Lab 4 *Socket Programming*

CAN201 – Introduction to Networking

Dr. Fei Cheng & Dr. Wenjun Fan

Content

- Function and lambda expression
- Recommended python programming style
- Socket programming
  - UDP
  - TCP

How to define a function?

```python
# Define a python function first
# ==============================
def the_1st_fun():
    print('The 1st python function')


# Let call the function:
the_1st_fun()

# When we run a python file, the python interpreter will
# execute the code line by line.
# But the interpreter will not run the function directly
# until the function is called.
```

How to pass arguments?

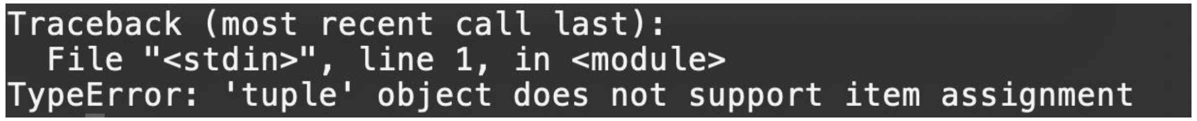
```
# Pass an argument to the function
# ================================
def the_2nd_fun(a):
    print('The number is %d' % a)


the_2nd_fun(100)

# How about passing a string?
the_2nd_fun('100')


# How to solve this issue?
def the_2nd_2_fun(a):
    if isinstance(a, int):
        print('The number is %d' % a)
    else:
        print('Could you please pass a integer?')


the_2nd_2_fun('100')
the_2nd_2_fun(100)
```

How to pass arguments?

```python
# The passed value will be changed by the function or not?
def the_2rd_3_fun(x):
    print('The passed argument is', x)
    x = 'Aha, I change.'
    print(x)


x = 10.0
the_2rd_3_fun(x)
print('x is ', x)


# How about passing a list to a function?
def the_2nd_4_fun(x):
    print('The passed argument is', x)
    x[0] = 10
    print('Now, x is ',x)


x = [1, 2, 3]
the_2nd_4_fun(x)
print('Finally, x is', x)
```

How to pass arguments?

```
# How about passing a tuple? PS. tuple is a immutable list
x = (1, 2, 3)
the_2nd_4_fun(x)
print('Finally, x is', x)
```



```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

```
# Why a number cannot be changed by a function,
# while a list can be changed?

# A: In python, there are two types of variables:
# Mutable: list, dict
# Immutable: tuples, strings, numbers(int, float...)

# When you pass a list, each item you modified will affect
# the original list (the list out of the function)
# For a dict, it is similar to a list. Have a try...

# Analogy to C/C++ language:
# mutable variable is "Pass by pointer or reference";
# immutable variable is "Pass by value".
```

How to pass arguments?

```
# How to pass many arguments?
def the_2nd_5_fun(x, y, z):
    print('x, y, z are', x, y, z)


a, b, c = 10, 20, 30  # Yes! This is a good way to set values
# Call the function in this way:
the_2nd_5_fun(a, b, c)
# Or
the_2nd_5_fun(y=b, x=a, z=c)


# How to set a default value for a function
def the_2nd_6_fun(x, y, z=1000):
    print('x, y, z are', x, y, z)


# Call the function in this way:
the_2nd_6_fun(10, 20)
```

How to pass arguments?

```
# How to use a global variable?
g = 10


def the_2nd_7_fun():
    print('The global variable is', g)


the_2nd_7_fun()
# Seems that you can use directly, but:


def the_2nd_8_fun():
    if g == 10:
        g = 20
    print(g)


the_2nd_8_fun()

# Oh no! Something wrong
```

How to pass arguments?

```
# This is the right way:
def the_2nd_9_fun():
    global g
    if g == 10:
        g = 20
    print('The global variable is', g)


the_2nd_9_fun()
```

How to return a value or values?

```
# Return values from function
# ============================
def the_3rd_fun(x):
    return x ** 2   # ** means power  （乘方）


a = 10
r = the_3rd_fun(a)
print('The result is', r)


# How to return many values
# Method 1: using list
def the_3rd_2_fun(x):
    l = [x, x + 1, x * 2]
    return l


l = the_3rd_2_fun(10)
print(l)
```

How to return a value or values?

