Computer Network HW1

B05902050

黃子源

1. Program Structure:

   Part A:

   For part A, just use socket to connect to IRCDJS and send some message to the server.

   Principal code fragment:

```
horoscope = ['Capricorn', 'Aquarius', 'Pisces', 'Aries', 'Taurus',
IRCSocket = socket.socket( socket.AF_INET, socket.SOCK_STREAM )
IRCSocket.connect( ( '127.0.0.1', 6667 ) )
Msg1 = 'NICK bot_b05902050 \r\n'
Msg2 = 'USER b05902050 \r\n'
Msg3 = 'JOIN #CN_DEMO \r\n'
Msg4 = "PRIVMSG #CN_DEMO :I'm b05902050 \r\n"

IRCSocket.send(bytes(Msg1, encoding = 'utf-8'))
IRCSocket.send(bytes(Msg2, encoding = 'utf-8'))
IRCSocket.send(bytes(Msg3, encoding = 'utf-8'))
IRCSocket.send(bytes(Msg4, encoding = 'utf-8'))
```

   Part B:

   I create a list for the twelve zodiacs, and receive the messages sent from the socket. If the message is in zodiac list, send a message of the fortune to the socket.

   Principal code fragment:

```
while True:
    IRCMsg = IRCSocket.recv( 4096 ).decode()
    if IRCMsg.split(' ')[0] == 'PING':
        continue
    Splitstring = IRCMsg.split(':')
    name = Splitstring[1]
    name = name.split('!')[0]
    #print(name)
    IRCMsg = Splitstring[-1:][0][:-2]
    print(IRCMsg)
    if IRCMsg in horoscope:
        IRCSocket.send(bytes("PRIVMSG " + name + " :今日運勢 想不到吧 \r\n", encoding = 'utf-8'))
```

   Part C:

   If the message sent from the socket is ‘!guess’, create a random 1 to 10 integer, and make the user guess. Here I use a while loop to prevent other unrelated message interrupt the guess process until the user get the right number.

Principal code fragment:

```
if IRCMsg == '!guess':
    IRCSocket.send(bytes("PRIVMSG " + name + " :猜一個1~10之間的數字！ \r\n", encoding = 'utf-8'))
    ans = randint(1,10)
    record = [0 for i in range(10)]
    while True:
        IRCMsg = IRCSocket.recv( 4096 ).decode()
        Splitstring = IRCMsg.split(':')
        IRCMsg = Splitstring[-1:][0][:-2]
        if IRCMsg.isnumeric():
            guess = int(IRCMsg)
            if 1 <= guess <= 10:
                if guess == ans:
                    IRCSocket.send(bytes("PRIVMSG " + name + " :正確答案為" + str(ans) + "! 恭喜猜中\r\n", encoding =
                    break
                elif guess < ans:
                    msg = "大於" + str(guess) + "!"
                else:
                    msg = "小於" + str(guess) + "!"
                if record[guess - 1] == 1:
                    IRCSocket.send(bytes("PRIVMSG " + name + " :你猜過" + str(guess) + "了=_= " + msg + " \r\n", enco
                else:
                    IRCSocket.send(bytes("PRIVMSG " + name + " :" + msg + " \r\n", encoding = 'utf-8'))
                    record[guess - 1] = 1
```

Part D:

This part is quite complicated, I use BeautifulSoup and Requests to complete the work. First I create the url of searching mode in Youtube, and get its html code. After that, I use function find() and select() to find the link that we want to find, and send it to the user.

Principal code fragment:

```
if IRCMsg[:6] == '!song ':
    searchMsg = IRCMsg[6:]
    url = "https://www.youtube.com/results?search_query=" + searchMsg
    request = requests.get(url)
    soup = BeautifulSoup(request.content, "html.parser")
    song = soup.find("div", {"class": "yt-lockup-video"})
    data = song.select("a[rel='spf-prefetch']")
    IRCSocket.send(bytes("PRIVMSG " + name + " :https://www.youtube.com" + data[0].get("href") +
```

Part E:

In this part, to know whether to read from the socket or to send message to the socket, I use function select in case that the program will be blocked. Select will wait until either standard input or socket is ready, and return the list of the ready set. After select, if the standard input is ready, then read the input message and send it to the socket; if the socket is ready, read the message sent from the socket and print them.

Principal code fragment:

```
if IRCMsg == '!chat':
    print("========" + name + "想跟你聯繫========")
    inputs = [sys.stdin, IRCSocket]
    stop = 0
    while True:
        if stop:
            print("========" + name + "已離開聊天室========")
            break
        readable, writeable, exceptional = select.select(inputs,[],[])
        for read in readable:
            if read == sys.stdin:
                sendMsg = input()
                IRCSocket.send(bytes("PRIVMSG " + name + " :" + sendMsg + '\r\n', encoding = 'utf-8'))
            elif read == IRCSocket:
                IRCMsg = IRCSocket.recv( 4096 ).decode()
                Splitstring = IRCMsg.split(':')
                IRCMsg = Splitstring[-1:][0][:-2]
                print(name + " :",IRCMsg)
                if IRCMsg == "!bye":
                    stop = 1
                    break
```

2. Challenge and Solution

   In this homework, I have encountered some difficulties, but I solved most of them.

   The first difficulty is the installation of ircdjs, for I don't want to implement the task by Ubuntu or Linux. So I look on the Internet and use home brew to install ircdjs on mac os.

   The second difficulty is that in subtask E, select function can't predict whether the next readable string is from standard input or socket, so we can't print the character '>' before standard input like the example in the hw1 specification. The solution may be using the thread instead of select, since select can do multiprocess at the same time, it can do input() and read socket at the same time, so theoretically, it can print '>' before the input.

3. Reflections:

   I think this homework is really interesting, coding a chatting robot made me fulfilled. Maybe next time we can make a server by ourselves instead of ircdjs.