Міністерство освіти і науки України
Прикарпатський національний університет
імені В.Стефаника

Факультет математики та інформатики
Кафедра інформаційних технологій

Людинно-машинна взаємодія

Лабораторна робота № 9

Тема: Створення проекту «Калькулятор»

Варіант 2

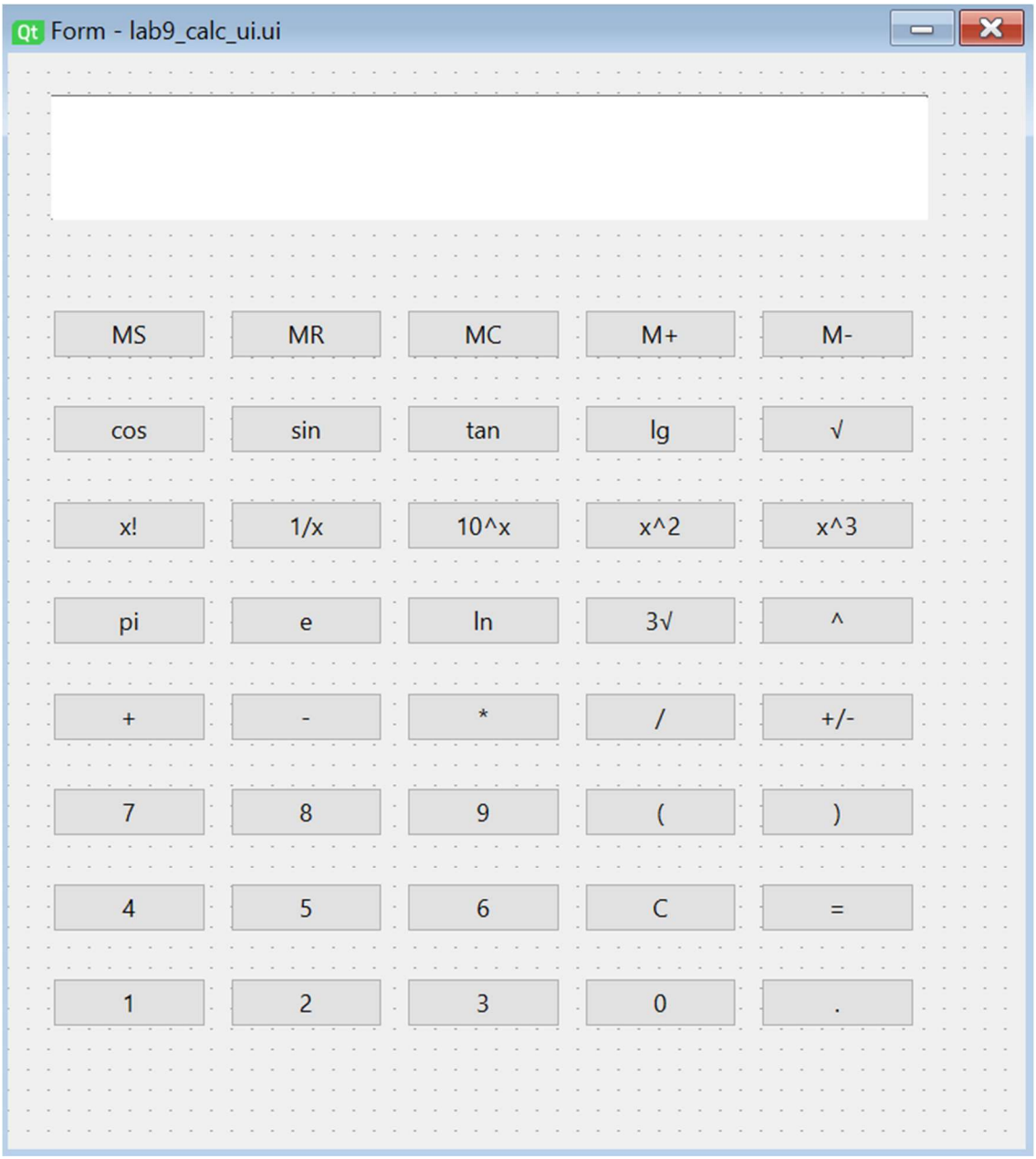
Виконав: ***Гук Д.П.***
Група ІПЗ-31
Дата:2 грудня 2023 р.
Викладач: Пікуляк М.В.

Івано-Франківськ – 2023

Мета роботи:
Створити проект «Калькулятор» за допомогою ICP “Qt-Creator”

Завдання для виконання :
Створити калькулятор за допомогою ICP “Qt-Creator”

Текст скриптів і зображення діалогових вікон QtCreator з виконаним завданням :



