**ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**"ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ"**

Лабораторная работа № 5

Тема: «Использование протокола VNC при разработке систем удаленного администрирования.»

Проверил: Выполнил:

асс. каф. ПИ ст. гр. ПИ-18б

Ищенко А.П. Моргунов А.Г.

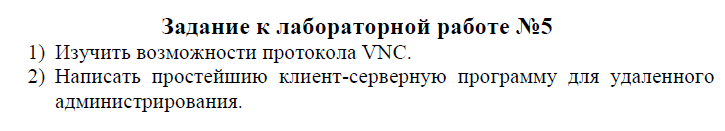
\_\_\_\_.\_\_\_\_.2021г. \_\_\_\_.\_\_\_\_.2021г.

ст. преп. ПИ

Чернышова А.В.

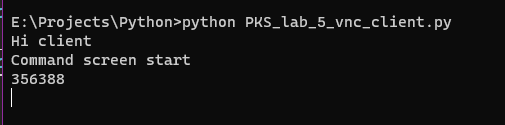
\_\_\_\_.\_\_\_\_.2021г.

Донецк – 2021

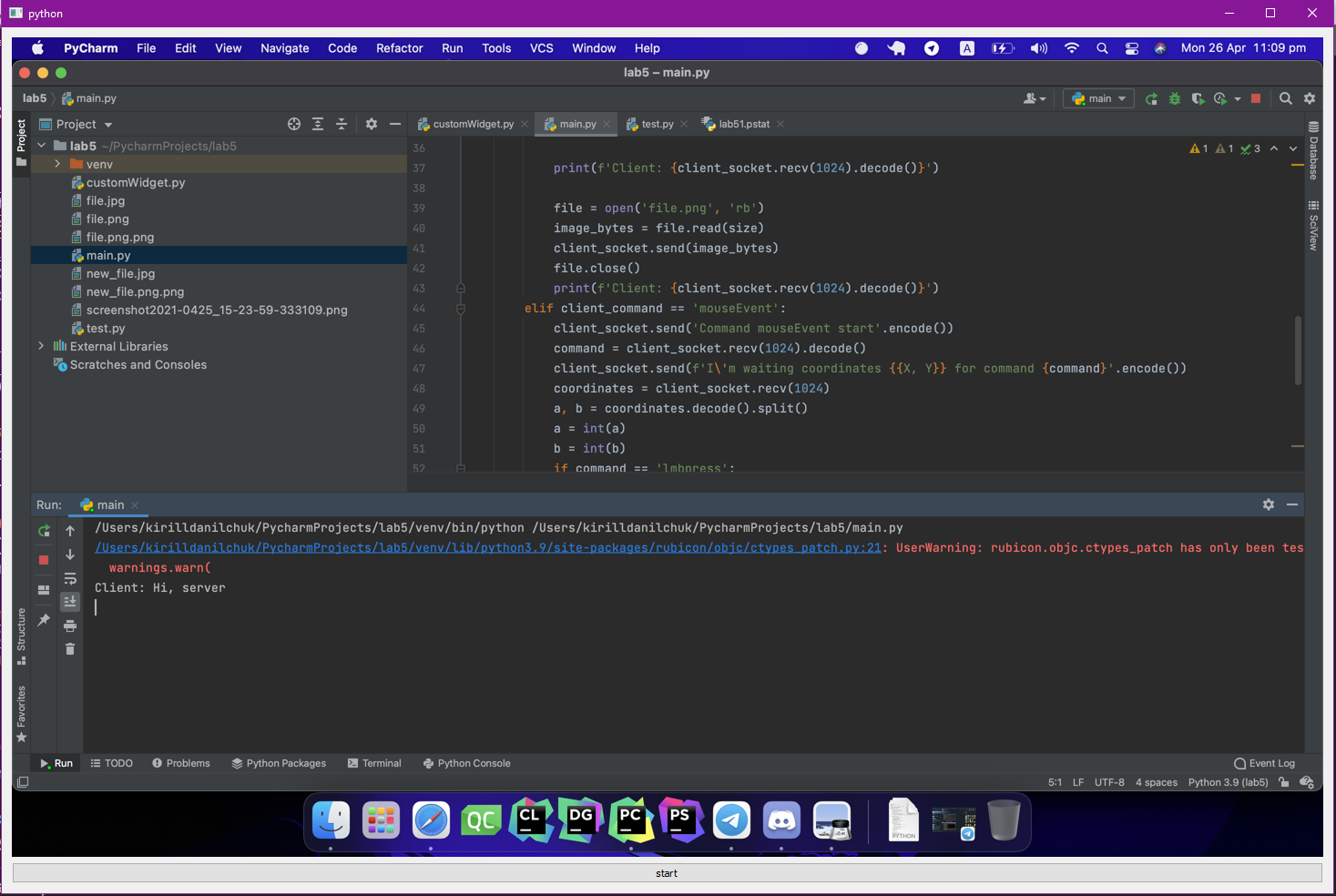


**Клиент**

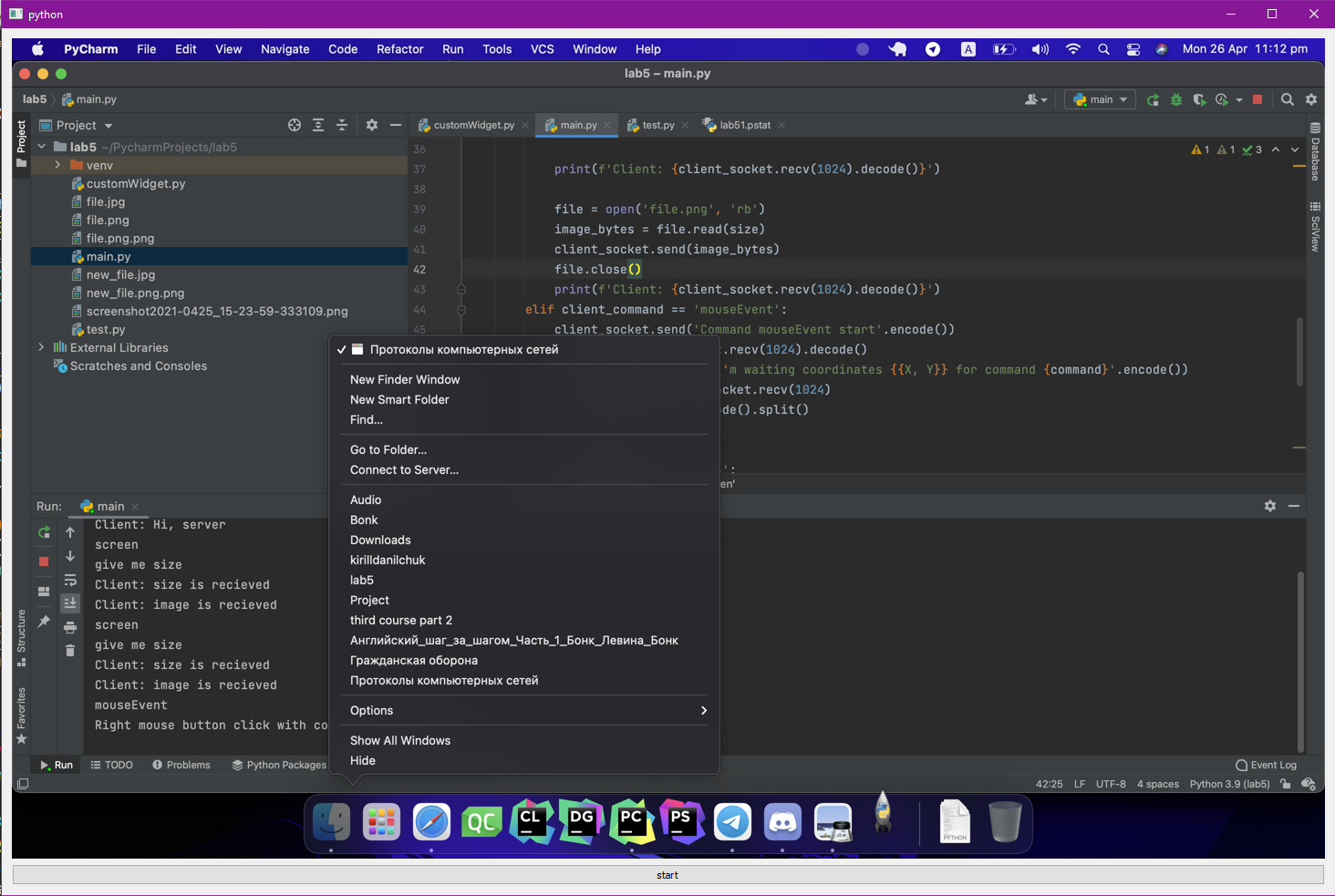
Начало работы клиента (Приветствие сервера, принятая сервером команда, размер скриншота, который будет отправлен)



