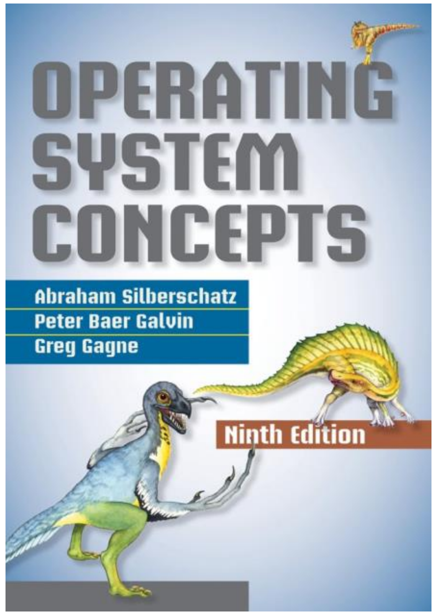
Lecture 3: Shell Programming

Lecturer: Prof. Zichen Xu

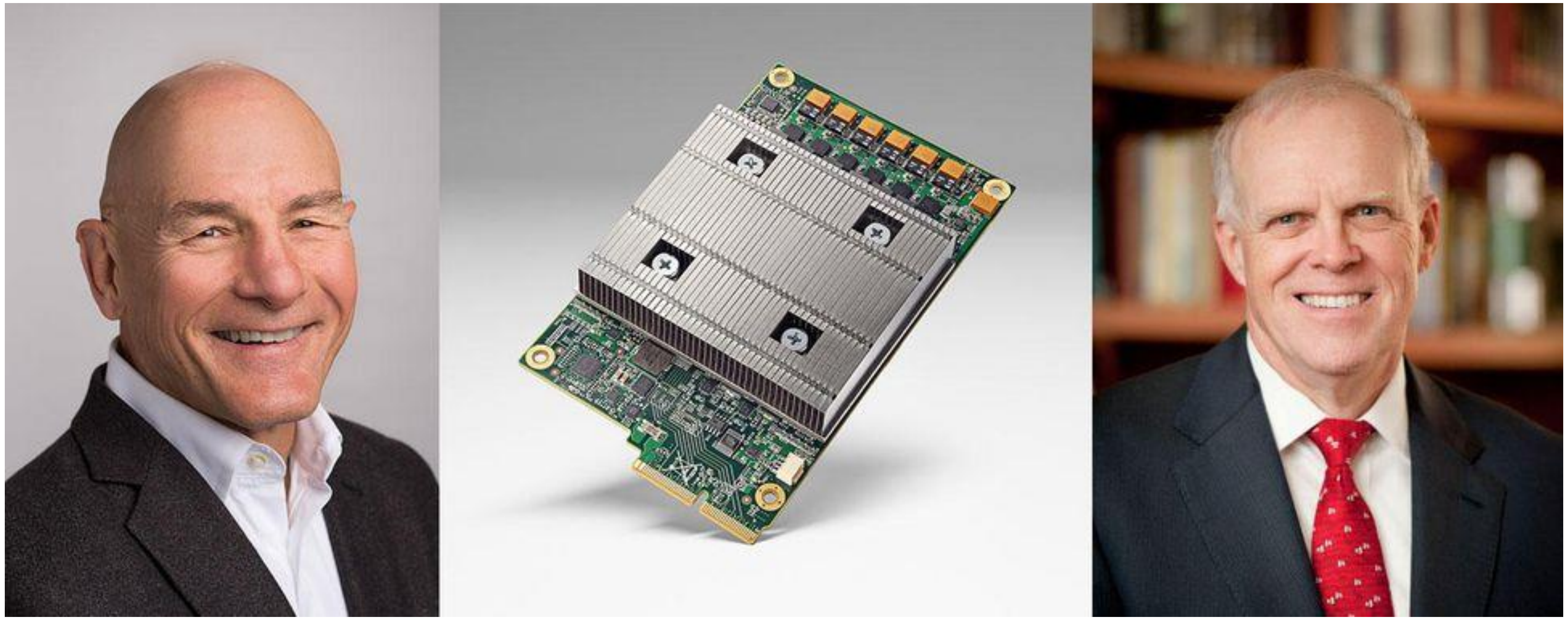
Books

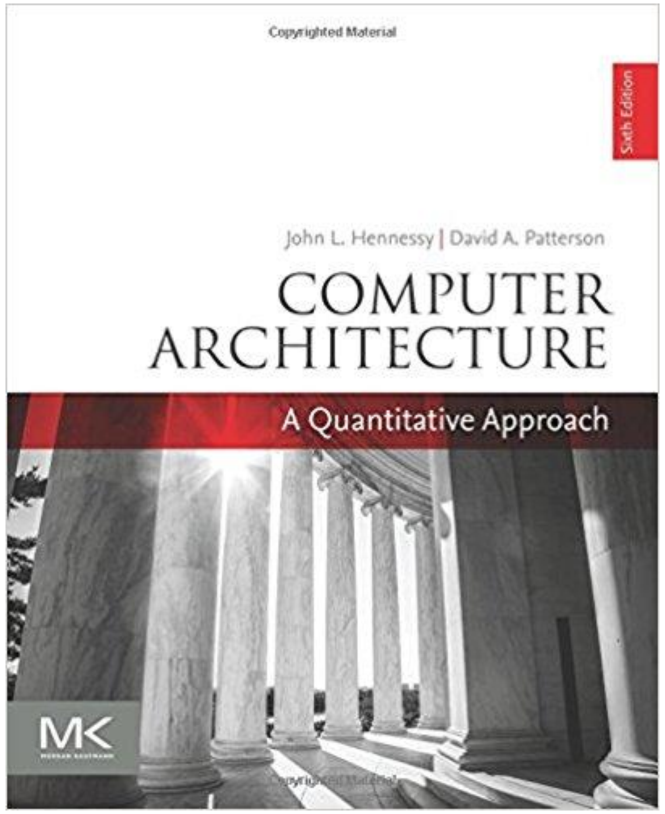
- 于渊，《自己动手写操作系统》，电子工业出版社，2005-08-01，ISBN：9787121015779
- Abraham Silberschatz, Peter B. Galvin, Greg Gagne, 《Operating System Concepts》，Wiley, 2012-12-17, ISBN: 9781118063330



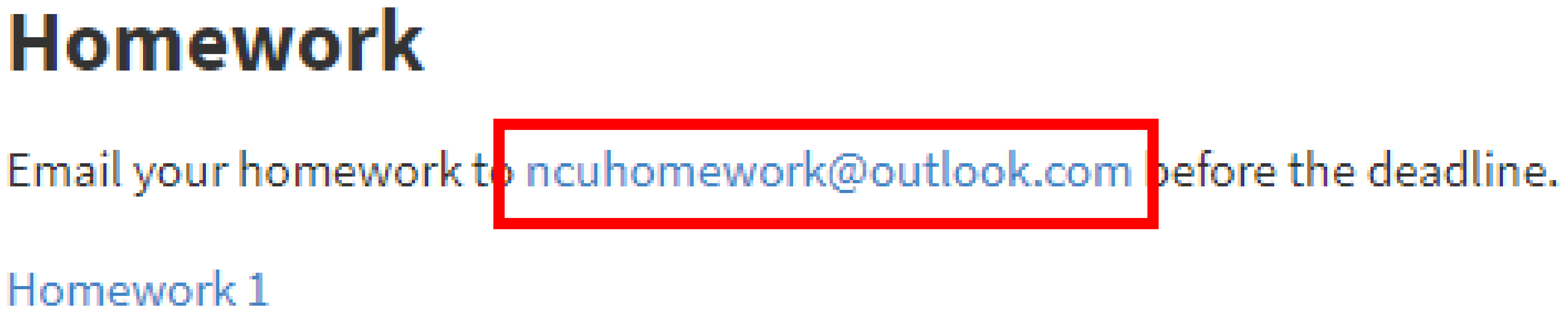
Books

- John L. Hennessy, David A. Patterson, 《Computer Architecture, Sixth Edition: A Quantitative Approach》, Morgan Kaufmann, ISBN-13: 978-0128119051





Homework



Homework

Email your homework to ncuhomework@outlook.com before the deadline.

Homework 1

Outline

- Structure of a shell command
- Write a shell command
- Execute a shell command
- Configure a shell command
- Use shell commands as functions

Concurrent/Multiple Execution

- We have introduced the concurrency in Linux
  - ;
  - &
  - &&

- [["a" = "b"]] && echo ok

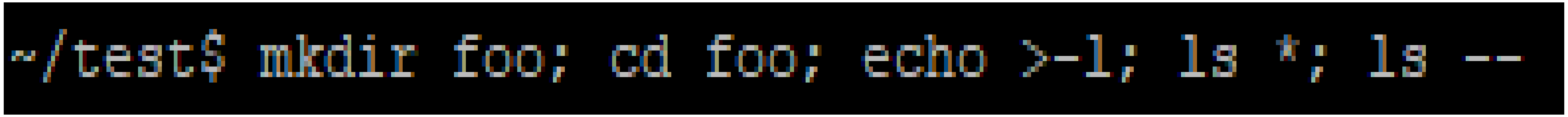
- [["a" = "b"]]; echo ok

Batching

- Shell script is a way to utilize these functions
    - ls a* -l; free;df; echo "this is long command"
    - ls a* -l; free;df; echo "this is \

