

工管四乙 109370210 黃鈺凱

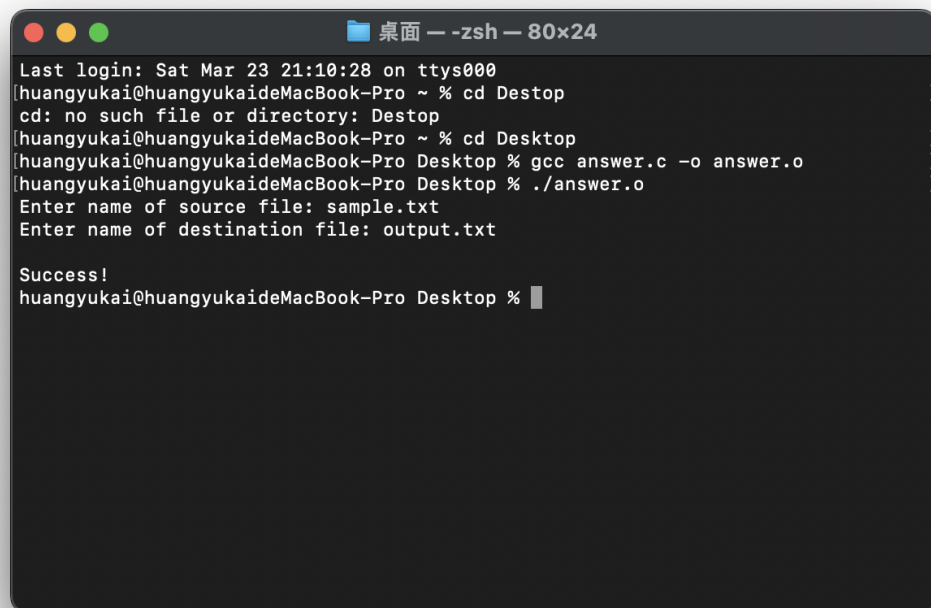
Programming problem

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:

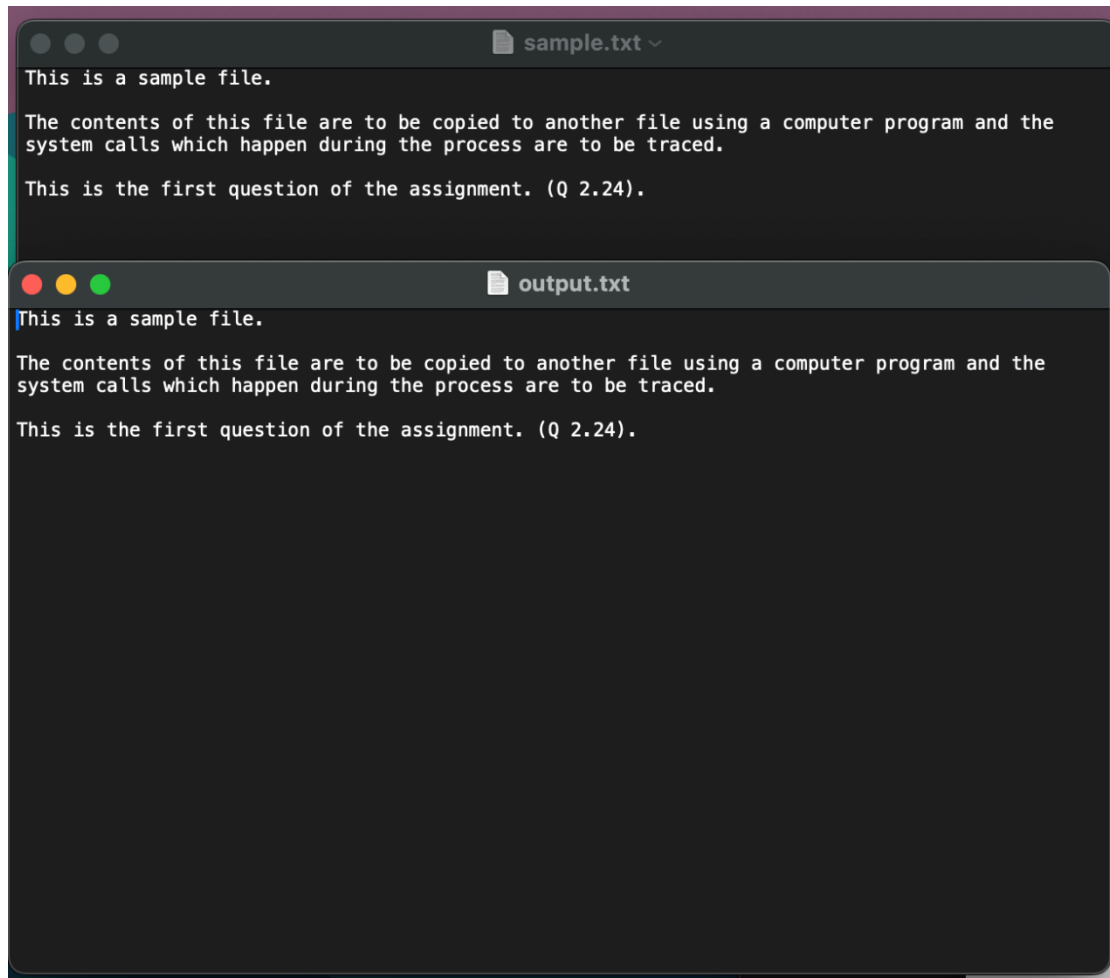
A terminal window titled '桌面 - zsh - 80x24' showing the execution of a program. The user navigates to the Desktop directory, compiles 'answer.c' into 'answer.o', and runs './answer.o'. The program prompts for source and destination file names, which are 'sample.txt' and 'output.txt' respectively. The execution is successful.

```
Last login: Sat Mar 23 21:10:28 on ttys000
huangyukai@huangyukaideMacBook-Pro ~ % cd Destop
cd: no such file or directory: Destop
huangyukai@huangyukaideMacBook-Pro ~ % cd Desktop
huangyukai@huangyukaideMacBook-Pro Desktop % gcc answer.c -o answer.o
huangyukai@huangyukaideMacBook-Pro Desktop % ./answer.o
Enter name of source file: sample.txt
Enter name of destination file: output.txt

Success!
huangyukai@huangyukaideMacBook-Pro Desktop %
```

