Develop a demo console app (preferably C#) – a TRADE NETTER that behaves in the following way:

- Functionally, it processes a list of trades, calculating the **PNL** which indicates how much money has been made/lost on buying and selling.
- Firstly, a balancing process is run which matches buys and sells, in a FIFO manner, depending on the order or their entry:
    - BUY 2 lots + SELL 1 lot + BUY 3 lots
        - ⇨ PNL 1 lot + BUY 1 lot + BUY 3 lots
        - ⇨ *PNL 1 lot + BUY 4 lots*
    - BUY 2 lots + BUY 1 lot + SELL 4 lots
        - ⇨ BUY 3 lots + SELL 4 lots
        - ⇨ *SELL 1 lot*
- The resulting **PNL** is the gain or loss of the balancing process (if we have a trade buying for 100 and selling for 110, we have made a profit of 10 for each unit). The amount of the PNL is negative is we lose money and positive is we receive money.
- The **balancing algorithm** is calculated in the order of first come/first processed.

Some examples:

1. Three trades with the same direction:

INPUT	**Direction**	**Quantity**	**Price**	**Underlying**
Trade 1	Buy	2	100	Oil
Trade 2	Buy	2	110	Oil
Trade 3	Buy	3	102	Oil
Output				
Netted **PNL**	-	-	-	-

2. Two trades with the opposite directions of the same quantity:

INPUT	**Direction**	**Quantity**	**Price**	**Underlying**
Trade 1	Buy	2	100	Oil
Trade 2	Sell	2	110	Oil
Output				
Netted **PNL**	-	-2*100 + 2*110	-	-

3. Two opposite trades with different quantities:

INPUT	**Direction**	**Quantity**	**Price**	**Underlying**
Trade 1	Buy	1	100	Oil
Trade 2	Sell	4	110	Oil
Output				
Netted **PNL**	-	-1*100 + 1*110	-	-

4. Trades are processed in FIFO manner (in the order they are placed in the main input list):

INPUT	Direction	Quantity	Price	Underlying
Trade 1	Buy	1	100	Oil
Trade 2	Sell	4	110	Oil
Trade 3	Buy	4	120	Oil
Output				
Netted **PNL**	-	(-1*100 + 1*110) + (-3*120 + 3*110)	-	-

5. Different Underlying result however only one netted PNL:

INPUT	Direction	Quantity	Price	Underlying
Trade 1	Buy	1	100	Oil
Trade 2	Sell	4	110	Gas
Trade 3	Buy	2	120	Gas
Trade 4	Sell	5	115	Oil
Output				
Netted **PNL**	-	(-1*100 + 1*115) + (2*110 - 2*120)	-	-

- Try to use SOLID principles and an OOP architecture.
- The Trade Model needs to have a Direction, a Quantity (int), a Price (double), and an Underlying.
- The Core of the Algorithm will take a list (or other data structure) of trades and return the PNL.
- The efficiency of the algorithm is taking in consideration when reviewing the algorithm, but correctness is the most important factor.
- Unit tests are not required, rather nice to have.