```
// This source code is subject to the terms of the Mozilla Public License 2.0 at https://mozilla.org/MPL/2.0/ // © DeuceDavis

//@version=5 indicator("CPR Option Sale Strategy", overlay=true)

tradeSource = input.source(close, title='Source For Trigger')

autoTF = input.string('Auto', title="Select Higher Timeframe", options=['Auto','Manual'], group='Pivot Settings') manPivotTF = input.timeframe('D', title='If Manual, Time Frame for Pivots', group='Pivot Settings', options=['60','90', '120','D', '2D', '3D', '4D', 'W', '2W', '3W', '4W', 'M','2M', '3M', '6M'])

//nearStrikePremiumRisk = input.int(1, "Premium Risk Level[1-5 (Higher is riskier)]", maxval=5, step=1, // group="Near Strike Calculation Method") nearStrikeMethod = input.string('Traditional Pivot S1/R1', title="Near Strike Calculation Source", options=['Traditional Pivot S1/R1', 'Camarilla Pivot R1/S1', 'Camarilla Pivot R2/S2', 'Manual Offset From Central Pivot Point'], group="Near Strike Calculation Method")

nearStrikeManualOffset = input.float(0, "Manual Offset Value", group="Near Strike Calculation Method")

spreadMethod = input.string('Auto', title='Determine Spread', options=['Auto', 'Manual'], group='Spread Method') manSpreadValue = input.float(5.0, title="Spread Width, if Manual", group='Spread Method')

showPriorDayHiLo = input.bool(false, title='Show Prior Day High and Low', group='Visual') showCurrentWeekHiLo= input.bool(false, title='Show Current Week Highs and Lows', group='Visual')

priorDayHigh = request.security(syminfo.tickerid, 'D', high[1], lookahead=barmerge.lookahead_on) priorDayLow = request.security(syminfo.tickerid, 'D', low[1], lookahead=barmerge.lookahead_on) currentWeekHigh = request.security(syminfo.tickerid, 'W', high, lookahead=barmerge.lookahead_on) currentWeekLow = request.security(syminfo.tickerid, 'W', low, lookahead=barmerge.lookahead_on)

pdHigh = plot(showPriorDayHiLo ? priorDayHigh : na, title="Prior Day High", color=color.new(color.red, 0)) pdLow = plot(showPriorDayHiLo ? priorDayLow : na, title="Prior Day Low", color=color.new(color.green, 0)) cwHigh = plot(showCurrentWeekHiLo ? currentWeekHigh : na, title="Current Week High", color=color.new(color.red, 0)) cwLow = plot(showCurrentWeekHiLo ? currentWeekLow : na, title="Current Week Low", color=color.new(color.green, 0))
```

```
//pivotPeriodShift =input.int(defval=1, title='Lookback Period for Pivot', group='Primary Settings') pivotPeriodShift = 1

sendStandardAlert = input.bool(false, title='Enable Standard Alert', group='Alerts') sendDiscordWebhookAlert = input.bool(false, title='Enable Discord Webhook Alert', group='Alerts') sendPremiumZoneChangeAlert = input.bool(false, title='Enable Price Movement Into Different Zone', group='Alerts')

plotTS1R1 = input.bool(false, title='Plot Traditional Pivot R1/S1', group='Traditional Pivot Plots') plotTS2R2 = input.bool(false, title='Plot Traditional Pivot R2/S2', group='Traditional Pivot Plots') plotTS3R3 = input.bool(false, title='Plot Traditional Pivot R3/S3', group='Traditional Pivot Plots')

plotCS1R1 = input.bool(false, title='Plot Camarilla Pivot R1/S1', group='Camarilla Pivot Plots') plotCS2R2 = input.bool(false, title='Plot Camarilla Pivot R2/S2', group='Camarilla Pivot Plots') plotCS3R3 = input.bool(false, title='Plot Camarilla Pivot R3/S3', group='Camarilla Pivot Plots') plotCS4R4 = input.bool(false, title='Plot Camarilla Pivot R4/S4', group='Camarilla Pivot Plots') plotCS5R5 = input.bool(false, title='Plot Camarilla Pivot R5/S5', group='Camarilla Pivot Plots')

