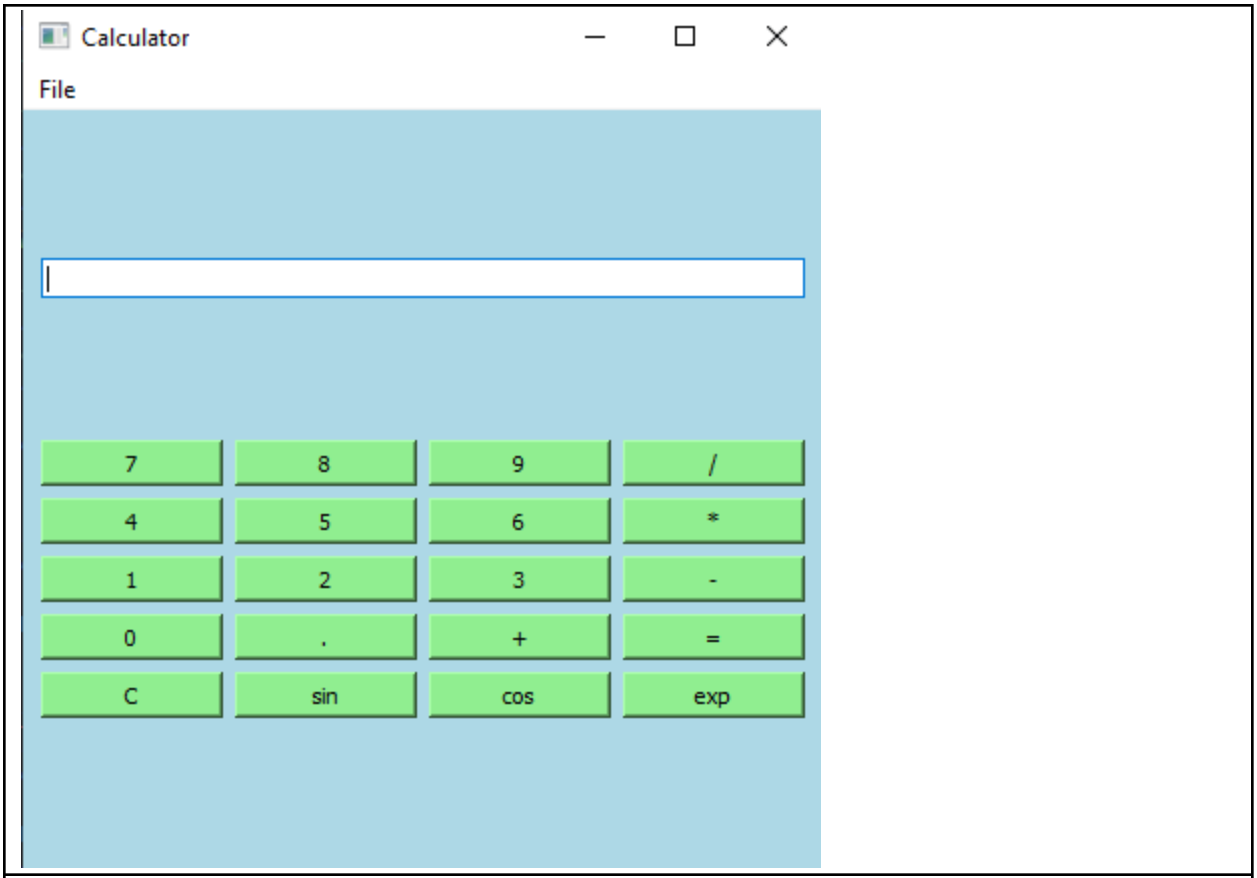
Laboratory Activity 6 - GUI Design: Layout and Styling	
Course Code: CPE009B	**Program:** Computer Engineering
Course Title: Object-Oriented Programming	**Date Performed: November 6, 2024**
Section: CPE21S1	**Date Submitted: November 6, 2024**
Name(s): GASPAR, AARON ROWEN O.	**Instructor: Ms. Maria Rizette Sayo**