Клас CalculatorLogic :

```
class CalculatorLogic:
    def __init__(self, form):
        self.form = form
        self.setup_buttons()
        self.memory = []
        self.pending_operation = None
        self.display_text = ''
        self.result_shown = False


    def setup_buttons(self):

self.form.pushButton.clicked.connect(self.memory_store)

self.form.pushButton_17.clicked.connect(self.memory_rec
all)

self.form.pushButton_25.clicked.connect(self.memory_cle
ar)

self.form.pushButton_31.clicked.connect(self.memory_add
)

self.form.pushButton_26.clicked.connect(self.memory_pop
)


self.form.pushButton_16.clicked.connect(self.addition)

self.form.pushButton_22.clicked.connect(self.division)

self.form.pushButton_19.clicked.connect(self.subtractio
n)
```

```
self.form.pushButton_28.clicked.connect(self.multiplica
tion)


self.form.pushButton_2.clicked.connect(self.cos)

self.form.pushButton_15.clicked.connect(self.sin)

self.form.pushButton_21.clicked.connect(self.tan)


self.form.pushButton_37.clicked.connect(self.factorial)

self.form.pushButton_4.clicked.connect(self.power)

self.form.pushButton_24.clicked.connect(self.square_roo
t)

self.form.pushButton_32.clicked.connect(self.three_root
)

self.form.pushButton_40.clicked.connect(self.power_of_t
en)

self.form.pushButton_38.clicked.connect(self.power_two)

self.form.pushButton_39.clicked.connect(self.power_thre
e)

self.form.pushButton_36.clicked.connect(self.power_minu
s_one)

self.form.pushButton_30.clicked.connect(self.logarithm_
10)
```

```
self.form.pushButton_27.clicked.connect(self.logarithm_
nat)

        self.form.pushButton_5.clicked.connect(lambda:
self.add_to_display('7'))
        self.form.pushButton_6.clicked.connect(lambda:
self.add_to_display('4'))
        self.form.pushButton_7.clicked.connect(lambda:
self.add_to_display('1'))
        self.form.pushButton_18.clicked.connect(lambda:
self.add_to_display('8'))
        self.form.pushButton_11.clicked.connect(lambda:
self.add_to_display('5'))
        self.form.pushButton_9.clicked.connect(lambda:
self.add_to_display('2'))
        self.form.pushButton_23.clicked.connect(lambda:
self.add_to_display('9'))
        self.form.pushButton_10.clicked.connect(lambda:
self.add_to_display('6'))
        self.form.pushButton_34.clicked.connect(lambda:
self.add_to_display('3'))
        self.form.pushButton_35.clicked.connect(lambda:
self.add_to_display('0'))
        self.form.pushButton_3.clicked.connect(lambda:
self.add_to_display('3.14'))
        self.form.pushButton_20.clicked.connect(lambda:
self.add_to_display('2.72'))
        self.form.pushButton_13.clicked.connect(lambda:
self.add_to_display('('))
        self.form.pushButton_41.clicked.connect(lambda:
self.add_to_display(')'))

self.form.pushButton_29.clicked.connect(self.change_sig
n)
```

```
self.form.pushButton_33.clicked.connect(self.decimal_pr
essed)



self.form.pushButton_8.clicked.connect(self.calculate_r
esult)

self.form.pushButton_14.clicked.connect(self.clear_disp
lay)



    def change_sign(self):
        try:
            value = float(self.form.lineEdit.text())
            value *= -1
            self.form.lineEdit.setText(str(value))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def decimal_pressed(self):
        current_text = self.form.lineEdit.text()
        if '.' not in current_text:
            self.form.lineEdit.setText(current_text +
'.')

    def memory_pop(self):
        index = self.get_memory_index()
        try:
            if index is not None and 0 <= index <
len(self.memory):
                self.form.lineEdit.setText("")
                self.memory_index = index
```

```
self.form.lineEdit.setPlaceholderText("Enter value to
subtract")

self.form.pushButton_26.clicked.disconnect()

self.form.pushButton_26.clicked.connect(lambda:
self.memory_store_value('pop'))
            else:
                self.form.lineEdit.setText("Invalid
memory index")
        except ValueError:
            self.form.lineEdit.setText("Error")

    def memory_add(self):
        index = self.get_memory_index()
        try:
            if index is not None and 0 <= index <
len(self.memory):
                self.form.lineEdit.setText("")
                self.memory_index = index

self.form.lineEdit.setPlaceholderText("Enter value to
add")

self.form.pushButton_31.clicked.disconnect()

self.form.pushButton_31.clicked.connect(lambda:
self.memory_store_value('add'))
            else:
                self.form.lineEdit.setText("Invalid
memory index")
        except ValueError:
            self.form.lineEdit.setText("Error")
```

```
    def get_memory_index(self):
        try:
            index = int(self.form.lineEdit.text())
            return index
        except ValueError:
            return None

    def memory_store_value(self, operation):
        try:
            value = float(self.form.lineEdit.text())
            if hasattr(self, 'memory_index'):
                index = self.memory_index
                if operation == 'add':
                    self.memory[index] += value
                    delattr(self, 'memory_index')

self.form.lineEdit.setPlaceholderText("")

self.form.pushButton_31.clicked.disconnect()

self.form.pushButton_31.clicked.connect(self.memory_add
)
                    self.form.lineEdit.setText("")
                elif operation == 'pop':
                    self.memory[index] -= value
                    delattr(self, 'memory_index')

self.form.lineEdit.setPlaceholderText("")

self.form.pushButton_26.clicked.disconnect()

self.form.pushButton_26.clicked.connect(self.memory_pop
)
                    self.form.lineEdit.setText("")
            else:
```

```
                self.form.lineEdit.setText("Please
select a memory index first")
        except ValueError:
            self.form.lineEdit.setText("Error")

    def memory_store(self):
        input_text = self.form.lineEdit.text()
        if input_text:
            try:
                self.memory.append(float(input_text))
                self.form.lineEdit.setText("")
            except ValueError:
                self.form.lineEdit.setText("Invalid
input for memory")
        else:
            self.form.lineEdit.setText("")

    def memory_recall(self):
        try:
            index = int(self.form.lineEdit.text())
            if 0 <= index < len(self.memory):
                recalled_value = self.memory[index]

self.form.lineEdit.setText(str(recalled_value))
            elif len(self.memory) == 0:
                self.form.lineEdit.setText("Memory is
empty")
            else:
                self.form.lineEdit.setText("Index out
of range")
        except ValueError:
            self.form.lineEdit.setText("Invalid index")

    def memory_clear(self):
        self.memory.clear()
```

```
    def cos(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.cos(math.radians(value))
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")



    def sin(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.sin(math.radians(value))
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def tan(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.tan(math.radians(value))
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def logarithm_10(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.log10(value)
            self.form.lineEdit.setText(str(result))
        except ValueError:
```

```
            self.form.lineEdit.setText("Error")

    def logarithm_nat(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.log(value)
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def factorial(self):
        try:
            value = int(self.form.lineEdit.text())
            result = math.factorial(value)
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")
        except OverflowError:
            self.form.lineEdit.setText("Result too
large")

    def addition(self):
        try:
            current_text = self.form.lineEdit.text()
            if '+' not in current_text:
                self.form.lineEdit.setText(current_text
+ ' + ')
            else:
                parts = current_text.split(' + ')
                if len(parts) == 2:
                    num1, num2 = map(float, parts)
                    result = num1 + num2

self.form.lineEdit.setText(str(result))
                    self.result_shown = True
```

```
        except Exception as e:
            self.form.lineEdit.setText("Error")

    def division(self):
        try:
            current_text = self.form.lineEdit.text()
            if '/' not in current_text:
                self.form.lineEdit.setText(current_text
+ ' / ')
            else:
                parts = current_text.split(' / ')
                if len(parts) == 2:
                    num1, num2 = map(float, parts)
                    result = num1 / num2

self.form.lineEdit.setText(str(result))
                    self.result_shown = True
        except Exception as e:
            self.form.lineEdit.setText("Error")

    def subtraction(self):
        try:
            current_text = self.form.lineEdit.text()
            if '-' not in current_text:
                self.form.lineEdit.setText(current_text
+ ' - ')
            else:
                parts = current_text.split(' - ')
                if len(parts) == 2:
                    num1, num2 = map(float, parts)
                    result = num1 - num2

self.form.lineEdit.setText(str(result))
                    self.result_shown = True
        except Exception as e:
```

```
            self.form.lineEdit.setText("Error")

    def modulus(self):
        try:
            current_text = self.form.lineEdit.text()
            if '%' not in current_text:
                self.form.lineEdit.setText(current_text
+ ' % ')
            else:
                parts = current_text.split(' % ')
                if len(parts) == 2:
                    num1, num2 = map(float, parts)
                    result = num1 % num2

self.form.lineEdit.setText(str(result))
                    self.result_shown = True
        except Exception as e:
            self.form.lineEdit.setText("Error")

    def multiplication(self):
        try:
            current_text = self.form.lineEdit.text()
            if '*' not in current_text:
                self.form.lineEdit.setText(current_text
+ ' * ')
            else:
                parts = current_text.split(' * ')
                if len(parts) == 2:
                    num1, num2 = map(float, parts)
                    result = num1 * num2

self.form.lineEdit.setText(str(result))
                    self.result_shown = True
        except Exception as e:
            self.form.lineEdit.setText("Error")
```

```
    def power(self):
        try:
            current_text = self.form.lineEdit.text()
            if '**' not in current_text:
                self.form.lineEdit.setText(current_text
+ ' ** ')
            else:
                parts = current_text.split(' ** ')
                if len(parts) == 2:
                    num1, num2 = map(float, parts)
                    result = num1 ** num2

self.form.lineEdit.setText(str(result))
                    self.result_shown = True
        except Exception as e:
            self.form.lineEdit.setText("Error")
    def square_root(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.sqrt(value)
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def three_root(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.pow(value, (1/3))
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def power_two(self):
        try:
```

```
            value = float(self.form.lineEdit.text())
            result = math.pow(value, 2)
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def power_minus_one(self):
        try:
            value = float(self.form.lineEdit.text())
            result = 1 / value
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def power_three(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.pow(value, 3)
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def power_of_ten(self):
        try:
            value = float(self.form.lineEdit.text())
            result = math.pow(10, value)
            self.form.lineEdit.setText(str(result))
        except ValueError:
            self.form.lineEdit.setText("Error")

    def add_to_display(self, value):
        current_text = self.form.lineEdit.text()
        if self.result_shown:
            self.result_shown = False
            self.display_text = ''
```

```
        self.form.lineEdit.setText(current_text +
value)

    def clear_display(self):
        self.form.lineEdit.setText("")
        self.display_text = ''
        self.result_shown = False

    def calculate_result(self):
        try:
            expression = self.form.lineEdit.text()
            result = eval(expression)
            self.form.lineEdit.setText(str(result))
            self.result_shown = True
        except Exception as e:
            self.form.lineEdit.setText("Error")
```

