Laboratorium **Programowanie w języku Python 2** Wydział Elektrotechniki Automatyki I Informatyki Politechnika Świętokrzyska	
Studia: **Stacionarne I stopnia**	Kierunek: **Informatyka**
Data wykonania: **6.05.2021**	Grupa: **3ID16B**
Imię I nazwisko: **Arkadiusz Więcław**	Temat ćwiczenia: **Serializacja**

Zad 1:

Serwer:

```
import pickle, socket

"""Funkcja zapisuje do pliku  slownik wykorzystujac pickle i odczytuje
z niego"""
def pick_file():
    dict1 = {"www":"bbb", "sas":"ccc", "sas":"fff"}
    pick_out = open("file", "wb")
    print("Zapisano do pliku; ")
    pickle.dump(dict1, pick_out)
    pick_out.close()
    print("Odczytano zawartosc pliku; ")
    pickle_in = open("file", "rb")
    zawartosc = pickle.load(pickle_in)
    print(zawartosc)
pick_file()

""" Funkcja tworzy serwer ktory odczytuje wiadomosc wysylywane przez
klienta dane sa odcztytywane za pomocu
pickle"""
def server_pickle():
    print("\nSerwer zaczyna nasluchiwanie.....")
    HOST = 'localhost'
    PORT = 40008
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind((HOST, PORT))
    s.listen(1)
    conn, addr = s.accept()
    print('Klient polaczyl sie z serwerem', addr)
    data = conn.recv(4096)
    data_variable = pickle.loads(data)
    conn.close()
    print('Dane otrzymane od klienta')
    print(data_variable)

"""Prosty przyklad wykorzystujace pickle"""
def example_pickle():
    Nasa = {'mam' : 'Dad', 'nasa' : 'Ojej Nasa','n' : 2, 'r' : 40000}
```

```
    serial_ex = pickle.dumps(Nasa)
    received = pickle.loads(serial_ex)
    print("\n ", received)

example_pickle()
server_pickle()
```

Wynik:

```
Zapisano do pliku;

Odczytano zawartosc pliku;

{'www': 'bbb', 'sas': 'fff'}


  {'mam': 'Dad', 'nasa': 'Ojej Nasa', 'n': 2, 'r': 40000}


Serwer zaczyna nasluchiwanie.....
```

Client:

```
import pickle, socket

""" Funkcja tworzy klienta który bedzię zapisyweał wiadomość
wykorzystując pickle """
def client_pickle():
    HOST = 'localhost'
    PORT = 40008
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((HOST, PORT))
    variable = "Tekstowanie klienta"
    data_string = pickle.dumps(variable)
    s.send(data_string)
    s.close()
    print ('Wyslano wiadomosc do klienta')

client_pickle()
```

Wynik:

```
Wyslano wiadomosc do klienta
```

Przykład joblib:

```
import joblib, os, socket, threading
from tempfile import mkdtemp

"""Funkcja zapisuję i odczytuje zawartość pliku wykorzystujac
serializacje za pomoca funkcji joblib """
def file_job():
    date = {"powo":"frok", "smart":"abc", "horn":"awsa", "ad":"eed"}
    savedir = mkdtemp()
    filename = os.path.join(savedir, 'Test joblib')
    print("Zapisano dane do pliku: ")
    with open(filename, 'wb') as f:
       joblib.dump(date, f)
    print("Odczyta zawartosc pliku: ")
    with open(filename, 'rb') as f:
        print(joblib.load(f))

file_job()
```

Wynik:

```
Zapisano dane do pliku:

Odczyta zawartosc pliku:

{'powo': 'frok', 'smart': 'abc', 'horn': 'awsa', 'ad': 'eed'}
```

Zad 2:

