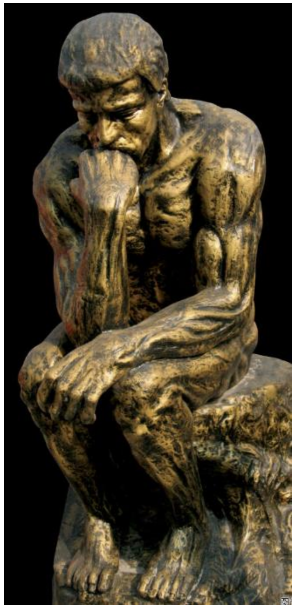
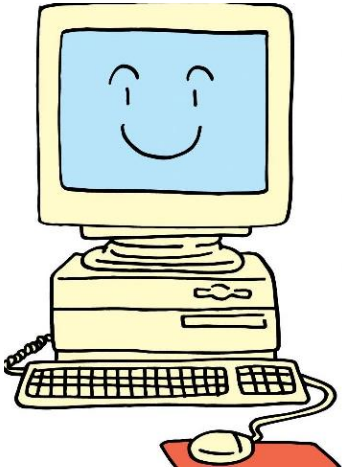
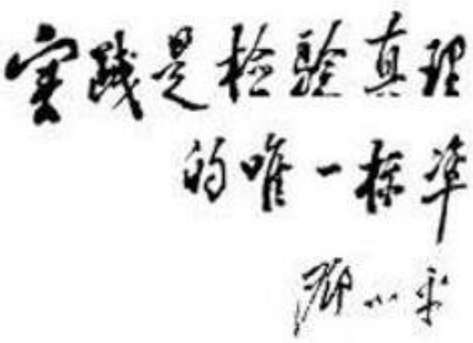
计算物理 第二部分

第1讲







实践是检验真理
的唯一标准
邓小平

李强　北京大学物理学院中楼411

qliphy0@pku.edu.cn，　15210033542

http://www.phy.pku.edu.cn/~qiangli/CP2017.html

1	9.9	9.10	9.11	9.12	9.13 (中秋)	Sat./Sun.
2	9.16	9.17	9.18	9.19	9.20	
3	9.23	9.24	9.25	9.26	9.27	
4	9.30	10.1	10.2	10.3	10.4	国庆
5	10.7	10.8	10.9	10.10	10.11	
6	10.14	10.15	10.16	10.17	10.18	
7	10.21	10.22	10.23	10.24	10.25	10.23-27 CLHCP
8	10.28	10.29	10.30	10.31	11.1	
9	11.4	11.5	11.6	11.7	11.8	
10	11.11	11.12	11.13	11.14	11.15	
11	11.18	11.19	11.20	11.21	11.22	
12	11.25	11.26	11.27	11.28	11.29(习题课)	
13	12.2	12.3	12.4	12.5	12.6	
14	12.9	12.10	12.11	12.12	12.13	CMSDAS 12.9-13
15	12.16	12.17 (习题课)	12.18	12.19	12.20	CMSWeek 12.16-20
16	12.23	12.24	12.25	12.26	12.27 (待定)	
17	12.30	12.31	1.1	1.2	1.3	
18	1.6	1.7	1.8	1.9	1.10	

计算物理
理教406
周一3-4
周二5-6

习题课
二教413
周五7-8

期末
12.31下午

12.27 下午3:10-5:00点 补课

12.31 下午2-4点一教201

12月2日　当天晚12点　交 后半程第一次作业
12月16日 当天晚12点　交 后半程第二次作业
12月26日 当天晚12点　交 后半程第三次作业

12.27 下午3:10-5点 补课 二教407
12.31 下午2-4点一教201 期末

1. “计算 ” 介绍
2. 常微分方程
3. 偏微分方程
4. 快速傅里叶变换
5. 随机数
6. 蒙特卡洛方法
7. 有限元方法
8. 机器学习初步

参考材料：

科学出版社， 计算物理学， 刘金远等
世界图书出版公司， An Introduction to Computational Physics， Tao Pang
科学出版社， 计算物理学， 马文淦
世界图书出版公司， A First Course in Computational Physics and Object-Oriented Programming with C++， David Yevick

1. “计算 ” 介绍

Linux操作系统

Fortran、**C++**、**Python**简介

CERN ROOT数据处理

作图软件

并行计算 **MPI**

版本维护 **github**

2. 常微分方程

概念介绍

初值问题： 欧拉法， Runge-Kutta法

微分方程组： 电磁场中电子运动

高阶微分方程

- 薛定谔方程
- 混沌，Lorenz吸引子

刚性微分方程， Gear方法

边值问题：

1D泊松方程 差分法(三对角矩阵)，打靶法 ,
混合边值问题

3. 偏微分方程A

概念介绍

1阶对流方程: 差分法 (不同格式), vonNeumann stability

抛物形方程(1阶扩散方程):

线上法， stability, FTCS/CN差分

非线性PDE: Burgers, KdV, …

4. 偏微分方程B

2阶线性PDE: Elliptic, Parabolic, Hyperbolic

椭圆方程(泊松方程):

2D/3D

spectral method 傅里叶变换

抛物形方程(2D扩散方程):

5-6. 随机数、蒙特卡洛方法

介绍：

布冯实验，概率密度
期望值，方差，大数法则

随机数 实现, 随机统计检验

取样： 直接，变换，取舍， Importance Sampling
积分： VEGAS

随机行走、马尔可夫
事例产生

7-8. 其他

有限元方法

机器学习在高能物理中的应用

MVA, BDT, Neural Network

1. “计算 ” 介绍，常微分方程

Linux操作系统

Ubuntu

Fortran，C++简介

Euler法 示例

CERN ROOT数据处理软件

Histogram, bin, error-bar

Pi计算示例

GNUPLOT作图

动态图示例

并行计算

版本维护

Linux之前

1983 图灵奖：
Ken Thompson， Dennis M. Ritchie
for their development of generic operating systems theory and specifically for the implementation of the UNIX operating system.

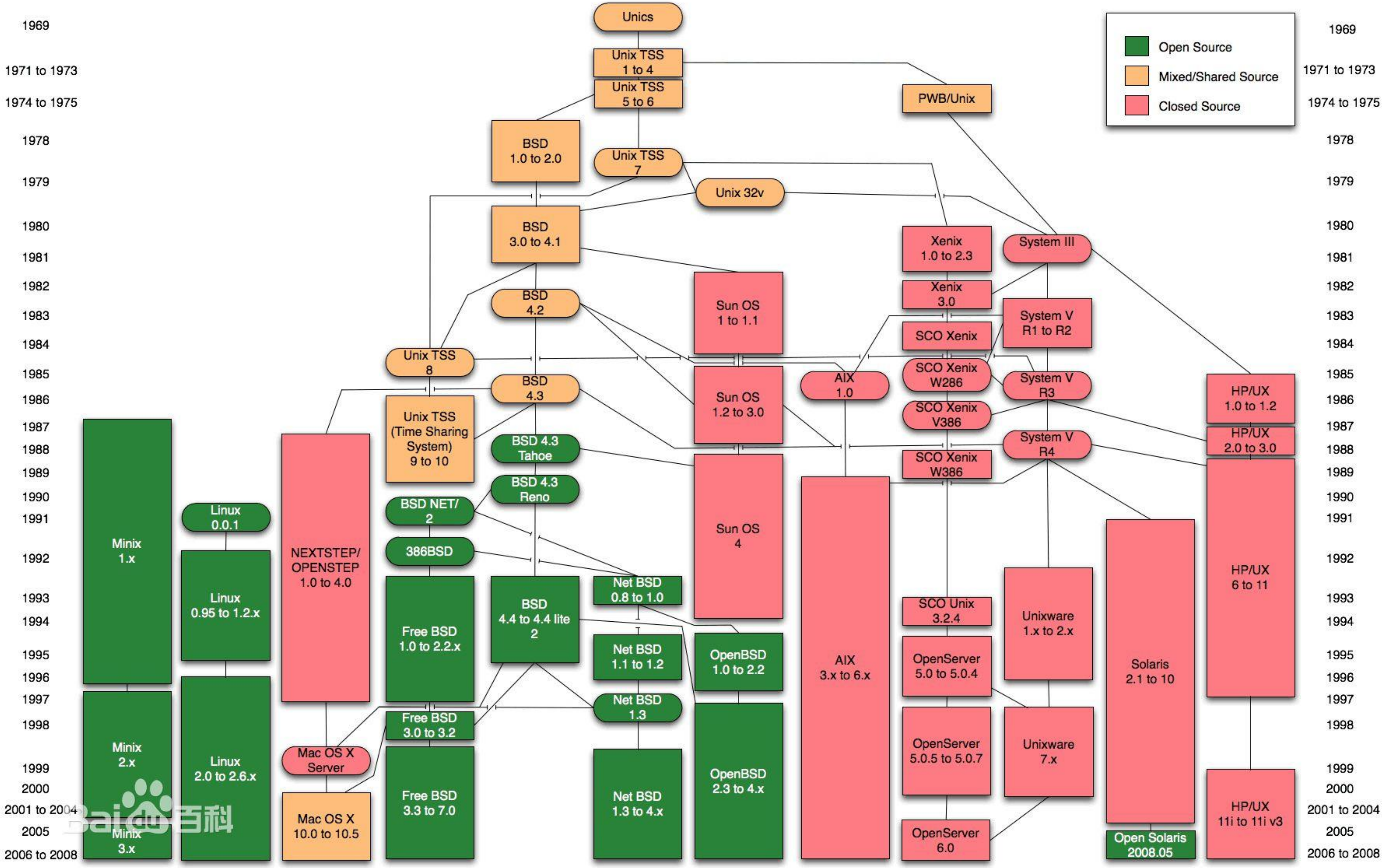
- In 80’s, Microsoft’s DOS (disk operating system) was the dominated OS for PC
- Apple MAC was better, but expensive
- UNIX was much better, but much, much more expensive. Only for minicomputer for commercial applications
- **People was looking for a UNIX based system, which is cheaper and can run on PC**
- Both DOS, MAC and UNIX were proprietary, i.e., the source code of their kernel is protected
- No modification is possible without paying high license fees

GNU Linux

GNU是“GNU is Not Unix”的递归缩写。

- Established in 1984 by Richard Stallman, who believes that software should be free from restrictions against copying or modification in order to make better and efficient computer programs
- GNU通用公共许可证（GNU General Public License，GPL）。即“反版权”（或称Copyleft）概念。

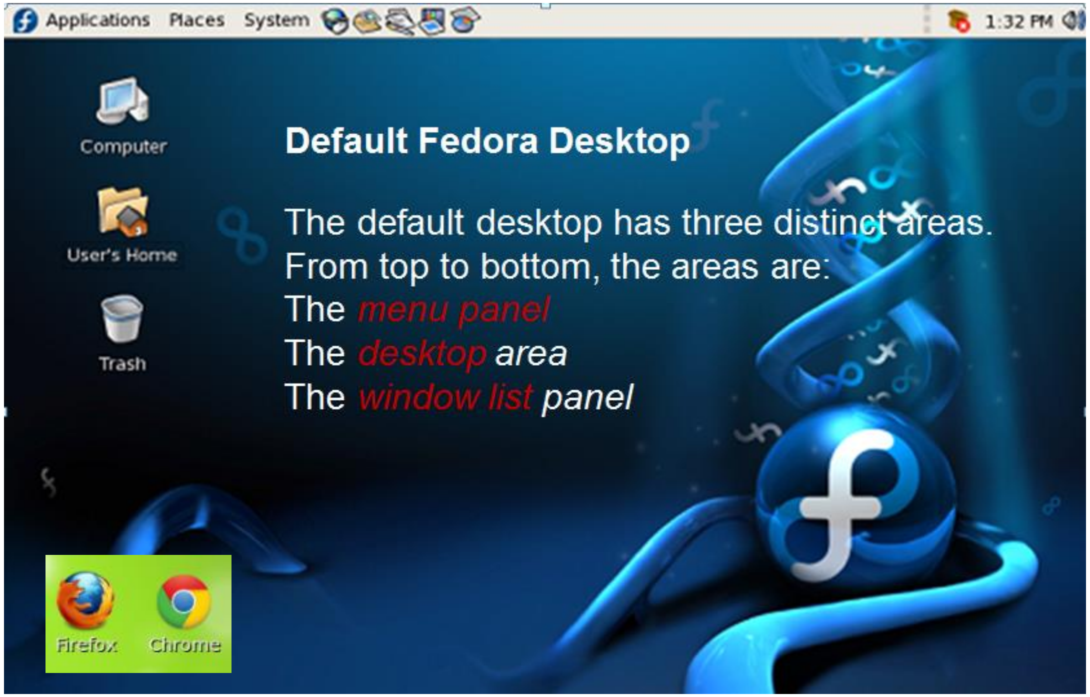
1991年Linus Torvalds编写出了与UNIX兼容的Linux操作系统内核并在GPL条款下发布。Linux之后在网上广泛流传，许多程序员参与了开发与修改。1992年Linux与其他GNU软件结合，完全自由的操作系统正式诞生。该操作系统往往被称为“GNU/Linux”或简称Linux。



🟩 Open Source
🟧 Mixed/Shared Source
🟥 Closed Source
1969
1971 to 1973
1974 to 1975
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001 to 2004
2005
2006 to 2008
Unics
Unix TSS 1 to 4
Unix TSS 5 to 6
PWB/Unix
BSD 1.0 to 2.0
Unix TSS 7
Unix 32v
BSD 3.0 to 4.1
Xenix 1.0 to 2.3
System III
Sun OS 1 to 1.1
Xenix 3.0
BSD 4.2
System V R1 to R2
SCO Xenix
Unix TSS 8
SCO Xenix W286
AIX 1.0
System V R3
HP/UX 1.0 to 1.2
BSD 4.3
Sun OS 1.2 to 3.0
Unix TSS (Time Sharing System) 9 to 10
SCO Xenix V386
BSD 4.3 Tahoe
System V R4
HP/UX 2.0 to 3.0
SCO Xenix W386
BSD 4.3 Reno
Minix 1.x
NEXTSTEP/ OPENSTEP 1.0 to 4.0
Linux 0.0.1
BSD NET/ 2
386BSD
Sun OS 4
Net BSD 0.8 to 1.0
Linux 0.95 to 1.2.x
Free BSD 1.0 to 2.2.x
BSD 4.4 to 4.4 lite 2
Net BSD 1.1 to 1.2
OpenBSD 1.0 to 2.2
AIX 3.x to 6.x
SCO Unix 3.2.4
OpenServer 5.0 to 5.0.4
Unixware 1.x to 2.x
Solaris 2.1 to 10
HP/UX 6 to 11
Net BSD 1.3
Free BSD 3.0 to 3.2
Minix 2.x
Linux 2.0 to 2.6.x
Mac OS X Server
OpenServer 5.0.5 to 5.0.7
Unixware 7.x
Free BSD 3.3 to 7.0
Net BSD 1.3 to 4.x
OpenBSD 2.3 to 4.x
HP/UX 11i to 11i v3
Mac OS X 10.0 to 10.5
Minix 3.x
OpenServer 6.0
Open Solaris 2008.05

Linux系统

Arch Linux, CentOS, Debian, Fedora, Gentoo Linux, Linux Mint, Mageia, openSUSE and Ubuntu, together with commercial distributions such as Red Hat Enterprise Linux and SUSE Linux Enterprise Server.



