

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«КАЛИНИНГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Л. Г. Высоцкий

**ВЫСОКОУРОВНЕВЫЕ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ  
(ВТП)**

Учебно-методическое пособие  
по выполнению контрольных работ  
для студентов, обучающихся в бакалавриате по направлению  
09.03.03 Прикладная информатика (заочное отделение)

Калининград  
Изд-во ФГБОУ ВО «КГТУ»  
2022

УДК 004.9(075)

Рецензент:

кандидат педагогических наук, доцент кафедры прикладной информатики  
ФГБОУ ВО «Калининградский государственный технический  
университет» Е. Ю. Заболотнова

**Высоцкий, Л. Г.**

Высокоуровневые технологии программирования (ВТП): учебно-методическое пособие по выполнению контрольных работ для студентов, обучающихся в бакалавриате по направлению подготовки 09.03.03 Прикладная информатика (заочное отделение) /**Л. Г. Высоцкий.**– Калининград: Изд-во ФГБОУ ВО «КГТУ», 2022. – 99 с.

Учебно-методическое пособие включает материалы по двум контрольным работам, которые должны быть выполнены студентами при прохождении курса «Высокоуровневые технологии программирования». Все работы выполняются по индивидуальным заданиям. В пособие также включены подробные указания по оформлению отчетов по работам.

Учебно-методическое пособие рассмотрено и одобрено в качестве локального электронного методического материала кафедрой прикладной информатики института цифровых технологий ФГБОУ ВО «Калининградский государственный технический университет» 19 сентября 2022 г., протокол № 3

Учебно-методическое пособие по изучению дисциплины рекомендовано к использованию в качестве локального электронного методического материала в учебном процессе методической комиссией ИЦТ 20 сентября 2022 г., протокол № 6

УДК 004.9 (075)

© Федеральное государственное бюджетное образовательное учреждение высшего образования «Калининградский государственный технический университет», 2022 г.  
© Высоцкий Л. Г., 2022 г.

## Оглавление

Введение.....	4
Контрольная работа №1.....	6
Задание № 1 Основные элементы пользовательского интерфейса .....	6
Задание № 2 Разработка меню.....	36
Контрольная работа №2.....	44
Задание №3 Разработка усложненного Python-проекта.....	44
Задание №4 Управление временем в Python-проектах .....	66
Задание № 5 Графика на основе канвы .....	82
Критерии и нормы оценки контрольной работы.....	95
СПИСОК РЕКОМЕНДУЕМЫХ ИСТОЧНИКОВ:.....	96
Приложение .....	98

## Введение

Основной целью выполнения студентами заочной формы обучения контрольных работ по дисциплине «Высокоуровневые технологии программирования» является закрепление на практике, углубление и обобщение знаний, полученных студентами во время изучения дисциплины «Высокоуровневые технологии программирования», а также определение места данной дисциплины в формировании будущей специальности.

Контрольные работы позволяют:

- систематизировать, закрепить и расширить теоретические знания и практические навыки, полученные на лекциях и при самостоятельном изучении дисциплины, применить их при решении конкретной задачи, связанной с будущей специальностью;
- развить навыки самостоятельной работы;
- повысить уровень подготовки по специальным разделам (алгоритмизация, кодирование, отладка) дисциплины «Высокоуровневые технологии программирования».

Лабораторный практикум должен подготовить студентов к самостоятельному применению высокоуровневых технологий программирования и в дальнейшем использовать их при выполнении курсовой и подготовке выпускной квалификационной работ. Дополнительными задачами являются:

- приобретение навыков оформления печатной работы в соответствии с предъявленными требованиями для использования этих навыков при подготовке будущей квалификационной работы;
- приобретение навыков в процессе защит лабораторных работ отстаивания своей точки зрения, обоснования выбираемых решений.

Данные контрольные работы ориентированы, прежде всего, на освоение студентами практических навыков формирования графического пользовательского интерфейса программным путем средствами модуля tkinter языка Python. Именно с конструирования данного интерфейса начинается создание большинства современных больших программных систем. А далее уже к нему привязывается весь функционал, реализуемый системой в соответствии с требованиями.