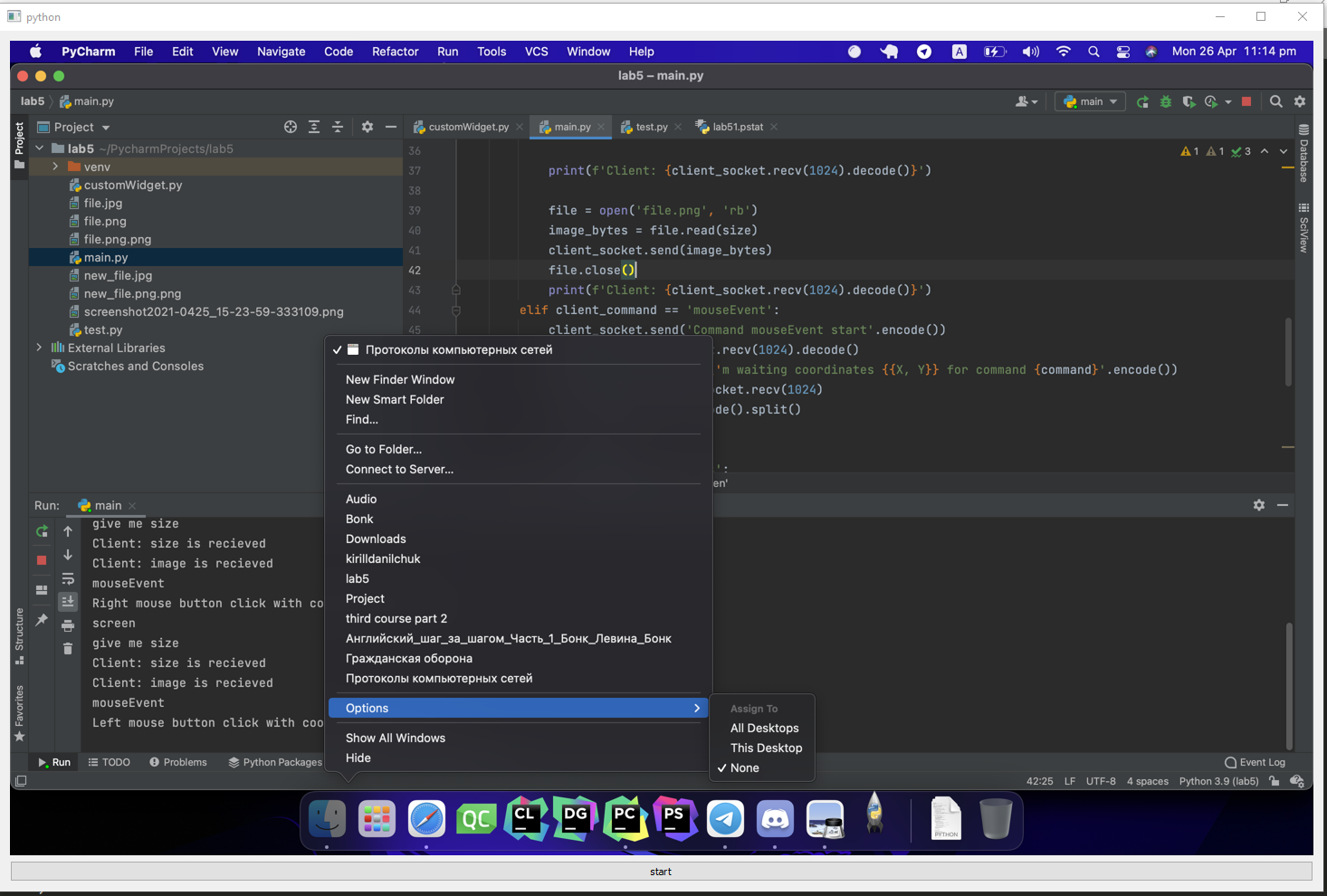
Вывод полученного скрина на форму



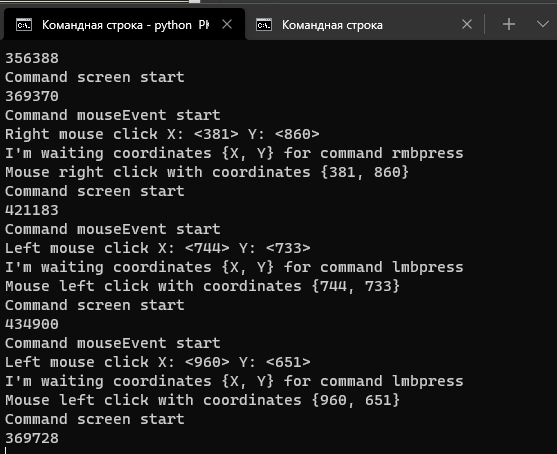
Нажатие ПКМ



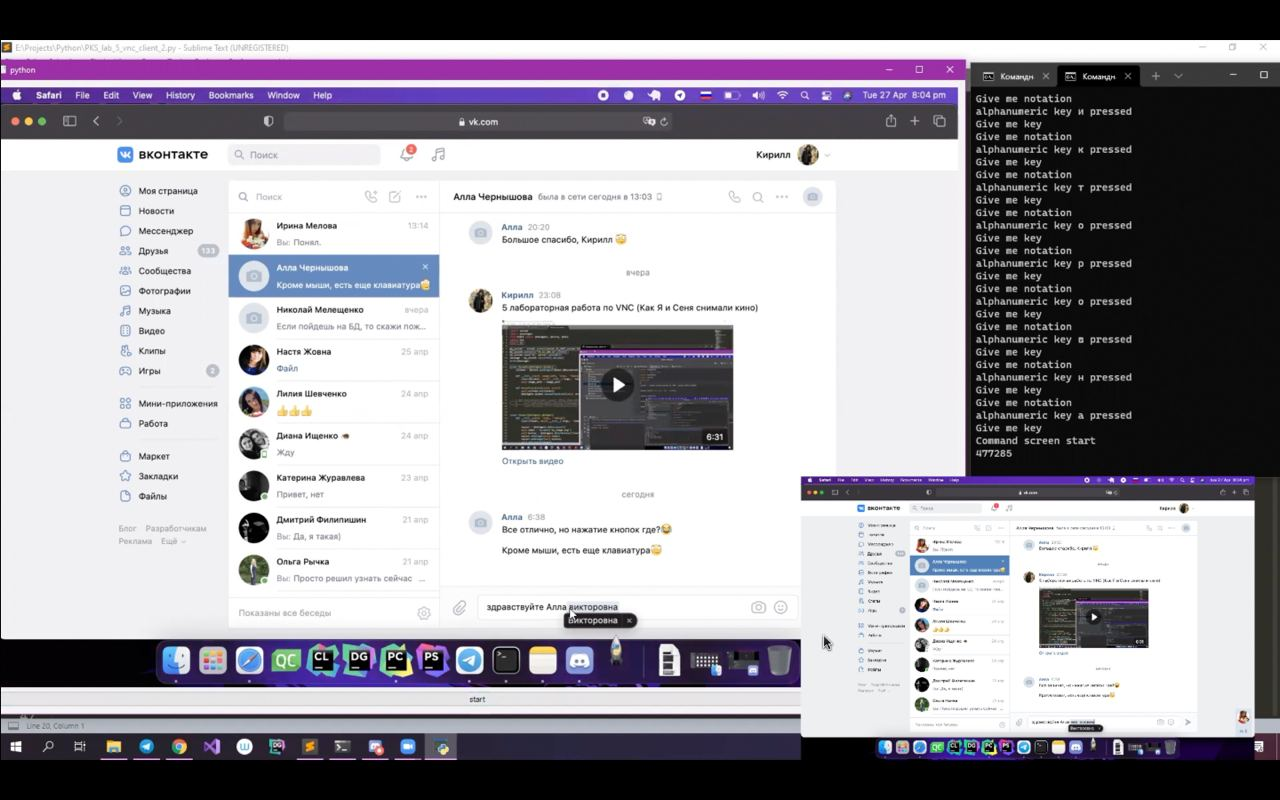
Нажатие ЛКМ



Логирование команд

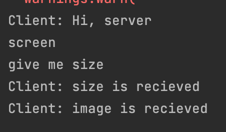


Отправка нажатия клавиш

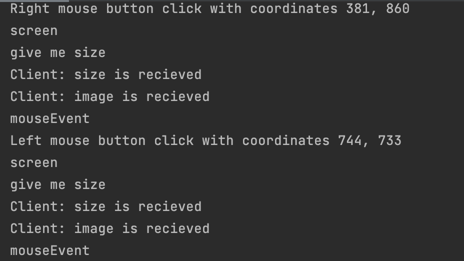


**Сервер**

Приветствие сервера



Логирование команд



Программный код

**Клиент**

from pynput import keyboard

import socket

import pyautogui

from PyQt5 import QtWidgets, QtCore, QtGui

import sys

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

my\_socket.connect(("78.31.180.13", 51000))

my\_socket.send("Hi, server".encode())

message = my\_socket.recv(4096).decode()

print(message)

class MyLabel(QtWidgets.QLabel):

clicked = QtCore.pyqtSignal(QtGui.QMouseEvent)

def \_\_init\_\_(self, image\_path, \*args, \*\*kwargs):

super(MyLabel, self).\_\_init\_\_(\*args, \*\*kwargs)

