```
START

// **   DECLARE CONSTANTS

// These are the options of the various properties of the pizza available
SizesAvailable["Small", "Medium", "Large"]  // The size of the pizza
BasesAvailable["Thick", "Thin"]  // The type of base of the pizza
ToppingsAvailable["Pepperoni", "Chicken", "Extra Cheese", "Mushrooms", "Spinach",
"Olives"]  // The toppings available

MaxToppings ← 3  // The maximum number of toppings that can be taken


// **   DECLARE VARIABLES
CurrentID ← 0  // The running unique ID of the order
OrdersCount ← 0  // The running total of the number of confirmed orders
Close ← FALSE  // Status of more orders

Highest ← 0
HighestIndex ← 0
Lowest ← 1000
LowestIndex ← 0
ToppingsSum ← 0

Sizes[1:3]  // Running tracker of the size taken in an order
Bases[1:2]  // Running tracker of the pizza base taken in an order
Toppings[1:6]  // Running tracker of the toppings taken in an order

TotalSizes[1:3]  // Running counter of the sizes taken
TotalBases[1:2]  // Running counter of the pizza bases taken
TotalToppings[1:6]  // Running counter of the toppings taken

// Initialize the array with all values 0
FOR Count ← 1 TO 3  // Iterate 3 times for 3 values
    TotalSizes[Count] ← 0  // Write 0 to the current value
NEXT Count

// Initialize the array with all values 0
FOR Count ← 1 TO 2  // Iterate 2 times for 2 values
    TotalBases[Count] ← 0  // Write 0 to the current value
NEXT Count

// Initialize the array with all values 0
FOR Count ← 1 TO 6  // Iterate 6 times for 6 values
    TotalToppings[Count] ← 0  // Write 0 to the current value
NEXT Count

// **   TASK 1
// Use a default status "Alter" to customize the pizza
// Input the values of each attribute and validate them
// Give the customer a choice to alter the order, confirm it or cancel it
// If they choose to alter, re-input the values
// If they confirm it, provide them with a new order number.

// **   TASK 2
// Increment a counter of number of pizzas if an order is confirmed
// Add the value of the Counters[] to the TotalCounters[]
// Output the number of pizzas ordered.


REPEAT

    Status ← "Alter"  // Default status to input values

    // Input and validate the values
    WHILE Status = "Alter" DO  // As long as the status is "Alter"

        // Reset the running trackers
        FOR Count ← 1 TO 2
            Sizes[Count] ← FALSE
            Bases[Count] ← FALSE
            Bases[3] ← FALSE
            Toppings[Count] ← FALSE
            Toppings[Count + 3] ← FALSE
            Toppings[Count + 4] ← FALSE
        NEXT Count

        // Output the available options

        // Output the sizes
        PRINT "The following sizes are available to choose from:"
        FOR Count ← 1 TO 3  // Iterate 3 times for 3 sizes
            PRINT SizesAvailable[Count]  // Output the available sizes
        NEXT Count

        // Output the bases
        PRINT "The following bases are available to choose from:"
        FOR Count ← 1 TO 2  // Iterate 2 times for 2 pizza bases
            PRINT BasesAvailable[Count]  // Output the available bases
        NEXT Count

        // Output the toppings
        PRINT "The following toppings are available to choose from:"
        FOR Count ← 1 TO 6  // Iterate 6 times for 6 toppings
            PRINT ToppingsAvailable[Count]  // Output the available toppings
        NEXT Count

        //Input and validate the size of the pizza
        REPEAT  // Validation loop
            PRINT "Please enter the size of the pizza you would like:"  // Input prompt
            INPUT Size  // Input the size

            SizeValid ← FALSE  // Set flag as invalid

            // Check if the size is valid
            FOR Count ← 1 TO 3  // Iterate 3 times for 3 sizes
                IF Size = SizesAvailable[Count]  // If a match is found from the available sizes
                THEN
                    SizeValid ← TRUE  // Set flag as valid
                    Sizes[Count] ← TRUE  // Set flag as selected
                ENDIF
            NEXT Count

        UNTIL SizeValid = TRUE  // Unless the size is invalid, break out of the loop

        // Input and validate the type of pizza base
        REPEAT  // Validation loop
            PRINT "Please enter the type base of the pizza you would like:"  // Input prompt
            INPUT Base  // Input the size

            BaseValid ← FALSE  // Set flag as invalid

            FOR Count ← 1 TO 2  // Iterate 2 times for two sizes
                IF Base = BasesAvailable[Count]  // If a match is found from the available pizza bases
                THEN
                    BaseValid ← TRUE  // Set flag as valid
                    Bases[Count] ← TRUE  // Set flag as selected
                ENDIF
            NEXT Count

        UNTIL BaseValid = TRUE  // Unless the type of pizza base is invalid, break out of the loop

        // Input and validate the number of toppings the customer wants
