1. 下载6.828 shell的源代码 sh.c ，代码主要包含两部分：解析Shell命令和运行命令，并且这里只考虑简单的命令，如下：

ls > y

cat < y | sort | uniq | wc > y1

cat y1

rm y1

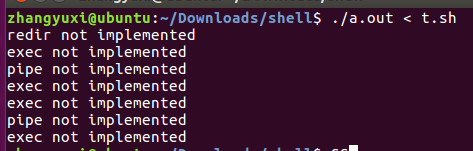
ls | sort | uniq | wc

rm y

将上面的命令保存在 t.sh 文件中作为测试脚本，以便之后使用。

使用gcc命令编译 sh.c 文件，会在同一文件夹下生成 a.out 可执行文件。

运行刚刚编译好的可执行文件：



我们需要实现的就是这几个功能。

2. 分析sh.c:

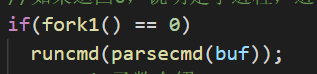


使用getcmd函数获取命令：

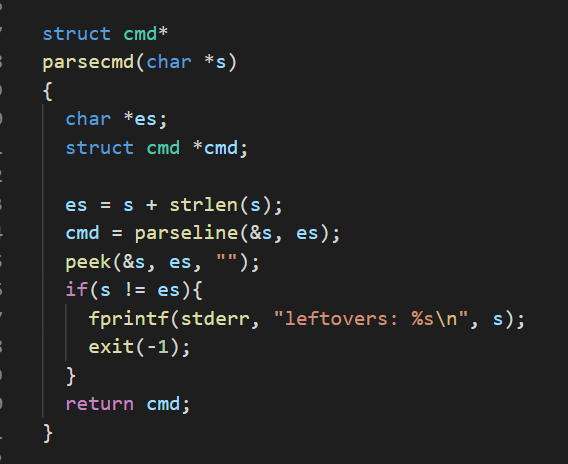


之后会进入 while 循环，首先判断如果输入的是切换目录的命令 cd 就直接执行目录切换操作，否则就要 fork() 一个子进程，进而进行处理：

函数 fork1() 会fork一个子进程，并返回父／子进程的pid。main() 函数内通过判断返回的pid来判断当前执行的是哪个进程，从而在子进程中接着执行相应的命令；父进程中使用 wait(&r) 进行阻塞，等待子进程返回后再继续执行。

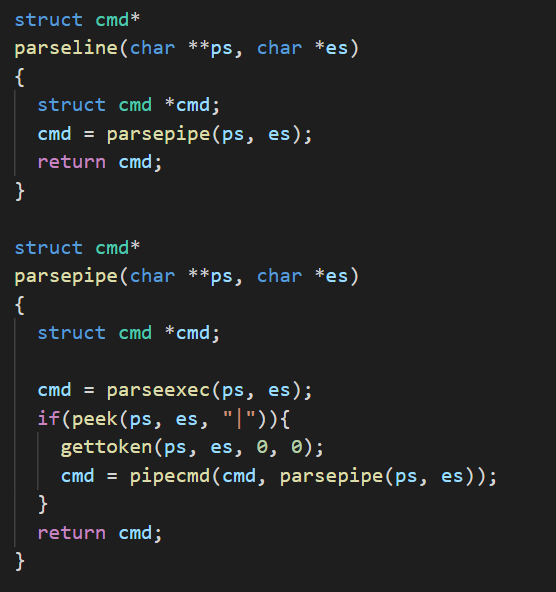


这个时候，子进程开始调用parsecmd()函数去分析输入的命令字符串：

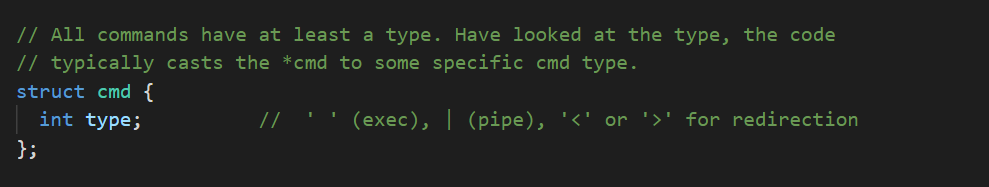


es指针指向字符串的末端，确切的说是空字符处

然后去调用 parseline(&s, es)，parseline() 就是一层简单的封装．实际核心函数是parsepipe：



返回的是一个结构体 cmd ，cmd结构体的定义如下：



可以推断parsecmd()这个函数的功能主要是判断输入的命令的种类。

函数 runcmd() 如下：





这个函数接受一个参数：结构体 cmd ，而且通过这个结构体中的 type 值进行进一步的处理。从 switch case 语句的判断条件可以看出，将命令的类型分成三类，分别是： case '' 可执行命令、 case '<' case '>' 两个输入输出重定向命令和 case '|' 管道命令。我们要做的就是补全不同类型命令里具体执行命令的代码。