Клас Ui_Form :

```
class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Calculator")
        Form.resize(470, 500)
        self.lineEdit =
QtWidgets.QLineEdit(parent=Form)
        self.lineEdit.setGeometry(QtCore.QRect(22, 22,
422, 61))
        self.lineEdit.setObjectName("lineEdit")
        lineEdit_font = self.lineEdit.font()
        lineEdit_font.setPointSize(16)
        self.lineEdit.setFont(lineEdit_font)
        self.pushButton =
QtWidgets.QPushButton(parent=Form)
        self.pushButton.setGeometry(QtCore.QRect(22,
```

```
126, 75, 24))
        self.pushButton.setObjectName("pushButton")
        self.pushButton_2 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_2.setGeometry(QtCore.QRect(22,
173, 75, 24))
        self.pushButton_2.setObjectName("pushButton_2")
        self.pushButton_3 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_3.setGeometry(QtCore.QRect(22,
267, 75, 24))
        self.pushButton_3.setObjectName("pushButton_3")
        self.pushButton_4 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_4.setGeometry(QtCore.QRect(370,
267, 75, 24))
        self.pushButton_4.setObjectName("pushButton_4")
        self.pushButton_5 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_5.setGeometry(QtCore.QRect(22,
361, 75, 24))
        self.pushButton_5.setObjectName("pushButton_5")
        self.pushButton_6 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_6.setGeometry(QtCore.QRect(22,
408, 75, 24))
        self.pushButton_6.setObjectName("pushButton_6")
        self.pushButton_7 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_7.setGeometry(QtCore.QRect(22,
455, 75, 24))
        self.pushButton_7.setObjectName("pushButton_7")
        self.pushButton_8 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_8.setGeometry(QtCore.QRect(370,
```

```
408, 75, 24))
        self.pushButton_8.setObjectName("pushButton_8")
        self.pushButton_9 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_9.setGeometry(QtCore.QRect(109,
455, 75, 24))
        self.pushButton_9.setObjectName("pushButton_9")
        self.pushButton_10 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_10.setGeometry(QtCore.QRect(196, 408,
75, 24))

self.pushButton_10.setObjectName("pushButton_10")
        self.pushButton_11 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_11.setGeometry(QtCore.QRect(109, 408,
75, 24))

self.pushButton_11.setObjectName("pushButton_11")
        self.pushButton_13 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_13.setGeometry(QtCore.QRect(283, 361,
75, 24))

self.pushButton_13.setObjectName("pushButton_13")
        self.pushButton_14 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_14.setGeometry(QtCore.QRect(283, 408,
75, 24))

self.pushButton_14.setObjectName("pushButton_14")
```

```
        self.pushButton_15 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_15.setGeometry(QtCore.QRect(109, 173,
75, 24))

self.pushButton_15.setObjectName("pushButton_15")
        self.pushButton_16 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_16.setGeometry(QtCore.QRect(22,
314, 75, 24))

self.pushButton_16.setObjectName("pushButton_16")
        self.pushButton_17 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_17.setGeometry(QtCore.QRect(109, 126,
75, 24))

self.pushButton_17.setObjectName("pushButton_17")
        self.pushButton_18 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_18.setGeometry(QtCore.QRect(109, 361,
75, 24))

self.pushButton_18.setObjectName("pushButton_18")
        self.pushButton_19 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_19.setGeometry(QtCore.QRect(109, 314,
75, 24))

self.pushButton_19.setObjectName("pushButton_19")
        self.pushButton_20 =
```

```
QtWidgets.QPushButton(parent=Form)

self.pushButton_20.setGeometry(QtCore.QRect(109, 267,
75, 24))

self.pushButton_20.setObjectName("pushButton_20")
        self.pushButton_21 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_21.setGeometry(QtCore.QRect(196, 173,
75, 24))

self.pushButton_21.setObjectName("pushButton_21")
        self.pushButton_22 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_22.setGeometry(QtCore.QRect(283, 314,
75, 24))

self.pushButton_22.setObjectName("pushButton_22")
        self.pushButton_23 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_23.setGeometry(QtCore.QRect(196, 361,
75, 24))

self.pushButton_23.setObjectName("pushButton_23")
        self.pushButton_24 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_24.setGeometry(QtCore.QRect(370, 173,
75, 24))

self.pushButton_24.setObjectName("pushButton_24")
        self.pushButton_25 =
```

```
QtWidgets.QPushButton(parent=Form)

self.pushButton_25.setGeometry(QtCore.QRect(196, 126,
75, 24))

self.pushButton_25.setObjectName("pushButton_25")
        self.pushButton_26 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_26.setGeometry(QtCore.QRect(370, 126,
75, 24))

self.pushButton_26.setObjectName("pushButton_26")
        self.pushButton_27 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_27.setGeometry(QtCore.QRect(196, 267,
75, 24))

self.pushButton_27.setObjectName("pushButton_27")
        self.pushButton_28 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_28.setGeometry(QtCore.QRect(196, 314,
75, 24))

self.pushButton_28.setObjectName("pushButton_28")
        self.pushButton_29 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_29.setGeometry(QtCore.QRect(370, 314,
75, 24))

self.pushButton_29.setObjectName("pushButton_29")
        self.pushButton_30 =
```

```
QtWidgets.QPushButton(parent=Form)

self.pushButton_30.setGeometry(QtCore.QRect(283, 173,
75, 24))

self.pushButton_30.setObjectName("pushButton_30")
        self.pushButton_31 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_31.setGeometry(QtCore.QRect(283, 126,
75, 24))

self.pushButton_31.setObjectName("pushButton_31")
        self.pushButton_32 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_32.setGeometry(QtCore.QRect(283, 267,
75, 24))

self.pushButton_32.setObjectName("pushButton_32")
        self.pushButton_33 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_33.setGeometry(QtCore.QRect(370, 455,
75, 24))

self.pushButton_33.setObjectName("pushButton_33")
        self.pushButton_34 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_34.setGeometry(QtCore.QRect(196, 455,
75, 24))

self.pushButton_34.setObjectName("pushButton_34")
        self.pushButton_35 =
```

```
QtWidgets.QPushButton(parent=Form)

self.pushButton_35.setGeometry(QtCore.QRect(283, 455,
75, 24))

self.pushButton_35.setObjectName("pushButton_35")
        self.pushButton_36 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_36.setGeometry(QtCore.QRect(109, 220,
75, 24))

self.pushButton_36.setObjectName("pushButton_36")
        self.pushButton_37 =
QtWidgets.QPushButton(parent=Form)
        self.pushButton_37.setGeometry(QtCore.QRect(22,
220, 75, 24))

self.pushButton_37.setObjectName("pushButton_37")
        self.pushButton_38 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_38.setGeometry(QtCore.QRect(283, 220,
75, 24))

self.pushButton_38.setObjectName("pushButton_38")
        self.pushButton_39 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_39.setGeometry(QtCore.QRect(370, 220,
75, 24))

self.pushButton_39.setObjectName("pushButton_39")
        self.pushButton_40 =
QtWidgets.QPushButton(parent=Form)
```

```
self.pushButton_40.setGeometry(QtCore.QRect(196, 220,
75, 24))

self.pushButton_40.setObjectName("pushButton_40")
        self.pushButton_41 =
QtWidgets.QPushButton(parent=Form)

self.pushButton_41.setGeometry(QtCore.QRect(370, 361,
75, 24))

self.pushButton_41.setObjectName("pushButton_41")

        buttons = [
            self.pushButton, self.pushButton_2,
self.pushButton_3, self.pushButton_4,
            self.pushButton_5, self.pushButton_6,
self.pushButton_7, self.pushButton_8,
            self.pushButton_9, self.pushButton_10,
self.pushButton_11, self.pushButton_13,
            self.pushButton_14, self.pushButton_15,
self.pushButton_16, self.pushButton_17,
            self.pushButton_18, self.pushButton_19,
self.pushButton_20, self.pushButton_21,
            self.pushButton_22, self.pushButton_23,
self.pushButton_24, self.pushButton_25,
            self.pushButton_26, self.pushButton_27,
self.pushButton_28, self.pushButton_29,
            self.pushButton_30, self.pushButton_31,
self.pushButton_32, self.pushButton_33,
            self.pushButton_34, self.pushButton_35,
self.pushButton_36, self.pushButton_37,
            self.pushButton_38, self.pushButton_39,
self.pushButton_40, self.pushButton_41
        ]
```

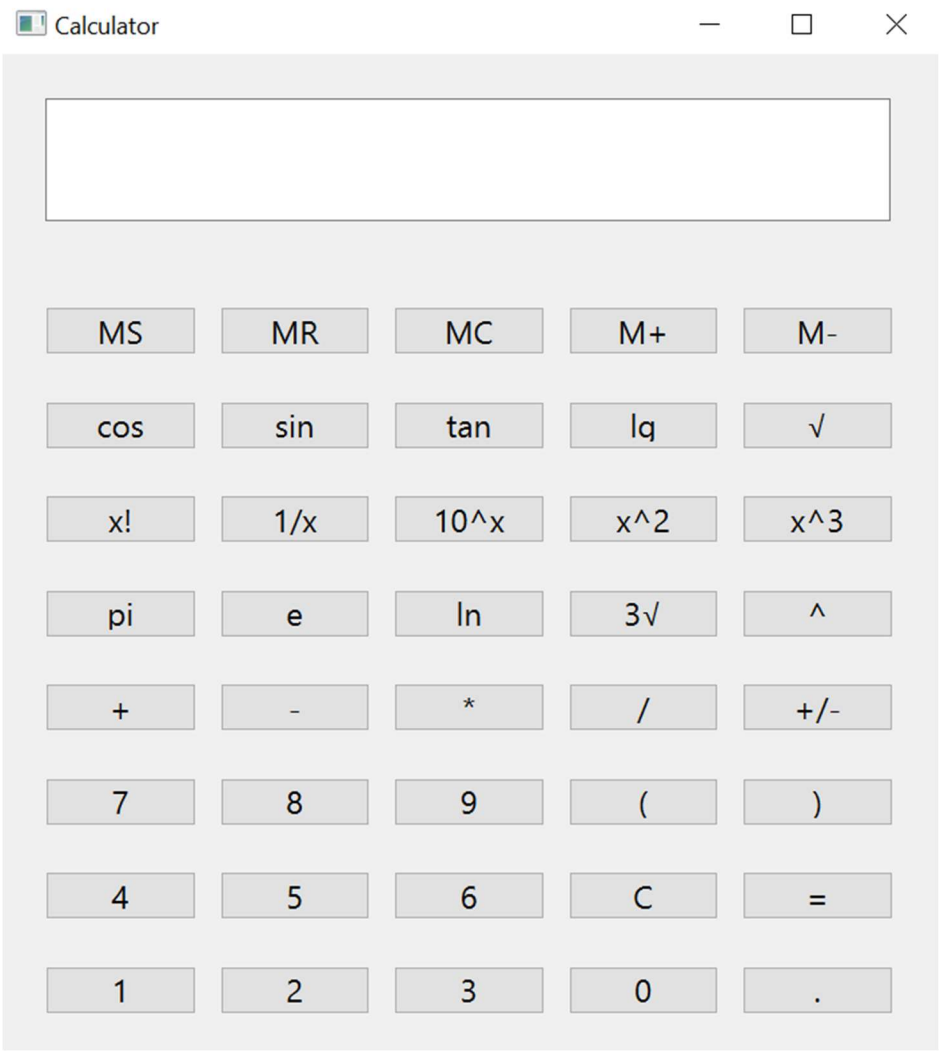
```
        for button in buttons:
            button_font = button.font()
            button_font.setPointSize(12)
            button.setFont(button_font)

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form",
"Calculator"))
        self.pushButton.setText(_translate("Form",
"MS"))
        self.pushButton_2.setText(_translate("Form",
"cos"))
        self.pushButton_3.setText(_translate("Form",
"pi"))
        self.pushButton_4.setText(_translate("Form",
"^"))
        self.pushButton_5.setText(_translate("Form",
"7"))
        self.pushButton_6.setText(_translate("Form",
"4"))
        self.pushButton_7.setText(_translate("Form",
"1"))
        self.pushButton_8.setText(_translate("Form",
"="))
        self.pushButton_9.setText(_translate("Form",
"2"))
        self.pushButton_10.setText(_translate("Form",
"6"))
        self.pushButton_11.setText(_translate("Form",
```

```
"5"))
        self.pushButton_13.setText(_translate("Form",
"("))
        self.pushButton_14.setText(_translate("Form",
"C"))
        self.pushButton_15.setText(_translate("Form",
"sin"))
        self.pushButton_16.setText(_translate("Form",
"+"))
        self.pushButton_17.setText(_translate("Form",
"MR"))
        self.pushButton_18.setText(_translate("Form",
"8"))
        self.pushButton_19.setText(_translate("Form",
"-"))
        self.pushButton_20.setText(_translate("Form",
"e"))
        self.pushButton_21.setText(_translate("Form",
"tan"))
        self.pushButton_22.setText(_translate("Form",
"/"))
        self.pushButton_23.setText(_translate("Form",
"9"))
        self.pushButton_24.setText(_translate("Form",
"√"))
        self.pushButton_25.setText(_translate("Form",
"MC"))
        self.pushButton_26.setText(_translate("Form",
"M-"))
        self.pushButton_27.setText(_translate("Form",
"ln"))
        self.pushButton_28.setText(_translate("Form",
"*"))
        self.pushButton_29.setText(_translate("Form",
"+/-"))
```

```
        self.pushButton_30.setText(_translate("Form",
"lg"))
        self.pushButton_31.setText(_translate("Form",
"M+"))
        self.pushButton_32.setText(_translate("Form",
"3√"))
        self.pushButton_33.setText(_translate("Form",
"."))
        self.pushButton_34.setText(_translate("Form",
"3"))
        self.pushButton_35.setText(_translate("Form",
"0"))
        self.pushButton_36.setText(_translate("Form",
"1/x"))
        self.pushButton_37.setText(_translate("Form",
"x!"))
        self.pushButton_38.setText(_translate("Form",
"x^2"))
        self.pushButton_39.setText(_translate("Form",
"x^3"))
        self.pushButton_40.setText(_translate("Form",
"10^x"))
        self.pushButton_41.setText(_translate("Form",
")"))
```