tradeOnMon = input.bool(true, title='Mondays', group='Trade Days (Good for Dailies)') tradeOnTue = input.bool(true, title='Tuesdays', group='Trade Days (Good for Dailies)') tradeOnWed = input.bool(true, title='Wednesdays', group='Trade Days (Good for Dailies)') tradeOnThu = input.bool(true, title='Thursdays', group='Trade Days (Good for Dailies)') tradeOnFri = input.bool(true, title='Fridays', group='Trade Days (Good for Dailies)')

showHistoricalTradeLabel = input.bool(false, title='Show Historical Strike Value Labels', group='Visual') showFuture = input.bool(false, title='Extend Developing Levels Into Future', group='Visual') showHistoricalResults = input.bool(true, title='Show Historical Win/Loss Percentages', group='Visual') backtestStartDate = input.time(timestamp("1 Jan 2020"), group="Visual", title="Start Date", group="Backtest Time Period", tooltip="This start date is in the time zone of the exchange " + "where the chart's instrument trades. It doesn't use the time " + "zone of the chart or of your computer.") backtestEndDate = input.time(timestamp("1 Jan 2099"), group="Visual", title="End Date", group="Backtest Time Period", tooltip="This end date is in the time zone of the exchange " + "where the chart's instrument trades. It doesn't use the time " + "zone of the chart or of your computer.")

var tradeTheDay = true switch dayofweek == dayofweek.monday => tradeTheDay := tradeOnMon dayofweek == dayofweek.tuesday => tradeTheDay := tradeOnTue
```

```
dayofweek == dayofweek.wednesday => tradeTheDay := tradeOnWed dayofweek == dayofweek.thursday => tradeTheDay := tradeOnThu dayofweek == dayofweek.friday => tradeTheDay := tradeOnFri

//auto higher time frame HTFo = timeframe.period == '1' ? '30' : timeframe.period == '3' ? '60' : timeframe.period == '5' ? '240' : timeframe.period == '15' ? 'D' : timeframe.period == '30' ? 'D' : timeframe.period == '45' ? 'D' : timeframe.period == '60' ? 'W' : timeframe.period == '120' ? 'W' : timeframe.period == '180' ? 'W' : timeframe.period == '240' ? 'W' : timeframe.period == 'D' ? 'W' : timeframe.period == 'W' ? '4W' : 'D'

pivotTF = autoTF == 'Auto' ? HTFo : manPivotTF

devPivotTFHigh = request.security(syminfo.tickerid, pivotTF, high, lookahead=barmerge.lookahead_on) devPivotTFLow = request.security(syminfo.tickerid, pivotTF, low, lookahead=barmerge.lookahead_on) devPivotTFClose = request.security(syminfo.tickerid, pivotTF, close, lookahead=barmerge.lookahead_on) pivotTFHigh = request.security(syminfo.tickerid, pivotTF, high[pivotPeriodShift], lookahead=barmerge.lookahead_on) pivotTFLow = request.security(syminfo.tickerid, pivotTF, low[pivotPeriodShift], lookahead=barmerge.lookahead_on) pivotTFClose = request.security(syminfo.tickerid, pivotTF, close[pivotPeriodShift], lookahead=barmerge.lookahead_on) priorPivotTFHigh = request.security(syminfo.tickerid, pivotTF, high[pivotPeriodShift + 1], lookahead=barmerge.lookahead_on) priorPivotTFLow = request.security(syminfo.tickerid, pivotTF, low[pivotPeriodShift + 1], lookahead=barmerge.lookahead_on) priorPivotTFClose = request.security(syminfo.tickerid, pivotTF, close[pivotPeriodShift + 1], lookahead=barmerge.lookahead_on) pivotLastBar = request.security(syminfo.tickerid, pivotTF, barstate.islast, lookahead=barmerge.lookahead_on)

RANGE = pivotTFHigh- pivotTFLow

//CPR Calculations centralPivot = (pivotTFHigh + pivotTFLow + pivotTFClose) / 3 tempBottomPivot = (pivotTFHigh + pivotTFLow) / 2 tempTopPivot = (centralPivot - tempBottomPivot) + centralPivot

bottomPivot = (tempBottomPivot > tempTopPivot) ? tempTopPivot : tempBottomPivot topPivot = bottomPivot == tempBottomPivot ? tempTopPivot : tempBottomPivot
```

