SCIENTIFIC CALCULATOR

INTRODUCTION: This project focuses on the developing and designing of SCIENTIFIC CALCULATOR using python language. A scientific calculator is an essential and a very useful tool for not only students but also scientists, engineers and anyone who deals with mathematical computations. Our main motive to build a user friendly program capable of performing both basic operations like subtraction, addition, multiplication, matrix calculations and basic trigonometric calculations. We understood the real time application of python program in mathematical problem solving and strengthened our understanding of functions, modules, and error handling.

OBJECTIVE: The core objectives of this project were:

1. TO CREATE A TOOL AS USER FRIENDLY INTERFACE which helps the user to easily interact with the calculator and obtain the results.

2. TO IMPROVE PROBLEM SOLVING SKILLS AND HAVE A LOGICAL THINKING in the areas like designing, inputs handling, execution and writing of codes.

3. TO UNDERSTAND PYTHON BY APPLYING DIFFERENT FUNCTIONS AND LIBARARIES required in the project and attain basic knowledge about them.

4. TO BRING THE FACILITIES OF CALCULATOR IN A SOFTWARE FORMAT which are easily accessible.

5. TO DESIGN A FUNCTIONAL CALCULATOR that performs a wide range of mathematical operations.

PROBLEMS FACED IN THE PROJECT DEVELEOPMENT :

1. PROVIDING SCIENTIFIC FUNCTIONS:
   The trigonometric functions like sin, cos, tan and logarithmic functions required proper use of math module we faced challenges in converting degrees to radians and handling undefined values.

2. HANDLING INVALID INPUTS:
   At first, the program would crash whenever the user entered an invalid number, letter, or symbol. For example, typing a string where a number was expected caused a runtime error.

3. ERROR HANDLING FOR MATHEMATICAL LIMITATIONS:
   Operations such as division by zero, square root of negative numbers, or logarithm of non-positive values caused errors. We had to find ways to manage such cases without crashing the program.

4. HANDLING THE MATRIX OPERATIONS:
   Adding matrix operations and handling in the inputs of matrix format was one of the unique and most difficult part of our project.

5. CREATING A STRUCTURES MENU SYSTEM:
   Designing a clear, easy-to-navigate menu was harder than expected. With many operations, the program became confusing without proper organization.

SOLUTIONS TO THE PROBLEMS:

1. USING TRY-EXCEPT BLOCKS FOR INPUT VALIDATION:
   To solve invalid input issues, we used try-except statements. These blocks catch errors and display friendly messages instead of terminating the program, making it more robust.

2. UTILIZING THE MATH MODULE CORRECTLY:
   We studied the math library functions in detail. For trigonometric functions, we converted degrees to radians using math.radians(). We also handled special cases like undefined values by adding conditional statements.

3. DIVIDING THE PROGRAM INTO FUNCTIONS:
   To manage the complexity, we created separate functions for each operation. This removed repetition, made the code cleaner, and simplified debugging.

4. DESIGNING A CLEAR AND CATEGORIZED MENU:
   We divided operations into groups such as Basic, Scientific, and Advanced. This helped the user navigate easily and made the overall program more organized.

5. ADDING LOGICAL CHECKS FOR MATH ERRORS:
   We added conditions like checking if the denominator is zero, if the value inside a square root is non-negative, or if log input is positive. This prevented undefined operations.

CONCLUSION:

Creating the calculator project enhanced our knowledge of python programming of functions and modules, error handling. It also improved our ability of code skills, designing and problem solving. The project successfully met its objectives and gave us practical experience that will be helpful for future programming tasks.