数据结构和算法语言的学习环境设置和安装

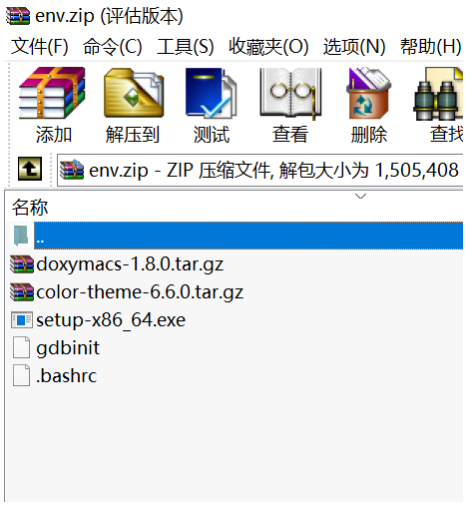
王何宇*
数学科学学院,
浙江大学

1 Cygwin下g++编程环境的设置

对迷恋Windows的同学而言，也许Cygwin是一个更加让人安心的选项。它本质上还是一个Windows应用软件，但是确具有几乎完全的Linux功能，当前，在Linux社区中，Cygwin 也被认为是一种Linux发行版。

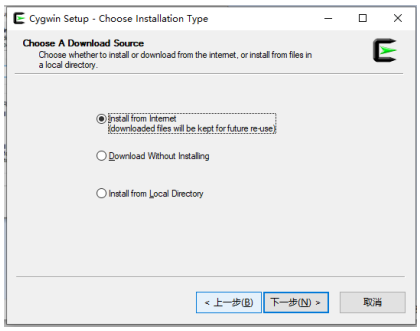
1.1 Cygwin安装步骤

下载群里的文件env.zip，解压后如下图：

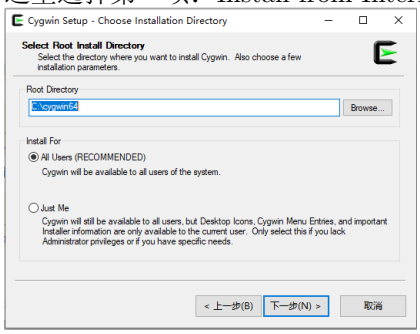


建议将其中的setup-x86_64.exe保存到一个合理的位置，比如在Downloads下新建一个bak文件夹，并将此类可能会继续使用的下载保存在一起（保持文件归类整洁是个重要的工作习惯）。双击运行，选下一步，如下图：

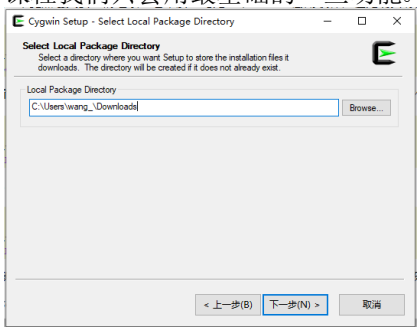
*email: wangheyu@zju.edu.cn



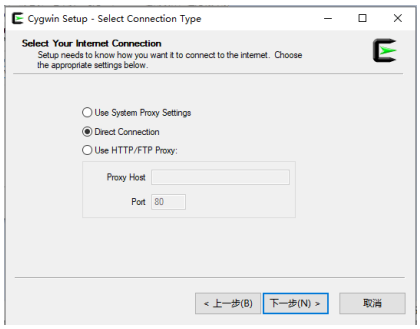
这里选择第一项：Install from Internet。然后继续下一步。看到如图：



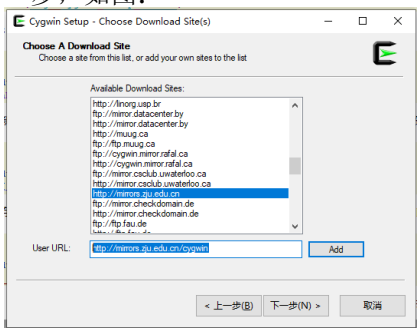
这里一般保持默认就可以，或者你自己重新选一个安装路径。注意Cygwin完全安装以后体量较大，会有好几个G，但一般我们不需要安装太多东西。特别是本课程我们只会用最基础的一些功能。再下一步，如图：



Cygwin会自动下载各种需要安装的包并缓存在这个目录下，如果你最近不打算改动安装，可以删除这个文件夹以节省空间，保留它可以避免重复下载。再下一步，如图：



这里选择上网方式，一般情况下选第二项：Direct Connection。除非你在一个需要代理服务器的环境。学校的VPN一旦打开也属于Direct Connection。再下一步，如图：



