Лабораторная работа № 3

I. Создание первой программы: Калькулятор

Прочитайте текст лабораторной работы, разберите и сделайте все примеры

В Python для настольных приложений есть несколько библиотек создания переносимых графических интерфейсов пользователя (Graphical User Interface, GUI). Например, tkinter, wxPython, PyQt и др. Библиотека tkinter входит в стандартную установку Python. Для работы с ней не надо ничего доставлять, нужно просто подключить модуль tkinter к программе. Программы, написанные на Python с GUI tkinter, переносимы, т.е. должны работать в разных системах (Windows, Unix, Mac OS X). Причем, создается графический интерфейс со свойственным каждой системе видом. Библиотека Tk, на которой основана библиотека tkinter используется также языками сценариев Perl, Ruby, PHP, Common Lisp и Tcl.

Пример 1. Минимальная программа с визуальным интерфейсом tkinter

Рассмотрим простейший пример с tkinter. Откройте Pyzo, создайте в нем файл “pr1.pyw” и наберите в нем код примера 1. Расширение лучше выбирать pyw, вместо py. Тогда при запуске программы не будет появляться окно консоли.

```
from tkinter import *   # подключение библиотеки tkinter
# Создаем функцию без параметров  (обработчик нажатия кнопки):
def hello( ):
    print("На кнопку нажали")
# Создаем метку и прикрепляем ее к верху формы (окно программы):
Label(text = "Hello World").pack(side=TOP)
# Если явно в программе не создавать главное окно, то библиотека tkinter создаст
# его самостоятельно.
# Создаем кнопку, подключаем обработчик и прикрепляем ее к низу формы:
Button(text = "Hello event World", command=hello).pack(side=BOTTOM)
# Запускаем цикл обработки сообщений:
mainloop( )
```

У двух компонент в этом примере выбран минимальный набор свойств: надписи text и обработчик command для кнопки. Метод pack размещает компоненты на форме.

При создании компонент можно задавать больше параметров (цвет, шрифт, отступы, граница и т.п.). Эти параметры можно менять в дальнейшем. Через параметр command можно подключить только один обработчик без параметров. Более сложные обработчики с параметрами подключаются иначе.

Пример 2. Создание главного окна, задание свойств виджетов

В следующем примере явно создадим главную форму, а также у компонент (их еще называют виджетами) установим некоторые свойства.

```
from tkinter import * # подключение библиотеки tkinter

# Метод Tk создает главное окно, ссылку сохраним в переменную root:
root = Tk( )
# по умолчанию размер окна будет выбран по его содержимому,
# положение окна выбирает операционная система

# Можно задать размеры и положение окна самостоятельно (W x H + X + Y):
root.geometry("400x200+150+100")
# Можно задать минимальные и максимальные размеры
# root.minsize(width= 150, height = 150)
# root.maxsize(width= 500, height = 500)

# Заголовок окна
root.title("Hello world!!!")

# Свойства виджета можно задавать
# (1) при создании,
# (2) обращаясь как к словарю (виджет['свойство']=...) или
# (3) с помощью метода config:
# задаем при создании (1) text и цвет фона bg = background:
w = Label(root, text = "Hello config world", bg = 'black')
# первый параметр root – виджет, на котором будет размещена метка.
w['fg'] = 'yellow'    # (2) задаем цвет текста fg = foreground
w.config(font = ('times', 20, 'bold')) # (3) задаем шрифт
# задаем высоту height (в строках), ширину width (в символах),
# ширину db = borderwidth и стиль бордюра relief:
w.config(height = 3, width = 20, bd = 8, relief = RAISED)
# задаем вид курсора мыши над виджетом:
w.config(cursor = "cross")

# размещаем компонент с помощью метода pack:
w.pack(side = TOP, expand = YES, fill = BOTH)
# – прижать к верхнему краю (можно не указывать, side = TOP по умолчанию)
#  – растягивать виджет при изменении размеров окна expand = YES
#  по двум направлениям fill = BOTH (можно еще = X или = Y)
```

```
# Вывод главного окна и запуск цикла обработки сообщений:
root.mainloop( )
```

