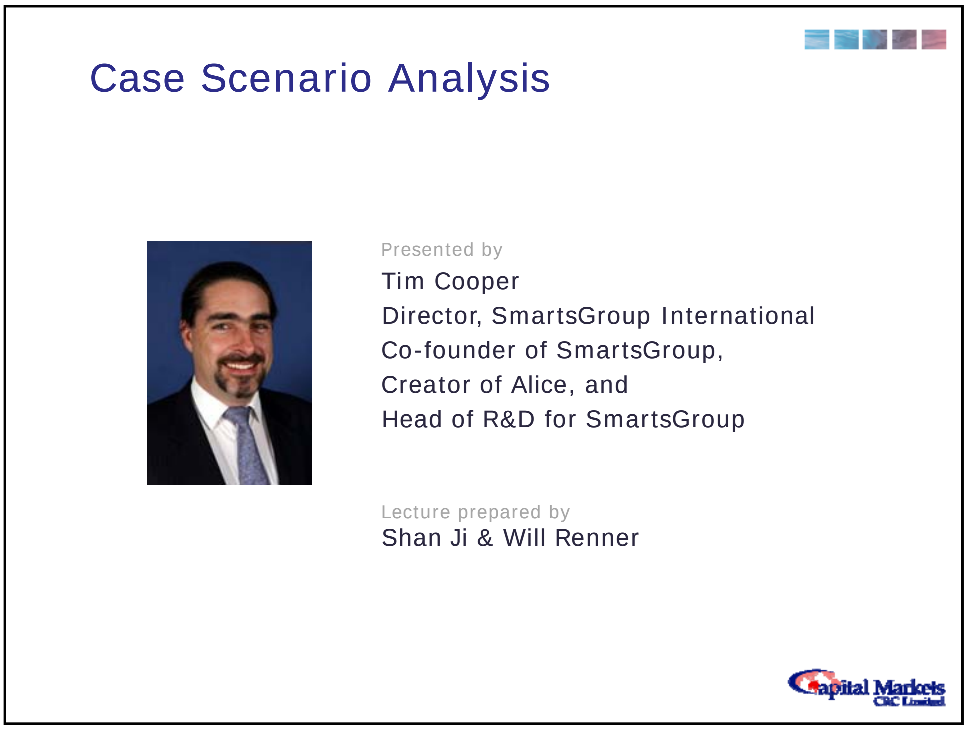
Capital Markets CRC Limited

MARKET DATA ANALYSIS USING ALICE

Case Scenario Analysis

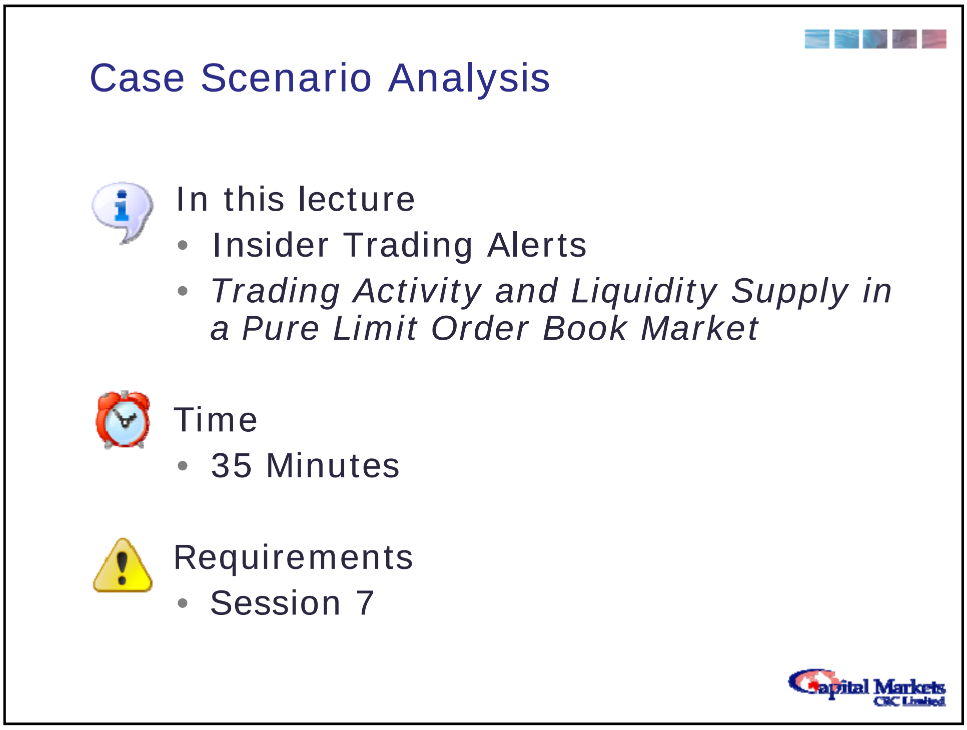


Case Scenario Analysis

Presented by

Tim Cooper
Director, SmartsGroup International
Co-founder of SmartsGroup,
Creator of Alice, and
Head of R&D for SmartsGroup

Lecture prepared by
Shan Ji & Will Renner

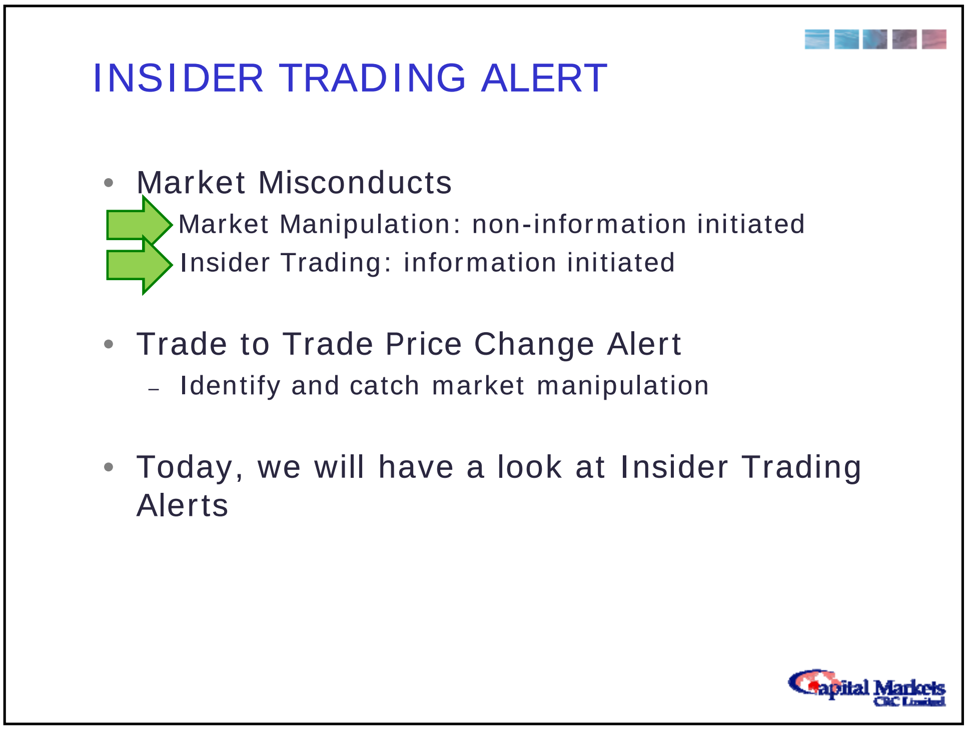


Case Scenario Analysis

In this lecture
- Insider Trading Alerts
- *Trading Activity and Liquidity Supply in a Pure Limit Order Book Market*

Time
- 35 Minutes

Requirements
- Session 7



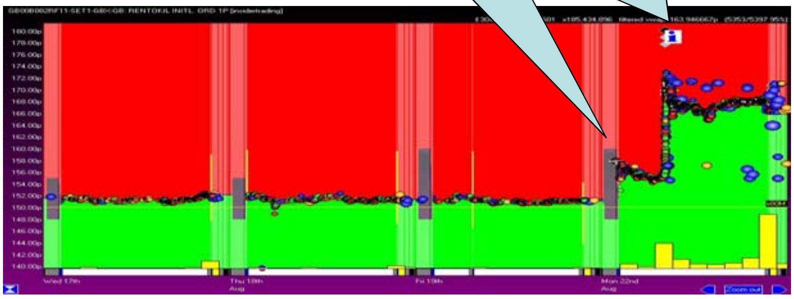
INSIDER TRADING ALERT

- Market Misconducts
  - ➡ Market Manipulation: non-information initiated
  - ➡ Insider Trading: information initiated
- Trade to Trade Price Change Alert
  - – Identify and catch market manipulation
- Today, we will have a look at Insider Trading Alerts

INSIDER TRADING ALERT

Example

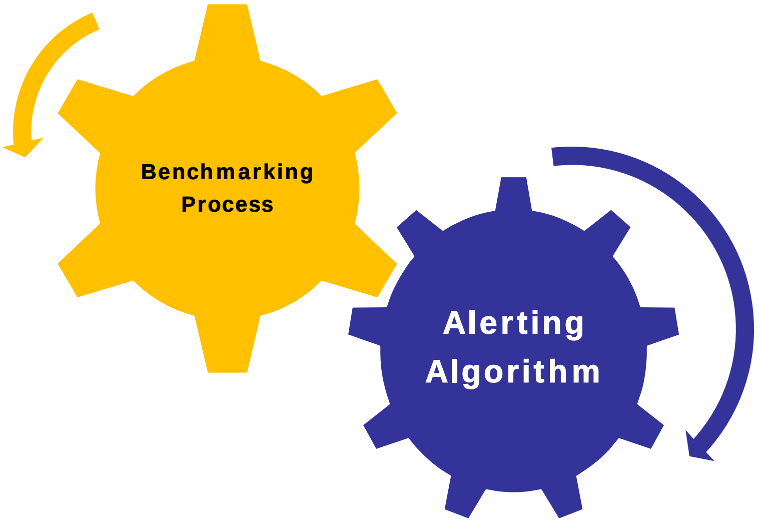
- Detects unusual price movements prior to information releases.
- Identify the participant that makes the most movement to the suspect security



