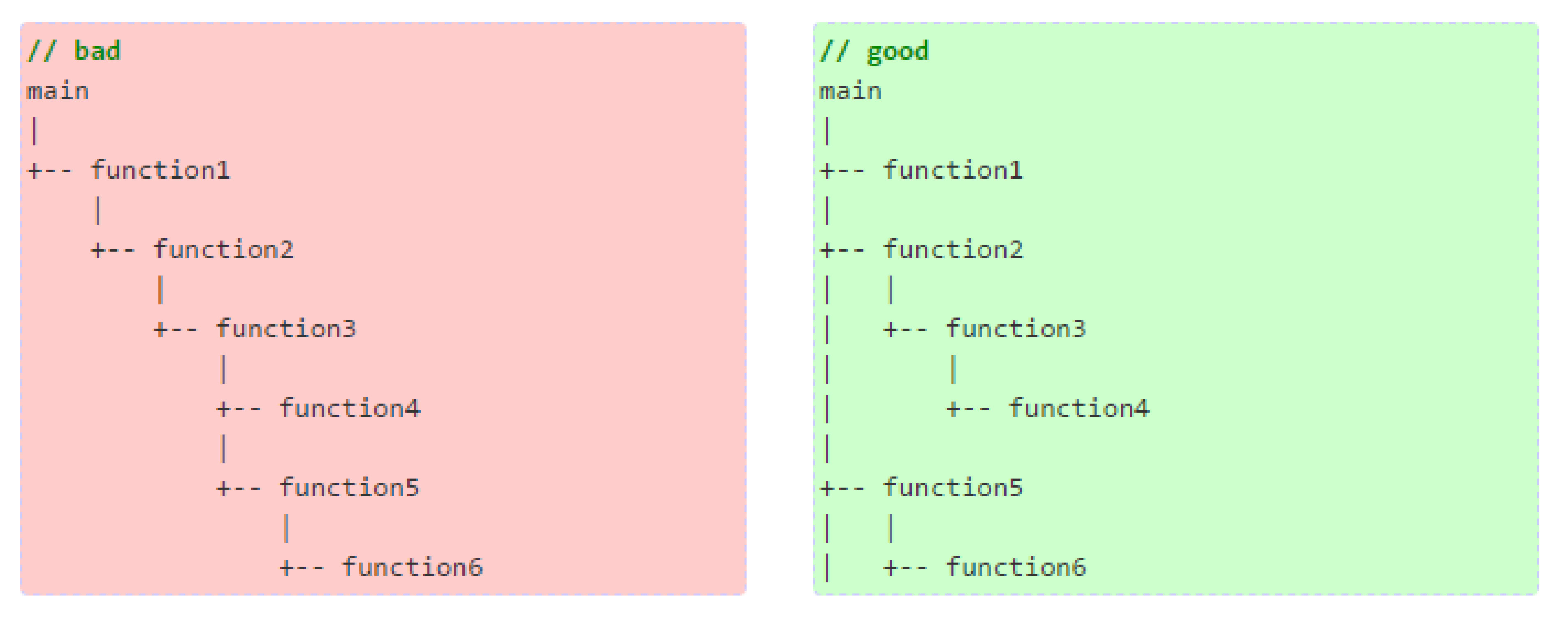


```
OS_EVENT  *OSSemCreate (INT16U cnt)
{
#if OS_CRITICAL_METHOD == 3                          /* Allocate storage for CPU status register */
    OS_CPU_SR  cpu_sr;
#endif
    OS_EVENT  *pevent;


    if (OSIntNesting > 0) {                          /* See if called from ISR ...               */
        return ((OS_EVENT *)0);                      /* ... can't CREATE from an ISR             */
    }
    OS_ENTER_CRITICAL();
    pevent = OSEventFreeList;                        /* Get next free event control block        */
    if (OSEventFreeList != (OS_EVENT *)0) {          /* See if pool of free ECB pool was empty   */
        OSEventFreeList = (OS_EVENT *)OSEventFreeList->OSEventPtr;
    }
    OS_EXIT_CRITICAL();
    if (pevent != (OS_EVENT *)0) {                   /* Get an event control block               */
        pevent->OSEventType = OS_EVENT_TYPE_SEM;
        pevent->OSEventCnt  = cnt;                   /* Set semaphore value                      */
        pevent->OSEventPtr  = (void *)0;             /* Unlink from ECB free list                */
        OS_EventWaitListInit(pevent);                /* Initialize to 'nobody waiting' on sem.   */
    }
    return (pevent);
} ? end OSSemCreate ?

/*$PAGE*/□
```

