```python
''' tk_calculator_tiny2.py
A updated tiny calculator using the Tkinter GUI toolkit
you can type in functions contained in module math
for instance type in  tan(pi/180)  then click  =
tested with Python27 and Python33  by  vegaseat  13nov2013
'''

# avoid integer division by Python2
from __future__ import division
from math import *
from functools import partial
try:
    # Python2
    import Tkinter as tk
except ImportError:
    # Python3
    import tkinter as tk

class MyApp(tk.Tk):
    def __init__(self):
        # the root will be self
        tk.Tk.__init__(self)
        self.title("Tiny TK Calculator")
        # use width x height + x_offset + y_offset (no spaces!)
        #self.geometry("300x150+150+50")
        # or set x, y position only
        self.geometry("+150+50")
        self.memory = 0
        self.create_widgets()

    def create_widgets(self):
        # this also shows the calculator's button layout
        btn_list = [
        '7',  '8',  '9',  '*',  'C',
        '4',  '5',  '6',  '/',  'M->',
        '1',  '2',  '3',  '-',  '->M',
        '0',  '.',  '=',  '+',  'neg' ]

        rel = 'ridge'
        # create all buttons with a loop
        r = 1
        c = 0
        for b in btn_list:
            # partial takes care of function and argument
            cmd = partial(self.calculate, b)
            tk.Button(self, text=b, width=5, relief=rel,
                command=cmd).grid(row=r, column=c)
            c += 1
            if c > 4:
                c = 0
                r += 1
        # use an Entry widget for an editable display
        self.entry = tk.Entry(self, width=33, bg="yellow")
        self.entry.grid(row=0, column=0, columnspan=5)

    def calculate(self, key):
        if key == '=':
            # guard against the bad guys abusing eval()
            if '_' in self.entry.get():
                self.entry.insert(tk.END, " not accepted!")
            # here comes the calculation part
            try:
                result = eval(self.entry.get())
                self.entry.insert(tk.END, " = " + str(result))
            except:
                self.entry.insert(tk.END, "--> Error!")
        elif key == 'C':
            self.entry.delete(0, tk.END)  # clear entry
        elif key == '->M':
            self.memory = self.entry.get()
            # extract the result
            if '=' in self.memory:
                ix = self.memory.find('=')
                self.memory = self.memory[ix+2:]
            self.title('M=' + self.memory)
        elif key == 'M->':
            if self.memory:
               self.entry.insert(tk.END, self.memory)
        elif key == 'neg':
            if '=' in self.entry.get():
                self.entry.delete(0, tk.END)
            try:
                if self.entry.get()[0] == '-':
                    self.entry.delete(0)
                else:
                    self.entry.insert(0, '-')
            except IndexError:
                pass
        else:
            # previous calculation has been done, clear entry
            if '=' in self.entry.get():
                self.entry.delete(0, tk.END)
            self.entry.insert(tk.END, key)


app = MyApp()
app.mainloop()
```