Пример 3. Задание событий

В следующем примере создадим 3 кнопки и подключим к ним более сложные обработчики с параметрами. У первых двух кнопок будем использовать свойство command (как в примере 1) для подключения обработчиков, а к третьей кнопке подключим обработчик с помощью более универсального метода bind. Метод bind способен связывать с виджетом разнообразные обработчики событий с параметрами, не только нажатие на кнопку.

```
from tkinter import *

root = Tk( )   # – создаем главное окно
root.title("Привет мир!!!")  #  – заголовок окна

# Создаем метку и сохраняем ссылку lab1 на нее,
# чтобы в дальнейшем обращаться к ней по этому имени:
lab1 = Label(root, text = "Текст на метке",  bg = '#F5F5F5', fg = '#0000FF',
                font = ('times', 20, 'bold'))
# Размещаем метку на форме, прижимаем ее к верхнему краю.
# Метка будет растягиваться вместе с формой
lab1.pack(side = TOP, expand = YES, fill = BOTH)

# Создаем функцию с параметром x – обработчик нажатия кнопки 1 и 2.
# Связывание через свойство command:
def oncommand(x):
    print("Нажали на кнопку", x)

# Создаем функцию с параметром event – обработчик нажатия кнопки 3.
# Связывание через bind, в event будут передаваться ОС параметры события:
def onclick(event):
    # печатаем имя виджета и координаты мыши при щелчке:
    print("Нажали на кнопку %s X = %s Y = %s" % (event.widget, event.x, event.y))
    # при каждом нажатии будем увеличивать padx и pady – внешние границы
    x = int(str(but3["padx"]))
    x += 1
    but3.config(padx = x, pady = x)

# Создаем две кнопки, задаем свойство command:
```

```
but1 = Button(root, text = "кнопка 1", command = (lambda: oncommand(1)))
but1.pack( )
but2 = Button(root, text = "кнопка 2", command = (lambda: oncommand(2)))
but2.pack( )
# Т. к. command можно присваивать только функции без параметров, создаем
# lambda-функцию без параметров (обертку), внутри которой вызывается функция
# с параметрами. Параметр передаем 1 и 2 для первой и второй кнопок,
# соответственно.

# Создаем третью кнопку:
but3 = Button(root)
but3.config(text = "кнопка 3")  # – устанавливаем текст на кнопке
# Назначаем реакцию на событие <Button-1> (нажатие левой кнопки мыши)
# – функцию onclick с одним параметром (используется операционной системой
# для передачи данных в программу):
but3.bind("<Button-1>", onclick)
but3.pack( )
root.mainloop( )
```

Пример 4. Виджет Frame и метод размещения компонент pack

Рассмотрим еще один виджет Frame. Он служит для разделения области окна программы на отдельные прямоугольные участи и размещения на них других виджетов (аналог панели в C# и Delphi). Разберем также подробней метод размещения компонент pack.

```
from tkinter import *
root = Tk( )  # создаем главное окно

# Создаем один frame:
win1 = Frame(root, bg = '#555555')

# Фрейм будет растягиваться при изменении размеров главного окна:
win1.pack(expand = YES, fill = BOTH)

# Расположим на нем последовательно 4 метки.
# Метки прижаты к разным краям виджета.
Label(win1, text = "TOP 1", width=10, height=4, bg='yellow').pack(side=TOP)
Label(win1, text = "LEFT 2", width=10, height=4, bg='red').pack(side=LEFT)
Label(win1, text = "BOTTOM 3", width=10, height=4, bg='green').pack(side=BOTTOM)
Label(win1, text = "RIGHT 4", width=10, height=4, bg='white').pack(side=RIGHT)
```

```
# Метод pack размещает 1-ю метку и прижимает ее вверх (занимает всю кромку).
# При размещении 2-й метки используется оставшаяся область и метка
# прижимается влево и т. д. размещаются 3-я и 4-я метки.
# При изменении размеров окна сохраняется принцип: место под компоненты
# выделяется в порядке их размещения. Если  сжимать окно, то компоненты
# будут пропадать в порядке обратном их размещению.

# Создаем второй frame:
win2 = Frame(root, bg='#770077')
# Фрейм прижимается вверх (по умолчанию) растягивается по оси X:
win2.pack(expand = YES, fill = X)
# Расположим на нем метку и редактор.
# Метка прижата влево и добавляем ей отступы padx и pady:
Label(win2, text = "x =").pack(side = LEFT, padx = 10, pady = 10)
# Редактор тоже с отступами и растягивается по оси X:
Entry(win2).pack(side = LEFT, padx = 10, pady = 10, expand = YES, fill = X)
root.mainloop( )
```

Подробней о методе размещения компонент pack можно посмотреть в списке литературы указанном в конце. Например, ссылки на электронные страницы https://younglinux.info/tkinter/pack или https://metanit.com/python/tutorial/9.4.php

