Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики.

Факультет инфокоммуникационных технологий.

Практическая работа №1 по теме:
«Работа с сокетами» по дисциплине:
Web-программирование.

Выполнил: Кирильцев Роман
Группа: К33401

Преподаватель: Говоров Антон Игоревич

Санкт-Петербург, 2020

Цель работы: овладеть практическими навыками и умениями реализации web-серверов и использования сокетов.

Практическое задание 1:

Реализовать клиентскую и серверную часть приложения. Клиент отсылает серверу сообщение «Hello, server». Сообщение должно отразиться на стороне сервера. Сервер в ответ отсылает клиенту сообщение «Hello, client». Сообщение должно отобразиться у клиента.

Код программы:

client.py server.py

```python
import socket
from threading import Thread
import threading


class Server:

    def __init__(self):
        sock.bind(('localhost', 9090))
        sock.listen(10)
        print("Server was started on socket:", sock)
        print("Wait for users...")

    def get_socket(self):
        while True:
            client, addr = sock.accept()
            clients.append({"connection": client, "socket": addr})
            print("connected:", addr)
            Thread(target=self.check_messages, args=(client,)).start()

    def check_messages(self, client):
        while True:
            try:
                data = client.recv(4096)
                data = str(data.decode("utf-8"))
                print(data)
                self.target_cast(client, "Hello, client!")
            except Exception as ex:
```

```python
            except Exception as ex:
                self.delete_user_socket(client)
                break

    def delete_user_socket(self, client):
        for user in clients:
            if user["connection"] == client:
                clients.remove(user)
                break

    def target_cast(self, client, message):
        message += "\n"
        client.send(message.encode())


if __name__ == "__main__":
    sock = socket.socket()
    clients = []
    server = Server()
    thread = threading.Thread(target=server.get_socket())
    thread.start()
    thread.join()
```

client.py server.py

```python
import socket

sock = socket.socket()
sock.connect(('localhost', 9090))
sock.send('Hello, server'.encode("UTF-8"

data = sock.recv(1024)
sock.close()

print(data.decode("UTF-8"))
```

Результат работы программы:

erminal: Local Local (2) +

```
:\Users\rk319\Documents\Study\untitled\lab1>py client.py
ello, client!


:\Users\rk319\Documents\Study\untitled\lab1>
```

```
Terminal:  Local ×  Local (2) ×  +
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab1.py
Системе не удается найти указанный путь.

C:\Users\rk319\Documents\Study\untitled>cd lab1

C:\Users\rk319\Documents\Study\untitled\lab1>py server.py
Server was started on socket: <socket.socket fd=524, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090)>
Wait for users...
connected: ('127.0.0.1', 54874)
Hello, server
```

Практическое задание 2:

Реализовать клиентскую и серверную часть приложения. Клиент запрашивает у сервера выполнение математической операции, параметры, которые вводятся с клавиатуры. Сервер обрабатывает полученные данные и возвращает результат клиенту.

Вариант: а) Теорема Пифагора

Код программы:

```
client.py ×  server.py ×
import socket
from threading import Thread
import threading
import math


class Server:

    def __init__(self):
        sock.bind(('localhost', 9090))
        sock.listen(10)
        print("Server was started on socket:", sock)
        print("Wait for users...")

    def get_socket(self):
        while True:
            client, addr = sock.accept()
            clients.append({"connection": client, "socket": addr})
            print("connected:", addr)
            Thread(target=self.check_messages, args=(client,)).start()

    def check_messages(self, client):
        while True:
            try:
                data = client.recv(4096)
                data = str(data.decode("utf-8"))
                if not data == "Square Equations":
```

```python
                data = client.recv(4096)
                data = str(data.decode("utf-8"))
                if not data == "Square Equations":
                    self.target_cast(client, "Такой задачи нет(")
                    continue
                self.target_cast(client, "Введите катеты a, b через пробел")
                data = client.recv(4096)
                data = str(data.decode("utf-8"))
                a, b = map(float, [i.strip() for i in data.split()])
                self.target_cast(client, "Гипотенуза равна: %.2f" % math.sqrt(math.pow(a,2)+math.pow(b,2)))
            except Exception as ex:
                print(ex)
                self.delete_user_socket(client)
                break

    def delete_user_socket(self, client):
        print("delete user", client)
        for user in clients:
            if user["connection"] == client:
                clients.remove(user)
                break

    def target_cast(self, client, message):
        message += "\n"
        client.send(message.encode())

