1. vim编辑器：

1.1 Vim的几种模式：

1.1.1 命令模式：

一般命令模式、底部命令模式：

可以使用快捷键命令，或按:输入命令行。

1.1.2 插入模式：

可以输入文本，在正常模式下，按i、a、o等都可以进入插入模式。

1.1.3 可视模式（visual mode）：

正常模式下按v可以进入可视模式， 在可视模式下，移动光标可以选择文本。

按V进入**可视行模式**， 总是整行整行的选中。

ctrl+v进入可视块模式。

```
viw   选中单词
vis   选中整段
vi(   选中括号内的所有字符
```

1.2 最常用的命令：

1.2.1 进入编辑模式：

```
a    在当前光标后面插入
A    在行尾插入
i    在当前光标插入
I    在行首插入
o    在下一行插入
O    在上一行插入

ciw       删除当前一个单词，并进入插入模式
ct"       删除双引号中间的内容，并进入插入模式
ctf       删除从当前位置到下一个f字符之间的字符
```

1.2.2 自动填充：

在输入模式下输入部分字符然后输入
ctrl +x 和 ctrl + n 用正在编辑的文件的内容作为关键字补齐。
ctrl +x 和 ctrl + f 以当前目录内的文件名作为关键字补齐。
ctrl +x 和 ctrl + o 以拓展名作为语法填充，以vim内建的关键词补齐。（必须要写对文件拓展名，比如.html,.php等等）

1.2.3 光标定位：

A|end|$ 将光标定位到行尾
0|home|^ 将光标定位到行首

gg 将光标定位到文件第一行
G 将光标定位到最后一行
w 前移一个单词，光标停在下一个单词开头；
e 前移一个单词，光标停在下一个单词末尾；
b 后移一个单词，光标停在上一个单词开头；

nG 到文件第n行。
:n 移动到第n行。

fm 快速定位到当前行，光标之后的第一个m字符处（通常在行首使用），如果有多个相同字符，可以使用分号；进行下一个选择。
Fm 同上，只是反向查找。

1.2.4 复制/剪切/粘贴：

yy 复制整行
p(小写) 粘贴（在下一行粘贴）
P（大写) 粘贴（在上一行粘贴）
y$|y+end 从光标当前位置复制到行尾。
y0 从光标当前位置复制到行首。

d$|D|d+end 删除（剪切）当前位置到行尾的内容。
d0|d+home 删除（剪切）当前位置到行首的内容。
[n] dd: 删除（剪切）1(n)行。
:m,nd 剪切m行到n行的内容。
dgg: 剪切光标以上的所有行。
dG: 剪切光标以下的所有行。
daw和das：剪切一个词和剪切一个句子，即使光标不在词首和句首也没关系

1.2.5 替换（一般命令模式）：

```
r: 替换光标处的字符，同样支持汉字。
R: 进入替换模式，按esc回到正常模式。
```

1.2.6 替换（底部命令模式）：

```
:s/old/new        用new替换当前行第一个old。
:s/old/new/g          用new替换当前行所有的old。
:n1,n2s/old/new/g         用new替换文件n1行到n2行所有的old。
:%s/old/new/g         用new替换文件中所有的old。
:%s/^/xxx/g       在每一行的行首插入xxx，^表示行首。
:%s/$/xxx/g
在每一行的行尾插入xxx，$表示行尾。

所有替换命令末尾加上c，每个替换都将需要用户确认。 如：%s/old/new/gc
```

1.2.7 文档保存/退出：

```
:wq       保存并退出。
ZZ        保存并退出。
:x        保存并退出

:q        文档未编辑时，可以直接退出。
:q!       文档编辑过了，强行退出，不保存。

:w        保存修改。
:w file2      将修改的文件保存到另外一个文件。
:r  file3     读取当前目录下的file3内容到当前文档。

:e!       一个文档修的地方太多了，最后自己都不知道哪里改对了，哪里改错了，可以使用重新加载文档，丢弃已做的改动。



:Sex          水平分割一个窗口，浏览文件系统，等同vim -o file1 file2；
:Vex          垂直分割一个窗口，浏览文件系统,等同于vim -O file1 file2；

ctrl +w   连按两次w,可实现两个窗口的切换
%     快速匹配括号
```

```
gf          打开以光标所在字符串为文件名的文件。然后使用ctrl + ^ 可以再次回到源文件。
```

1.2.8 基本排版:

```
<          先输入v进入可视化模式，向左缩进一个shiftwidth
>          先输入v进入可视化模式，向右缩进一个shiftwidth
:ce(nter) 本行文字居中
:le(ft) 本行文字靠左
:ri(ght) 本行文字靠右
```

1.2.9 编辑多个文件:

```
vim a.txt b.txt c.txt -p
* 使用:next(:n)编辑下一个文件。
* :2n 编辑下2个文件。
* 使用:previous或:N编辑上一个文件。
* 使用:wnext，保存当前文件，并编辑下一个文件。
* 使用:wprevious，保存当前文件，并编辑上一个文件。
```