cprRange = topPivot - bottomPivot

//Traditional Pivot Point Calculations tS1 = 2 * centralPivot - pivotTFHigh tS2 = centralPivot - (pivotTFHigh - pivotTFLow) tS3 = tS1 - (pivotTFHigh - pivotTFLow) tR1 = 2 * centralPivot - pivotTFLow tR2 = centralPivot + (pivotTFHigh - pivotTFLow) tR3 = tR1 + (pivotTFHigh - pivotTFLow)

//Camarilla Pivot Point Calculations camR5 = pivotTFHigh / pivotTFLow * pivotTFClose camR4 = pivotTFClose + RANGE * 1.1 / 2 camR3 = pivotTFClose + RANGE * 1.1 / 4 camR2 = pivotTFClose + RANGE * 1.1 / 6 camR1 = pivotTFClose + RANGE * 1.1 / 12 camS1 = pivotTFClose - RANGE * 1.1 / 12 camS2 = pivotTFClose - RANGE * 1.1 / 6 camS3 = pivotTFClose - RANGE * 1.1 / 4 camS4 = pivotTFClose - RANGE * 1.1 / 2 camS5 = pivotTFClose - (camR5 - pivotTFClose)

//Developing CPR & tR1 & tS1 devCentralPivot = (devPivotTFHigh + devPivotTFLow + devPivotTFClose) / 3 devTempBottomPivot = (devPivotTFHigh + devPivotTFLow) / 2 devTempTopPivot = (devCentralPivot - devTempBottomPivot) + devCentralPivot

devBottomPivot = (devTempBottomPivot > devTempTopPivot) ? devTempTopPivot : devTempBottomPivot devTopPivot = devBottomPivot == devTempBottomPivot ? devTempTopPivot : devTempBottomPivot devTS1 = 2 * devCentralPivot - devPivotTFHigh devTR1 = 2 * devCentralPivot - devPivotTFLow

//spreadValue Floor & Ceiling spreadValueCalc(calcValue, spreadValue, callOrPutSide) => var retValue = 0.00 switch callOrPutSide == 'Call' => retValue := (math.ceil(calcValue/spreadValue)) * spreadValue callOrPutSide == 'Put' => retValue := (math.floor(calcValue/spreadValue)) * spreadValue

//Strike Values spreadValue = spreadMethod == "Auto" ? close >= 1000 ? 50.0 : close >= 500 ? 10.0 : close >= 100 ? 5.0 : close >= 50 ? 2.5 : close >= 20 ? 1 : close < 20 ? 0.5 : 0 : manSpreadValue

callTriggerValue = nearStrikeMethod == 'Traditional Pivot S1/R1' ? tS1 : nearStrikeMethod == 'Camarilla Pivot R1/S1' ? camS1 : nearStrikeMethod == 'Camarilla Pivot R2/S2' ? camS2 : centralPivot

putTriggerValue = nearStrikeMethod == 'Traditional Pivot S1/R1' ? tR1 : nearStrikeMethod == 'Camarilla Pivot R1/S1' ? camR1 : nearStrikeMethod == 'Camarilla Pivot R2/S2' ? camR2 : centralPivot

