Lesson 4 Python Conditional Statement

1. Conditional Statement Introduction

Conditional statement controls the execution of different blocks of code through judging whether the condition holds and based on the result of condition expression.

1.1 Conditional Expression

Conditional expression consists of operator and operand. Take “**a<4**” for example. “**a**” and “**4**” are operand，and “**<**” is operator.

Judgment conditions can be any element with Boolean properties, including data, variables, and expressions composed of variables and operators. If its Boolean property is "True", the condition holds; if it is "False", the condition doesn’t hold.

The table lists the commonly used operators by conditional expression.

Type	Operator	Mathematical Symbol	Meaning
Arithmetic operator	+	+	add
	-	-	subtraction
	*	×	multiplication
	/	÷	Division, and the result is floating point number
	//		Division, and the result is a

			rounded down integer
	%	%	Modules/ Surplus
	**	^	Exponentiation
Comparison Operators	==	＝	Equal
	!=	≠	Not equal
	>	＞	Greater than
	<	＜	Less than
	>=	⩾	Greater than or equal to
	<=	⩽	Less than or equal to
Membership Operator	in	∈	Exist/ belong to
	not in	∉	Not exist/ not belong to

Python is able to combine the judgement condition logically through the reserved words “not”, “and” and “or”.

1） not: it represents the “**not**” relationship for an individual condition. If the Boolean property of the “**condition**” is “**True**”, the Boolean property of the “**not condition**” is “**False**”. If Boolean property of the “**condition**” is “**False**”, the Boolean property of the “**not condition**” is “**True**”

2） and: it represents “**and**” relationship between several conditions. Only when the Boolean properties of the conditions connected by “and” are **all “True”**, the Boolean property of the logic expression is “**True**”, otherwise

“**False**”.

3） or: it represents “**or**” relationship between several conditions. Only when the Boolean properties of the conditions connected by “or” are **all “False”**, the Boolean property of the logic expression is “**False**”, otherwise “**True**”.

1.2 Selection Statement

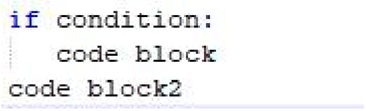
There are three types of selection statements, including single-branch selection structure, double-branch selection structure and multi-branch selection structure.

1) Single-branch selection structure

The syntax and execution process of single-branch selection structure are as follow.

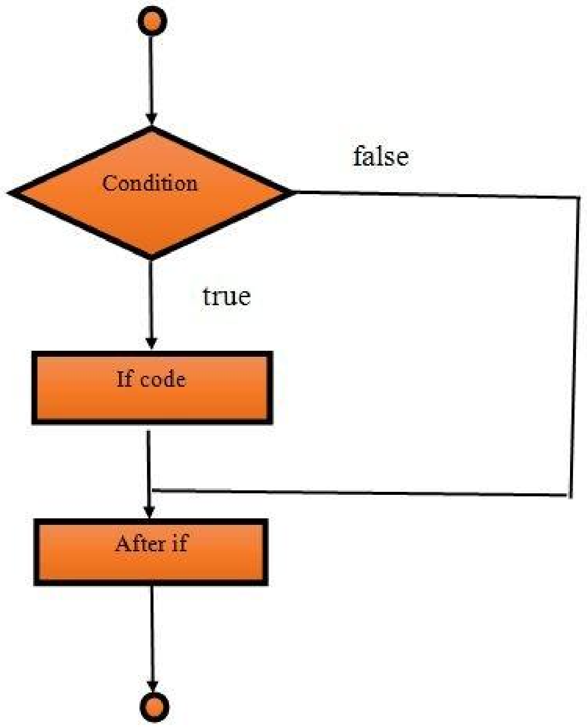
Syntax	Execution Process

<table>
<tr>
<td>



```
if condition:
    code block
code block2
```

</td>
<td>



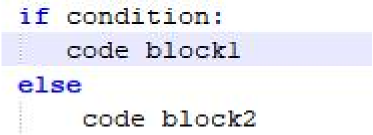
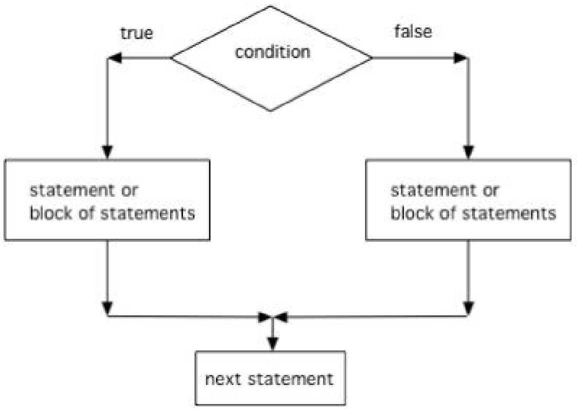
</td>
</tr>
</table>

If the condition of “if statement” is true, block of code 1 and 2 will be executed in sequence. Otherwise, skip block of code 1 and directly execute block of code 2.