1.2.10 改变大小写:

```
shift  ~: 反转光标所在字符的大小写。
可视模式下的U：把选中的文本变为大写或小写。
v 可视字符 ，可以选择多个字符 ，再按U ，可以将小写切换为大写
V 可视行，可以选择多行，再按u，可以将大写切换为小写
```

1.2.11 文件加解密:

```
vim -x file: 开始编辑一个加密的文件。
:X                    为当前文件设置密码。
:set key=        去除文件的密码。
```

1.2.12 折叠:

```
zf      创建折叠的命令，可以在一个可视区域上使用该命令，可缩减空间方便阅读；
zo      打开折叠的文本
zc      收起折叠；
zd      删除当前行的折叠，删除之后就不能再次折叠了；
za      打开/关闭当前折叠；
zfap        折叠光标所在的段；
```

1.3 常见设置：

1.3.1 智能缩进：

```
:set si

取消：
:set nosi
```

1.3.2 自动对齐：

```
:set ai

取消：
:set noai
```

1.3.3 显示行号：

```
:set nu

取消：
:set nonu
```

1.3.4 语法高亮：

```
:syntax on

取消：
:syntax off
```

1.3.5 显示换行符：

```
:set list

取消：
:set nolist
```

1.3.6 设置文件格式：

```
设置为windows格式：
:set fileformat=dos
设置为unix格式：
:set fileformat=unix

简写：
:set ff=unix|dos
```

1.3.7 增量搜索：

```
:set incsearch
```

1.3.8 显示命令：

```
:set showcmd
```

1.3.9 更换主题：

```
显示当前使用的主题
:colorscheme

显示所有可用的主题
:colorscheme 空格 + ctrl +d

更换主题颜色：
:colorscheme  morning
```

2. linux软件管理概念：

Linux整个体系的关键不在于系统本身，而是在于可以基于Linux系统去安装和配置企业中相关的软件、数据及应用程序，所以对软件的维护是运维工程师的重中之重。

3. 软件管理的三种方式：

- rpm管理：
- yum管理：
- 源码方式管理：

Linux软件包管理大致可分为二进制包、源码包，使用的工具也各不相同。

Linux常见软件包分为两种，分别是源代码包（Source Code）、二进制包（Binary Code），源代码包是没有经过编译的包，需要经过GCC、C++编译器环境编译才能运行，二进制包无需编译，可以直接安装使用。

通常而言，可以通过后缀简单区别源码包和二进制包，例如.tar.gz、.zip、.rar结尾的包通常称之为源码包，以.rpm结尾的软件包称之为二进制包。真正区分是否为源码还是二进制还得基于代码里面的文件来判断，例如包含.h、.c、.cpp、.cc等结尾的源码文件，称之为源码包，而代码里面存在bin可执行文件，称之为二进制包。

CentOS操作系统中有一款默认软件管理的工具，红帽包管理工具（Red Hat Package Manager，RPM）。
使用RPM工具可以对软件包实现快速安装、管理及维护。RPM管理工具适用的操作系统包括：CentOS，RedHat，Fedora，SUSE等，RPM工具常用于管理.rpm后缀结尾的软件包。

3.1 RPM管理：

3.1.1 rpm工作原理：

当我们使用rpm工具安装软件包时，它会首先找到软件包中的一个记录文件，该文件记录了这个软件安装时需要的依赖包，如果包已经存在，则可以顺利安装，如果不存在，则提示缺少相应的依赖。

RPM优点：

- 软件已经编译打包，所以传输和安装方便，让用户免除编译。
- 在安装之前，会先检查系统的磁盘、操作系统版本等，避免错误安装。
- 在安装好之后，软件的信息都已经记录在linux主机的数据库上，方便查询、升级和卸载。

RPM缺点：

- 软件包安装的环境必须与打包时的环境一致。

- 必须安装了软件的依赖软件。

3.1.2 rpm常用命令：

```
### 安装包，如果依赖包没装，则会提示检测依赖失败：
rpm -ivh tomcat-7.0.76-2.el7.noarch.rpm

# 忽略依赖，强制安装：
rpm -ivh --nodeps  tomcat-7.0.76-2.el7.noarch.rpm

# 卸载包：
rpm -e nginx

# 查询某个包是否有安装：
rpm -q nginx|httpd

# 查询当前系统所有已安装的包：
rpm  -qa

# 统计系统所有已安装的包：
rpm -qa | wc -l

# 查询已安装包的版本信息：
rpm -qi nginx

# 查询已经安装的包的所有文件路径：
rpm -ql nginx

# 查询已安装的包的配置文件路径：
rpm -qc nginx

# 查询已安装的包文档路径：
rpm -qd  nginx

# 根据指定的文件(一般指的是命令)，反向查找软件包：
rpm -qf `which netstat`

# 查看软件包在安装过程中执行脚本：
rpm -q --scripts httpd

# 查询一个未安装的包的文件分布路径：
```

```
rpm -qpl tomcat-7.0.76-2.el7.noarch.rpm
```