```

```python
                break

    def target_cast(self, client, message):
        message += "\n"
        client.send(message.encode())


if __name__ == "__main__":
    sock = socket.socket()
    clients = []
    server = Server()
    thread = threading.Thread(target=server.get_socket())
    thread.start()
    thread.join()
```

```python
import socket

task = "Square Equations"
encoding = "UTF-8"
sock = socket.socket()
sock.connect(('localhost', 9090))

sock.send(task.encode(encoding))
data = sock.recv(1024)
print(data.decode("UTF-8"))

numbers = input()

sock.send(numbers.encode(encoding))
data = sock.recv(1024)
print(data.decode("UTF-8"))
sock.close()
```

Результат работы программы:

```
Terminal:  Local  Local (2)  +

C:\Users\rk319\Documents\Study\untitled\lab2>py server.py
Server was started on socket: <socket.socket fd=476, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090)>
Wait for users...
connected: ('127.0.0.1', 54883)
[WinError 10053] Программа на вашем хост-компьютере разорвала установленное подключение
delete user <socket.socket fd=488, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090), raddr=('127.0.0.1', 54883)>
```

```
Terminal:  Local  Local (2)  +
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.
C:\Users\rk319\Documents\Study\untitled>cd lab2

C:\Users\rk319\Documents\Study\untitled\lab2>py client.py
Введите катеты a, b через пробел

3 4
Гипотенуза равна: 5.00


C:\Users\rk319\Documents\Study\untitled\lab2>
```

Практическое задание 3:

Реализовать серверную часть приложения. Клиент подключается к серверу. В ответ клиент получает http-сообщение, содержащее html-страницу, которую сервер подгружает из файла index.html.

Код программы:

client.py server.py index.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Page</title>
</head>
<body>

<H1>WebPage</H1>

</body>
</html>
```

client.py server.py index.html

```python
import socket
from threading import Thread
import threading


class Server:

    def __init__(self):
        sock.bind(('localhost', 9090))
        sock.listen(10)
        print("Server was started on socket:", sock)
        print("Wait for users...")

    def get_socket(self):
        while True:
            client, addr = sock.accept()
            clients.append({"connection": client, "socket": addr})
            print("connected:", addr)
            Thread(target=self.check_messages, args=(client,)).start()

    def check_messages(self, client):
        while True:
            try:
                data = client.recv(1024)
                self.target_cast(client, open("index.html").read())
            except Exception as ex:
                self.delete_user_socket(client)
                break
```

client.py server.py index.html

```python
                self.target_cast(client, open("index.html").read())
            except Exception as ex:
                self.delete_user_socket(client)
                break

    def delete_user_socket(self, client):
        for user in clients:
            if user["connection"] == client:
                clients.remove(user)
                break

    def target_cast(self, client, message):
        headers = bytes(f'HTTP/1.0 200 OK\nContent-Type: text/html\n\n{message}', 'utf-8')
        # message += "\n"
        client.send(headers)


if __name__ == "__main__":
    sock = socket.socket()
    clients = []
    server = Server()
    thread = threading.Thread(target=server.get_socket())
    thread.start()
    thread.join()
```

client.py server.py index.html

```python
import socket

sock = socket.socket()
sock.connect(('localhost', 9090))
sock.send('Hello, server'.encode("UTF-8"))

data = sock.recv(1024)
sock.close()

print(data.decode("UTF-8"))
```

Результат работы программы:

Terminal: Local (2) Local +

```
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab3

C:\Users\rk319\Documents\Study\untitled\lab3>py server.py
Server was started on socket: <socket.socket fd=424, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090)>
Wait for users...
connected: ('127.0.0.1', 54936)
```

```
Terminal:   Local (2)   Local   +
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab3

C:\Users\rk319\Documents\Study\untitled\lab3>py client.py
HTTP/1.0 200 OK
Content-Type: text/html

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Page</title>
</head>
<body>

<H1>WebPage</H1>

</body>
</html>

C:\Users\rk319\Documents\Study\untitled\lab3>
```

Практическое задание 4:

Реализовать двухпользовательский или многопользовательский чат. Реализация многопользовательского часа позволяет получить максимальное количество баллов.

Код программы:

client1.py client2.py client3.py server.py

```python
import socket
from threading import Thread
import threading


class Server:

    def __init__(self):
        sock.bind(('localhost', 9090))
        sock.listen(10)
        print("Server was started on socket:", sock)
        print("Wait for users...")

