Systemy wbudowane

Laboratorium szóste
SPRAWOZDANIE

Wykonał:
Jakub Pietrus 241174

Prowadzący:
dr inż. Wojciech Rafajłowicz

Data wykonania ćwiczenia:
25.04.2020

Termin zajęć:
Czwartek 11.15 - 13.00

1 Wstęp

Celem ćwiczenia było opracowanie w języku Python serwera na podstawie kalkulatora RPN dostępnego przez protokół telnet dla jednego klienta.
Wykorzystano do tego bibliotekę socket, oraz operator.

2 Wnioski

Python udostępnia bibliotekę socket za pomocą, której w prosty sposób jesteśmy w stanie postawić serwer hosta, oraz klienta. Konfiguracja przebiega sprawnie i szybko, dzięki świetnie napisanej dokumentacji. Największy problem stanowi zakodowanie i odkodowywanie znaków z kodu bajtowego. Należy o tym pamiętać za każdym razem gdy odbieramy, lub wysyłamy jakieś dane. Przesyłanie danych można usprawnić jeszcze bardziej korzystając z biblioteki pickle, która pozwala wymieniać całe obiekty, które python koduje i enkoduje z kodu bajtowego. Stwarza to ogromne możliwośći w wymianie danych między klientem a serwerem.

Fragment kodu programu znajduje się na następnej stronie.

3 Kod

```
import socket

HOST = '127.0.0.1'
PORT = 9990

while True:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind((HOST, PORT))
    s.listen(5)
    clientsocket, address = s.accept()

    while True:
            stack = []
            numbers = []
            expression = ''
            while True:
                data = clientsocket.recv(512)
                while data != ENTER:
                    decoded = data.decode()
                    if len(decoded) > 0:
                        numbers.append(decoded)
                    temp = clientsocket.recv(512)
                    if temp == BACKSPACE:
                        if len(numbers) > 0:
                            numbers.pop()
                        data = b''
                        continue
                    if temp == SPACE:
                        continue
                    data = temp
                for i in numbers:
                    expression += i
                if expression == 'q':
                    exit()
                elif expression == 'clear':
                    stack = []
                    expression = ''
                    numbers = []
                    continue
                elif len(expression) == 0:
                    continue
                stack = calculate(expression.split(), stack)
                print(str(stack))
                clientsocket.send(byte_encode(stack))
                expression = ''
                numbers = []
```