Insider trading prior to announcement

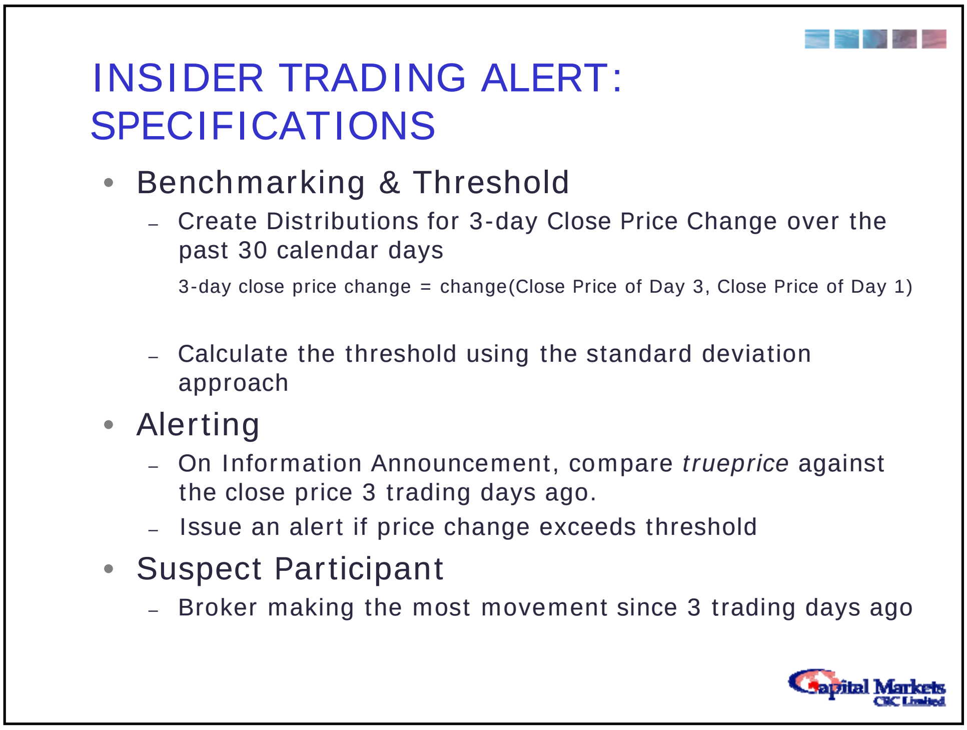
Information announcement

ALERT DESIGN



Benchmarking Process

Alerting Algorithm

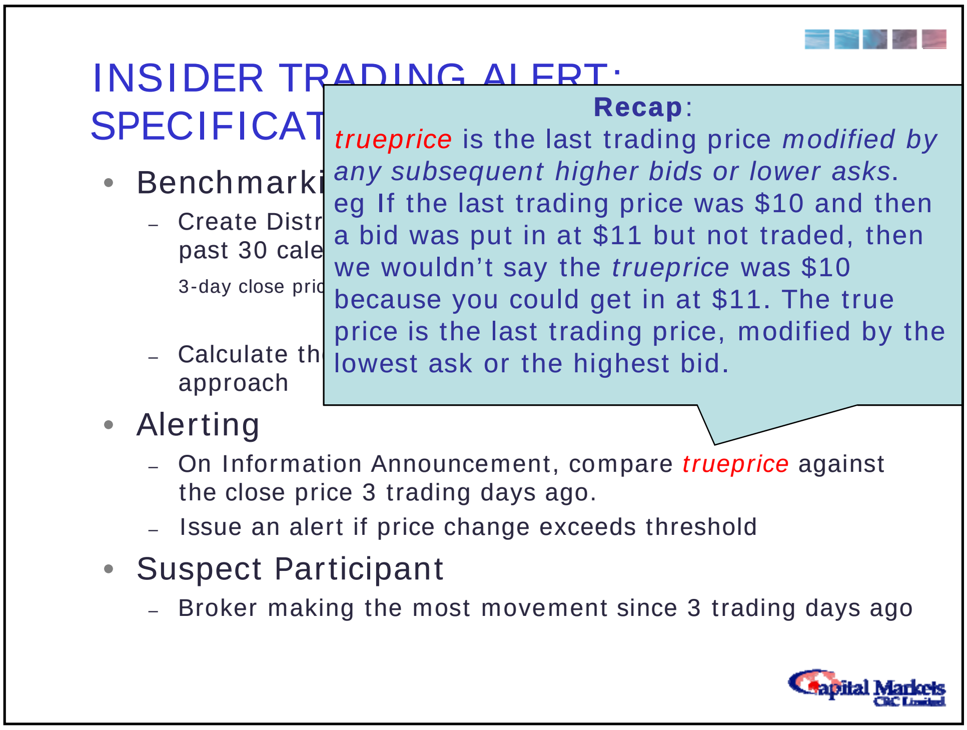


INSIDER TRADING ALERT: SPECIFICATIONS

- Benchmarking & Threshold
  - Create Distributions for 3-day Close Price Change over the past 30 calendar days

    3-day close price change = change(Close Price of Day 3, Close Price of Day 1)

  - Calculate the threshold using the standard deviation approach
- Alerting
  - On Information Announcement, compare *trueprice* against the close price 3 trading days ago.
  - Issue an alert if price change exceeds threshold
- Suspect Participant
  - Broker making the most movement since 3 trading days ago



INSIDER TRADING ALERT: SPECIFICAT

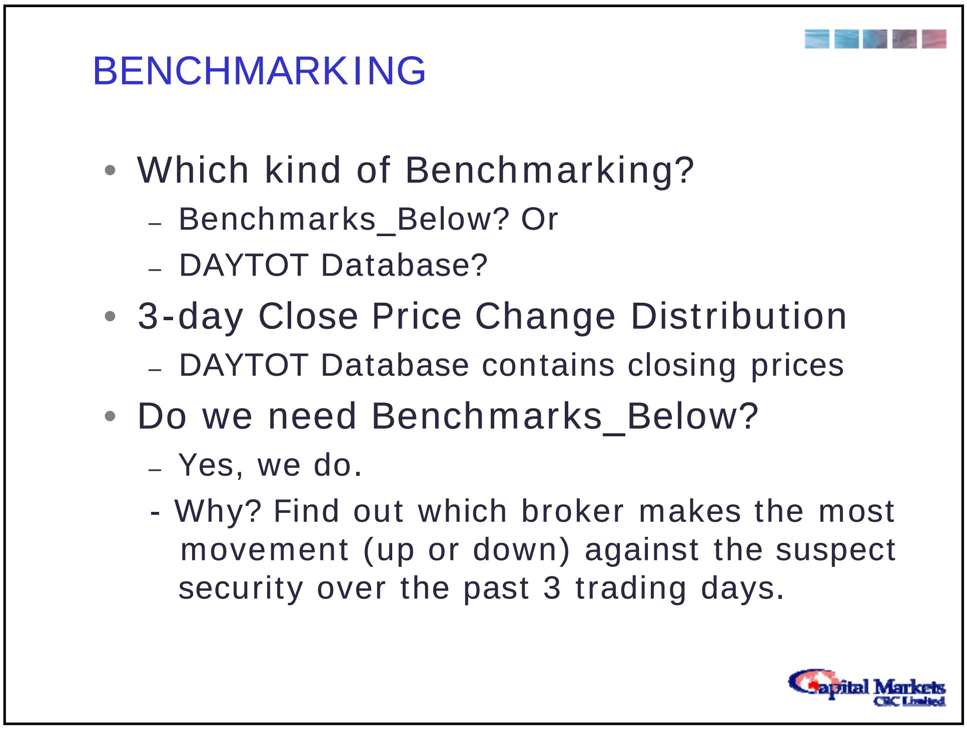
- Benchmarki
  - Create Distr past 30 cale

    3-day close pric

  - Calculate th approach

Recap:
trueprice is the last trading price *modified by any subsequent higher bids or lower asks.*
eg If the last trading price was $10 and then a bid was put in at $11 but not traded, then we wouldn't say the *trueprice* was $10 because you could get in at $11. The true price is the last trading price, modified by the lowest ask or the highest bid.

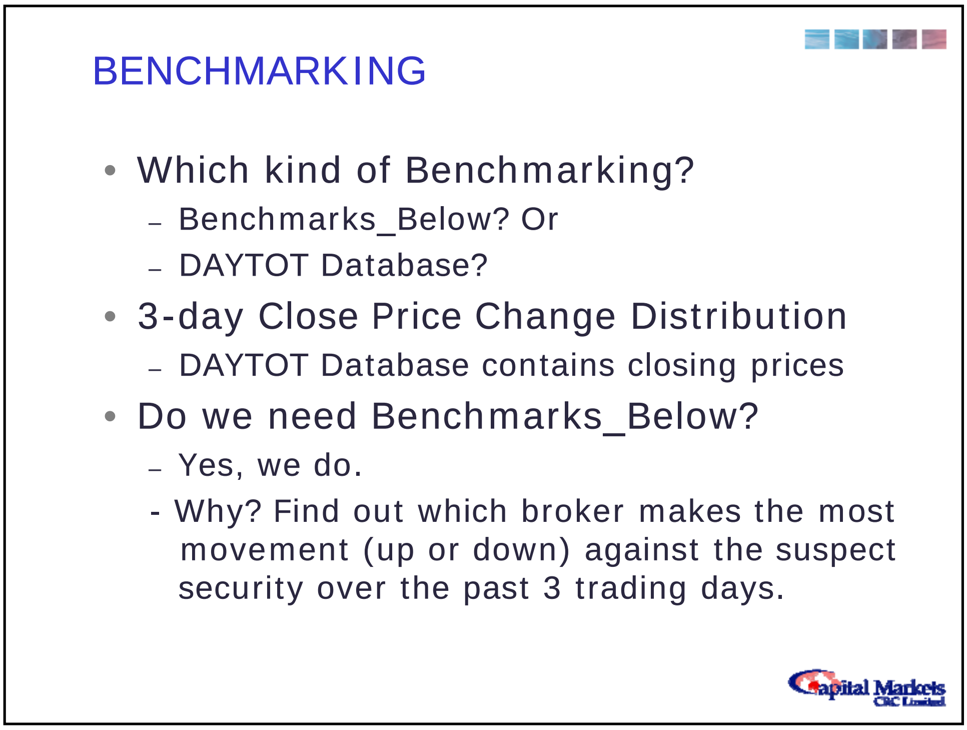
- Alerting
  - On Information Announcement, compare *trueprice* against the close price 3 trading days ago.
  - Issue an alert if price change exceeds threshold
- Suspect Participant
  - Broker making the most movement since 3 trading days ago



BENCHMARKING

- Which kind of Benchmarking?
  - Benchmarks_Below? Or
  - DAYTOT Database?
- 3-day Close Price Change Distribution
  - DAYTOT Database contains closing prices
- Do we need Benchmarks_Below?
  - Yes, we do.
  - Why? Find out which broker makes the most movement (up or down) against the suspect security over the past 3 trading days.

