calculator

By

นายปพน แซ่จ๊ะ 65543206021-9

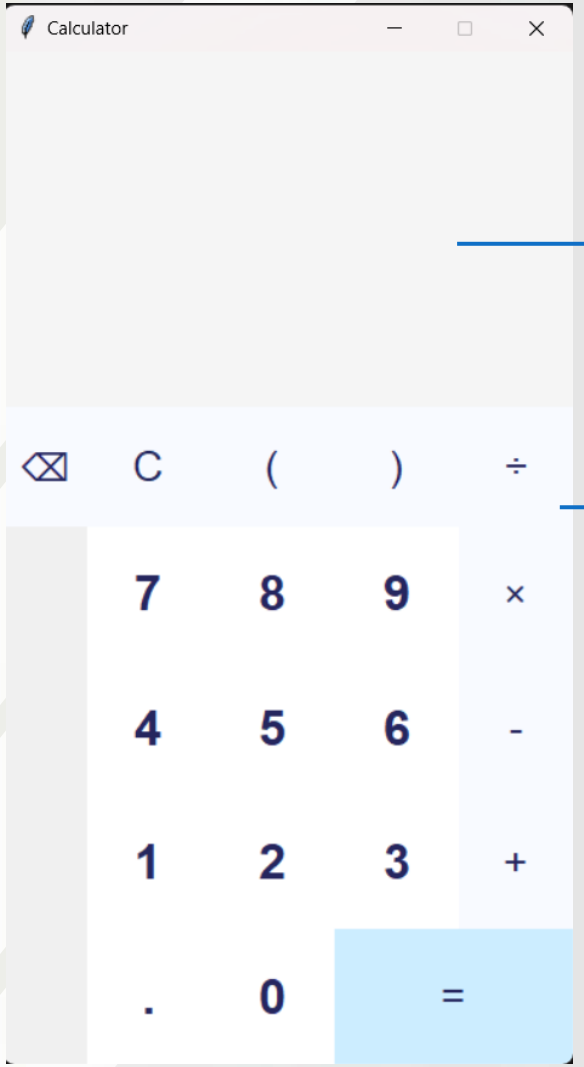
นางสาวเมธาวีย์ วรรณลักษณ์ 65543206076-3

Class
Base Calculator

Advanced
Calculator

Class
Base Calculator

Base Calculator



1.การเริ่มต้น (__init__):

- สร้างหน้าต่างหลักของเครื่องคิดเลขด้วย **tk.Tk()**
- กำหนดตัวแปรสำคัญ:
  - **total_expression:** นิพจน์ทั้งหมดที่จะแสดงผลด้านบน
  - **current_expression:** ค่าหรือตัวเลขปัจจุบันที่กำลังกรอก
  - **is_squaring_mode, last_result, initial_number:** ตัวบ่งชี้สถานะสำหรับการคำนวณขั้นสูง
- สร้างพื้นที่แสดงผลและปุ่มต่าง ๆ

2.พื้นที่แสดงผล:

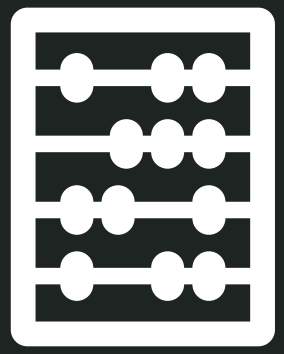
- **create_display_frame:** สร้างเฟรมสำหรับแสดงผลการคำนวณ
- **create_display_labels:** ใช้ **Label** เพื่อแสดงนิพจน์ทั้งหมดและค่าปัจจุบันบนหน้าจอ

3.การสร้างปุ่ม:

- **create_buttons_frame:** จัดเลย์เอาต์กริดสำหรับปุ่มเครื่องคิดเลข
- **create_digit_buttons:** สร้างปุ่มตัวเลข **(0-9)** และจุดทศนิยม
- **create_operator_buttons:** สร้างปุ่มเครื่องหมายบวก ลบ คูณ หาร
- **create_special_buttons:** สร้างปุ่มพิเศษ เช่น ปุ่มลบ ปุ่มวงเล็บ และปุ่มเท่ากับ

4.การจัดการคีย์บอร์ดและการคำนวณ:

- **bind_keys:** เชื่อมปุ่มบนคีย์บอร์ดกับปุ่มใน **GUI** เพื่อให้สามารถใช้งานคีย์บอร์ดได้
- **evaluate:** เมธอดที่รับผิดชอบการคำนวณผลลัพธ์



