# Computer Programming 1 Lab

2023-04-19

# Outline

- UNIX Commands
- Greedy algorithm
- HW05

# UNIX commands

a) Set the permission for the “CP2” directory in your home directory as follows: the owner has read and write permissions, the users in the same group have read and execute permissions, and other users have only read permission for the files/directories in the directory

answer

```
chmod 654 ~/CP1
```

## b) Concatenate all .txt file into a file named “concat.txt”

## answer

```
cat ./*.txt > ./concat.txt
```

## c) Continuing from (b), count the number of lines in the file.

## answer

```
wc -l ./concat.txt
```

## d) Continuing from (b), find out all the line(s) containing “int”

## answer

```
grep “int” ./concat.txt
```

## e) Execute a program named “YJSP” at background.

## answer

```
./YJSP &
```

## f) Continuing from (e), please show the PID of the background process.

### answer

```
jobs -l
jobs -p
ps -u $USER
```

g) Continuing from (f), assuming the PID for the process is “1919810”. Please save it in an environment variable named “YJSP_PID”.

answer

```
export YJSP_PID=”1919810”
```

## h) Continuing from (g), please terminate the background process.

## answer

```
kill $YJSP_PID
```

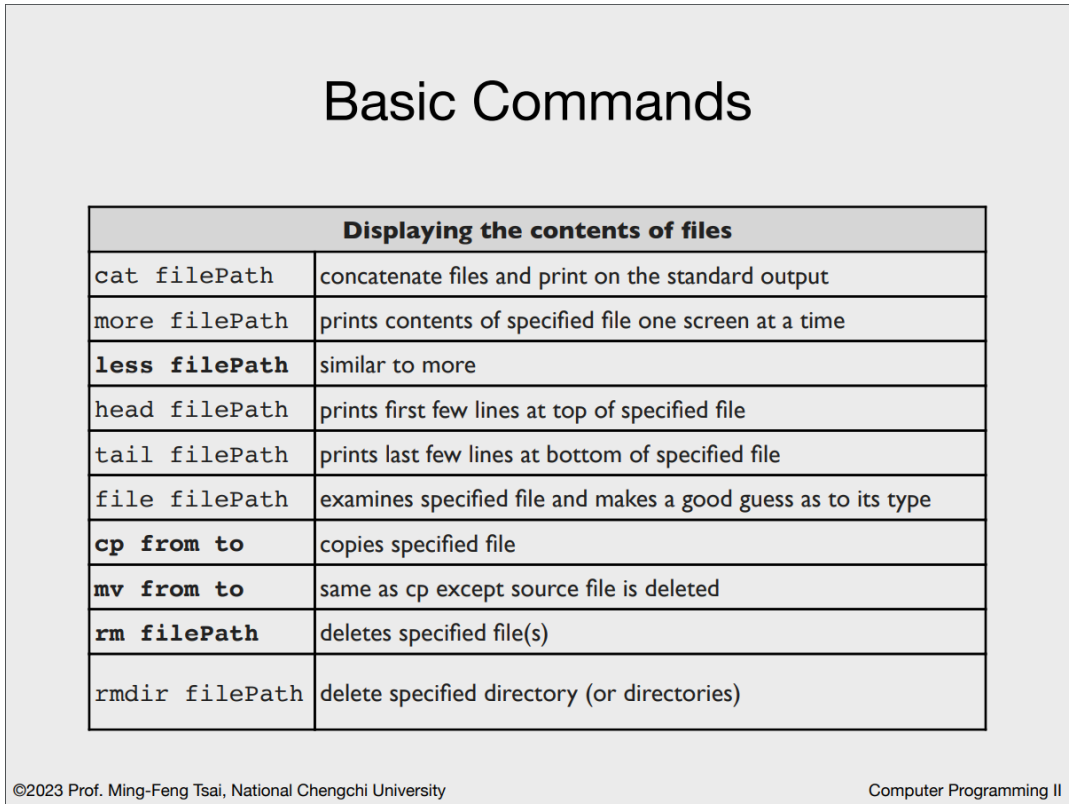
i) Continuing from (e), please terminate the background process. You can't use any information gained from any other questions, and we assume there's only one background process exists, which is our target. (4pts).

