常用工具：

软件包管理工具：yum

查看软件包：yum list　yum list installed;　yum list |grep gdb

安装软件包：

yum install package_name

yum install　lrzsz

yum install　gcc gcc-c++

yum install　gdb

yum install　git

移除软件包：

yum remove package_name

写代码相关工具：编辑器vim/编译器gcc/g++/调式器gdb

编辑器：**vim**

1、模式：12种，常见的有三种：

插入模式:主要是使用命令进行文件内容操作

普通模式：插入数据

底行模式:用于文件内容的保存和退出

所有模式都是围绕普通模式切换的

vim打开一个文件默认就是处于普通模式

模式切换：

由普通模式切换到插入模式：

i：从光标所在字符开始插入

I：光标移动到行首，开始插入

a：光标向后移动一个字符，开始插入

A：光标移动到行尾，开始插入

o：在光标所在行下行添加新行，开始插入

O：在光标所在行上行添加新行，开始插入

由插入模式切换到普通模式：ESC

由普通模式切换到低行模式：：

:w 保存

:q 退出

:wq 保存并退出

:q! 强制退出不保存

由低行模式切换到普通模式：ESC

普通模式下常见指令：

光标移动：hjkl 左上下右；

ctrl+f/b 向上/下翻页；

gg/G 跳转到文档第一行/最后一行；

w/b单词向右走，单词向回走；

增删改查：yy/nyy复制/复制n行；

p粘贴：向光标所在行下方进行粘贴；

P：向光标所在行上方进行粘贴；

dd：剪切光标所在行内容

ndd删除：从光标所在行开始，向下剪切n行

其他操作：ctrl+r：撤销的恢复(反撤销)；

x:删除光标所在位置的字符；

u撤销；

gg=G:全文对齐

编译器：**gcc/g++**

将用户所写的高级语言代码解释成为机器可识别指令；

gcc main.c

编译阶段：

预处理：展开所有代码 gcc-E（只进行预处理）

编译：进行语义语法纠错，若无错误，则将程序解释为汇编语言 gcc-S

汇编：将汇编语言解释成为二进制机器指令 gcc-c

链接：将所有.o文件和库文件进行打包，最总生成一个可执行程序

库文件：大佬们封装的函数所打包的一个代码文件

链接方式：

动态链接：链接的是动态库

在可执行程序添加库中的函数符号信息表，生成可执行程序比较

小，因为并没有把函数的实现直接加入到可执行程序中，因此运行

程序的时候需要依赖动态库的存在，但是在内存中多个程序可以共

用相同的库，降低了内存中的代码冗余

静态链接：链接的是静态库

在生成可执行程序时，直接将库中的所有代码实现都写入到了可执

行程序中，生成的可执行程序比较大，在运行程序时，若多个程序

都是静态链接了同一个库文件，会在内存中造成代码冗余，但是静

态链接的程序运行的是时候不需要依赖库的存在

gcc默认链接方式：动态链接，并且在生产可执行程序的时候默认链接了标准c库

调试器：gdb

调式的前提：生成一个debug版本的程序

gcc默认生成的是release版本的程序，因此需要在编译程序的时候就使用gcc -g选项生成debug程序（向可执行程序中添加调试符号信息）

程序调试：

1、**gdb加载程序符号信息**　gdb ./main　gdb->file main

2、**流程控制指令**

run：直接运行程序

start：开始逐步调试

list：查看调试行附近行代码（list file：line）

until“直接运行到指定行（until file:line）

next：下一步，直接运行函数

step：下一步，跟踪进入函数

continue:从当前位置开始继续运行程序

3、**断点相关指令**

break：打断点

info break：查看断点信息

delete breakid：删除指定断点

break file:line 给指定文件指定行打断点

break func_name 给指定函数打断点

watch variable_name(变量名称)：给变量打断点（变量监控）

4、**其他指令**

quit：退出

backtrace：查看调用栈信息

print：打印变量的值

项目管理工具：项目构建工具(make/Makefile)/项目的版本管理工具git

Makefile：普通的文本文件-------记录项目的构建流程规则

make：Makefile解释器--逐行解释Makefile中的项目构建规则，完成项目的构建

Makefile的编写规则：

目标对象：依赖对象

\t：要执行的指令

目标对象：要生成的可执行程序名称

依赖对象：源码文件。通过与目标对象的最后一次修改时间判断，目标对象是否需要重新生成

预定义对象：

$^ ： 表示所有的依赖对象

$@ ： 表示目标对象

$< ： 表示第一个依赖对象

. PHONY：声明伪对象----不管对象是否最新，是否存在，每次都需要

重新生成(通常用于生成clean)

make的解释执行规则：

1、make在Makefile中只寻找第一个目标对象进行生成，生成完毕则退出

2、在生成目标对象之前，先判断依赖对象是否需要生成，（先找依赖对象的生成规则，生成依赖对象，再去生成目标对象）

wildcard：获取当前路径下以.c结尾的文件名

patsubst：字符替换，将变量内容中的后缀名，从.c修改为.o赋值给obj

项目的版权管理工具：git---分布式管理工具

从服务器上克隆一个仓库 git clone http://....

添加修改的文件 git add ./*

进行本地提交 git commit -m "本次提交的备注信息"

同步到服务器 git push origin master

1、修改sudoers配置----临时为当前用户操作赋予一下管理员权限

/ect/sudoers

修改配置文件的步骤：

1、su root

2、chmod u+w /etc/sudoers

3、vim /etc/sudoers

4、： 90

跳转到90行附近后，添加当前用户信息

5、： wq 保存退出

2、编写一个最简单的进度条程序：

\n换行符的作用：数据换行，针对标准输出的时候，还有刷新缓冲区

\r回车符的作用：让光标移动到起始位置