Advanced Calculator

AdvancedCalculator

```python
def __init__(self):
    super().__init__()
    self.is_continuing = False  # Fla
    self.last_operator = None  # Trac
    self.last_operand = None  # Track
```

```python
def evaluate(self):
    # If we're continuing from a result, use the last result
    if self.is_continuing and self.last_result is not None and
self.last_operator and self.last_operand is not None:
        # Repeat the last operation with the last operand
        expression = f"{self.last_result}{self.last_operator}{self.last_operand}"
    else:
        # First-time evaluation or a fresh input
        expression = self.total_expression + self.current_expression

    # Handle cases like operator at the end of expression
    if expression.endswith(("*", "/", "+", "-")):
        expression = expression[:-1]  # Remove trailing operator
```

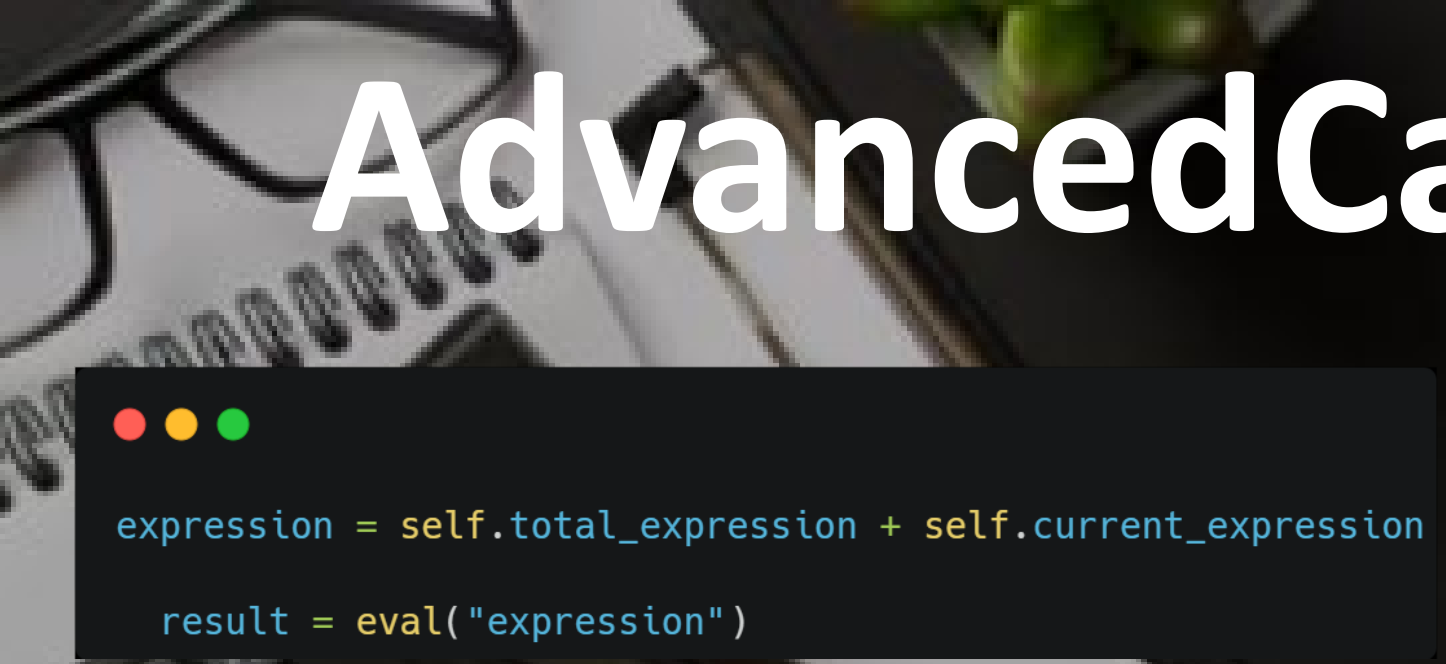
1. Constructor (__init__)

- def __init__(self): ตัวสร้างของคลาส AdvancedCalculator เรียกใช้ super().__init__() เพื่อสืบทอดฟังก์ชันการทำงานจากคลาสแม่ (BaseCalculator)
- มีการเพิ่มตัวแปรเพิ่มเติมเฉพาะคลาส AdvancedCalculator:
  - self.is_continuing: ใช้สำหรับติดตามว่านิพจน์ที่กำลังทำงานอยู่เป็นการดำเนินต่อจากผลลัพธ์ก่อนหน้า (True) หรือไม่
  - self.last_operator: ใช้สำหรับบันทึกเครื่องหมายการคำนวณล่าสุด เช่น +, -, *, /
  - self.last_operand: ใช้สำหรับบันทึกตัวเลขสุดท้ายที่ถูกใช้งานในนิพจน์ (operand)