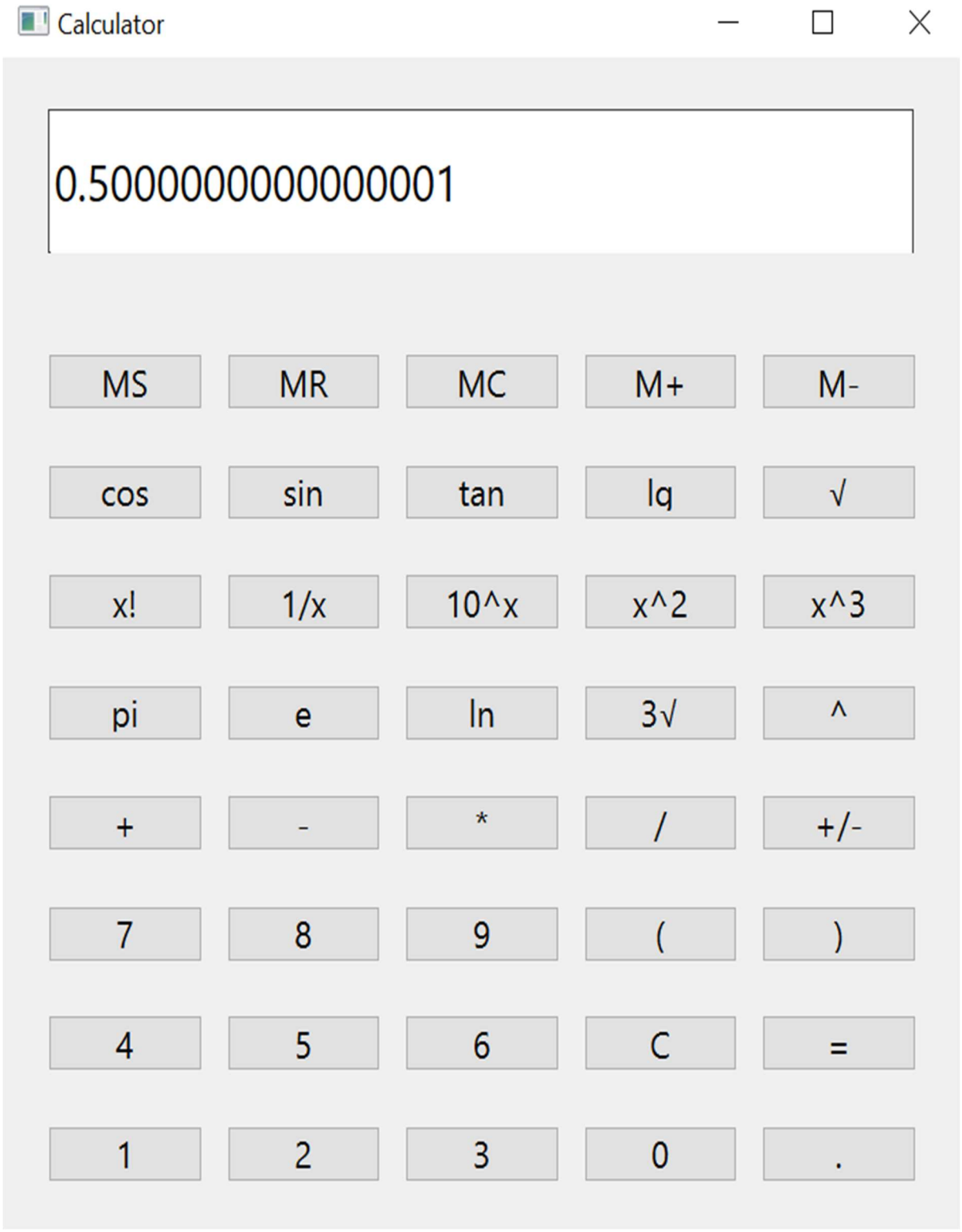
Скрін-шоти виконання завдання лабораторної роботи :

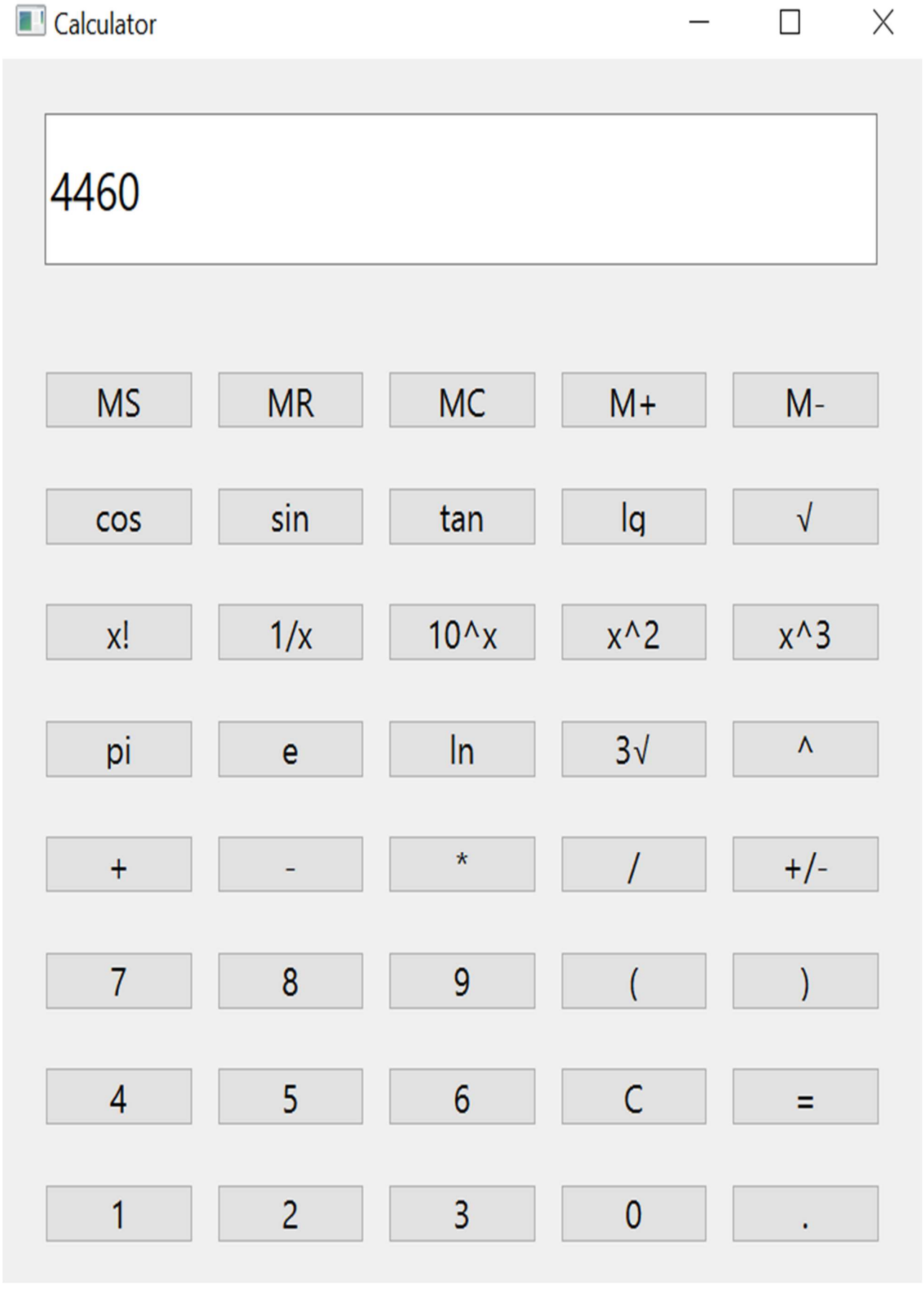
Графічний інтерфейс програми :