Task:

```
import sys
import math
from PyQt5.QtWidgets import QApplication, QMainWindow, QVBoxLayout, QWidget, QLineEdit, QPushButton, QGridLayout,
from PyQt5.QtGui import QKeySequence

class Calculator(QMainWindow):
    def __init__(self):
        super().__init__()
        self.initUI()

    def initUI(self):
        self.setWindowTitle('Calculator')
        self.setGeometry(100, 100, 400, 400)

        self.centralWidget = QWidget(self)
        self.setCentralWidget(self.centralWidget)
        self.layout = QVBoxLayout(self.centralWidget)

        self.display = QLineEdit(self)
        self.layout.addWidget(self.display)

        self.gridLayout = QGridLayout()
        self.layout.addLayout(self.gridLayout)

        buttons = [
            ('7', 0, 0), ('8', 0, 1), ('9', 0, 2), ('/', 0, 3),
            ('4', 1, 0), ('5', 1, 1), ('6', 1, 2), ('*', 1, 3),
            ('1', 2, 0), ('2', 2, 1), ('3', 2, 2), ('-', 2, 3),
            ('0', 3, 0), ('.', 3, 1), ('+', 3, 2), ('=', 3, 3),
            ('C', 4, 0), ('sin', 4, 1), ('cos', 4, 2), ('exp', 4, 3)
        ]

        for text, row, col in buttons:
            button = QPushButton(text, self)
            button.clicked.connect(self.on_click)
            button.setStyleSheet("background-color: lightgreen; color: black;")
            self.gridLayout.addWidget(button, row, col)
```

```
        self.createMenuBar()

    def createMenuBar(self):
        menubar = self.menuBar()
        fileMenu = menubar.addMenu('File')

        saveAction = QAction('Save', self)
        saveAction.triggered.connect(self.save_to_file)
        fileMenu.addAction(saveAction)

        loadAction = QAction('Load', self)
        loadAction.triggered.connect(self.load_from_file)
        fileMenu.addAction(loadAction)

        exitAction = QAction('Exit', self)
        exitAction.setShortcut(QKeySequence('Ctrl+Q'))
        exitAction.triggered.connect(self.close)
        fileMenu.addAction(exitAction)

    def on_click(self):
        sender = self.sender().text()

        if sender == 'C':
            self.display.clear()
        elif sender == '=':
            try:
                result = str(eval(self.display.text()))
                self.display.setText(result)
            except Exception as e:
                self.display.setText('Error')
        elif sender == 'sin':
            try:
                result = str(math.sin(math.radians(float(self.display.text()))))
                self.display.setText(result)
            except Exception as e:
                self.display.setText('Error')
```

```
        elif sender == 'cos':
            try:
                result = str(math.cos(math.radians(float(self.display.text()))))
                self.display.setText(result)
            except Exception as e:
                self.display.setText('Error')
        elif sender == 'exp':
            try:
                result = str(math.exp(float(self.display.text())))
                self.display.setText(result)
            except Exception as e:
                self.display.setText('Error')
        else:
            self.display.setText(self.display.text() + sender)

    def save_to_file(self):
        options = QFileDialog.Options()
        fileName, _ = QFileDialog.getSaveFileName(self, "Save File", "", "Text Files (*.txt);;All Files (*)", opt
        if fileName:
            with open(fileName, 'w') as file:
                file.write(self.display.text())

    def load_from_file(self):
        options = QFileDialog.Options()
        fileName, _ = QFileDialog.getOpenFileName(self, "Open File", "", "Text Files (*.txt);;All Files (*)", opt
        if fileName:
            with open(fileName, 'r') as file:
                self.display.setText(file.read())

if __name__ == '__main__':
    app = QApplication(sys.argv)
    calculator = Calculator()
    calculator.show()
    sys.exit(app.exec_())
```



Conclusion:

This calculator program demonstrates the power and flexibility of PyQt5 for creating functional and user-friendly GUI applications. By incorporating arithmetic and advanced mathematical functions, as well as file handling capabilities, it provides a comprehensive tool for users. The inclusion of error handling ensures robustness, while the menu options enhance usability. This project highlights the importance of combining functionality with a clean and intuitive interface to create effective software solutions.