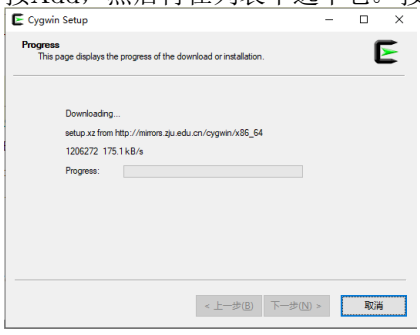
这一步很重要，选择一个速度更快的安装服务器。在校网内时，显然选择浙大自己的镜像服务器时最佳选择，如果在列表中找不到

http://mirrors.zju.edu.cn

或者有但无法连接，那么就手工在User URL：中输入

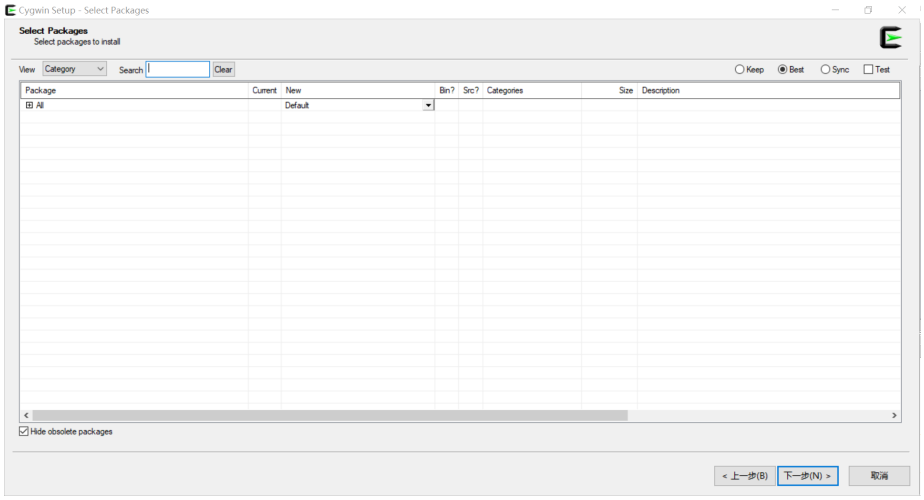
http://mirrors.zju.edu.cn/cygwin

然后按Add，然后再在列表中选中它。按下一步：

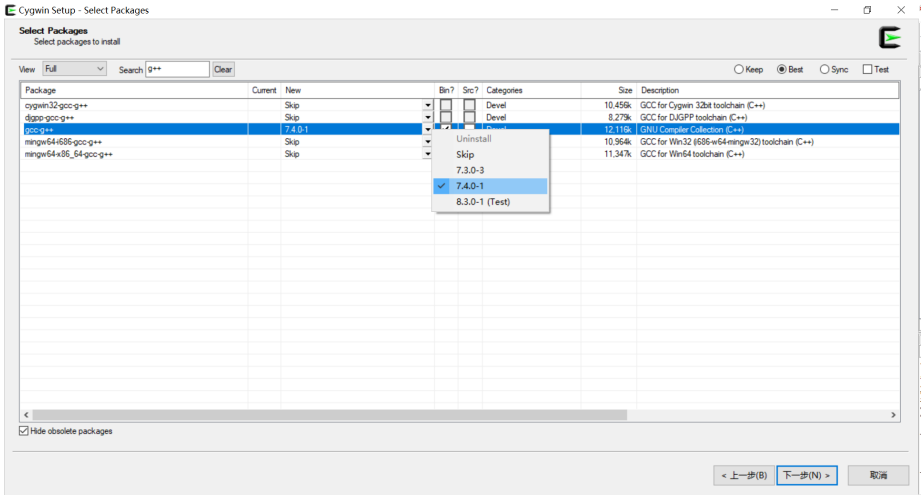


此时会出现如上图的界面，只要进度条在滚动就正常，这个界面持续时间和网速有关，如果停留1分钟以上，说明校网又在抽风，请检查连接。这里其实

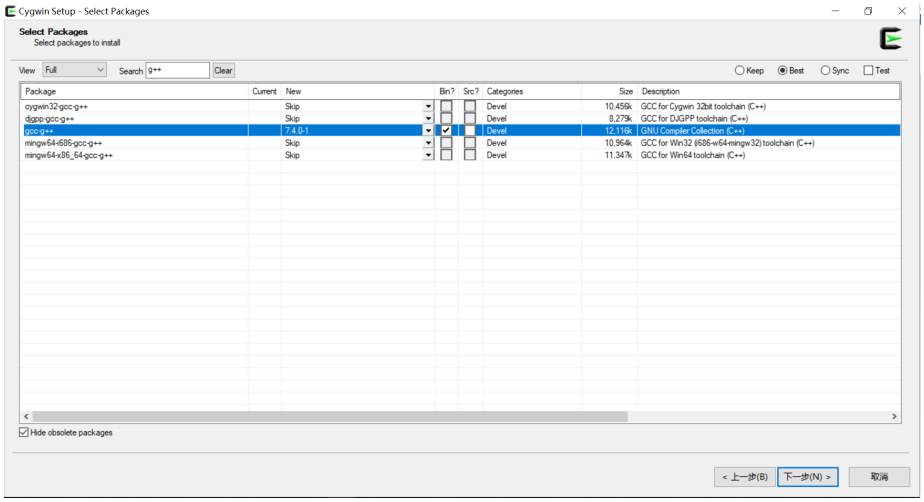
在下载全部可用安装包的目录，完成之后会看见如下图：



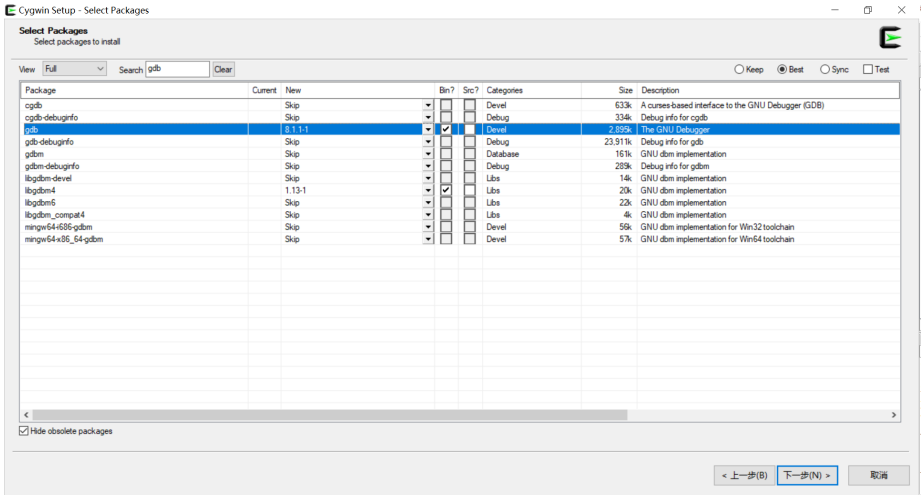
现在开始选择需要安装的软件包，首先是c++编译器，在View下拉列表中选Full，在Search中输入g++，这时会出来全部可用的g++软件，建议选择最标准的gcc-g++，在Skip后面的小箭头下选择7.4.0-1（一般我们总是选最新的但不带Test的版本）。具体见图：



