1. **实验内容**

使用fork(), exec(), dup2(), pipe(), open(), wait()等系统调用编写C语言程序完成与下列shell命令等价的功能。

grep -v usr < /etc/passwd | wc –l > r.txt; cat r.txt

提示：为简化编程，不需要用strtok断词，直接用execlp实现能达到shell命令相同功能的程序即可。例如：execlp("grep", "grep", "-v", "usr", 0);

1. **实验步骤**
2. 分析并尝试执行所给命令。

|  |  |
| --- | --- |
| 命令 | 含义 |
| grep -v usr | 将输入中含有usr字段的内容剔除，并输出其余内容 |
| < /etc/passwd | 将位于”/etc/passwd”的文件作为输入传递给grep命令 |
| | wc –l | 通过Linux的管道功能，将grep命令的输出作为wc的输入进行传递，wc参数为”-l”，用于统计行数 |
| > r.txt; | 将wc命令的输出保存至默认当前目录下的”r.txt”内，并结束命令分析 |
| cat r.txt | 通过cat命令，将”r.txt”的内容默认输出至屏幕 |



观察到结果符合分析。

1. 依照题意编写C程序源代码，并编译。

代码附后。

1. 尝试运行，观察输出结果是否符合预期。
2. **源代码**

#include <sys/types.h>

#include <unistd.h>

#include <sys/wait.h>

#include <fcntl.h>

#include <stdlib.h>

int main()

{

int sv;

int in = open("/etc/passwd", O\_RDONLY);

int out = open("r.txt", O\_WRONLY);

int pfd[2];

pipe(pfd);

if(fork() == 0) //grep

{

dup2(in, 0);

dup2(pfd[1], 1);

close(pfd[1]);

close(pfd[0]);

execlp("grep", "grep", "-v", "usr", NULL);

exit(0);

}

else if(fork() == 0) //wc

{

dup2(pfd[0], 0);

dup2(out, 1);

close(pfd[1]);

close(pfd[0]);

execlp("wc", "wc", "-l", NULL);

exit(0);

}

close(pfd[1]);

close(pfd[0]);

wait(&sv);

wait(&sv);

//cat

execlp("cat", "cat", "r.txt", NULL);

return 0;

}

1. **实验结果**



结果符合预期功能。

1. **实验总结**

通过本次实验的编写，我对Linux中的管道功能的使用和作用理解得更为深入，学会了分析命令中的”|”的具体含义，以及如何通过程序重定向输入输出，以及程序中如何通过无名管道在父子/兄弟进程间单向传输字节流。体会到了Linux中一切皆文件的思想。

非常感谢老师的实验设计！