self.image\_path = image\_path

def mousePressEvent(self, event):

self.clicked.emit(event)

QtWidgets.QLabel.mousePressEvent(self, event)

class Viewer(QtWidgets.QWidget):

def \_\_init\_\_(self, \*args, \*\*kwargs):

super(Viewer, self).\_\_init\_\_(\*args, \*\*kwargs)

layout = QtWidgets.QVBoxLayout()

self.label = MyLabel('new\_file.png')

self.button = QtWidgets.QPushButton("start")

layout.addWidget(self.label)

layout.addWidget(self.button)

self.setLayout(layout)

self.button.clicked.connect(self.on\_button\_clicked)

self.label.clicked.connect(self.mouseClick)

self.button.setAutoDefault(False)

self.button.setDefault(False)

def mouseClick(self, event):

button = event.button()

my\_socket.send("mouseEvent".encode())

print(my\_socket.recv(1024).decode())

if button == 1:

print(f"Left mouse click X: <{event.x()}> Y: <{event.y()}>")

my\_socket.send("lmbpress".encode())

print(my\_socket.recv(1024).decode())

elif button == 2:

print(f"Right mouse click X: <{event.x()}> Y: <{event.y()}>")

my\_socket.send("rmbpress".encode())

print(my\_socket.recv(1024).decode())

coordinates = str(event.x()) + ' ' + str(event.y())

my\_socket.send(coordinates.encode())

print(my\_socket.recv(1024).decode())

pyautogui.sleep(0.3)

self.update\_screen()

def on\_button\_clicked(self):

self.update\_screen()

def update\_screen(self):

my\_socket.send("screen".encode())

print(my\_socket.recv(1024).decode())

my\_socket.send("give me size".encode())

size = int(my\_socket.recv(1024).decode())

print(size)

my\_socket.send("size is recieved".encode())

image = my\_socket.recv(size)

my\_socket.send("image is recieved".encode())

file = open('new\_file.png','wb')

file.write(image)

file.close()

self.label.setPixmap(QtGui.QPixmap.fromImage(QtGui.QImage('new\_file.png').scaled(1440, 900)))