Capital Markets CRC Limited



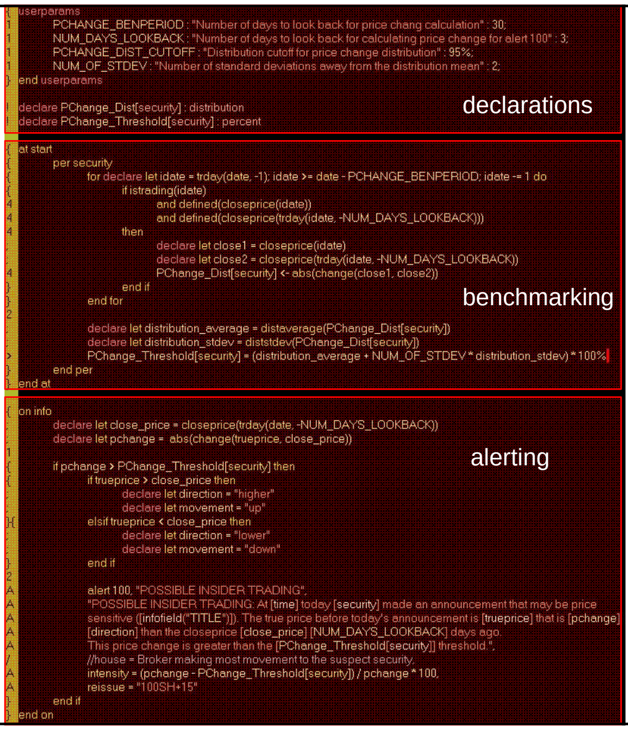
BENCHMARKING

- Which kind of Benchmarking?
  - Benchmarks_Below? Or
  - DAYTOT Database?
- 3-day Close Price Change Distribution
  - DAYTOT Database contains closing prices
- Do we need Benchmarks_Below?
  - Yes, we do.
  - Why? Find out which broker makes the most movement (up or down) against the suspect security over the past 3 trading days.

Capital Markets CRC Limited

ALERTING ALGORITHMS

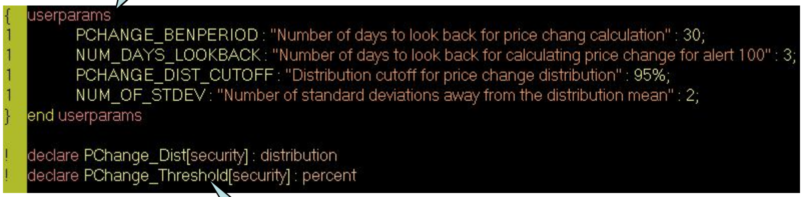
- **Trigger Alert On Which "When Clause"?**
  on info
- **Alert Code: 100**
- **Suspect Parameter**
  house=broker making most movement (up or down) since 3 trading days ago.
- **Intensity Parameter**
  *intensity=(Trueprice - Threshold) / Trueprice * 100*
- **Reissue Parameter**
  Reissue = "100SH+15"
  - Reissue once intensity increases by 15 for each suspect security/broker pair



```
userparams
        PCHANGE_BENPERIOD : "Number of days to look back for price chang calculation" : 30;
        NUM_DAYS_LOOKBACK : "Number of days to look back for calculating price change for alert 100" : 3;
        PCHANGE_DIST_CUTOFF : "Distribution cutoff for price change distribution" : 95%;
        NUM_OF_STDEV : "Number of standard deviations away from the distribution mean" : 2;
end userparams

declare PChange_Dist[security] : distribution
declare PChange_Threshold[security] : percent

at start
        per security
                for declare let idate = trday(date, -1); idate >= date - PCHANGE_BENPERIOD; idate -= 1 do
                        if istrading(idate)
                                and defined(closeprice(idate))
                                and defined(closeprice(trday(idate, -NUM_DAYS_LOOKBACK)))
                        then
                                declare let close1 = closeprice(idate)
                                declare let close2 = closeprice(trday(idate, -NUM_DAYS_LOOKBACK))
                                PChange_Dist[security] <- abs(change(close1, close2))
                        end if
                end for

                declare let distribution_average = distaverage(PChange_Dist[security])
                declare let distribution_stdev = diststdev(PChange_Dist[security])
                PChange_Threshold[security] = (distribution_average + NUM_OF_STDEV * distribution_stdev) * 100%
        end per
end at

on info
        declare let close_price = closeprice(trday(date, -NUM_DAYS_LOOKBACK))
        declare let pchange =  abs(change(trueprice, close_price))

        if pchange > PChange_Threshold[security] then
                if trueprice > close_price then
                        declare let direction = "higher"
                        declare let movement = "up"
                elsif trueprice < close_price then
                        declare let direction = "lower"
                        declare let movement = "down"
                end if

                alert 100, "POSSIBLE INSIDER TRADING",
                "POSSIBLE INSIDER TRADING: At [time] today [security] made an announcement that may be price
                sensitive ([infofield("TITLE")]). The true price before today's announcement is [trueprice] that is [pchange]
                [direction] than the closeprice [close_price] [NUM_DAYS_LOOKBACK] days ago.
                This price change is greater than the [PChange_Threshold[security]] threshold.",
                //house = Broker making most movement to the suspect security,
                intensity = (pchange - PChange_Threshold[security]) / pchange * 100,
                reissue = "100SH+15"
        end if
end on
```

DECLARATIONS

User Parameters



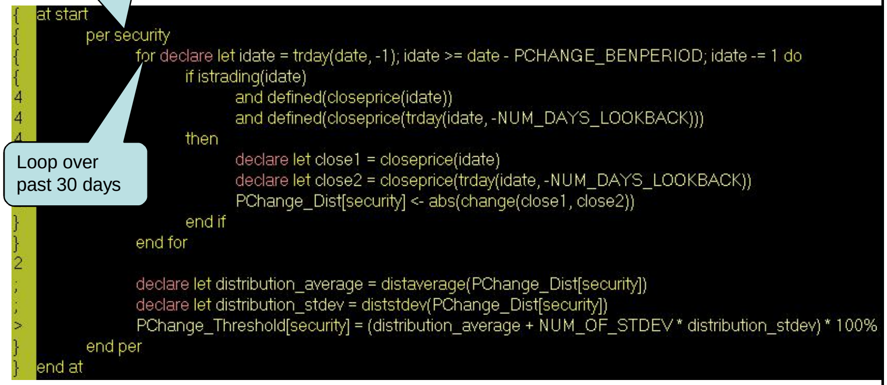
```
userparams
        PCHANGE_BENPERIOD : "Number of days to look back for price chang calculation" : 30;
        NUM_DAYS_LOOKBACK : "Number of days to look back for calculating price change for alert 100" : 3;
        PCHANGE_DIST_CUTOFF : "Distribution cutoff for price change distribution" : 95%;
        NUM_OF_STDEV : "Number of standard deviations away from the distribution mean" : 2;
end userparams

declare PChange_Dist[security] : distribution
declare PChange_Threshold[security] : percent
```

Global Variables

BENCHMARKING

- Looping DAYTOT Database At Start

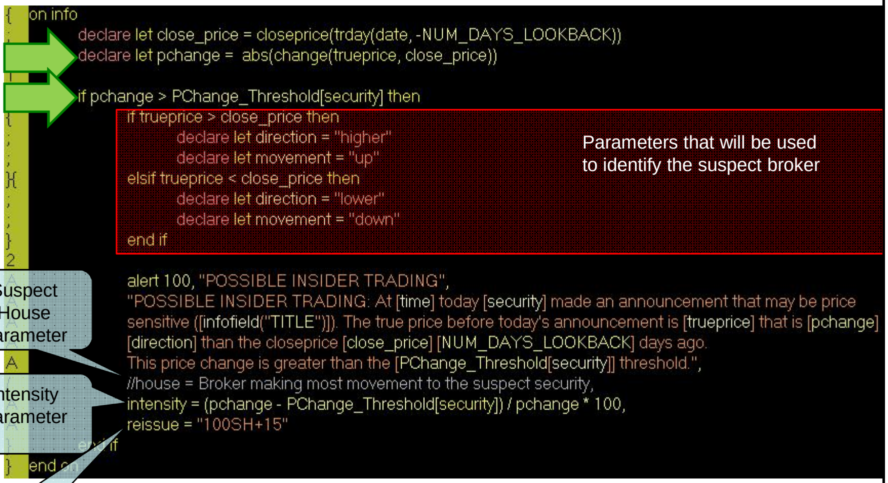
Loop over all securities



```
at start
        per security
                for declare let idate = trday(date, -1); idate >= date - PCHANGE_BENPERIOD; idate -= 1 do
                        if istrading(idate)
                                and defined(closeprice(idate))
                                and defined(closeprice(trday(idate, -NUM_DAYS_LOOKBACK)))
                        then
                                declare let close1 = closeprice(idate)
                                declare let close2 = closeprice(trday(idate, -NUM_DAYS_LOOKBACK))
                                PChange_Dist[security] <- abs(change(close1, close2))
                        end if
                end for

                declare let distribution_average = distaverage(PChange_Dist[security])
                declare let distribution_stdev = diststdev(PChange_Dist[security])
                PChange_Threshold[security] = (distribution_average + NUM_OF_STDEV * distribution_stdev) * 100%
        end per
end at
```