- 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.

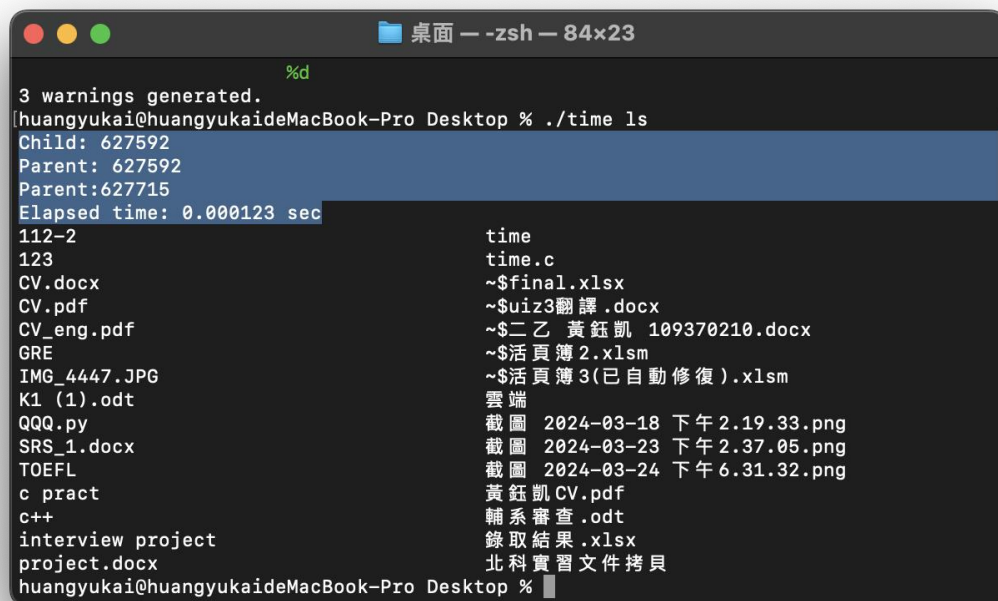


Q3.19

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:

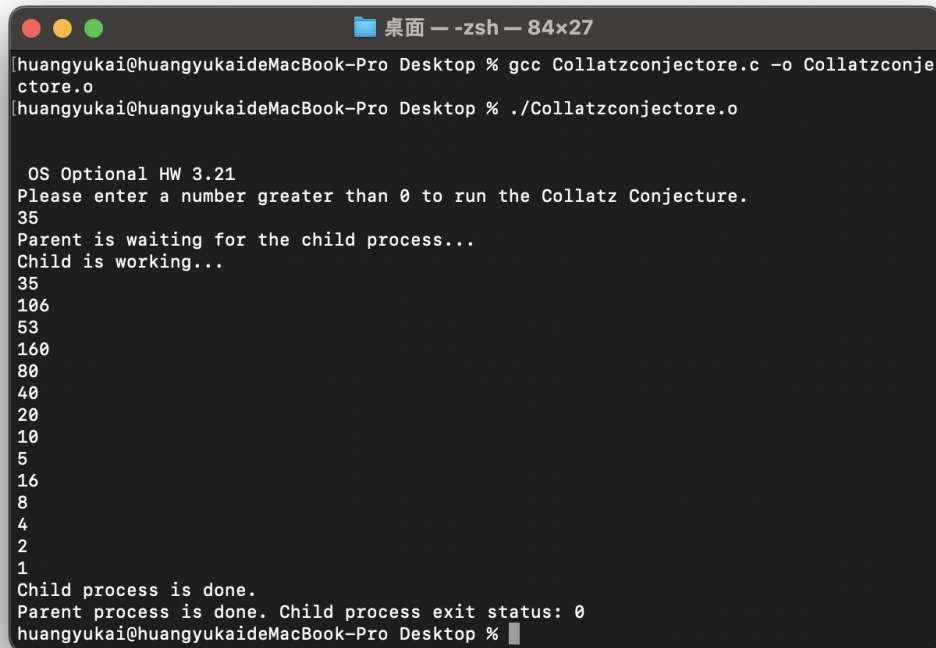


```
桌面 — zsh — 84x23
3 warnings generated.
huangyukai@huangyukaideMacBook-Pro Desktop % ./time ls
Child: 627592
Parent: 627592
Parent:627715
Elapsed time: 0.000123 sec
112-2
123
CV.docx
CV.pdf
CV_eng.pdf
GRE
IMG_4447.JPG
K1 (1).odt
QQQ.py
SRS_1.docx
TOEFL
c pract
c++
interview project
project.docx
huangyukai@huangyukaideMacBook-Pro Desktop %
```

The terminal window displays the output of the `./time ls` command. It shows the child and parent process IDs (627592 and 627715), the elapsed time (0.000123 sec), and a list of files on the desktop. The files are listed in two columns: the first column contains the file names, and the second column contains the file names followed by their modification dates and times.

[Optional] Q.3.21

After command `./Collatzconjecture.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.

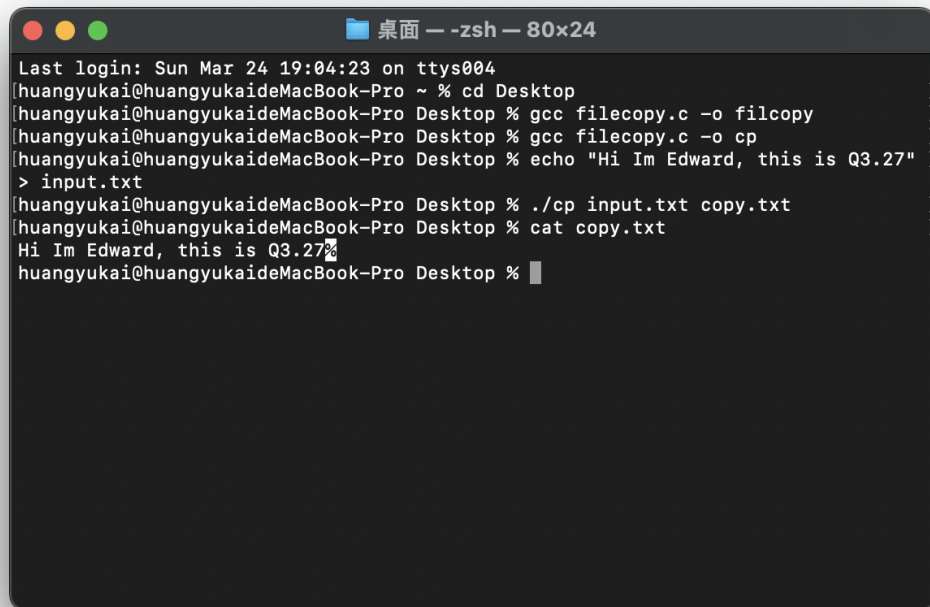


```
桌面 — zsh — 84x27
[huangyukai@huangyukaideMacBook-Pro Desktop % gcc Collatzconjecture.c -o Collatzconje]
[huangyukai@huangyukaideMacBook-Pro Desktop % ./Collatzconjecture.o]

OS Optional HW 3.21
Please enter a number greater than 0 to run the Collatz Conjecture.
35
Parent is waiting for the child process...
Child is working...
35
106
53
160
80
40
20
10
5
16
8
4
2
1
Child process is done.
Parent process is done. Child process exit status: 0
huangyukai@huangyukaideMacBook-Pro Desktop %
```

