

Họ và tên: Đồng Quốc Thắng

Mã số sinh viên: 23521421

Lớp: IT007.P11.CTTN

## HỆ ĐIỀU HÀNH BÁO CÁO LAB 2

### CHECKLIST (Đánh dấu x khi hoàn thành)

**Lưu ý mỗi câu phải làm đủ 3 yêu cầu**

#### I. CLASSWORK

	BT 1	BT 2	BT 3	BT 4	BT 5
Trình bày cách làm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Chụp hình minh chứng	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Giải thích kết quả	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

#### II. HOMEWORK

	BT1	BT2	BT3	BT4	BT5	BT6	BT7
Trình bày cách làm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Chụp hình minh chứng	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Giải thích kết quả	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

### III. BONUS

	BT1	BT2
Trình bày cách làm	<input type="checkbox"/>	<input type="checkbox"/>
Chụp hình minh chứng	<input type="checkbox"/>	<input type="checkbox"/>
Giải thích kết quả	<input type="checkbox"/>	<input type="checkbox"/>

**Tư chấm điểm:** 10

*\*Lưu ý: Xuất báo cáo theo định dạng PDF, đặt tên theo cú pháp:  
<MSSV>\_LABx.pdf*

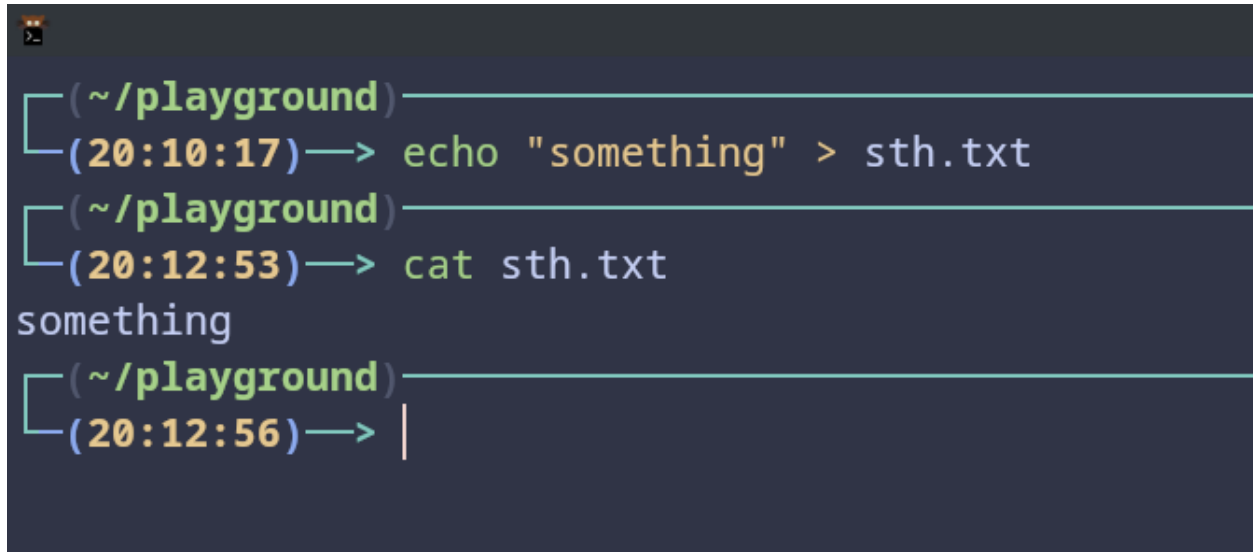
## I. CLASSWORK

### 1. Classwork 01

Chuyển hướng đầu ra: >

Cách làm: Gõ dấu ">" sau 1 câu lệnh và gõ tên file sau dấu > để chuyển hướng đầu ra của câu lệnh vào file đó