Loop over past 30 days

ALERTING



```
on info
        declare let close_price = closeprice(trday(date, -NUM_DAYS_LOOKBACK))
        declare let pchange =  abs(change(trueprice, close_price))

        if pchange > PChange_Threshold[security] then
                if trueprice > close_price then
                        declare let direction = "higher"
                        declare let movement = "up"
                elsif trueprice < close_price then
                        declare let direction = "lower"
                        declare let movement = "down"
                end if

                alert 100, "POSSIBLE INSIDER TRADING",
                "POSSIBLE INSIDER TRADING: At [time] today [security] made an announcement that may be price
                sensitive ([infofield("TITLE")]). The true price before today's announcement is [trueprice] that is [pchange]
                [direction] than the closeprice [close_price] [NUM_DAYS_LOOKBACK] days ago.
                This price change is greater than the [PChange_Threshold[security]] threshold.",
                //house = Broker making most movement to the suspect security,
                intensity = (pchange - PChange_Threshold[security]) / pchange * 100,
                reissue = "100SH+15"
        end if
end on
```

Parameters that will be used to identify the suspect broker

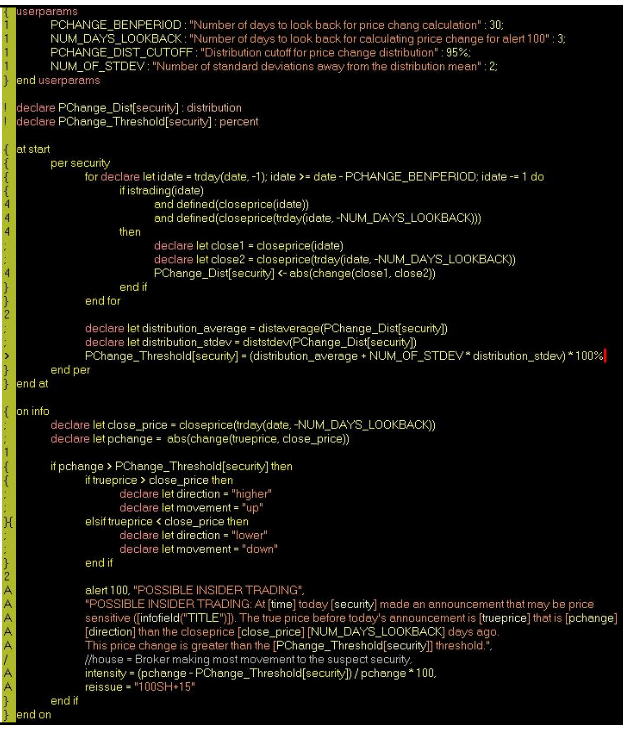
Suspect House Parameter

Intensity Parameter

Reissue Parameter

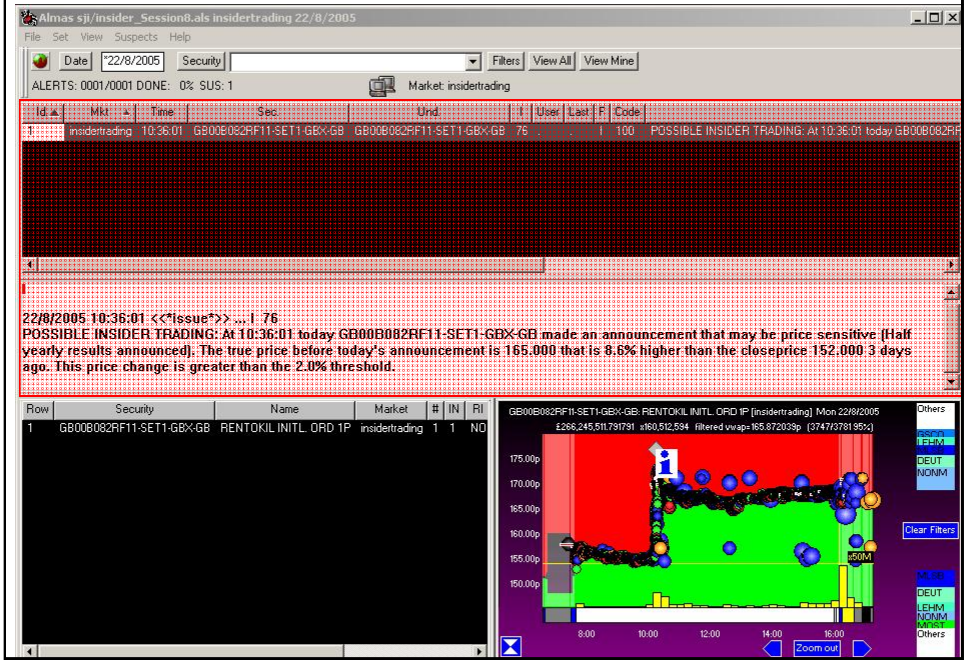
Exercise 1

Please run this Sample Code in ALDIT for 22/08/2008.



```
userparams
        PCHANGE_BENPERIOD : "Number of days to look back for price chang calculation" : 30;
        NUM_DAYS_LOOKBACK : "Number of days to look back for calculating price change for alert 100" : 3;
        PCHANGE_DIST_CUTOFF : "Distribution cutoff for price change distribution" : 95%;
        NUM_OF_STDEV : "Number of standard deviations away from the distribution mean" : 2;
end userparams

declare PChange_Dist[security] : distribution
declare PChange_Threshold[security] : percent

at start
        per security
                for declare let idate = trday(date, -1); idate >= date - PCHANGE_BENPERIOD; idate -= 1 do
                        if istrading(idate)
                                and defined(closeprice(idate))
                                and defined(closeprice(trday(idate, -NUM_DAYS_LOOKBACK)))
                        then
                                declare let close1 = closeprice(idate)
                                declare let close2 = closeprice(trday(idate, -NUM_DAYS_LOOKBACK))
                                PChange_Dist[security] <- abs(change(close1, close2))
                        end if
                end for

                declare let distribution_average = distaverage(PChange_Dist[security])
                declare let distribution_stdev = diststdev(PChange_Dist[security])
                PChange_Threshold[security] = (distribution_average + NUM_OF_STDEV * distribution_stdev) * 100%
        end per
end at

on info
        declare let close_price = closeprice(trday(date, -NUM_DAYS_LOOKBACK))
        declare let pchange =  abs(change(trueprice, close_price))

        if pchange > PChange_Threshold[security] then
                if trueprice > close_price then
                        declare let direction = "higher"
                        declare let movement = "up"
                elsif trueprice < close_price then
                        declare let direction = "lower"
                        declare let movement = "down"
                end if

                alert 100, "POSSIBLE INSIDER TRADING",
                "POSSIBLE INSIDER TRADING: At [time] today [security] made an announcement that may be price
                sensitive ([infofield("TITLE")]). The true price before today's announcement is [trueprice] that is [pchange]
                [direction] than the closeprice [close_price] [NUM_DAYS_LOOKBACK] days ago.
                This price change is greater than the [PChange_Threshold[security]] threshold.",
                //house = Broker making most movement to the suspect security,
                intensity = (pchange - PChange_Threshold[security]) / pchange * 100,
                reissue = "100SH+15"
        end if
end on
```

OUTPUT IN ALMAS



ALERTS: 0001/0001 DONE: 0% SUS: 1

Market: insidertrading

Id.	Mkt	Time	Sec.	Und.	I	User	Last	F	Code	
1	insidertrading	10:36:01	GB00B082RF11-SET1-GBX-GB	GB00B082RF11-SET1-GBX-GB	76	.	.	I	100	POSSIBLE INSIDER TRADING: At 10:36:01 today GB00B082RF

22/8/2005 10:36:01 <<*issue*>> ... I 76
POSSIBLE INSIDER TRADING: At 10:36:01 today GB00B082RF11-SET1-GBX-GB made an announcement that may be price sensitive (Half yearly results announced). The true price before today's announcement is 165.000 that is 8.6% higher than the closeprice 152.000 3 days ago. This price change is greater than the 2.0% threshold.

Row	Security	Name	Market	#	IN	RI
1	GB00B082RF11-SET1-GBX-GB	RENTOKIL INITL. ORD 1P	insidertrading	1	1	NO

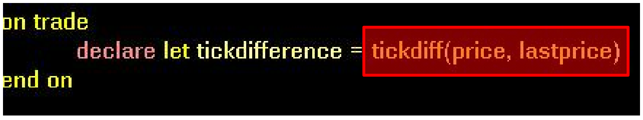
GB00B082RF11-SET1-GBX-GB: RENTOKIL INITL. ORD 1P [insidertrading] Mon 22/8/2005
£266,245,511.791791 x160,512,594 filtered vwap=165.872039p (3747/3781 95%)
175.00p
170.00p
165.00p
160.00p
155.00p
150.00p
8:00
10:00
12:00
14:00
16:00
x50M
Others
GSCO
LEHM
MLSB
DEUT
NONM
MLSB
DEUT
LEHM
NONM
MOST
Others

SUSPECT BROKER

- Find the broker that makes the most movements to the suspect security since 3 trading days ago
- Define Movements
  - **Tick: minimum price change value**
    - E.g., For BHP, every time its price changes, the minimum price change (up or down) will be $0.1.
    - Hence, for BHP, 1 tick = $0.1.
    - When BHP is previously traded at $35 and you want to sell BHP above the previous price, the minimum price you can offer is $35.10
  - **Tick value normally increases as price increases**
    - E.g. For BHP, when its price is between $20 - $30, 1 tick = $0.1
    - When its price is between $30 – $50, 1 tick = $0.2

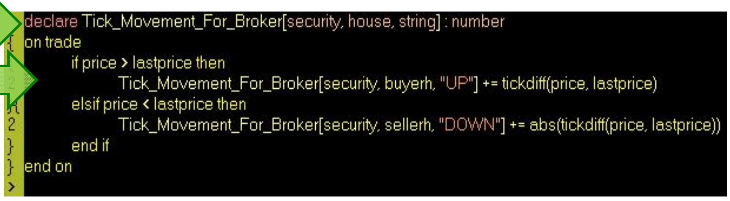
SUSPECT BROKER

- In ALICE, we use the following function to calculate tick differences between two prices:



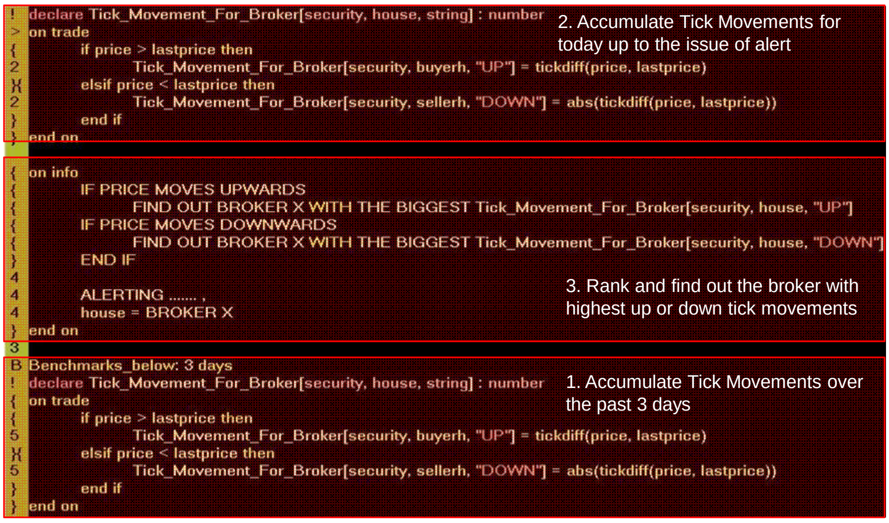
```
on trade
        declare let tickdifference = tickdiff(price, lastprice)
end on
```

To find the broker who makes the most tick movements, we need to accumulate up and down tick movements for each security/broker pair



```
declare Tick_Movement_For_Broker[security, house, string] : number
on trade
        if price > lastprice then
                Tick_Movement_For_Broker[security, buyerh, "UP"] += tickdiff(price, lastprice)
        elsif price < lastprice then
                Tick_Movement_For_Broker[security, sellerh, "DOWN"] += abs(tickdiff(price, lastprice))
        end if
end on
```

SUSPECT BROKER



```
declare Tick_Movement_For_Broker[security, house, string] : number
on trade
        if price > lastprice then
                Tick_Movement_For_Broker[security, buyerh, "UP"] = tickdiff(price, lastprice)
        elsif price < lastprice then
                Tick_Movement_For_Broker[security, sellerh, "DOWN"] = abs(tickdiff(price, lastprice))
        end if
end on
```

2. Accumulate Tick Movements for today up to the issue of alert

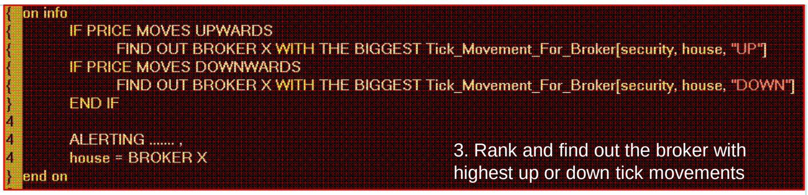
```
on info
        IF PRICE MOVES UPWARDS
                FIND OUT BROKER X WITH THE BIGGEST Tick_Movement_For_Broker[security, house, "UP"]
        IF PRICE MOVES DOWNWARDS
                FIND OUT BROKER X WITH THE BIGGEST Tick_Movement_For_Broker[security, house, "DOWN"]
        END IF

        ALERTING ....... ,
        house = BROKER X
end on
```

3. Rank and find out the broker with highest up or down tick movements

```
Benchmarks_below: 3 days
declare Tick_Movement_For_Broker[security, house, string] : number
on trade
        if price > lastprice then
                Tick_Movement_For_Broker[security, buyerh, "UP"] = tickdiff(price, lastprice)
        elsif price < lastprice then
                Tick_Movement_For_Broker[security, sellerh, "DOWN"] = abs(tickdiff(price, lastprice))
        end if
end on
```

1. Accumulate Tick Movements over the past 3 days

SUSPECT BROKER



```
on info
        IF PRICE MOVES UPWARDS
                FIND OUT BROKER X WITH THE BIGGEST Tick_Movement_For_Broker[security, house, "UP"]
        IF PRICE MOVES DOWNWARDS
                FIND OUT BROKER X WITH THE BIGGEST Tick_Movement_For_Broker[security, house, "DOWN"]
        END IF

        ALERTING ....... ,
        house = BROKER X
end on
```

3. Rank and find out the broker with highest up or down tick movements

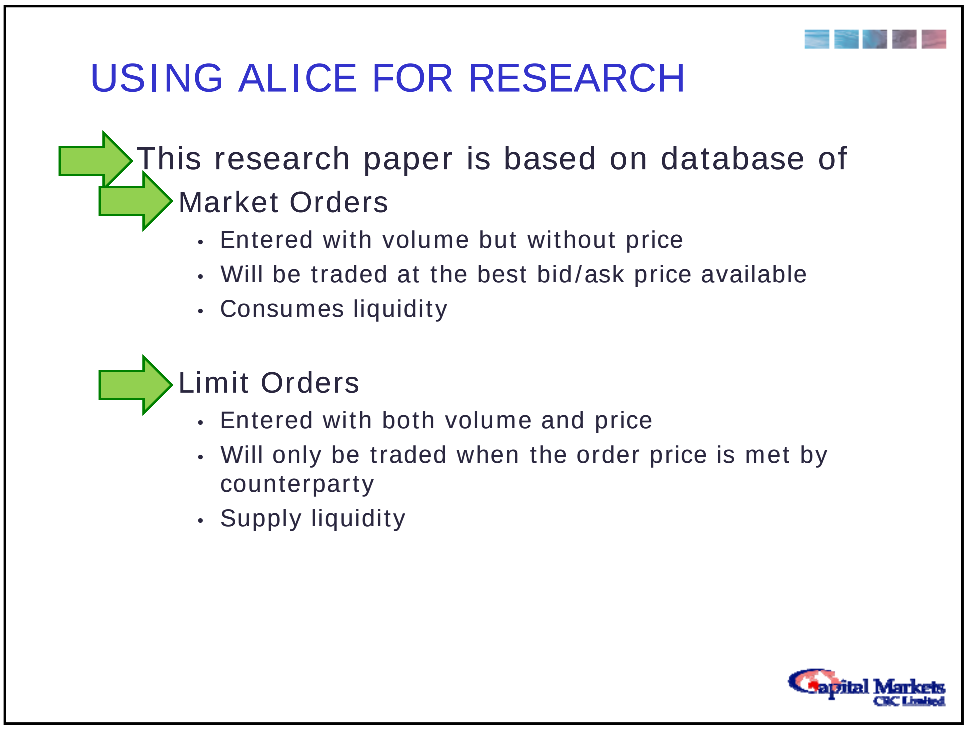
- How to rank and find out the broker with the highest up or down tick movements?
  - Similar to Activity 2 from the Session 6 Task
  - Create arrays to hold any broker seen for each security traded
  - Use FOR loops to compare tick movements for each security/broker pair stored
  - Find out the broker with the highest tick movements
- Try to understand the above logics as this will be the Activity for this Session

Capital Markets CRC Limited

USING ALICE FOR RESEARCH

- We have demonstrated how to design and create an Insider Trading Alert
- Now let's have a look how ALICE can be helpful to create a database for your academic research
- *Trading Activity and Liquidity Supply in a Pure Limit Order Book Market by Grammig et.al 2004*

Capital Markets CRC Limited



USING ALICE FOR RESEARCH

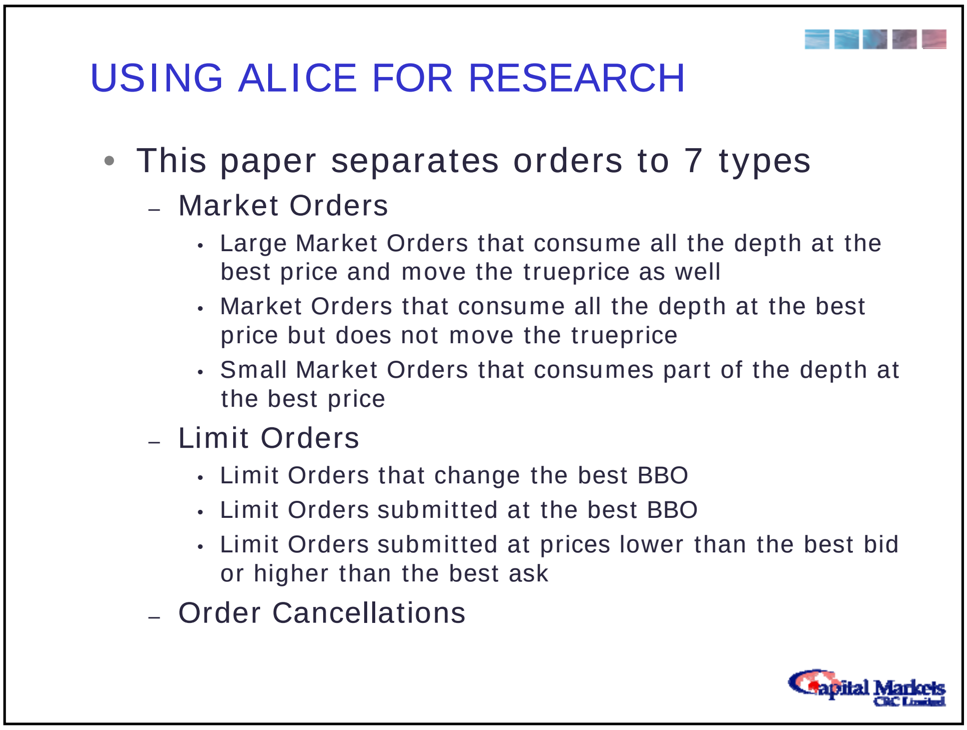
This research paper is based on database of