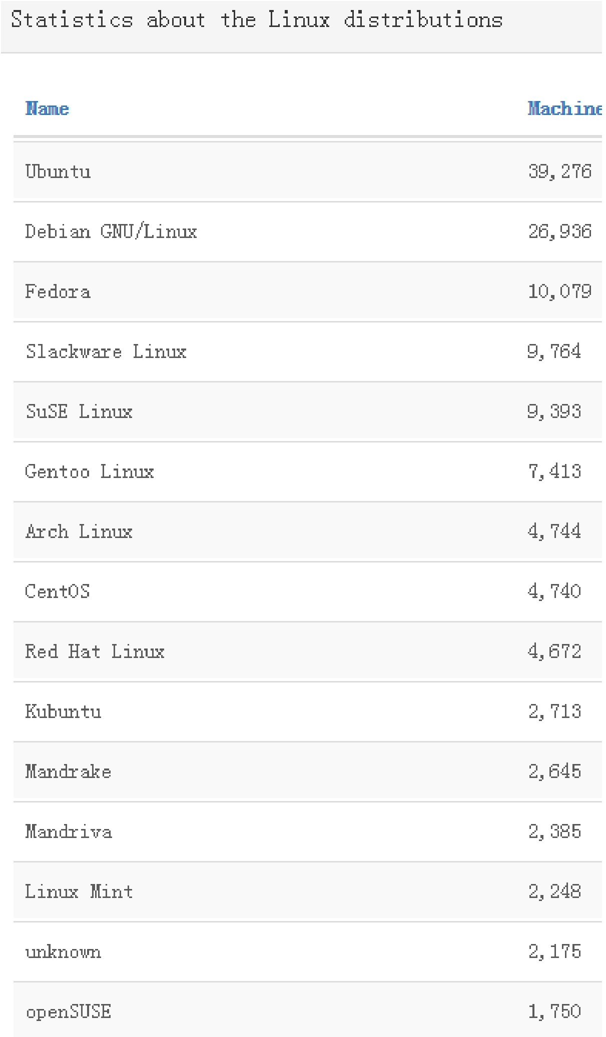
Default Fedora Desktop

The default desktop has three distinct areas. From top to bottom, the areas are:

The *menu panel*

The *desktop area*

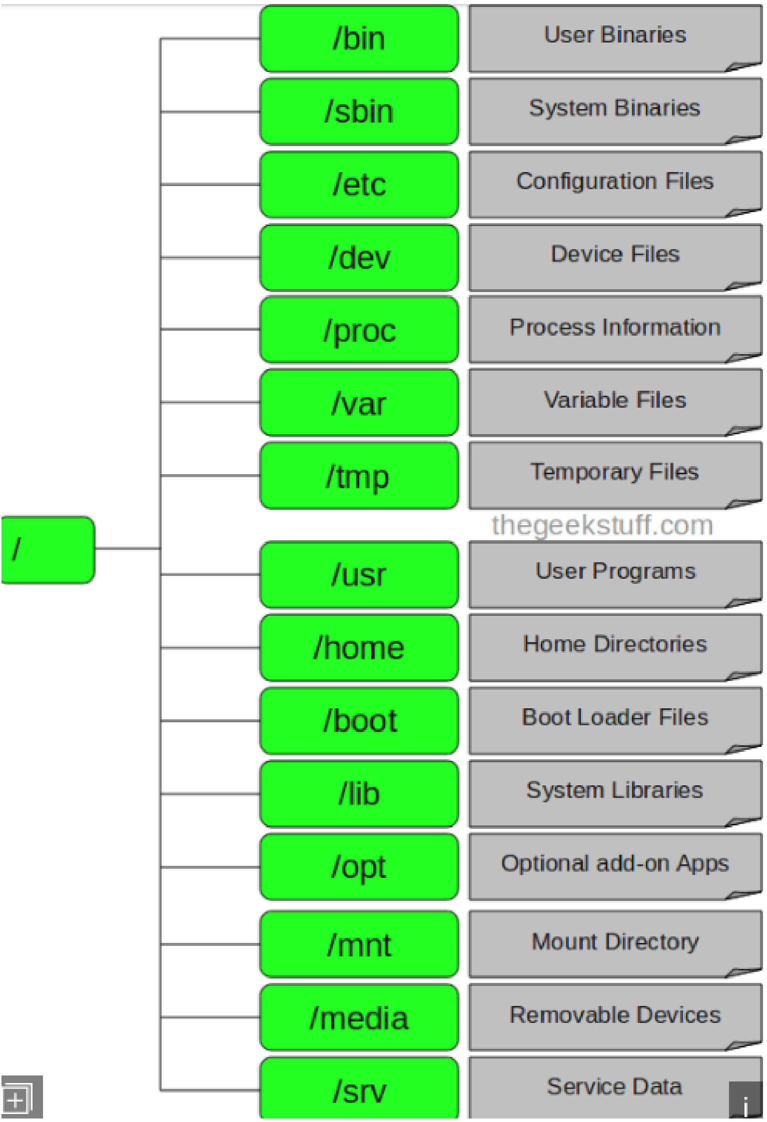
The *window list panel*



Statistics about the Linux distributions

Name	Machine
Ubuntu	39,276
Debian GNU/Linux	26,936
Fedora	10,079
Slackware Linux	9,764
SuSE Linux	9,393
Gentoo Linux	7,413
Arch Linux	4,744
CentOS	4,740
Red Hat Linux	4,672
Kubuntu	2,713
Mandrake	2,645
Mandriva	2,385
Linux Mint	2,248
unknown	2,175
openSUSE	1,750

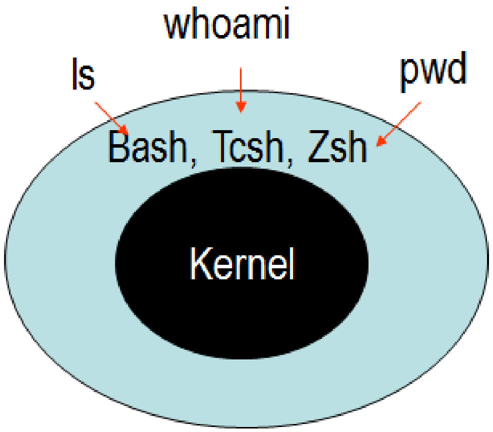
Linux Shell以及文件系统



/

/bin	User Binaries
/sbin	System Binaries
/etc	Configuration Files
/dev	Device Files
/proc	Process Information
/var	Variable Files
/tmp	Temporary Files
/usr	User Programs
/home	Home Directories
/boot	Boot Loader Files
/lib	System Libraries
/opt	Optional add-on Apps
/mnt	Mount Directory
/media	Removable Devices
/srv	Service Data

thegeekstuff.com



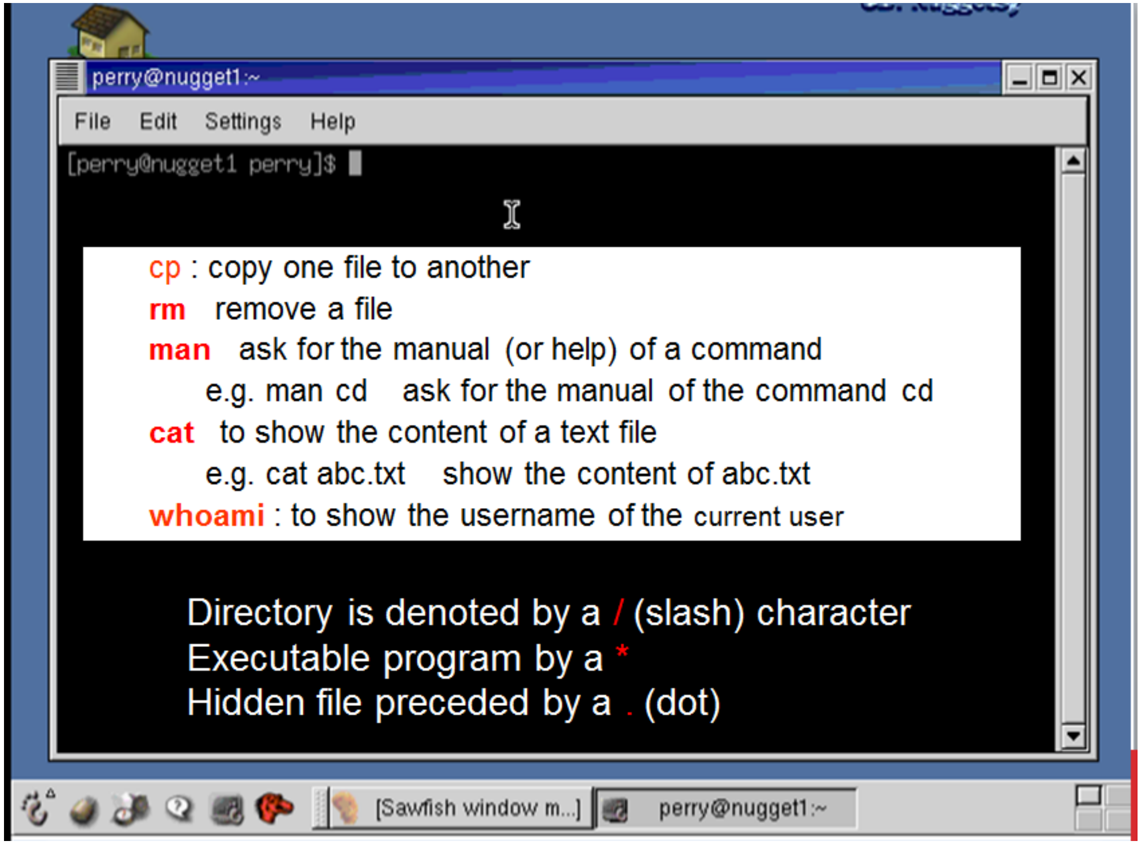
whoami

ls

pwd

Bash, Tcsh, Zsh

Kernel



```
[perry@nugget1 perry]$
```

cp : copy one file to another

rm remove a file

man ask for the manual (or help) of a command

e.g. man cd ask for the manual of the command cd

cat to show the content of a text file

e.g. cat abc.txt show the content of abc.txt

whoami : to show the username of the current user

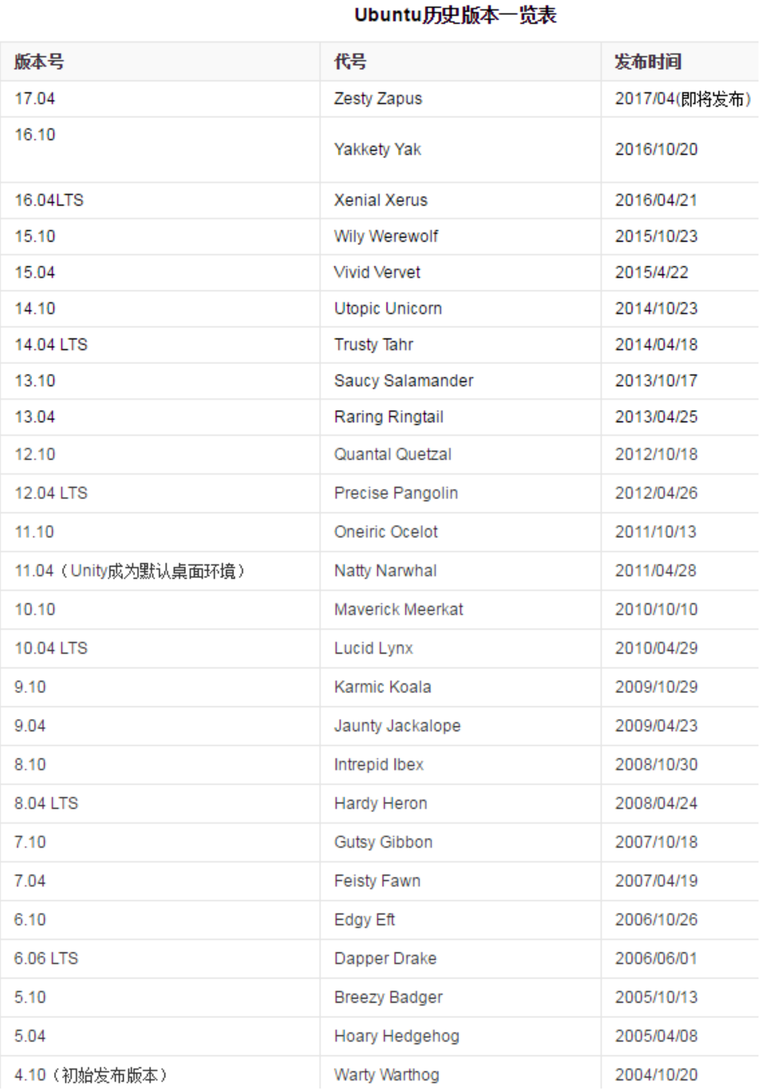
Directory is denoted by a / (slash) character

Executable program by a *

Hidden file preceded by a . (dot)

Ubuntu

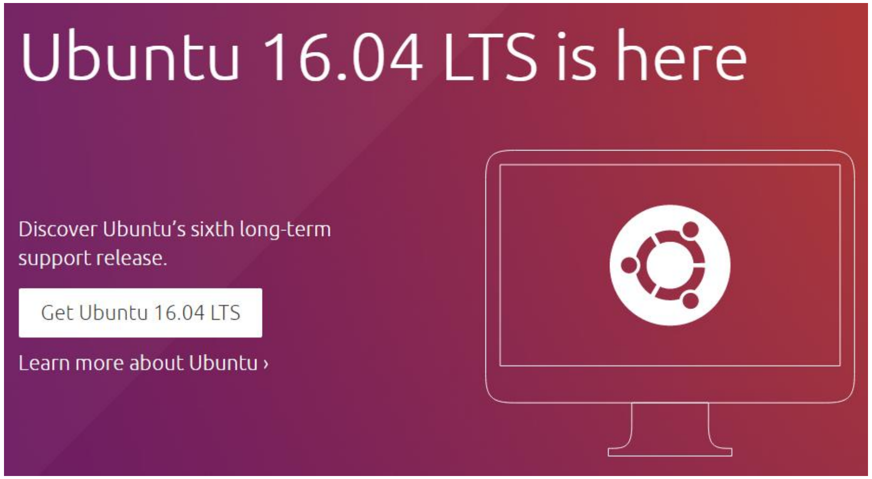
Ubuntu是一个以桌面应用为主的开源GNU/Linux操作系统，Ubuntu 是基于Debian GNU/Linux，支持x86、amd64（即x64）和ppc架构，由全球化的专业开发团队（Canonical Ltd）打造的



Ubuntu历史版本一览表

版本号	代号	发布时间
17.04	Zesty Zapus	2017/04(即将发布)
16.10	Yakkety Yak	2016/10/20
16.04LTS	Xenial Xerus	2016/04/21
15.10	Wily Werewolf	2015/10/23
15.04	Vivid Vervet	2015/4/22
14.10	Utopic Unicorn	2014/10/23
14.04 LTS	Trusty Tahr	2014/04/18
13.10	Saucy Salamander	2013/10/17
13.04	Raring Ringtail	2013/04/25
12.10	Quantal Quetzal	2012/10/18
12.04 LTS	Precise Pangolin	2012/04/26
11.10	Oneiric Ocelot	2011/10/13
11.04（Unity成为默认桌面环境）	Natty Narwhal	2011/04/28
10.10	Maverick Meerkat	2010/10/10
10.04 LTS	Lucid Lynx	2010/04/29
9.10	Karmic Koala	2009/10/29
9.04	Jaunty Jackalope	2009/04/23
8.10	Intrepid Ibex	2008/10/30
8.04 LTS	Hardy Heron	2008/04/24
7.10	Gutsy Gibbon	2007/10/18
7.04	Feisty Fawn	2007/04/19
6.10	Edgy Eft	2006/10/26
6.06 LTS	Dapper Drake	2006/06/01
5.10	Breezy Badger	2005/10/13
5.04	Hoary Hedgehog	2005/04/08
4.10（初始发布版本）	Warty Warthog	2004/10/20

Arguably *the* most user-friendly version of Linux.
Huge repository of (free) software available - by far the most of any Linux distro



Ubuntu 16.04 LTS is here

Discover Ubuntu's sixth long-term support release.