        REPEAT  // Validation loop
            PRINT "How many toppings do you want on your pizza? You may enter any whole number 0 and 3."  // Input prompt
            INPUT ToppingChoice  // Input the number of toppings the customer wants
        UNTIL ToppingChoice <= MaxToppings  // Unless the number of toppings is greater than 3, break out of the loop

        FOR CountO ← 1 TO ToppingChoice  // Iterate as many times as the toppings taken

            // Input and validate the topping
            REPEAT  // Validation loop
                PRINT "Please enter the topping on the pizza you would like:"  // Input prompt
                INPUT Topping  // Input the topping

                ToppingValid ← FALSE  // Set flag as invalid

                FOR CountI ← 1 TO 6  // Iterate 6 times for 6 toppings
                    IF Topping = ToppingsAvailable[CountI]  // If a match is found from the available toppings
                    THEN
                        ToppingValid ← TRUE  // Set flag as valid
                        Toppings[CountI] ← TRUE  // Set flag as selected
                    ENDIF
                NEXT Count

            UNTIL ToppingValid = TRUE  // Unless the topping is invalid, break out of the loop

        NEXT CountO  // Move on to the next topping

        // Allow the customer to choose whether they want to alter their order, confirm it or cancel it
        PRINT "Do you want to Alter your order, Confirm or Not proceed?"  // Input prompt
        INPUT Status  // Input whether the customer wants to alter their order, confirm it or cancel it

    ENDWHILE  // Unless they want to alter their order, break out of the loop

    // Give the customer a unique order ID if they have confirmed it
    IF Status = "Confirm"  // If the customer has confirmed their order
    THEN
        PRINT "Your unique order number is:", CurrentID  // Print out the unique ID
        CurrentID ← CurrentID + 1  // Increment the ID for the next confirmed order
        OrdersCount ← OrdersCount + 1  // Increment the counter for confirmed orders

        // Record how many of each size has been ordered
        FOR Count ← 1 TO 3  // Iterate 3 times for 3 sizes
            IF Sizes[Count] = TRUE  // If a size is recorded
                THEN TotalSizes[Count] ← TotalSizes[Count] + 1  // Increment the counter
            ENDIF
        NEXT Count

        // Record how many of each pizza base has been ordered
        FOR Count ← 1 TO 2  // Iterate 2 times for 2 pizza bases
            IF Bases[Count] = TRUE  // If a pizza base is recorded
                THEN TotalBases[Count] ← TotalBases[Count] + 1  // Increment the counter
            ENDIF
        NEXT Count

        // Record how many of each topping has been ordered
        FOR Count ← 1 TO 6  // Iterate 6 times for 6 toppings
            IF Toppings[Count] = TRUE  // If a topping has been ordered
                THEN TotalToppings[Count] ← TotalToppings[Count] + 1  // Increment the counter
            ENDIF
        NEXT Count

    ENDIF


    PRINT "Do you want to exit the program?"  // Input prompt
    INPUT BOOLEAN Close  // Ask the staff if all orders are done

UNTIL Close = TRUE  // Break out of the loop unless more pizzas are to be ordered

PRINT OrdersCount, "pizzas were ordered."  // Output how many pizzas were ordered

// **   TASK 3
// Calculate the total number of toppings ordered
// Calculate the highest ordered toppings
// Calculate the lowest ordered toppings
// Express both values as a percentage of the total orders

FOR Count ← 1 TO 6  // Iterate 6 times for 6 toppings
    ToppingsSum ← ToppingsSum + TotalToppings[Count]  // Add to the running total to calculate the sum

    // Calculate the highest sales
    IF TotalToppings[Count] > Highest  // If the current topping sold more than the running most popular topping
    THEN
        Highest ← TotalToppings[Count]  // Update the running most popular topping
        HighestIndex ← Count  // Record the array index of the topping
    ENDIF

    // Calculate the lowest sales
    IF (TotalToppings[Count] < Lowest) AND (TotalToppings[Count] > 0)  // If the current topping sold less than the running least popular topping and it sold in the first place
    THEN
        Lowest ← TotalToppings[Count]  // Update the running least popular topping
        LowestIndex ← Count  // Record the array index of the topping
    ENDIF

NEXT Count

PRINT Toppings[HighestIndex], "was the most popular topping and accounted for" ((Highest/ToppingsSum) * 100), "% of the toppings sales."  // Output the most popular toppings
PRINT Toppings[LowestIndex], "was the least popular topping and accounted for" ((Lowest/ToppingsSum) * 100), "% of the toppings sales."  // Output the least popular toppings

// This is the end of the program
// All required tasks have been completed.

END

```