callNearStrikeSource = nearStrikeMethod == 'Traditional Pivot S1/R1' ? math.max(pivotTFHigh, tR1) : nearStrikeMethod == 'Camarilla Pivot R1/S1' ?

```
math.max(pivotTFHigh, camR1) : nearStrikeMethod == 'Camarilla Pivot R2/S2' ?
math.max(pivotTFHigh, camR2) : centralPivot + nearStrikeManualOffset

putNearStrikeSource = nearStrikeMethod == 'Traditional Pivot S1/R1' ?
math.min(pivotTFLow, tS1) : nearStrikeMethod == 'Camarilla Pivot R1/S1' ?
math.min(pivotTFLow, camS1) : nearStrikeMethod == 'Camarilla Pivot R2/S2' ?
math.min(pivotTFLow, camS2) : centralPivot - nearStrikeManualOffset

calcCallValue = close < callTriggerValue ? topPivot : math.avg(centralPivot,
callNearStrikeSource) callStrikeValue = spreadValueCalc(calcCallValue, spreadValue,
'Call') callStrikeValue2 = callStrikeValue + spreadValue calcPutValue = close >
putTriggerValue ? bottomPivot : math.avg(centralPivot, putNearStrikeSource)
putStrikeValue = spreadValueCalc(calcPutValue, spreadValue, 'Put') putStrikeValue2
= putStrikeValue - spreadValue icCallStrikeValue = spreadValueCalc(tR1,spreadValue,
'Call') icCallStrikeValue2 = icCallStrikeValue + spreadValue icPutStrikeValue =
spreadValueCalc(tS1, spreadValue, 'Put') icPutStrikeValue2 = icPutStrikeValue -
spreadValue

//Tomorrow's Strikes calcDevCallValue = close < devTS1 ? devTopPivot :
math.avg(devCentralPivot, math.max(devPivotTFHigh, devTR1)) calcDevPutValue =
close > devTR1 ? devBottomPivot : math.avg(devCentralPivot,
math.min(devPivotTFLow, devTS1)) devCallStrikeValue =
spreadValueCalc(calcDevCallValue, spreadValue, 'Call') devPutStrikeValue =
spreadValueCalc(calcDevPutValue, spreadValue, 'Put')

plot(callStrikeValue, "Current Call", style=plot.style_line,
color=color.new(color.orange, 75)) plot(putStrikeValue, "Current Put",
style=plot.style_line, color=color.new(color.orange, 75))

plot(devCallStrikeValue, "Developing Call", style=plot.style_line,
color=color.new(color.yellow, 95)) plot(devPutStrikeValue, "Developing Put",
style=plot.style_line, color=color.new(color.yellow, 95)) plot(devTopPivot,
"Developing Top Pivot", style=plot.style_line, color=color.new(color.blue, 95))
plot(devCentralPivot, "Developing Central Pivot", style=plot.style_line,
color=color.new(color.purple, 95)) plot(devBottomPivot, "Developing Bottom Pivot",
style=plot.style_line, color=color.new(color.blue, 95))

//Plots //Central Pivot Range plot(topPivot, "CP Top", style=plot.style_circles)
plot(centralPivot, "Central Pivot", style=plot.style_cross, linewidth=2,
color=color.orange) plot(bottomPivot, "CP Bottom", style=plot.style_circles)
```

```
//Traditional Pivots plot(plotTS1R1 ? tR1 : na, "tR1", style=plot.style_circles,
color=color.red) plot(plotTS1R1 ? tS1 : na, "tS1", style=plot.style_circles,
color=color.green) plot(plotTS2R2 ? tR2 : na, "tR2", style=plot.style_circles,
color=color.new(color.red, 30)) plot(plotTS2R2 ? tS2 : na, "tS2",
style=plot.style_circles, color=color.new(color.green, 30)) plot(plotTS3R3 ? tR3 : na,
"tR3", style=plot.style_circles, color=color.new(color.red, 60)) plot(plotTS3R3 ? tS3 :
na, "tS3", style=plot.style_circles, color=color.new(color.green, 60))

// plot(plotCS3R3 ? camR3 : na, "cR3",style=plot.style_line, color=color.red)
plot(plotCS3R3 ? camS3 : na, "cS3", style=plot.style_line, color=color.green)
plot(plotCS1R1 ? camR1 : na, "cR1",style=plot.style_line, color=color.new(color.red,
75)) plot(plotCS1R1 ? camS1 : na, "cS1", style=plot.style_line,
color=color.new(color.green, 75)) plot(plotCS2R2 ? camR2 : na,
"cR2",style=plot.style_line, color=color.new(color.red, 90)) plot(plotCS2R2 ? camS2 :
na, "cS2", style=plot.style_line, color=color.new(color.green, 90)) plot(plotCS4R4 ?
camR4 : na, "cR4",style=plot.style_line, color=color.new(color.red, 25)) plot(plotCS4R4
? camS4 : na, "cS4", style=plot.style_line, color=color.new(color.green, 25))
plot(plotCS5R5 ? camR5 : na, "cR5",style=plot.style_line, color=color.new(color.red,
90)) plot(plotCS5R5 ? camS5 : na, "cS5", style=plot.style_line,
color=color.new(color.green, 90))

//New Bar is_newbar(res, sess) => t = time(res, sess) na(t[1]) and not na(t) or t[1] < t

openingBarClosed = is_newbar(pivotTF, session.regular)[1] openingBarOpen =
is_newbar(pivotTF, session.regular)

//Trade Calculations string tradeType = switch tradeSource > topPivot => "Sell Put"
tradeSource < bottomPivot => "Sell Call" (tradeSource < topPivot and tradeSource >
bottomPivot) => "Sell IC"

var tradeDecision = "No Trade" var strikeValue = 0.00 var strikeValue2 = 0.00 var
yPlacement = 0.00 var labelStyle = label.style_label_lower_right

var tradeLabelText = "None" var alertText ="None"

inBacktestRange = time >= backtestStartDate and time <= backtestEndDate

if openingBarClosed and tradeTheDay and inBacktestRange tradeDecision :=
tradeType[1] strikeValue := tradeDecision == "Sell Put" ? putStrikeValue :
tradeDecision == "Sell Call" ? callStrikeValue : tradeDecision == "Sell IC" ?
icCallStrikeValue : na strikeValue2 := tradeDecision == "Sell Put" ? putStrikeValue2 :
```

