

# 復旦大學

## Python 入门与套接字编程

专业班级： 信息安全智慧技术班

学号： 21307130076

姓名： 杨乙

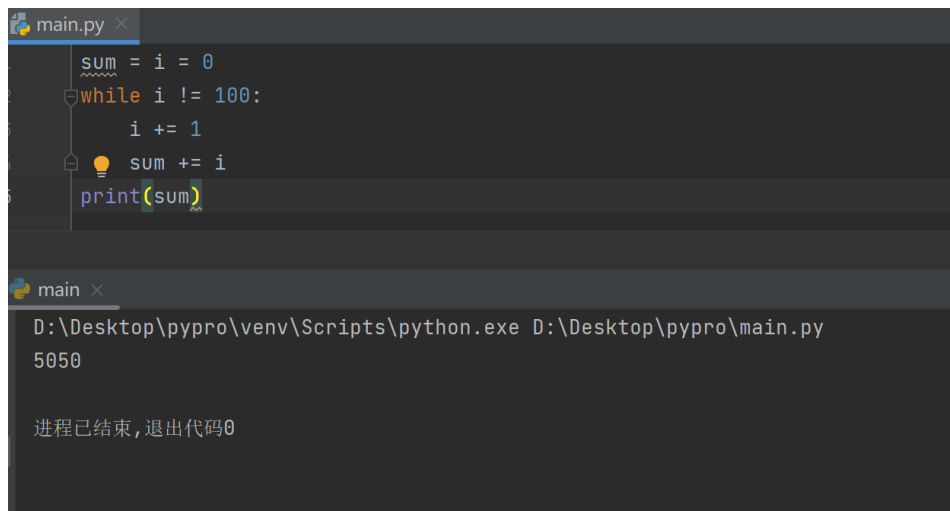
### 【实验目的】

学习 python 并尝试套接字编程

### 【实验结果】

#### 2. python 练习

1) 使用 while 来计算 1 到 100 的总和：



```
main.py ×
sum = i = 0
while i != 100:
    i += 1
    sum += i
print(sum)

main ×
D:\Desktop\pypro\venv\Scripts\python.exe D:\Desktop\pypro\main.py
5050

进程已结束,退出代码0
```

2) 判断用户输入的年份是否为闰年:

```
main.py ×
while True:
    y = int(input("Enter a particular year (Enter -1 to exit): "))
    if y == -1:
        break
    if y % 4 == 0 and y % 100 != 0 or y % 400 == 0:
        print(y, "is a leap year")
    else:
        print(y, "is not a leap year")

while True > if y % 4 == 0 and y % 100 != 0 ...

main ×
D:\Desktop\pypro\venv\Scripts\python.exe D:\Desktop\pypro\main.py
Enter a particular year (Enter -1 to exit): 2020
2020 is a leap year
Enter a particular year (Enter -1 to exit): 1999
1999 is not a leap year
Enter a particular year (Enter -1 to exit): 1000
1000 is not a leap year
Enter a particular year (Enter -1 to exit): -1

进程已结束,退出代码0
```

3) 输入两个数, 输出其最大公约数:

```
main.py ×
1 a = int(input("Please enter the first number: "))
2 b = int(input("Please enter the second number: "))
3 if a < b:
4     a, b = b, a
5 r = a % b
6 while r:
7     a, b = b, r
8     r = a % b
9 print("The GCD is", b)
10

main ×
D:\Desktop\pypro\venv\Scripts\python.exe D:\Desktop\pypro\main.py
Please enter the first number: 16
Please enter the second number: 24
The GCD is 8

进程已结束,退出代码0
```

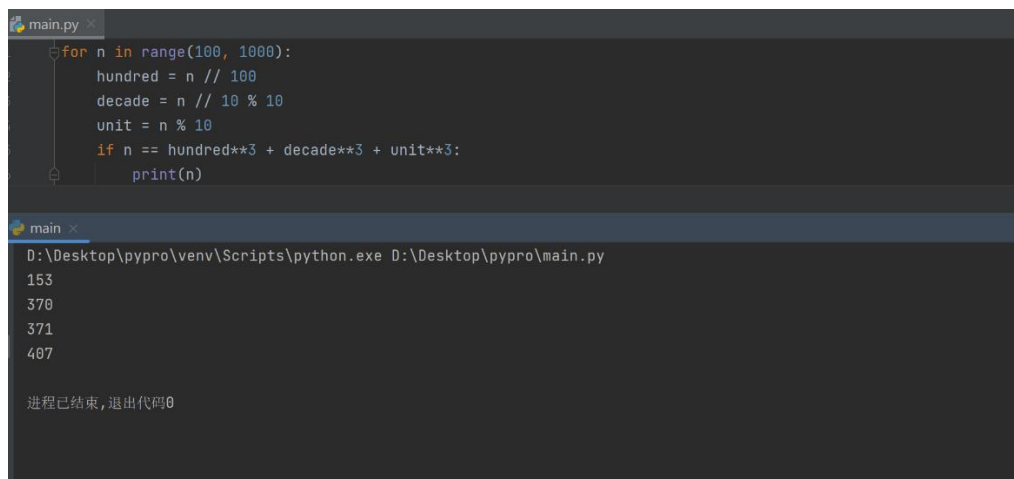
4) 有 1、2、3、4 个数字，能组成多少个互不相同且无重复数字的三位数？都是多少？

```
main.py x
1  L = []
2  num = 0
3  for i in range(1, 5):
4      for j in range(1, 5):
5          for k in range(1, 5):
6              if i != j and i != k and j != k:
7                  number = str(i) + str(j) + str(k)
8                  L.append(number)
9                  num += 1
10 print("The combination method is", num)
11 for number in L:
12     print(number)

main x
D:\Desktop\pypro\venv\Scripts\python.exe D:\Desktop\pypro\main.py
The combination method is 24
123
124
132
134
142
143
213
214
231
234
241
243
312
314
321
324
341
342
412
413
421
423
431
432

进程已结束,退出代码0
```

