工管四乙 109370210 黃鈺凱

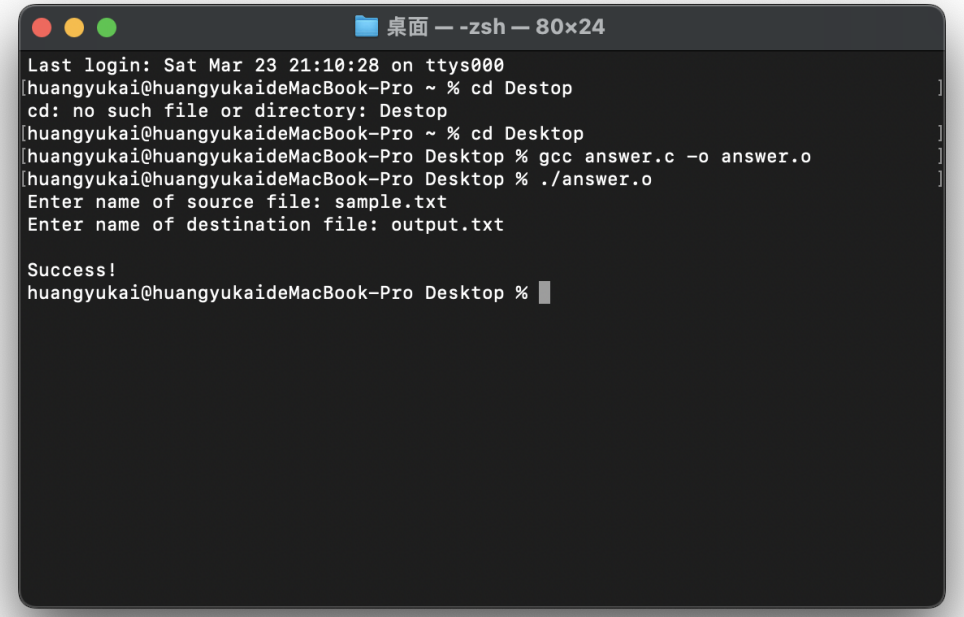
Programming problem

Q.2.24

Steps to run the program:

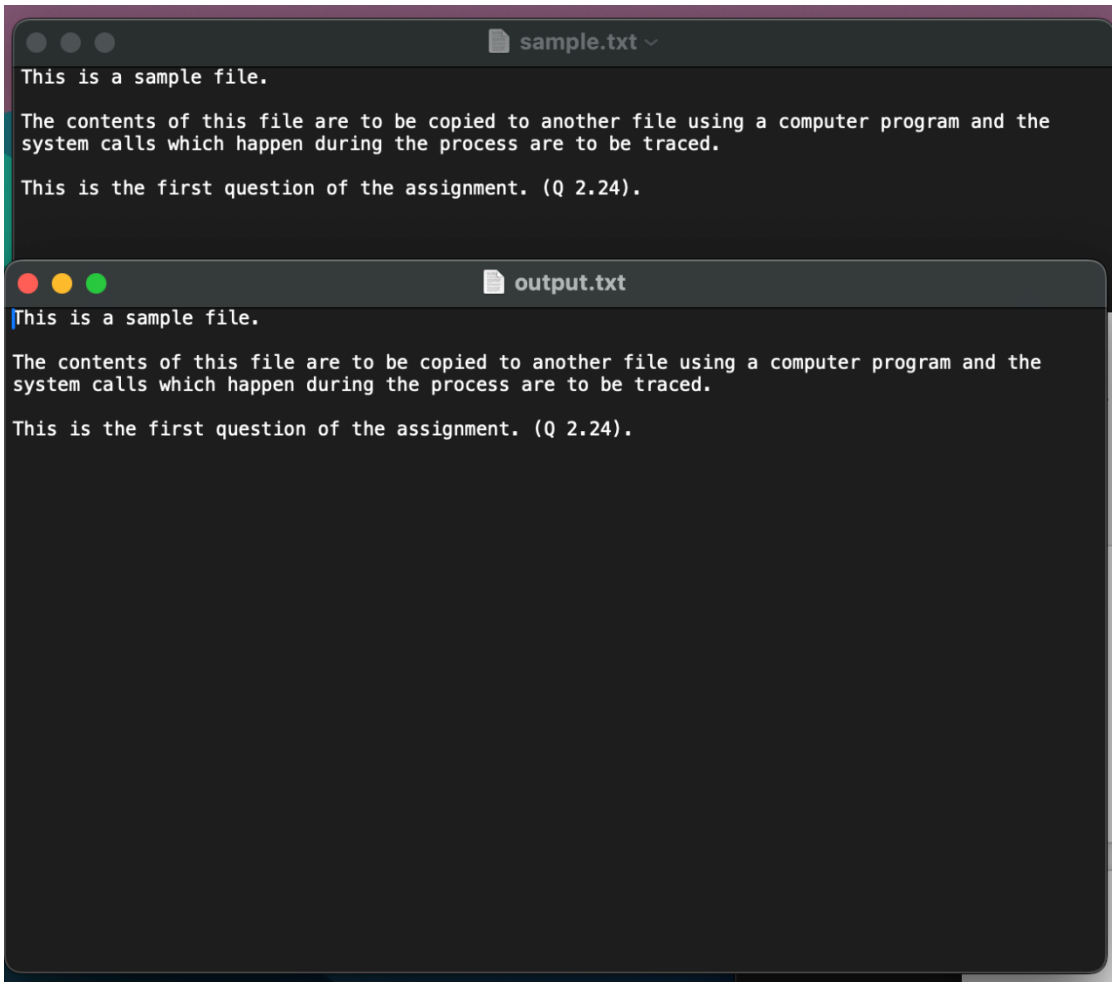
- Create a text file used as a input file (sample.txt) which will be copied to the output file (output.txt) under same directory.
- Execute ./answer.o
- The program will prompt to enter the name of the input and output file.
- Contents are copied to output.txt will display when program finished.

Snapshot:



- After running the program, command sudo dtrace ./answer.o to create a log file in which the system calls are logged to trace the system calls.
- Contents are copied to output.txt will display when program finished.

Below screenshots show that the content between input and output are same.

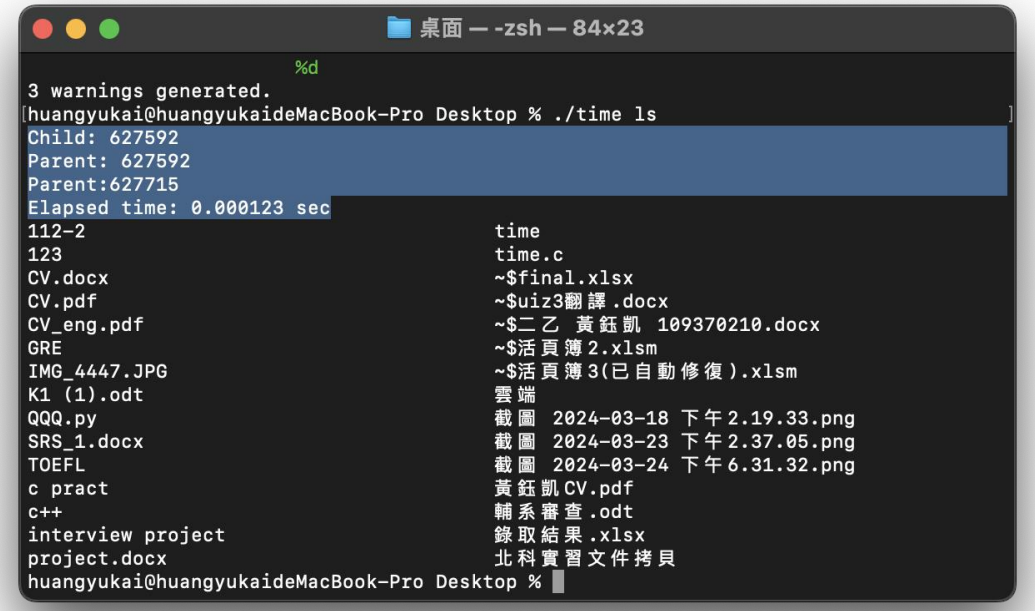


Q3.19

Steps to run the program:

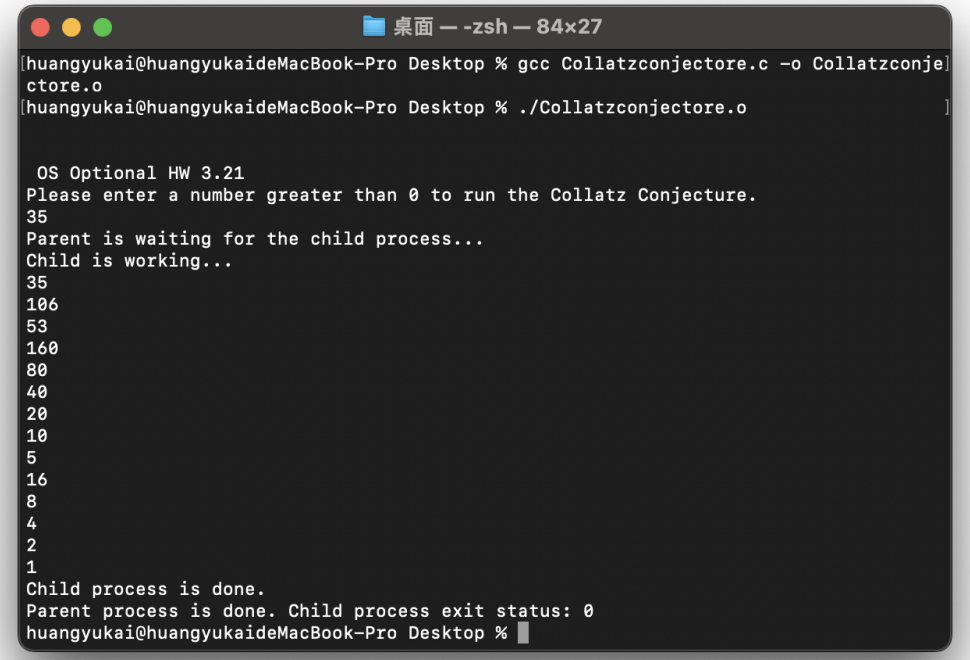
- Create a programming which could generate first version of having the child process which write the starting time and also has the second version which child write starting time into the pipe.
- Execute ./time ls
- The programming will read all of files which on Desktop
- Output will content first version child starting time, parents time and second version which parent read from the child write into the pipe
- Finally, show all <command> file in command line.

Snapshot:



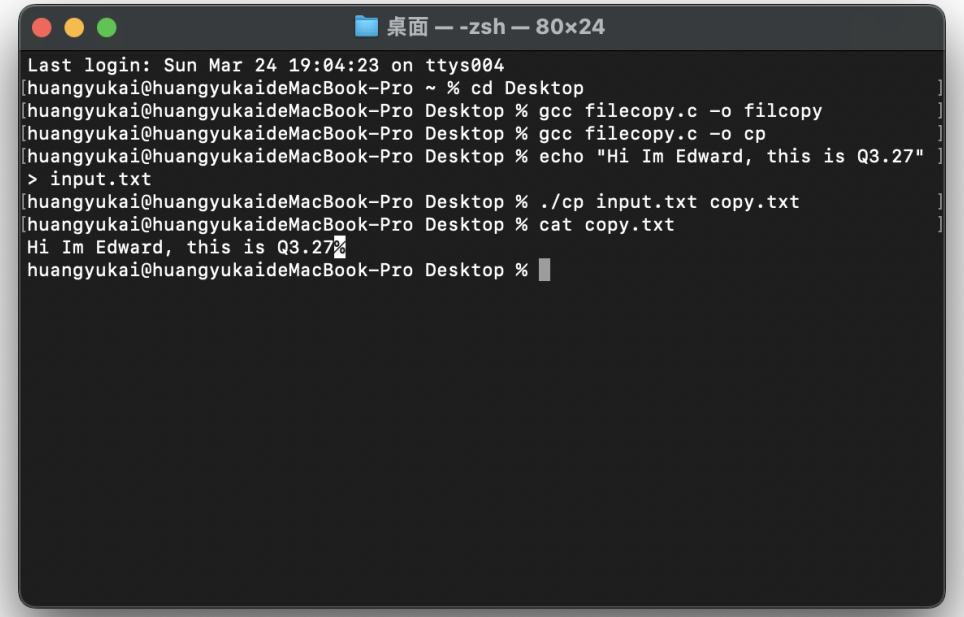
[Optional] Q.3.21

After command `./Collatzconjectore.o` , the child process will output the sequence of numbers generated from the algorithm specified by the Collatz conjecture, it checks to see if it is even or odd number and performs the appropriate equation. This repeat until it reaches 1, and then breaks out of the loop. This works because the parent and child processes have their own copies of the data.



[optional] Q.3.27

- gcc filecopy.c -o cp
- In this question, I commanded echo ”Hi Im Edward, this is Q3.27.” > input.txt to create a .txt file with content in.
- ./cp input.txt copy.txt to compile.
- Eventually, command cat copy.txt will show the content within copy.txt!



Programming Projects:

Chap. 2 – Project 1:Linux Kernel Modules

Part I:

- Design a kernel module that creates a /proc file named /proc/jiffies that reports the current value of jiffies when the /proc/jiffies file is read, such as with the command
- This kernel module jiffies.c is compiled using Makefile, thus, enter make command.
- The compilation produces several files. The file jiffies.ko represents the compiled kernel module.

```
edwall201@edwall201-VirtualBox:~/Desktop$ make
make -C /lib/modules/6.5.0-26-generic/build M=/home/edwall201/Desktop modules
make[1]: Entering directory '/usr/src/linux-headers-6.5.0-26-generic'
warning: the compiler differs from the one used to build the kernel
  The kernel was built by: x86_64-linux-gnu-gcc-12 (Ubuntu 12.3.0-1ubuntu1~22.04
) 12.3.0
  You are using:           gcc-12 (Ubuntu 12.3.0-1ubuntu1~22.04) 12.3.0
  CC [M]  /home/edwall201/Desktop/jiffies.o
/home/edwall201/Desktop/jiffies.c: In function 'proc_read':
/home/edwall201/Desktop/jiffies.c:26:5: warning: ignoring return value of 'copy_
to_user' declared with attribute 'warn_unused_result' [-Wunused-result]
   26 |     copy_to_user(usr_buf, buffer, rv);
      |     ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  MODPOST /home/edwall201/Desktop/Module.symvers
  CC [M]  /home/edwall201/Desktop/jiffies.mod.o
  LD [M]  /home/edwall201/Desktop/jiffies.ko
  BTF [M] /home/edwall201/Desktop/jiffies.ko
Skipping BTF generation for /home/edwall201/Desktop/jiffies.ko due to unavailabi
lity of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-6.5.0-26-generic'
```

Loading:

- Kernel modules are loaded using the sudo insmod jiffies.ko command
- To check whether the module has loaded, enter lsmod | grep jiffies command. And it will show jiffies. To check the contents of this message, enter cat/proc/jiffies to show.

```
edwall201@edwall201-VirtualBox:~/Desktop$ sudo insmod /home/edwall201/Desktop/ji
ffies.ko
[sudo] password for edwall201:
Sorry, try again.
[sudo] password for edwall201:
edwall201@edwall201-VirtualBox:~/Desktop$ lsmod | grep jiffies
jiffies                12288  0
edwall201@edwall201-VirtualBox:~/Desktop$ cat /proc/jiffies
4295587586
```

Removing:

- Removing the kernel module involves invoking the rmmod command:
  sudo rmmod jiffies

```
edwall201@edwall201-VirtualBox:~/Desktop$ sudo rmmod jiffies
```

Part II:

- Design a kernel module that creates a proc file named /proc/seconds that reports the number of elapsed seconds since the kernel module was loaded.
- The compilation produces several files. The file Second.ko represents the compiled kernel module.

```
edwall201@edwall201-VirtualBox:~/Desktop$ make
make -C /lib/modules/6.5.0-26-generic/build M=/home/edwall201/Desktop modules
make[1]: Entering directory '/usr/src/linux-headers-6.5.0-26-generic'
warning: the compiler differs from the one used to build the kernel
  The kernel was built by: x86_64-linux-gnu-gcc-12 (Ubuntu 12.3.0-1ubuntu1~22.04) 12.3.0
  You are using:           gcc-12 (Ubuntu 12.3.0-1ubuntu1~22.04) 12.3.0
  CC [M]  /home/edwall201/Desktop/Second.o
  MODPOST /home/edwall201/Desktop/Module.symvers
  CC [M]  /home/edwall201/Desktop/Second.mod.o
  LD [M]  /home/edwall201/Desktop/Second.ko
  BTF [M] /home/edwall201/Desktop/Second.ko
Skipping BTF generation for /home/edwall201/Desktop/Second.ko due to unavailability of vml
inux
make[1]: Leaving directory '/usr/src/linux-headers-6.5.0-26-generic'
```

Loading:

- Kernel modules are loaded using the sudo insmod Second.ko command
- To check whether the module has loaded, enter lsmod | grep Second command. And it will show Seconds. To check the contents of this message, enter cat/proc/seconds to show.

```
edwall201@edwall201-VirtualBox:~/Desktop$ sudo insmod /home/edwall201/Desktop/Second.ko
edwall201@edwall201-VirtualBox:~/Desktop$ lsmod | grep Second
Second                  12288  0
edwall201@edwall201-VirtualBox:~/Desktop$ cat /proc/second
cat: /proc/second: No such file or directory
edwall201@edwall201-VirtualBox:~/Desktop$ cat /proc/seconds
Kernel Module is running : 39 seconds
```

Removing:

- Removing the kernel module involves invoking the rmmod command:
  sudo rmmod Second

```
edwall201@edwall201-VirtualBox:~/Desktop$ sudo rmmod Second
```

Chap. 3 – Project 2 Project 1: UNIX Shell and History Feature

Steps to run the program:

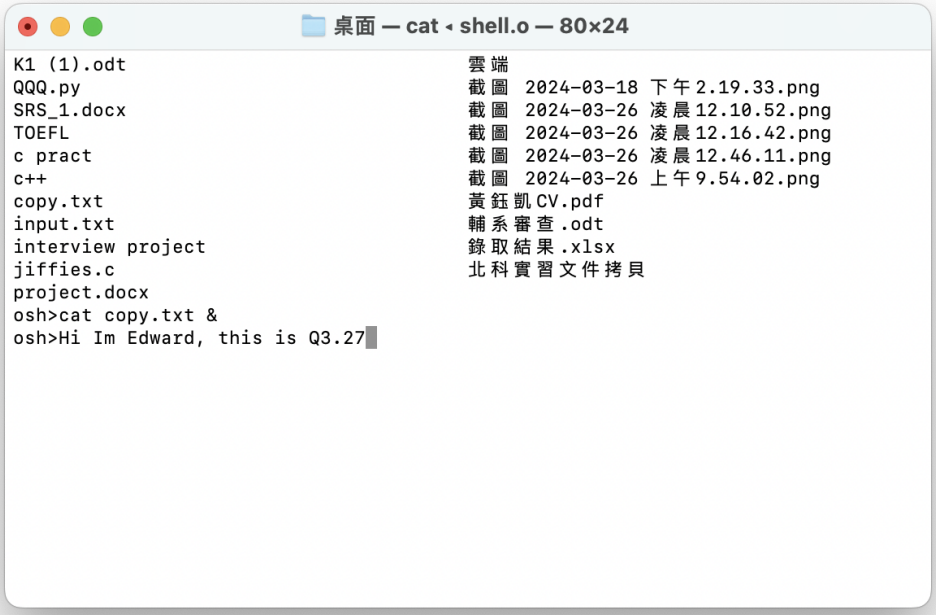
- This project consists of designing a C program to serve as a shell interface that accepts user commands and then executes each command in a separate process.
- Execute ./shell.o
- For the following snapshots to show simple examples.

Functions:

1. Execute a regular command, for example:
   - ls
   - ls -l
   - cat [filename]
   - clear
   - ...

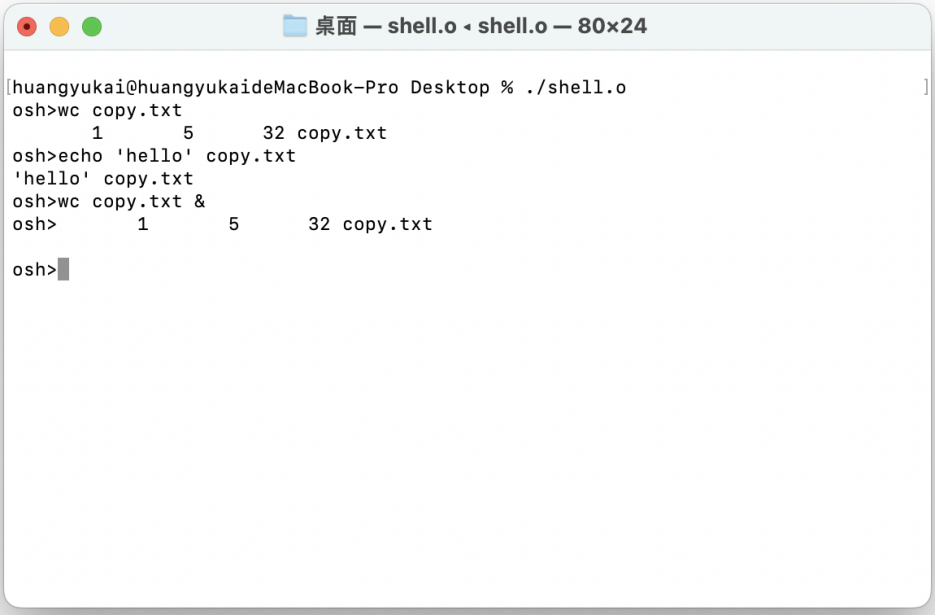
2. Use the background function '&' with regular commands:
   - ls &
   - ls -l &
   - cat [filename] &

- ...

