实验题目： 实验 3 使用 TCP 实现智能聊天机器人

实验环境： Python、PyCharm 等

一、 实验目的

1. 熟悉标准库 socket 的用法。
2. 熟悉 TCP 的工作原理。
3. 理解端口号的概念与作用。
4. 熟悉 socket 编程。
5. 熟练掌握字典的使用。
6. 熟悉集合的常用运算。
7. 了解 os.path 中 commonprefix()函数的用法。
8. 熟练掌握字符串的常用方法。

二、 实验内容

编写聊天程序的服务端代码和客户端代码。完成后,先启动服务端代码，然后启动客户端程序输入问题,服务端可以返回相应的答案。要求服务端代码具 有一定的智能，能够根据不完整的问题识别客户端真正要问的问题。

三、 实验步骤及结果

1.实验源代码

服务端应用程序 chatServer 的源代码编写如下：

```
# 导入 socket 模块
import socket

# 导入完成功能需要的其他模块
from posixpath import split
from os.path import commonprefix

# 完成简单对话需要的字典
conversation = {"What's your name?" : "Yuzu.",
```

```
                "How old are you?" : "18 years old.",
                "Which sports do you like?" : "Figure skating.",
                "Which athlete do you support most in your favorite field?" :
"Hanyu Yuzuru.",
                "What's your favorite food?" : "Potato.",
                "Where are you from?" : "Shanxi Province, China",
                "What's the weather?" : "Sunny day, very nice.",
                "Bye." : "Bye."}

# 创建服务器，使用 IPv4 协议，TCP 套接字
chatServer = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# 绑定端口和端口号
chatServer.bind(('127.0.0.1', 8000));
# 开始侦听，队列长度为 1
chatServer.listen(1)
# 使用阻塞方法 accept 以等待客户机连接请求
clientSocket, clientAddress = chatServer.accept()
# 接收客户机请求后输出客户机的信息
print('Connection from', clientAddress)

# 循环以接收和回送客户机数据
while(1) :
    Setence = clientSocket.recv(1024).decode()      # 接收数据
    if not Setence :
        break                                       # 接收到空数据时终止循环
    print("Data from Client: ", Setence)            # 将接收到的语句输出

    # 猜测内容
    guess = ''
    maxsame = 0
    for key in conversation.keys():
        Sentence = ' '.join(Setence.split())        # 删除多余空格

        # 找到第一个与输入非常相似的键
        if len(commonprefix([key, guess])) > len(key) * 0.8:
            guess = key
            break

        # 每一个都不能与输入的字符串以高匹配度匹配时，此时选择重合度较高的一个
        samelength = len(set(Sentence.split()) & set(key.split()))
        if samelength > maxsame:
            guess = key
            maxsame = samelength
    # 回复合适的信息
```

```
    Reply = conversation.get(guess, "Sorry, I can not answer your
question.!")
    clientSocket.sendall(Reply.encode())

clientSocket.close()
chatServer.close()
```

客户端应用程序 chatclient 源代码如下：

```
# 导入 socket 模块
# from re import L
import sys
import socket

# 创建客户机
chatclient = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# 连接服务器
try:
    chatclient.connect(('127.0.0.1', 8000))
except Exception as e:
    print("Error! Try it again!")
    sys.exit()

# 循环接收用户输入，并发送到服务器，接收服务器的回送
while(1):
    Sentence = input("Q: ");                    # 接收用户输入数据
    chatclient.sendall(Sentence.encode());      # 把数据转化为 bytes 对象，
并发送到服务器
    if not Sentence:
        break                                   # 如果发送数据为空，终止循环
    Reply = chatclient.recv(1024)               # 从服务器端接收数据
    Reply = Reply.decode()
    print("A: ", Reply)                         # 输出接收到的数据
    if Sentence.lower() == 'bye.':
        break;                                  # 结束对话

chatclient.close()                              # 关闭客户机
```

2.测试用例及测试结果

输入服务端源代码设置的字典中的语句对代码进行测试，同时测试能否根据不完整的问题识别出客户端真正想要问的问题。测试结果如下图。

```
========== RESTART: D:\Codefield\Python\Demo\Exp3_ChatServer.py ==========
Connection from ('127.0.0.1', 54867)
Data from Client:  What's your name?
Data from Client:  How r u?
Data from Client:  How old r u?
Data from Client:  Which spots do you like?
Data from Client:  Which athelete do u support most?
Data from Client:  What is ur favorite food?
Data from Client:  where r u from?
Data from Client:  What's the weather?
Data from Client:  Bye.
>>>
```

Ln: 51 Col:

```
========== RESTART: D:\Codefield\Python\Demo\Exp3_ChatClient.py ==========
Q: What's your name?
A:  Yuzu.
Q: How r u?
A:  I'm fine, thank you.
Q: How old r u?
A:  18 years old.
Q: Which spots do you like?
A:  Figure skating.
Q: Which athelete do u support most?
A:  Hanyu Yuzuru.
Q: What is ur favorite food?
A:  Potato.
Q: where r u from?
A:  Shanxi Province, China
Q: What's the weather?
A:  Sunny day, very nice.
Q: Bye.
A:  Bye.
Q:
>>>
```

四、 实验分析

1.对服务端和客户端的具体代码编写不太熟悉，对 ppt 和老师课堂上讲过的内容进行反复的研读，独立自主地根据 ppt 练习代码编写。

2.服务端识别出与字典中的问题不相同但相似的语句，这部分功能的实现需要使用 commonprefix 和集合 set 的功能。集合的部分知识在实验二中初步接触过，在这一次实验中对 commonprefix 和 set 分别进行了初步和深入的学习。

3.在实验中接收和发送的信息均为字符串，对字符串的一些操作和字符串转化为列表的某些操作进行了学习。

4.最终的程序可以实现对与服务端字典中表述不同的问题进行简单的判断，但是这种判断是以单词的相同程度来判断的，如果出现单词拼写错误，可能会识别成单词相近的问题，如下图所示。因此该“智能聊天机器人”的智能程度仍需要更复杂的代码强化。

```
========== RESTART: D:\Codefield\Python\Demo\Exp3_ChatServer.py ==========
Connection from ('127.0.0.1', 55310)
Data from Client:  How r u?
Data from Client:  How od are you?
```

Ln: 56 Col:

```
>>>
========== RESTART: D:\Codefield\Python\Demo\Exp3_ChatClient.py ==========
Q: How r u?
A:  I'm fine, thank you.
Q: How od are you?
A:  I'm fine, thank you.
Q:
```