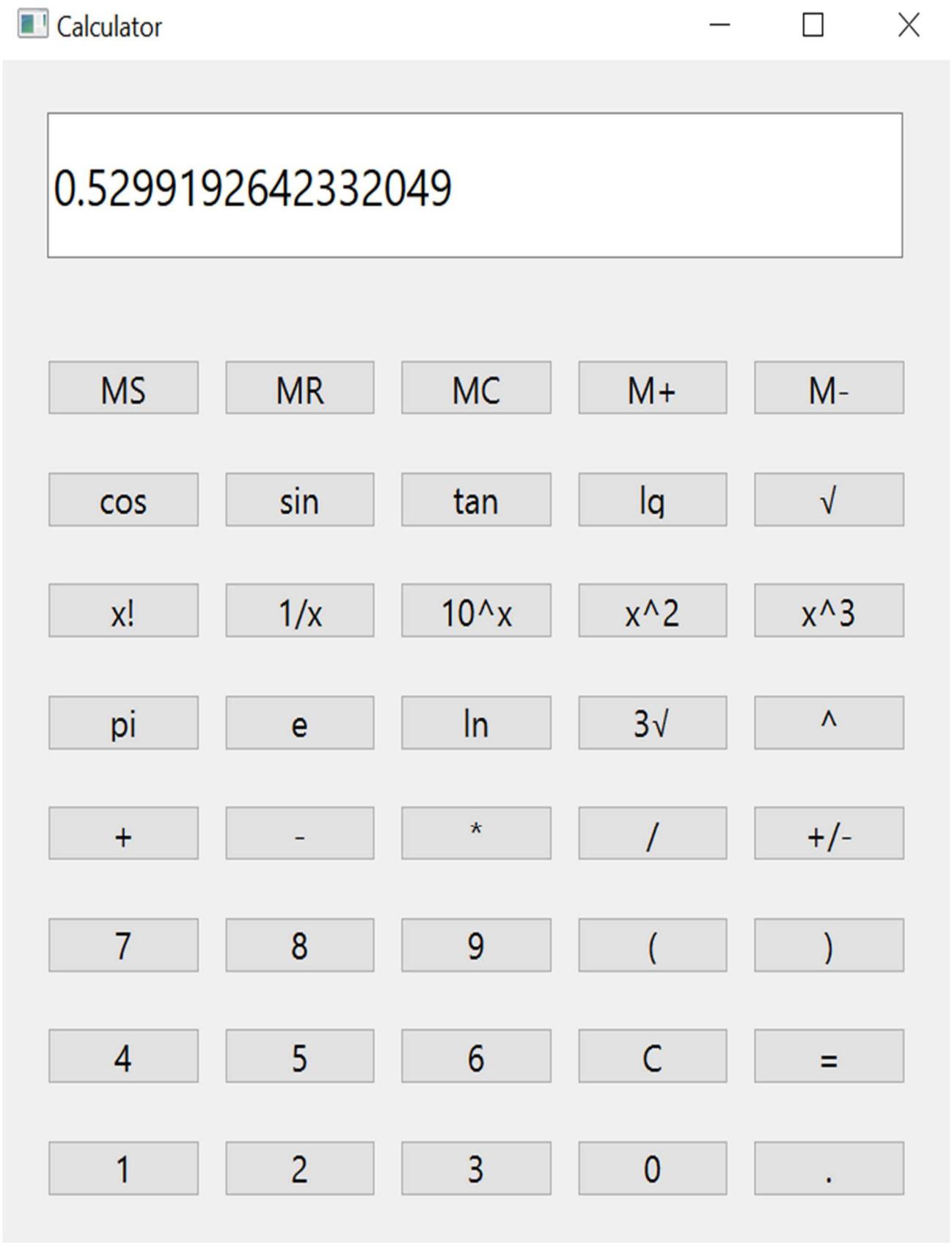


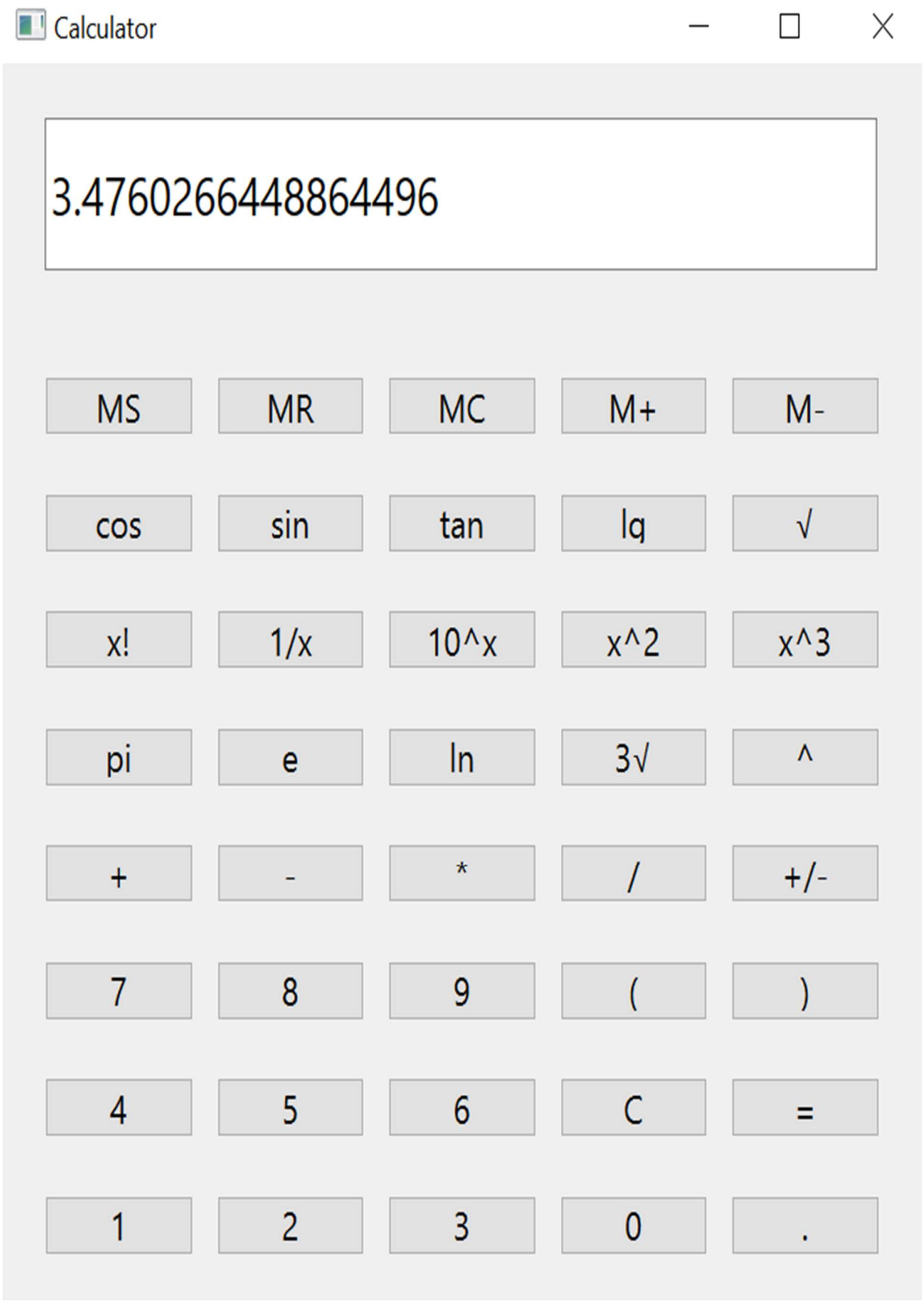
Конторльні приклади для демонстрування роботи програми :

1) $\boldsymbol{cos(60)}$
2) $\boldsymbol{20 * (6^3 + 7)}$
3) $\boldsymbol{sin(32)}$
4) $\boldsymbol{\sqrt[3]{42}}$


Calculator
0.5000000000000001
MS
MR
MC
M+
M-
cos
sin
tan
lg
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.

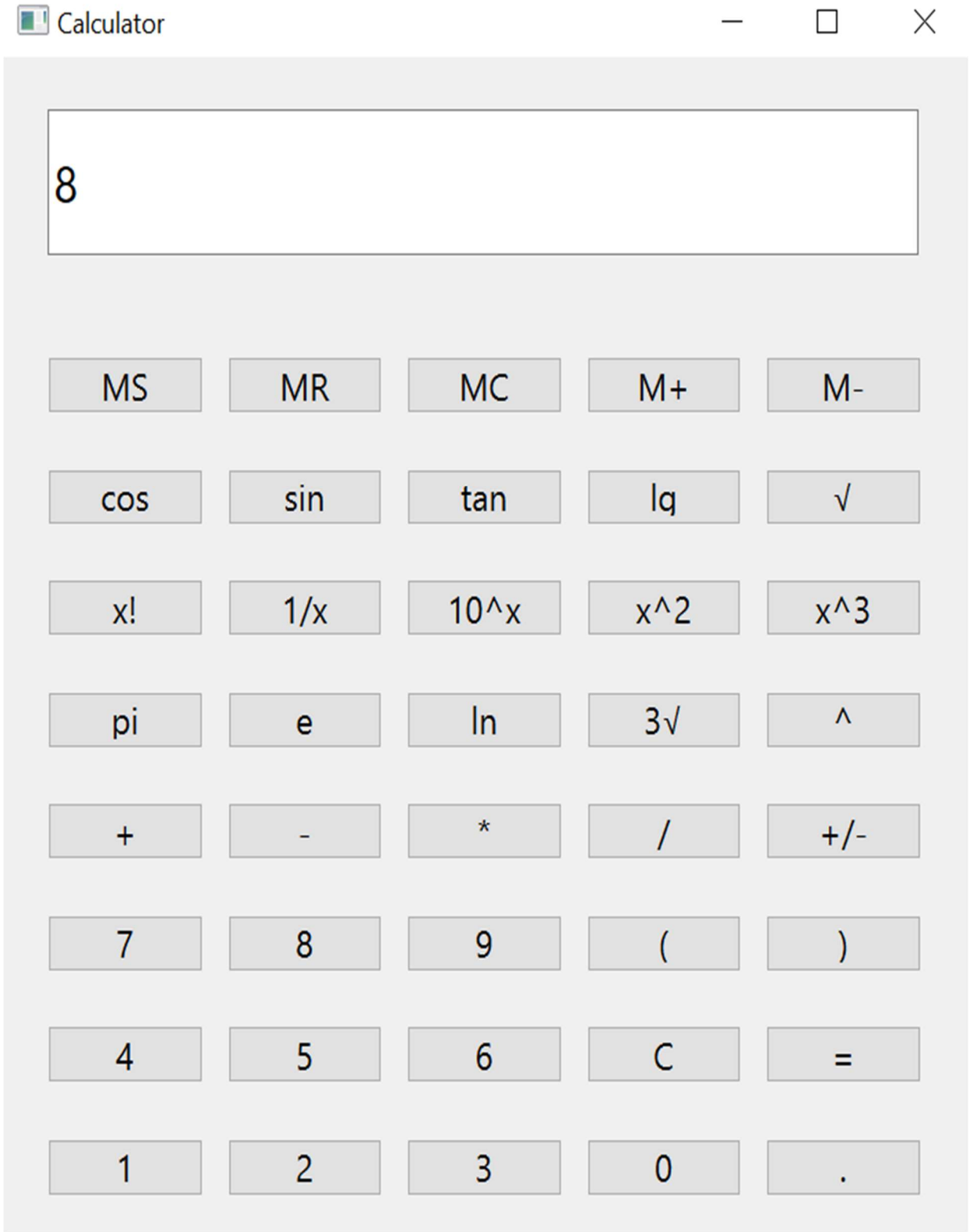

Calculator
4460
MS
MR
MC
M+
M-
cos
sin
tan
lq
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.

