Задание №1

```
#Server.py
import socket

msg = 'Hello, Client'

sock = socket.socket(socket.AF_INET)
sock.bind(('localhost',9090))
sock.listen(1)
while True:
    connection, addr = sock.accept()
    data = connection.recv(1024)
    print(data.decode())
    connection.send(msg.encode('utf-8'))
    connection.close()


#Clinet.py
import socket

sock = socket.socket(socket.AF_INET)
sock.connect(('localhost',9090))
msg = 'Hello,Server'

sock.send(msg.encode('utf-8'))
data = sock.recv(1024)
print(data.decode())

sock.close()
```

Выводы:

```
PS E:\works\web\lab_1> python .\client.py

Hello, Client

PS E:\works\web\lab_1> python .\Server.py

Hello,Server
```

Задание №2

```
#Server.py
import socket

sock = socket.socket(socket.AF_INET)
sock.bind(('localhost',9090))
sock.listen(1)
while True:
    connection, addr = sock.accept()
    pack = connection.recv(1024)
    pack = pack.decode()
    pack = pack.split()
    data = pack[1:]
    data = list(map(float, data))
    print(data)
    if pack[0] == "a":
        msg = data[0]**2 + data[1]**2
    elif pack[0] == "b":
        func = (-data[1]+(data[1]**2 - 4*data[0]*data[2])**0.5)/data[0]*2
        msg = [func, -func]
    elif pack[0] == "c":
        msg = ((data[0] + data[1])*data[2])/2
    elif pack[0] == "d":
        msg = data[0] * data[1]

    msg = str(msg)
    connection.send(msg.encode('utf-8'))
    connection.close()

#Client.py
import socket

sock = socket.socket(socket.AF_INET)
sock.connect(('localhost',9090))
msg = input()

sock.send(msg.encode('utf-8'))
data = sock.recv(1024)
print(data.decode())

sock.close()
```

Выводы:

```
PS E:\works\web\lab_1> python .\client.py
a 1 3
10.0
```

```
PS E:\works\web\lab_1> python .\Server.py
[1.0, 3.0]
```

Задание №3

```
#Server.py
import socket

sock = socket.socket(socket.AF_INET)
sock.bind(('localhost',9090))
sock.listen(1)
while True:
    connection, addr = sock.accept()
    pack = connection.recv(1024)
    connection.sendall(b"HTTP/1.0 200 OK\nContent-Type: text/html\n\n"
     + open("index.html", "rb").read())
    connection.close()

#Client.py
import socket

sock = socket.socket(socket.AF_INET)
sock.connect(('localhost',9090))

sock.sendall(bytes(f'.', 'utf-8'))
data = sock.recv(1024)
print(data.decode('utf-8'))
sock.close()
```

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
</head>

<body>
    <p>Test page</p>
</body>

</html>
```

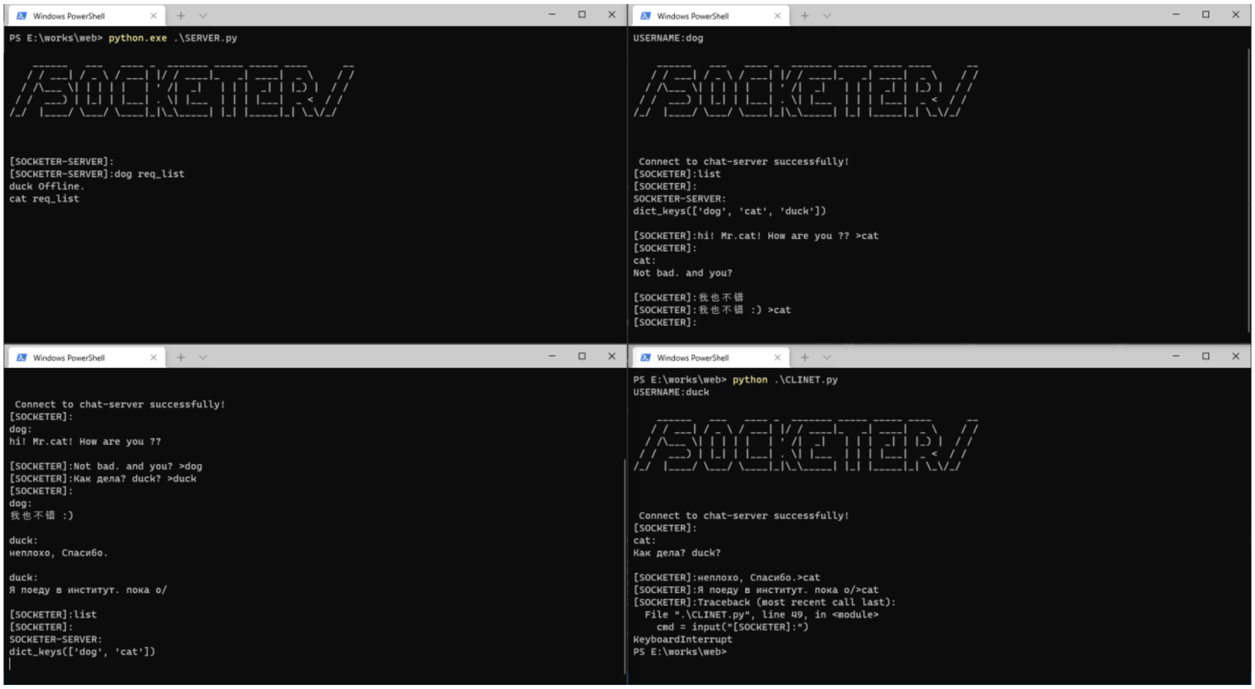
Выводы:

```
PS E:\works\web\lab_1> python .\client.py
HTTP/1.0 200 OK
Content-Type: text/html

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
</head>

<body>
    <p>Test page</p>
</body>

</html>
```

Задание №4:

```
#server.py
import json
import socket
from threading import Thread


ADDRESS = ('localhost', 12321)
CONNECTION = None
SERVERNAME = "SOCKETER-SERVER"
CONNECTION_POOL = {}
init_msg = """
    ______   ___   ____ _  _______ _____ _____ ____      __
   / / ___| / _ \ / ___| |/ / ____|_   _| ____|  _ \    / /
  / /\___ \| | | | |   | ' /|  _|   | | |  _| | |_) |  / /
 / /  ___) | |_| | |___| . \| |___  | | | |___|  _ <  / /
/_/  |____/ \___/ \____|_|\_\_____| |_| |_____|_| \_\/_/


"""


def init():
    global CONNECTION
    CONNECTION = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    CONNECTION.bind(ADDRESS)
    CONNECTION.listen(5)
    print(init_msg)

def send_msg(socket1, username, cmd, data, target):
    jdata = {}
    jdata['cmd'] = cmd
    jdata['target'] = target
    jdata['from'] = username
    jdata['data'] = data
    jstr = json.dumps(jdata)
    socket1.sendall(jstr.encode('utf-8'))


def connection_control():
    while True:
        client, client_address = CONNECTION.accept()
        newThread = Thread(target=msg_control,args=(client,))
        newThread.setDaemon(True)
        newThread.start()
```

```
def msg_control(client):
    client.sendall((init_msg + "\n Connect to chat-
server successfully!").encode('utf-8'))
    while True:
        try:
            data = client.recv(1024).decode('utf-8')
            jdata = json.loads(data)
            user = jdata['from']
            target = jdata['target']
            if jdata['cmd'] == "act_connect":
                CONNECTION_POOL[user] = client
            if jdata['cmd'] == "act_send_msg":
                if jdata['target'] not in CONNECTION_POOL:
                    send_msg(client,SERVERNAME,"ERROR",f"ERROR:User{jdata[
'target']} does not online!",user)
                else:
                    target_connection = CONNECTION_POOL[target]
                    send_msg(target_connection,user,"act_send_msg",jdata['
data'],target)
            elif jdata['cmd'] == "act_req_list":
                print(user + " req_list")
                userlist = str(CONNECTION_POOL.keys())
                send_msg(client,SERVERNAME,"return_list",userlist,user)
        except Exception:
            remove_client(user)
            break

def remove_client(user):
    connection = CONNECTION_POOL[user]
    if None != connection:
        connection.close()
        CONNECTION_POOL.pop(user)
        print(f"{user} Offline.")



if __name__ == "__main__":
    init()
    t1 = Thread(target=connection_control)
    t1.setDaemon(True)
    t1.start()
    while True:
        cmd = input("[SOCKETER-SERVER]:")
        if cmd == "list":
            print(CONNECTION_POOL.keys())
        if cmd == "exit":
            break
        if cmd == "list1":
            print(CONNECTION_POOL)