[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!



```
桌面 — zsh — 80x24
Last login: Sun Mar 24 19:04:23 on ttys004
huangyukai@huangyukaideMacBook-Pro ~ % cd Desktop
huangyukai@huangyukaideMacBook-Pro Desktop % gcc filecopy.c -o filcopy
huangyukai@huangyukaideMacBook-Pro Desktop % gcc filecopy.c -o cp
huangyukai@huangyukaideMacBook-Pro Desktop % echo "Hi Im Edward, this is Q3.27"
> input.txt
huangyukai@huangyukaideMacBook-Pro Desktop % ./cp input.txt copy.txt
huangyukai@huangyukaideMacBook-Pro Desktop % cat copy.txt
Hi Im Edward, this is Q3.27
huangyukai@huangyukaideMacBook-Pro Desktop %
```

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 unavailability 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/jiffies.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 vmlinux
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`
- ...



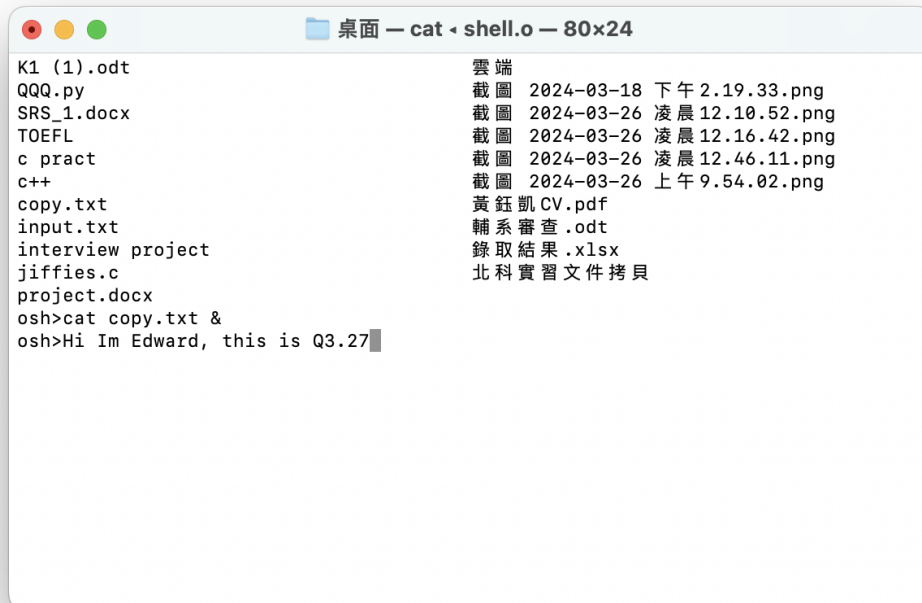
A terminal window titled "桌面 — shell.o — 80x24" showing the compilation and execution of a shell program. The user runs `gcc shell.c -o shell.o` and then `./shell.o`. The prompt changes to `osh>`. The user enters `ls`, and the terminal displays a two-column listing of files and directories in the current directory, including `112-2`, `123`, `CV.docx`, `CV.pdf`, `CV_eng.pdf`, `GRE`, `IMG_4447.JPG`, `K1 (1).odt`, `QQQ.py`, `SRS_1.docx`, `TOEFL`, `c pract`, `c++`, `copy.txt`, `input.txt`, `interview project`, `jiffies.c`, `project.docx`, `shell.c`, `shell.o`, `~$final.xlsx`, `~$uiz3翻譯.docx`, `~$二乙 黃鈺凱 109370210.docx`, `~$活頁簿2.xlsx`, `~$活頁簿3(已自動修復).xlsx`, `雲端`, `截圖 2024-03-18 下午2.19.33.png`, `截圖 2024-03-26 凌晨12.10.52.png`, `截圖 2024-03-26 凌晨12.16.42.png`, `截圖 2024-03-26 凌晨12.46.11.png`, `截圖 2024-03-26 上午9.54.02.png`, `黃鈺凱CV.pdf`, `輔系審查.odt`, `錄取結果.xlsx`, and `北科實習文件拷貝`. The prompt returns to `osh>`.

```
[huangyukai@huangyukaideMacBook-Pro Desktop % gcc shell.c -o shell.o
huangyukai@huangyukaideMacBook-Pro Desktop % ./shell.o
osh>ls
112-2                                shell.c
123                                shell.o
CV.docx                            ~$final.xlsx
CV.pdf                            ~$uiz3翻譯.docx
CV_eng.pdf                        ~$二乙 黃鈺凱 109370210.docx
GRE                              ~$活頁簿2.xlsx
IMG_4447.JPG                      ~$活頁簿3(已自動修復).xlsx
K1 (1).odt                        雲端
QQQ.py                          截圖 2024-03-18 下午2.19.33.png
SRS_1.docx                      截圖 2024-03-26 凌晨12.10.52.png
TOEFL                            截圖 2024-03-26 凌晨12.16.42.png
c pract                          截圖 2024-03-26 凌晨12.46.11.png
c++                              截圖 2024-03-26 上午9.54.02.png
copy.txt                        黃鈺凱CV.pdf
input.txt                      輔系審查.odt
interview project              錄取結果.xlsx
jiffies.c                     北科實習文件拷貝
project.docx
osh>
```

