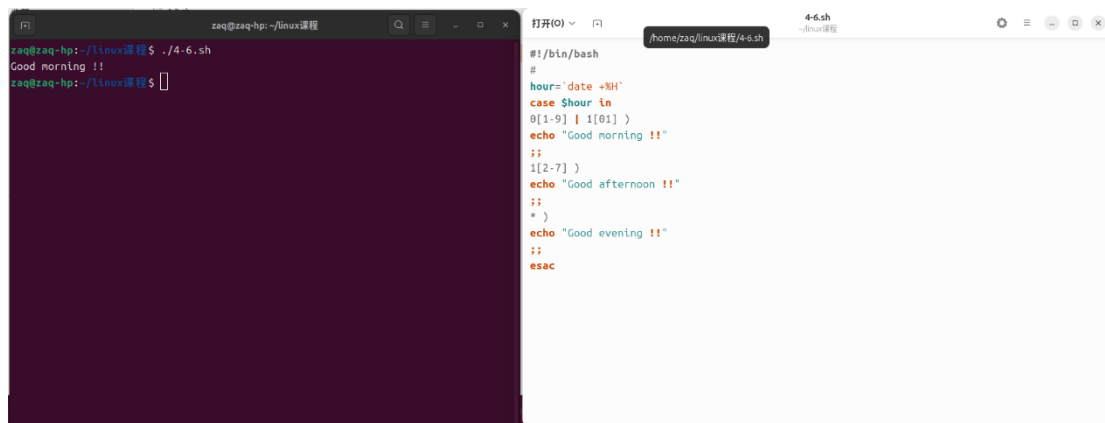


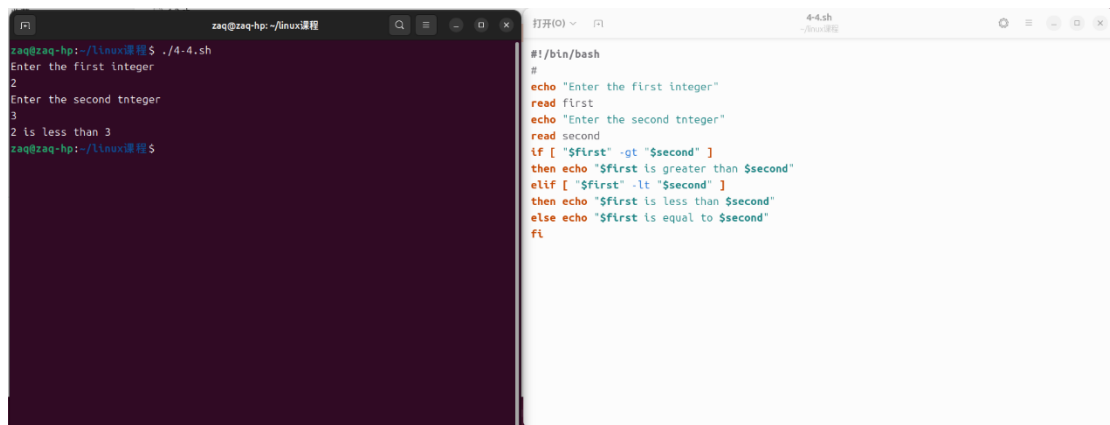
1. Obtain the system time, and check whether it is in the morning, afternoon, or evening.



```
zaq@zaq-hp: ~/linux课程
zaq@zaq-hp: ~/linux课程 $ ./4-6.sh
Good morning !!
zaq@zaq-hp: ~/linux课程 $
```

```
#!/bin/bash
#
hour=`date +%H`
case $hour in
0[1-9] | 1[01] )
echo "Good morning !!"
;;
1[2-7] )
echo "Good afternoon !!"
;;
* )
echo "Good evening !!"
;;
esac
```

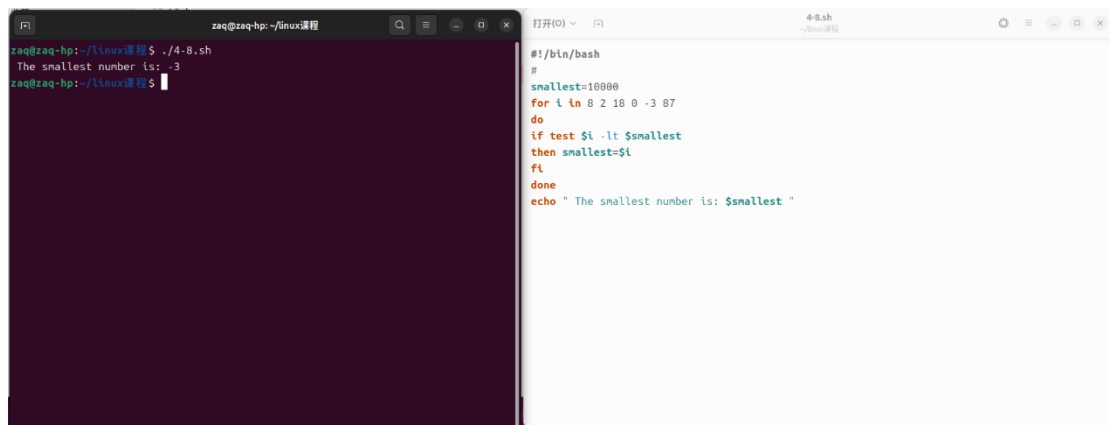
2. Input two number, check which one is greater, and output the result.



```
zaq@zaq-hp: ~/linux课程
Enter the first integer
2
Enter the second integer
3
2 is less than 3
zaq@zaq-hp: ~/linux课程 $
```

```
#!/bin/bash
#
echo "Enter the first integer"
read first
echo "Enter the second integer"
read second
if [ "$first" -gt "$second" ]
then echo "first is greater than $second"
elif [ "$first" -lt "$second" ]
then echo "first is less than $second"
else echo "first is equal to $second"
fi
```