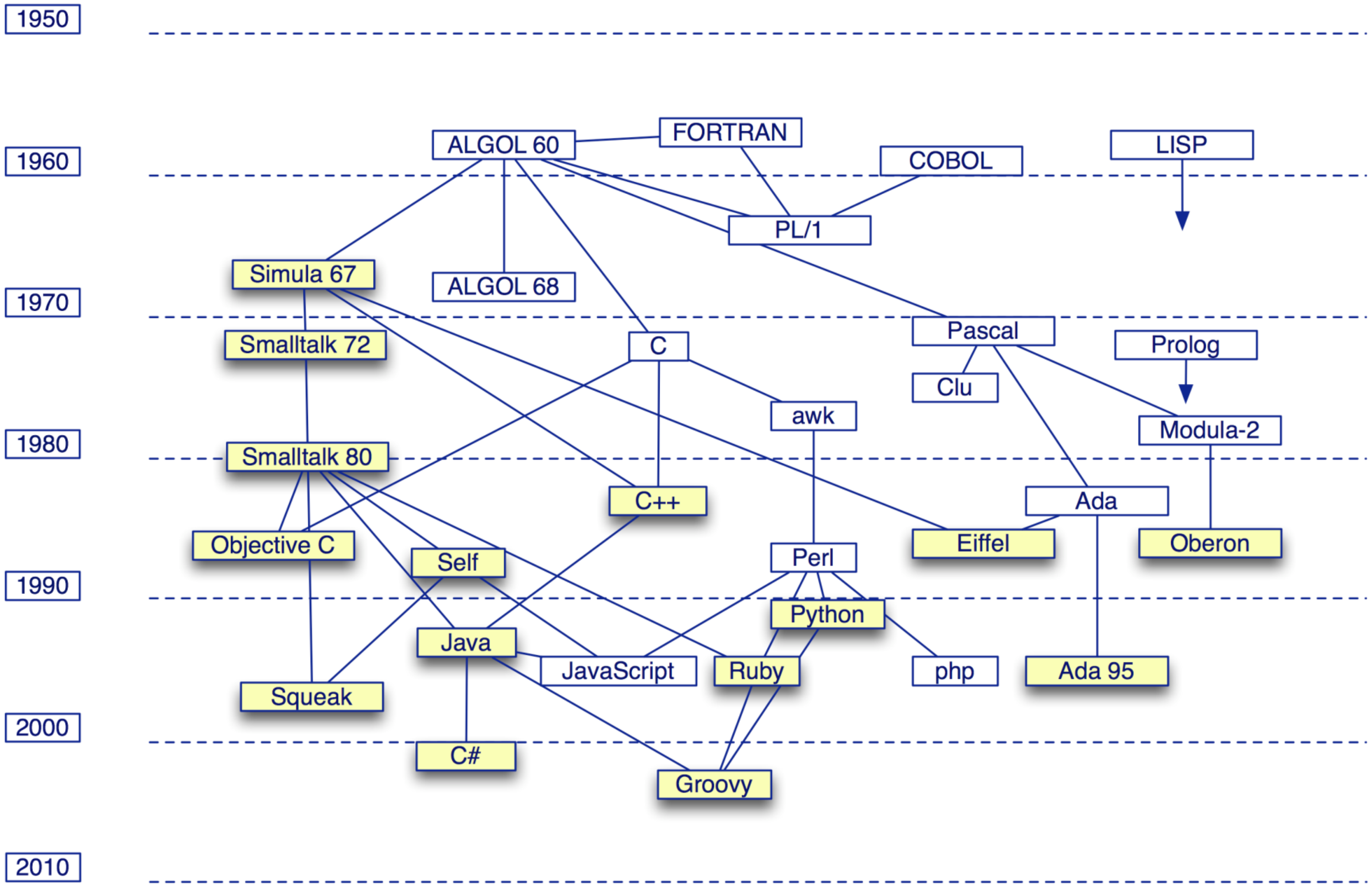
可与Windows双系统安装;可硬盘，U盘，光盘安装; 安装过程很自动

安装编译环境

```
sudo apt-get install build-essential
      gcc, g++,....
sudo apt-get install gfortran
```

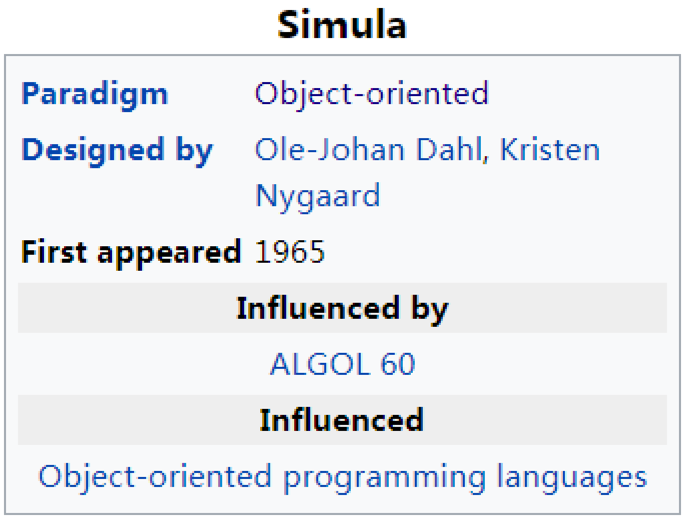
C++简介

Object-oriented language genealogy



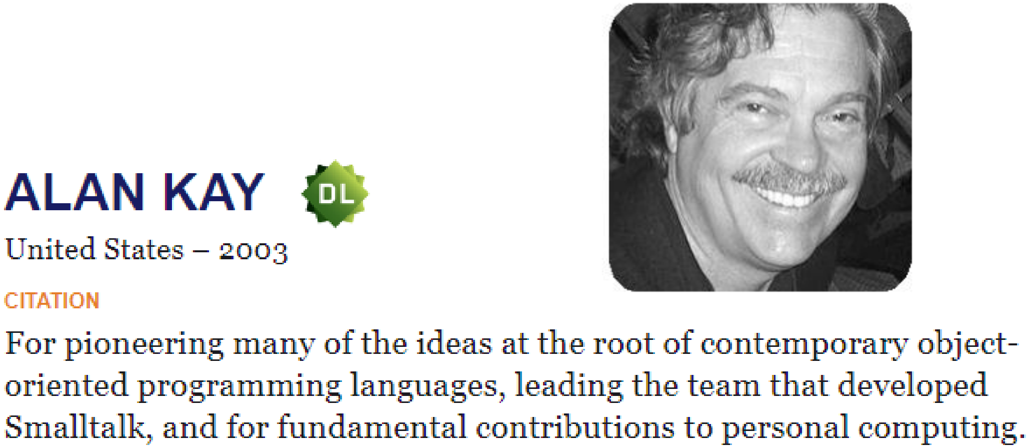
1979年， Bjarne Stroustrup到了Bell实验室，开始从事将C改良为带类的C（*C with classes*）的工作。1983年该语言被正式命名为C++。

Simula is considered the first object-oriented programming language. As its name suggests, Simula was designed for doing simulations, and the needs of that domain provided the framework for many of the features of object-oriented languages today.



Simula

Paradigm	Object-oriented
Designed by	Ole-Johan Dahl, Kristen Nygaard
First appeared	1965
Influenced by	
ALGOL 60	
Influenced	
Object-oriented programming languages	

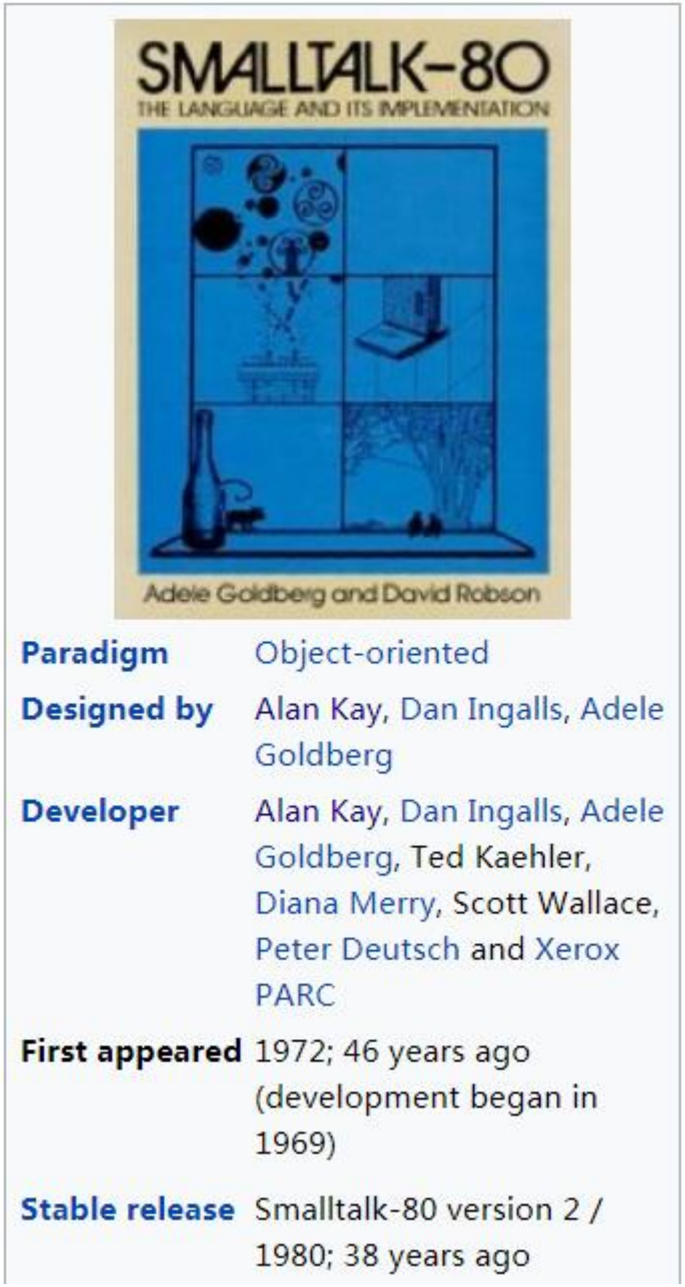


ALAN KAY

United States – 2003

CITATION

For pioneering many of the ideas at the root of contemporary object-oriented programming languages, leading the team that developed Smalltalk, and for fundamental contributions to personal computing.



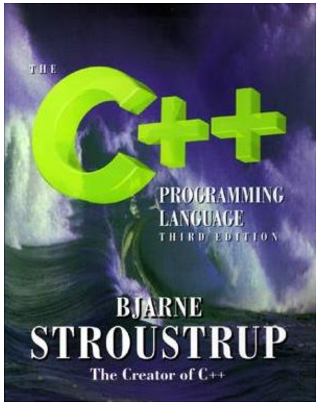
SMALLTALK-80
THE LANGUAGE AND ITS IMPLEMENTATION
Adele Goldberg and David Robson

Paradigm	Object-oriented
Designed by	Alan Kay, Dan Ingalls, Adele Goldberg
Developer	Alan Kay, Dan Ingalls, Adele Goldberg, Ted Kaehler, Diana Merry, Scott Wallace, Peter Deutsch and Xerox PARC
First appeared	1972; 46 years ago (development began in 1969)
Stable release	Smalltalk-80 version 2 / 1980; 38 years ago

C++简介

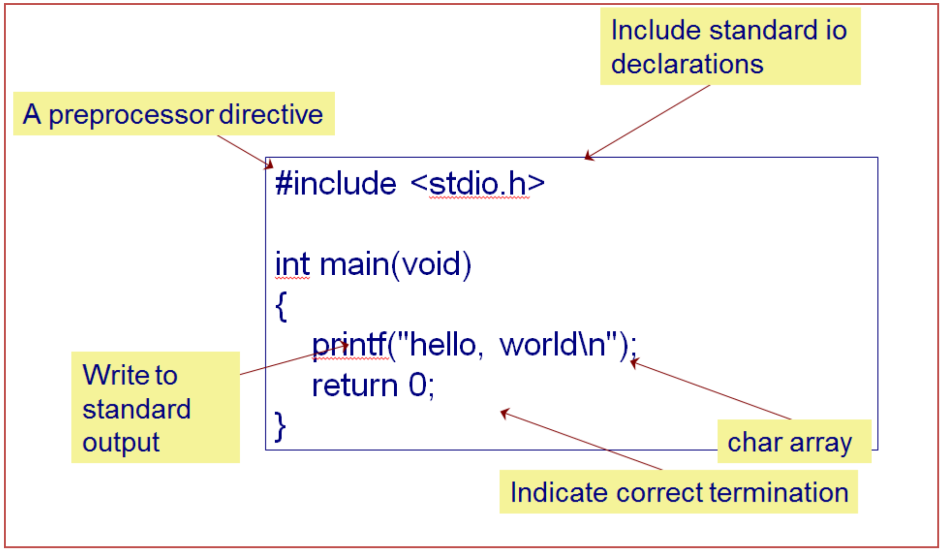
A *“better C”* that supports:

- Systems programming
- Object-oriented programming (*classes* & *inheritance*)
- Programming-in-the-large (*namespaces*, *exceptions*)
- Generic programming (*templates*)
- Reuse (large class & template libraries)



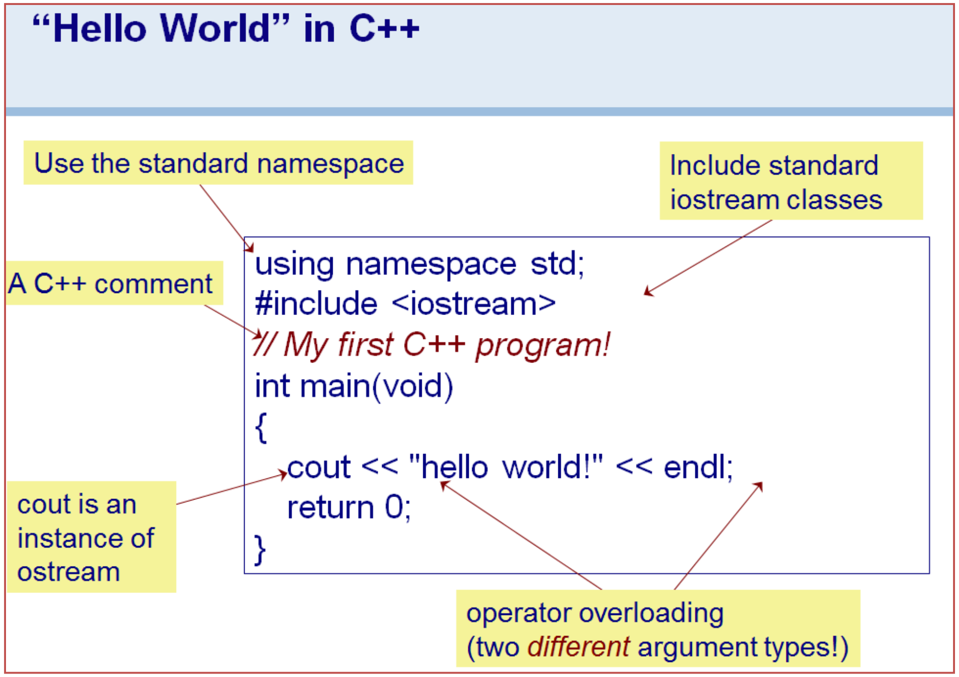
Most C programs are also C++ programs.

gcc hello.c –o hello



```
#include <stdio.h>

int main(void)
{
   printf("hello, world\n");
   return 0;
}
```

Include standard io declarations
A preprocessor directive
Write to standard output
char array
Indicate correct termination