2) Double-branch selection structure

The syntax and execution process of double-branch selection structure are as follow.

Syntax	Execution Process

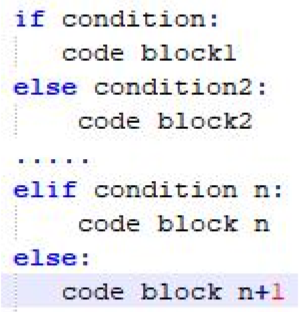
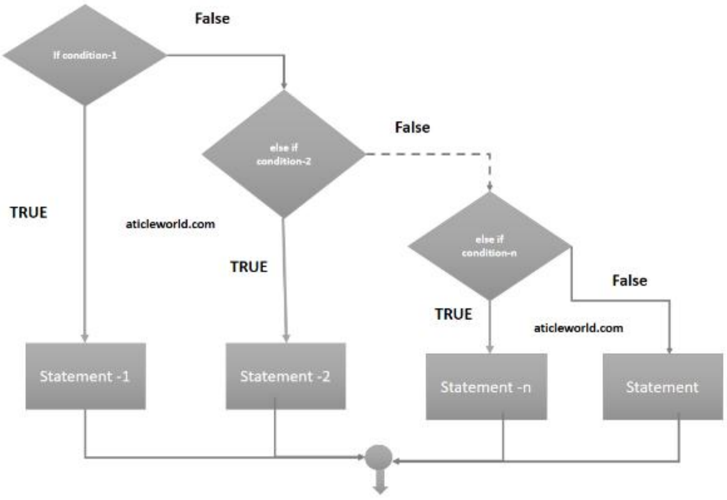
```
if condition:
    code block1
else
    code block2
```

condition
true
false
statement or block of statements
statement or block of statements
next statement

If the condition of “if statement” is true, block of code 1 will be executed. Otherwise, block of code 2 will be executed.

3) Multi-branch selection structure

The syntax and execution process of multi-branch selection structure are as follow.

Syntax	Execution Process
	

```
if condition:
    code block1
else condition2:
    code block2
.....
elif condition n:
    code block n
else:
    code block n+1
```

If the condition of “if statement” is true, block of code 1 will be executed.

If the condition of “if statement” is false, conditions of elif statements will be judged in sequence. When the condition is true, the corresponding block of code will be executed.

If neither conditions of “if statement” nor elif statement are false, block of code n+1 will be executed.

2. Operation Steps

In this routine, the program will calculate BMI value based on the input height and weight, and then perform health assessment.

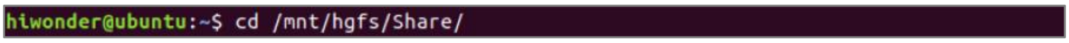
Before operation, please copy the routine “**conditional_statement.py**” stored in “**3. Python->Python Basic and Advanced Lesson->Lesson 4 Python Conditional Statement-> Routine Code”** to the shared folder.

For the configuration of shared folder, please refer to the file in “**2. Linux Introduction and Usage-> Linux Basic Lesson->Lesson 3 Linux Installation and Source Replacement**”.

Note: the input command should be case sensitive and keywords can be complemented by “Tab” key.

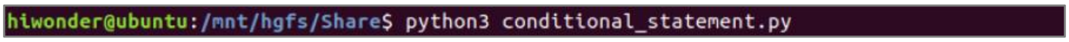
1） Start virtual machine, and click “⠿”, and then click “>_” or press “**Ctrl+Alt+T**” to open command line terminal.

2） Input “**cd /mnt/hgfs/Share/**” command and press Enter to enter the shared folder.



```
hiwonder@ubuntu:~$ cd /mnt/hgfs/Share/
```

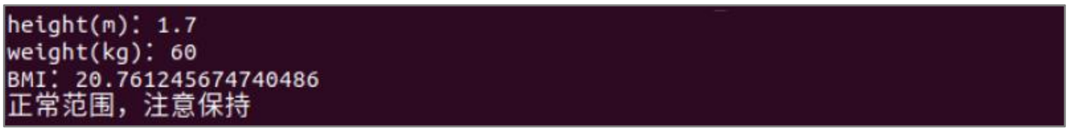
3） Input “**python3 conditional_statement.py**” command and press Enter to run the routine.



```
hiwonder@ubuntu:/mnt/hgfs/Share$ python3 conditional_statement.py
```