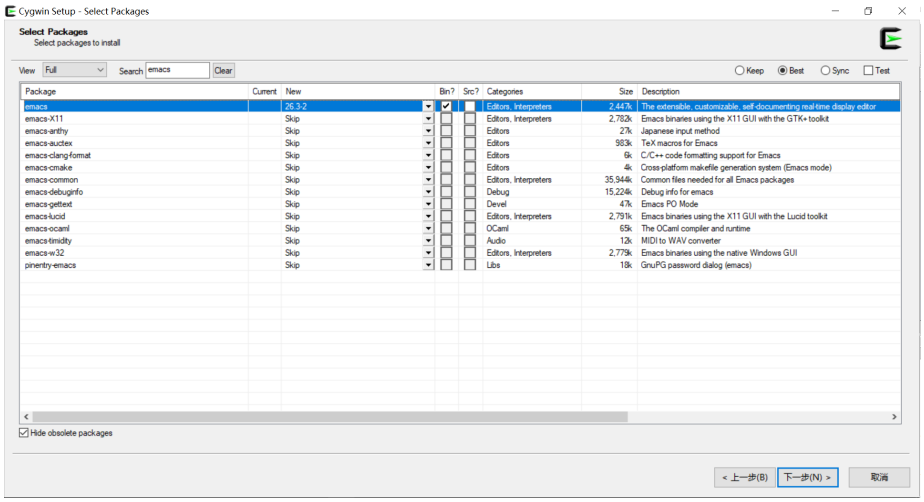
选好后的样子：



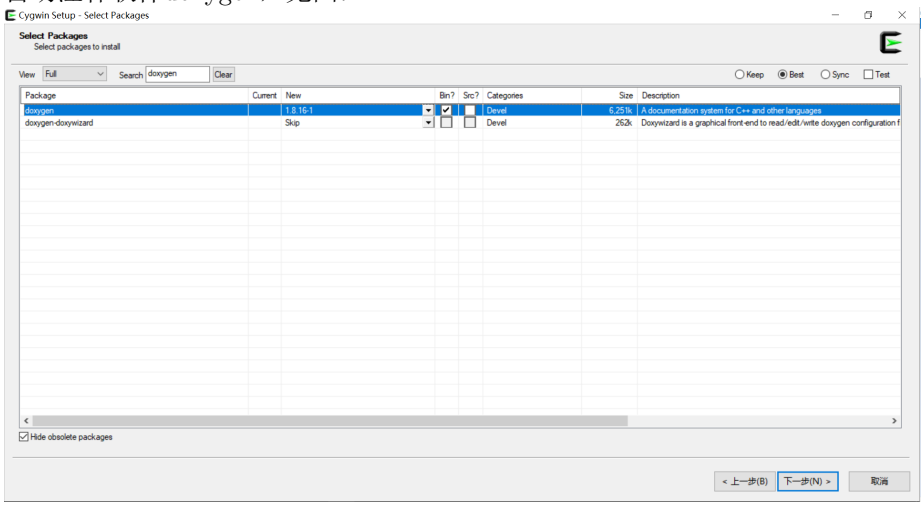
然后选择调试器gdb，重新在Search框中输入gdb，选择标准的gdb，版本还是选择Test前最新。如图：



