**Detailed Breakdown of "The Tiny Oracle" Indicator Code**

This script, written in Pine Script version 5, creates a technical indicator named "The Tiny Oracle" used on TradingView. The indicator analyzes market conditions using various statistical measures and technical analysis tools, then provides buy, sell, or neutral signals. Here's a detailed breakdown of the code, explaining each part step-by-step in a way that is easy to understand.

**1. Indicator Declaration**

pinescript

Copy code

//@version=5

//@Yaseen Khalil

indicator("The Tiny Oracle", overlay=true)

* **//@version=5**: This line specifies that the script uses version 5 of Pine Script, which is the latest version with updated features.
* **//@Yaseen Khalil**: A comment that indicates the author of the script.
* **indicator("The Tiny Oracle", overlay=true)**: Defines the indicator with the name "The Tiny Oracle." The overlay=true parameter means that the indicator will display directly on the price chart rather than in a separate panel.

**2. User Inputs and Data Retrieval**

pinescript

Copy code

// User inputs

symbolInput = input.symbol("CRYPTOCAP:TOTAL", title="Symbol")

resolutionInput = input.timeframe("", title="Resolution")

src = request.security(symbolInput, resolutionInput, close)

* **symbolInput**: Allows the user to select a symbol (default is the total cryptocurrency market cap).
* **resolutionInput**: Lets the user set the time frame for the analysis (e.g., daily, weekly).
* **src**: Fetches the closing price of the selected symbol and timeframe using TradingView’s request.security function.

**3. Alpha, Beta, and Z-Score Calculations**

**Explanation of Terms:**

* **Alpha**: Measures how much a security has performed compared to a benchmark (like S&P 500). A positive alpha means it outperformed, while a negative alpha indicates underperformance.
* **Beta**: Measures volatility compared to the market. A beta greater than 1 indicates higher volatility, while a beta less than 1 indicates lower volatility.
* **Z-Score**: Indicates how far a value is from its mean in terms of standard deviations. It helps identify if the current value is unusual compared to historical data.

pinescript

Copy code

// Function to calculate Alpha, Beta, and Z-Score

calcAlphaBetaZ(source, benchmark) =>

sourceReturns = ta.change(source)

benchmarkReturns = ta.change(benchmark)

cov = ta.sma(sourceReturns \* benchmarkReturns, 20) - ta.sma(sourceReturns, 20) \* ta.sma(benchmarkReturns, 20)

sourceDev = ta.stdev(sourceReturns, 20)

benchDev = ta.stdev(benchmarkReturns, 20)

alpha = cov / (benchDev \* benchDev)

beta = cov / (sourceDev \* sourceDev)

zscore = (source - ta.sma(source, 20)) / ta.stdev(source, 20)

[alpha, beta, zscore]

* **Function Purpose**: This function calculates alpha, beta, and the Z-score of the selected symbol compared to a benchmark (like S&P 500).
* **Calculations**:
  + **sourceReturns and benchmarkReturns**: Calculate the changes (returns) of the source symbol and the benchmark.
  + **cov (Covariance)**: Measures how the returns of the source and the benchmark move together.
  + **sourceDev and benchDev**: Standard deviations of the returns, which measure volatility.
  + **alpha**: Calculated as the covariance divided by the square of the benchmark’s standard deviation.
  + **beta**: Calculated as the covariance divided by the square of the source’s standard deviation.
  + **zscore**: Measures how far the current price is from its 20-period simple moving average (SMA), scaled by its standard deviation.

pinescript

Copy code

// Using S&P 500 (SPX) as the benchmark for Alpha, Beta, Z-Score calculations

benchmark = request.security("SP:SPX", resolutionInput, close)

[alphaVal, betaVal, zScoreVal] = calcAlphaBetaZ(src, benchmark)

* **Benchmark Selection**: Sets S&P 500 (SPX) as the benchmark for comparison.
* **Alpha, Beta, Z-Score Calculation**: Calls the function to calculate these values using the selected symbol (src) and the benchmark.

**4. Interpreting Alpha, Beta, and Z-Score Values**

pinescript

Copy code

// Alpha, Beta, Z-Score Buy, Sell, Neutral Logic

alphaReading = alphaVal > 0 ? "Buy" : alphaVal < 0 ? "Sell" : "Neutral"

betaReading = betaVal > 1 ? "Buy" : betaVal < 1 ? "Sell" : "Neutral"

zScoreReading = zScoreVal < -1 ? "Buy" : zScoreVal > 1 ? "Sell" : "Neutral"

* **Interpretations**:
  + **Alpha**: A positive value suggests a "Buy" because it outperforms the benchmark, and a negative value suggests a "Sell."
  + **Beta**: A beta greater than 1 (more volatility) suggests a "Buy," while less than 1 suggests a "Sell."
  + **Z-Score**: A Z-score less than -1 suggests the price is significantly lower than average ("Buy"), and above 1 suggests it's significantly higher ("Sell").

**5. Formatting Values with Scientific Notation**

pinescript

Copy code

// Function to format values to resemble E notation manually

formatENotation(value) =>

absValue = math.abs(value)

if absValue >= 1000 or absValue < 0.01

exponent = math.floor(math.log10(absValue))

mantissa = value / math.pow(10, exponent)

mantissaStr = str.tostring(mantissa, "#.##")

expStr = str.tostring(exponent)

mantissaStr + "e" + expStr

else

str.tostring(value, "#.##")

* **Purpose**: Converts large or small numbers into scientific notation (E notation) for easier display, especially when values are extremely high or low.