Ví dụ: echo "something" > sth.txt

A terminal window with a dark background and light green text. It shows three lines of commands and their outputs. The first line is a prompt '(~/playground) ' followed by the command 'echo "something" > sth.txt'. The second line is a prompt '(~/playground) ' followed by the command 'cat sth.txt', with the output 'something' on the next line. The third line is a prompt '(~/playground) ' followed by a cursor '|'.

```
(~/playground) —→ echo "something" > sth.txt
(~/playground) —→ cat sth.txt
something
(~/playground) —→ |
```

Giải thích: dấu ">" sẽ chuyển hướng từ stdout đến file mà mình chọn, ở đây chọn là file sth.txt. Câu lệnh echo bình thường sẽ in ra stdout mà ở đây ta dùng ">" nên file sth.txt giờ đây có chữ something.

Nối thêm đầu ra: >>

Cách làm: Gõ dấu ">>" sau 1 câu lệnh và gõ tên file sau dấu >> để nối thêm đầu ra của câu lệnh vào file đó

Ví dụ: echo "something2" >> sth.txt

```
(~/playground) —————
(20:10:17)→ echo "something" > sth.txt
(~/playground) —————
(20:12:53)→ cat sth.txt
something
(~/playground) —————
(20:12:56)→ echo "something2" >> sth.txt
(~/playground) —————
(20:16:02)→ cat sth.txt
something
something2
(~/playground) —————
(20:16:05)→ |
```

Giải thích: dấu ">>" sẽ append từ stdout đến file mà mình chọn, ở đây chọn là file sth.txt. Câu lệnh echo bình thường sẽ in ra stdout mà ở đây ta dùng ">>" nên file sth.txt giờ đây có thêm dòng something2.

Chuyển hướng đầu vào: <

Cách làm: Gõ dấu "<" sau 1 câu lệnh và gõ tên file sau dấu < để chuyển hướng đầu vào file đó vào câu lệnh( câu lệnh lấy input là content của file đó)

Ví dụ: sort < sth.txt

```
(~/playground)
(20:19:28)→ cat sth.txt
something
something15
something2
something-1
(~/playground)
(20:19:29)→ sort < sth.txt
something
something-1
something15
something2
(~/playground)
(20:19:33)→ |
```

Giải thích: dấu “<” sẽ chuyển hướng từ stdin của 1 file vào câu lệnh. Nên sau khi chuyển hướng đầu vào của file vào câu lệnh sort. Câu lệnh sort sẽ in ra content của file sau khi được sort

Here document: <<

Cách làm: Gõ dấu “<<” sau 1 delimiter và sau đó gõ các dòng cho câu lệnh đó. Việc gõ vào sẽ dừng khi gõ tới delimiter

```
something2
[ (~/playground) —————
[ (20:19:33) —> sort << turtle
—(heredoc) — 3
—(heredoc) — 5
—(heredoc) — 2
—(heredoc) — 1
—(heredoc) — turtle
1
2
3
5
[ (~/playground) —————
[ (20:23:22) —> |
```

Giải thích: ở đây chọn delimiter là turtle, hàm sort sẽ sort các số được nhập vào.

Chuyển hướng lỗi: 2>

Tương tự như > nhưng mà ở đây thay vì là chuyển stdout vào file thì ta sẽ chuyển stderr vào file

```
(~/playground)
(20:25:19)→ abcdefgh 2> sth.txt
(~/playground)
(20:25:30)→ cat sth.txt
zsh: command not found: abcdefgh
(~/playground)
(20:25:34)→ |
```

Giải thích: ở đây gõ 1 lệnh không tồn tại trong máy là abcdefgh, và chuyển hướng lỗi vào file sth.txt. Ta sẽ không thấy báo lỗi ngoài terminal, thay vì đó nó sẽ báo lỗi vào trong file sth.txt

Nói thêm lỗi: 2>>

```
(~/playground)
(20:25:34)→ abcdef 2>> sth.txt
(~/playground)
(20:26:55)→ cat sth.txt
zsh: command not found: abcdefgh
zsh: command not found: abcdef
(~/playground)
(20:26:57)→ |
```