В результате выполнения контрольных работ студент должен расширить свои знания в:

- области средств и возможностей языка Python;
- типовых элементов графического пользовательского интерфейса (GUI);

- стандартов оформления технической документации;
- повысить свои умения в:
- систематизации и конкретизации теоретических знаний в вопросах создания программных систем;
- поиске и анализе информации, необходимой по темам контрольных работ;
- использовании средств и функциональных возможностей языка реализации программных приложений по каждой контрольной работе;
- разработке и тестировании программных приложений, включающих графический пользовательский интерфейс;
- разработке структуры сложного программного приложения;
- формулировании выводов и предложений;
- приобрести навыки:
- самостоятельного решения возникающих вопросов, возникающих в ходе создания сложной программной системы;
- системного мышления через определение целей и постановку задачи каждой контрольной работы;
- практического использования теоретического материала по программированию, освоенного за предыдущее время обучения;
- оформления программной документации в соответствии со стандартами;
- самостоятельного освоения дополнительного технического материала по теме работы.

Учебное пособие содержит материалы по двум контрольным работам. Первая выполняется в третьем семестре обучения и включает два задания, вторая – в четвертом семестре и включает три задания. Каждая работа оформляется под единым титульным листом, представленным в **Приложении**. Каждое содержание контрольной работы заканчивается списком используемой литературы. Требования к оформлению каждого отдельного задания в рамках единой контрольной работы приводятся в конце описания задания.

## Контрольная работа № 1

### Задание № 1 Основные элементы пользовательского интерфейса

#### *Теоретический раздел*

В настоящее время почти все программные приложения, которые создаются для конечного пользователя, используют графический интерфейс (graphical user interface – GUI).

Основные компоненты графического интерфейса пользователя, проектируемого в среде программирования Python, содержит библиотека Tk, написанные на языке программирования Tcl. Tkinter – это пакет для Python'a, предназначенный для работы с библиотекой Tk в составе модуля tkinter. Не вдаваясь в подробности, Tkinter можно охарактеризовать как переводчик с языка Python на язык Tcl. Программа пишется на языке Python, а код модуля tkinter после этого переводит её инструкции на язык Tcl, который понимает библиотека Tk.

Tkinter импортируется в программное приложение стандартно для модуля Python любым из способов:

- `import tkinter,`
- `from tkinter import *`,
- `import tkinter as tk.`

Далее, чтобы написать программу на базе GUI, надо выполнить приблизительно следующие шаги:

1. Создать главное окно.
2. Создать виджеты и выполнить конфигурацию их свойств (опций).
3. Расположить виджеты в главном окне
4. Определить события, на которые будет реагировать программа.
5. Определить обработчики событий, т. е. функционал приложения (как будет реагировать программа).
6. Запустить цикл обработки событий.

Шаблон листинга для создания GUI имеет вид:

```
from tkinter import *  
root = Tk()  
.....
```

`root.mainloop()`

`Tk()` - это функция `tkinter`, открывающая главное окно любого приложения.

Метод `mainloop` циклически получает информацию о событиях от системы окна и отправляет их обработчикам приложения.

Объект окна верхнего уровня создается от класса `Tk` модуля `tkinter`. Переменную, связываемую с этим объектом, обычно называют `root` (корень), хотя программист может дать ей и любое другое имя:

```
root = Tk()
```

Размер окна можно задать через его свойство:

```
root.geometry("400x220")
```

Запрещает реконфигурацию окна команда:

```
root.resizable(0,0) или root.resizable(width=False, height=False).
```

В этом случае из заголовка исчезают кнопки реконфигурирования окна.

С помощью команды

```
<окно>.minsize(width=<размер>, height=<размер>)
```

можно устанавливать минимальный размер окна, а командой `maxsize` - максимальный. Если они равны, то окно будет постоянного размера, т. е. его нельзя изменить во время работы программы.