3.实现可执行命令。

对于accsee()函数：

功能: 确定文件的访问权限。即，检查某个文件的存取方式，比如说是只读方式、只写方式等。如果指定的存取方式有效，则函数返回0，否则函数返回-1。

对于execv() 函数：

我们实际使用的是 execv() 函数来运行shell命令。

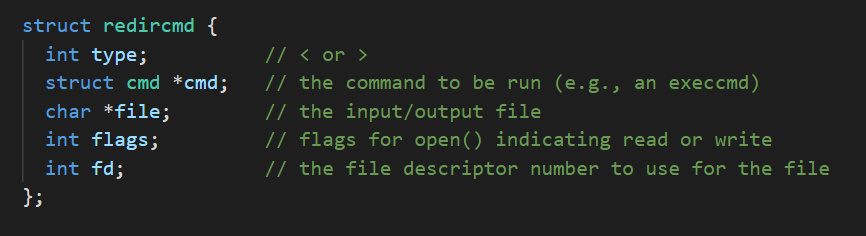
execv()用来执行参数path 字符串所代表的文件路径, 与execl()不同的地方在于execv()只需两个参数, 第二个参数利用数组指针来传递给执行文件。如果执行成功则函数不会返回, 执行失败则直接返回-1, 失败原因存于errno 中。

思路：

通过access函数判断当前目录是否存在该命令，如果不存在则到/bin目录下寻找。使用execv函数执行命令。如果有就执行，否则抛出错误。

4.实现IO重定向命令

首先看一下结构体 redircmd 的定义：



再看一下open函数：

该函数的作用是建立一个文件描述符，其他的函数可以通过文件描述符对指定文件进行读取与写入的操作。

返回值：若所有欲核查的权限都通过了检查则返回0 值, 表示成功, 只要有一个权限被禁止则返回-1.

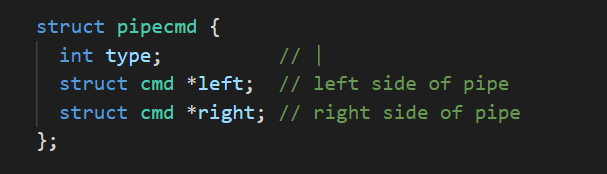
实现思路：

首先关闭重定向命令的文件描述符，然后使用open命令打开重定向文件，由于open命令会自动选择最小的文件标识符0(标准输入)或者1(标准输出)，实现重定向。

5.实现管道命令：

管道是一种把两个进程（如fork出来的父子进程）之间的标准输入和标准输出连接起来的机制，从而提供一种让多个进程间通信的方法，当进程创建管道时，每次都需要提供两个文件描述符来操作管道。其中一个对管道进行写操作，另一个对管道进行读操作。对管道的读写与一般的IO系统函数一致，使用write()函数写入数据，使用read()读出数据。

先看一下结构体 pipecmd 的定义：



管道命令的标志是符号 | ，| 的左面和右面分别是不同的命令，我们需要逐步的执行这些命令。

pip函数：

这个函数的作用是在两个进程之间建立一个管道，并生成两个文件描述符 fd[0] 和 fd[1] ，分别对应管道的读取端和写入端，这两个进程可以使用这两个文件描述符进行读写操作，即实现了这两个进程之间的通信。

若成功则返回 0 ，否则返回 -1 ，错误原因存于 errno 中。

dup函数：

这个函数的功能是复制一个现存的文件描述符。

返回值：若成功则返回一个指向相同文件的新的文件描述符（且为当前可用文件描述符中的最小值 ），失败则返回 -1 。

代码实现思路：

在父进程中创建1个无名管道，再fork 2个子进程，关闭管道左边进程的标准输出，使用dup函数复制管道输 入标识符(使1代表管道输入)，关闭管道输入输出标识符；关闭管道右边进程的标准输入，使用dup函数复制管道输处标 识符(使0代表管道输出)，最后关闭父进程管道的输入输出标识符(防止干扰)。