Giải thích: tương tự như >>, nhưng 2>> sẽ nối thêm lỗi vào file chứ không xóa hết content của file rồi mới viết vào như 2>

Chuyển hướng cả đầu ra và lỗi: &>

```
(~/playground) —
(20:28:55) —> (echo "fun stuff" && abcd) &> sth.txt
(~/playground) —
(20:29:06) —> cat sth.txt
fun stuff
zsh: command not found: abcd
(~/playground) —
(20:29:13) —> |
```

Giải thích: ở đây sẽ chuyển hướng cả đầu ra( từ command echo) và chuyển hướng lỗi( từ command không tồn tại abcd) vào file sth.txt. Cần phải có dấu ngoặc, nếu không nó chỉ hiểu là thực hiện command echo, sau đó thực hiện command abcd và chuyển hướng đầu ra và lỗi vào trong file sth.txt

Nói thêm cả đầu ra và lỗi: &>>

```
(~/playground) —
(20:29:13) —> (echo "more fun stuff" && abcde) &>> sth.txt
(~/playground) —
(20:31:17) —> cat sth.txt
fun stuff
zsh: command not found: abcd
more fun stuff
zsh: command not found: abcde
(~/playground) —
(20:31:22) —> |
```

Giải thích: &>> thực hiện giống &> nhưng sẽ nối vào file thay vì xóa file và viết từ đầu.

## 2. Classwork 02

Giải thích command

ps aux | grep apache | awk '{print \$2}' | xargs kill -9

ps aux là để hiện ra tất cả các process, trong đó:



Báo cáo thực hành môn Hệ điều hành - Giảng viên: Thân Thế Tùng.

a = show tất cả các process của các user

u = show user/owner

x = show process không thuộc terminal đó

grep apache để lấy dòng có chữ apache.

awk '{print \$2}' là để in ra các dòng có chữ apache sao cho chỉ lấy cột thứ 2( lấy pid)

xargs kill -9 là để force kill pid vừa lấy được ở trên.

### 3. Classwork 03

Cách làm: dùng read để nhập vào thông tin, so sánh và rẽ nhánh đơn giản bằng if để so thông tin nhập vào với thông tin đã có sẵn để kiểm tra người dùng.

```
(~/IT007)
(20:40:03)→ ./auth.sh
Nhap ten:
Thang
Nhap MSSV:
23521421
Yello
(~/IT007)
(20:40:11)→ |
```

Giải thích: script sẽ dùng #! hay shebang để chỉ đến /bin/bash để interpret script. Sau đó dùng chmod +x auth.sh để cho quyền execute và dùng ./auth.sh để chạy script. Script này chỉ có dùng if và so sánh string cơ bản.

#### 4. Classwork 04

Cách làm: Kiểm tra chỉ có 1 argument là điểm, kiểm tra điểm chỉ là số từ khoảng 0 đến 1000, và sử dụng if else để chuyển điểm số sang điểm chữ

```
(~/IT007) —
(20:49:53) —> ./mark_convert.sh
Hay nhập điểm( 1 số từ command line argument)
(~/IT007) —
(20:49:56) —> ./mark_convert.sh 845
Score: 845 => Grade: A
(~/IT007) —
(20:50:00) —> |
```

```
vim mark_convert.sh
1 #!/bin/bash
2
3 if [ "$#" -ne 1 ]; then
4     echo "Hay nhập điểm( 1 số từ command line argument)"
5     exit 1
6 fi
7
8 score=$1
9
10 if ! [[ "$score" =~ ^[0-9]+$ ]] || [ "$score" -lt 0 ] || [ "$score" -gt 1000 ]; then
11     echo "Hay nhập điểm trong khoảng từ 0 đến 1000"
12     exit 1
13 fi
14
15 if [ "$score" -ge 900 ] && [ "$score" -le 1000 ]; then
16     grade="A+"
17 elif [ "$score" -ge 800 ] && [ "$score" -lt 900 ]; then
18     grade="A"
19 elif [ "$score" -ge 700 ] && [ "$score" -lt 800 ]; then
20     grade="B+"
21 elif [ "$score" -ge 600 ] && [ "$score" -lt 700 ]; then
22     grade="B"
23 elif [ "$score" -ge 500 ] && [ "$score" -lt 600 ]; then
24     grade="C"
25 else
26     grade="D/F"
27 fi
28
29 echo "Score: $score => Grade: $grade"
```

