Use Case Description Describes Everything the App

Does Step-By-Step

I. User clicks a number button

> N3. With each number button press add the new value to the end of the first and update entry

II. User clicks a math button

> N1. Make sure entry has a value
>
> N2. Switch boolean values representing math buttons to false on entry
>
> N2. Have Button pass in the math function pressed
>
> N4. Store the entry value on entry to this function (Class Field)
>
> N4. Clear the entry field?

III. User clicks another number button

IV. User clicks equal button and the result shows

> N1. Make sure a math function was clicked
>
> N2. Check which math function was clicked and provide the correct solution

Note 1 : Since every button requires the previous button to have been clicked make sure the click occurred

Note 2 : Make a way to track which math button was clicked last

Note 3 : Think about a way to handle the user entering both single numbers and multiple numbers

Note 4 : Track the first number in the entry box after a math button is clicked

Note 5 : What about division problems caused by an integer division?

a. Convert to float each time we retrieve, or
store values in the entry

CODE

```
from tkinter import *
from tkinter import ttk

class Calculator:

    # Stores the current value to display in the entry
    calc_value = 0.0

    # Will define if this was the last math button clicked
    div_trigger = False
    mult_trigger = False
    add_trigger = False
    sub_trigger = False

    # Called anytime a number button is pressed
    def button_press(self, value):

        # Get the current value in the entry
        entry_val = self.number_entry.get()

        # Put the new value to the right of it
        # If it was 1 and 2 is pressed it is now 12
        # Otherwise the new number goes on the left
        entry_val += value

        # Clear the entry box
        self.number_entry.delete(0, "end")

        # Insert the new value going from left to right
        self.number_entry.insert(0, entry_val)

    # Returns True or False if the string is a float
    def isfloat(self, str_val):
        try:

            # If the string isn't a float float() will throw a
            # ValueError
            float(str_val)

            # If there is a value you want to return use return
            return True
        except ValueError:
            return False
```

```
# Handles logic when math buttons are pressed
def math_button_press(self, value):

    # Only do anything if entry currently contains a number
    if self.isfloat(str(self.number_entry.get())):

        # make false to cancel out previous math button click
        self.add_trigger = False
        self.sub_trigger = False
        self.mult_trigger = False
        self.div_trigger = False

        # Get the value out of the entry box for the calculation
        self.calc_value = float(self.entry_value.get())

        # Set the math button click so when equals is clicked
        # that function knows what calculation to use
        if value == "/":
            print("/ Pressed")
            self.div_trigger = True
        elif value == "*":
            print("* Pressed")
            self.mult_trigger = True
        elif value == "+":
            print("+ Pressed")
            self.add_trigger = True
        else:
            print("- Pressed")
            self.sub_trigger = True

        # Clear the entry box
        self.number_entry.delete(0, "end")

# Performs a mathematical operation by taking the value before
# the math button is clicked and the current value. Then perform
# the right calculation by checking what math button was clicked
# last
def equal_button_press(self):

    # Make sure a math button was clicked
    if self.add_trigger or self.sub_trigger or self.mult_trigger or self.div_trigger:

        if self.add_trigger:
            solution = self.calc_value + float(self.entry_value.get())
        elif self.sub_trigger:
            solution = self.calc_value - float(self.entry_value.get())
        elif self.mult_trigger:
            solution = self.calc_value * float(self.entry_value.get())
        else:
            solution = self.calc_value / float(self.entry_value.get())
            print(self.calc_value, " ", float(self.entry_value.get()),
                                      " ", solution)

            # Clear the entry box
            self.number_entry.delete(0, "end")

            self.number_entry.insert(0, solution)


    def __init__(self, root):
        # Will hold the changing value stored in the entry
        self.entry_value = StringVar(root, value="")

        # Define title for the app
        root.title("Calculator")

        # Defines the width and height of the window
        root.geometry("430x220")

        # Block resizing of Window
        root.resizable(width=False, height=False)

        # Customize the styling for the buttons and entry
        style = ttk.Style()
        style.configure("TButton",
                    font="Serif 15",
                    padding=10)

        style.configure("TEntry",
                    font="Serif 18",
                    padding=10)

        # Create the text entry box
        self.number_entry = ttk.Entry(root,
                    textvariable=self.entry_value, width=50)
        self.number_entry.grid(row=0, columnspan=4)

        # ----- 1st Row -----

        self.button7 = ttk.Button(root, text="7", command=lambda: self.button_press('7')).grid(row=1,
column=0)

        self.button8 = ttk.Button(root, text="8", command=lambda: self.button_press('8')).grid(row=1,
column=1)

        self.button9 = ttk.Button(root, text="9", command=lambda: self.button_press('9')).grid(row=1,
column=2)

        self.button_div = ttk.Button(root, text="/", command=lambda:
self.math_button_press('/')).grid(row=1, column=3)

        # ----- 2nd Row -----
        self.button4 = ttk.Button(root, text="4", command=lambda: self.button_press('4')).grid(row=2, column=0)

        self.button5 = ttk.Button(root, text="5", command=lambda: self.button_press('5')).grid(row=2, column=1)

        self.button6 = ttk.Button(root, text="6", command=lambda: self.button_press('6')).grid(row=2, column=2)

        self.button_mult = ttk.Button(root, text="*", command=lambda: self.math_button_press('*')).grid(row=2, column=3)

        # ----- 3rd Row -----

        self.button1 = ttk.Button(root, text="1", command=lambda: self.button_press('1')).grid(row=3, column=0)

        self.button2 = ttk.Button(root, text="2", command=lambda: self.button_press('2')).grid(row=3, column=1)

        self.button3 = ttk.Button(root, text="3", command=lambda: self.button_press('3')).grid(row=3, column=2)

        self.button_add = ttk.Button(root, text="+", command=lambda: self.math_button_press('+')).grid(row=3, column=3)

        # ----- 4th Row -----

        self.button_clear = ttk.Button(root, text="AC", command=lambda: self.button_press('AC')).grid(row=4, column=0)

        self.button0 = ttk.Button(root, text="0", command=lambda: self.button_press('0')).grid(row=4, column=1)

        self.button_equal = ttk.Button(root, text="=", command=lambda: self.equal_button_press()).grid(row=4, column=2)

        self.button_sub = ttk.Button(root, text="-", command=lambda: self.math_button_press('-')).grid(row=4, column=3)

# Get the root window object
root = Tk()

# Create the calculator
calc = Calculator(root)

# Run the app until exited
root.mainloop()
```