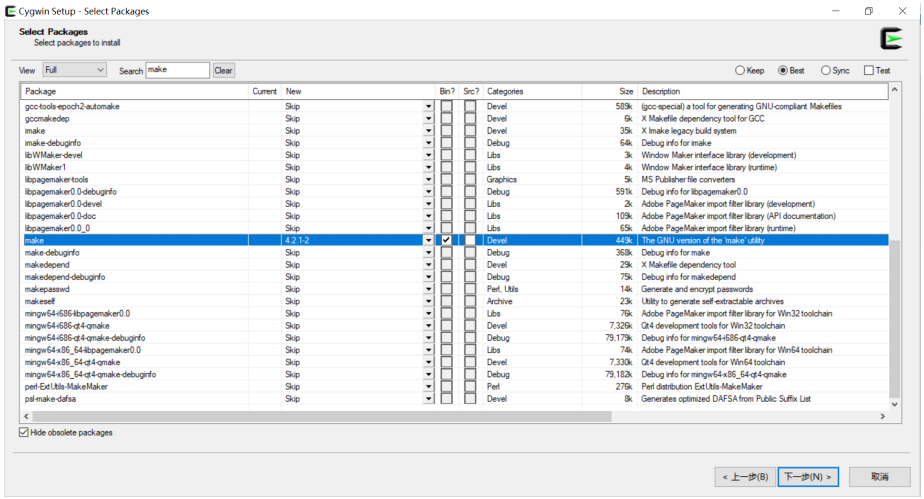
接着是源码编辑软件emacs：



自动注释软件doxygen，见图：

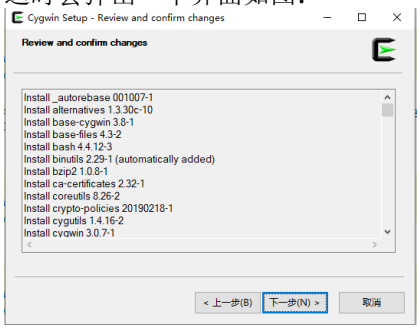


最后再加一个项目编译软件make（我们还没有介绍过），如图：

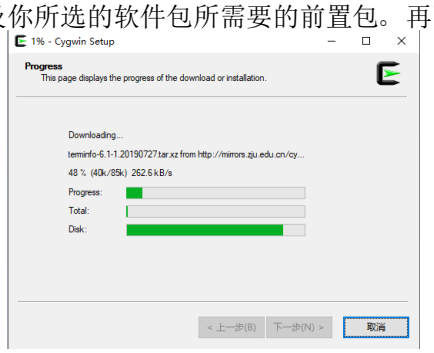


暂时就先装这些必须的软件。选择下一步。

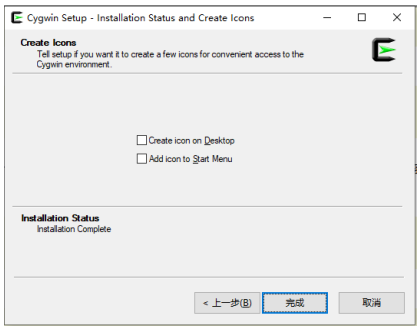
这时会弹出一个界面如图：



让你确认全部安装的软件包。注意Cygwin会自动选择一些必须的基础包，以及你所选的软件包所需要的前置包。再下一步，安装终于开始了，如图：



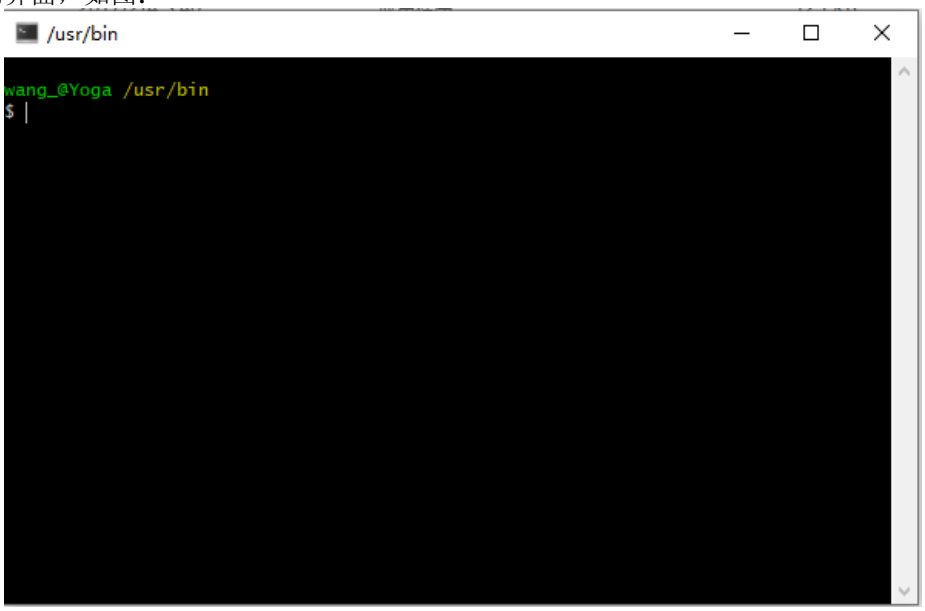
会有进度条出现。现在影响速度的除了网速，还有你机器的速度。一直到出现图：



再按完成，则Cygwin安装成功。这里如果勾选上两个选框（第一次安装建议勾上），则会自动在桌面和命令栏建立快捷链接。在我的机器上，这样安装的硬盘占用空间是778M。

1.2 Cygwin的配置和使用

双击Cygwin的图标，我们会看到一个土土的终端界面，这就是Cygwin的启动界面，如图：