2. ฟังก์ชันการคำนวณ (evaluate)

- ฟังก์ชัน evaluate(self) เป็นฟังก์ชันหลักในการคำนวณนิพจน์ทางคณิตศาสตร์:
  - หากเป็นการคำนวณต่อจากผลลัพธ์เดิม (เช็คจาก self.is_continuing), จะทำการคำนวณโดยใช้ค่าผลลัพธ์เดิมและเครื่องหมายตัวล่าสุด
  - จัดการนิพจน์ที่ไม่สมบูรณ์ เช่น ลบเครื่องหมายคำนวณท้ายสุดออก และปิดวงเล็บที่เปิดค้างไว้
  - ใช้ eval() เพื่อประมวลผลนิพจน์ และบันทึกผลลัพธ์ลงใน self.current_expression
  - จัดการข้อผิดพลาดที่อาจเกิดขึ้น (เช่น การแบ่งด้วยศูนย์) โดยแสดงข้อความ "Error" หากคำนวณผิดพลาด

AdvancedCalculator



```
expression = self.total_expression + self.current_expression

  result = eval("expression")
```

ป้อนข้อมูลดังนี้ในเครื่องคิดเลข:
1.ผู้ใช้กด 2
2.ผู้ใช้กด +
3.ผู้ใช้กด 3
4.ผู้ใช้กด *
5.ผู้ใช้กด 4
6.ผู้ใช้กด = เพื่อทำการคำนวณ

สถานะของตัวแปรก่อนการกดปุ่ม =
- self.total_expression = "2 + 3" (นี่คือการคำนวณที่เกิดขึ้นก่อนหน้านี้)
- self.current_expression = "* 4" (นี่คือค่าที่ผู้ใช้ป้อนในขณะนี้)

1.expression จะมีค่าเป็น "2 + 3 * 4“
2.result = eval("2 + 3 * 4")
self.current_expression จะถูกตั้งค่าเป็น "14" และแสดงผลที่หน้าจอ

AdvancedCalculator

```python
def append_operator(self, operator):
    # Allow continuing from the last result when an operator is pressed after `=`
    if self.is_continuing:
        self.total_expression = str(self.last_result)
        self.current_expression = ""
        self.is_continuing = False  # Reset continuation flag

    super().append_operator(operator)
```

3. การจัดการการใส่เครื่องหมาย **(append_operator)**

- ฟังก์ชัน **append_operator(self, operator)** ทำหน้าที่เพิ่มเครื่องหมายการคำนวณ เช่น **+, -, *, /** ลงในนิพจน์
- หากกำลังดำเนินการต่อจากผลลัพธ์ก่อนหน้า **(self.is_continuing)**, จะทำการรีเซ็ตนิพจน์ และเริ่มต้นใหม่ด้วยผลลัพธ์ล่าสุด

AdvancedCalculator

```
        try:
            result = eval(expression)
            # Update current expression with the result
            self.current_expression = str(result)
            self.last_result = result

            # Store the last operator and operand for repeated evaluation if not already doing so
            if not self.is_continuing:
                self.last_operator = self.total_expression[-1] if self.total_expression else None
                self.last_operand = self.current_expression

            # Clear total expression for next calculation
            self.total_expression = ""
            self.is_continuing = True

        except Exception:
            self.current_expression = "Error"
            self.last_result = None
            self.last_operator = None
            self.last_operand = None
```