Market Orders

- Entered with volume but without price
- Will be traded at the best bid/ask price available
- Consumes liquidity

Limit Orders

- Entered with both volume and price
- Will only be traded when the order price is met by counterparty
- Supply liquidity



USING ALICE FOR RESEARCH

- This paper separates orders to 7 types
  - Market Orders
    - Large Market Orders that consume all the depth at the best price and move the trueprice as well
    - Market Orders that consume all the depth at the best price but does not move the trueprice
    - Small Market Orders that consumes part of the depth at the best price
  - Limit Orders
    - Limit Orders that change the best BBO
    - Limit Orders submitted at the best BBO
    - Limit Orders submitted at prices lower than the best bid or higher than the best ask
  - Order Cancellations

USING ALICE FOR RESEARCH

- We will learn

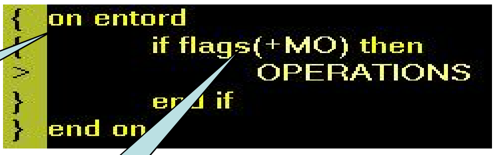
1) How to identify each of those 7 order types in ALICE
2) How to print details of orders from these 7 types into csv files
3) How to read in order detail csv files created in 2) and produce summary databases based on those csv files

Capital Markets CRC Limited

MARKET ORDERS

- In ALICE, we use flag MO to mark market orders
- The following function will return true if an order is a market order:

flags(+MO) = true for market order



```
on entord
        if flags(+MO) then
                OPERATIONS
        end if
end on
```

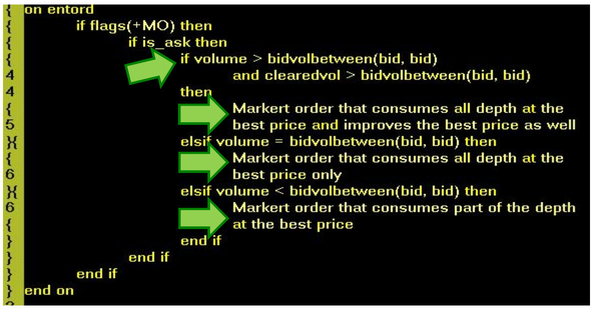
When Clause:
on entord

Check if it's a
market order

Capital Markets CRC Limited

MARKET ORDERS

- How to check whether a market order will consume all the depth at the best price?
- ALICE function:
  - bidvolbetween(price, price)
  - askvolbetween(price, price)
  - clearedvol
- For market order that is an ask



```
on entord
        if flags(+MO) then
                if is_ask then
                        if volume > bidvolbetween(bid, bid)
                                and clearedvol > bidvolbetween(bid, bid)
                        then
                                Markert order that consumes all depth at the
                                best price and improves the best price as well
                        elsif volume = bidvolbetween(bid, bid) then
                                Markert order that consumes all depth at the
                                best price only
                        elsif volume < bidvolbetween(bid, bid) then
                                Markert order that consumes part of the depth
                                at the best price
                        end if
                end if
        end if
end on
```

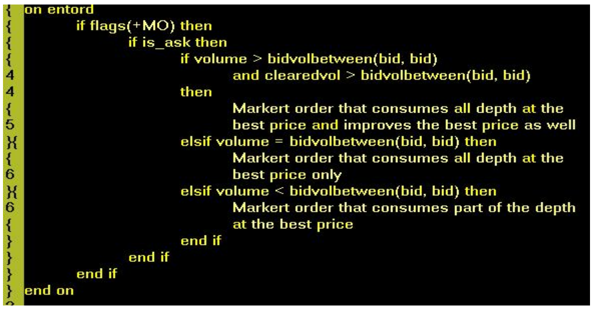
MARKET ORDERS

- How to check whether a market order will consume all the depth at the best price?
- ALICE function:
  - bidvolbetween(price, price)
  - askvolbetween(price, price)
  - clearedvol
- For market order that is an ask

Bid 1 $10 x500
Bid 2 $10 x300

Enter Market Order
Ask x 600

Clearedvol = x600



```
on entord
        if flags(+MO) then
                if is_ask then
                        if volume > bidvolbetween(bid, bid)
                                and clearedvol > bidvolbetween(bid, bid)
                        then
                                Markert order that consumes all depth at the
                                best price and improves the best price as well
                        elsif volume = bidvolbetween(bid, bid) then
                                Markert order that consumes all depth at the
                                best price only
                        elsif volume < bidvolbetween(bid, bid) then
                                Markert order that consumes part of the depth
                                at the best price
                        end if
                end if
        end if
end on
```

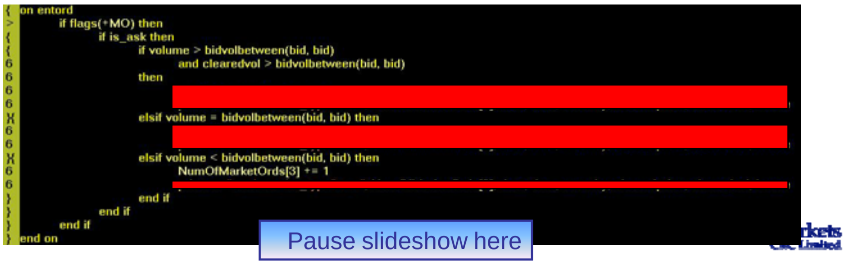
Exercise 2

- Now we need to print details of those market orders into csv files: number index, date, time, security, cleared price, cleared volume, broker
- Based on the following screenshot, fill in the parts highlighted in red with ALICE codes that will print the above market order details into three csv files with following names:

MarketOrder_Type1.csv
MarketOrder_Type2.csv
MarketOrder_Type3.csv

Assume we have declared the following variable

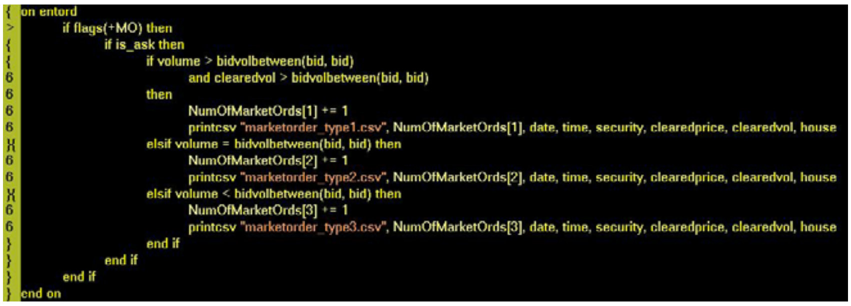
declare NumOfMarketOrds[number] : number



```
on entord
        if flags(+MO) then
                if is_ask then
                        if volume > bidvolbetween(bid, bid)
                                and clearedvol > bidvolbetween(bid, bid)
                        then


                        elsif volume = bidvolbetween(bid, bid) then


                        elsif volume < bidvolbetween(bid, bid) then
                                NumOfMarketOrds[3] += 1

                        end if
                end if
        end if
end on
```

Pause slideshow here

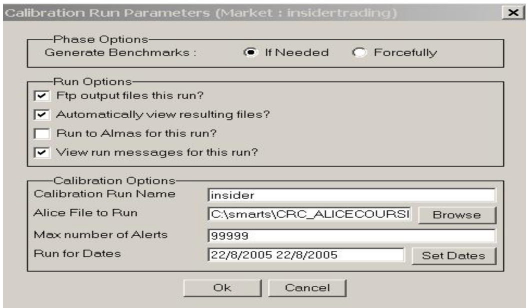
Exercise 2: Sample Solution



```
on entord
        if flags(+MO) then
                if is_ask then
                        if volume > bidvolbetween(bid, bid)
                                and clearedvol > bidvolbetween(bid, bid)
                        then
                                NumOfMarketOrds[1] += 1
                                printcsv "marketorder_type1.csv", NumOfMarketOrds[1], date, time, security, clearedprice, clearedvol, house
                        elsif volume = bidvolbetween(bid, bid) then
                                NumOfMarketOrds[2] += 1
                                printcsv "marketorder_type2.csv", NumOfMarketOrds[2], date, time, security, clearedprice, clearedvol, house
                        elsif volume < bidvolbetween(bid, bid) then
                                NumOfMarketOrds[3] += 1
                                printcsv "marketorder_type3.csv", NumOfMarketOrds[3], date, time, security, clearedprice, clearedvol, house
                        end if
                end if
        end if
end on
```

Exercise 2: Let run the solution

- Please make sure you select the following options from the Calibration Run Parameters Window:
  - Ftp Output files this Run
  - Automatically view resulting files



Calibration Run Parameters (Market : insidertrading)

Phase Options
Generate Benchmarks : ● If Needed ○ Forcefully

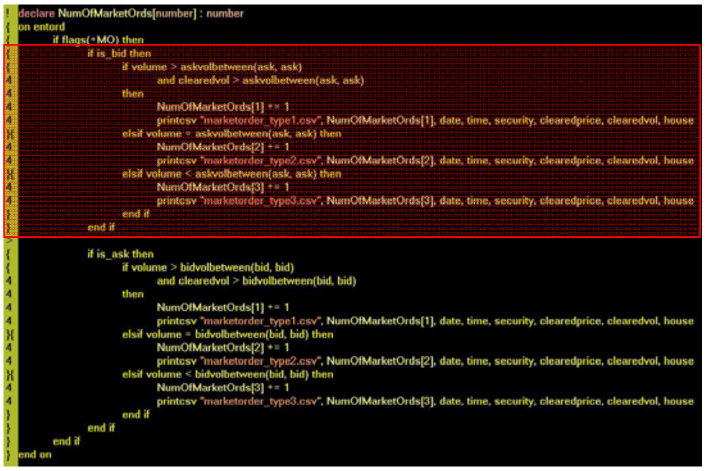
Run Options
☑ Ftp output files this run?
☑ Automatically view resulting files?
□ Run to Almas for this run?
☑ View run messages for this run?