answer

```
kill `jobs -p`
```

# How to prepare

## 1. Memorize all the commands mentioned in PPT.



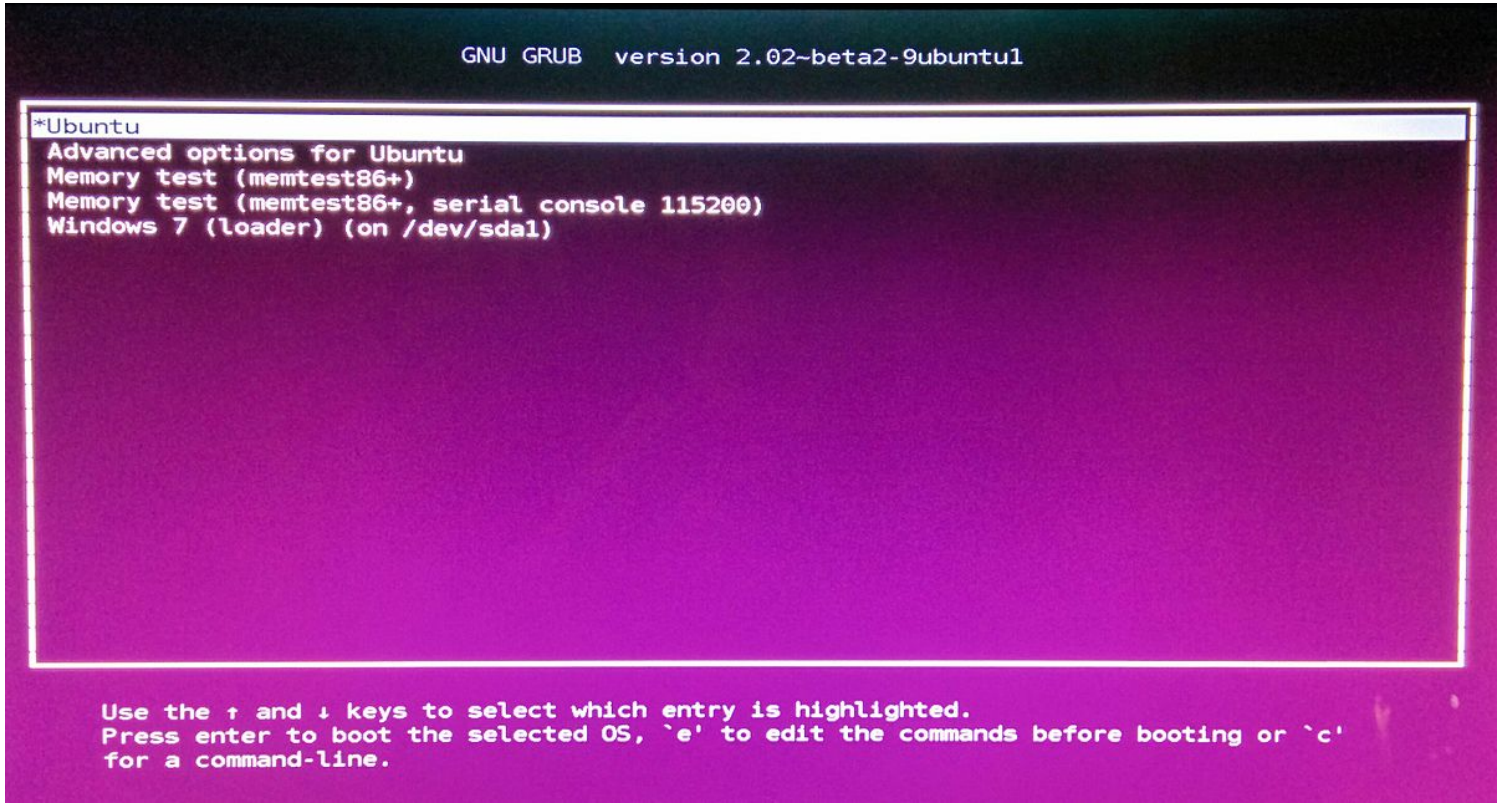
Basic Commands

| Displaying the contents of files | |
|---|---|
| `cat filePath` | concatenate files and print on the standard output |
| `more filePath` | prints contents of specified file one screen at a time |
| **`less filePath`** | similar to more |
| `head filePath` | prints first few lines at top of specified file |
| `tail filePath` | prints last few lines at bottom of specified file |