**6. Technical Indicators Calculations**

These sections calculate various popular indicators used in technical analysis, providing signals for traders:

**RSI, Stochastic, CCI, ADX, and Momentum**

pinescript

Copy code

rsiVal = ta.rsi(src, 14)

stochK = ta.stoch(source = src, high = src, low = src, length = 14)

cciVal = ta.cci(src, 20)

adxLength = 14

tr = ta.tr(true)

tr14 = ta.rma(tr, adxLength)

plusDM = math.max(ta.change(ta.highest(close, 1)), 0)

minusDM = math.max(-ta.change(ta.lowest(close, 1)), 0)

plusDM14 = ta.rma(plusDM, adxLength)

minusDM14 = ta.rma(minusDM, adxLength)

plusDI = 100 \* plusDM14 / tr14

minusDI = 100 \* minusDM14 / tr14

dx = 100 \* math.abs(plusDI - minusDI) / (plusDI + minusDI)

adxVal = ta.rma(dx, adxLength)

momentumVal = src - src[10]

* **RSI (Relative Strength Index)**: Measures the speed and change of price movements, helping identify overbought or oversold conditions.
* **Stochastic K**: Indicates momentum and shows the position of the current price relative to its high-low range over time.
* **CCI (Commodity Channel Index)**: Measures the current price relative to its average price, identifying trend reversals or extreme conditions.
* **ADX (+DI and -DI)**: Measures the strength of a trend, with high values indicating a strong trend.
* **Momentum**: Compares the current price to the price 10 periods ago to gauge the speed of price changes.

**MACD, Awesome Oscillator, and Williams %R**

pinescript

Copy code

aoVal = ta.sma(src, 5) - ta.sma(src, 34)

fastMA = ta.ema(src, 12)

slowMA = ta.ema(src, 26)

macdVal = fastMA - slowMA

signalLine = ta.ema(macdVal, 9)

macdHist = macdVal - signalLine

lengthWill = 14

highestHigh = ta.highest(src, lengthWill)

lowestLow = ta.lowest(src, lengthWill)

willrVal = (highestHigh - src) / (highestHigh - lowestLow) \* -100

* **Awesome Oscillator**: Shows market momentum by comparing a short-term and a long-term SMA.
* **MACD (Moving Average Convergence Divergence)**: A trend-following momentum indicator that shows the relationship between two moving averages of a security’s price.
* **Williams %R**: A momentum indicator that measures overbought and oversold levels.

**7. Signal Conditions and Sum Calculations**

pinescript

Copy code

// Oscillator Signal Conditions

rsiRed = rsiVal < 30 ? 1 : 0

rsiBlue = rsiVal > 70 ? 1 : 0

// ... (similar conditions for other indicators)

oscRedSum = rsiRed + stochRed + cciRed + adxRed + aoRed + momentumRed + macdRed + willrRed

oscBlueSum = rsiBlue + stochBlue + cciBlue + adxBlue + aoBlue + momentumBlue + macdBlue + willrBlue

* **Signal Conditions**: Sets conditions for each indicator to produce a red (sell) or blue (buy) signal based on thresholds.
* **Summing Signals**: Aggregates the signals from all oscillators to determine the overall sentiment (bullish or bearish).

**8. Moving Averages and Overall Score Calculation**

pinescript

Copy code

// Moving Averages Calculations and Signals

ema5Red = src < ta.ema(src, 5) ? 1 : 0

ema5Blue = src > ta.ema(src, 5) ? 1 : 0

// ... (similar conditions for other moving averages)

maRedSum = ema5Red + sma5Red + ema10Red + sma10Red + ema20Red + sma20Red + ema50Red + sma50Red + ema100Red + sma100Red + ema200Red + sma200Red

maBlueSum = ema5Blue + sma5Blue + ema10Blue + sma10Blue + ema20Blue + sma20Blue + ema50Blue + sma50Blue + ema100Blue + sma100Blue + ema200Blue + sma200Blue

// Calculate Overall Buy, Sell, Neutral Score

buyScore = (alphaReading == "Buy" ? 1 : 0) + (betaReading == "Buy" ? 1 : 0) + (zScoreReading == "Buy" ? 1 : 0) + oscBlueSum + maBlueSum

sellScore = (alphaReading == "Sell" ? 1 : 0) + (betaReading == "Sell" ? 1 : 0) + (zScoreReading == "Sell" ? 1 : 0) + oscRedSum + maRedSum

// Determine Overall Label

overallLabel = buyScore > sellScore ? "Buy" : sellScore > buyScore ? "Sell" : "Neutral"

* **Overall Score Calculation**: Combines signals from alpha, beta, Z-score, oscillators, and moving averages to produce a final buy, sell, or neutral score.

**9. Displaying Results in a Table**

pinescript

Copy code

// Create Table

var table alphaBetaZTable = table.new(position.bottom\_right, 13, 4, bgcolor=color.rgb(10, 10, 10, 50))

// Update Table with Alpha, Beta, Z-Score and Readings

if bar\_index == bar\_index

table.cell(alphaBetaZTable, 0, 0, "Alpha(α)", text\_color=color.white, text\_size=size.small, bgcolor=color.rgb(30, 30, 30, 10))

// ... (similar updates for Beta, Z-Score, and other indicators)

* **Purpose**: Creates a table to visually display the calculated alpha, beta, Z-score values, and overall buy/sell/neutral signals on the chart.