Báo cáo thực hành môn Hệ điều hành - Giảng viên: Thân Thế Tùng.

Giải thích: Lần đầu tiên chỉ chạy script mà không viết gì thêm nên không có command line argument => script không chạy được. Lần thứ 2 chạy với argument 845 nên đã đưa ra được kết quả là điểm A

## 5. Classwork 05

Cách làm: dùng for loop theo kiểu C cho yêu cầu thứ 2 và while loop( dùng while true), break khi người dùng nhập đúng( cho yêu cầu thứ nhất)

```
1 #!/bin/bash
2
3 while true
4 do
5     echo "Nhap ten: "
6     read ten
7     echo "Nhap MSSV: "
8     read mssv
9
10    if [ $ten = "Thang" ] && [ $mssv = "23521421" ]
11    then
12        echo "Yello"
13        break
14    else
15        echo "access denied"
16        echo "Hay nhap lai"
17    fi
18 done
19 exit 0
20
```

```
1 #!/bin/bash
2
3 for ((i=0; i<5; i++))
4 do
5     echo "Nhap ten: "
6     read ten
7     echo "Nhap MSSV: "
8     read mssv
9
10    if [ $ten = "Thang" ] && [ $mssv = "23521421" ]
11    then
12        echo "Yello"
13        exit 0
14    else
15        echo "access denied"
16        if [ $i -lt 4 ]
17        then
18            echo "Hay nhap lai"
19        else
20            echo "Ban da het lan thu"
21        fi
22    fi
23 done
24 exit 0
25
~
~
```

```
(~/IT007)
(21:02:54) → ./auth2.sh
Nhap ten:
Thang
Nhap MSSV:
123
access denied
Hay nhap lai
Nhap ten:
3
Nhap MSSV:
123
access denied
Hay nhap lai
Nhap ten:
Thang
Nhap MSSV:
23521421
Yello
```

```
(21:04:17)→ ./auth3.sh
Nhap ten:
123
Nhap MSSV:
3
access denied
Hay nhap lai
Nhap ten:
123
Nhap MSSV:
3
access denied
Hay nhap lai
Nhap ten:
123
Nhap MSSV:
3
access denied
Hay nhap lai
Nhap ten:
123
Nhap MSSV:
3
access denied
Ban da het lan thu
(~/IT007)
(21:04:25)→ |
```

Giải thích: 2 ảnh đầu là code cho 2 script được yêu cầu. Ảnh thứ 3 là khi dùng vòng lặp while loop, chương trình sẽ hỏi đến khi nhập đúng. Ảnh thứ 4 là khi dùng vòng lặp for loop và chỉ có 5 lần thử.