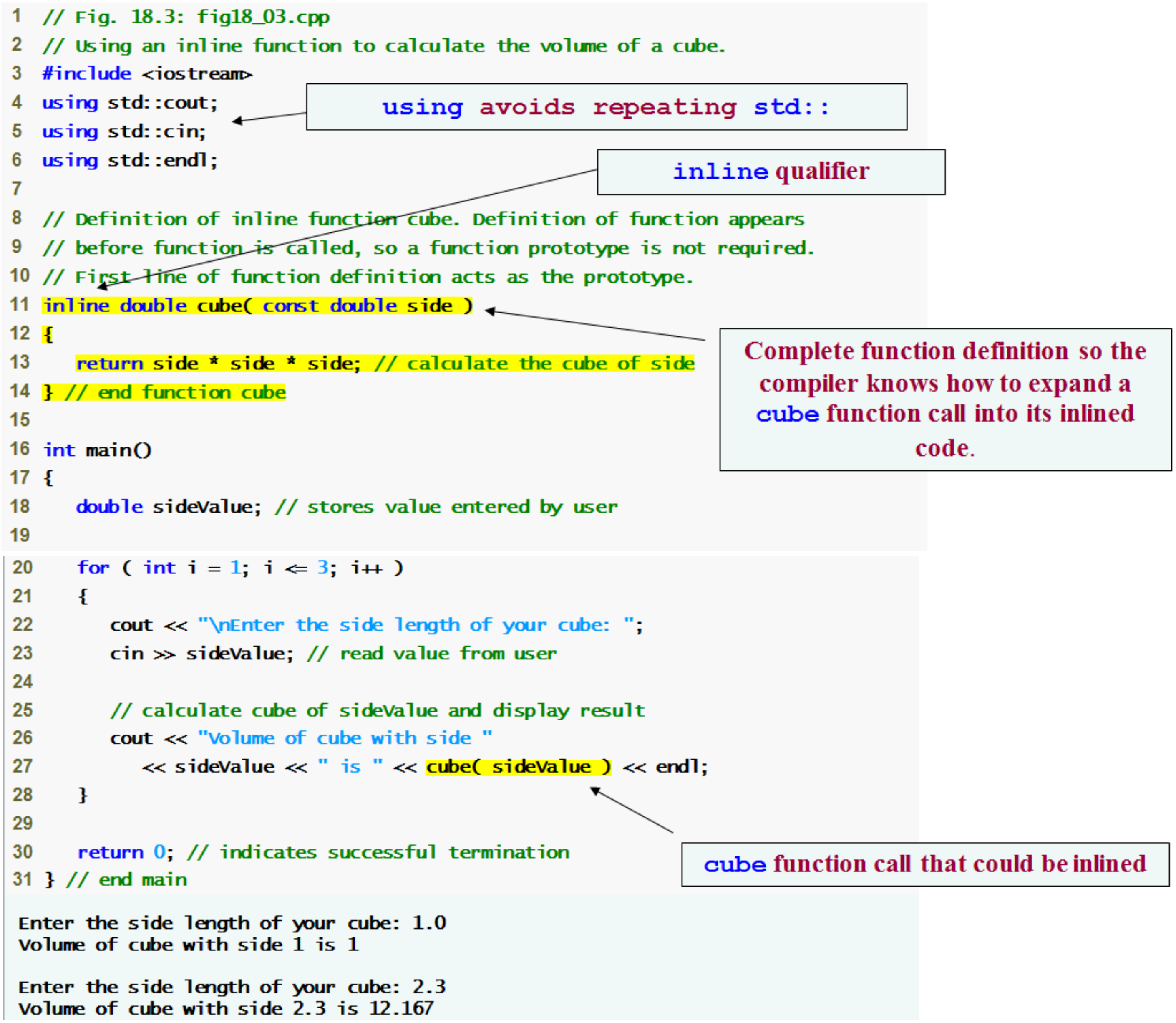
“Hello World” in C++

```
using namespace std;
#include <iostream>
// My first C++ program!
int main(void)
{
   cout << "hello world!" << endl;
   return 0;
}
```

Use the standard namespace
Include standard iostream classes
A C++ comment
cout is an instance of ostream
operator overloading (two *different* argument types!)

编译： Use gcc, Visual Studio, etc.

File types： .cc, .cp, .cpp, .CPP, .cxx, .c++, .C , .h, .H



```cpp
// Fig. 18.3: fig18_03.cpp
// Using an inline function to calculate the volume of a cube.
#include <iostream>
using std::cout;
using std::cin;
using std::endl;

// Definition of inline function cube. Definition of function appears
// before function is called, so a function prototype is not required.
// First line of function definition acts as the prototype.
inline double cube( const double side )
{
   return side * side * side; // calculate the cube of side
} // end function cube

int main()
{
   double sideValue; // stores value entered by user

   for ( int i = 1; i <= 3; i++ )
   {
      cout << "\nEnter the side length of your cube: ";
      cin >> sideValue; // read value from user

      // calculate cube of sideValue and display result
      cout << "Volume of cube with side "
         << sideValue << " is " << cube( sideValue ) << endl;
   }

   return 0; // indicates successful termination
} // end main
```

```
Enter the side length of your cube: 1.0
Volume of cube with side 1 is 1

Enter the side length of your cube: 2.3
Volume of cube with side 2.3 is 12.167
```

using avoids repeating std::

inline qualifier

Complete function definition so the compiler knows how to expand a cube function call into its inlined code.

cube function call that could be inlined

Fortran

JOHN BACKUS

United States – 1977

1977图灵奖

CITATION

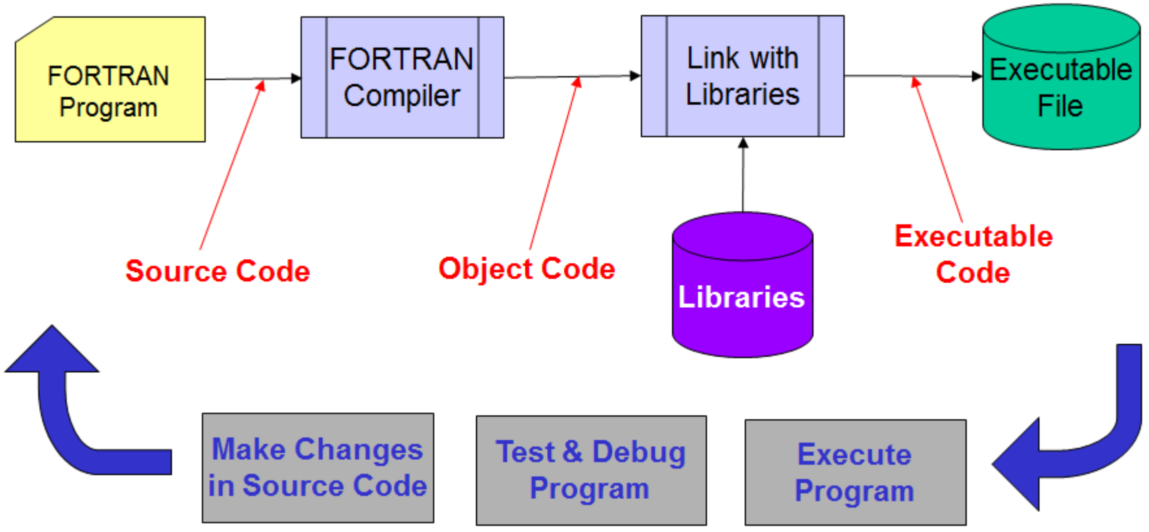
For profound, influential, and lasting contributions to the design of practical high-level programming systems, notably through his work on FORTRAN, and for seminal publication of formal procedures for the specification of programming languages.

- One of the oldest computer languages
  - created by John Backus and released in 1957
  - designed for scientific and engineering computations
- Version history
  - FORTRAN 1957
  - FORTRAN II
  - FORTRAN IV
  - FORTRAN 66 (released as ANSI standard in 1966)
  - FORTRAN 77 (ANSI standard in 1977)
  - FORTRAN 90 (ANSI standard in 1990)
  - FORTRAN 95 (ANSI standard version)
  - FORTRAN 2003 (ANSI standard version)

Fortran

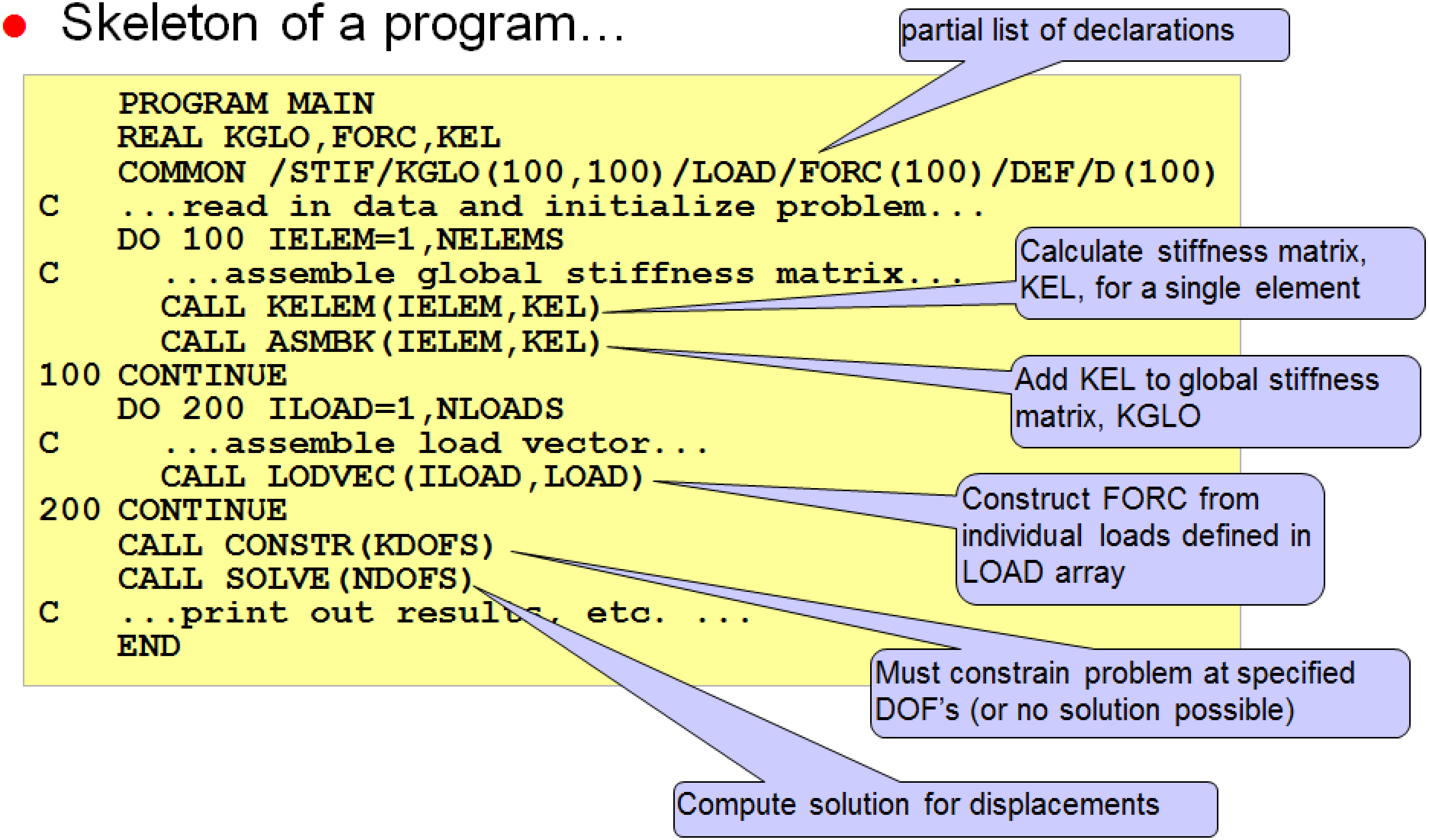
- **FORTRAN was created to solve scientific and engineering problems**
- **Introduced integer and floating point variables**
- Introduced array data types for math computations
- Introduced subroutines and subfunctions
- There is a free compiler in Unix-Linux systems
  - f77, g77 - **g95, gfortran**

- FORTRAN is a complied language (like C) so the source code (what you write) must be converted into machine code before it can be executed (e.g. Make command)



Fortran 结构

- Skeleton of a program…



```
      PROGRAM MAIN
      REAL KGLO,FORC,KEL
      COMMON /STIF/KGLO(100,100)/LOAD/FORC(100)/DEF/D(100)
C     ...read in data and initialize problem...
      DO 100 IELEM=1,NELEMS
C        ...assemble global stiffness matrix...
         CALL KELEM(IELEM,KEL)
         CALL ASMBK(IELEM,KEL)
100   CONTINUE
      DO 200 ILOAD=1,NLOADS
C        ...assemble load vector...
         CALL LODVEC(ILOAD,LOAD)
200   CONTINUE
      CALL CONSTR(KDOFS)
      CALL SOLVE(NDOFS)
C     ...print out results, etc. ...
      END
```

partial list of declarations

Calculate stiffness matrix, KEL, for a single element

Add KEL to global stiffness matrix, KGLO

Construct FORC from individual loads defined in LOAD array

Must constrain problem at specified DOF’s (or no solution possible)

Compute solution for displacements

Python

- Open source general-purpose language.
- Object Oriented, Modular
- Easy to interface with C++/C/Java/Fortran
- Interactive environment
- Interpreted and therefore slower than compiled languages

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales

斐波那契数列

```
def fib(n):
    a, b = 0, 1
    while a < n:
      print(a, end=' ')
      a, b = b, a+b
    print()
fib(1000)
```

```
$ python3.4 1.py
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

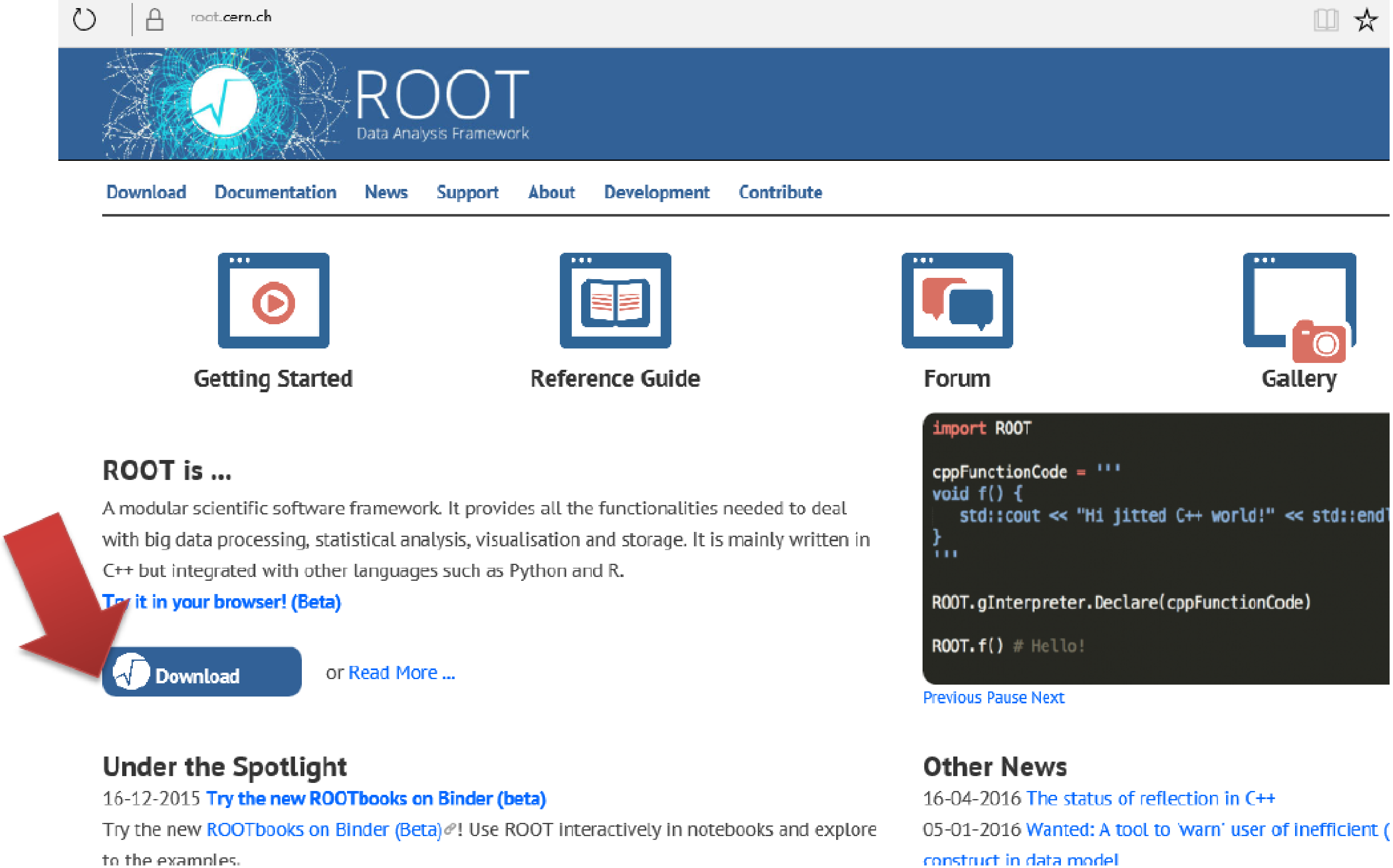
end=' '不换行是python3版本的用法，python2版本无法编译

CERN Root

欧洲核子中心(CERN)开发的大型科学数据处理软件

基于Cint(C/C++ interpreter, C-int)是一个C++解释器, 和GCC、VC等编译器不同,它是解释执行C++代码的

https://root.cern.ch/



ROOT is ...

A modular scientific software framework. It provides all the functionalities needed to deal with big data processing, statistical analysis, visualisation and storage. It is mainly written in C++ but integrated with other languages such as Python and R.

Try it in your browser! (Beta)

or Read More ...