我们首先熟悉一下终端命令，准确的说是shell命令。所谓shell，就是一个最基本的人机对话界面，它是基于字符和命令行的，我们输入每一个命令，都需要拍回车键发出，操作系统收到以后就会以一定的方式回显，如果有错，就会报错。Linux的一个基本原则是，如果正常完成任务，没有任何错误，则不提供任何反馈，即所谓“没有消息就是好消息”。这是因为在古老的年代，程序员都是通过非常缓慢的网络连接非常低速的主机，甚至传递字符都是一个

负担，因此尽量减少不必要的通讯。然而到了今天，这一原则有了新的意义：减少对程序员不必要的干扰，提高工作效率。Linux下的很多文化都带有这种从Unix继承过来的精简到近乎废土的风格。一旦你熟悉了这种文化，可能标志着你真正进入了计算机世界。同时你的编程效率也会有一个非线性的提高。

大家应该已经知道，Linux和Windows一样，具有树状文件结构。这棵树的根用/表示。命令：

cd /

的效果就是把当前目录设为根目录。cd是change directory的缩写。而ls则是列出当前目录下的全部文件和子目录。

ls

尝试一下这两个命令。首次启动时，Cygwin会自动建立属于你的用户目录，当前用户目录有一个快捷表示~ ，因此在任何位置输入：

cd ~

就会进入自己的用户目录。初始时用户目录是空的。你可能已经注意到我的光标之前有一串字符总是跟随：

wang_@Yoga

这里wang_是我的用户名，Yoga是我的机器名。这些都是Cygwin自动设置的，你则会根据你的具体情况显示。你可能会觉得这个光标不够方便，也许你记得我上课时，光标之前会自动跟随当前目录的提示，而且有时你的显示甚至会有一些乱七八糟的字符，想解决这个问题，在Windows下，把env.zip文件中的.bashrc这个文件，复制到：