3. Find the minimal value in a given list.

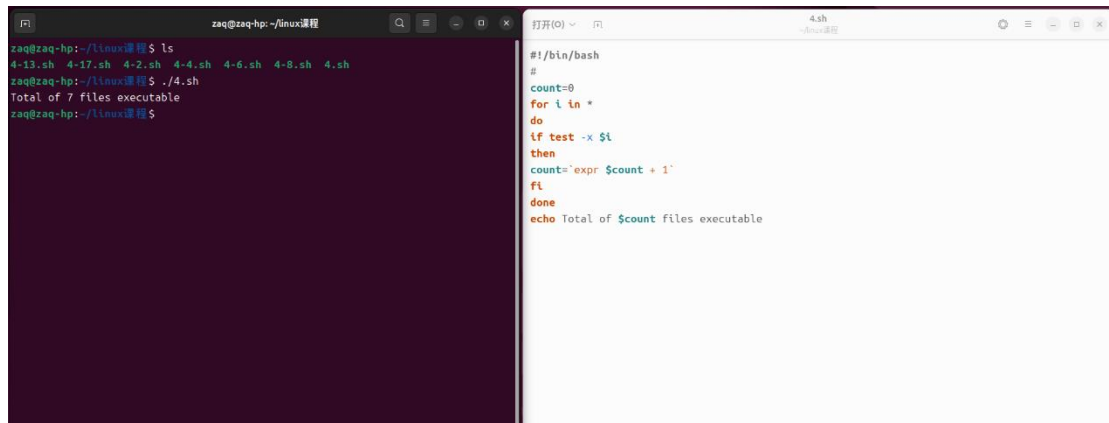


The image shows a terminal window on the left and a code editor on the right. The terminal window displays the execution of a script that finds the smallest number in a list. The code editor shows the script's source code.

```
zaq@zaq-hp: ~/linux课程
zaq@zaq-hp:~/linux课程 $ ./4-8.sh
The smallest number is: -3
zaq@zaq-hp:~/linux课程 $
```

```
#!/bin/bash
#
smallest=10000
for i in 8 2 10 0 -3 87
do
if test $i -lt $smallest
then smallest=$i
fi
done
echo " The smallest number is: $smallest "
```

4. Calculate the number of executive file in the current directory.

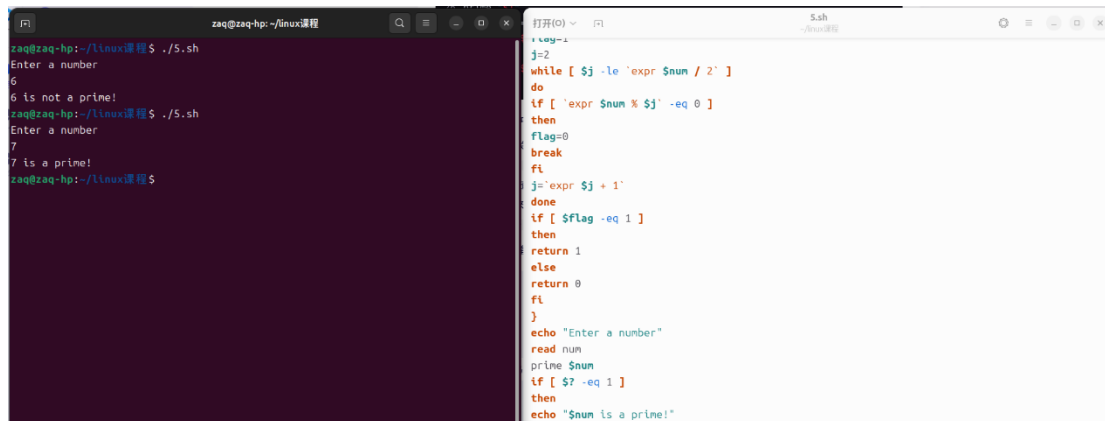


The image shows a terminal window with a dark background. The prompt is `zaq@zaq-hp: ~/linux课程`. The user has run `ls`, listing files `4-13.sh`, `4-17.sh`, `4-2.sh`, `4-4.sh`, `4-6.sh`, `4-8.sh`, and `4.sh`. Then they ran `./4.sh`, which printed "Total of 7 files executable".

On the right, a separate window titled `4.sh` shows the script code:

```
#!/bin/bash
#
count=0
for i in *
do
if test -x $i
then
count=`expr $count + 1`
fi
done
echo Total of $count files executable
```

5. Check whether a given number is a prime, you have to write a function, and call the function.



The image shows a terminal window on the left and a code editor on the right. The terminal window displays the execution of a script that checks if a number is prime. The code editor shows the script's source code, which uses a while loop to test divisibility from 2 up to the number itself. If a divisor is found, it returns 0 (not prime); otherwise, it returns 1 (prime).

```
zaq@zaq-hp: ~/linux课程
zaq@zaq-hp:~/linux课程$ ./5.sh
Enter a number
6
6 is not a prime!
zaq@zaq-hp:~/linux课程$ ./5.sh
Enter a number
7
7 is a prime!
zaq@zaq-hp:~/linux课程$
```

```
#!/bin/bash
flag=1
j=2
while [ $j -le `expr $num / 2` ]
do
    if [ `expr $num % $j` -eq 0 ]
    then
        flag=0
        break
    fi
    j=`expr $j + 1`
done
if [ $flag -eq 1 ]
then
    return 1
else
    return 0
fi
}
echo "Enter a number"
read num
prime $num
if [ $? -eq 1 ]
then
    echo "$num is a prime!"
fi
```