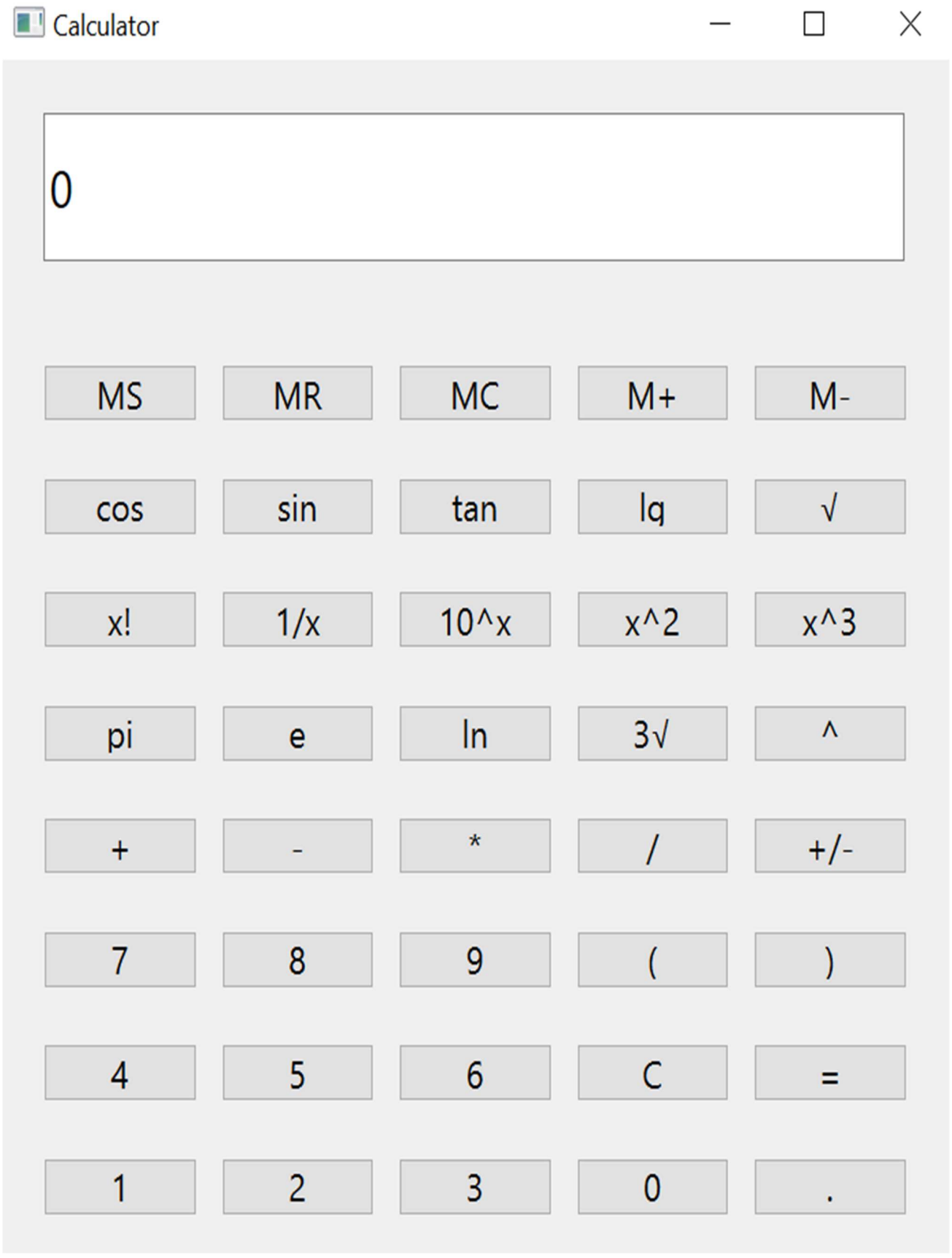

Calculator
0.5299192642332049
MS
MR
MC
M+
M-
cos
sin
tan
lg
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.

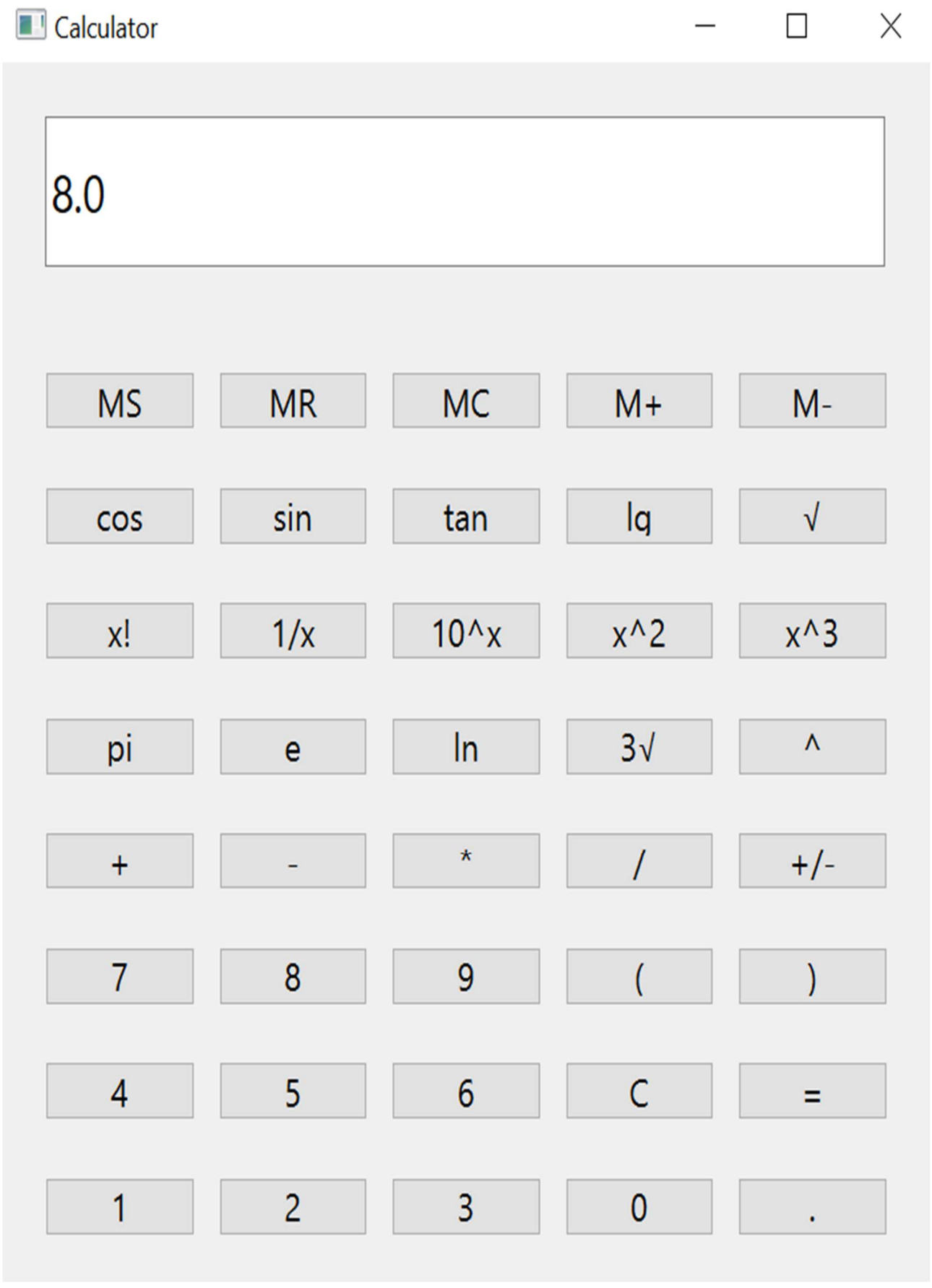

Calculator
3.4760266448864496
MS
MR
MC
M+
M-
cos
sin
tan
lg
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.