Calibration Options
Calibration Run Name: insider
Alice File to Run: C:\smarts\CRC_ALICECOURSI
Max number of Alerts: 99999
Run for Dates: 22/8/2005 22/8/2005

- When the calibration run is finished, csv files will automatically open

MARKET ORDER

- Now let's apply the same logic to market orders that are bids

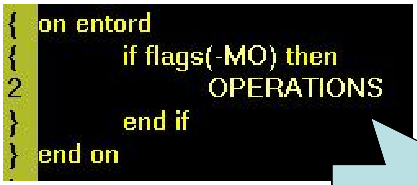


```
declare NumOfMarketOrds[number] : number
on entord
        if flags(+MO) then
                if is_bid then
                        if volume > askvolbetween(ask, ask)
                                and clearedvol > askvolbetween(ask, ask)
                        then
                                NumOfMarketOrds[1] += 1
                                printcsv "marketorder_type1.csv", NumOfMarketOrds[1], date, time, security, clearedprice, clearedvol, house
                        elsif volume = askvolbetween(ask, ask) then
                                NumOfMarketOrds[2] += 1
                                printcsv "marketorder_type2.csv", NumOfMarketOrds[2], date, time, security, clearedprice, clearedvol, house
                        elsif volume < askvolbetween(ask, ask) then
                                NumOfMarketOrds[3] += 1
                                printcsv "marketorder_type3.csv", NumOfMarketOrds[3], date, time, security, clearedprice, clearedvol, house
                        end if
                end if

                if is_ask then
                        if volume > bidvolbetween(bid, bid)
                                and clearedvol > bidvolbetween(bid, bid)
                        then
                                NumOfMarketOrds[1] += 1
                                printcsv "marketorder_type1.csv", NumOfMarketOrds[1], date, time, security, clearedprice, clearedvol, house
                        elsif volume = bidvolbetween(bid, bid) then
                                NumOfMarketOrds[2] += 1
                                printcsv "marketorder_type2.csv", NumOfMarketOrds[2], date, time, security, clearedprice, clearedvol, house
                        elsif volume < bidvolbetween(bid, bid) then
                                NumOfMarketOrds[3] += 1
                                printcsv "marketorder_type3.csv", NumOfMarketOrds[3], date, time, security, clearedprice, clearedvol, house
                        end if
                end if
        end if
end on
```

LIMIT ORDERS

- We use the MO flag to identify market orders, what about limit orders?
- Yes, orders without the MO flag are limit orders:

flags(-MO) = true for limit orders



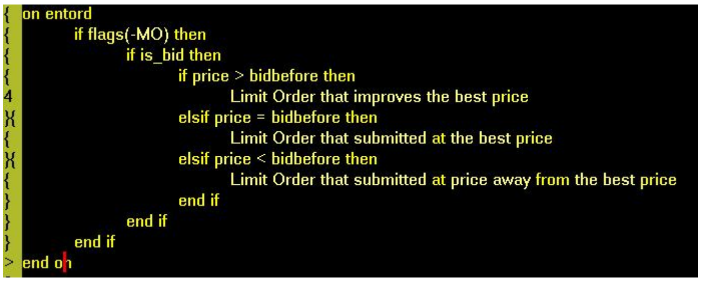
```
on entord
    if flags(-MO) then
        OPERATIONS
    end if
end on
```

You could also use:

If NOT flags(+MO)

LIMIT ORDERS

- How to check whether a limit order changes the best price?
- ALICE Function
  - Bidbefore
  - Askbefore
- For Limit Orders that are bids

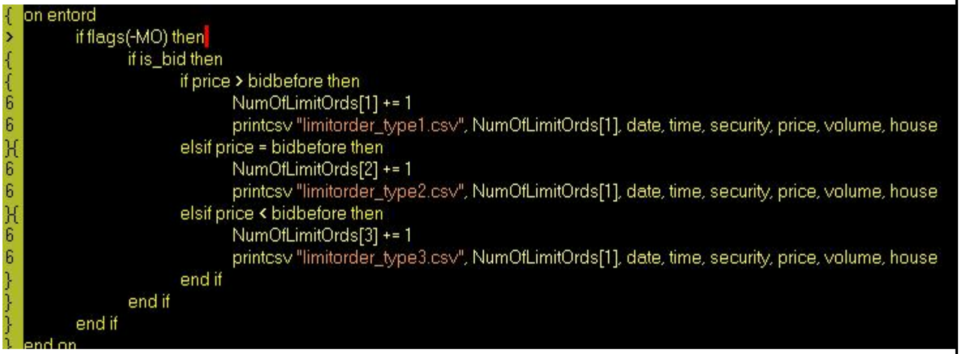


```
on entord
    if flags(-MO) then
        if is_bid then
            if price > bidbefore then
                Limit Order that improves the best price
            elsif price = bidbefore then
                Limit Order that submitted at the best price
            elsif price < bidbefore then
                Limit Order that submitted at price away from the best price
            end if
        end if
    end if
end on
```

LIMIT ORDERS

- Now let's print the following details of limit orders into csv files:

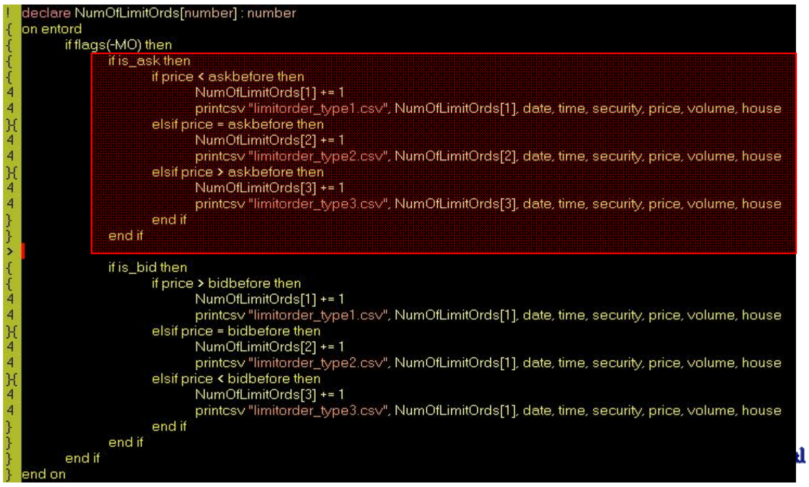
  number index, date, time, security, price, volume, broker



```
on entord
        if flags(-MO) then
                if is_bid then
                        if price > bidbefore then
                                NumOfLimitOrds[1] += 1
                                printcsv "limitorder_type1.csv", NumOfLimitOrds[1], date, time, security, price, volume, house
                        elsif price = bidbefore then
                                NumOfLimitOrds[2] += 1
                                printcsv "limitorder_type2.csv", NumOfLimitOrds[1], date, time, security, price, volume, house
                        elsif price < bidbefore then
                                NumOfLimitOrds[3] += 1
                                printcsv "limitorder_type3.csv", NumOfLimitOrds[1], date, time, security, price, volume, house
                        end if
                end if
        end if
end on
```

EXERCISE 3

- Now let's print the details of limit orders that are asks to csv files
- Here is the sample solution



```
declare NumOfLimitOrds[number] : number
on entord
        if flags(-MO) then
                if is_ask then
                        if price < askbefore then
                                NumOfLimitOrds[1] += 1
                                printcsv "limitorder_type1.csv", NumOfLimitOrds[1], date, time, security, price, volume, house
                        elsif price = askbefore then
                                NumOfLimitOrds[2] += 1
                                printcsv "limitorder_type2.csv", NumOfLimitOrds[2], date, time, security, price, volume, house
                        elsif price > askbefore then
                                NumOfLimitOrds[3] += 1
                                printcsv "limitorder_type3.csv", NumOfLimitOrds[3], date, time, security, price, volume, house
                        end if
                end if

                if is_bid then
                        if price > bidbefore then
                                NumOfLimitOrds[1] += 1
                                printcsv "limitorder_type1.csv", NumOfLimitOrds[1], date, time, security, price, volume, house
                        elsif price = bidbefore then
                                NumOfLimitOrds[2] += 1
                                printcsv "limitorder_type2.csv", NumOfLimitOrds[1], date, time, security, price, volume, house
                        elsif price < bidbefore then
                                NumOfLimitOrds[3] += 1
                                printcsv "limitorder_type3.csv", NumOfLimitOrds[1], date, time, security, price, volume, house
                        end if
                end if
        end if
end on
```

READ IN CSV FILES

- Finally, let's look at how to read in those csv files we just created for market orders and limit orders
- ALICE Function: read
  - Syntax: read "full file path", indx1, … indx t, array 1, … array N

E.g.,
declare Array1[number] : price
read "file.csv", number, Array1

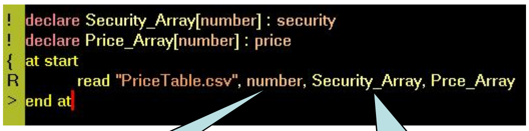
If you only specify the csv file name here, ALICE will use the default path

- Let see an example:

Suppose we have the following csv file

1	BHP	40.15
2	CVC	20.25
3	CIA	30.35
4	RIO	40.25
5	WES	12.35
6	VOD	40.95

To use ALICE to read in this table, we need to do:



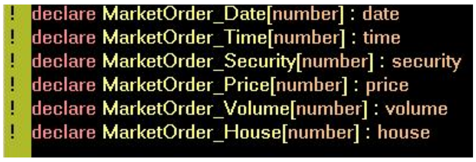
```
! declare Security_Array[number] : security
! declare Price_Array[number] : price
{ at start
R         read "PriceTable.csv", number, Security_Array, Prce_Array
> end at
```

In this case, we only have one index which is the first column from the csv and it's a number

The name of arrays are placed after the index. Remember, only the name of the array is needed