3. Program Outcome

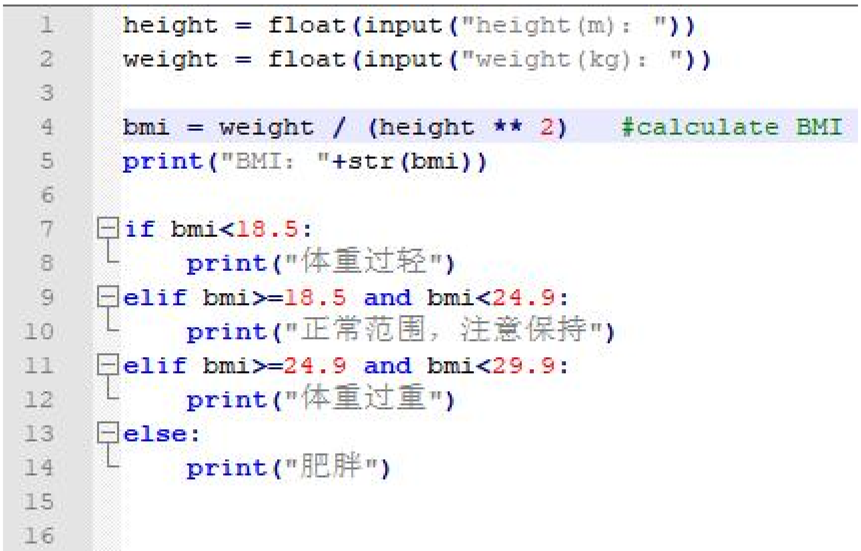
nput height and weight in sequence, and press Enter. Then the terminal will print the corresponding BMI and assessment result.



```
height(m): 1.7
weight(kg): 60
BMI: 20.761245674740486
正常范围，注意保持
```

4. Program Analysis

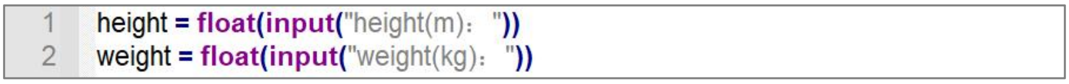
The routine can be found in “**4. OpenCV Computer Vision Lesson->Image Processing->Lesson 5 Image Processing--Morphological Processing->Routine Code**”.



```
height = float(input("height(m): "))
weight = float(input("weight(kg): "))

bmi = weight / (height ** 2)   #calculate BMI
print("BMI: "+str(bmi))

if bmi<18.5:
    print("体重过轻")
elif bmi>=18.5 and bmi<24.9:
    print("正常范围，注意保持")
elif bmi>=24.9 and bmi<29.9:
    print("体重过重")
else:
    print("肥胖")


```

1) Data input

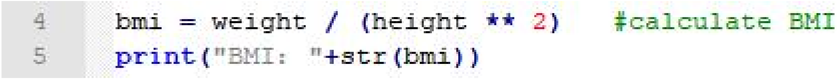
Call input() function to receive the input data. The hints are inside the parenthesis.



```
height = float(input("height(m)： "))
weight = float(input("weight(kg)： "))
```

2) Data calculation

Process the input data based on BMI calculation formula, and then print the result on the terminal through **print()** function.



```
bmi = weight / (height ** 2)    #calculate BMI
print("BMI： "+str(bmi))
```

The format of print() function is as follow.

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

The first parameter “**objects**” is the output object. When several objects are output at one time, objects should be separated by comma “**,**”.

The second parameter “**sep**” is used to insert the string between the objects. The default value is a space.

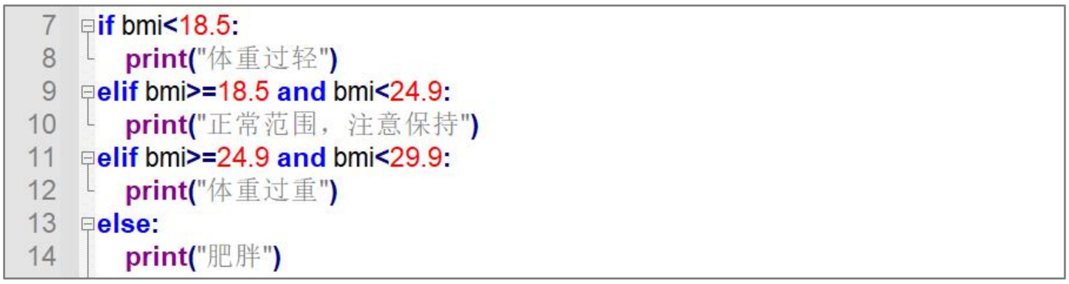
The third parameter “**end**” is used to add string at the end of the output. The default value is a line break.

The fourth parameter “**file**” is the object with a write function, the default value is “sys.stdout”, that is screen.

The fifth parameter “**flush**” is used to control and output cache. The default value is “**False**”.

3) Range Judgement

As there are two or more judgement results, multi-branch structure should be adopted.



```
if bmi<18.5:
    print("体重过轻")
elif bmi>=18.5 and bmi<24.9:
    print("正常范围，注意保持")
elif bmi>=24.9 and bmi<29.9:
    print("体重过重")
else:
    print("肥胖")
```

1） When BMI is less than 18.5, the corresponding BMI value and “underweight” will be printed.

2） When BMI is less than 24.9 and greater than or equal to 18.5, the corresponding BMI value and “normal, please keep it” will be printed.

3） When BMI is less than 29.9 and greater than or equal to 24.9, the corresponding BMI value and “overweight” will be printed.

4） When BMI is greater than 29.9, the corresponding BMI value and “obese” will be printed.