tradeDecision == "Sell Call" ? callStrikeValue2 : tradeDecision == "Sell IC" ? icPutStrikeValue : na

```
tradeLabelText := tradeDecision + ' for ' + str.tostring(strikeValue, '#.##')
yPlacement :=
    tradeDecision == "Sell Put" ? putStrikeValue :
    tradeDecision == "Sell Call" ? callStrikeValue :
    tradeDecision == "Sell IC" ? icCallStrikeValue : na
labelStyle :=
    tradeDecision == "Sell Put" ? label.style_label_upper_left:
    tradeDecision == "Sell Call" ? label.style_label_lower_left:
    tradeDecision == "Sell IC" ?  label.style_label_lower_left: label.style_label_lower_le



alertText :=
    tradeDecision == "Sell Put" ?
       "Sell Put Strike " + str.tostring(strikeValue, '#.##') + " | " +
       "Buy Put Strike " + str.tostring(strikeValue2, '#.##')  :
    tradeDecision == "Sell Call" ?
       "Sell Call Strike " + str.tostring(strikeValue, '#.##') + " | " +
       "Buy Call Strike " + str.tostring(strikeValue2, '#.##')  :
    tradeDecision == "Sell IC" ?
       "IC | Sell Call Strike " + str.tostring(icCallStrikeValue, '#.##') + " | " +
       "Buy Call Strike " + str.tostring(icCallStrikeValue2, '#.##') + " | " +
       "Sell Put Strike " + str.tostring(icPutStrikeValue, '#.##') + " | " +
       "Buy Put Strike " + str.tostring(icPutStrikeValue2, '#.##') : na

if showHistoricalTradeLabel or pivotLastBar
   label.new(x=bar_index, y=yPlacement, color=color.new(color.blue,50),
       textcolor=color.white, size=size.small,
       style=labelStyle,
       text=alertText)

alertText := "Ticker:  " + syminfo.ticker + " - " + alertText
```

plot((tradeTheDay and not openingBarOpen and inBacktestRange) ? strikeValue : na, "Strike Value", color=color.white, style=plot.style_linebr)