2. Use the background function '&' with regular commands:

- `ls &`
- `ls -l &`
- `cat [filename] &`

• ...



A terminal window titled "桌面 — cat < shell.o — 80x24". The window displays two columns of file names and a list of commands. The first column lists files: K1 (1).odt, QQQ.py, SRS_1.docx, TOEFL, c pract, c++, copy.txt, input.txt, interview project, jiffies.c, project.docx, osh>cat copy.txt &, and osh>Hi Im Edward, this is Q3.27. The second column lists files: 雲端, 截圖 2024-03-18 下午 2.19.33.png, 截圖 2024-03-26 凌晨 12.10.52.png, 截圖 2024-03-26 凌晨 12.16.42.png, 截圖 2024-03-26 凌晨 12.46.11.png, 截圖 2024-03-26 上午 9.54.02.png, 黃鈺凱 CV.pdf, 輔系審查.odt, 錄取結果.xlsx, and 北科實習文件拷貝.

```
桌面 — cat < shell.o — 80x24
K1 (1).odt          雲端
QQQ.py             截圖 2024-03-18 下午 2.19.33.png
SRS_1.docx         截圖 2024-03-26 凌晨 12.10.52.png
TOEFL              截圖 2024-03-26 凌晨 12.16.42.png
c pract            截圖 2024-03-26 凌晨 12.46.11.png
c++                截圖 2024-03-26 上午 9.54.02.png
copy.txt           黃鈺凱 CV.pdf
input.txt          輔系審查.odt
interview project  錄取結果.xlsx
jiffies.c          北科實習文件拷貝
project.docx
osh>cat copy.txt &
osh>Hi Im Edward, this is Q3.27
```

3. Provide command history:

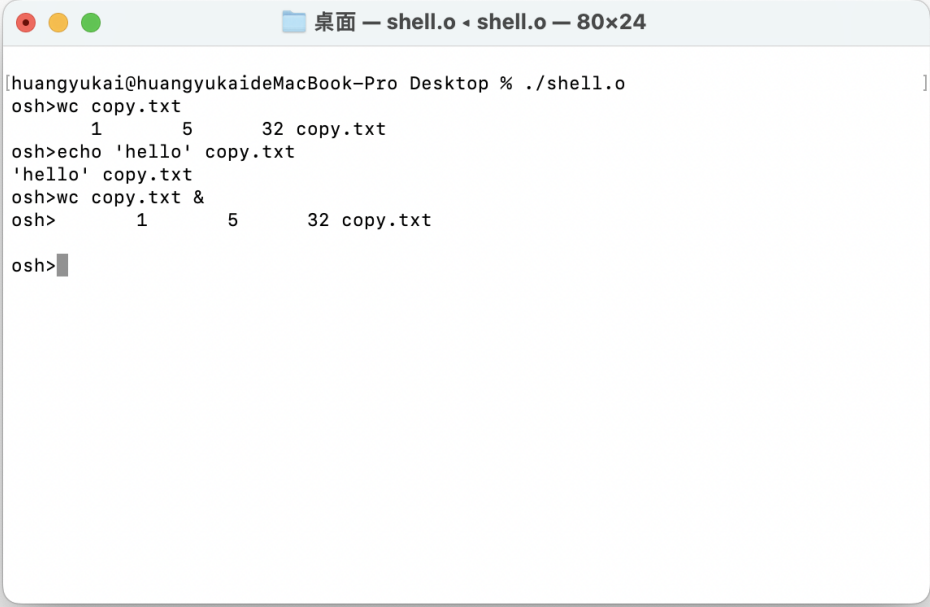
- Press !! to execute the previous command.