```
import ROOT

cppFunctionCode = '''
void f() {
   std::cout << "Hi jitted C++ world!" << std::endl
}
'''

ROOT.gInterpreter.Declare(cppFunctionCode)

ROOT.f() # Hello!
```

Under the Spotlight

16-12-2015 Try the new ROOTbooks on Binder (beta)

Try the new ROOTbooks on Binder (Beta)! Use ROOT interactively in notebooks and explore to the examples.

Other News

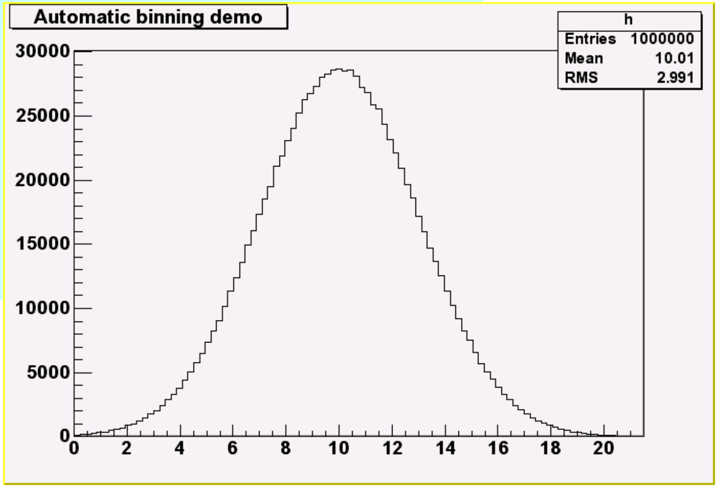
16-04-2016 The status of reflection in C++

05-01-2016 Wanted: A tool to 'warn' user of inefficient (construct in data model

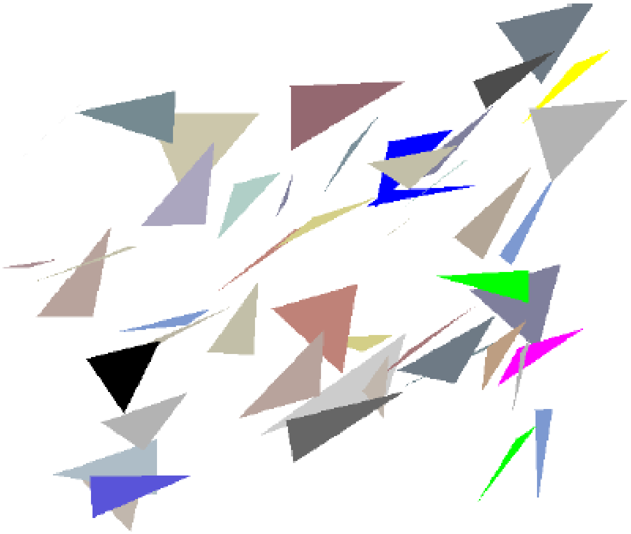
CERN Root: Histogram

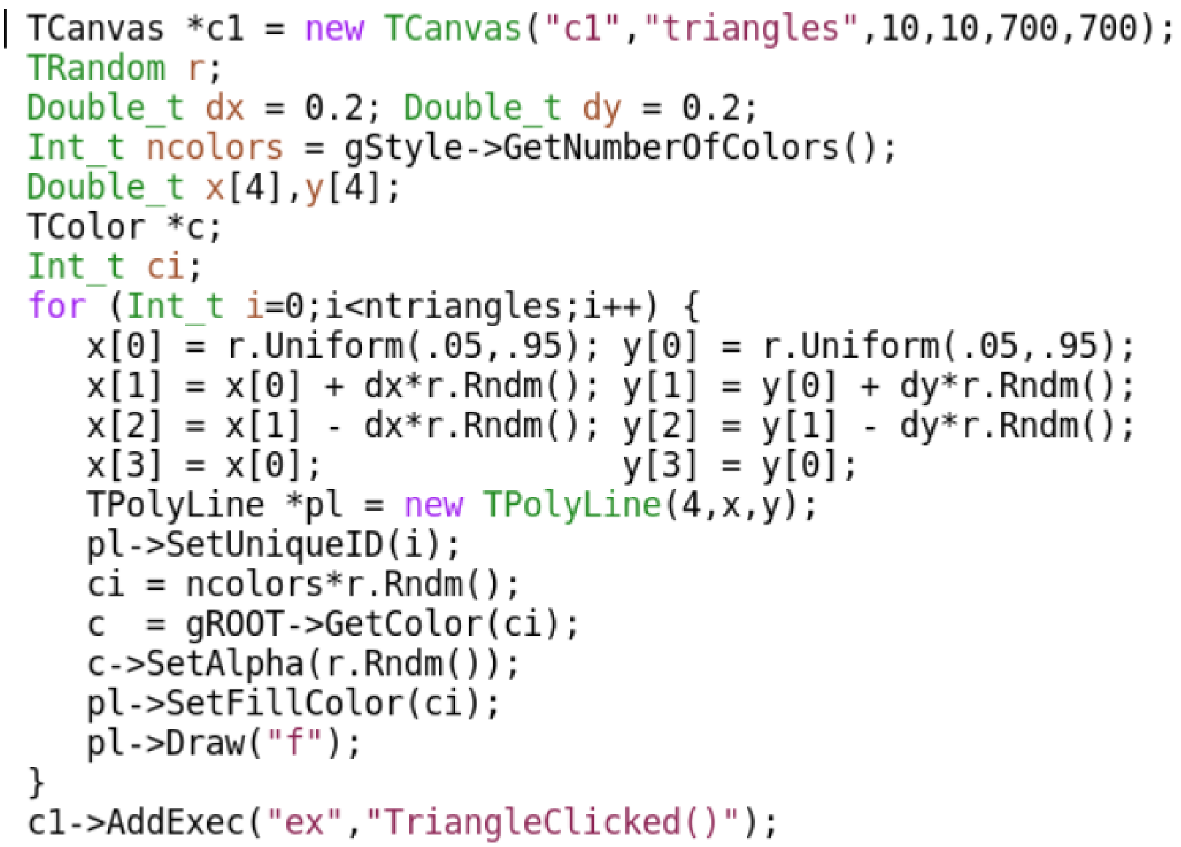
Bin， Events， Mean, RMS

```
#include "TH1.h"
#include "TF1.h"

void demoauto() {
   TF1 *f1 = new TF1("f1","gaus",0,30);
   f1->SetParameters(1,10,3);
   TH1F *h = new TH1F("h","Automatic binning demo",100,0,20);
   for (Int_t i=0;i<1000000;i++) {
      h->Fill(f1->GetRandom());
   }
   h->Draw();
}
```



CERN Root 例1： 三角图形

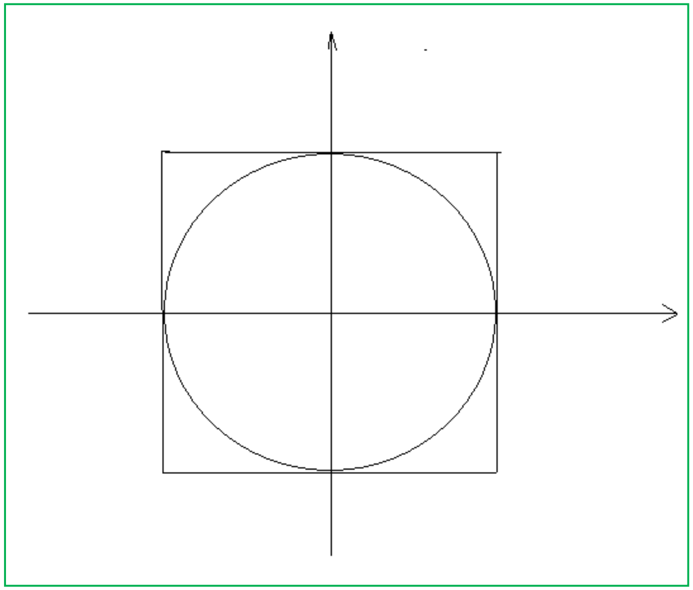




```
TCanvas *c1 = new TCanvas("c1","triangles",10,10,700,700);
TRandom r;
Double_t dx = 0.2; Double_t dy = 0.2;
Int_t ncolors = gStyle->GetNumberOfColors();
Double_t x[4],y[4];
TColor *c;
Int_t ci;
for (Int_t i=0;i<ntriangles;i++) {
   x[0] = r.Uniform(.05,.95); y[0] = r.Uniform(.05,.95);
   x[1] = x[0] + dx*r.Rndm(); y[1] = y[0] + dy*r.Rndm();
   x[2] = x[1] - dx*r.Rndm(); y[2] = y[1] - dy*r.Rndm();
   x[3] = x[0];               y[3] = y[0];
   TPolyLine *pl = new TPolyLine(4,x,y);
   pl->SetUniqueID(i);
   ci = ncolors*r.Rndm();
   c  = gROOT->GetColor(ci);
   c->SetAlpha(r.Rndm());
   pl->SetFillColor(ci);
   pl->Draw("f");
}
c1->AddExec("ex","TriangleClicked()");
```

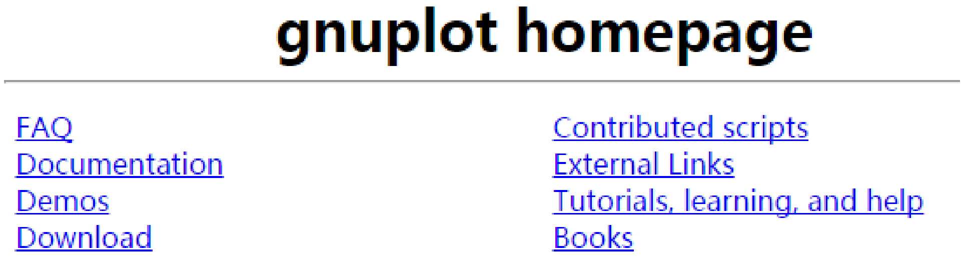
CERN Root例2： 随机数计算Pi

```
void random()
{
//TRandom3, is based on the "Mersenne
Twister generator", and is the recommended
one, since it has good random proprieties
(period of about 10**6000 ) and it is fast
        // create random number generator
        gRandom = new TRandom3(0);
        gRandom2 = new TRandom3(1);
        int n=20000;
        int ftot=0;
        for (int i = 0; i < n; ++i) {
          double x=gRandom->Uniform(-1,1);
          double y=gRandom2->Uniform(-1,1);
          double r=sqrt(x*x+y*y);
          if(r<1.0) {ftot++;}
        }

       cout<<4*float(ftot)/float(n)<<endl;

}
```



GNUPLOT

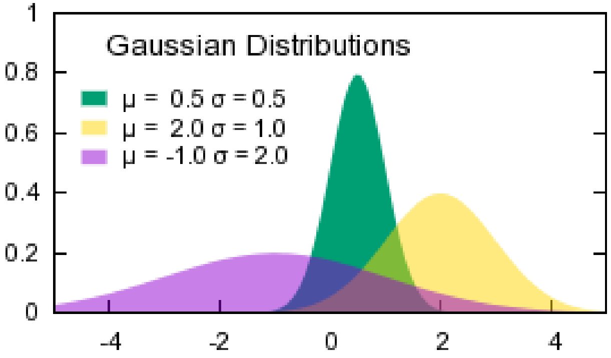
http://www.gnuplot.info/



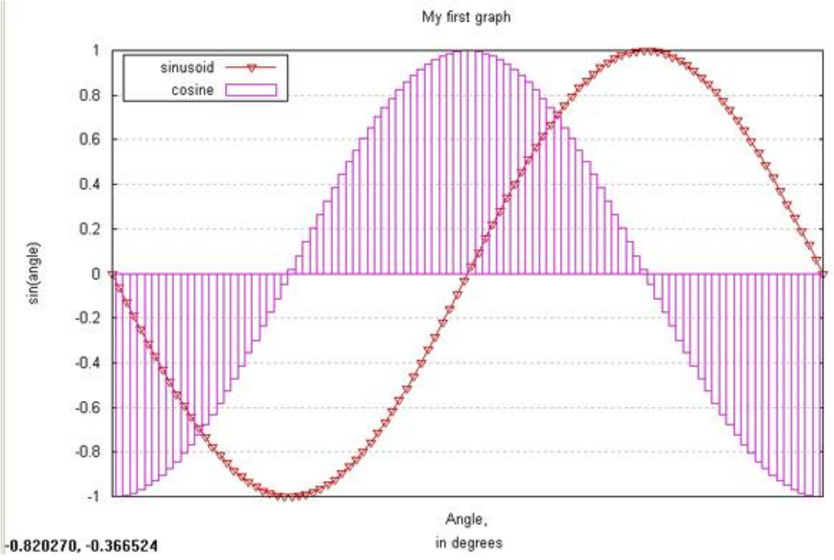
gnuplot homepage

FAQ
Documentation
Demos
Download

Contributed scripts
External Links
Tutorials, learning, and help
Books



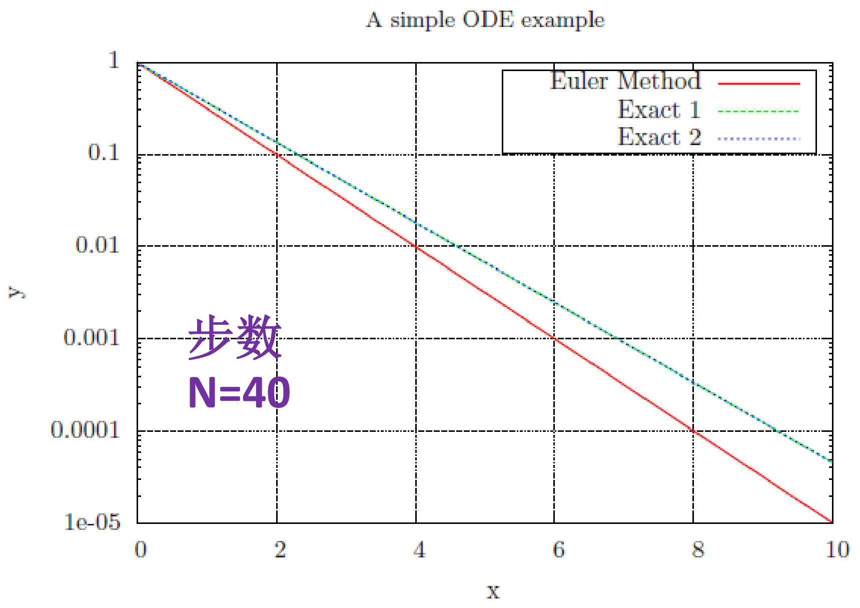
Gnuplot is a portable **command-line driven graphing utility** for Linux, OS/2, MS Windows, OSX, VMS, and many other platforms. The source code is copyrighted but freely distributed (i.e., you don't have to pay for it). It was **originally created to allow scientists and students to visualize** mathematical functions and data **interactively**, but has grown to support many non-interactive uses such as web scripting. It is also used as a plotting engine by third-party applications like Octave. Gnuplot has been supported and under active development since 1986.



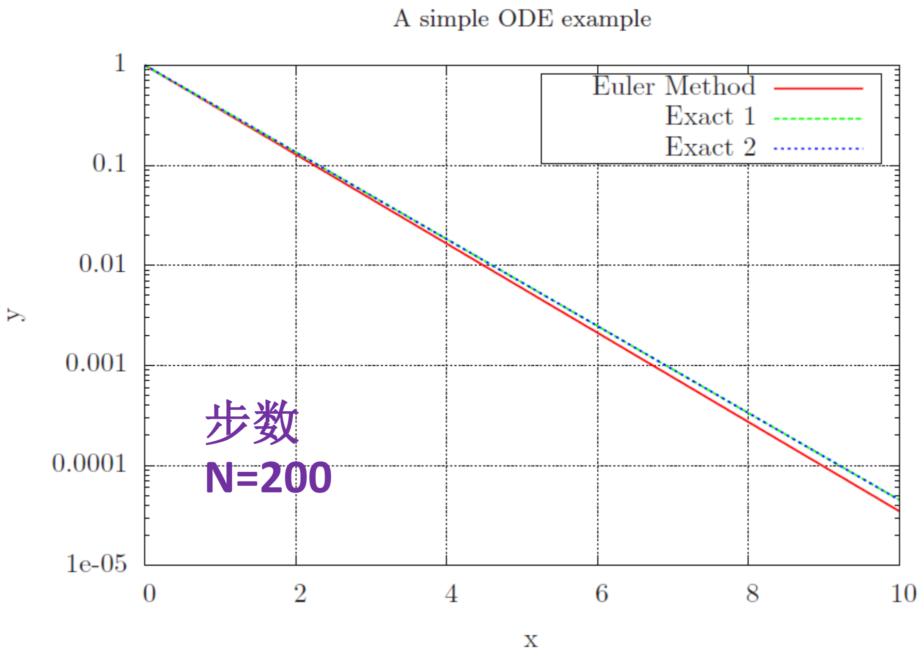
例： 常微分方程求解

An ODE Example: $y' = -y$,
with y(0)=1. x:[0-10]
Explicit Method