Функції роботи з пам’яттю :

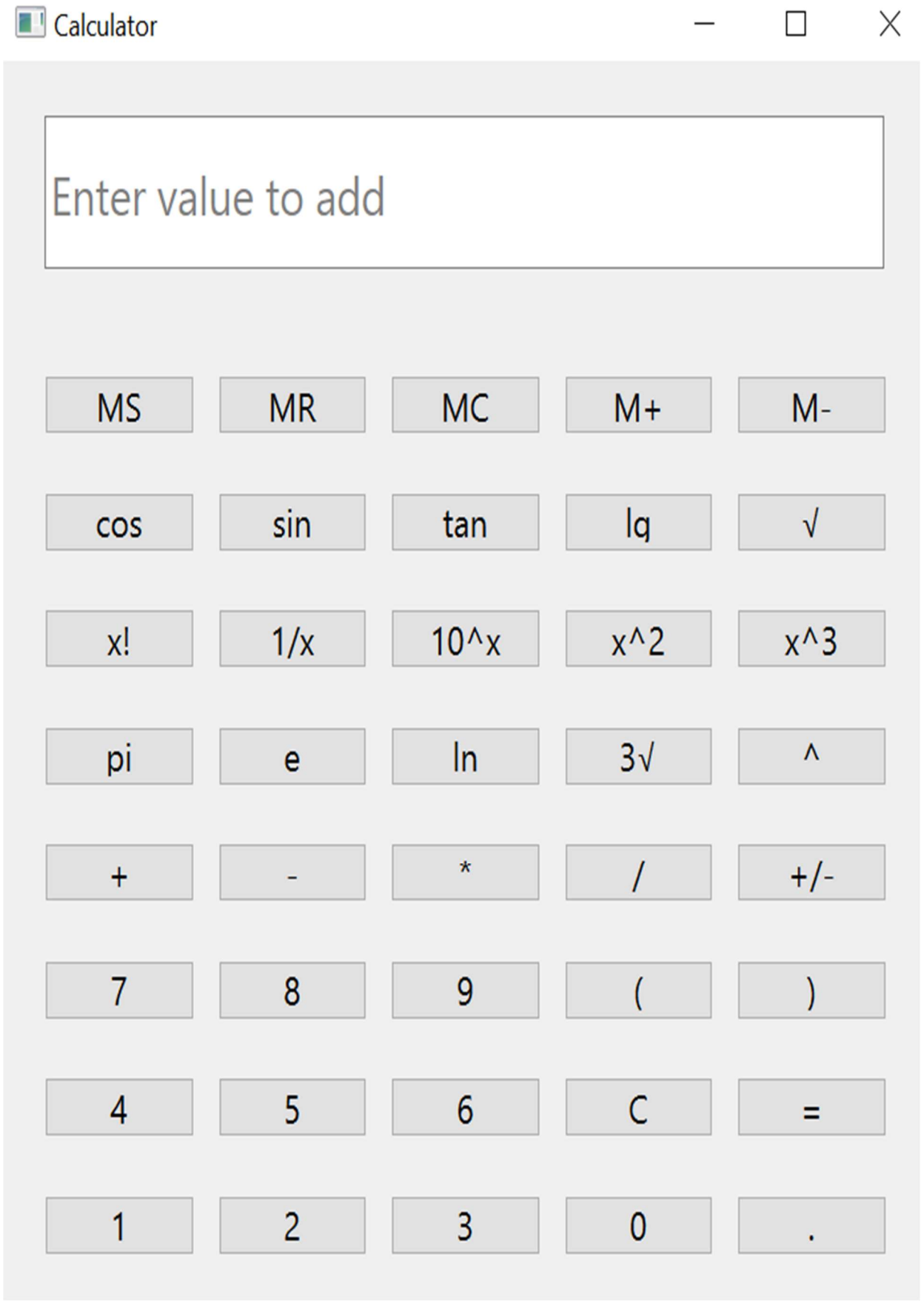
1) MS і MR :

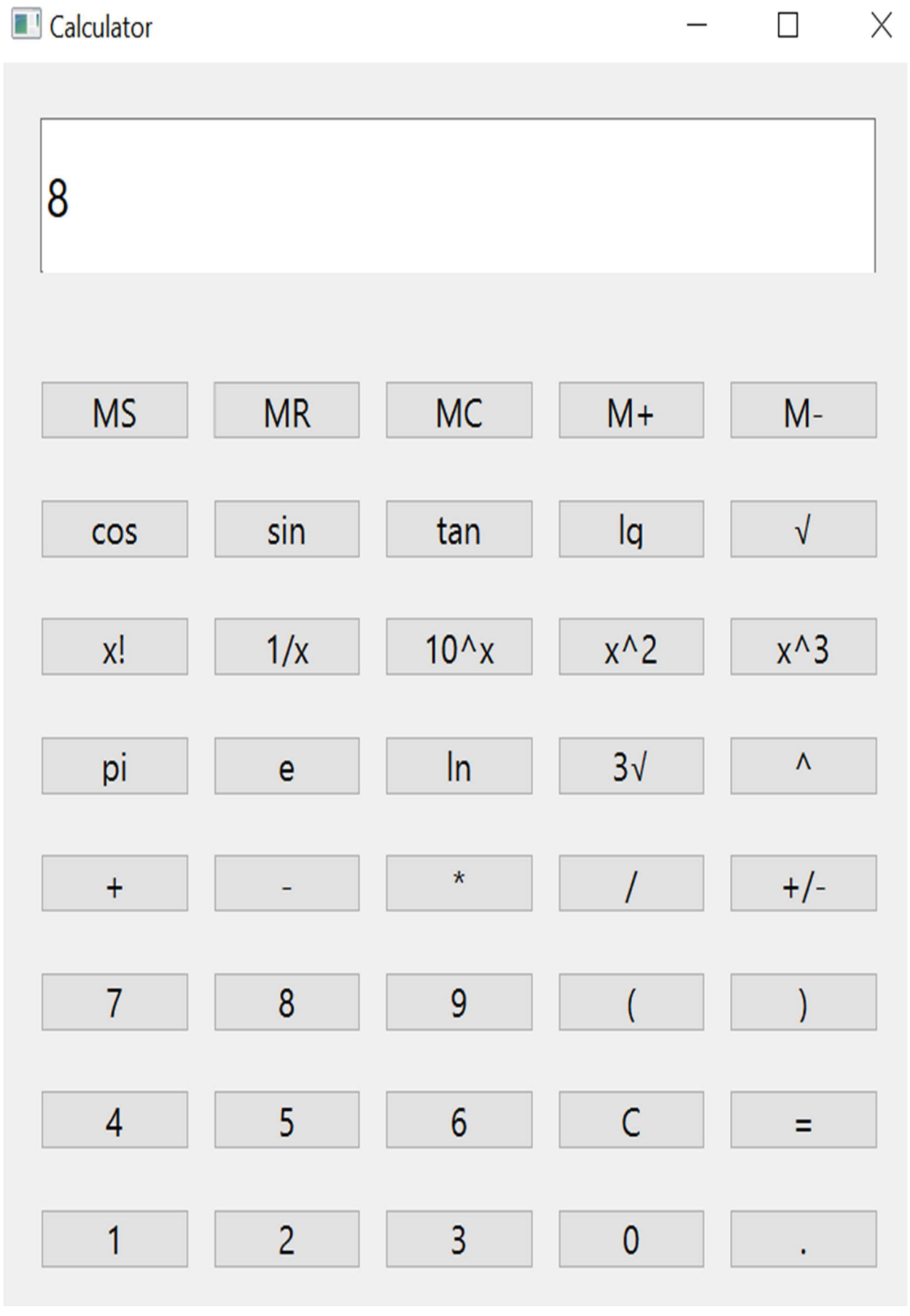


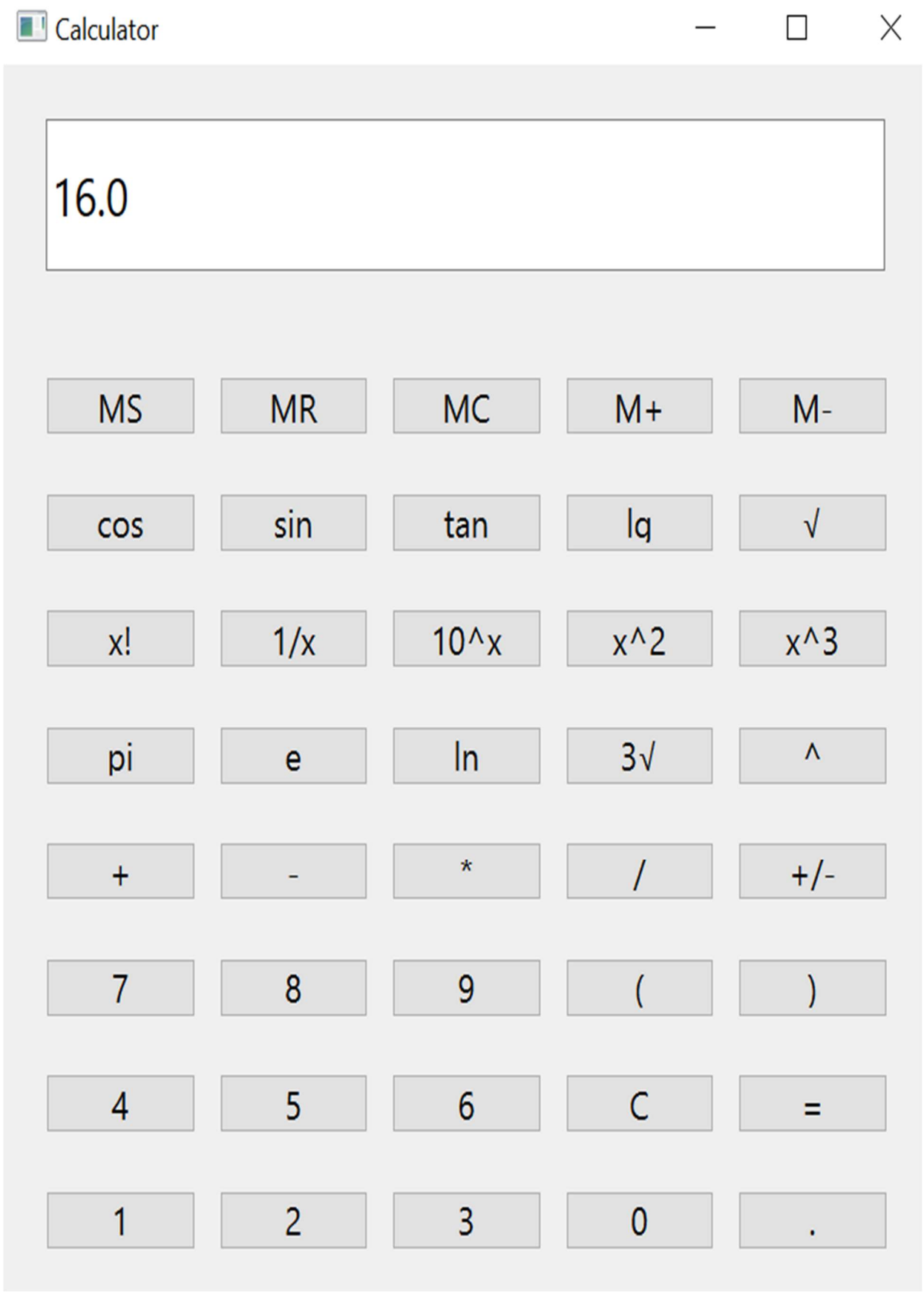

Calculator
0
MS
MR
MC
M+
M-
cos
sin
tan
lg
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.