| `file filePath` | examines specified file and makes a good guess as to its type |
| **`cp from to`** | copies specified file |
| **`mv from to`** | same as cp except source file is deleted |
| **`rm filePath`** | deletes specified file(s) |
| `rmdir filePath` | delete specified directory (or directories) |




If you don't like this way, you can try something else...

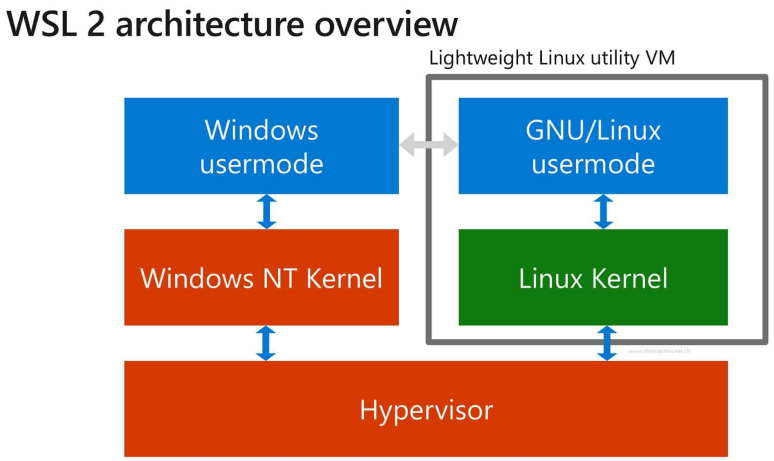
# 2. Install a dual boot systyem

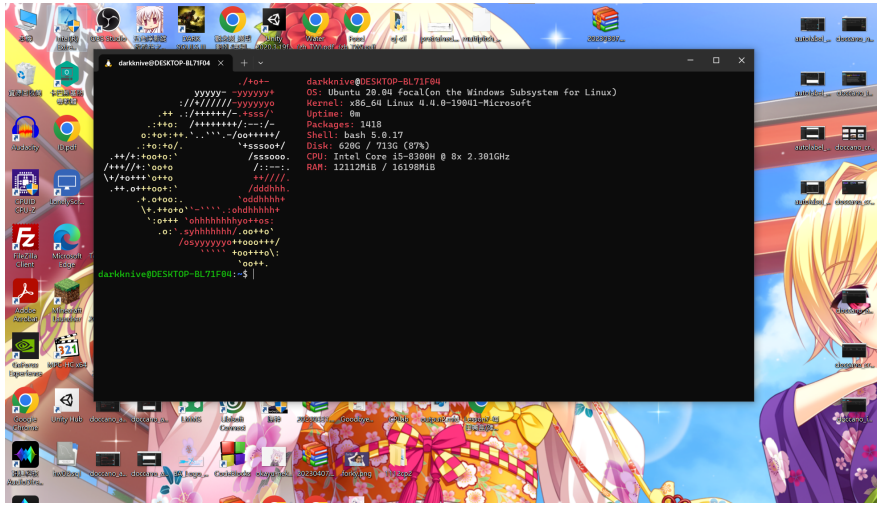



: But this is too hard/complicated!

Maybe you can try something else...

