**云南大学软件学院**

**实 验 报 告**

课程： 计算机网络实践 任课教师： 李海

姓名： 陈俊宏 学号： 20211060245 专业： 人工智能 成绩：



**实验三、套接字编程**

**1．实验目的：**

两人一组，编写一个客户端、服务器程序，掌握Socket编程原理。能够编写一些简单的网络程序，能够获取诸如IP地址和URL地址信息；在此基础上掌握Socket编程原理，并能在本机上分别编写客户端和服务器程序，还能运行出实验结果。

**2．实验环境：**

连入局域网络的主机一台。

**3．实验指导：**

Socket API是实现进程间通信的一种编程设施，也是一种为进程间提供底层抽象的机制。理解socket编程模型及其原理。

**4．实验分析，回答下列问题。**

（1）编写获取本机及服务器IP地址的程序，依次输入三个URL地址，并能获得对应的IP地址。

A. 给出程序的代码。

import socket  
  
  
while True:  
 url = input("请输入你要查询的URL地址：") # 设置循环从而不断输入URL地址  
 if url == "host": # 判断当输入host时，返回本机的ip地址  
 hostname = socket.gethostname()  
 ip = socket.gethostbyname(hostname)  
 print(f"本机IP地址为{ip}")  
 elif url == "exit": # 程序出口  
 break  
 else: # 当输入不为"exit"也不为"host"时，返回对应URL的IP地址  
 ip = socket.gethostbyname(url)  
 print(f"{url}对应的IP地址为{ip}")

1. 给出三个URL地址：www.ynu.edu.cn,www.ibm.com,www.sina.com对应的IP地址截图。

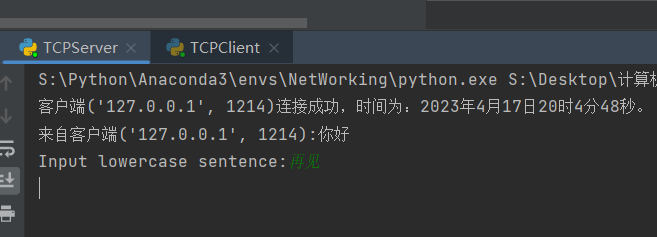


1. 分别编写可以进行简单通讯的服务器和客户端的Socket程序，服务器端的端口号为2023，如果遇到冲突，可以改为3033。(**分布用UDP和TCP实现**)

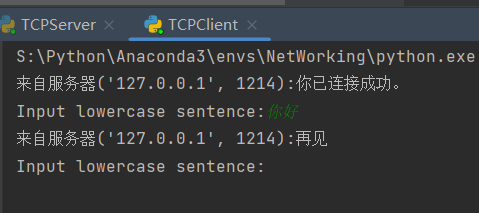
A. 给出服务器端程序的代码。

# TCPServer  
import time  
from socket import \*  
  
serverPort = 2023 # 设置端口  
serverSocket = socket(AF\_INET, SOCK\_STREAM) # 创建TCP Socket  
serverSocket.bind(('localhost', serverPort)) # 将套接字绑定到地址  
serverSocket.listen(1) # 设置监听器  
time\_tuple = time.localtime(time.time()) # 获取连接时间  
connectionSocket, addr = serverSocket.accept() # 获取连接体和链接地址  
print(  
 f"客户端{addr}连接成功，时间为：{time\_tuple[0]}年{time\_tuple[1]}月{time\_tuple[2]}日{time\_tuple[3]}时{time\_tuple[4]}分{time\_tuple[5]}秒。")  
while True:  
 sentence = connectionSocket.recv(1024) # 接收客户端TCP数据  
 print(f"来自客户端{addr}:{sentence.decode('UTF-8')}")  
 msg = input('Input lowercase sentence:') # 回复客户端TCP数据  
 if sentence == exit:  
 break  
 connectionSocket.send(msg.encode("UTF-8"))  
connectionSocket.close()

1. 给出客户端程序的代码。
2. # TCPClient  
   from socket import \*  
     
   serverName = 'localhost'  
   serverPort = 2023  
   clientSocket = socket(AF\_INET, SOCK\_STREAM) # 设置TCP Socket客户端  
   clientSocket.connect((serverName, serverPort)) # 连接服务器  
   address = clientSocket.getsockname() # 获取并打印服务器地址  
   print(f"来自服务器{address}:你已连接成功。")  
   while True:  
    sentence = input('Input lowercase sentence:')  
    if sentence == exit:  
    break  
    clientSocket.send(sentence.encode("UTF-8")) # 向服务器发送TCP数据  
    modifiedSentence = clientSocket.recv(1024) # 接收服务器返回的TCP数据  
    print(f"来自服务器{address}:{modifiedSentence.decode('UTF-8')}")  
   clientSocket.close()
3. 当客户端连接到服务器端后，服务器端可以显示“客户端(主机名)连接成功，时间为：年月日时分秒。”显示服务器端的运行截图。