def on\_press(self, key):

my\_socket.send('keyEvent'.encode())

print(my\_socket.recv(1024).decode())

try:

print(f'alphanumeric key {key.char} pressed')

my\_socket.send('key'.encode())

print(my\_socket.recv(1024).decode())

k = key.char

my\_socket.send(k.encode())

except AttributeError:

self.button.setEnabled(False)

print(f'special key {key} pressed')

my\_socket.send('code'.encode())

print(my\_socket.recv(1024).decode())

if key == keyboard.Key.esc:

my\_socket.send('esc'.encode())

elif key == keyboard.Key.space:

my\_socket.send('space'.encode())

elif key == keyboard.Key.backspace:

my\_socket.send('backspace'.encode())

elif key == keyboard.Key.enter:

my\_socket.send('enter'.encode())

else:

my\_socket.send('trash'.encode())

self.button.setEnabled(True)

def on\_release(self, key):

pass

app = QtWidgets.QApplication(sys.argv)

v = Viewer()

v.show()

listener = keyboard.Listener(

on\_press=v.on\_press,

on\_release=v.on\_release

)

listener.start()

app.exec\_()

my\_socket.close()

**сервер**

import ImageGrab

from PIL import Image

from pynput import keyboard

import pyautogui

import socket

import os

controller = keyboard.Controller()

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

socket.bind(('192.168.0.102', 51000))

socket.listen(1)

client\_socket, address\_client = socket.accept()

client\_socket.settimeout(60)

message = client\_socket.recv(4096)

print(f'Client: {message.decode()}')

client\_socket.send('Hi client'.encode())

try:

while True:

client\_command = client\_socket.recv(1024).decode()

print(client\_command)

if client\_command == 'screen':

client\_socket.send('Command screen start'.encode())

print(client\_socket.recv(1024).decode())

image = ImageGrab.grab()

image = image.resize((1440, 900))

image.save('/Users/kirilldanilchuk/PycharmProjects/lab5/file.png')

# screen send

size = os.path.getsize('file.png')

client\_socket.send(str(size).encode())

print(f'Client: {client\_socket.recv(1024).decode()}')

file = open('file.png', 'rb')

image\_bytes = file.read(size)

client\_socket.sendall(image\_bytes)

file.close()

print(f'Client: {client\_socket.recv(1024).decode()}')

elif client\_command == 'mouseEvent':

client\_socket.send('Command mouseEvent start'.encode())

command = client\_socket.recv(1024).decode()

client\_socket.send(f'I\'m waiting coordinates {{X, Y}} for command {command}'.encode())

coordinates = client\_socket.recv(1024)

a, b = coordinates.decode().split()

a = int(a)

b = int(b)

if command == 'lmbpress':

pyautogui.click(button='left', x=a, y=b)

print(f'Left mouse button click with coordinates {a}, {b}')

client\_socket.send(f'Mouse left click with coordinates {{{a}, {b}}}'.encode())

elif command == 'rmbpress':

pyautogui.click(button='right', x=a, y=b)

print(f'Right mouse button click with coordinates {a}, {b}')

client\_socket.send(f'Mouse right click with coordinates {{{a}, {b}}}'.encode())

elif client\_command == 'keyEvent':

client\_socket.send('Give me notation'.encode())

key\_or\_code = client\_socket.recv(1024).decode()

if key\_or\_code == 'key':

client\_socket.send('Give me key'.encode())

key = client\_socket.recv(1024).decode()

controller.press(key)

print('Click ' + key)

elif key\_or\_code == 'code':

client\_socket.send('Give me code'.encode())

key = client\_socket.recv(1024).decode()

print('Click ' + key)

if key == 'enter':

controller.press(keyboard.Key.enter)

print('enter')

elif key == 'esc':

controller.press(keyboard.Key.esc)

print('esc')

elif key == 'space':

controller.press(keyboard.Key.space)

print('space')

elif key == 'backspace':

controller.press(keyboard.Key.backspace)

print('backspace')

elif key == 'trash':

print('Странная кнопка')

elif client\_command == 'exit':

break

finally:

client\_socket.close()

socket.close()