C:/cygwin64/home/你的用户目录

下，如果你的Cygwin没有安装在C:/cygwin64下，那么这里要作相应的修改。我的这个目录就是：

C:/cygwin64/home/wang_

这里已经有一个.bashrc，覆盖掉。然后关闭Cygwin，再重新打开。现在显示舒服多了。这个文件其实是shell的配置文件，里面规定了文本的显示方式和格式，有兴趣的同学自己可以研究一下，但不要轻易改动，或者改动前，一定要留一个原文件的备份。这也是Linux 社区的另一个风格，所有的配置文件，都是可读的文本文件。在对它们做出修改之前，切记备份。备份可以用复制命令：

cp .bashrc bak.bashrc

在shell下，以.开头的文件都会自动隐藏，但只是看不见，一样可以操作。你用

ls -a

命令就可以看到隐藏文件和目录。而

ls -l

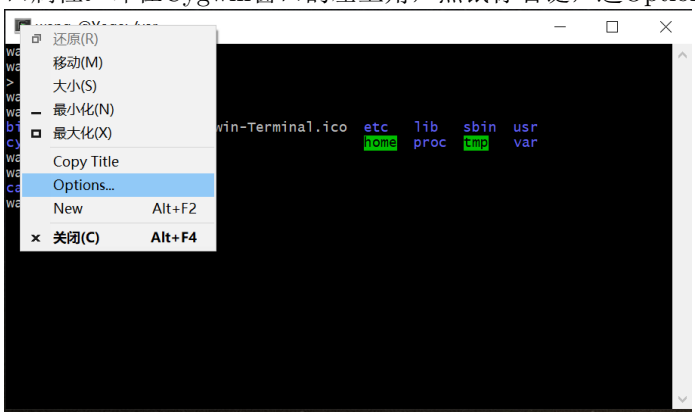
命令可以看到文件和目录的详细信息。这两个参数可以结合使用：

ls -al

以上都可以尝试一下。

cd /
ls -al
你会看到，根目录下，有bin、dev、etc、lib、sbin、usr、var、home、tmp这几个目录。这些是几乎所有Linux/Unix操作系统都具有的目录结构，包括Mac OS（它其实只是穿了漂亮马甲的Linux）。它们有一些约定的含义，但未必总是被认真执行。一般来说，bin下放的是系统即一般可执行文件，dev下是和设备有关文件，lib是各种c/c++的已编译库，就是我们已经遇到过的类似于.o文件的那些。整个Liunx系统都是用c语言编写的，因此它的一切功能都基于这些可执行库。sbin是需要有管理员权限才能执行的可执行文件。但是注意，在Cygwin下继承了Windows的文件权限——没有权限！所以实际上Cygwin下任何用户都是管理员，具有最高权限，可以随时杀掉整个系统，所以一定要小心。所以它只能是一个学习系统。usr一般会放应用软件，其中usr/local一般会放用户自己安装的软件。var曾经的含义是临时目录，但现在大都用来放系统日志。一台web服务器，一般也会把自己的web文件根目录放在var/www下，具体原因不清楚了。home目录是放所有用户数据的目录。一般一台正常的Linux主机，会有大量的用户，他们每人都会在home目录下开始自己的个人用户目录。一般普通用户只能修改自己用户目录下的文件，其他位置的文件都需要管理员才能修改。唯一的例外是tmp目录，这是真正的临时目录，专门用于放置临时数据，这里的数据在一定的周期内会自动清除，所以不能放有用的长期数据。

如果你对Cygwin中的窗口、字体不满，有一个办法是直接在Windows中修改窗口属性。即在Cygwin窗口的左上角，点鼠标右键，选Options...如图：



然后在里面做各种修改。直到自己满意。

1.3 Emacs的设置

为了编程，我们需要一个文本编辑软件，推荐大家寻找一个功能丰富、结构简单、界面丑陋的软件。简单丑陋的原因是不愿意花费太多资源在漂亮的窗口以及动态效果上，同时在一些极端情况下，我们仍然需要通过基本网络通讯

如ssh连接远程主机，这个时候一个轻量级的编辑软件显得格外重要。Linux系统下流行的两大编辑软件vim和emacs都符合这个风格。在网络上能找到关于二者的大量教程。大家自己去学习一下。我本人使用的是emacs，所以下面适当介绍一下emacs的配置和使用。不打算用emacs的同学可以跳过这一节。除了这两者，还可以考虑的编辑软件有gedit、eclipse、vscode等等，但它们大都需要更多的硬件资源。

我们在安装Cygwin的时候已经选择了emacs，所以现在在shell下可以直接用emacs命令启动它。顺便说一下退出的命令是control-x-c，同时按住这三个键（以后用C-xc表示）。emacs定义了大量的类似快捷键以加快工作效率，并且它自带shell，号称一旦启动emacs，你就再也不需要其他软件……（当然是扯淡！）一般我们还是只把它当作一个编辑软件使用，正常的启动方式是：

emacs 文件名

比如：

emacs hello.cpp

你接下去可以如此完成一个hello.cpp，然后按C-xs保存，再用C-xc退出。尝试一下在外部用g++编译。你会注意到emacs有一点自带的语法高亮功能，但似乎不如我上课时那么好看和快捷。和基础shell一样，emacs同样有配置文件，就是用户目录下的.emacs文件。这里规定了emacs的状态、颜色、字体、附加功能和快捷键等等。在介绍一个简单的配置之前，我们先做两个准备工作，为我们的emacs提供一个颜色模板和一个自动生成doxygen注释功能。