```
     t                     output              exact
 0.0000000E+00  0.1000000E+01  0.1000000E+01
 0.2000000E+00  0.8000000E+00  0.8187308E+00
 0.4000000E+00  0.6400000E+00  0.6703200E+00
 0.6000000E+00  0.5120000E+00  0.5488116E+00
 0.8000000E+00  0.4096000E+00  0.4493290E+00
 0.1000000E+01  0.3276800E+00  0.3678794E+00
 ……
 0.2000000E+01  0.1073742E+00  0.1353353E+00
 ……
 0.5000000E+01  0.3777893E-02  0.6737947E-02
 0.8000000E+01  0.1329228E-03  0.3354626E-03
……
 0.9400000E+01  0.2787593E-04  0.8272407E-04
 0.9600000E+01  0.2230075E-04  0.6772874E-04
 0.9800000E+01  0.1784060E-04  0.5545160E-04
 0.1000000E+02  0.1427248E-04  0.4539993E-04
```



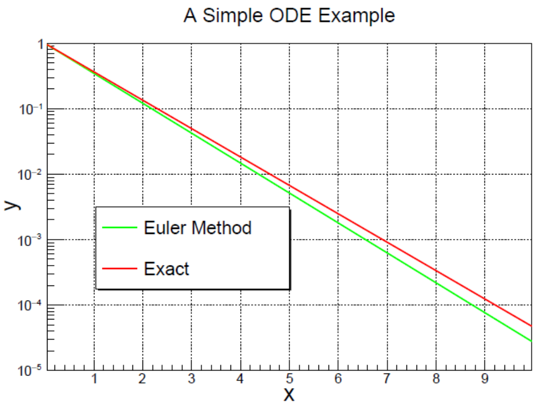
步数
N=40



步数
N=200

例2： CERN ROOT作图

```
void rootplot()
{
  TCanvas *c1 = new TCanvas("c1","",200,10,700,500);
  c1->SetGrid();

  Int_t n = 101;
  float Rbin[101],R2U[101],R2C[101];

  float xbin,xp,yp;
  int flag=0;
  int k0=0;
  int ka=0;
  int kb=0;
  FILE *fp = fopen("plot2.gnu","r");
  while (1) {
   flag=fscanf(fp,"%f %f %f",&xbin,&xp,&yp);
   if(flag!=3)break;
   Rbin[k0]=xbin;
   R2U[k0]=xp;
   R2C[k0]=yp;
   k0++;
  }
```

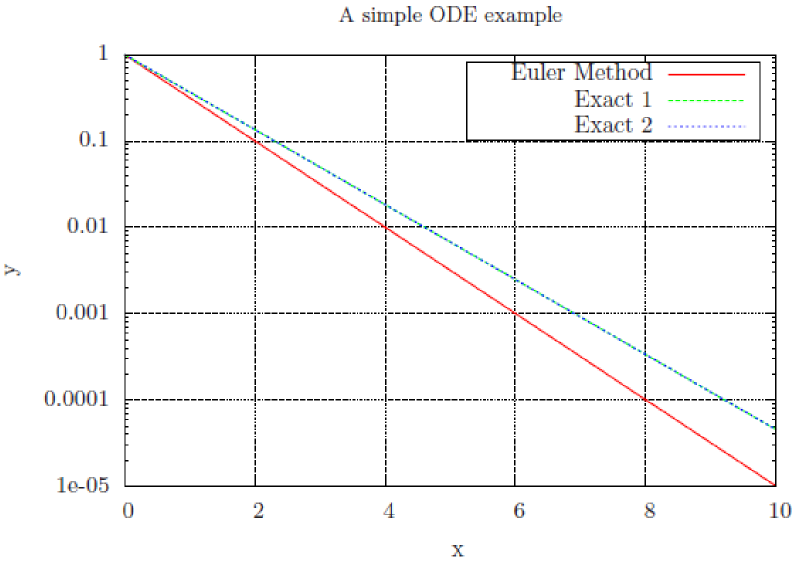


```
 TMultiGraph *mg = new TMultiGraph();
  mg->SetTitle("A Simple ODE Example; x; y");
  gr = new TGraph(n,Rbin,R2U);
  gr->SetLineWidth(2);
  gr->SetLineColor(3);
  gr2 = new TGraph(n,Rbin,R2C);
  gr2->SetLineWidth(2);
  gr2->SetLineColor(2);
  mg->Add(gr);
  mg->Add(gr2);
  mg->Draw("AC");
  mg->GetXaxis()->SetRangeUser(0.1,9.9);
  mg->GetYaxis()->SetRangeUser(0.00001,1.);
  mg->GetXaxis()->SetTitle("x");
  mg->GetXaxis()->SetTitleSize(.06);
  mg->GetYaxis()->SetTitle("y");
  mg->GetYaxis()->SetTitleSize(.06);
  mg->GetXaxis()->CenterTitle();
  mg->GetYaxis()->CenterTitle();
  mg->GetXaxis()->SetTitleOffset(0.6);
  mg->GetYaxis()->SetTitleOffset(0.6);
  c1->SetLogy();
  gPad->Modified();
  TLegend *l1 = new TLegend(0.18,0.3,0.5,0.5);
  l1->SetBorderSize(2);
  l1->SetFillColor(0);
  l1->AddEntry(gr,"Euler Method","l");
  l1->AddEntry(gr2,"Exact","l");
  l1->Draw();
  c1->SaveAs("example-N100-2.pdf");
}
```

例3： GNUPLOT作图

tot.tex

```
\documentclass[17pt]{article}
\usepackage{graphicx,color}
\begin{document}
\thispagestyle{empty}
\input{data.tex}
\end{document}
```



```
set pointsize 2.0
set tics in
set log y
set mytics 10
set grid
set xrange[0:10]
set terminal epslatex color
set output "data.tex"
set title '{\small A simple ODE example}'
set ylabel 'y'
set xlabel 'x'
set xlabel offset 0.5
set ylabel offset 2.5
set key box
plot "plot.gnu"   index 0 u ($1):($2)    title 'Euler Method' w l  lt 1 lw 3 ,\
    "plot.gnu"   index 0 u ($1):($3)    title 'Exact 1' w l  lt 2 lw 3 ,\
    exp(-x)      title'Exact 2'   w l  lt 3  lw 3
set output
!latex tot.tex
!dvips -E tot.dvi -o example1.eps
!epstopdf example1.eps
```

并行计算

Modern parallel programing: **open-MP** and **MPI**.

–Lots of processors connected and coordinating to solve large problems

M P I = Message Passing Interface.

https://www.mpi-forum.org/docs/

A standardized collection of routines (functions) which is implemented for each programing language (Fortran, C, C++, Python).

First standardized in 1994 (MPI-1.0) and second in 1997 (MPI-2.0) and (MPI-3.0) after 2012. Currently MPI-2.0 is most widely used.

The two most widely used implementations are

- MPICH http://www.mpich.org
- open-MPI http://www.open-mpi.org

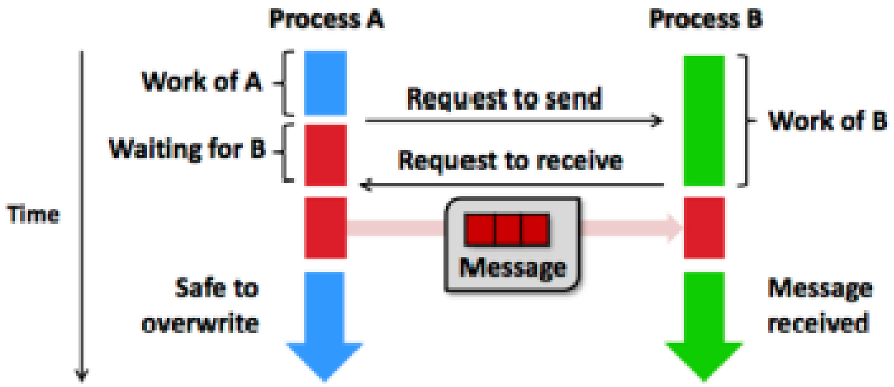
  Python implementation mpi4py

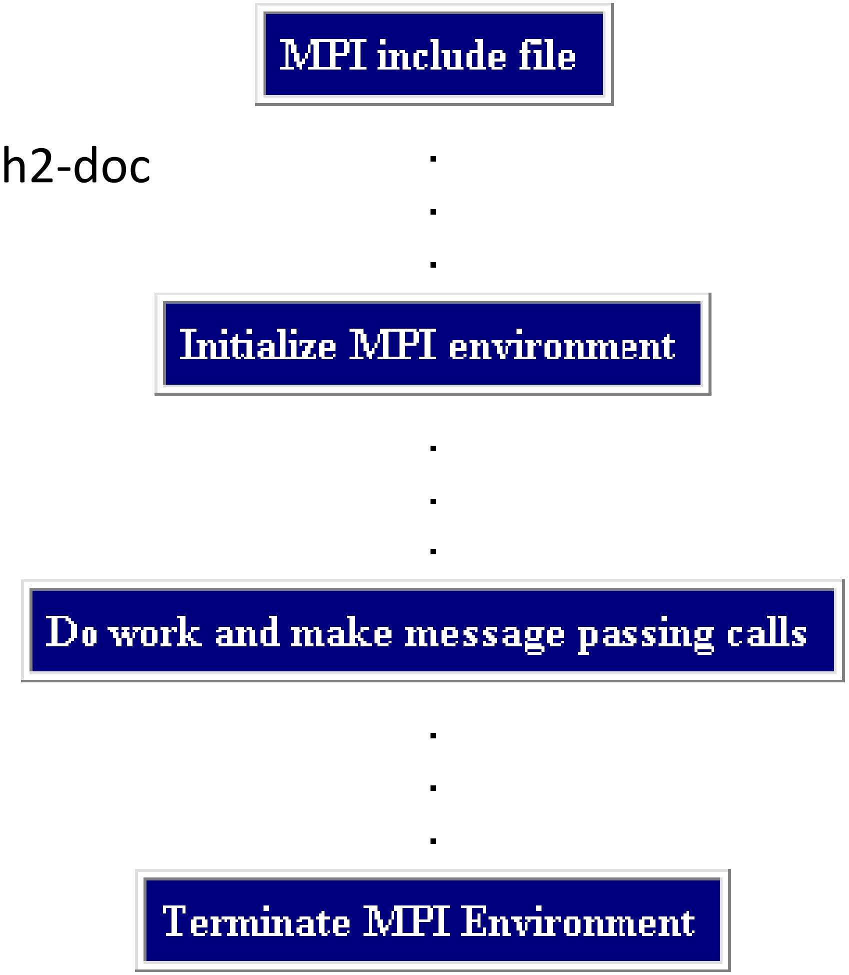
  (Note that open-MPI has nothing to do with openMP)

并行计算

```
sudo apt-get install libcr-dev mpich2 mpich2-doc
```

```
mpic++ -o example example.c
mpirun -n 2 ./example
```





MPI include file

.
.
.

Initialize MPI environment

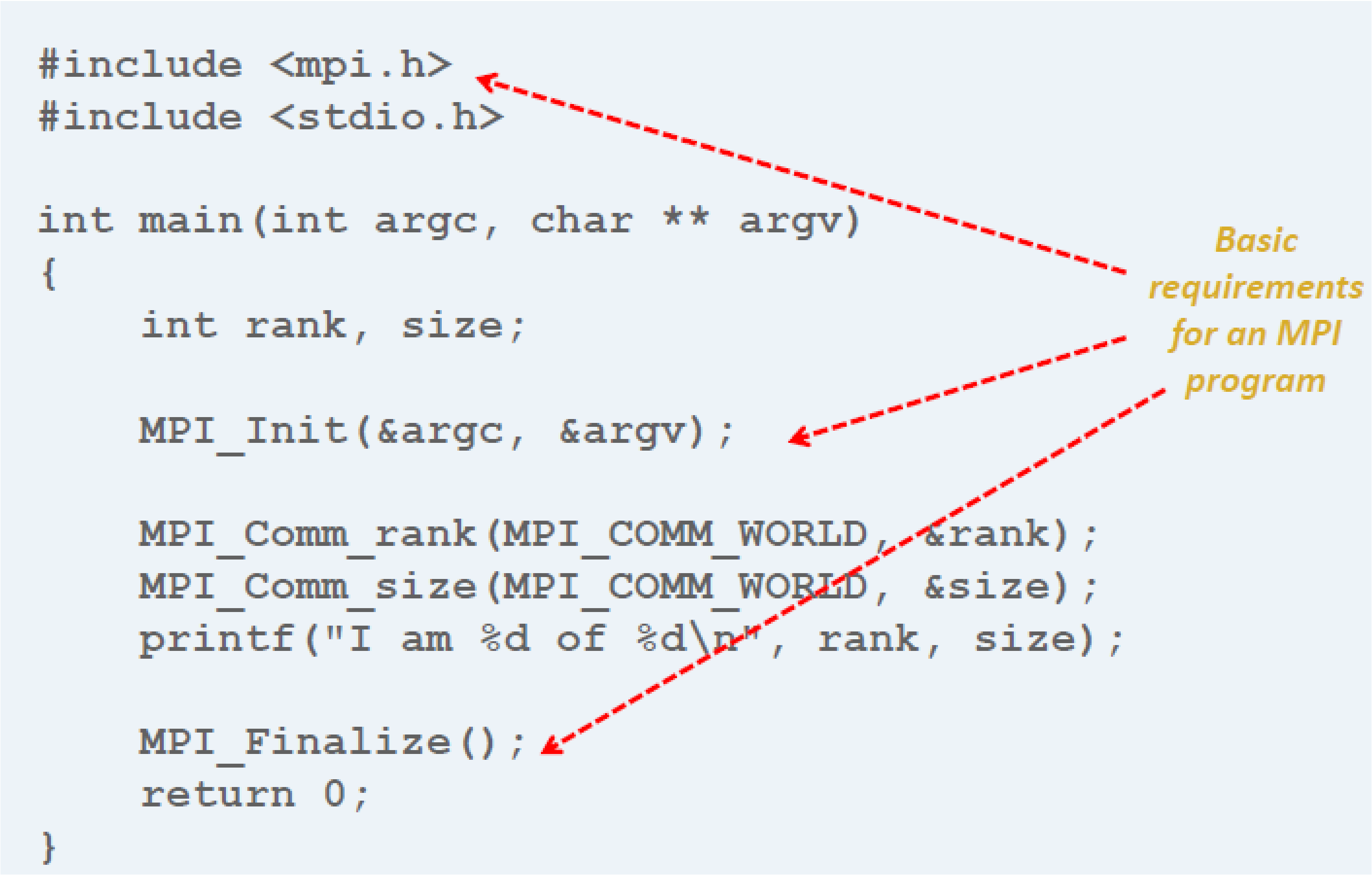
.
.
.

Do work and make message passing calls

.
.
.

Terminate MPI Environment

并行计算



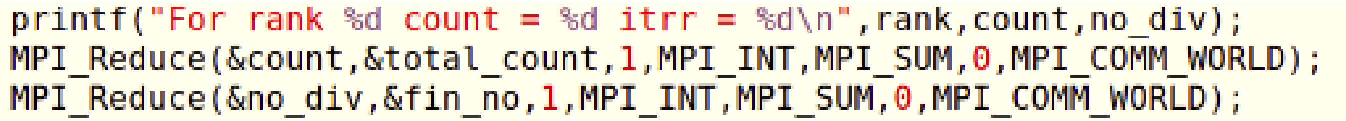
```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char ** argv)
{
     int rank, size;