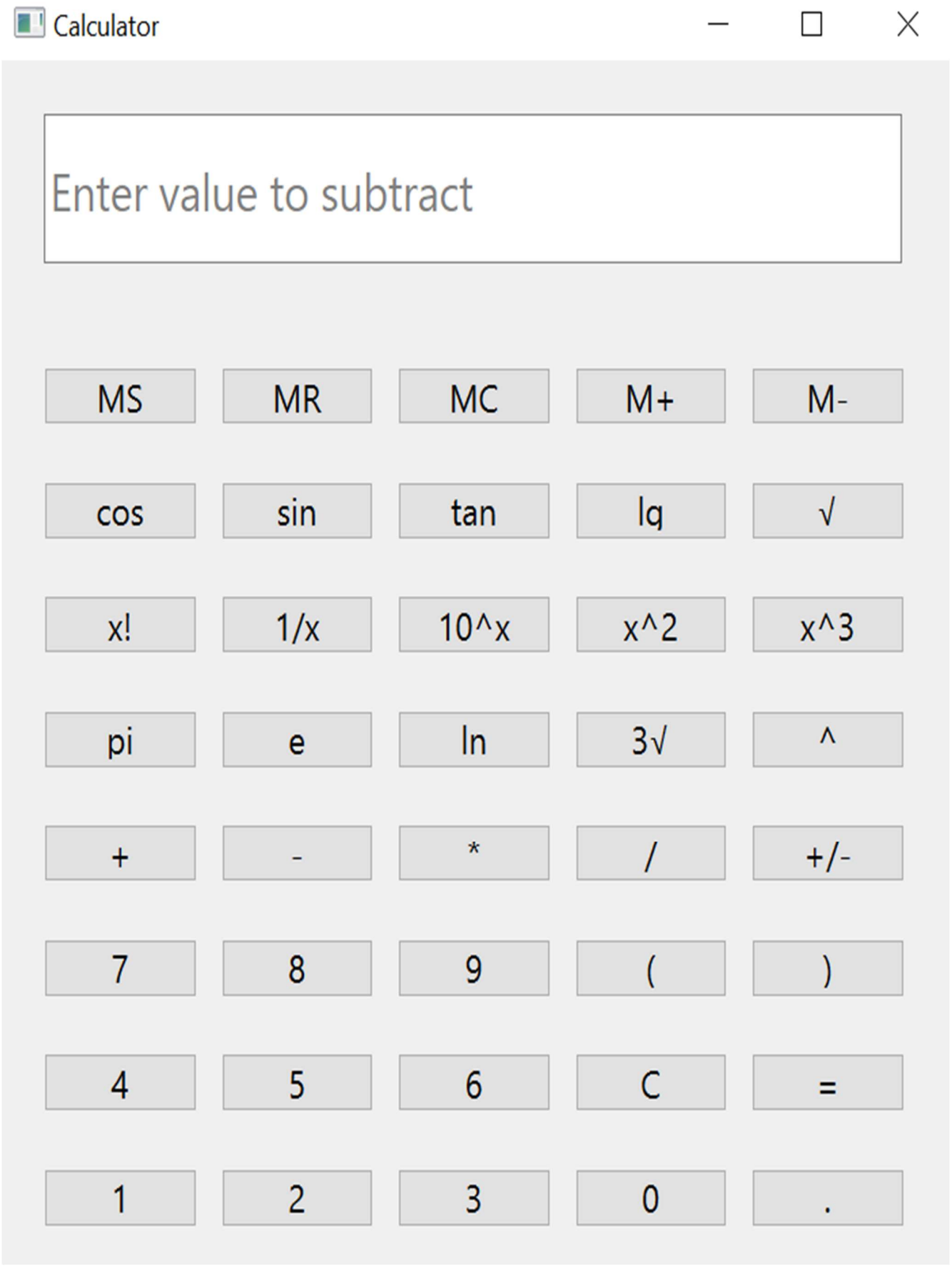

Calculator
8.0
MS
MR
MC
M+
M-
cos
sin
tan
lg
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.

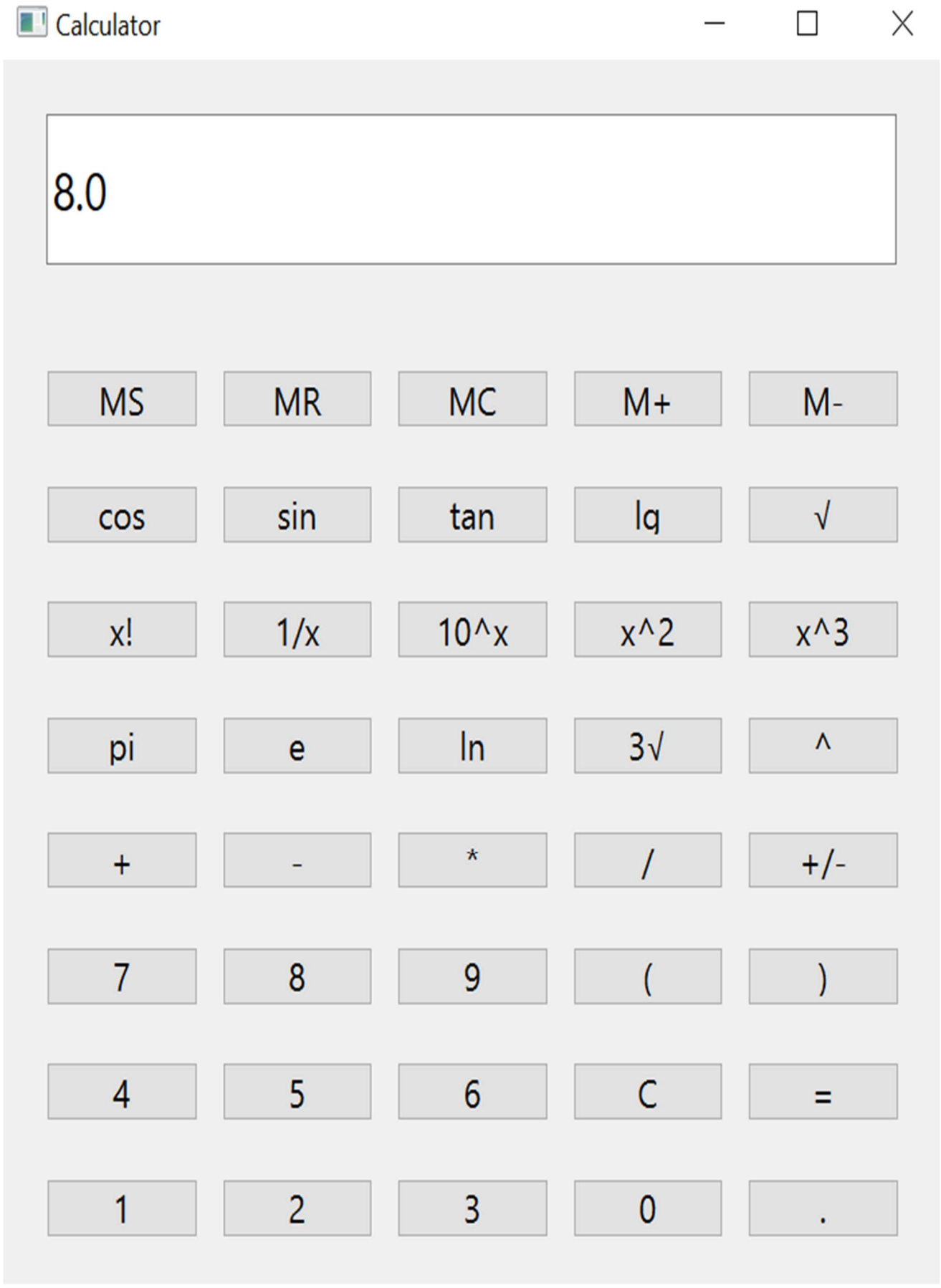
2) **Функції M+ і M- :**


Calculator
Enter value to add
MS
MR
MC
M+
M-
cos
sin
tan
lq
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.


Calculator
8
MS
MR
MC
M+
M-
cos
sin
tan
lg
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.


Calculator
16.0
MS
MR
MC
M+
M-
cos
sin
tan
lg
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.


Calculator
Enter value to subtract
MS
MR
MC
M+
M-
cos
sin
tan
lg
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.


Calculator
8.0
MS
MR
MC
M+
M-
cos
sin
tan
lq
√
x!
1/x
10^x
x^2
x^3
pi
e
ln
3√
^
+
-
*
/
+/-
7
8
9
(
)
4
5
6
C
=
1
2
3
0
.