在env.zip文件（注意我不断地在更新env.zip文件，如果你发现什么东西找不到，那么你需要去群里重新下载一下）中，将color-theme-6.6.0.tar.gz和doxymacs-1.8.0.tar.gz 这两个文件复制到你的用户目录下（在Windows中做这件事，参见上面配置shell的.bashrc），然后进入Cygwin，确认在~下已经能看到它们：

cd ~

ls

先解压：

tar -zxvf color-theme-6.6.0.tar.gz

tar -zxvf doxymacs-1.8.0.tar.gz

你会看见用户目录下多出两个文件夹：color-theme-6.6.0和doxymacs-1.8.0。对emacs的外挂可以选择安装在emacs单独的外挂目录下，即用户目录下的.emacs.d文件夹。这是一个隐藏文件夹，我们这里将emacs的配色模板安装在这里，因为这个完全是emacs的一个外挂功能。在用户目录下，将color-theme-6.6.0移动到.emacs.d下：

cd ~

mv color-theme-6.6.0 .emacs.d

这是一个用lisp语言写的外挂，不需要编译，直接就能用。第二个外挂，主要功

能是在emacs中配合doxygen，增加一些快捷键自动产生符合doxygen要求的注释形式。主要参见表1。这个外挂包是用C语言编写的，所以需要编译安装。安装前必须确认gcc和make都应经正确安装。

命令	中文解释
C-c d ?	查找当前鼠标点下的符号的文档。
C-c d r	重新扫描tags文件。
C-c d f	为函数插入Doxygen注释。
C-c d i	为文件插入Doxygen注释。
C-c d ;	为当前成员插入Doxygen注释。
C-c d m	插入多行注释。
C-c d s	插入单行注释。
C-c d @	插入环绕当前区域的注释。

表 1: doxymacs的快捷键。

首先将之前已经解压的doxymacs-1.8.0移动到/usr/local下：
cd ~
mv doxymacs-1.8.0 /usr/local
然后进入此目录：
cd /usr/local/doxymacs-1.8.0
先检查配置：
./configure
然后开始编译：make
很快就出现第一个错误：
doxymacs_parser.c:35:10: 致命错误：libxml/parser.h：No such file or directory
#include <libxml/parser.h>

C语言编译和C++一样，需要正确的头文件（.h），所以需要检查parser.h的实际位置：
cd /usr/include
ls libxml*
我们发现确实找不到和libxml有关的头文件。所以需要安装。重新启动setup-x86_64.exe，删掉了也没关系，重新从env.zip中获取也行。现在搜索libxml，你会看见我们已经安装了libxm2，这是libxml的一个可执行版本，但并没有提供相关头文件。一般一个包的相关头文件提供在带dev后缀的包中，这里就是：libxml2-devel
注意选择版本号和上面的libxml2一致。然后下一步到底。再回到Cygwin，重新检查一下libxml：

cd /usr/include
ls libxml*
结果出现一个libxml2，进去看看：
cd libxml2
ls
结果还有一个libxml，再进去：
cd libxml
ls
这时我们发现我们需要的parser.h就在里面。它的实际位置是：
pwd
看到：
/usr/include/libxml2/libxml 按照Linux的惯例，在/usr/include文件夹下的头文件，是可以直接include的，现在include的文件是libxml/parser.h，和实际位置差了一个libxml2。解决的办法是增加一个硬链接：
cd /usr/include
ln -s libxml2/libxml .
这相当于一个别名，现在通过/usr/include/libxml也可以访问到parser.h。回到doxymacs安装目录，重新编译：
cd /usr/local/doxymacs-1.8.0
make

make是自动编译管理，它如果因为某个错误中断了，一点修复再make，可以自动继续下去，不必人工干涉。然而现在错误似乎更多了……找到错误的第一条：
doxymacs_parser.o：在函数‘main’中：
/usr/local/doxymacs-1.8.0/c/doxymacs_parser.c:514：对‘xmlCheckVersion’未定义的引用
这个的问题主要是doxymacs已经太老了，它引用的库是libxml，而现在这个库已经改名为libxml2，因此也需要做手工调整。检查make的具体过程，你会发现它曾经：
make[1]: 进入目录“/usr/local/doxymacs-1.8.0/c”
gcc -Wall -Werror -fexpensive-optimizations -fomit-frame-pointer -g -O2 -o doxymacs_parser.exe doxymacs_parser.o
我们重复这个过程：
cd /usr/local/doxymacs-1.8.0/c
gcc -Wall -Werror -fexpensive-optimizations -fomit-frame-pointer -g -O2 -o doxymacs_parser.exe doxymacs_parser.o
这次看到了一样的错误，所以我们定位了错误就是在将doxymacs_parser.o链接

