Assignment 3.1

ENE 419 – Computer Networks. Due: Nov. 28 (tentative)

(short) Programming assignment (individual assignment)

You will develop an asynchronous web socket client. Your client is subscribed to a streaming channel that constantly transmits textual information.

The receiving information is so called market data of Bitcoin (this is equivalent to buying/selling information about Bitcoin at the moment, it's same as stock market.) Your client should receive the market data of Bitcoin@KRW (Korean Won) market and print out the formatted information about trading market.

For testing:

Try wscat to receive the data first and take a look.

How to install:

```
$ npm install -g wscat
$ wscat -c wss://api.upbit.com/websocket/v1
connected (press CTRL+C to quit)
```

After connection, your input to web socket:

```
[{"ticket":"UNIQUE_TICKET"},{"type":"trade","codes":["KRW-
BTC"]},{"type":"orderbook","codes":["KRW-BTC"]}]
```

The outputs should be similar to the following: Two types – "orderbook" and "trade"


```
{"type":"orderbook","code":"KRW-
BTC","timestamp":1573459450456,"total_ask_size":9.85837409,"total_bid_size":
11.02657870,"orderbook_units":[{"ask_price":1.0239E7,"bid_price":1.0227E7,"a
sk_size":2.15998779,"bid_size":0.8738},{"ask_price":1.024E7,"bid_price":1.02
26E7,"ask_size":0.00558381,"bid_size":1.63319311},{"ask_price":1.0241E7,"bid
_price":1.0225E7,"ask_size":0.31548534,"bid_size":1.33452901},{"ask_price":1
.0242E7,"bid_price":1.0224E7,"ask_size":0.476535,"bid_size":1.10566939},{"as
k_price":1.0244E7,"bid_price":1.0223E7,"ask_size":1.276,"bid_size":0.05},{"a
sk_price":1.025E7,"bid_price":1.0222E7,"ask_size":0.1056932,"bid_size":0.001
},{"ask_price":1.0252E7,"bid_price":1.0221E7,"ask_size":1.0,"bid_size":1.104
89188},{"ask_price":1.0253E7,"bid_price":1.022E7,"ask_size":0.3108,"bid_size
":0.09164093},{"ask_price":1.0254E7,"bid_price":1.0219E7,"ask_size":0.8,"bid
_size":0.52252612},{"ask_price":1.0257E7,"bid_price":1.0218E7,"ask_size":0.6
569,"bid_size":2.03877255},{"ask_price":1.0258E7,"bid_price":1.0217E7,"ask_s
ize":0.14637194,"bid_size":1.02936282},{"ask_price":1.0259E7,"bid_price":1.0
216E7,"ask_size":0.08990106,"bid_size":0.0097231},{"ask_price":1.026E7,"bid_
price":1.0215E7,"ask_size":0.3045,"bid_size":0.92357427},{"ask_price":1.0261
E7,"bid_price":1.0214E7,"ask_size":1.76361595,"bid_size":0.0506969},{"ask_pr
ice":1.0262E7,"bid_price":1.0213E7,"ask_size":0.447,"bid_size":0.25719862}],
"stream_type":"REALTIME"}

{"type":"trade","code":"KRW-
BTC","timestamp":1573459450571,"trade_date":"2019-11-
11","trade_time":"08:04:10","trade_timestamp":1573459450000,"trade_price":10
239000.0,"trade_volume":0.11325823,"ask_bid":"BID","prev_closing_price":1049
6000.00000000,"change":"FALL","change_price":257000.00000000,"sequential_id"
:1573459450000000,"stream_type":"REALTIME"}
```


…

For your program:

If orderbook, print out the first element in “orderbook_units”. If trade, print out the three categories like the following example. The printing order is as they arrive at the client.

$./your_stream_client.py

```
ask_price: 1.0239E7, bid_price: 1.0227E7, ask_size: 2.15998779, bid_size:
0.8738

trade_price:10239000.0, trade_volume: 0.11325823, ask_bid:BID

…
```

Please refer to the following references for sample:

For python 2.7 https://pypi.org/project/websocket_client/

For python >= 3.6.1 https://websockets.readthedocs.io/en/stable/intro.html

You may need to know how to process json format in python.

If you choose to do this in other languages, feel free to do so. Just let me know first.

SUBMISSION:

1. Print your source code (python or others) and submit it in class.
2. Print the output of your program and submit in class (first 10 responses only)