//Prior Day Trade Result var wins = 0 var losses = 0 var callWins = 0 var callLosses = 0 var icWins = 0 var icLosses = 0 var putWins = 0 var putLosses = 0 var callCount = 0 var icCount = 0 var putCount = 0 var tradeResult = "No Trade"

if openingBarOpen and tradeTheDay[1] and inBacktestRange[1] if tradeDecision == "Sell Put" wins := close[1] > strikeValue ? wins + 1 : wins losses := close[1] < strikeValue ? losses + 1 : losses tradeResult := close[1] > strikeValue ? "Win" : "Loss" putWins := close[1] > strikeValue ? putWins + 1 : putWins putLosses := close[1] < strikeValue ? putLosses + 1 : putLosses putCount := putCount + 1 if tradeDecision == "Sell Call" wins := close[1] < strikeValue ? wins + 1 : wins losses := close[1] > strikeValue ? losses + 1 : losses tradeResult := close[1] < strikeValue ? "Win" : "Loss" callWins := close[1] < strikeValue ? callWins + 1 : callWins callLosses := close[1] > strikeValue ? callLosses + 1 : callLosses callCount := callCount + 1 if tradeDecision == "Sell IC" wins := (close[1] > strikeValue2 and close[1] < strikeValue) ? wins + 1 : wins losses := (close[1] < strikeValue2 or close[1] > strikeValue) ? losses + 1 : losses tradeResult := (close[1] > strikeValue2 and close[1] < strikeValue) ? "Win" : "Loss" icWins := (close[1] > strikeValue2 and close[1] < strikeValue) ? icWins + 1 : icWins icLosses := (close[1] < strikeValue2 or close[1] > strikeValue) ? icLosses + 1 : icLosses icCount := icCount + 1

```
label.new(x=bar_index -1, y=yPlacement, color=color.new(tradeResult == "Win" ? co
    textcolor=color.white, size=size.small,
    style=labelStyle,
    text=tradeResult)
```

// Results & Current Trade

string premiumValue = switch (strikeValue < bottomPivot and close < bottomPivot) or (strikeValue > topPivot and close > topPivot) => "Optimal (Hedge)" (strikeValue < bottomPivot and close < centralPivot) or (strikeValue > topPivot and close > centralPivot) => "Good" (strikeValue < bottomPivot and close < topPivot) or (strikeValue > topPivot and close > bottomPivot) => "Decent" => "Standard"