Пример 5. Напишем пример простой «Калькулятор»

```
from tkinter import *   # подключаем tkinter
from tkinter.messagebox import * # подключаем диалоговые окна tkinter

root = Tk( )  # создаем главное окно

# Устанавливаем минимальные и максимальные размеры окна:
root.minsize(width = 350, height = 150)
root.maxsize(width = 500, height = 300)
root.title("Калькулятор") # заголовок окна

# Создадим 3 фрейма: fr_xy, fr_op и fr_res для размещения компонент.
fr_xy = Frame(root) # фрейм fr_xy  (компоненты для ввода чисел x, y).
fr_xy.pack(side = TOP, expand = YES, fill = X)

# На нем размещаем две метки и два редактора для ввода чисел x, y:
lx = Label(fr_xy, text = "x = ")
lx.pack(side = LEFT, padx = 10, pady = 10)
```

```
entX = Entry(fr_xy)
entX.insert(0, 0) # – в редактор записываем в позицию 0 число 0
entX.pack(side = LEFT, padx=10, pady=10)
entX.focus( ) # – редактор будет выбран при старте (иметь фокус ввода)

ly = Label(fr_xy, text = "y = ")
ly.pack(side=LEFT, padx=10, pady=10)
entY = Entry(fr_xy)
entY.insert(0, 0)
entY.pack(side = LEFT, padx=10, pady=10)

# Создание фрейма с заголовком fr_op  (выбор операции):
fr_op = LabelFrame(root, text = "Операция:")
fr_op.pack(side = TOP, expand=YES, fill=X)

# Операцию будем выбирать с помощью виджета Radiobutton:
oper = ['+', '-', '*', '/']   # – список операций

# Вводим строковую переменную tkinter, ее свяжем с выбором Radiobutton
varOper = StringVar( )
# В цикле создаем 4 кнопки Radiobutton (связываем их с переменной):
for op in oper:
      Radiobutton(fr_op, text = op, variable = varOper, value = op).pack(side = LEFT,
                        padx = 20, pady = 10)
varOper.set(oper[0]) # Устанавливаем текущее значение ‘+’

# Создаем 3-й фрейм fr_res (вычисление значения и вывод результата):
fr_res = Frame(root)
fr_res.pack(side = TOP, expand = YES, fill = BOTH)

# Обработчик кнопки:
def OnButtunResult( ):
  # Защищенный блок, будем пытаться перевести текст из редактора в число:
    try:
        x = float(entX.get())  # извлекаем число из 1-го редактора
    except ValueError: # если не получилось, выдаем сообщение и выходим
        showerror("Ошибка заполнения", "Переменная x не является числом")
        return
  # Защищенный блок 2:
```

```
    try:
        y = float(entY.get())
    except ValueError:
        showerror("Ошибка заполнения", "Переменная y не является числом")
        return
  # В переменную op записываем выбранную операцию:
    op = varOper.get( )
  # Вычисляем:
    if op == '+':   res = x + y
    elif op == '-': res = x - y
    elif op == '*': res = x * y
    elif op == '/':
        if y != 0: res = x / y
        else: res = 'NAN'
    else:
        res = 'операция выбрана неправильно'
  # Вывод результата на метку:
    lres['text'] = res
# Обработчик кнопки закончился.

# Создаем кнопку и метку, к кнопке присоединяем обработчик:
Button(fr_res, text = "=", width = 10, command = OnButtunResult).pack(side = LEFT,
            padx = 30, pady = 20)
lres = Label(fr_res, text = "")
lres.pack(side = LEFT, padx = 30, pady = 20)

# Запуск цикла обработки сообщений:
root.mainloop( )
```

II. Задание

На основе созданных программ выполните следующие задания. Данные для заданий должны браться из редакторов Entry. Результаты работы выводится на метки Label. Для каждого задания создавайте новое приложение.

Перед выполнением заданий разберите и сделайте все примеры.

Замечание. Более подробную информацию о tkinter см. [1, 4–12].

В примере 5 «Калькулятор»:

1. Поменять цвета фона фреймов и текста на них, изменить курсор при наведении на какой-то из фреймов.
2. Добавить операции «%» и «//» в Radiobutton и в обработчик.
3. Добавить операцию «корень квадратный из x» и «квадрат y» в Radiobutton и в обработчик.

В заданиях 4–7 пользователь в редакторах задает значения x, y, z, по щелчку на кнопке считываются эти значения, и программа на метку выдает ответ.

4. Вычислите значение функции

$$f(x) = \frac{\sin(\pi(x^2 - x - 2))}{\cos(\pi x) + 1}.$$

   Найдите область определения функции, в случае если начальные данные не входят в ООП, выдайте сообщение об этом.