Serwer:

```
import socket, sys , json
"Funkcja zapisuje dane do pliku a potem jest odczytuje"
def file_json():
    data = {"menu": {"id": "file","value": "File","popup":
{"menuitem":
       [{"value": "New", "onclick": "CreateNewDoc()"},{"value":
"Open", "onclick": "OpenDoc()"}]}}}
    print("Zapisywanie do pliku")
    with open('file_json', 'w') as f:
        json.dump(data, f, sort_keys=True)
    print("Odczytywania zawartosc pliku")
```

```
    with open('file_json', 'r') as f:
        data = json.load(f)
    print(data,"\n")

file_json()
"""
Funkcja pokazuje przyklad demostrujacy proste wykorzystanie biblioteki
json
"""
def example_json():
    date =  '{ "mame":"Arkadiusz", "ag":23}'
    datwe_json = json.loads(date)
    print(datwe_json)
example_json()

def server_json():
    print("\nServer nasłuchuje.....")
    HOST = 'localhost'
    PORT = 40008
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind((HOST, PORT))
    s.listen(1)
    conn, addr = s.accept()
    print('Klinet połączył się z serwerem ', addr)
    data = conn.recv(4096)
    data_variable = json.loads(data)
    conn.close()
    print('Wiadomosc zastała wyslana przez klienta')
    print(data_variable)

server_json()
```

Wynik:

```
Zapisywanie do pliku

Odczytywania zawartosc pliku

{'menu': {'id': 'file', 'popup': {'menuitem': [{'onclick':
'CreateNewDoc()', 'value': 'New'}, {'onclick': 'OpenDoc()', 'value':
'Open'}]}, 'value': 'File'}}


{'mame': 'Arkadiusz', 'ag': 23}
```

```
Server nasłuchuje.....
```

Client:

```
import socket, sys , json
"""
Funkcje ma za zadania wysłać i odebrać wiadomość w postaci formatu
json.
"""
def network_json():
    HOST, PORT = "localhost", 40008
    m = {"id": 4, "name": "Arkadiusz"}
    data = json.dumps(m)
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        print("Klient połączył się z serwerem ")
        sock.connect((HOST, PORT))
        print("Klient wysłał wiadomość do serwera ")
        sock.sendall(bytes(data,encoding="utf-8"))
    finally:
        sock.close()
network_json()
```

Wynik:

```
Klient połączył się z serwerem

Klient wysłał wiadomość do serwera
```

Zad 3:

Serwer:

```
import socket, sys, msgpack

"""Funkcja prezentuje prosty przykład serializacji MessagePack """
def seriable_msgpack():
    data = {
        "list": [135, 4, 18.99, 56789, "Word is dead"],
        "string": "John Snow",
        "int": 259
    }
    packed = msgpack.packb(data)
```

```
    data_loaded = msgpack.unpackb(packed)
    print(data_loaded)

seriable_msgpack()
"""
Funkcja tworzy serwer TCP który będzie odbierał wiadomość od klienta.
"""
def server_msg():
    print("\nServer nasłuchuję.....")
    HOST = 'localhost'
    PORT = 40008
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind((HOST, PORT))
    s.listen(1)
    conn, addr = s.accept()
    print('Klient połaczył się z serwerem', addr)
    data = conn.recv(4096)
    data_variable = msgpack.unpackb(data, use_list=False, raw=False)
    conn.close()
    print('Dane klienta')
    print(data_variable)

server_msg()
```

Wynik:

```
{'list': [135, 4, 18.99, 56789, 'Word is dead'], 'string': 'John
Snow', 'int': 259}


Server nasłuchuję.....
```

Client:

```
import socket, sys, msgpack, threading
"""
Funkcje ma za zadania wysłać wiadomość o różnym typie danych
wykorzystując biblioteke MessagePack.
"""
def client_msg():
    HOST, PORT = "localhost", 40008
    str_w = "John Snow"
    int_t = 125
    fl_t = 456.28
```

```
    bol_t = True
    data = msgpack.packb((str_w, int_t, fl_t, bol_t ),  use_bin_type =
True)
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        print("Klient połączył się z serwerem")
        sock.connect((HOST, PORT))
        print("Klient wysłał wiadomość do serwera")
        sock.sendall(data)
    finally:
        sock.close()
client_msg()
```

Wynik:

```
Klient połączył się z serwerem

Klient wysłał wiadomość do serwera
```