PSI Laboratorium 1

Zespół Z43:

```
Mateusz Brzozowski
Bartłomiej Krawczyk
Jakub Marcowski
Aleksandra Sypuła
```

Zadanie 1

Napisz zestaw dwóch programów – klienta wysyłającego i serwera odbierającego datagramy UDP. Wykonaj ćwiczenie w kolejnych inkrementalnych wariantach (rozszerzając kod z poprzedniej wersji).

Zadanie 1.1

> Klient wysyła a serwer odbiera datagramy o stałym, niewielkim rozmiarze (rzędu kilkudziesięciu bajtów). Datagramy mogą zawierać ustalony „na sztywno” lub generowany napis – np. „abcde….”, „bcdef…”, itd. Powinno być wysyłanych kilka datagramów, po czym klient powinien kończyć pracę. Serwer raz uruchomiony pracuje aż do zabicia procesu. Serwer wyprowadza na stdout adres klienta przysyłającego datagram.

> Wykonaj programy w dwóch wariantach: C oraz Python. Sprawdzić i przetestować działanie „między platformowe”, tj. klient w C z serwerem Python i vice versa.

Napisane po dwa programy w języku C oraz Python - client.c, server.c oraz client.py, server.py działające w protokole UDP. Klient wysyła do serwera określoną liczbę datagramów o stałym rozmiarze. Istnieje możliwość zdefiniowania własnego hosta oraz portu dla klienta przez podanie przy wywołaniu client.c oraz client.py jako pierwszy argument numer hosta a drugi numer portu. Jeśli wartości te nie zostaną podane, klient będzie korzystał z `localhosta` oraz portu `8000`. Serwer można skonfigurować w analogiczny sposób.

Klient wysyła datagramy o wielkości zadeklarowanej w stałej `DATA_GRAM_LENGTH` w ilości `DATA_GRAM_NUMBER`. Przed wysłaniem datagramu, przesyłane dane są wyświetlane na ekranie. Serwer po odbiorze datagramu wyświetla informacje o adresie, z którego zostały wysłane dane, a także otrzymaną zawartość zdekodowaną na `ASCII`.

Rozwiązanie zostało przetestowane w każdej konfiguracji klient-serwer również w przypadku działania między platformowego.

Wszystkie konfiguracje klient-serwer można uruchomić korzystając z docker-compose.yaml. Po wywołaniu `docker compose build`, można wybrać konkretną konfigurację klienta i serwera w wybranych językach np. `z43_client_python_2` z klientem w Pythonie oraz serwerem w C.

Możliwe jest również jedną komendą uruchomienie wszystkich konfiguracji:

```
docker-compose up --build
```

Zadanie 1.2

> Na bazie wersji 1.1 napisać klienta, który wysyła kolejne datagramy o przyrastającej wielkości np. 1, 100, 200, 1000, 2000… bajtów. Sprawdzić jaki był maksymalny rozmiar wysłanego (przyjętego) datagramu. Ustalić z dokładnością do jednego bajta jak duży datagram jest obsługiwany. Wyjaśnić.

> To zadanie można wykonać korzystając z kodu klienta i serwera napisanych w C lub w Pythonie.

Program wykonany w Pythonie z klientem wysyłającym do serwera datagramy o rosnącym rozmiarze. Początkowy rozmiar wysyłanych datagramów określony jest przez zmienną `data_length` i w kolejnych iteracjach zwiększany o skok `jump`. Skok jest zwiększany jeśli wysłanie się powiedzie, a zmniejszany jeśli się nie powiedzie. Przy zerowym skoku osiągamy maksymalną długość wysyłanych danych i klient kończy działanie.

Serwer po odbiorze datagramu wyświetla informacje o rozmiarze odebranego datagramu.

Jeśli rozmiar datagramu jest zbyt duży do wysłania, klient wyświetla informacje o błędzie:

```
Exception while sending data: [WinError 10040] A message sent on a datagram
socket was larger than the internal message buffer or some other network
limit, or the buffer used to receive a datagram into was smaller than the
datagram itself.
```

i koryguje rozmiar danych.

Po uruchomieniu programu maksymalny rozmiar przyjętych danych z datagramu wyniósł 65507B.