5. Вычислите значение функции

$$g(x, y) = \sqrt[4]{(y^2 + y - 2)/(x^2 - 1)}.$$

   Найдите область определения функции, в случае если начальные данные не входят в ООП, выдайте сообщение об этом.

6. Составьте программу, которая вводит три целых числа x, y, z и выводит на экран значение $\max(x, y, z)$. Напишите свою функцию поиска максимума.
7. Составьте программу, которая вводит три целых числа x, y, z и выводит на экран значение $\text{sign}(x) \cdot \min(y, z)$.

В следующих заданиях поменяйте у виджетов стандартные свойства: цвет, шрифт, отступы и т. п.

8. Решить квадратное уравнение. Пользователь в редакторах задает коэффициенты квадратного уравнения a, b, c. По щелчку на кнопке считываются эти значения (в защищенных блоках) и программа выдает количество вещественных корней и их значения.

9. Решить биквадратное уравнение. Пользователь в редакторах задает коэффициенты биквадратного уравнения a, b, c. По щелчку на кнопке считываются эти значения (в защищенных блоках) и программа выдает количество вещественных корней и их значения.

---

10. Составьте программу, которая вводит натуральное число n и выводит на экран все его делители.
11. Составьте программу, которая находит наибольший общий делитель двух натуральных чисел m и n.

---

12. Используя методы строк выяснить, является ли данный текст десятичной записью целого числа. Не используя защищенные блоки. Алгоритм проверки оформить в виде функции. Учесть, что число может быть со знаком “–5” или “+4” и удалить начальные и завершающие пробелы.
13. Найти самое короткое слово в тексте. Если таких слов несколько, то найти последнее. Алгоритм поиска оформить в виде функции.

---

14. Элементы массива A – целые числа. Напишите функцию, находящую максимальное число среди отрицательных чисел массива. Если отрицательных чисел нет, выдать сообщение об этом (для поиска максимума напишите функцию).
15. Элементы массива A – целые числа. Напишите функцию, находящую максимальное число среди положительных чисел массива. Если положительных чисел нет, выдать сообщение об этом (для поиска максимума напишите функцию).

---

16. **(Виджет флажок Checkbutton)** Создайте программу, которая опрашивает пользователя, результат выдает на метку. В программе используйте виджет Checkbutton для ответов на вопросы с двумя возможными вариантами. Например, спрашивается ФИО, пол, есть ли уже 18 лет, нравится ли зеленый цвет и согласны ли Вы на обработку Ваших персональных данных.
    Замечание. Флажки Checkbutton см. [1, стр. 602, 599 (переменные)] и электронные ресурсы.
17. **(Виджет ползунок Scale)** Создайте программу, которая опрашивает пользователя, результат выдает на метку. В программе используйте виджет Scale для ответов на вопросы с выбором целочисленного значения. Например, спрашивается ФИО, курс, группа.
    Замечание. Ползунки Scales см. [1, стр. 614, 599 (переменные)] и электронные ресурсы.

Литература

[1] **Лутц М. Программирование на Python, том I, 4-е издание. Пер.с англ. СПб.: Символ-Плюс, 2011. 992 с.**

[2] Лутц М. Изучаем Python, 4-е издание. Пер.с англ. СПб.: Символ-Плюс, 2011. 1280 с.

[3] Лутц М. Программирование на Python, том II, 4-е издание. Пер.с англ. СПб.: Символ-Плюс, 2011. 992 с.

Интернет-ресурсы

[4] https://www.python.org, https://docs.python.org/3/ (Documentation Python)

[5] https://docs.python.org/3/library/tkinter.html#module-tkinter (tkinter doc)

[6] http://www.pythonware.com/library/, http://effbot.org/tkinterbook/

[7] https://www.tcl.tk/, https://www.tcl.tk/man/tcl8.6/TkCmd/contents.htm (Tk Documentation)

[8] https://tkdocs.com/tutorial/index.html (TkDocs)

Интернет-ресурсы
(русскоязычные электронные учебники)

[9] https://younglinux.info/tkinter.php (Tkinter. Программирование GUI на Python. Курс)

[10] https://metanit.com/python/tutorial/9.1.php (Глава 9 электронного учебника по Python. «Создание графического интерфейса»)

[11] https://ru.wikiversity.org/wiki/Курс_по_библиотеке_Tkinter_языка_Python#Entry (Курс по библиотеке Tkinter языка Python)

[12] https://pythonru.com/uroki/obuchenie-python-gui-uroki-po-tkinter#toc (Обучение Python GUI (уроки по Tkinter))