4. การเก็บตัวแปรเพื่อการคำนวณซ้ำ

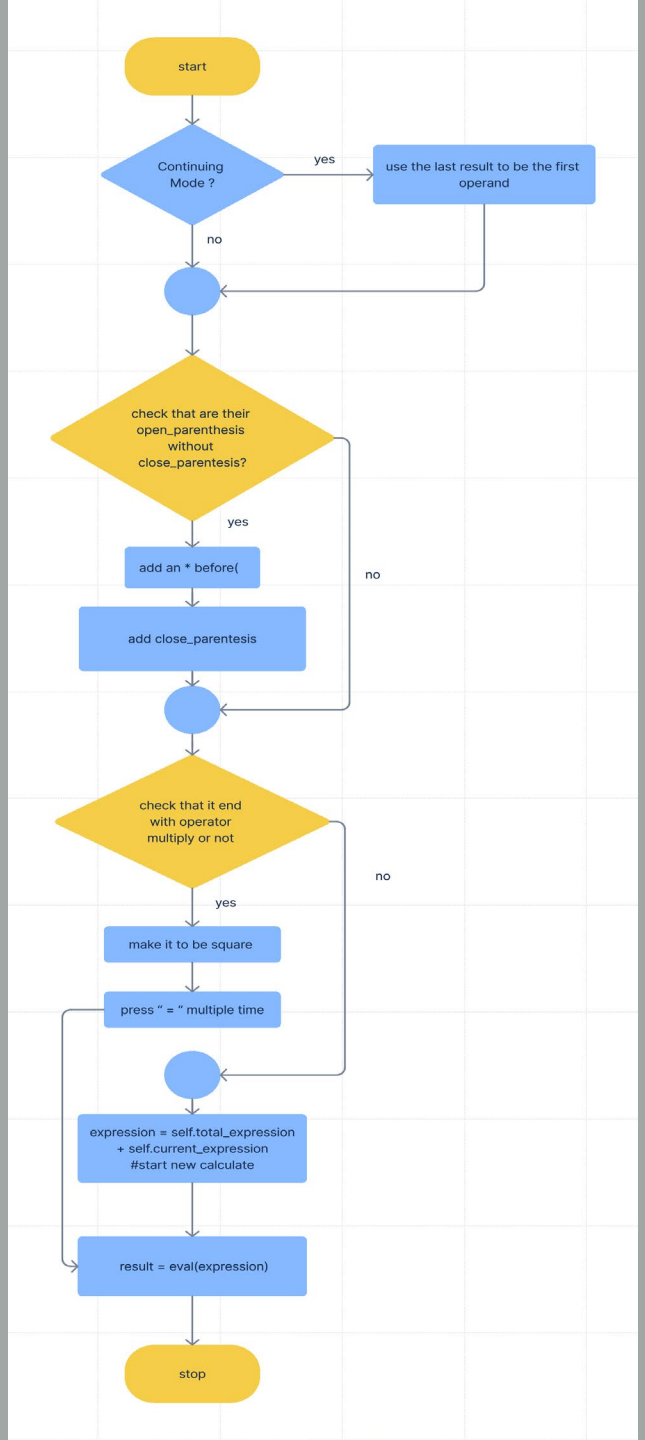
- **self.last_operator:** บันทึกเครื่องหมายการคำนวณล่าสุด เพื่อใช้ในการคำนวณต่อเนื่อง
- **self.last_operand:** เก็บค่าตัวเลขล่าสุดที่ถูกคำนวณ เพื่อสามารถทำการคำนวณซ้ำได้ (เช่น กด = หลายครั้งเพื่อทำการคำนวณซ้ำกับตัวเลขและเครื่องหมายเดิม)
- กรณี **3* =** ยกกำลัง

AdvancedCalculator

```
# Close any open parenthesis
        open_parenthesis_count = expression.count('(')
        close_parenthesis_count = expression.count(')')
        if open_parenthesis_count > close_parenthesis_count:
            expression += ')' * (open_parenthesis_count - close_parenthesis_count)
```

5. **open_parenthesis_count:** นับจำนวนวงเล็บเปิด (ในตัวแปร **expressionclose_parenthesis_count:** นับจำนวนวงเล็บปิด) ในตัวแปร **expression**
ถ้าวงเล็บเปิดมากกว่าวงเล็บปิด โค้ดจะเติมวงเล็บปิด) ให้เท่ากับจำนวนวงเล็บเปิดที่ยังขาดอยู่ เพื่อทำให้วงเล็บสมดุล
* แก้ปัญหาตรงส่วน **A*(B = AB** นั่นเอง

Flowchart


start
Continuing Mode ?
yes
use the last result to be the first operand
no
check that are their open_parenthesis without close_parentesis?
yes
no
add an * before(
add close_parentesis
check that it end with operator multiply or not
yes
no
make it to be square
press " = " multiple time
expression = self.total_expression + self.current_expression #start new calculate
result = eval(expression)
stop

By

นายปพน แซ่จ๊ะ 6554320602l-9

นางสาวเมธาวีย์ วรรณลักษณ์ 65543206076-3