      > long command"

- Like other systems, Linux usually batches all the commands in order.



```
~/test$ mkdir foo; cd foo; echo >-l; ls *; ls --
```

One Shell Command Example

```
#!/bin/sh
echo "Mr.$USER,Today is:"
echo &date  "+%B%d%A"
echo "Whish you a lucky day !"
```

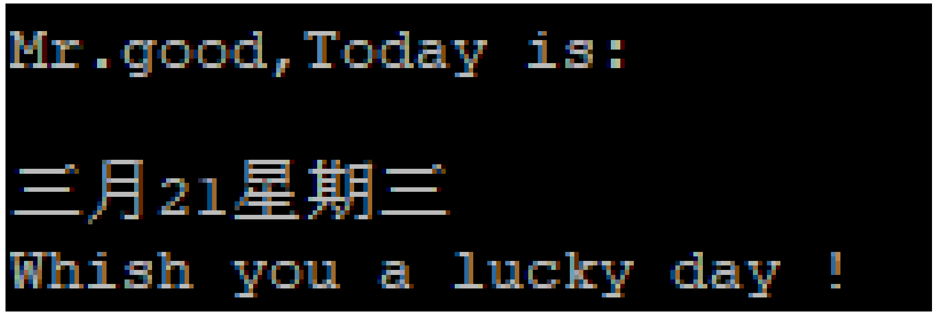
- #!/bin/sh indicates a bash shell command
- echo controls the output from date
- %B%d%A represents the output format

How to Execute Such a Script?

- Make it executable

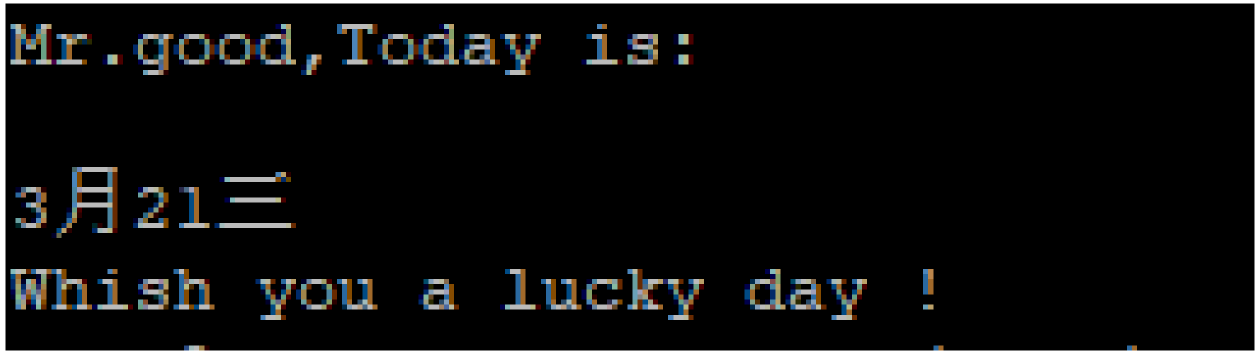
- And execute it
  - ./test
  - bash test
  - Make it into an environment variable
  - Make the folder/directory into Path

Output



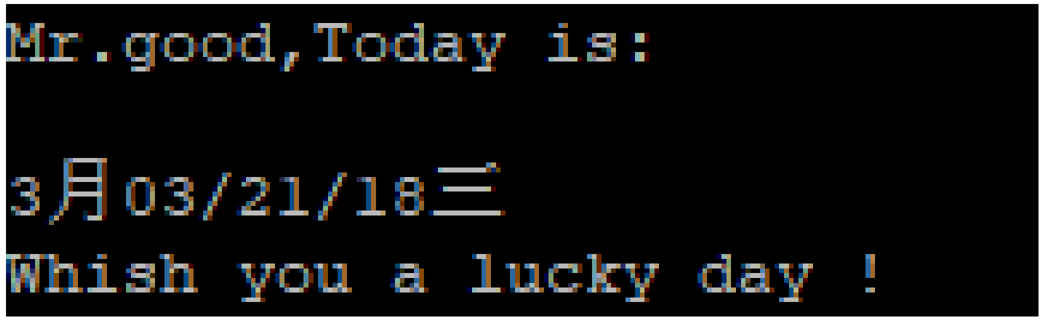
```
Mr.good,Today is:

三月21星期三
Whish you a lucky day !
```



```
Mr.good,Today is:

3月21三
Whish you a lucky day !
```



```
Mr.good,Today is:

3月03/21/18三
Whish you a lucky day !
```

Another Example

- Write a shell command to display the file information under /root. Then, create a folder called CS, make a file called exe, and make the file executable.

