一. find 文件查找

1. 查找txt和pdf文件：find . \(-name "*.txt" -o -name "*.pdf" \) -print
2. 查找所有非txt文件：find . ! -name "*.txt" -print
3. 指定深度搜索：find . -maxdepth 1 -type f
4. 按类型搜索：find . -type d -print //只列出所有目录
   -type f 文件
   -type l 符号链接
5. 按时间搜索
   -atime 访问时间 (单位是天，分钟单位则是-amin，以下类似)
   -mtime 修改时间 （内容被修改）
   -ctime 变化时间 （元数据或权限变化）
   最近7天被访问过的所有文件：find . -atime 7 -type f -print
6. 按大小搜索：find . -type f -size +2k //寻找大于2k的文件
7. 按权限查找：find . -type f -perm 644 -print //找具有可执行权限的所有文件
8. 按用户查找：find . -type f -user weber -print // 找用户weber所拥有的文件
9. 删除当前目录下所有swp文件：find . -type f -name "*.swp" -delete
10. 执行动作（强大的exec）
    find . -type f -user root -exec chown weber {} \; //将当前目录下的所有权变更为weber
    注：{}是一个特殊的字符串，对于每一个匹配的文件，{}会被替换成相应的文件名；
11. 将找到的文件全都copy到另一个目录：find . -type f -mtime +10 -name "*.txt" -exec cp {} OLD \;
12. -print的定界符
    默认使用'\n'作为文件的定界符；
    -print0 使用'\0'作为文件的定界符，这样就可以搜索包含空格的文件

二. Grep文本搜索

grep match_patten file // 默认访问匹配行
常用参数
1). -o 只输出匹配的文本行 **VS** -v 只输出没有匹配的文本行
2). -c 统计文件中包含文本的次数：grep -c "text" filename
3). -n 打印匹配的行号
4). -i 搜索时忽略大小写
5). -l 只打印文件名

1. 在多级目录中对文本递归搜索(程序员搜代码的最爱）：grep "class" . -R -n
2. 匹配多个模式：grep -e "class" -e "vitural" file

三. xargs命令行参数

xargs 能够将输入数据转化为特定命令的命令行参数；这样，可以配合很多命令来组合使用。比如grep，比如find；

xargs参数说明
1). -d 定义定界符 （默认为空格 多行的定界符为 \n）
2). -n 指定输出为多行
3). -0 指定\0为输入定界符
4). -I {} 指定替换字符串，这个字符串在xargs扩展时会被替换掉,用于待执行的命令需要多个参数时
   cat file.txt | xargs -I {} ./command.sh -p {} -1

例如统计程序行数：find source_dir/ -type f -name "*.cpp" -print0 |xargs -0 wc -l

四. Sort排序

-n 按数字进行排序 VS -d 按字典序进行排序
-r 逆序排序
-k N 指定按第N列排序
-b 忽略前导空白字符

1. sort -nrk 1 data.txt
2. sort -bd data // 忽略像空格之类的前导空白字符

五. Uniq消除重复行

1. 消除重复行：sort unsort.txt | uniq
2. 统计各行在文件中出现的次数：sort unsort.txt | uniq -c
3. 找出重复行：sort unsort.txt | uniq -d
   可指定每行中需要比较的重复内容：-s 开始位置 -w 比较字符数

六. 用Tr进行替换

1. 通用用法
   echo 12345 | tr '0-9' '9876543210' //加解密转换，替换对应字符
   cat text| tr '\t' ' ' //制表符转空格
2. tr删除字符
   cat file | tr -d '0-9' // 删除所有数字
3. -c 求补集
   cat file | tr -c '0-9' //获取文件中所有数字
   cat file | tr -d -c '0-9 \n' //删除非数字数据
4. tr压缩字符
   tr -s 压缩文本中出现的重复字符；最常用于压缩多余的空格
   cat file | tr -s ' '

七. cut 按列切分文本

1. 截取文件的第2列和第4列：cut -f2,4 filename
2. 去文件除第3列的所有列：cut -f3 --complement filename
   -d 指定定界符：
   cat -f2 -d";" filename
3. cut 取的范围
   N- 第N个字段到结尾
   -M 第1个字段为M
   N-M N到M个字段
4. cut 取的单位
   -b 以字节为单位
   -c 以字符为单位
   -f 以字段为单位（使用定界符）

八. paste按列拼接文本

1. 拼接2个文本：paste file1 file2
2. 默认的定界符是制表符，可以用-d指明定界符：paste file1 file2 -d “,”

九. WC统计行和字符

1. 统计行数：wc -l file
2. 统计单词数：wc -w file
3. 统计字符数：wc -c file

十 . Sed文本替换

1. 首处替换：sed 's/text/replace_text/' file
2. 全局替换：sed 's/test/replace_test/g' file
   默认替换后，输出替换后的内容，如果需要直接替换原文件,使用-i
   sed -i 's/test/replace_test/g' file
3. 移除空白行：sed '/^$/d' file

十一. Awk数据流处理

1. awk脚本结构
   awk ' BEGIN{ statements } statements2 END{ statements } '
   工作方式
   1). 执行begin中语句块；
   2). 从文件或stdin中读入一行，然后执行statements2，重复这个过程，直到文件全部被读取完毕；
   3). 执行end语句块；
2. 特殊变量： NR NF $0 $1 $2
   1). NR:表示记录数量，在执行过程中对应当前行号；
   2). NF:表示字段数量，在执行过程总对应当前行的字段数；
   3). $0:这个变量包含执行过程中当前行的文本内容；
   4). $1:第一个字段的文本内容；
   5). $2:第二个字段的文本内容；

3. 打印每一行的第二和第三个字段：awk '{print $2, $3}' file
4. 统计文件的行数：awk ' END {print NR}' file
5. 累加每一行的第一个字段：echo -e "1\n 2\n 3\n 4\n" | awk 'BEGIN{num = 0; print "begin";} {sum += $1;} END {print "=="; print sum }'
6. 在awk中使用循环
   for(i=0;i<10;i++){print $i;}

十二. 迭代文件中的行、单词和字符

1. 迭代文件中的每一行
   1). while 循环法
   ```
   while read line;
   do
       echo $line;
   done < file.txt
   ```
2. awk法：cat file.txt| awk '{print}'
3. 迭代一行中的每一个单词：for word in $line; do echo $word; done