ПРАКТИЧЕСКАЯ РАБОТА

- Импортируем библиотеку tkinter

```
from tkinter import *
```

- Создаем окно, описываем необходимые параметры

```
# Создание графического интерфейса
root = Tk()
root.title("Калькулятор")
```

- Добавляем поле для ввода или label в котором будет оражаться результат

Создаем виджет Entry, который представляет поле ввода для отображения и ввода чисел и операций. Мы устанавливаем его ширину в 30 символов и толщину границы в 5 пикселей. Затем мы используем метод grid() для размещения этого виджета в первой строке и первом столбце (row=0, column=0) с объединением 3х ячеек (columnspan=3). Если хотите, чтобы в одной строке было 4 кнопки, а значит, и 4 ячейки, то объединяйте 4. Мы также добавляем отступы (padx и pady) для создания пространства между полем ввода и кнопками.

```
entry = Entry(root, width=30, borderwidth=5)
entry.grid(row=0, column=0, columnspan=3, padx=10, pady=10)

# Создание списка кнопок
buttons = [
    "1", "2", "3",
    "4", "5", "6",
    "7", "8", "9",
    "0", "+", "-",
    "*", "/", "=", "Clear"]
```

Прошу заметить, что когда мы используем расположение grid(), размер окна задавать необязательно, размер ячеек, а следовательно, и размер окна корректируется автоматически.

- Прописываем цикл для расстановки кнопок

```
# Создание кнопок с помощью цикла
row = 1
col = 0
for button in buttons:
    if col == 3:
        col = 0
        row += 1

    btn = Button(root, text=button, padx=40, pady=20, command=lambda button=button: button_click_handler(button))
    btn.grid(row=row, column=col)
    col += 1
```

- Мы инициализируем переменные row и col со значениями 1 и 0 соответственно. Эти переменные будут использоваться для определения текущей строки и столбца, в которых будут размещаться кнопки.

- Затем мы начинаем цикл for, который проходит по каждому элементу в списке buttons.

- Внутри цикла мы проверяем значение переменной col. Если оно равно 3, это означает, что мы достигли конца строки, и мы сбрасываем значение col в 0 и увеличиваем значение row на 1. Это позволяет перейти на следующую строку.

- Мы создаем кнопку с помощью метода Button(). Мы используем текущий элемент button из списка buttons в качестве текста кнопки. Мы также устанавливаем размеры кнопки с помощью параметров padx и pady. Каждая кнопка связывается с функцией обработчика button_click_handler() с помощью параметра command. Мы используем лямбда-функцию, чтобы передать текст кнопки в качестве аргумента.

- Затем мы используем метод grid() для размещения кнопки в текущей строке (row) и столбце (col).

- В конце каждой итерации цикла мы увеличиваем значение col на 1, чтобы перейти к следующему столбцу в текущей строке.

Не забываем о обработчике событий в самом конце кода!!!!!!

```
root.mainloop()
```

Как работает lambda-функция, вы можете еще раз прочитать в материале пролого занятия, а вот коротки пример и его описания, работы анонимной функции с кнопками:

```
from tkinter import *

root = Tk()

def print_num(num):
    print(num)

button1 = Button(root, text="Кнопка 1", command=lambda: print_num(1))
button1.pack()

button2 = Button(root, text="Кнопка 2", command=lambda: print_num(2))
button2.pack()

mainloop()
```

В этом примере, лямбда-выражение lambda: print_num(1) связывается с кнопкой button1. Когда кнопка button1 нажимается, вызывается анонимная функция, которая в свою очередь вызывает функцию print_num с аргументом 1.
В этом примере создаются две кнопки с помощью библиотеки tkinter. Каждая кнопка имеет связанную с ней lambda-функцию в качестве обработчика события command. Когда кнопка нажимается, соответствующая lambda-функция вызывается и выводит число в консоль.

Анонимные функции удобны в случаях, когда требуется определить простую функцию, которая будет использоваться только в одном месте и не требует дополнительного именования.

- Необходимо создать функции: для получения текста с кнопки и записываем в поле для ввода/label

```
def button_click(number):
    current = entry.get()
    entry.delete(0, END)
    entry.insert(END, current + str(number))
```

- Когда пользователь нажимает кнопку с цифрой, функция button_click вызывается с аргументом number, который содержит значение цифры, соответствующей нажатой кнопке.