- What do you need to do?

Argc and Argv

- int main(int argc, char** argv)
- int main(int argc, char* argv[])

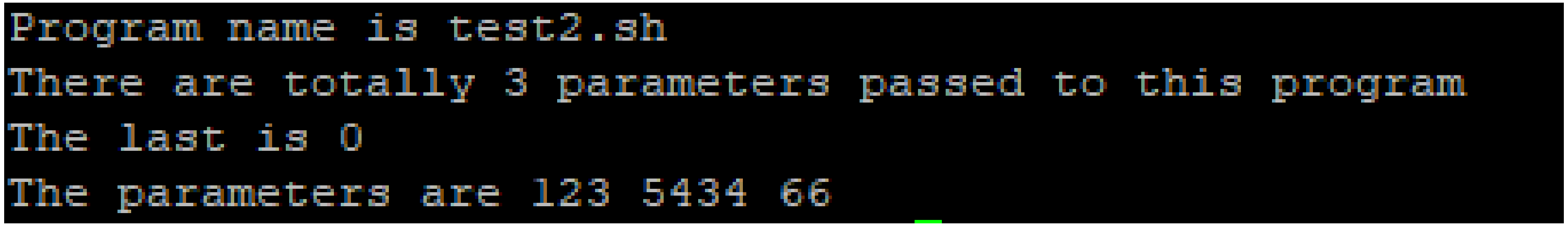
- What would happen in Linux?
  - $N

Default Variables in Linux

- $0
  - The directory
- $#
  - The total number of variables passed
- $?
  - The exit code
- $*
  - All strings/chars of all variables

One More Example

- For $0、$#、$?、$*， write a shell to test it



```
Program name is test2.sh
There are totally 3 parameters passed to this program
The last is 0
The parameters are 123 5434 66
```

The Input of a Program

- Create variables and their assignment
- Call and recall/reassign these variables
- Read your input string

Yet Another Example

```
#!/bin/sh
echo  "please input name of directory "
read  DIRECTORY
cd  $DIRECTORY
ls –l
```

Yet Another Example

```
#!/bin/sh
read x y
z=`expr $x + $y`
echo "The sum is $z"
```

Expression Comparison

- =
- !=
- -n
- -z

```
#!/bin/bash
read  ar1
read  ar2
[ "$ar1" = "$ar2" ]
echo $?
```

Logic Operations

- !
- -a
- -o

```
#!/bin/bash
part1 ="1111"
part2 =""    #part2为空
 [ "$part1"  -a  "$part2" ]
echo $?
 [ "$part1"  -o  "$part2" ]
```

Files

- -d
- -f
- -L
- -r
- -s
- -w
- -x

Another Example

```
#!/bin/bash
[ -d  /root/zb ]
echo $?
```

Expression Comparison

```
#!/bin/bash
echo “Please enter the directory name or file name”
read  DORF
if [ -d $DORF ]
then
  ls $DORF
elif [ -f $DORF]
then
cat $DORF
else
  echo “input error!”
fi
```

For-loop Expression

```
#!/bin/sh
for i in a,b,c,e,i   2,4,6,8
 do
  echo $i
 done
```

For-loop Expression

```
#!/bin/sh
data="a,b, c,d"
IFSBAK=$IFS
IFS=,
for item in $data
do
echo Item: $item
done
IFS=$IFSBAK
```

Add Them Together

```
#!/bin/sh
total=0
for ((j=1;j<=100;j++));
  do
     total=`expr $total + $j`
  done
echo "The result is $total"
```

While Loop

```
while *Expr*
  do
    *Operation*
  done
```

Float-point Calculation

```
#! /bin/bash
read n
total=0.000
an=0.000
for((num=1;num<=$n;num++));do
i=$num
if [ $i != 0 ]
then
an=`echo "scale=3;1.000/$i" | bc`
total=`echo "scale=3;$total+$an" | bc`
fi
done
echo $total
```

Select a Game

```
do
case $input in
1) gnomine;;
2) gnobots2;;
3) gtali;;
4) gnotski;;
5) gnibbles;;
6) gnotravex;
7) gnome-stones;
*) echo "Please selected 1\2\3\4\5\6\7\8 " ;;
esac
done
```

Functions

```
#!/bin/sh
add( )
{
 a=$1
 b=$2
 z=`expr $a + $b`
 echo "The sum is $z"
}
add  $1  $2
```

Conclusion

- We have started on shell programming
- We have discussed the syntax of shell programming
- We have categorized the shell codes and samples
- We will start on golang programming next week