```
strategy("Iron Strategy", overlay=true, initial_capital = 10000,
default_qty_value = 100, pyramiding = 1, slippage = 0, currency=currency.USDT,
default_qty_type=strategy.percent_of_equity)

// Time condition
time_cond = time > timestamp("UTC+1", 2018, 01,01) and time <
timestamp("UTC+1",3000,01,01)

//Get RSI user Input
Lookback    = input.int(title      = 'Lookback', group = 'RSI', defval = 1000)
rsilength   = input.int(title      = 'RSI Length', group = 'RSI', defval = 30)
rsiOB       = input.float(title    = 'RSI Overbought', group = 'RSI', defval =
80.0)
rsiOS       = input.float(title    = 'RSI Oversold', group = 'RSI', defval =
20.0)

// Get CCI User Input
Lookbackcci    = input.int(title = 'Lookbackcci', group = 'CCI', defval = 1000)
cciLength      = input.int(title      = 'cci length', group = 'CCI', defval =
30)
cciOB          = input.float(title    = 'cci overbought', group = 'CCI', defval
= 78)
cciOS          = input.float(title    = 'cci oversold', group = 'CCI', defval =
68)

// Get RSI Value
rsi      = ta.rsi(close,rsilength)
rsishort = rsi > rsiOB
rsilong  = rsi < rsiOS

// Get CCI Value
cci       = ta.cci(close,cciLength)
ccishort  = cci > cciOB
ccilong   = cci < cciOS

// Get MACD User Input
fast_length   = input(title = 'Fast Length', group = 'MACD', defval = 12)
slow_length   = input(title    = 'Slow Length',group = 'MACD', defval = 26)
src           = input(title = 'Source', group = 'MACD', defval = close)
signal_length = input.int(title = 'Signal Smoothing', group ='MACD', minval=1,
maxval=50, defval=9)
```

```
sma_source    = input.string(title = 'Occilator MA Type', group ='MACD', defval
='EMA', options =['SMA', 'EMA'])
sma_signal    = input.string(title = 'Signal Line MA Type', group = 'MACD',
defval = 'EMA', options=['SMA', 'EMA'])

// Calculate MACD
fast_ma = sma_source == "SMA" ? ta.sma(src, fast_length) : ta.ema(src,
fast_length)
slow_ma = sma_source == "SMA" ? ta.sma(src, slow_length) : ta.ema(src,
slow_length)
macd    = fast_ma - slow_ma
signal  = sma_signal == "SMA" ? ta.sma(macd, signal_length) : ta.ema(macd,
signal_length)

//  Input Controls For EMA
lengthSEMA = input.int(title = 'Short EMA Length', defval = 13, minval = 1,
maxval = 100, step = 1 )
lengthLEMA = input.int(title = 'Long EMA Length', defval = 34, minval = 1,
maxval = 200, step = 1 )
SEMA = ta.ema(close,lengthSEMA)
LEMA = ta.ema(close,lengthLEMA)
EMA_CROSSOVER = ta.crossover(SEMA,LEMA)
EMA_CROSSUNDER = ta.crossunder(SEMA,LEMA)

// Check For Zero-Point Crosses
Crossup   = ta.crossover(macd,0) and ta.crossover(macd,signal) and
EMA_CROSSOVER
Crossdown = ta.crossunder(macd,0) and ta.crossunder(macd,signal) and
EMA_CROSSUNDER

// Get Supertrend Indicator?

//Long + Short Condition
Longcondition  = rsilong or ccilong or Crossup
Shortcondition = rsishort or ccishort or Crossdown

// Signal
if Longcondition and time_cond
    strategy.entry("Long",strategy.long)

if Shortcondition and time_cond
    strategy.entry("Short",strategy.short)
```

```
// Draw to chart

plotshape(Longcondition,style=shape.triangleup,color=color.green, location=
location.belowbar)
plotshape(Shortcondition,style=shape.triangledown,color=color.red, location=
location.abovebar)

//Simplified Cobra Table
import EliCobra/CobraMetrics/3 as table
disp_indd = input.string("Equity" , title = "Display Table" ,tooltip =
'Strategy", "Equity", "Open Profit", "Gross Profit", "Net Profit',
options=["Strategy", "Equity", "Open Profit", "Gross Profit", "Net Profit"])
table.cobraTable()
plot(table.curve(disp_indd))
```