- В первой строке функции мы получаем текущее значение из поля ввода с помощью метода get() для виджета entry. Это значение сохраняется в переменной current.

- Затем мы очищаем содержимое поля ввода с помощью метода delete() и передаем ему аргументы 0 и END. Это удаляет все символы из поля ввода.

- После этого мы используем метод insert() для вставки нового значения в поле ввода. Мы передаем ему аргументы END и current + str(number). END указывает, что новое значение будет вставлено в конец поля ввода. current + str(number) объединяет текущее значение с нажатой цифрой и создает новое значение, которое будет отображено в поле ввода.

Функция button_click выполняет следующие действия:
Получение текущего значения из поля ввода:

- С помощью метода get() мы получаем текущее значение из поля ввода. Например, если в поле ввода уже было введено число "5", то текущее значение будет равно "5". Это значение сохраняется в переменной current.

Очистка содержимого поля ввода:

- С помощью метода delete() мы удаляем все символы из поля ввода. Мы передаем методу delete() два аргумента: 0 и tk.END. 0 указывает на начальную позицию в поле ввода, а tk.END указывает на конечную позицию. Таким образом, все символы в поле ввода будут удалены.

Добавление нажатой цифры в конец текущего значения:

- Мы используем метод insert() для вставки нового значения в поле ввода. Мы передаем ему два аргумента: tk.END и current + str(number). tk.END указывает, что новое значение будет вставлено в конец поля ввода. current + str(number) объединяет текущее значение, которое мы сохранили в переменной current, с нажатой цифрой (number). Например, если текущее значение было "5", а нажата цифра "3", то новое значение будет "53". Это новое значение будет отображено в поле ввода.

```
def button_clear():
    entry.delete(0, END)
```

Это функция обработчика для кнопки "Clear". Она очищает содержимое поля ввода (entry), удаляя все символы.

```
def button_equal():
        result = eval(entry.get())
        entry.delete(0, END)
        entry.insert(END, result)
```

Функция button_equal() выполняет следующие действия:

Получение текущего значения из поля ввода:

- С помощью метода get() мы получаем текущее значение из поля ввода. Это значение будет представлять математический пример, например, "5+3".

Вычисление результата примера:

- Мы используем функцию eval() для вычисления результата примера. Функция eval() принимает строку с математическим выражением и возвращает его результат. Например, если текущее значение в поле ввода равно "5+3", то функция eval() вычислит это выражение и вернет результат, который будет равен 8.

Очистка поля ввода и вставка результата:

- Мы используем метод delete() для очистки содержимого поля ввода. Мы передаем ему аргументы 0 и tk.END, чтобы удалить все символы из поля ввода.
- Затем мы используем метод insert() для вставки результата примера в поле ввода. Мы передаем ему аргументы tk.END и строку, которая объединяет исходный пример и его результат. Например, если пример был "5+3", а результат равен 8, то в поле ввода будет отображено "5+3=8".

Если вам кажется что функции button_click() и button_equal() одинаковые, то это не так. Они отличаются друг от друга и выполняют разные задачи.

- Функция button_click() используется для обработки нажатия кнопок с цифрами. Она добавляет нажатую цифру в конец текущего значения в поле ввода.

- Функция button_equal() используется для обработки нажатия кнопки "равно" и выполнения математического выражения, которое находится в поле ввода. Она вычисляет результат выражения, очищает поле ввода и вставляет результат в поле ввода вместе с исходным выражением.

Таким образом, button_click() используется для добавления цифр в поле ввода, а button_equal() используется для выполнения математических выражений и отображения их результатов.

```python
# Функция для обработки нажатия кнопки
def button_click_handler(button):
    if button == "=":
        button_equal()
    elif button == "Clear":
        button_clear()
    else:
        button_click(button)
```

Это функция обработчика для кнопок. В зависимости от нажатой кнопки она вызывает соответствующую функцию обработчика. Если нажата кнопка "=", вызывается функция button_equal(). Если нажата кнопка "Clear", вызывается функция button_clear(). В противном случае вызывается функция button_click() с передачей текста кнопки в качестве аргумента.