GUI Calculator

Using OOP

Instead of using library use manual Methods

```
97      def calculate(self):
98          try:
99              self.expression = self.expression.replace( _old: '^', _new: '**')
100             result = str(self.evaluate_expression(self.expression))
101             self.equation.set(result)
102             self.expression = result
103         except:
104             self.equation.set(" error ")
105             self.expression = ""
106
107     def evaluate_expression(self, expr):  1 usage
108         expr = expr.replace("sin", "self.my_sin")
109         expr = expr.replace("cos", "self.my_cos")
110         expr = expr.replace("tan", "self.my_tan")
111         expr = expr.replace("sqrt", "self.my_sqrt")
112
113         return eval(expr, {"self": self})  #
114
115     def create_scientific_buttons(self):  1 usage
116         scientific_buttons = [
117             ('sin', 1, 0), ('cos', 1, 1), ('tan', 1, 2), ('log', 1, 3), ('ln', 1, 4),
118             ('sqrt', 4, 4), ('^', 3, 4), ('(', 2, 4), (')', 1, 4)
119         ]
```

```python
    def my_sin(self, x):  1 usage
        x = self.to_radians(x)
        sin_x = 0
        for i in range(10):
            sin_x += ((-1) ** i * (x ** (2 * i + 1))) / self.factorial(2 * i + 1)
        return sin_x

    def my_cos(self, x):  1 usage
        x = self.to_radians(x)
        cos_x = 0
        for i in range(10):
            cos_x += ((-1) ** i * (x ** (2 * i))) / self.factorial(2 * i)
        return cos_x

    def my_tan(self, x):
        sin_x = self.my_sin(x)
        cos_x = self.my_cos(x)
        return sin_x / cos_x if cos_x != 0 else "undefined"

    def my_sqrt(self, x):
        guess = x / 2
        for _ in range(10):
            guess = (guess + x / guess) / 2  # Newton's method
        return guess

    def to_radians(self, degrees):  2 usages
        return degrees * 3.141592653589793 / 180

    def factorial(self, n):  2 usages
        if n == 0 or n == 1:
            return 1
        result = 1
        for i in range(2, n + 1):
            result *= i
        return result
```

Add Quadratic Equation
That returns only ONE ‘X’ Value

$$4x^2 - 5x - 12 = 0$$

To implement this without using external libraries, we will:

1. **Manually compute the discriminant**: $D = b^2 - 4ac.$
2. **Manually compute the square root** using our `my_sqrt()` function.
3. **Calculate the two possible values of x** using the quadratic formula.

Formula

$$ax^2 + bx + c = 0$$

a, b, c = known numbers, where a ≠ 0

x = the unknown

If **D > 0**, two real and distinct solutions exist.

If **D = 0** , one real and repeated solution exists.

If **D < 0**, complex solutions exist (we will only return "No Real Solutions").