**rm 123.txt**删除文件

**通配符**

**\*代替n位**

**？代替一位**

**[]限定范围**

wcyclay@ubuntu:~/Desktop/wcy$ ls

123.txt 143.txt 183.txt 243.txt wcy.txt

133.txt 153.txt 23 253.txt

wcyclay@ubuntu:~/Desktop/wcy$ ls 2\*

23 243.txt 253.txt

wcyclay@ubuntu:~/Desktop/wcy$ ls 2?

23

wcyclay@ubuntu:~/Desktop/wcy$ ls 1[12345]3.txt

123.txt 133.txt 143.txt 153.txt

wcyclay@ubuntu:~/Desktop/wcy$ ls 1[1-5]3.txt

123.txt 133.txt 143.txt 153.txt

**重定向**

**基本IO**

**编辑**

**cmd > file 把 stdout 重定向到 file 文件中；**

**cmd >> file 把 stdout 重定向到 file 文件中(追加)；**

**cmd 1> file 把 stdout 重定向到 file 文件中；**

**cmd > file 2>&1 把 stdout 和 stderr 一起重定向到 file 文件中；**

**cmd 2> file 把 stderr 重定向到 file 文件中；**

**cmd 2>> file 把 stderr 重定向到 file 文件中(追加)；**

**cmd >> file 2>&1 把 stdout 和 stderr 一起重定向到 file 文件中(追加)；**

**cmd < file >file2 cmd 命令以 file 文件作为 stdin，以 file2 文件作为 stdout；**

**cat <>file 以读写的方式打开 file；**

**cmd < file cmd 命令以 file 文件作为 stdin；**

**cmd << delimiter Here document，从 stdin 中读入，直至遇到 delimiter 分界符。**

**进阶IO**

**编辑**

**>&n 使用系统调用 dup2复制文件描述符 n 并把结果用作标准输出；**

**<&n 标准输入复制自文件描述符 n；**

**<&- 关闭标准输入（键盘）；**

**>&- 关闭标准输出；**

**n<&- 表示将 n 号输入关闭；**

**n>&- 表示将 n 号输出关闭；**

**上述所有形式都可以前导一个数字，此时建立的文件描述符由这个数字指定而不是缺省的 0 或 1。如：**

**... 2>file 运行一个命令并把错误输出(文件描述符 2)定向到 file。**

**... 2>&1 运行一个命令并把它的标准输出和输出合并。(严格的说是通过复制文件描述符 1 来建立文件描述符 2 ，但效果通常是合并了两个流。)**

**我 们对 2>&1详细说明一下 ：2>&1 也就是 FD2=FD1 ，这里并不是说FD2 的值 等于FD1的值，因为 > 是改变送出的数据通道，也就是说把 FD2 的 “数据输出通道” 改为 FD1 的 “数据输出通道”。如果仅仅这样，这个改变好像没有什么作用，因为 FD2 的默认输出和 FD1的默认输出本来都是 monitor，一样的！ 但是，当 FD1 是其他文件，甚至是其他 FD 时，这个就具有特殊的用途了。请大家务必理解这一点。**

**exec 1>outfilename # 打开文件outfilename作为stdout。**

**exec 2>errfilename # 打开文件 errfilename作为 stderr。**

**exec 0<&- # 关闭 FD0。**

**exec 1>&- # 关闭 FD1。**

**exec 5>&- # 关闭 FD5。**