var table resultsDisplay = table.new(position.bottom_right, 4, 5) var table currentSetup = table.new(position.top_right, 2, 3) if barstate.islast var tfLabelText ="nothing" tfLabelText := '----Current Setup----\n\n' tfLabelText := tfLabelText + 'Pivot Timeframe: ' + pivotTF tfLabelText := tfLabelText + '\n' + tradeLabelText table.cell(currentSetup, 0, 0, 'Pivot TF', text_color=color.white) table.cell(currentSetup, 1, 0, pivotTF, text_color=color.white) table.cell(currentSetup, 0, 1, tradeDecision, text_color=color.white) table.cell(currentSetup, 1, 1, str.tostring(strikeValue, '#.##'), text_color=color.white) table.cell(currentSetup, 0, 2, 'Timing', text_color=color.white) table.cell(currentSetup, 1, 2, premiumValue, text_color=color.white)

```
//Plot Future Values
if showFuture
    futureCall = line.new(x1=bar_index, y1=devCallStrikeValue, x2=bar_index + 5, y2=
    futurePut = line.new(x1=bar_index, y1=devPutStrikeValue, x2=bar_index + 5, y2=c
    futureTopPivot = line.new(x1=bar_index, y1=devTopPivot, x2=bar_index + 5, y2=c
    futureCentralPivot = line.new(x1=bar_index, y1=devCentralPivot, x2=bar_index +
    futureBottomPivot = line.new(x1=bar_index, y1=devBottomPivot, x2=bar_index +


    line.delete(futureCall[1])
    line.delete(futurePut[1])
    line.delete(futureTopPivot[1])
    line.delete(futureCentralPivot[1])
    line.delete(futureBottomPivot[1])


// We only populate the table on the last bar.
if showHistoricalResults
    table.cell(resultsDisplay, 1, 0, 'Trades', text_color=color.white)
    table.cell(resultsDisplay, 2, 0, 'Wins', text_color=color.white)
    table.cell(resultsDisplay, 3, 0, 'Win %', text_color=color.white)
    table.cell(resultsDisplay, 0, 1, 'All', text_color=color.white)
    table.cell(resultsDisplay, 1, 1, str.tostring(wins+losses, '#'), text_color=color.white
    table.cell(resultsDisplay, 2, 1, str.tostring(wins, '#'), text_color=color.white)
    table.cell(resultsDisplay, 3, 1, str.tostring(wins/(wins+losses)*100, '#.#') , text_colo
    table.cell(resultsDisplay, 0, 2, 'Calls', text_color=color.white)
    table.cell(resultsDisplay, 1, 2, str.tostring(callWins + callLosses, '#'), text_color=co
    table.cell(resultsDisplay, 2, 2, str.tostring(callWins, '#') , text_color=color.white)
    table.cell(resultsDisplay, 3, 2, str.tostring(callWins/(callWins+callLosses)*100, '#.#
    table.cell(resultsDisplay, 0, 3, 'Puts', text_color=color.white)
    table.cell(resultsDisplay, 1, 3, str.tostring(putWins + putLosses, '#'), text_color=co
    table.cell(resultsDisplay, 2, 3, str.tostring(putWins, '#') , text_color=color.white)
    table.cell(resultsDisplay, 3, 3, str.tostring(putWins/(putWins + putLosses)*100, '#.
    table.cell(resultsDisplay, 0, 4, 'ICs', text_color=color.white)
    table.cell(resultsDisplay, 1, 4, str.tostring(icWins + icLosses, '#'), text_color=color.
    table.cell(resultsDisplay, 2, 4, str.tostring(icWins, '#') , text_color=color.white)
    table.cell(resultsDisplay, 3, 4, str.tostring(icWins/(icWins + icLosses)*100, '#.#') , te
```

```
//Alerts

sendAlert = close < 6000

webHookAlertMessage = '{"content":"' + alertText + '"}'

//{"content":"{{ticker}} Long Entry Triggered | Open: {{open}} | Close: {{close}} | Low: {{low}} |High: {{high}}"}

if (sendStandardAlert and openingBarClosed) alert(alertText, alert.freq_once_per_bar)

if (sendDiscordWebhookAlert and openingBarClosed) alert(webHookAlertMessage, alert.freq_once_per_bar)

premiumZoneChange = (premiumValue == "Optimal (Hedge)" and premiumValue[1]!= "Optimal (Hedge)") or (premiumValue == "Good" and (premiumValue[1] != "Optimal (Hedge)" and premiumValue[1] != "Good")) or (premiumValue == "Decent" and (premiumValue[1] != "Optimal (Hedge)" and premiumValue[1] != "Good" and premiumValue[1] != "Decent"))

if premiumZoneChange and sendPremiumZoneChangeAlert alert(syminfo.ticker + " Premium Zone has changed to " + premiumValue, alert.freq_once_per_bar)
```