C++ Style Guide

- Google C++ Style Guide
- JPL Coding Standard
- Stanford Style:
  - [http://web.stanford.edu/class/cs106b/resources/style-guide/]
- PPP Style:
  - [http://isocpp.github.io/CppCoreGuidelines/CppCoreGuidelines]
- SJTU Style
  - [https://github.com/sjtu-se117/CppGuidelines/blob/master/CppGuidelines.md]

Example



```
// bad
main
|
+-- function1
    |
    +-- function2
        |
        +-- function3
            |
            +-- function4
            |
            +-- function5
                |
                +-- function6
```

```
// good
main
|
+-- function1
|
+-- function2
|   |
|   +-- function3
|       |
|       +-- function4
|
+-- function5
|   |
|   +-- function6
```

http://web.stanford.edu/class/cs106b/resources/style-guide/

A good program

- Correctness
- Maintainability
- Performance
- Security
- Scalability
- Availability
- Power-efficiency

```
<?php
/**
***   ┏┓      ┏┓
*   ┏┛┻━━━━━┛┻┓
*   ┃             ┃
*   ┃      ━      ┃
*   ┃  ┳┛    ┗┳  ┃
*   ┃             ┃
*   ┃ ...  ┻  ... ┃
*   ┃             ┃
*   ┗━┓      ┏━┛
*****  ┃      ┃  Code is far away from bug with the animal protecting.
*****  ┃      ┃  神兽护佑,代码无bug.
*****  ┃      ┗━━━┓
*****  ┃            ┣┓
*****  ┃            ┏┛
*****  ┗┓┓┏━━┳┓┏┛
*******  ┃┫┫    ┃┫┫
*******  ┗┻┛    ┗┻┛
```

Security: Example insider attack

- Hidden trap door in Linux, Nov 2003
  - Allows attacker to take over a computer
  - Practically undetectable change
  - Uncovered by anomaly in CVS usage
- Inserted line in wait4()

```
if ((options == (__WCLONE|__WALL)) && (current->uid = 0))
            retval = -EINVAL;
```

  - Looks like a standard error check
  - Anyone see the problem?

See: http://lwn.net/Articles/57135/

A good program

- Correctness
- Maintainability
- Performance
- Security
- Scalability
- Availability
- Power-efficiency

```
<?php
/**
***    ┏┓      ┏┓
*    ┏┛┻━━━━━━━┛┻┓
*    ┃           ┃
*    ┃     ━     ┃
*    ┃  ┳┛   ┗┳  ┃
*    ┃           ┃
*    ┃  ...  ┻  ...  ┃
*    ┃           ┃
*    ┗━┓       ┏━┛
*****  ┃       ┃Code is far away from bug with the animal protecting.
*****  ┃       ┃神兽护佑,代码无bug.
*****  ┃       ┗━━━┓
*****  ┃           ┣┓
*****  ┃           ┏┛
*****  ┗┓┓┏━┳┓┏┛
*******  ┃┫┫ ┃┫┫
*******  ┗┻┛ ┗┻┛
```

Python vs. C++

Guido van Rossum

Bjarne Stroustrup

- Interpreted
- Very high level language
- Writing code is quick and easy
- *Python* code runs more slowly, but call precompiled C/C++ Libraries
- Dynamic type system

- Compile and Link
- Low-level language (but standardized higher-level libraries available)
- Writing code takes longer
- Code runs very fast
- Static type system

test.py

```python
def f(n):
    if n == 1:
        return 1
    else:
        #print n
        return n*f(n-1)

print f(100)
```

```
python test.py
```

test.cpp

```cpp
#include <iostream>
#include<gmpxx.h>
using namespace std;

mpz_class  f(int n){
    if (n ==  1)
        return 1;
    else
        return n*f(n-1);
}

int main()
{
    cout << f(100) << endl;
}
```

```
g++ test.cpp -o test -lgmpxx -lgmp
```

Next

- Chapter 9: Dictionary (数据结构（C++语言版）第三版)