5) 打印出所有的“水仙花数”:



```
main.py ×
for n in range(100, 1000):
    hundred = n // 100
    decade = n // 10 % 10
    unit = n % 10
    if n == hundred**3 + decade**3 + unit**3:
        print(n)

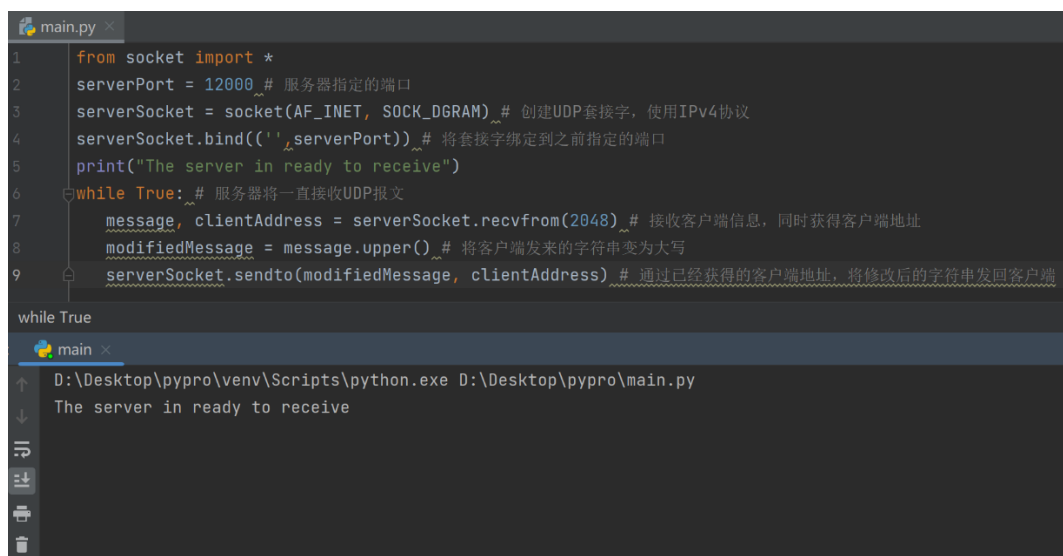
main ×
D:\Desktop\pypro\venv\Scripts\python.exe D:\Desktop\pypro\main.py
153
370
371
407

进程已结束,退出代码0
```

3. 分别运行 TCP、UDP 的服务端和客户端代码，进行测试实践，理解所用的函数，将结果截图至文档

1) UDP

服务端运行代码:



```
main.py ×
1 from socket import *
2 serverPort = 12000 # 服务器指定的端口
3 serverSocket = socket(AF_INET, SOCK_DGRAM) # 创建UDP套接字, 使用IPv4协议
4 serverSocket.bind(('', serverPort)) # 将套接字绑定到之前指定的端口
5 print("The server in ready to receive")
6 while True: # 服务器将一直接收UDP报文
7     message, clientAddress = serverSocket.recvfrom(2048) # 接收客户端信息, 同时获得客户端地址
8     modifiedMessage = message.upper() # 将客户端发来的字符串变为大写
9     serverSocket.sendto(modifiedMessage, clientAddress) # 通过已经获得的客户端地址, 将修改后的字符串发回客户端

while True
main ×
D:\Desktop\pypro\venv\Scripts\python.exe D:\Desktop\pypro\main.py
The server in ready to receive
```

