

Shell Script Practice

1. 建立conda虛擬環境:

A. 建立pytorch的執行環境 (指令不需寫在一份script)。

```
(pytorch2-gpu) yucheng@yucheng-System-Product-Name:~$ python
Python 3.9.17 (main, Jul 5 2023, 20:41:20)
[GCC 11.2.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> import torch
>>> torch.cuda.is_available()
True
```

B. 建立tensorflow的執行環境 (指令不需寫在一份script)。

```
(tf-gpu) yucheng@yucheng-System-Product-Name:~$ python
Python 3.9.17 (main, Jul 5 2023, 20:41:20)
[GCC 11.2.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
2023-07-10 00:45:59.983796: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library libcudart.so.10.1
>>> tf.test.is_gpu_available()

2023-07-10 00:46:09.106093: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1406] Created TensorFlow device (/device:GPU:0 with 10209 MB memory) -> physical GPU (device: 0, name: NVIDIA GeForce GTX 1080 Ti, pci bus id: 0000:01:00.0, compute capability: 6.1)
True
```

2. 寫一份能達成以下需求的Dockerfile, 並以此Dockerfile使用podman建立一個container。

需求:

- A. 能存取顯卡 (nvidia-smi should work)
- B. 安裝pytorch-gpu
- C. 安裝git、vim、curl、htop、tmux

```
root@636265f34da1:/workspace# python
Python 3.8.12 | packaged by conda-forge | (default, Oct 12 2021, 21:59:51)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> torch.cuda.is_available()
True
>>>
[2]+  Stopped                  python
root@636265f34da1:/workspace# git --version
git version 2.25.1
root@636265f34da1:/workspace# htop --version
htop 2.2.0 - (C) 2004-2019 Hisham Muhammad
Released under the GNU GPL.

root@636265f34da1:/workspace# tmux -V
tmux 3.0a
root@636265f34da1:/workspace# |
```

3. 寫一份script, 輸出當前機器上的顯卡資訊。格式如下圖。

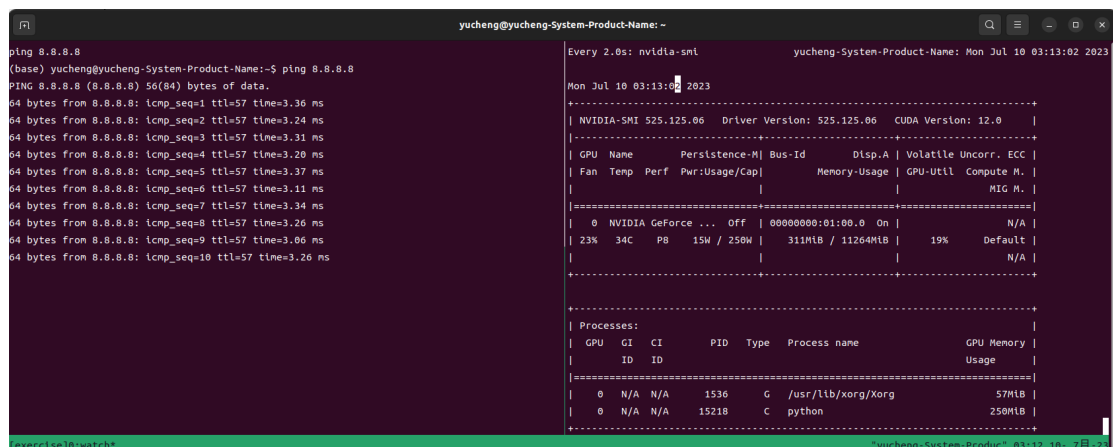
```
(base) NV04% ./gpu_info.sh
Driver Version: 510.85.02
CUDA Version: 11.6
GPUs: 4
1: NVIDIA GeForce GTX 1080 Ti
2: NVIDIA GeForce GTX 1080 Ti
3: NVIDIA GeForce GTX 1080 Ti
4: NVIDIA GeForce GTX 1080 Ti
```

```
(base) NV27% ./gpu_info.sh
Driver Version: 520.61.05
CUDA Version: 11.8
GPUs: 4
1: NVIDIA GeForce RTX 3080 Ti
2: NVIDIA GeForce RTX 3080 Ti
3: NVIDIA GeForce RTX 3080 Ti
4: NVIDIA GeForce RTX 3080 Ti
```

4. 寫一份script, 輸出當前機器上的cpu資訊。格式如下圖。

```
(base) NV04% ./cpu_info.sh
Architecture:          x86_64
CPU(s):                56
Model name:             Intel(R) Xeon(R) CPU E5-2683 v3 @ 2.00GHz
```

