Limit Order Book Simulation

- Exchanges are marketplaces where users can buy and sell securities
- When you buy or sell a security you can ask for one of the two things
    - Price guarantee i.e you sell at a minimum price or buy at a maximum price
    - Time guarantee i.e you buy or sell immediately at whatever price is available
- Limit Orders
    - Price guarantee is provided by an order type called a limit order
    - Eg: User X places a limit sell order for 10 units of some stock at $10 each
        - Meaning X is willing to sell 10 units of some stock he/she owns for $10 or more
- Market Orders
    - Time guarantee is provided by an order type called as a market order
    - Eg: User Y places a market order for buying 10 units of some stock
        - Meaning Y is willing to buy 10 units of some stock at whatever price but as soon as possible
- Limit orders give liquidity and market orders take liquidity
- The set of all limit orders at any time is called a limit order book
    - Eg: https://pro.coinbase.com/trade/BTC-USD
- The best ask/bid price at any point is called the BBO (Best Bid Offer)

The goal of this exercise to create a Limit order book class.

The limit order book class should support the following methods
Assuming,

1. All users have infinite balances
2. Limit orders don't cross the BBO
3. A users market order can match with their limit order ie self-trading is allowed (to keep the exercise simple). In real markets self-trading is not allowed.

1. add_limit_order(side, user_id, quantity, price)
    a. Adds a new limit order to the book
    b. Returns a order_id that is unique for each call
2. place_market_order(side, quantity)
    a. Returns the filled quantity and average price for the filled quantity
3. cancel_limit_order(order_id)
    a. Cancel a previously placed limit order ie remove the order from the book
4. bbo()
    a. Prints the current best bid and offer

Example

Let’s instantiate a new orderbook and add some orders to it.

```
lob_example = Limit()
lob_example.add_limit_order(‘ask’, ‘alice’, 10, 100)
lob_example.add_limit_order(‘ask’, ‘bob’, 5, 90)
lob_example.add_limit_order(‘bid’, ‘charles’, 20, 85)
lob_example.add_limit_order(‘bid’, ‘dave’, 10, 80)
```

The orderbook will look like the following

```
{
      “asks”: [
            {
                  “order_id”: 1,
                  “user_id: “alice”,
                  “quantity”: 10,
                  “price”: 100
            },
            {
                  “order_id”: 2,
                  “user_id”: “bob”,
                  “quantity”: 5,
                  “price”: 90
            }
      ],
      “bids”: [
            {
                  “order_id”: 3,
                  “user_id: “charles”,
                  “quantity”: 20,
                  “price”: 85
            },
            {
                  “order_id”: 4,
                  “user_id”: “dave”,
                  “quantity”: 10,
                  “price”: 80
            }
      ]
```

```
}
```

The BBO for this orderbook is [85, 90], since those are the best prices available on either side.

Now, let's try and add another limit order

```
lob_example.add_limit_order('ask', 'eve', 10, 95)
```

This will result in the following orderbook

```
{
    "asks": [
        {
            "order_id": 1,
            "user_id: "alice",
            "quantity": 10,
            "price": 100
        },
        {
            "order_id": 5,
            "user_id": "eve",
            "quantity": 10,
            "price": 95
        },
        {
            "order_id": 2,
            "user_id": "bob",
            "quantity": 5,
            "price": 90
        }
    ],
    "bids": [
        {
            "order_id": 3,
            "user_id: "charles",
            "quantity": 20,
            "price": 85
        },
        {
            "order_id": 4,
            "user_id": "dave",
            "quantity": 10,
            "price": 80
```

```
            }
        ]
}
```

Let's try and place a market order to consume some of this liquidity.

```
lob_example.place_market_order('ask', 12)
```

This market order will start from the best ask and keep consuming limit orders until it gets filled up, or runs out of limit orders. In this case, it will fully consume order_id 2 and remove 7 units from order_id 5. The total price at which the market order was matched is 5 * 90 + 7 * 95

The return for this method will be [12, 92.91667]

And the orderbook will now look like the following

```
{
    "asks": [
        {
            "order_id": 1,
            "user_id: "alice",
            "quantity": 10,
            "price": 100
        },
        {
            "order_id": 5,
            "user_id": "eve",
            "quantity": 3,
            "price": 95
        }
    ],
    "bids": [
        {
            "order_id": 3,
            "user_id: "charles",
            "quantity": 20,
            "price": 85
        },
        {
            "order_id": 4,
            "user_id": "dave",
            "quantity": 10,
            "price": 80
```

```
            }
      ]
}
```

Let’s now cancel order 3

```
lob_example.cancel_limit_order(3)
```

The orderbook will look like the following

```
{
      “asks”: [
            {
                  “order_id”: 1,
                  “user_id: “alice”,
                  “quantity”: 10,
                  “price”: 100
            },
            {
                  “order_id”: 5,
                  “user_id”: “eve”,
                  “quantity”: 3,
                  “price”: 95
            }
      ],
      “bids”: [
            {
                  “order_id”: 4,
                  “user_id”: “dave”,
                  “quantity”: 10,
                  “price”: 80
            }
      ]
}
```

And the BBO for this orderbook will be [80, 95]