Текст заголовка окна можно задать командой:

```
root.title("GUI на Python")
```

Закрывается окно командой:

```
root.destroy()
```

Формирование графического пользовательского интерфейса сводится к выбору виджетов, которые представляют собой объекты уже преопределенных в Python'е классов, и параметризации свойств этих объектов. При создании виджета его класс обязательно указывается с заглавной буквы. Ниже рассмотрены характеристики наиболее часто используемых при создании GUI виджетов.

## **button (кнопка)**

`button` - простая кнопка для вызова некоторых действий (выполнения определенных команд).

Если в ходе работы программы не предусматривается никаких обращений к кнопке, то можно создавать кнопку без имени (идентификатора) по следующему синтаксису

**Batton**([<родитель>], [{свойство}])

А далее она размещается на пространстве формы с помощью некоторого менеджера разметки, например, `pack()`:

**Batton**([<родитель>][{, свойство}]). `pack()`

Если предполагается, что в ходе работы программы будет иметь место обращение к кнопке (например, параметры кнопки будут модифицироваться), то при создании кнопка должна быть поименована, т.е. она создается по синтаксису:

`<имя> = Batton`([<родитель>][{, свойство}])

При этом допускаются и русские имена кнопки. Ниже приводится фрагмент листинга, реализующий размещение на форме кнопки.

```
from tkinter import *
import time

root = Tk()
root.geometry('800x600')

б =Button()
б.place(x = 65, y = 100)

root.mainloop()
```

В этом примере расположение кнопки задается в абсолютных координатах окна, т. е. при таком варианте создания кнопки можно использовать менеджер геометрии `place()`.

К основным параметрам кнопки относятся:

- **bg/background**: фоновый цвет кнопки;
- **fg/foreground**: цвет текста кнопки;
- **font**: шрифт текста, например, `font="Arial 14"` - шрифт Arial высотой 14px, или `font=("Verdana", 13, "bold")` - шрифт Verdana высотой 13px с выделением жирным;

- **height**: высота кнопки (число строк);
- **width**: ширина кнопки (в символах);
- **justify**: устанавливает выравнивание текста. Значение **LEFT** выравнивает текст по левому краю, **CENTER** - по центру, **RIGHT** - по правому краю;
- **state**: устанавливает состояние кнопки, может принимать значения **DISABLED** (кнопка деактивирована), **ACTIVE**, **NORMAL** (по умолчанию);
- **text**: устанавливает текст на кнопке;
- **image (bitmap)** - ссылка на изображение, которое отображается на кнопке.

Свойства виджета задаются по имени, поэтому могут указываться в произвольном порядке:

```
b =Button(text = 'Пример', bg = 'red', fg = 'black')
```

или

```
b =Button(fg = 'black', bg = 'red', text = 'Пример')
```

Цвет можно задавать тремя способами:

- именем (156 цветов, указанных в табл. 1.1 – `named_colors`);
- номером цвета в шестнадцатеричном формате (например, "#555");
- смешивание трех основных цветов (красного, зеленого, синего). Каждая составляющая принимает значения в диапазоне 0 ÷ 255:

```
mycolor1 = "#%02X%02X%02X" % (200, 200, 20)
```

```
foreground=mycolor1
```

(X – число в шестнадцатеричной системе счисления, 2 – число позиций для каждого цвета в шестнадцатеричной форме, 0 – признак системы, отличной от десятичной).

Кнопка исходно находится в состоянии **NORMAL**, при нажатии она переходит в состояние **ACTIVE**, при отпускании снова возвращается в состояние **NORMAL**, т. е. можно программно анализировать её состояние. Но кнопку можно устанавливать и в статическое состояние **DISABLED**, т. е. она будет видима, но недоступна для нажатия (использования).

Пример создания изображения на кнопке:

```
img = PhotoImage(file = "car.png")
```

```
l = Label(image = img, fg = "#555", bg = "#ffffff", text = 'Метка')
```

Изображение должно предварительно быть загруженным в объект **PhotoImage**.