5. 寫一份自動開啟tmux的script。session名稱是exercise, 水平等分成兩個windows, 左邊執行ping 8.8.8.8, 右邊執行每兩秒刷新一次的nvidia-smi。



```
yucheng@yucheng-System-Product-Name: ~
ping 8.8.8.8
(base) yucheng@yucheng-System-Product-Name:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=57 time=3.36 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=57 time=3.24 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=57 time=3.31 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=57 time=3.20 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=57 time=3.37 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=57 time=3.11 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=57 time=3.34 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=57 time=3.26 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=57 time=3.06 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=57 time=3.26 ms

Every 2.0s: nvidia-smi
yucheng-System-Product-Name: Mon Jul 10 03:13:02 2023
+-----+
| NVIDIA-SMI 525.125.06 Driver Version: 525.125.06 CUDA Version: 12.0 |
+-----+
| GPU Name Persistence-M Bus-Id Disp.A | Volatile Uncorr. ECC | | | | | | |
| Fan Temp Perf Pwr:Usage/Cap | Memory-Usage | GPU-Util Compute M. |
| | | | | | | | |
+-----+
| 0 NVIDIA GeForce ... Off | 00000000:01:00.0 On | N/A | | | | | |
| 23% 34C P8 15W / 250W | 311MiB / 11264MiB | 19% Default |
| | | | | | | | |
+-----+
+-----+
| Processes: |
| GPU GI CI PID Type Process name GPU Memory |
| ID ID | | | | | | |
+-----+
| 0 N/A N/A 1536 G /usr/lib/xorg/Xorg 57MiB |
| 0 N/A N/A 15218 C python 250MiB |
+-----+
exercise]0:watch*
```

6. 計算files資料夾裡所有weight檔大小總和。weight檔固定以.pt結尾，輸出格式如下圖。

```
(base) yucheng@yucheng-System-Product-Name:~$ ./get_weights_size.sh
=====
file name      size(KB)      size(MB)
=====
task1.pt       85364930      81.41
task2.pt       85492666      81.53
task3.pt       85748138      81.78
=====
total          256605734     244.72
```

7. (leetcode 192):

Write a bash script to calculate the [frequency](#) of each word in a text file `words.txt`.

For simplicity sake, you may assume:

- `words.txt` contains only lowercase characters and space ' ' characters.
- Each word must consist of lowercase characters only.
- Words are separated by one or more whitespace characters.

Example:

Assume that `words.txt` has the following content:

```
the day is sunny the the
the sunny is is
```

Your script should output the following, sorted by descending frequency:

```
the 4
is 3
sunny 2
day 1
```

8. (leetcode 193):

Given a text file `file.txt` that contains a list of phone numbers (one per line), write a one-liner bash script to print all valid phone numbers.

You may assume that a valid phone number must appear in one of the following two formats: `(xxx) xxx-xxxx` or `xxx-xxx-xxxx`. (x means a digit)

You may also assume each line in the text file must not contain leading or trailing white spaces.

Example:

Assume that `file.txt` has the following content:

```
987-123-4567
123 456 7890
(123) 456-7890
```

Your script should output the following valid phone numbers:

```
987-123-4567
(123) 456-7890
```

9. (leetcode 194):

Given a text file `file.txt`, transpose its content.

You may assume that each row has the same number of columns, and each field is separated by the `' '` character.

Example:

If `file.txt` has the following content:

```
name age
alice 21
ryan 30
```

Output the following:

```
name alice ryan
age 21 30
```

10. (leetcode 195):

Given a text file `file.txt`, print just the 10th line of the file.

Example:

Assume that `file.txt` has the following content:

```
Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
Line 7
Line 8
Line 9
Line 10
```

Your script should output the tenth line, which is:

```
Line 10
```

11. 以下是一個顯示進度條的script。

```
1 #!/bin/bash
2
3 while :
4 do
5     clear
6     for i in {1..20}
7     do
8         echo -n "*"
9         sleep 0.1
10    done
11    clear
12 done
13
14
```

如何將進度條的*修改成綠色*?

12. 以下是一個輸出費式數列的script。

```
1 #!/bin/bash
2
3 list=(0 1)
4 for i in `seq 2 11`
5 do
6     list[$i] = list[-1] + list[-2]
7 done
8
9 echo ${list[@]}
10
```

要如何修改才能使其正常輸出？

13. 以下是一個統計process執行狀態的script。完成以下case中的TODO部分。

```
1 #!/bin/bash
2
3 ALL_PROCESS=$(ls /proc/ | egrep '[0-9]+')
4
5 running=0
6 stopped=0
7 sleeping=0
8 zombie=0
9
10 for pid in ${ALL_PROCESS[*]}
11 do
12     test -f /proc/$pid/status && state=$(egrep "State" /proc/$pid/status | awk '{print $2}')
13     case $state in
14         # TODO
15         # if $state == "R": running++
16         # elif $state == "T": stopped++
17         # elif $state == "S": sleeping++
18         # elif $state == "Z": zombie++
19     esac
20 done
21
22 echo -e "running: $running\nstopped: $stopped\nsleeping: $sleeping\nzombie: $zombie\n"
23
```

14. 以下是一個檢查機器port狀態的script。

```
1 #!/bin/bash
2
3 HOST=$1
4 PORT="22 25 80 8080"
5 for PORT in $PORT;
6 do
7     if echo &>/dev/null > /dev/tcp/$HOST/$PORT; then
8         echo "$PORT open"
9     else
10        echo "$PORT close"
11    fi
12 done
13
```