# 3. WSL

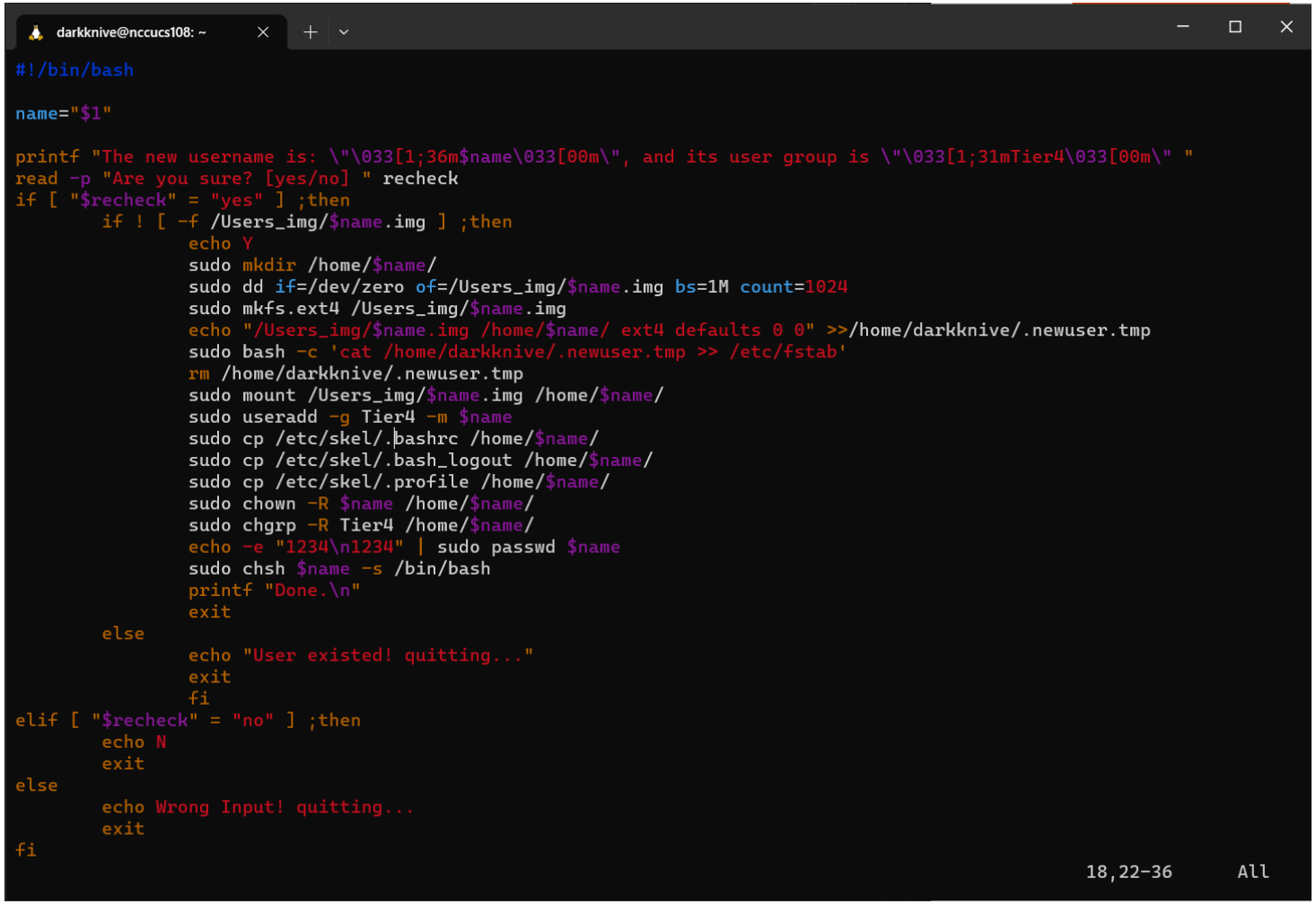


You can have a lightweighted linux environment without breaking your system!

