

Group 13 – Phase 1

Jun & Sonya

Usage of the program

We developed our own shell in C that replicates feature from the Linux commands or a program to execute including its name. We implemented single and composed commands using 0 to 3 pipes.

Example test cases and results

--- 3 pipes ---

cat words.txt | grep yasin | tee output1.txt | wc -l

```
$ cat words.txt | grep yasin | tee output1.txt | wc -l
6
```

cat words.txt | uniq | sort | head -10

```
$ cat words.txt | uniq | sort | head -10
Android
Banana
Cat
Coffee
Edamame
Hello
Hersheys
Mcdonalds
Starbucks
Sun
```

sort alphabets.txt | head -10 | tail -n 5 | tee output3.txt

```
$ sort alphabets.txt | head -10 | tail -n 5 | tee output3.txt
f
g
h
i
k
```

--- 2 pipes ---

sort words.txt | head -10 | grep 'a'

```
$ sort words.txt | head -10 | grep 'a'
```

Banana
Cat
Edamame
Mcdonalds
Starbucks

```
cat words.txt | grep yasin | wc -l
```

```
$ cat words.txt | grep yasin | wc -l
```

6

--- 1 pipe ---

```
cat alphabets.txt | tail -10
```

```
$ cat alphabets.txt | tail -10
```

l
w
q
r
z
f
u
g
n
h

```
cat words.txt | uniq
```

```
$ cat words.txt | uniq
```

```
Hello  
Mcdonalds  
Coffee  
yasin hey  
Starbucks  
Banana  
yasin  
Water  
Test  
fish  
yasin yo  
bread  
potato  
Cat  
Edamame  
yasin more  
Sun  
burger  
Hersheys  
Swimming  
Android
```

```
df | tee disk_usage.txt
```

```
$ df | tee disk_usage.txt
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
rootfs	249467900	216037344	33430556	87%	/
none	249467900	216037344	33430556	87%	/dev
none	249467900	216037344	33430556	87%	/run
none	249467900	216037344	33430556	87%	/run/lock
none	249467900	216037344	33430556	87%	/run/shm
none	249467900	216037344	33430556	87%	/run/user
tmpfs	249467900	216037344	33430556	87%	/sys/fs/cgroup
C:\	249467900	216037344	33430556	87%	/mnt/c
D:\	249470972	46119964	203351008	19%	/mnt/d

```
--- 0 pipes ---
```

```
cat alphabets.txt
```

```
$ cat alphabets.txt
```

```
b  
d  
k  
e  
m  
a  
c  
p  
i  
t  
y  
l  
w  
q  
r  
z  
f  
u  
g  
n  
h
```

```
ls -l
```

```
$ ls -l  
  
total 56  
-rwxrwxrwx 1 yaya1721 yaya1721    7 Mar 11 14:10 Makefile  
-rwxrwxrwx 1 yaya1721 yaya1721 17568 Mar 11 16:39 Phase1  
-rwxrwxrwx 1 yaya1721 yaya1721 11339 Mar 11 16:39 Phase1.c  
drwxrwxrwx 1 yaya1721 yaya1721   512 Mar 11 15:09 a  
-rwxrwxrwx 1 yaya1721 yaya1721    63 Mar 13 01:36 alphabets.txt  
-rwxrwxrwx 1 yaya1721 yaya1721   581 Mar 13 01:35 disk_usage.txt  
-rwxrwxrwx 1 yaya1721 yaya1721    32 Mar  5 16:44 file4.  
-rwxrwxrwx 1 yaya1721 yaya1721    54 Mar 13 01:24 output1.txt  
-rwxrwxrwx 1 yaya1721 yaya1721    48 Mar  5 21:59 output2.txt  
-rwxrwxrwx 1 yaya1721 yaya1721    14 Mar 13 01:30 output3.txt  
-rwxrwxrwx 1 yaya1721 yaya1721 17496 Mar 10 16:31 phase1.o  
-rwxrwxrwx 1 yaya1721 yaya1721   189 Mar  5 21:59 words.txt
```

```
man
```

```
$ man
```

```
What manual page do you want?
```

```
pwd
```

```
$ pwd
```

```
/mnt/c/Users/Sonya/Documents/GitHub/OS-Project/Phase1
```

```
touch dummy.txt
```

Create a file in the same directory called dummy.txt

```
rm dummy.txt
```

Remove the file, dummy.txt

```
ping google.com
```

```
$ ping google.com
```

```
PING google.com (216.58.207.110) 56(84) bytes of data:  
64 bytes from fjr02s04-in-fl4.1e100.net (216.58.207.110): icmp_seq=1 ttl=55 time=10.8 ms  
64 bytes from fjr02s04-in-fl4.1e100.net (216.58.207.110): icmp_seq=2 ttl=55 time=7.22 ms  
64 bytes from fjr02s04-in-fl4.1e100.net (216.58.207.110): icmp_seq=3 ttl=55 time=10.5 ms  
64 bytes from fjr02s04-in-fl4.1e100.net (216.58.207.110): icmp_seq=4 ttl=55 time=8.84 ms  
64 bytes from fjr02s04-in-fl4.1e100.net (216.58.207.110): icmp_seq=5 ttl=55 time=7.40 ms  
64 bytes from fjr02s04-in-fl4.1e100.net (216.58.207.110): icmp_seq=6 ttl=55 time=6.99 ms  
64 bytes from fjr02s04-in-fl4.1e100.net (216.58.207.110): icmp_seq=7 ttl=55 time=6.42 ms  
64 bytes from fjr02s04-in-fl4.1e100.net (216.58.207.110): icmp_seq=8 ttl=55 time=7.39 ms
```

Description of implementation

We developed our own shell in C programming language. After parsing the input from the command line, we determined how many pipes should be used in each command and called the corresponding function. Within each function, we forked child process and created pipes. Parent process should execute the last command after its child process executed the previous commands if the number of commands is larger than one. User is able to exit the program by inputting “exit” on the command line interface.