     MPI_Init(&argc, &argv);

     MPI_Comm_rank(MPI_COMM_WORLD, &rank);
     MPI_Comm_size(MPI_COMM_WORLD, &size);
     printf("I am %d of %d\n", rank, size);

     MPI_Finalize();
     return 0;
}
```

Basic requirements for an MPI program

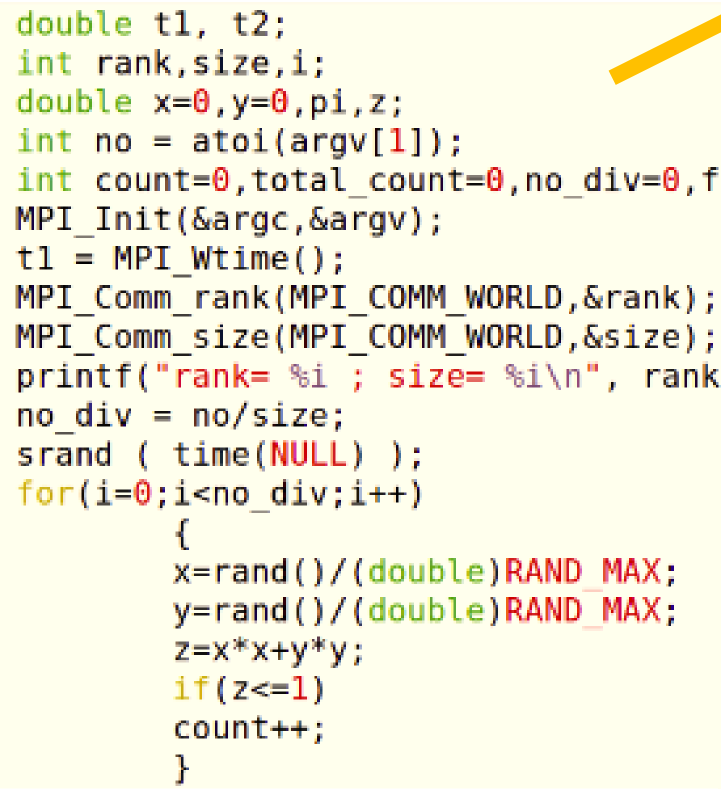
Data communication in MPI is like email exchange

–One process sends a copy of the data to another process (or a group of processes), and the other process receives it

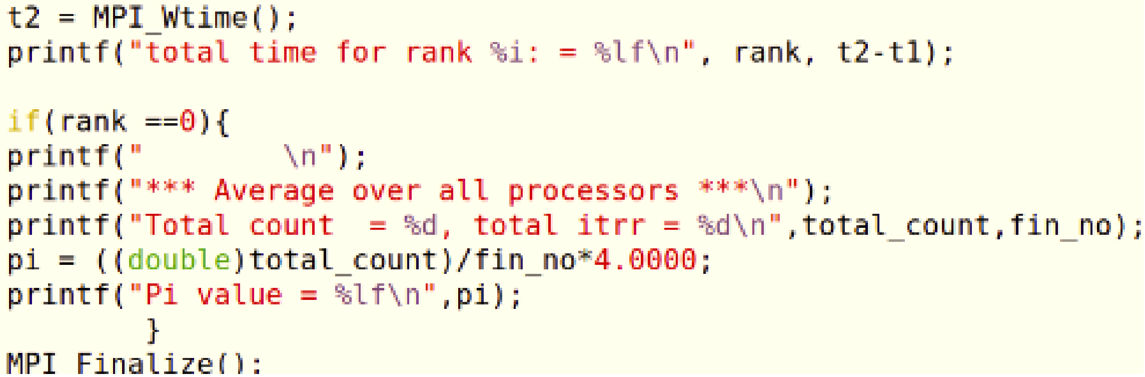
并行计算



```
printf("For rank %d count = %d itrr = %d\n",rank,count,no_div);
MPI_Reduce(&count,&total_count,1,MPI_INT,MPI_SUM,0,MPI_COMM_WORLD);
MPI_Reduce(&no_div,&fin_no,1,MPI_INT,MPI_SUM,0,MPI_COMM_WORLD);
```

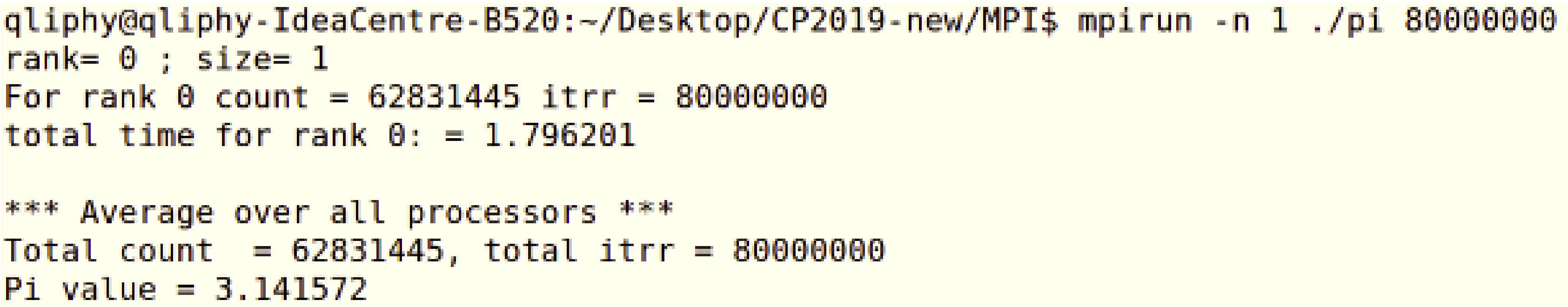


```
double t1, t2;
int rank,size,i;
double x=0,y=0,pi,z;
int no = atoi(argv[1]);
int count=0,total_count=0,no_div=0,fin_no = 0;
MPI_Init(&argc,&argv);
t1 = MPI_Wtime();
MPI_Comm_rank(MPI_COMM_WORLD,&rank);
MPI_Comm_size(MPI_COMM_WORLD,&size);
printf("rank= %i ; size= %i\n", rank, size);
no_div = no/size;
srand ( time(NULL) );
for(i=0;i<no_div;i++)
        {
        x=rand()/(double)RAND_MAX;
        y=rand()/(double)RAND_MAX;
        z=x*x+y*y;
        if(z<=1)
        count++;
        }
```

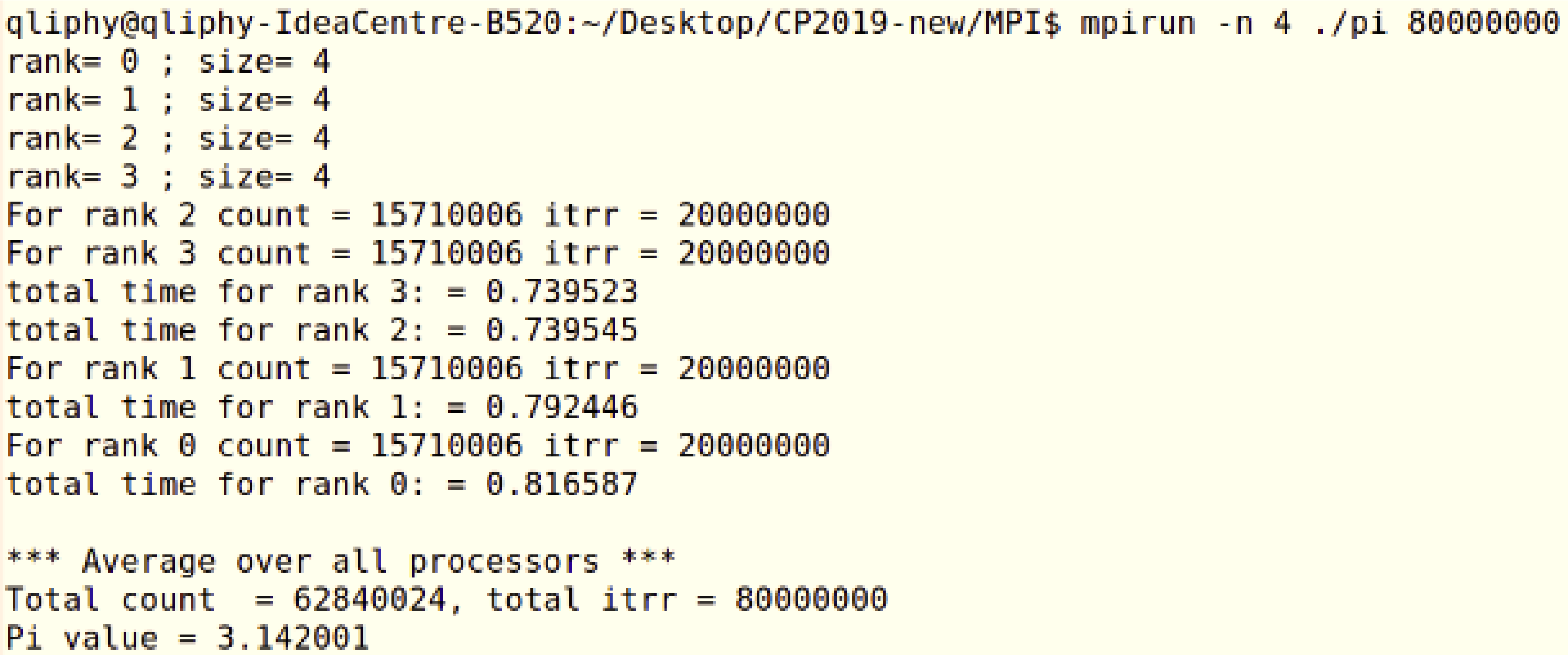


```
t2 = MPI_Wtime();
printf("total time for rank %i: = %lf\n", rank, t2-t1);

if(rank ==0){
printf("        \n");
printf("*** Average over all processors ***\n");
printf("Total count  = %d, total itrr = %d\n",total_count,fin_no);
pi = ((double)total_count)/fin_no*4.0000;
printf("Pi value = %lf\n",pi);
        }
MPI_Finalize();
```

并行计算



```
qliphy@qliphy-IdeaCentre-B520:~/Desktop/CP2019-new/MPI$ mpirun -n 1 ./pi 80000000
rank= 0 ; size= 1
For rank 0 count = 62831445 itrr = 80000000
total time for rank 0: = 1.796201

*** Average over all processors ***
Total count  = 62831445, total itrr = 80000000
Pi value = 3.141572
```

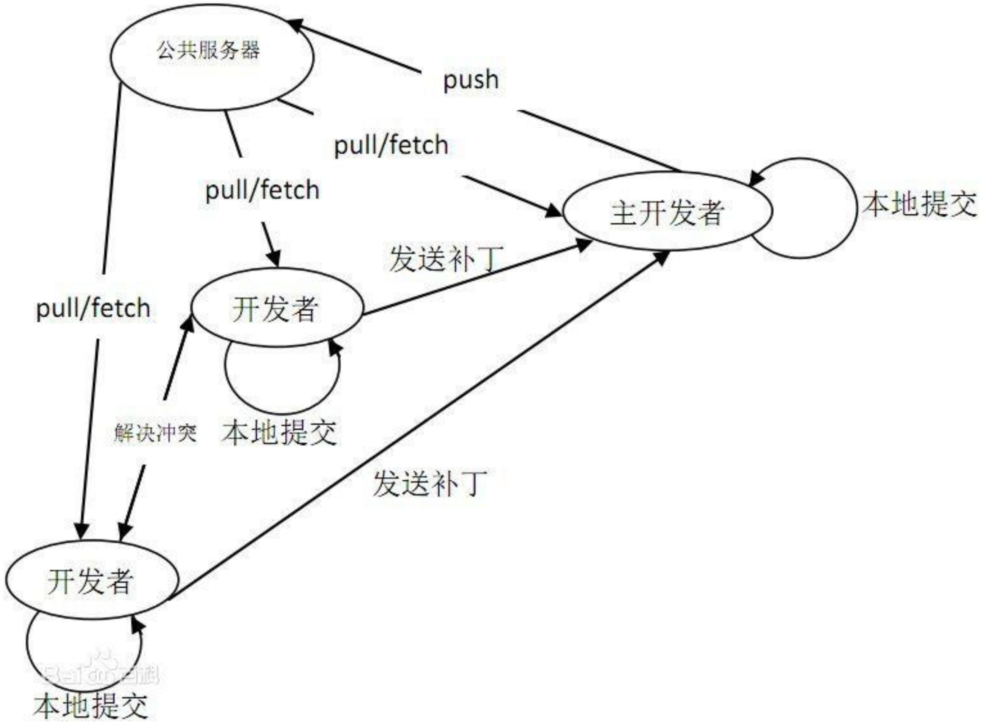


```
qliphy@qliphy-IdeaCentre-B520:~/Desktop/CP2019-new/MPI$ mpirun -n 4 ./pi 80000000
rank= 0 ; size= 4
rank= 1 ; size= 4
rank= 2 ; size= 4
rank= 3 ; size= 4
For rank 2 count = 15710006 itrr = 20000000
For rank 3 count = 15710006 itrr = 20000000
total time for rank 3: = 0.739523
total time for rank 2: = 0.739545
For rank 1 count = 15710006 itrr = 20000000
total time for rank 1: = 0.792446
For rank 0 count = 15710006 itrr = 20000000
total time for rank 0: = 0.816587