A terminal window titled "桌面 — cat < shell.o — 80x24". The window displays the same file listings as the previous image, but with additional commands and their output. The commands are: osh>cat copy.txt &, osh>Hi Im Edward, this is Q3.27, osh>!!, cat copy.txt &, and osh>Hi Im Edward, this is Q3.27. The output of the last command is: 雲端, 截圖 2024-03-18 下午 2.19.33.png, 截圖 2024-03-26 凌晨 12.10.52.png, 截圖 2024-03-26 凌晨 12.16.42.png, 截圖 2024-03-26 凌晨 12.46.11.png, 截圖 2024-03-26 上午 9.54.02.png, 黃鈺凱 CV.pdf, 輔系審查.odt, 錄取結果.xlsx, and 北科實習文件拷貝.

```
桌面 — cat < shell.o — 80x24
K1 (1).odt          雲端
QQQ.py             截圖 2024-03-18 下午 2.19.33.png
SRS_1.docx         截圖 2024-03-26 凌晨 12.10.52.png
TOEFL              截圖 2024-03-26 凌晨 12.16.42.png
c pract            截圖 2024-03-26 凌晨 12.46.11.png
c++                截圖 2024-03-26 上午 9.54.02.png
copy.txt           黃鈺凱 CV.pdf
input.txt          輔系審查.odt
interview project  錄取結果.xlsx
jiffies.c          北科實習文件拷貝
project.docx
osh>cat copy.txt &
osh>Hi Im Edward, this is Q3.27
osh>!!
cat copy.txt &
osh>Hi Im Edward, this is Q3.27
```

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)

A terminal window titled "桌面 — shell.o ◀ shell.o — 80x24" is shown. The prompt is [huangyukai@huangyukaideMacBook-Pro Desktop % ./shell.o]. The terminal displays the output of a script: osh>wc copy.txt, followed by a table with 3 columns: 1, 5, and 32 copy.txt. Then osh>echo 'hello' copy.txt, followed by 'hello' copy.txt. Then osh>wc copy.txt &, followed by osh> and the same table: 1, 5, 32 copy.txt. The prompt osh> is at the bottom.

```
[huangyukai@huangyukaideMacBook-Pro Desktop % ./shell.o]
osh>wc copy.txt
      1      5      32 copy.txt
osh>echo 'hello' copy.txt
'hello' copy.txt
osh>wc copy.txt &
osh>      1      5      32 copy.txt
osh>
```

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

```
桌面 — ls - shell.o — 80x24
[huangyukai@huangyukaideMacBook-Pro Desktop % ./shell.o
osh>printf '%s\n%s\n' 'OS ' 'OS' | wc -l
4
osh>ls -l | tail -4
-rw-r--r--@ 1 huangyukai staff 167826 2 26 14:21 黃鈺凱 CV.pdf
-rw-r--r--@ 1 huangyukai staff 12971 2 22 14:22 輔系審查.odt
-rw-r--r--@ 1 huangyukai staff 15626 3 25 16:19 錄取結果.xlsx
drwxr-xr-x@ 11 huangyukai staff 352 3 7 00:01 北科實習文件拷貝
osh>
```

9. Use of pipe() with '&':

- `printf 'OS \n OS\n' | wc -l &`
- `ls -l | tail -4 &`

10.Exit the program:

- `exit`

```
桌面 — -zsh — 80x24
[huangyukai@huangyukaideMacBook-Pro Desktop % ./shell.o
osh>printf '%s\n%s\n' 'OS ' 'OS' | wc -l
4
osh>ls -l | tail -4
-rw-r--r--@ 1 huangyukai staff 167826 2 26 14:21 黃鈺凱 CV.pdf
-rw-r--r--@ 1 huangyukai staff 12971 2 22 14:22 輔系審查.odt
-rw-r--r--@ 1 huangyukai staff 15626 3 25 16:19 錄取結果.xlsx
drwxr-xr-x@ 11 huangyukai staff 352 3 7 00:01 北科實習文件拷貝
osh>exit
huangyukai@huangyukaideMacBook-Pro Desktop %
```

Summary:

- Team members

工管四乙 黃鈺凱 109370210

資工三 林兆玄 110590021