```
# How about:
x1, x2, x3 = the_3rd_2_fun(10)
print(x1, x2, x3)


# Actually, we can use a very natural way to return values
def the_3rd_3_fun(x):
    return x, x + 1, x * 2


x1, x2, x3 = the_3rd_2_fun(10)
print(x1, x2, x3)

# If you do not need the second return value
x1, _, x3 = the_3rd_2_fun(10)
print(x1, x3)
```

function

DEMO

https://box.xjtlu.edu.cn/f/9b633aab9d9c4ed09bb9/

How to pass uncertain number of arguments?

```
# Uncertain number of arguments
# ==========================
# Method 1
def the_4th_fun(a, *args):
    print('a is', a)
    print('Others are', end=' ')
    for arg in args:
        print(arg, end=', ')  # Feel the meaning of "end"
    print('')


the_4th_fun(10, 20)
the_4th_fun(10, 20, 30)


# Method 2
def the_4th_2_fun(**args):
    for arg in args.keys():
        print(arg,'=',args[arg], end=', ')  # Feel the meaning of "end"
    print('')


the_4th_2_fun(a=1, b=2)
```

Lambda expression

- Small anonymous functions can be created with the lambda keyword
- `lambda a, b: a+b`

uncertain number of arguments & lambda

DEMO

https://box.xjtlu.edu.cn/f/5c8abc05313842ec9adf/

Lab practice – Where are numbers?

- Find all numbers from a string (int and float)
- Output a sorted number list (ascent) and their location in the string (start, length)
- input() should be used to get a string (we did not introduce “input”, but it is very easy to use, eg. x = input(‘Pls input a string’))
- Example:
  - input: '123klja32.23alk543$^&d'
  - output:
    number_list = [32.23, 123, 543]
    location_list = [[7,5], [0, 3], [15, 3]]
- Just practice, no check.

Recommended python programming style

```
import argparse


def _argparse():
    parser = argparse.ArgumentParser(description="This is description!")
    parser.add_argument('--input', action='store', required=True,
                        dest='path', help='The path of input file')
    parser.add_argument('--server', action='store', required=True,
                        dest='server', help='The hostname of server')
    parser.add_argument('--port', action='store', required=True,
                        dest='port', help='The port of server')
    return parser.parse_args()


def main():
    parser = _argparse()
    print(parser)
    print('Input file:', parser.path)
    print('Server:', parser.server)
    print('Port:', parser.port)


if __name__ == '__main__':
    main()
```

Demo: https://box.xjtlu.edu.cn/f/ab164c2770f84af789ff/

Socket programming

- IP address
  - 127.0.0.1 -> localhost
  - 192.168.xxx.xxx -> local network IP
  - Internet IP address
- Port number
  - 0~65535

Socket programming

- UDP:

```python
# server side
from socket import *

server_port = 12000
server_socket = socket(AF_INET, SOCK_DGRAM)
server_socket.bind(('', server_port))

print('The server is ready to receive.')

while True:
    message, client_address = server_socket.recvfrom(20480)
    print(message, client_address)
    modified_message = message.decode().upper()
    server_socket.sendto(modified_message.encode(), client_address)
```

```python
# client side
from socket import *

server_hostname = '127.0.0.1'
# '127.0.0.1' is only the localhost IP address for testing.
# If you want to send msg crossing different hosts,
# pls find the 'real' IP address using ipconfig (win) or ifconfig (linux/macos)

server_port = 12000

client_socket = socket(AF_INET, SOCK_DGRAM)

message = input('Input a sentence:')

client_socket.sendto(message.encode(), (server_hostname, server_port))
modified_message, server_address = client_socket.recvfrom(20480)
print(modified_message.decode(), server_address)
client_socket.close()
```

UDP Socket demo

https://box.xjtlu.edu.cn/f/081b694519024a8e94a7/

Socket programming

- TCP:

```
# server side
from socket import *

server_port = 12000
server_socket = socket(AF_INET, SOCK_STREAM)
server_socket.bind(('', server_port))
server_socket.listen(10)

print('The TCP server is listening')

while True:
    connection_socket, addr = server_socket.accept()
    sentence = connection_socket.recv(20480).decode()
    capitalized_sentence = sentence.upper()
    connection_socket.send(capitalized_sentence.encode())
```

```python
# client side
from socket import *

server_hostname = '127.0.0.1'
server_port = 12000
client_socket = socket(AF_INET, SOCK_STREAM)
client_socket.connect((server_hostname, server_port))

sentence = input('Input a sentence:')
client_socket.send(sentence.encode())
modified_message = client_socket.recv(20480)
print(modified_message)
client_socket.close()
```

TCP Socket demo

https://box.xjtlu.edu.cn/f/25ab85bc17e344deb449/

Try to send any data using socket!

”

Thanks