*** Average over all processors ***
Total count  = 62840024, total itrr = 80000000
Pi value = 3.142001
```

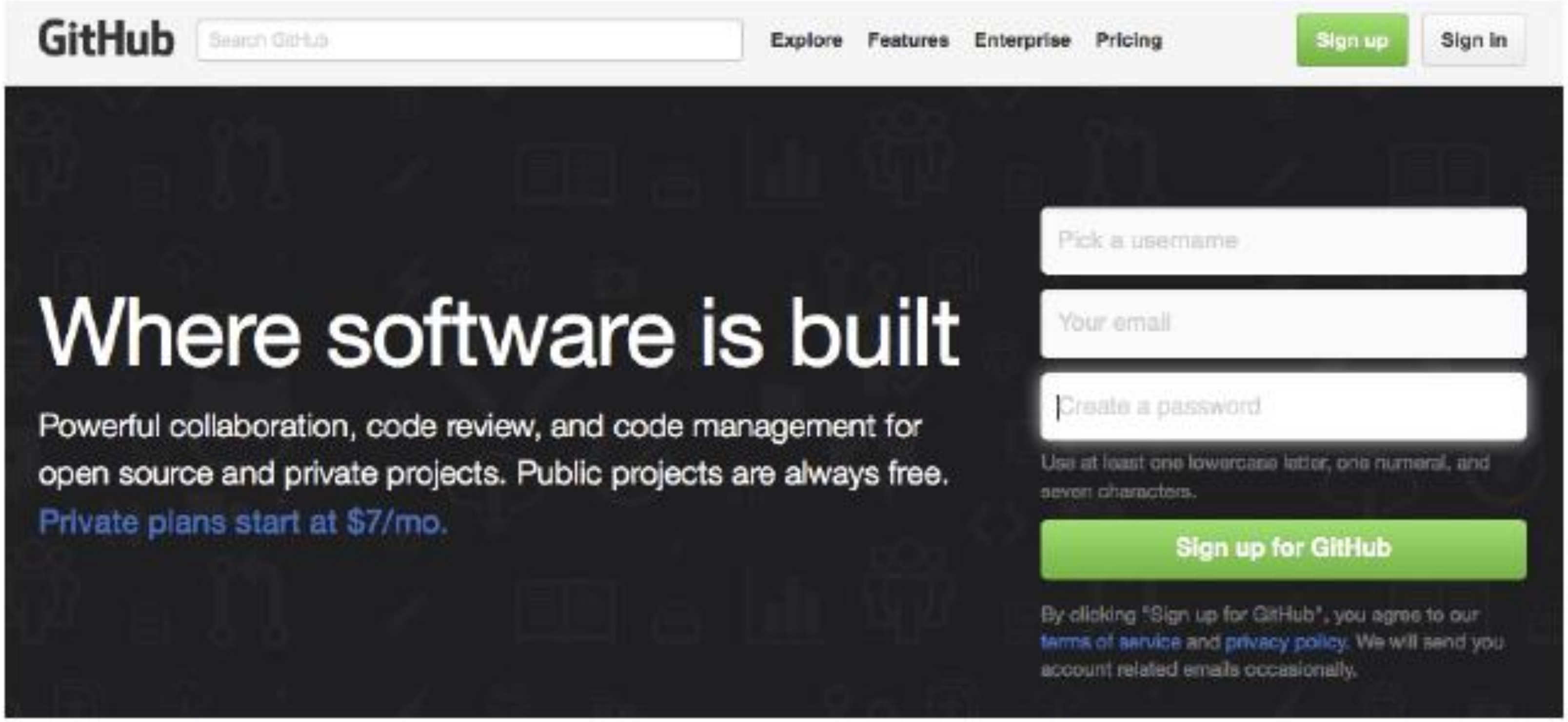
版本控制

- CVS (concurrent versions system)
- SVN (subversion)
- git

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.



Sign Up a Github Account



Where software is built

Powerful collaboration, code review, and code management for open source and private projects. Public projects are always free. Private plans start at $7/mo.

Use at least one lowercase letter, one numeral, and seven characters.

By clicking "Sign up for GitHub", you agree to our terms of service and privacy policy. We will send you account related emails occasionally.

You will receive an activation email to confirm your email address

Set Up Git

https://help.github.com/articles/set-up-git/

1. Download and install the latest version of Git.

In terminal:

1. $ git config --global user.name "*YOUR NAME*”
2. $ git config --global user.email "*YOUR EMAIL ADDRESS*"

Authenticating with GitHub from Git

- Connecting over HTTPS (recommended)

If you clone with HTTPS, you can
cache your GitHub password in Git using a credential helper.
https://help.github.com/articles/caching-your-github-password-ingit/

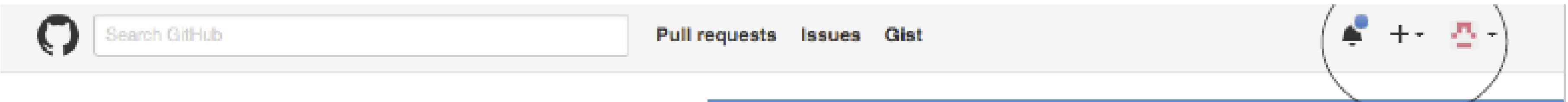
- Connecting over SSH

If you clone with SSH, you must generate SSH keys on each computer you use to push or pull from GitHub.

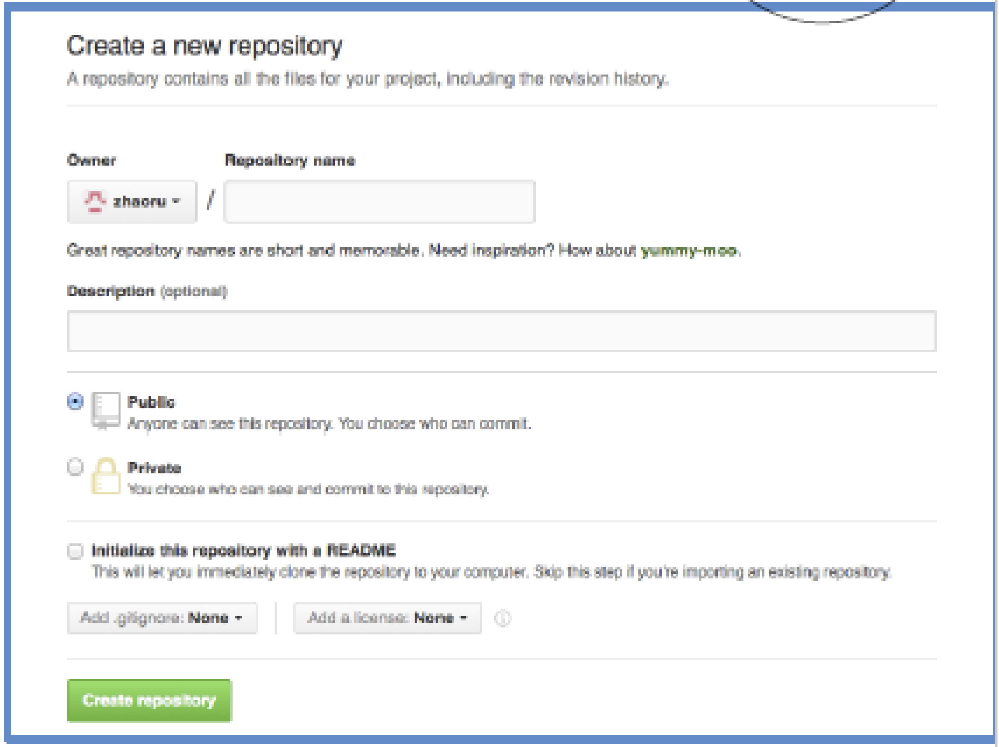
版本控制

Create a Repo

https://help.github.com/articles/create-a-repo/



- Set up “Repo Name”
- Public
- **Don’t** select “Initialize this repository with a README ”
- Create Repository



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: zhaoru / Repository name:

Great repository names are short and memorable. Need inspiration? How about yummy-moo.

Description (optional)

● Public
Anyone can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

□ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None

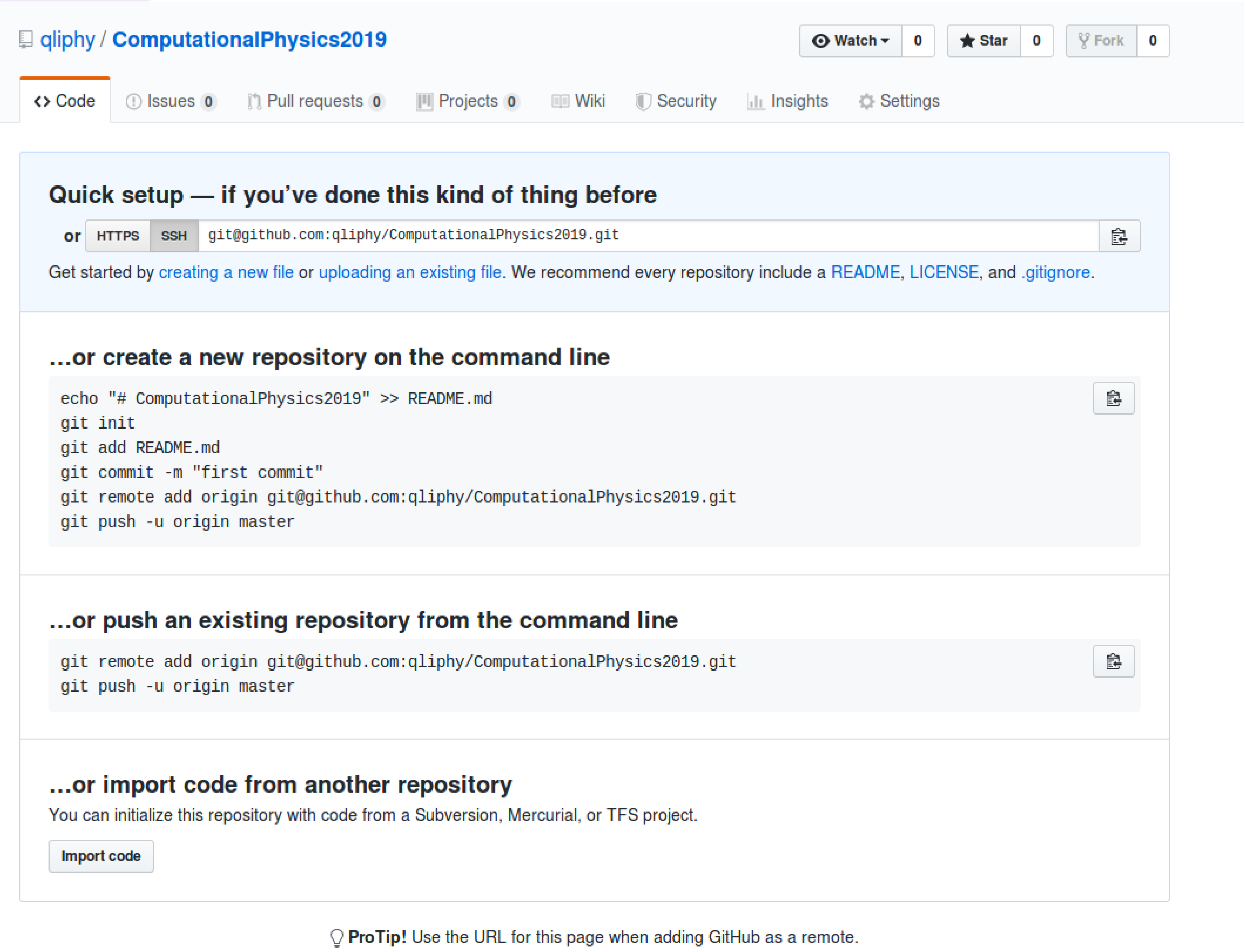
Upload your file to the Repo

In terminal:

```
$ cd “target folder”
$ echo "# TEST4" >> README.md
$ git init
$ git add README.md
$ git add *
$ git commit -m "first commit"
$ git remote add origin git@github.com:XXX/TEST4.git
(or https://)
$ git push -u origin master
```

Then refresh your website. Upload successfully!

版本控制



Quick setup — if you've done this kind of thing before

```
git@github.com:qliphy/ComputationalPhysics2019.git
```

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

…or create a new repository on the command line

```
echo "# ComputationalPhysics2019" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:qliphy/ComputationalPhysics2019.git
git push -u origin master
```

…or push an existing repository from the command line

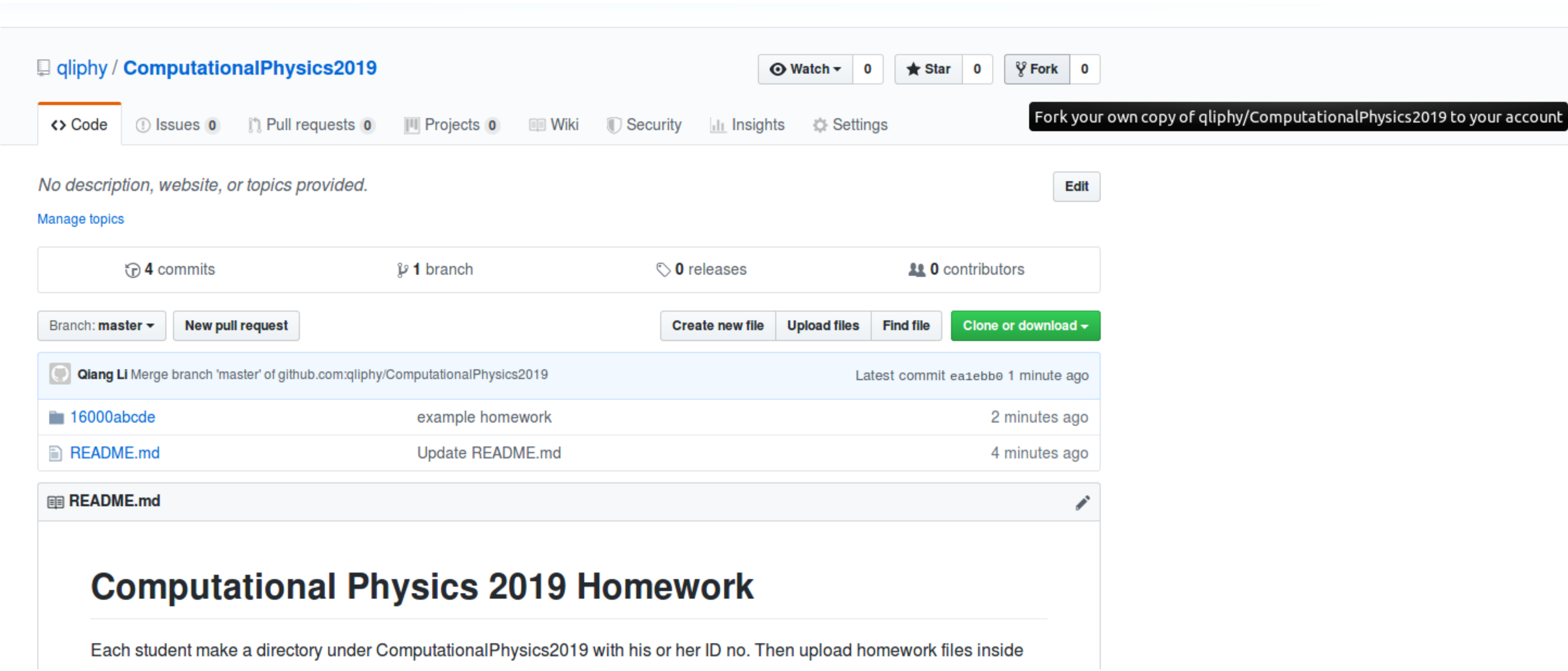
```
git remote add origin git@github.com:qliphy/ComputationalPhysics2019.git
git push -u origin master
```

…or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

ProTip! Use the URL for this page when adding GitHub as a remote.

版本控制



Fork your own copy of qliphy/ComputationalPhysics2019 to your account

No description, website, or topics provided.

4 commits | 1 branch | 0 releases | 0 contributors

Qiang Li Merge branch 'master' of github.com:qliphy/ComputationalPhysics2019 — Latest commit `ea1ebb0` 1 minute ago

16000abcde	example homework	2 minutes ago
README.md	Update README.md	4 minutes ago

README.md

Computational Physics 2019 Homework

Each student make a directory under ComputationalPhysics2019 with his or her ID no. Then upload homework files inside

作业： 每位同学push一个以学号命名的文件夹到qliphy/ComputationalPhysics2019 @github