МИНЕСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧЕРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«Брестский государственный технический университет»

Кафедра «Интеллектуальные информационные технологии»

Лабораторная работа №2

По дисциплине «Аппаратное и программное обеспечение сетей»

Тема: «Организация UDP – сервера и UDP – клиента»

Выполнил:

студент 3 курса

группы ИИ-21(2)

Пучинский А.А.

Проверил:

Степанчук В.И.

Брест 2023

Цель работы: 1) изучить основы программирования сетевых приложений Windows на базе библиотеки WINSOCK2.H; 2) приобрести навыки по практическому использованию библиотеки для реализации сетевых приложений в среде С++ на базе протоколов TCP.

Вариант 14.

|  |  |  |
| --- | --- | --- |
| Вариант задания для бригады студентов | Номер задания для реализации TCP (UDP) сервера (см. табл. 2). | Номер задания для реализации TCP (UDP) клиента (см. табл. 3). |
| 14 | 6 | 7 |

|  |  |
| --- | --- |
| Номер задания | Функция, выполняемая TCP-сервером |
| 6 | После приема каждой группы из 48 символов сервер отсылает подтверждение в виде контрольной суммы ASCII-кодов символов группы и количество символов, принятых сервером от начала сеанса. Если в потоке принятых символов встречается последовательность «~#~», то сервер отсылает сообщение об окончании сеанса и разрывает соединение. |

Таблица 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вариант | Ввод символов с отсылкой введенной строки по нажатию на клавишу | Ведение файла протокола событий, включающих:  1) время начала и окончания соединения;  2) передаваемую серверу строку и время передачи строки; | Возможность разрыва соединения при помощи команды:  **disconnect <адрес> <порт>** | 2) Автоматическое подключение к серверу с заданным по умолчанию адресом при запуске клиента |
| 7 | PgDn | 1), 2) | + | 2) |

Таблица 3

Реализация UDP-сервера:

import socket

import time

server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

server\_socket.bind(('localhost', 12345))

print("Server started")

last\_check = 0

all\_messages = ""

start\_time = time.time() # Установите время начала соединения

end\_time = None

while True:

data, address = server\_socket.recvfrom(1024)

all\_messages += data.decode()

if len(all\_messages) - last\_check > 48:

control\_sum = sum(map(lambda char: ord(char), all\_messages[last\_check:last\_check + 48]))

print("Сумма: ", control\_sum)

print("Количество всех символов: ", len(all\_messages))

server\_socket.sendto(f"!control\_sum: {control\_sum} {last\_check}".encode(), address)

server\_socket.sendto(f"!control\_len: {len(all\_messages)}".encode(), address)

last\_check += 48

if '~#~' in all\_messages:

server\_socket.sendto(f"!close\_connection".encode(), address)

end\_time = time.ctime()

break

print(data.decode())

server\_socket.sendto(data, address)

# Записываем время начала и окончания соединения

start\_time = time.ctime(start\_time)

end\_time = time.ctime(end\_time)

# Записываем в файл информацию о соединении

with open("log1.txt", "a") as file:

file.write(f"Начало соединения: {start\_time}\n")

file.write(f"Окончание соединения: {end\_time}\n")

file.write(f"Продолжительность соединения: {end\_time - start\_time:.2f} сек\n")

import socket

import keyboard

import time

server\_address = ('localhost', 12345)

# Функция для отправки данных на сервер

def send\_data(sock, data):

    try:

        sock.send(data.encode())

    except:

        print("Ошибка отправки данных")

# Создаем сокет и подключаемся к серверу

client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

client\_socket.connect(server\_address)

print("Клиент подключен к серверу")

try:

    while True:

        if keyboard.is\_pressed("Page Down"):  # Если нажата клавиша PgDn

            user\_input = input("Введите текст для отправки на сервер (или 'disconnect' для разрыва соединения): ")

            if user\_input.lower() == "disconnect":

                send\_data(client\_socket, "~#~")  # Отправляем последовательность "~#~" для разрыва соединения

                break

            send\_data(client\_socket, user\_input)

            response = client\_socket.recv(1024)  # Получение подтверждения от сервера

            print("Подтверждение от сервера:", response.decode())

except KeyboardInterrupt:

    pass

# Закрываем соединение

client\_socket.close()

Реализация UDP-клиента:

from datetime import datetime

import socket

import threading

import keyboard

import pyautogui

file = open("log1.txt", 'a')

# Задайте адрес и порт по умолчанию

default\_ip\_address = 'localhost'

default\_host = 12345

client = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

# client.connect((default\_ip\_address, default\_host)) # Удалите эту строку

file.write(str(datetime.now()) + " --- connection to server\n")

FLAG = True

def press():

while True:

keyboard.wait('home')

pyautogui.press('Enter')

def input\_handler():

global FLAG

all\_messages = ""

start\_time = datetime.now() # Время начала соединения

end\_time = None

server\_address = ('localhost', 12345)

while FLAG:

command = input(">>> ")

sent\_time = datetime.now() # Запомните время передачи сообщения

if '~#~' in command:

client.sendto(command.encode(), server\_address)

# client.close() # Удалите эту строку

file.write(f"Окончание соединения: {end\_time}\n")

file.write(str(datetime.now()) + " --- disconnect from server\n")

break

if 'disconnect' in command:

client.sendto("!disconnect".encode(), server\_address)

# client.close() # Удалите эту строку

FLAG = False

file.write(f"Окончание соединения: {end\_time}\n")

file.write(str(datetime.now()) + " --- disconnect from server\n")

file.close()

break

all\_messages += command

client.sendto(command.encode(), server\_address)

data, \_ = client.recvfrom(1024)

message = data.decode()

if "!close\_connection" in message:

FLAG = False

file.write(f"Окончание соединения: {end\_time}\n")

file.write(str(datetime.now()) + " --- disconnect from server\n")

file.close()

# client.close() # Удалите эту строку

break

elif "!control\_sum" in message:

control\_sum, last\_check = int(message.split()[1]), int(message.split()[2])

data, \_ = client.recvfrom(1024)

print("Сумма: ", control\_sum)

print("Количество всех символов: ", len(all\_messages))

if control\_sum == sum(map(lambda char: ord(char), all\_messages[last\_check:last\_check+48])) and len(all\_messages) == int(data.decode().split()[1]):

print("Сумма верная.")

else:

print("Сумма не верная.")

data, \_ = client.recvfrom(1024)

file.write(f"Sent: {command} --- {sent\_time}\n")

print(f"Sent: {command} --- {sent\_time}")

file.write(message + f" --- {datetime.now()}\n")

print(message)

end\_time = datetime.now() # Время окончания соединения

exit\_thread = threading.Thread(target=press)

input\_thread = threading.Thread(target=input\_handler)

exit\_thread.start()

input\_thread.start()

exit\_thread.join()

input\_thread.join()

file.close()

**Вывод:** 1) изучил основы программирования сетевых приложений Windows на базе библиотеки WINSOCK2.H; 2) приобрёл навыки по практическому использованию библиотеки для реализации сетевых приложений в среде С++ на базе протоколов TCP.