READ IN MarketOrder_Type3.csv

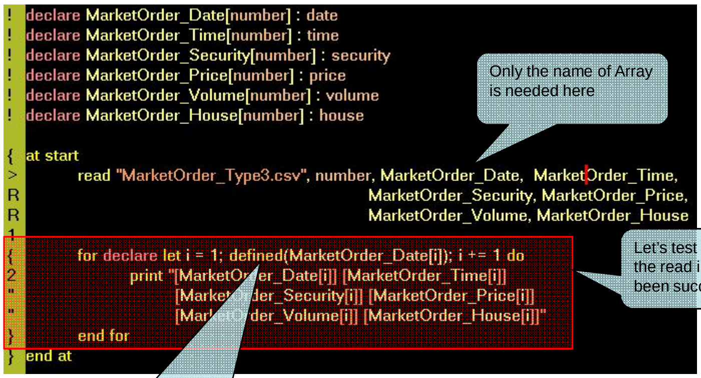
- Now let's try to read in MarketOrder_Type3.csv
- MarketOrder_Type3.csv contains
  - Column 1 Number Index
  - Column 2 Date
  - Column 3 Time
  - Column 4 Security
  - Column 5 Price
  - Column 6 Volume
  - Column 7 Broker
- Let's use column 1 as the index for arrays and declare 6 arrays for date, time, security, price, volume and broker



```
! declare MarketOrder_Date[number] : date
! declare MarketOrder_Time[number] : time
! declare MarketOrder_Security[number] : security
! declare MarketOrder_Price[number] : price
! declare MarketOrder_Volume[number] : volume
! declare MarketOrder_House[number] : house
```

READ IN MarketOrder_Type3.csv

- Csv file path: In this case, since LimitOrder_Type1.csv was stored at the default path, we only need to specify the file name for the read function.



```
declare MarketOrder_Date[number] : date
declare MarketOrder_Time[number] : time
declare MarketOrder_Security[number] : security
declare MarketOrder_Price[number] : price
declare MarketOrder_Volume[number] : volume
declare MarketOrder_House[number] : house

at start
        read "MarketOrder_Type3.csv", number, MarketOrder_Date,  MarketOrder_Time,
                                             MarketOrder_Security, MarketOrder_Price,
                                             MarketOrder_Volume, MarketOrder_House

        for declare let i = 1; defined(MarketOrder_Date[i]); i += 1 do
                print "[MarketOrder_Date[i]] [MarketOrder_Time[i]]
                       [MarketOrder_Security[i]] [MarketOrder_Price[i]]
                       [MarketOrder_Volume[i]] [MarketOrder_House[i]]"
        end for
end at
```

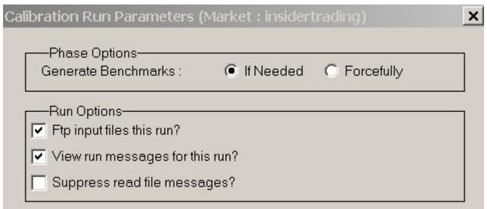
Only the name of Array is needed here

Let's test whether the read in has been successful

In this for loop, i is the row number from the csv file. The for loop will continue until it reaches the last row

READ IN MarketOrder_Type3.csv

- Now let's run this ALICE script
  - Make sure you select the Ftp Input Files this Run option

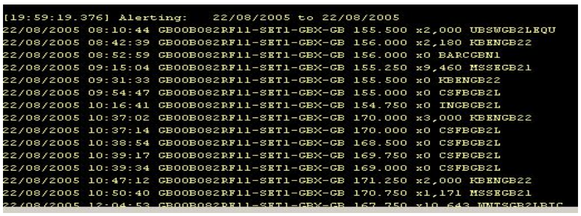


Calibration Run Parameters (Market : insidertrading)

Phase Options
Generate Benchmarks : ● If Needed ○ Forcefully

Run Options
☑ Ftp input files this run?
☑ View run messages for this run?
□ Suppress read file messages?

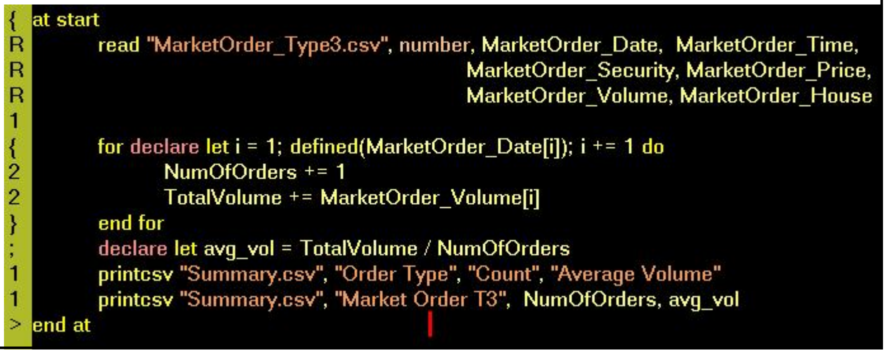
- You should observe the following outputs if you execute this previous ALICE Script



```
[19:59:19.376] Alerting:   22/08/2005 to 22/08/2005
22/08/2005 08:10:44 GB00B082RF11-SET1-GBX-GB 155.500 x2,000 UBSWGB2LEQU
22/08/2005 08:42:39 GB00B082RF11-SET1-GBX-GB 156.000 x2,180 KBENGB22
22/08/2005 08:52:59 GB00B082RF11-SET1-GBX-GB 156.000 x0 BARCGBN1
22/08/2005 09:15:04 GB00B082RF11-SET1-GBX-GB 155.250 x9,460 MSSEGB21
22/08/2005 09:31:33 GB00B082RF11-SET1-GBX-GB 155.500 x0 KBENGB22
22/08/2005 09:54:47 GB00B082RF11-SET1-GBX-GB 155.000 x0 CSFBGB2L
22/08/2005 10:16:41 GB00B082RF11-SET1-GBX-GB 154.750 x0 INGBGB2L
22/08/2005 10:37:02 GB00B082RF11-SET1-GBX-GB 170.000 x3,000 KBENGB22
22/08/2005 10:37:14 GB00B082RF11-SET1-GBX-GB 170.000 x0 CSFBGB2L
22/08/2005 10:38:54 GB00B082RF11-SET1-GBX-GB 168.500 x0 CSFBGB2L
22/08/2005 10:39:17 GB00B082RF11-SET1-GBX-GB 169.750 x0 CSFBGB2L
22/08/2005 10:39:34 GB00B082RF11-SET1-GBX-GB 169.000 x0 CSFBGB2L
22/08/2005 10:47:12 GB00B082RF11-SET1-GBX-GB 171.250 x2,000 KBENGB22
22/08/2005 10:50:40 GB00B082RF11-SET1-GBX-GB 170.750 x1,171 MSSEGB21
22/08/2005 12:04:53 GB00B082RF11-SET1-GBX-GB 167.750 x10,643 NWTSGB2LBIC
```

Exercise 4

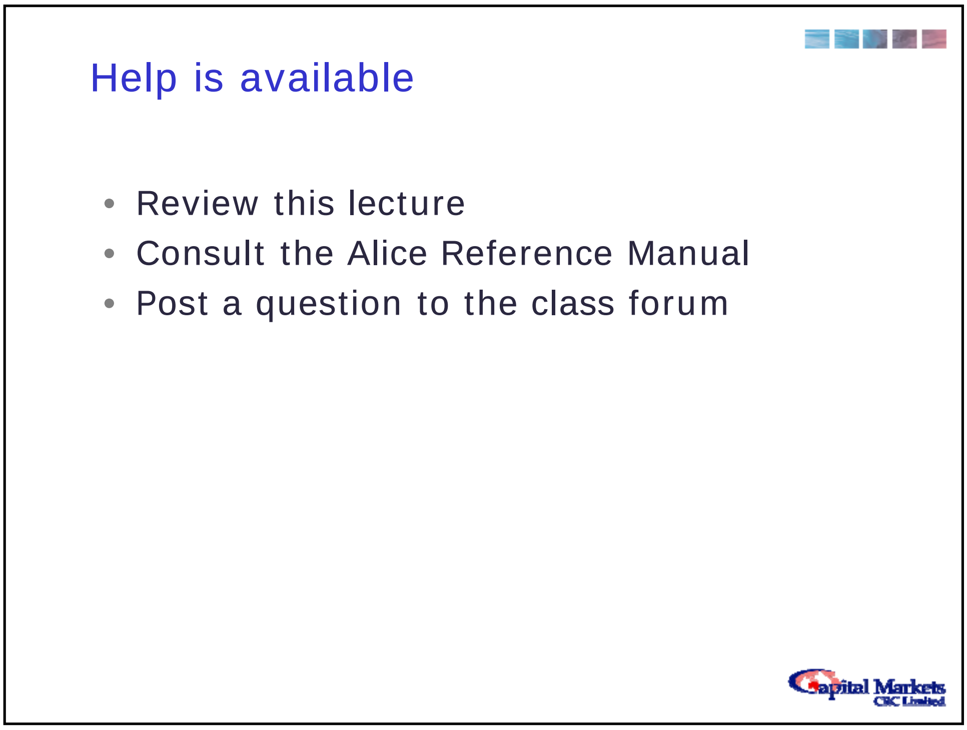
- Now, let create a new csv file that contains
  - The number of orders for each type
  - The average volume of orders for each order type
- You only need to print out the above statistics for Market Order Type 3 now
- Here is the sample solution



```
at start
        read "MarketOrder_Type3.csv", number, MarketOrder_Date,  MarketOrder_Time,
                                             MarketOrder_Security, MarketOrder_Price,
                                             MarketOrder_Volume, MarketOrder_House

        for declare let i = 1; defined(MarketOrder_Date[i]); i += 1 do
                NumOfOrders += 1
                TotalVolume += MarketOrder_Volume[i]
        end for
        declare let avg_vol = TotalVolume / NumOfOrders
        printcsv "Summary.csv", "Order Type", "Count", "Average Volume"
        printcsv "Summary.csv", "Market Order T3",  NumOfOrders, avg_vol
end at
```

Key terms and concepts

- Insider Trading Alert
- DAYTOT benchmarking
- On info
- Tick & tickdiff
- Suspect broker
- Market Order and Limit Order
- bidbefore & askbefore
- Cleared Volume



Help is available

- Review this lecture
- Consult the Alice Reference Manual
- Post a question to the class forum