上述script中，port是寫死的。如何將其修改成ip和port皆用parse input的形式？如下圖：

```
(base) NV04% ./port_status_ans.sh 127.0.0.1 80 8080
80 close
8080 open
```

15. 以下是一個檢查ip格式是否正確的script。若格式正確即退出執行，否則繼續輸入。請說明錯誤之處及如何修改。

```
1 #!/bin/bash
2
3 function check_ip(){
4     local IP=$1
5     VALID_CHECK=$(echo $IP | awk -F. '{ $1<=255 && $2<=255 && $3<=255 && $4<=255 {print "yes"}}')
6     if echo $IP|grep -E "[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}" >/dev/null; then
7         if [ $VALID_CHECK == "yes" ]; then
8             return 0
9         else
10            echo "$IP not available!"
11            return 1
12        fi
13    else
14        echo "Format error! Please input again."
15        return 1
16    fi
17 }
18
19 while true;
20 do
21     read -p "Please enter IP: " IP
22     check_ip($IP)
23     [ $? -eq 0 ] && break || continue
24 done
25
```

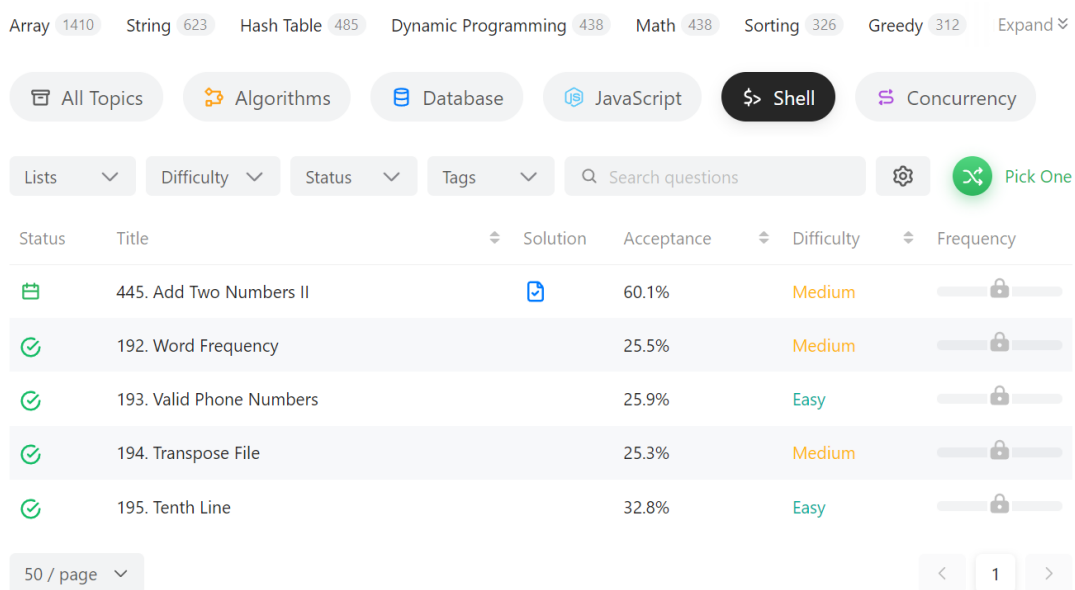
Submission:

檔名: [your_name].pdf

(交一份pdf檔就好)

Q1~Q6: 記錄指令或截圖script檔，並附上在機器上的執行結果，格式如題目附圖。在自己的機器或實驗室機器上都行。

Q7~Q10: leetcode的AC截圖。如下圖：



The screenshot shows the LeetCode search results for the 'Shell' category. The interface includes a top navigation bar with various topics like Array, String, Hash Table, etc. Below this, there are filters for 'All Topics', 'Algorithms', 'Database', 'JavaScript', 'Shell' (selected), and 'Concurrency'. A search bar is present with the text 'Search questions'. The results are displayed in a table with columns: Status, Title, Solution, Acceptance, Difficulty, and Frequency. The table lists five problems, all with a 'Solved' status (green checkmark). The problems are: 445. Add Two Numbers II (60.1% acceptance, Medium difficulty), 192. Word Frequency (25.5% acceptance, Medium difficulty), 193. Valid Phone Numbers (25.9% acceptance, Easy difficulty), 194. Transpose File (25.3% acceptance, Medium difficulty), and 195. Tenth Line (32.8% acceptance, Easy difficulty). Each problem has a 'Pick One' button next to it. At the bottom, there is a pagination control showing '50 / page' and a page number '1'.

Status	Title	Solution	Acceptance	Difficulty	Frequency
	445. Add Two Numbers II		60.1%	Medium	
	192. Word Frequency		25.5%	Medium	
	193. Valid Phone Numbers		25.9%	Easy	
	194. Transpose File		25.3%	Medium	
	195. Tenth Line		32.8%	Easy	

Q11~Q15: 記錄需要修改的地方及如何修改即可。