## bash script

- Help user do repeat and complicate works.
- Run on (mostly) all linux systems.
- Don't need to install script parser (such as python)
- Ultra FAST!


darkknive@nccucs108: ~
#!/bin/bash
name="$1"
printf "The new username is: \"\033[1;36m$name\033[00m\", and its user group is \"\033[1;31mTier4\033[00m\" "
read -p "Are you sure? [yes/no] " recheck
if [ "$recheck" = "yes" ] ;then
if ! [ -f /Users_img/$name.img ] ;then
echo Y
sudo mkdir /home/$name/
sudo dd if=/dev/zero of=/Users_img/$name.img bs=1M count=1024
sudo mkfs.ext4 /Users_img/$name.img
echo "/Users_img/$name.img /home/$name/ ext4 defaults 0 0" >>/home/darkknive/.newuser.tmp
sudo bash -c 'cat /home/darkknive/.newuser.tmp >> /etc/fstab'
rm /home/darkknive/.newuser.tmp
sudo mount /Users_img/$name.img /home/$name/
sudo useradd -g Tier4 -m $name
sudo cp /etc/skel/.bashrc /home/$name/
sudo cp /etc/skel/.bash_logout /home/$name/
sudo cp /etc/skel/.profile /home/$name/
sudo chown -R $name /home/$name/
sudo chgrp -R Tier4 /home/$name/
echo -e "1234\n1234" | sudo passwd $name
sudo chsh $name -s /bin/bash
printf "Done.\n"
exit
else
echo "User existed! quitting..."
exit
fi
elif [ "$recheck" = "no" ] ;then
echo N
exit
else
echo Wrong Input! quitting...
exit
fi
18,22-36 All

## Example

```
#! /bin/bash

for i in {1..10}; do
    echo $i
done
```

## Shebang

- A line start with "#!"
- Calls the executable to parse this script
    - In this case, */bin/bash* is called to parse this script.

## Example

```
#! /usr/bin/python3

for i in range(1, 6):
    print(i)
```

## results

```
darkknive@nccucs108:~$ ./count2.py
1
2
3
4
5
darkknive@nccucs108:~$
```

## Example

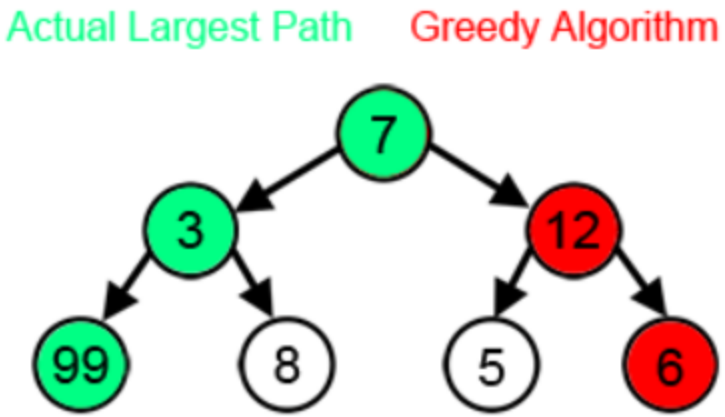
```
#! /bin/bash

g++ ./hw02.cpp

for i in {1..5}; do
    if [ $(./a.out < ./$i.in | diff ./$i.out - | wc -l) -ne 0 ]; then
        echo "data $i wrong!"
    else
        echo "data $i correct!"
    fi
done
```

And there's alot more tips! See this link.

# Greedy algorithm

# Greedy

- Always takes the best decision
- Only think for current condition
- No need of searching back.



## Exercise 2

**How to solve this problem?**

- Always keep the lowest price you've ever seen -> Pretend you've bought the stock at that price.
- Calculate how many money you can make *if* you sell the stock now.

# HW05

# Tips

- Always keep the highest two values for each mine. *Why?*
- How to consider a naive solution works or not? Time complexity!

# Any Questions?