## II. HOMEWORK

### 1. Homework 01

Cách làm: dùng chuyển hướng đầu vào >

```
turtle@EvilBrewHause:~/IT007/Lab2
[~/IT007/Lab2]—
(21:09:38)→ ls -la /etc > report_23521421.txt
[~/IT007/Lab2]—
(21:09:45)→ head report_23521421.txt
total 1640
drwxr-xr-x 110 root root 12288 16:42 2 Thg 10 .
drwxr-xr-x 18 root root 4096 06:59 10 Thg 4 ..
-rw-r--r-- 1 root root 44 25 Thg 3 2024 adjtime
drwxr-xr-x 3 root root 4096 25 Thg 3 2024 alsa
-rw-r--r-- 1 root root 541 22:23 8 Thg 4 anacrontab
drwxr-xr-x 2 root root 4096 22:04 15 Thg 9 apparmor.d
-rw-r--r-- 1 root root 1 01:02 8 Thg 4 arch-release
-rw-r--r-- 1 root root 0 23:01 3 Thg 7 arptables.conf
drwxr-xr-x 2 root root 4096 12 Thg 3 2024 audisp
[~/IT007/Lab2]—
(21:09:48)→ |
```

Giải thích: dùng `ls -la` để in ra chi tiết chế độ của tất cả các file, sau đó redirect về file `report_23521421.txt` và dùng command `head` để lấy ra các dòng đầu tiên trong file.

## 2. Homework 02

Cách làm: dùng hàm `sort` và chuyển hướng nối `>>`

```
turtle@EvilBrewHause:~/IT007/Lab2 | 147
[~/IT007/Lab2]—
(21:11:44)→ sort report_23521421.txt >> report_sorted.txt
[~/IT007/Lab2]—
(21:12:01)→ head report_
head: cannot open 'report_' for reading: No such file or directory
[~/IT007/Lab2]—
(21:12:03)→ head report_sorted.txt
drwx----- 2 root root 4096 14:46 16 Thg 8 cryptsetup-keys.d
drwx----- 2 root root 4096 4 Thg 3 2024 credstore
drwx----- 2 root root 4096 4 Thg 3 2024 credstore.encrypted
drwxr-x--- 2 root root 4096 25 Thg 3 2024 sudoers.d
drwxr-xr-x 110 root root 12288 16:42 2 Thg 10 .
drwxr-xr-x 12 root root 4096 22:05 1 Thg 10 xdg
drwxr-xr-x 18 root root 4096 06:59 10 Thg 4 ..
drwxr-xr-x 2 root root 4096 08:40 6 Thg 7 iptables
drwxr-xr-x 2 root root 4096 08:49 28 Thg 8 cifs-utils
drwxr-xr-x 2 root root 4096 08:49 28 Thg 8 request-key.d
[~/IT007/Lab2]—
(21:12:06)→ |
```

Báo cáo thực hành môn Hệ điều hành - Giảng viên: Thân Thế Tùng.

Giải thích: dùng sort report\_23521421.txt để sort file đó, và sử dụng >> report\_sorted.txt để chuyển hướng nối vào file report\_sorted.txt

### 3. Homework 03

Cách làm: dùng command wc và chuyển hướng đầu vào, đầu ra( <, >)

```
(~/IT007/Lab2) —————
(21:12:06) —> (wc -c < report_23521421.txt) > field2.txt
(~/IT007/Lab2) —————
(21:25:48) —> cat field2.txt
13494
(~/IT007/Lab2) —————
(21:25:51) —> |
```

Giải thích: dùng command wc -c để tìm số từ trong file report, ta chuyển hướng đầu vào của file report vào command wc -c để lấy số từ trong file. Sau đó chuyển hướng đầu ra vào file field2.txt. Kết quả là có 13494 từ.

### 4. Homework 04

Cách làm: dùng command tail, wc và pipe.

```
(~/IT007/Lab2) —————
(21:28:55) —> tail -9 report_23521421.txt | wc -c
505
(~/IT007/Lab2) —————
(21:29:00) —> |
```

Giải thích: dùng command tail -9 để lấy 9 dòng cuối cùng của file report\_23521421.txt. Sau đó pipe qua command wc -c để tính số từ có trong 9 dòng đó, kết quả là được 505 từ.