为doxymacs_parser.exe时产生的。而具体的原因时引用了错误的libxml。修改的方式是：
gcc -Wall -Werror -fexpensive-optimizations -fomit-frame-pointer -g -O2 -o doxymacs_parser.exe doxymacs_parser.o -lxml2
这里人工指定使用库libxml2。根据Linux规定，-lxml2的含义就是引用库文件libxml2.so，这个文件应该在/usr/lib下，我们看一下：
ls /usr/lib
我们会发现有一个文件叫：libxml2.dll.a，这是一个Cygwin在Windows下的替代版本。我们一样可以用-lxml2来引用。刚才那句编译命令仍然还有错误（试图编译一个很旧的包就会有这种问题）：
/usr/local/doxymacs-1.8.0/c/doxymacs_parser.c:672：对‘FreeHash’未定义的引用
这次这个错误更厉害，它是由于C语言升级造成的对老的C语法的不兼容。具体修改方法则是要直接改源码：
emacs doxymacs_parser.c
找到：
inline void FreeHash(void)
（在emacs中查找的办法是：C-s，然后输入关键字，比如FreeHash回车。）
上面这句话应该在99行。改成
void FreeHash(void)
（就是把inline去掉。）同样的问题还有403行
inline void FreeCompletionList(void)
改成
void FreeCompletionList(void)
第345行
inline int OutputCompletionList(void)
改成
int OutputCompletionList(void)

现在需要彻底重新编译一下doxymacs，回到doxymacs的安装目录：
cd /usr/local/doxymacs-1.8.0
重新编译：
make clean
make
会出现上面出现过的错误，同样的处理手段：
cd c
gcc -Wall -Werror -fexpensive-optimizations -fomit-frame-pointer -g -O2 -o doxymacs_parser.exe doxymacs_parser.o -lxml2

这次终于没有错误了。再回到安装目录，继续make：
cd ..
make
这次正常结束，安装完成。上面这段手工调整，需要一定的对C语言机制的理解。大家仔细体会一下。

我们这一章一直在安装emacs的两个扩展包，现在完成它。将env.zip中.emacs文件复制到Cygwin的用户目录，重新启动emacs，然后你就拥有和我上课一样的环境了。大家自己探寻一下，比如当你在文档中已经打过include，你要再打include，当你打了inc然后按一下TAB按钮看看。要完全实现上述过程对新手有一定挑战性。但建议大家尝试一下。会有收获。对Cygwin的配置我们先暂时告一段落，接下去的课程讲述时我会使用上述配置环境。

2 虚拟机VirtualBox以及Ubuntu 18.04的安装和配置

我们推荐的第二套配置方案是在Oracle公司出品的免费虚拟机软件VirtualBox中安装Ubuntu 18.04。这么做的好处是可以接触到一个完整版的Ubuntu，并且为逐步过渡到Linux下工作提供了条件（我当年也是这么转过去的）。相比Cygwin，正确安装和配置虚拟机，需要用户对计算机硬件构成有较多的了解，同时由于它是完全系统的Ubuntu，因此也要求用户有一定的系统管理能力。市面上主流的虚拟机有VMWare公司的VMPlayer和VMWorkStation；微软公司的Hyper-V；还有就是Oracle 公司的VirtualBox。它们的原理大同小异，都是在操作系统中重新构建一层虚拟的硬件接口，使其表现为和真正的硬件一样（对一个硬件而言，本质上的事情就是数据的输入和输出，从抽象的概念看，硬件和文件是同一个东西，在Unix/Linux系统下，这二者确实是不加区分的），然后我们可以在这个平台上继续安装独立的操作系统。这么做的目的主要是为开发和学习提供特殊平台，可以想象这样的虚拟系统，效率和速度是非常低下的。在上一章已经讲述过的shell环境的设置和emacs安装和配置，在所有Linux环境下都是通用的（包括苹果的Mac OS终端），因此本章不再重复。本章主要介绍如何虚拟机和Ubuntu系统的一些基本配置和使用方法。

2.1 VirtualBox虚拟机的安装

首先登录VirtualBox的官方网站：
https://www.virtualbox.org