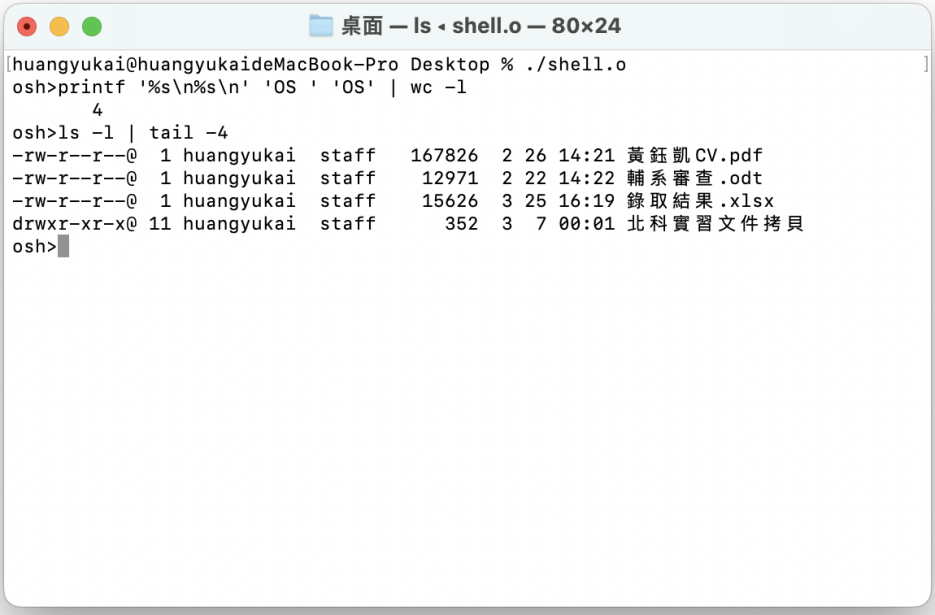


3. Provide command history:
    - Press !! to execute the previous command.

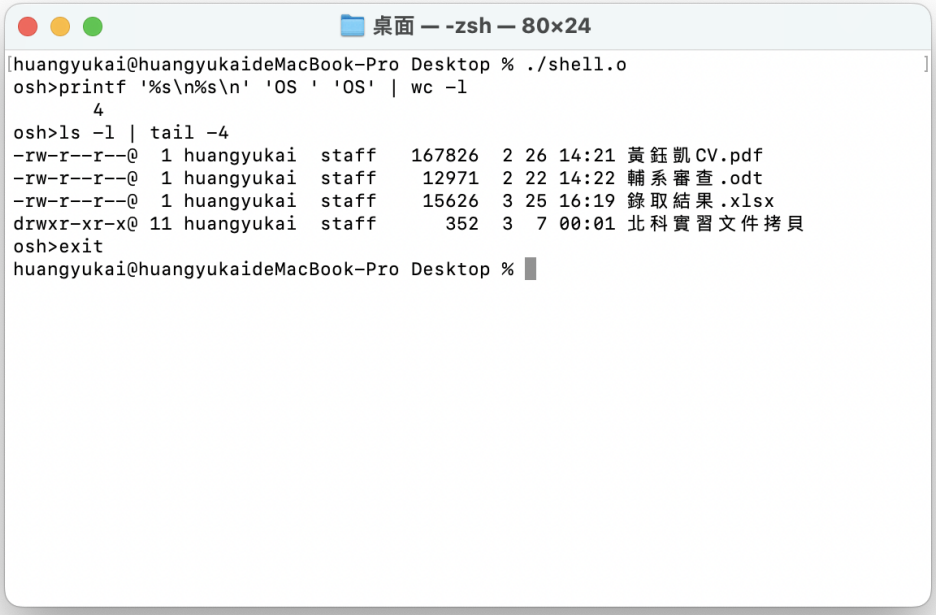
4. Input redirection:
    - wc < newfile (newfile already exists)
5. Output redirection:
    - echo 'hello' > newfile (newfile already exists)
6. Input redirection with '&':
    - wc < newfile &
7. Output redirection with '&':
    - echo 'I love OS' > newfile &
    - cat newfile (to open and view)



8. Use of pipe():
    - printf '%s\n%s\n' 'OS ' 'OS' | wc -l
    - ls -l | tail -4



9. Use of pipe() with '&':
    - printf 'OS \n OS\n' | wc -l &
    - ls -l | tail -4 &
10. Exit the program:
    - exit



Summary:

- Team members

  工管四乙 黃鈺凱 109370210

  資工三 林兆玄 110590021