### 5. Homework 05

Cách làm: chọn IFS là “;” đọc từng dòng của file, khi nào student\_id = id thì in ra kết quả và break.

```
vim grade.sh
6 fi
7
8 student_id=$1
9 IFS=';'
10
11 while read -r id name score; do
12     if [ $id = "StudentID" ]; then
13         continue
14     fi
15
16     if [ $id = $student_id ]; then
17
18         if [ "$score" -ge 900 ] && [ "$score" -le 1000 ]; then
19             grade="A+"
20         elif [ "$score" -ge 800 ] && [ "$score" -lt 900 ]; then
21             grade="A"
22         elif [ "$score" -ge 700 ] && [ "$score" -lt 800 ]; then
23             grade="B+"
24         elif [ "$score" -ge 600 ] && [ "$score" -lt 700 ]; then
25             grade="B"
26         elif [ "$score" -ge 500 ] && [ "$score" -lt 600 ]; then
27             grade="C"
28         else
29             grade="D/F"
30         fi
31         echo "Tên: $name"
32         echo "Điểm gốc: $score"
33         echo "Điểm chữ: $grade"
34         break
35     fi
36
37 done < gradebook.csv
```



```
└─(21:48:48)─> ./grade.sh 23560287
: integer expression expected
: integer expression expected
: integer expression expected
: integer expression expected
: integer expression expected
Tên: Student 15
Điểm gốc: 674
Điểm chữ: D/F
└─(~/IT007/Lab2)─
└─(21:48:50)─> l;s
```

Giải thích: sử dụng lệnh read để đọc 1 dòng 1 lần vào 3 tham số là id, name và score. Để đọc file này thì ta file redirect input từ file vào vòng lặp while, với mỗi dòng thì ta so ID, nếu ID đúng thì ta sẽ in thông tin sinh viên ra và break.

## 6. Homework 06

Cách làm: dùng regex để match tuổi và dùng if else để xác định độ tuổi

```
vim age.sh
1 #!/bin/bash
2
3 read -p "Nhập tuổi: " age
4
5 if ! [[ "$age" =~ ^[0-9]+$ ]] || [ "$age" -lt 0 ] || [ "$age" -gt 100 ]; then
6     echo "Tuổi là giá trị số từ 0 đến 100"
7 else
8     if [ "$age" -lt 12 ]; then
9         echo "Bạn là một đứa trẻ."
10    elif [ "$age" -ge 12 ] && [ "$age" -le 18 ]; then
11        echo "Bạn là một thiếu niên."
12    else
13        echo "Bạn là một người lớn."
14    fi
15 fi
16
```

```
(~/IT007/Lab2)
(21:53:33)→ ./age.sh
Nhap tuoi: 66
Bạn là một người lớn.
(~/IT007/Lab2)
(21:53:37)→ |
```

Giải thích: regex được dùng trong bài có nghĩa là:

^ là bắt đầu của string

^[0-9] tức là tuổi phải bắt đầu bằng 1 số

^[0-9]+\$ tức là tuổi có ít nhất 1 số và string kết thúc chỉ có tuổi

2 mệnh đề ở phía sau đảm bảo tuổi phải không nhỏ hơn 0 và không lớn hơn 100.

## 7. Homework 07

Cách làm: sử dụng command line argument để lấy search string và directory. Sau đó lặp qua tất cả các file trong directory đó, tìm string đó bằng command grep

```
vim hw
1 #!/bin/bash
2
3 if [ $# -ne 2 ]; then
4     echo "Command line argument gom string va duong dan"
5     exit 1
6 fi
7
8 search_string=$1
9 directory=$2
10
11 if [ ! -d "$directory" ]; then
12     echo "Khong ton tai $directory"
13     exit 1
14 fi
15
16 for file in "$directory"/*; do
17     if [ -f "$file" ]; then
18         found=$(grep "$search_string" "$file")
19         if [ -n "$found" ]; then
20             echo $file
21             echo $found
22             echo "-----"
23         fi
24     fi
25 done
~
~
```

Giải thích: các -ne -d -f -n trong if lần lượt có ý nghĩa là not equal, check là một directory, check là 1 file, và check là 1 string không rỗng.