1. 客户端可以显示服务器端的连接信息，即“来自服务器(IP地址)：你已连接成功。”以及“来自服务器：再见！”给出对应的运行截图。



1. 回答下列问题：

a. 为了完成通信，服务器和客户端需要引入哪些模块文件，它们的各自的作用是什么？

答：

服务器：

Import socket用于套接字连接的相关工作

Import time用于获得当前时间

客户端：

Import socket用于套接字连接的相关工作

b. 如果先运行客户端程序，程序会有什么现象，为什么会有这一现象？

答：会产生ConnectionRefusedError,被拒绝连接导致连接失败，因为服务器还没有运行，服务器端口未开放。

c. 如果服务器程序所在计算机没有连接网络，程序会发生什么错误？我们能捕获到什么错误信息？

答：由于我采取的是本地连接的方式，所以在计算机没有连接网络时，程序并没有发生错误；但是如果要进行局域网通信的话，如果计算机没有连接网络，可能会出现拒绝访问或者连接失败等问题。

（3）开发一个一次处理一个 HTTP 请求的 Web 服务器。该Web 服务器应接受并解析 HTTP 请求，从服务器的文件系统获取请求的文件，创建 HTTP 响应消息由请求的文件组成，前面有标题行，然后将响应直接发送到客户端。如果服务器中不存在请求的文件，则服务器应发送 HTTP“404 Not 未找到”消息返回到客户端。

在相关附件中找到 Web 服务器的框架代码。请完成程序代码。这 需要填写代码的位置在#Fill in start #Fill in end。每个地方可能需要一行或多行代码。 运行服务器请提前将HTML文件（例如HelloWorld.html）放在服务器所在的同一目录中。运行服务器程序。确定运行服务器的主机的 IP 地址（例如，128.238.251.26）。从另一个主机，打开浏览器并提供相应的 URL。例如：http://128.238.251.26:6789/HelloWorld.html ，

“HelloWorld.html”是放置在服务器目录中的文件的名称。另请注意端口的使用，冒号后的数字，需要将此端口号替换为在服务器代码。在上面的示例中，我们使用了端口号 6789。然后浏览器应显示网页的内容。如果省略“：6789”，浏览器将采用端口80，将获得仅当服务器侦听端口 80 时，来自服务器的网页。 然后尝试获取服务器上不存在的文件。应该会收到“404 未找到”消息。

要求： 提交完整的服务器代码以及程序的屏幕截图，验证实际上从服务器接收 HTML 文件的内容。

from socket import \*  
import sys  
  
serverSocket = socket(AF\_INET, SOCK\_STREAM)  
  
# 绑定 IP 地址和端口号  
serverSocket.bind(('192.168.64.1', 6789))  
  
# 开始监听连接  
print('Server is running on http://192.168.64.1:6789')  
  
while True:  
 # 等待客户端连接  
 print('Ready to serve...')  
 serverSocket.listen(1)  
 connectionSocket, addr = serverSocket.accept()  
  
 try:  
 # 接收客户端请求  
 message = connectionSocket.recv(1024).decode()  
  
 # 获取请求的文件路径  
 filename = message.split()[1][1:]  
  
 # 读取文件内容  
 with open(filename, 'r', encoding='utf-8') as f:  
 outputdata = f.readlines()  
  
 # 构造 HTTP 响应  
 response = 'HTTP/1.1 200 OK\r\n\r\n'  
 response += ''.join(outputdata)  
  
 # 发送响应给客户端  
 connectionSocket.sendall(response.encode())  
  
 # 关闭连接  
 connectionSocket.close()  
  
 except IOError:  
 # 构造 HTTP 响应  
 response = 'HTTP/1.1 404 Not Found\r\n\r\n'  
 response += '<h1>404 Not Found</h1>'  
  
 # 发送响应给客户端  
 connectionSocket.sendall(response.encode())  
  
 # 关闭连接  
 connectionSocket.close()  
  
serverSocket.close()  
sys.exit()