客户端运行代码：

```
main.py x
1 from socket import *
2 serverName = '127.0.0.1' # 服务器地址，本例中使用一台远程主机
3 serverPort = 12000 # 服务器指定的端口
4 clientSocket = socket(AF_INET, SOCK_DGRAM) # 创建UDP套接字，使用IPv4协议
5 message = input('Input lowercase sentence:') .encode() # 用户输入信息，并编码为bytes以便发送
6 clientSocket.sendto(message, (serverName, serverPort)) # 将信息发送到服务器
7 modifiedMessage, serverAddress = clientSocket.recvfrom(2048) # 从服务器接收信息，同时也能得到服务器地址
8 print(modifiedMessage.decode()) # 显示服务器返回的信息
9 clientSocket.close() # 关闭套接字

main x
D:\Desktop\pypro1\venv\Scripts\python.exe D:\Desktop\pypro1\main.py
Input lowercase sentence:udp test
UDP TEST
进程已结束, 退出代码0
```

## 2) TCP

服务端运行代码：

```
main.py x
1 from socket import *
2 serverPort = 12000
3 serverSocket = socket(AF_INET, SOCK_STREAM) # 创建TCP欢迎套接字，使用IPv4协议
4 serverSocket.bind(('', serverPort)) # 将TCP欢迎套接字绑定到指定端口
5 serverSocket.listen(1) # 最大连接数为1
6 print("The server in ready to receive")
7
8 while True:
9     connectionSocket, addr = serverSocket.accept() # 接收到客户连接请求后，建立新的TCP连接套接字
10    print('Accept new connection from %s:%s...' % addr)
11    sentence = connectionSocket.recv(1024) # 获取客户发送的字符串
12    capitalizedSentence = sentence.upper() # 将字符串改为大写
13    connectionSocket.send(capitalizedSentence) # 向用户发送修改后的字符串
14    connectionSocket.close() # 关闭TCP连接套接字

while True
main x
D:\Desktop\pypro\venv\Scripts\python.exe D:\Desktop\pypro\main.py
The server in ready to receive
```

客户端运行代码，输入字符串：

```
main.py x
1  from socket import *
2  serverName = '127.0.0.1' # 指定服务器IP地址
3  serverPort = 12000
4  clientSocket = socket(AF_INET, SOCK_STREAM) # 建立TCP套接字，使用IPv4协议
5  clientSocket.connect((serverName, serverPort)) # 向服务器发起连接
6
7  sentence = input('Input lowercase sentence:').encode() # 用户输入信息，并编码为bytes以便发送
8  clientSocket.send(sentence) # 将信息发送到服务器
9  modifiedSentence = clientSocket.recvfrom(1024) # 从服务器接收信息
10 print(modifiedSentence[0].decode()) # 显示信息
11 clientSocket.close() # 关闭套接字

main x
D:\Desktop\pypro1\venv\Scripts\python.exe D:\Desktop\pypro1\main.py
Input lowercase sentence: this is a test!
THIS IS A TEST!

进程已结束,退出代码0
|
```

服务器端显示：

```
main.py x
1  from socket import *
2  serverPort = 12000
3  serverSocket = socket(AF_INET, SOCK_STREAM) # 创建TCP欢迎套接字，使用IPv4协议
4  serverSocket.bind(('', serverPort)) # 将TCP欢迎套接字绑定到指定端口
5  serverSocket.listen(1) # 最大连接数为1
6  print("The server in ready to receive")
7
8  while True:
9      connectionSocket, addr = serverSocket.accept() # 接收到客户连接请求后，建立新的TCP连接套接字
10     print('Accept new connection from %s:%s...' % addr)
11     sentence = connectionSocket.recv(1024) # 获取客户发送的字符串
12     capitalizedSentence = sentence.upper() # 将字符串改为大写
13     connectionSocket.send(capitalizedSentence) # 向用户发送修改后的字符串
14     connectionSocket.close() # 关闭TCP连接套接字

while True
main x
D:\Desktop\pypro\venv\Scripts\python.exe D:\Desktop\pypro\main.py
The server in ready to receive
Accept new connection from 127.0.0.1:57304...
5
|
```

分析：

TCP 套接字编程和 UDP 套接字编程的代码中，二者的服务器端代码都定义了目的地端口号（12000），客户端代码都定义了目的地端口号（12000）和服务器地址（127.0.0.1）；二者的客户端和服务端都需要创建套接字用以通信，且服务器端需要将套接字绑定到之前指定的端口（客户端不需要绑定端口号：因为客户端的端口号是系统随机分配的，且客户端给服务器端发送数据时，会将自己的 IP 地址和端口号一起发送过去，服务器端可以找到客户端）

但它们的代码之间也有如下不同点：

- 1) TCP 套接字编程代码中，客户端需要通过 `connect` 函数发送连接请求，服务器端需要通过 `accept` 函数接收客户请求并建立 TCP 连接（由上面的打印结果，服务器在此过程中会获取客户端的 IP 地址和端口号），且需要事先确定最大连接数。而客户端没有以上代码
- 2) 因为创建不同协议的套接字，所以创建套接字时函数参数不同

由此可见，TCP 服务是面向连接的