实验环境

严禁抄袭 仅供参考 Blog:zhangshier.vip

## 一、实验题目

实验八: 网络程序设计

## 二、实验目的

- (1) 了解 TCP 和 UDP 的工作原理。
- (2) 了解 IP 地址和端口号的含义和作用。
- (3) 掌握 Socket 编程的基本方法和步骤。
- (4) 掌握常见网络程序编制方法。

## 三、实验内容

1、使用基于 TCP 或 UDP 套接字编写一个智能聊天机器人程序。

## 四、 实验步骤

- 1、题目一
  - (1) 问题分析(含解决思路、使用的数据结构、程序流程图等)

TCP 工作需要建立连接、数据连接、断开连接三个步骤。利用 socket 模块,设计服务端与客户端,通过设置 IP 和端口号将二者连接。在服务端建立词库,客户端给服务端发送消息,服务端与词库做匹配后返回给客户端并输出

socket 模块常用于 TCP 编程的方法有:

connect(address):连接远程计算机 send(bytes[,flags]):发送数据 recv(bufsize[,flags]):接收数据 bind(address):绑定地址,通常用于服务端 listen(backlog):开始监听,等待客户端连接 accept():响应客户端的请求

(2) 算法/代码描述(基本要求源代码)

server.py

import socket from os.path import commonprefix

# 建立聊天回复字典

# 服务端主机 IP 地址和端口号,空字符串表示本机任何可用 IP 地址

HOST = "192.168.43.214"

PORT = 50007

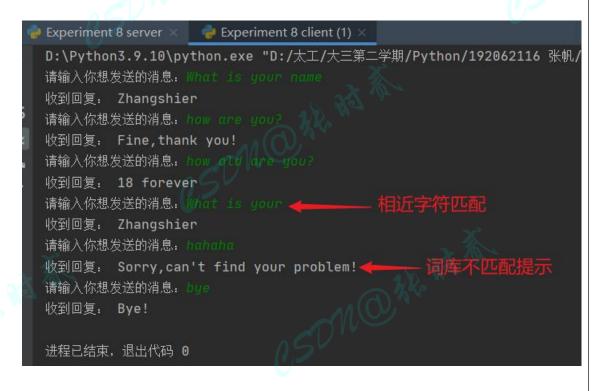
# 使用 IPV4 协议,使用 tcp 协议传输数据

1

```
s = socket.socket (socket.AF_INET, socket.SOCK_STREAM)
# 绑定端口和端口号
s.bind ((HOST, PORT))
# 开始监听, 规定最多支持 1 个客户端连接
s.listen (1)
print ('目前监听的端口号是:', PORT)
conn, addr = s.accept ()
print ('目前连接的 IP 地址是:', addr)
# 开始聊天
while True:
   # 最多可以接收 1024 比特大小的内容,并解码
   data = conn.recv (1024).decode ()
   # 如果是空,退出
   if not data:
       break
   print ('接收到的内容: ', data)
   # 尽可能猜测对方的意思
   m = 0
   key = "
   for k in words.keys ():
       # 删除多余的空白字符
       data = ' '.join ( data.split () )
       # 与某个键非常接近,就直接返回
       if len (commonprefix ([k, data])) > len (k)*0.7:
           key = k
           break
       # 使用选择法,选择一个重合度较高的键
       length = len ( set ( data.split () ) & set ( k.split () ) )
       if length > m:
           m = length
           key = k
   # 选择合适的信息进行回复
   conn.sendall ( words.get ( key, 'Sorry,can\'t find your problem! ' ).encode () )
conn.close ()
client.py
import socket
import sys
# 服务端主机 IP 地址和端口号 192.168.43.214
HOST = "192.168.43.214"
PORT = 50007
# 使用 IPV4 协议,使用 tcp 协议传输数据
```

```
s = socket.socket (socket.AF_INET, socket.SOCK_STREAM)
try:
   # 连接服务器
   s.connect ((HOST, PORT))
except Exception as e:
   print ('找不到服务器,请稍后重试!')
   sys.exit()
while True:
   c = input ('请输入你想发送的消息:')
   # 发送数据,使用 UTF-8 编码成字节码
   s.sendall (c.encode ())
   # 从服务端接收数据,大小最多为 1024 比特
   data = s.recv ( 1024 )
   #解码
   data = data.decode ()
   print ('收到回复:', data)
   if c.lower () == 'bye':
       break
# 关闭连接
s.close ()
```

(3) 运行结果(含执行结果验证、输出显示信息)



五、 出现的问题及解决的方法

无