#client.py

import socket
import json
import re
from threading import Thread

ADDRESS = ('127.0.0.1',12321)
USERNAME = ""

def init_connection():
    connection = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    connection.connect(ADDRESS)
    print(connection.recv(1024).decode('utf-8'))
    send_msg(connection,'act_connect','','s')
    return connection

def get_msg(socket1):
    while True:
        data = socket1.recv(1024).decode('utf-8')
        jdata = json.loads(data)
        print(f"\n{jdata['from']}:\n{jdata['data']}")

def send_msg(client, cmd, data, target):
    global USERNAME
    jdata = {}
    jdata['cmd'] = cmd
    jdata['target'] = target
    jdata['from'] = USERNAME
    jdata['data'] = data
    jstr = json.dumps(jdata)
    client.sendall(jstr.encode('utf-8'))

def get_username():
    global USERNAME
    while True:
        USERNAME = input("USERNAME:")
        if re.search(r'\w',USERNAME) is None:
            print("User NAME CAN NOT BE NONE!")
        else:
            break


if __name__ == "__main__":
    get_username()
    connection = init_connection()
    t1 = Thread(target=get_msg, args=(connection,))
    t1.setDaemon(True)
    t1.start()
    while True:
        cmd = input("[SOCKETER]:")
        if cmd == "list":
            send_msg(connection,'act_req_list','','s')
        if re.search(r'>',cmd) is not None:
            text, target_user = cmd.split(">")
            send_msg(connection,'act_send_msg',text,target_user)
```



```
PS E:\works\web> python.exe .\SERVER.py

   ______  ___   ____ _ _______ _____ _____ ____    __
  / / ___|/ _ \ / ___| |/ / ____|_   _| ____|  _ \  / /
 / /\___ \| | | | |   | ' /|  _|   | | |  _| | |_) |/ /
/ /  ___) | |_| | |___| . \| |___  | | | |___|  _ </ /
/_/  |____/ \___/ \____|_|\_\_____| |_| |_____|_| \_\/_/


[SOCKETER-SERVER]:
[SOCKETER-SERVER]:dog req_list
duck Offline.
cat req_list
```

```
USERNAME:dog

   ______  ___   ____ _ _______ _____ _____ ____    __
  / / ___|/ _ \ / ___| |/ / ____|_   _| ____|  _ \  / /
 / /\___ \| | | | |   | ' /|  _|   | | |  _| | |_) |/ /
/ /  ___) | |_| | |___| . \| |___  | | | |___|  _ </ /
/_/  |____/ \___/ \____|_|\_\_____| |_| |_____|_| \_\/_/


 Connect to chat-server successfully!
[SOCKETER]:list
[SOCKETER]:
SOCKETER-SERVER:
dict_keys(['dog', 'cat', 'duck'])

[SOCKETER]:hi! Mr.cat! How are you ?? >cat
[SOCKETER]:
cat:
Not bad. and you?

[SOCKETER]:我也不错
[SOCKETER]:我也不错 :) >cat
[SOCKETER]:
```

```


 Connect to chat-server successfully!
[SOCKETER]:
dog:
hi! Mr.cat! How are you ??

[SOCKETER]:Not bad. and you? >dog
[SOCKETER]:Как дела? duck? >duck
[SOCKETER]:
dog:
我也不错 :)

duck:
неплохо, Спасибо.

duck:
Я поеду в институт. пока o/

[SOCKETER]:list
[SOCKETER]:
SOCKETER-SERVER:
dict_keys(['dog', 'cat'])
|
```

```
PS E:\works\web> python .\CLINET.py
USERNAME:duck

   ______  ___   ____ _ _______ _____ _____ ____    __
  / / ___|/ _ \ / ___| |/ / ____|_   _| ____|  _ \  / /
 / /\___ \| | | | |   | ' /|  _|   | | |  _| | |_) |/ /
/ /  ___) | |_| | |___| . \| |___  | | | |___|  _ </ /
/_/  |____/ \___/ \____|_|\_\_____| |_| |_____|_| \_\/_/


 Connect to chat-server successfully!
[SOCKETER]:
cat:
Как дела? duck?

[SOCKETER]:неплохо, Спасибо.>cat
[SOCKETER]:Я поеду в институт. пока o/>cat
[SOCKETER]:Traceback (most recent call last):
  File ".\CLINET.py", line 49, in <module>
    cmd = input("[SOCKETER]:")
KeyboardInterrupt
PS E:\works\web>
```

Структура json:

FROM	CMD	DATA	TARGET

Примеры:

Connect:

Client send:

USERNAME	act_connect		SERVER

Server send:

SERVER		Init_msg	USERNAME

Send message:

Client send:

User1	act_send_msg	Content	User2

Server recive:

User1	act_send_msg	Content	User2

Server send:

User1	act_send_msg	Content	User2

Get online user’s list:

Client send:

USERNAME	act_req_list		SERVER

Server recive:

USERNAME	act_req_list		SERVER

Server send to User:

SERVER	Return_list	userlist	USERNAME