### III. BONUS

#### 1. Bonus 01

Cách làm: dùng git clone {url} để tải từ github, sau đó dùng hàm ls -l và lấy số hàng trong output (là số file + 1). Rồi in ra kết quả.

```
1 #!/bin/bash
2
3 if [ ! -d OS_LAB2_IMG ]
4 then
5     git clone https://github.com/locth/OS_LAB2_IMG.git
6 else
7     echo "Da co directory OS_LAB2_IMG"
8 fi
9
10 cd OS_LAB2_IMG
11
12 if [ ! -d PNG ]
13 then
14     mkdir PNG
15     mv *.png ./PNG
16     num_of_png=$((ls -l PNG | wc -l) - 1)
17     echo "Co $num_of_png file png"
18 fi
19
20 if [ ! -d JPG ]
21 then
22     mkdir JPG
23     mv *.jpg ./JPG
24     ls1=$(ls -l JPG)
25     num_of_jpg=$((ls -l JPG | wc -l) - 1)
26     echo "Co $num_of_jpg file jpg"
27 fi
28
~
```

```
(~/IT007/Lab2)
(08:40:10)→ ./bonus01.sh
Da co directory OS_LAB2_IMG
Co 18 file png
Co 60 file jpg
(~/IT007/Lab2)
(08:40:12)→ |
```

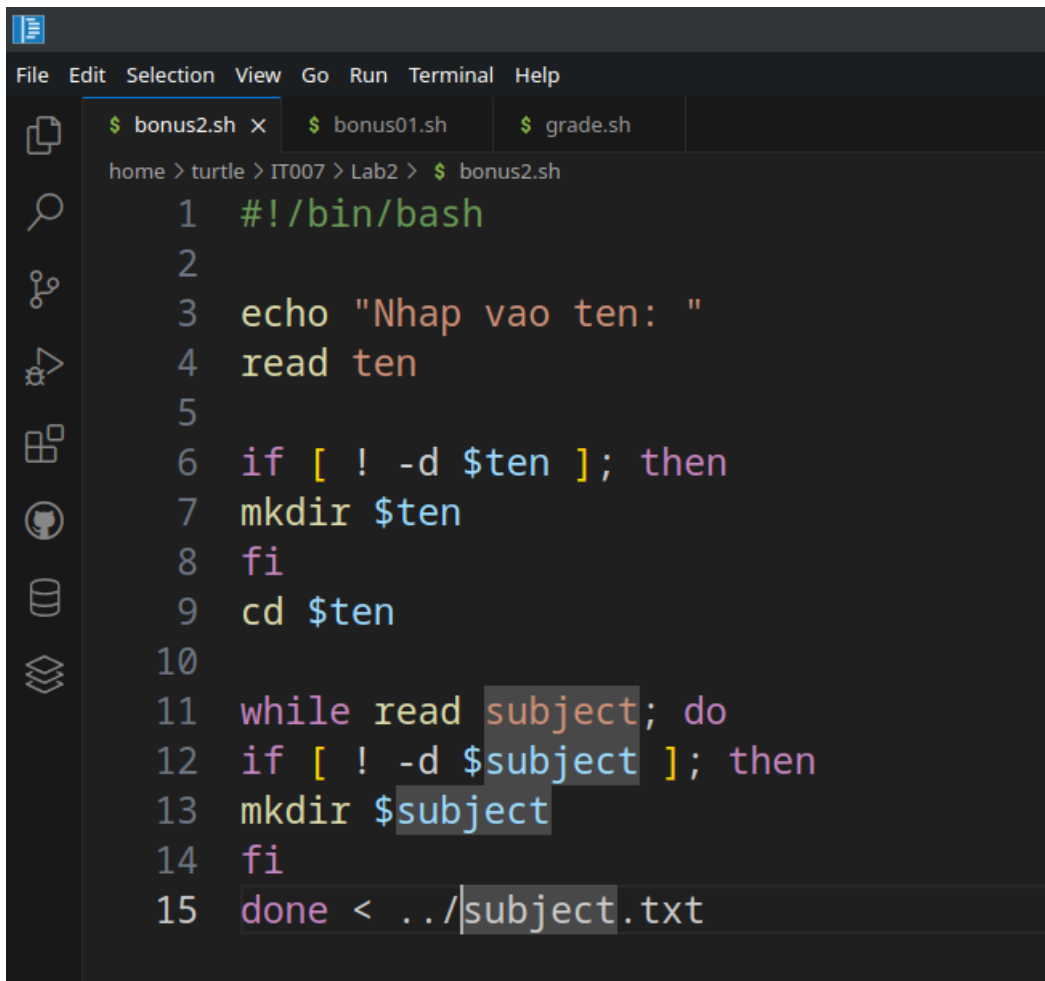
Báo cáo thực hành môn Hệ điều hành - Giảng viên: Thân Thế Tùng.