    def get_socket(self):
        while True:
            client, addr = sock.accept()
            clients.append({"connection": client, "socket": addr})
            print("connected:", addr)
            Thread(target=self.check_messages, args=(client,)).start()

    def check_messages(self, client):
        while True:
            try:
                data = client.recv(4096)
                message = str(data.decode("utf-8"))
                self.broadcast(client, message)
            except Exception as ex:
                self.delete_user_socket(client)
                break
```

client1.py client2.py client3.py server.py

```python
                message = str(data.decode("utf-8"))
                self.broadcast(client, message)
            except Exception as ex:
                self.delete_user_socket(client)
                break

    def delete_user_socket(self, client):
        for user in clients:
            if user["connection"] == client:
                clients.remove(user)
                break

    def broadcast(self, client, message: str):
        message += "\n"
        for user in clients:
            if user["connection"] != client:
                user["connection"].send(message.encode())


if __name__ == "__main__":
    sock = socket.socket()
    clients = []
    server = Server()
    thread = threading.Thread(target=server.get_socket())
    thread.start()
    thread.join()
```

client1.py client2.py client3.py server.py

```python
import socket
import threading

encoding = "UTF-8"
user_name = "user3"


def get_message():
    try:
        while True:
            message = sock.recv(1024).decode(encoding)
            print(message)
    except ConnectionResetError:
        pass


def send_message():
    try:
        while True:
            message = (user_name + ": " + input()).encode(encoding)
            sock.send(message)
    except Exception:
        pass

if __name__ == '__main__':
    sock = socket.socket()
    sock.connect(('localhost', 9090))

```

```python
    sock.connect(('localhost', 9090))

    get_message_thread = threading.Thread(target=get_message)
    get_message_thread.start()

    send_message_thread = threading.Thread(target=send_message)
    send_message_thread.start()

    get_message_thread.join()
    send_message_thread.join()

    sock.close()
```

client1.py client2.py client3.py server.py

```python
import socket
import threading

encoding = "UTF-8"
user_name = "user2"


def get_message():
    try:
        while True:
            message = sock.recv(1024).decode(encoding)
            print(message)
    except ConnectionResetError:
        pass


def send_message():
    try:
        while True:
            message = (user_name + ": " + input()).encode(encoding)
            sock.send(message)
    except Exception:
        pass


if __name__ == '__main__':
    sock = socket.socket()
    sock.connect(('localhost', 9090))

    get_message_thread = threading.Thread(target=get_message)
    get_message_thread.start()

    send_message_thread = threading.Thread(target=send_message)
    send_message_thread.start()

    get_message_thread.join()
    send_message_thread.join()

    sock.close()
```

client1.py client2.py client3.py server.py

```python
import socket
import threading

encoding = "UTF-8"
user_name = "user1"


def get_message():
    try:
        while True:
            message = sock.recv(1024).decode(encoding)
            print(message)
    except ConnectionResetError:
        pass


def send_message():
    try:
        while True:
            message = (user_name + ": " + input()).encode(encoding)
            sock.send(message)
    except Exception:
        pass

if __name__ == '__main__':
    sock = socket.socket()
    sock.connect(('localhost', 9090))

    get_message_thread = threading.Thread(target=get_message)
```

```python
    get_message_thread = threading.Thread(target=get_message)
    get_message_thread.start()

    send_message_thread = threading.Thread(target=send_message)
    send_message_thread.start()

    get_message_thread.join()
    send_message_thread.join()

    sock.close()
```

Результат работы программы:

```
Terminal:  Local  Local (2)  Local (3)  Local (4)  +

C:\Users\rk319\Documents\Study\untitled\lab4>py server.py
Server was started on socket: <socket.socket fd=540, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090)>
Wait for users...
connected: ('127.0.0.1', 54952)
connected: ('127.0.0.1', 54953)
connected: ('127.0.0.1', 54955)
```

```
Terminal:  Local  Local (2)  Local (3)  Local (4)  +
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab4

C:\Users\rk319\Documents\Study\untitled\lab4>py client1.py
Message1
user2: Message2

user3: Message3
```

```
Terminal:  Local  Local (2)  Local (3)  Local (4)  +
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab4

C:\Users\rk319\Documents\Study\untitled\lab4>py client2.py
user1: Message1

Message2
user3: Message3
```

```
Terminal:  Local  Local (2)  Local (3)  Local (4)  +
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab4

C:\Users\rk319\Documents\Study\untitled\lab4>py client3.py
user1: Message1

user2: Message2

Message3
```