Giải thích: trước đó em đã clone directory về rồi nên đã có dòng if detect là đã có local repo => ko clone nữa. Sau đó lần lượt kiểm tra directory có tồn tại hay không bằng if [ -d directory ] và dùng \*.png, \*.jpg để match tất cả các file có đuôi là png, jpg. Câu lệnh ls sẽ ra 1 hàng metadata nữa nên phải trừ bớt đi 1 để ra đúng số file.

## 2. Bonus 02

Cách làm: dùng lệnh while read var; do {code} done < file.txt

để đọc lần lượt các môn học có trong file subject.txt. Sau đó lần lượt kiểm tra nếu directory đó chưa tồn tại thì ta sẽ tạo với command mkdir.



```
File Edit Selection View Go Run Terminal Help
$ bonus2.sh x $ bonus01.sh $ grade.sh
home > turtle > IT007 > Lab2 > $ bonus2.sh
1  #!/bin/bash
2
3  echo "Nhap vao ten: "
4  read ten
5
6  if [ ! -d $ten ]; then
7  mkdir $ten
8  fi
9  cd $ten
10
11 while read subject; do
12 if [ ! -d $subject ]; then
13 mkdir $subject
14 fi
15 done < ../subject.txt
```

```
(~/IT007/Lab2) —
(19:49:55) —> ./bonus2.sh
Nhập vào tên:
ThangDQ
(~/IT007/Lab2) —
(19:50:04) —> ls -R ThangDQ
ThangDQ:
CS112  CS115  IT002  IT004  IT007  SS007

ThangDQ/CS112:


ThangDQ/CS115:

ThangDQ/IT002:

ThangDQ/IT004:

ThangDQ/IT007:

ThangDQ/SS007:
(~/IT007/Lab2) —
(19:50:09) —>
```

A terminal window with a dark blue background. The prompt is ~/IT007/Lab2. The user runs ./bonus2.sh, which prompts for a name. The user enters ThangDQ. Then the user runs ls -R ThangDQ, which lists the contents of the ThangDQ directory: CS112, CS115, IT002, IT004, IT007, and SS007. Below this, the terminal shows the paths ThangDQ/CS112:, ThangDQ/CS115:, ThangDQ/IT002:, ThangDQ/IT004:, ThangDQ/IT007:, and ThangDQ/SS007: in sequence. At the bottom, there is a taskbar with icons for a file explorer, a terminal, a web browser, and several other applications.

Giải thích: trong file subject.txt có 6 môn mà em học kì này: CS112, CS115, IT002, IT004, IT007 và SS007. Sau khi chạy script ở trên(với tên e nhập là ThangDQ) thì em dùng lệnh `ls -R ThangDQ` để hiển thị toàn bộ cây directory của ThangDQ, và nó hiện ra 6 subdir đã được tạo.