Таблица 1.1

black	k	dimgray	dimgray
gray	grey	darkgray	darkgrey
silver	lightgray	lightgrey	gainsboro
whitesmoke	w	white	snow
rosybrown	lightcoral	indianred	brown
firebrick	maroon	darkred	r
red	mistyrose	salmon	tomato
darksalmon	coral	orangered	lightsalmon
sienna	seashell	chocolate	saddlebrown
sandybrown	peachpuff	peru	linen
bisque	darkorange	burlywood	antiquewhite
tan	navajowhite	blanchedalmond	papayawhip
moccasin	orange	wheat	oldlace
floralwhite	darkgoldenrod	goldenrod	cornsilk
gold	lemonchiffon	khaki	palegoldenrod
darkkhaki	ivory	beige	lightyellow
lightgoldenrodyellow	olive	y	yellow
olivedrab	yellowgreen	darkolivegreen	greenyellow
chartreuse	lawngreen	honeydew	darkseagreen
palegreen	lightgreen	forestgreen	limegreen
darkgreen	g	green	lime
seagreen	mediumseagreen	springgreen	mintcream
mediumspringgreen	mediumaquamarine	aquamarine	turquoise
lightseagreen	mediumturquoise	azure	lightcyan
paleturquoise	darkslategray	darkslategrey	teal
darkcyan	c	aqua	cyan
darkturquoise	cadetblue	powderblue	lightblue
deepskyblue	skyblue	lightskyblue	steelblue
aliceblue	dodgerblue	lightslategray	lightslategray
slategray	slategrey	lightsteelblue	cornflowerblue
royalblue	ghostwhite	lavender	midnightblue
navy	darkblue	mediumblue	b
blue	slateblue	darkslateblue	mediumslateblue
mediumpurple	rebeccapurple	blueviolet	indigo
darkorchid	darkviolet	mediumorchid	thisle
plum	violet	purple	darkmagenta
m	fuchsia	magenta	orchid
mediumvioletred	deeppink	hotpink	lavenderblush
palevioletred	crimson	pink	lightpink

Кнопку можно временно сделать невидимой с помощью команды (для менеджера разметки place):

```
<имя_кнопки>.place_forget()
```

А чтобы вернуть видимость кнопке, ее нужно снова разместить командой:

```
<имя_кнопки>.place(<координаты>)
```

В процессе работы программы параметры кнопки можно менять двумя способами:

1) указанием модифицируемого свойства в квадратных скобках

```
б['text'] = "Изменено"
```

```
б['bg'] = '#000000'
```

2) использованием метода config, что позволяет модифицировать сразу несколько свойств

```
б.config(text = "Изменено")
```

Обычно кнопки используются для запуска/останова определенных действий, реализуемых функциями. С другой стороны, у любого виджета, в отличие от языков визуального программирования, нет предопределенного обработчика событий, как и нет свя-

занных с ним событий. Поэтому сначала создается обработчик, потом он привязывается к событию, а событие к виджету, т. е. один и тот же обработчик может быть привязан к нескольким виджетам. Различают два вида привязки:

1-й вариант привязки - через его свойство кнопки 'command':

```
<кнопка>(..., command = <обработчик-вызываемая функция>).pack()
```

Такой вариант может использоваться в обоих вариантах создания кнопки (без имени и с именем).

2-й вариант привязки - использование метода bind(). Структура привязки имеет вид:

```
<кнопка >.bind(<событие>, <обработчик-вызываемая функция>)
```

Пример такой привязки:

```
b = Button(fg = "#555", bg = "#ffffff", text = 'Пуск')
b.place(x = 50, y = 100)
```

```
b.bind('<Button-1>', push)
```

В данном случае клик левой клавиши мыши приводит к запуску процедуры push, созданной в тексте программы.

## **label (надпись)**

Этот виджет предназначен для отображения какой-либо надписи без возможности редактирования её пользователем с клавиатуры. Может также включать графическое изображение.

Стандартный вариант создания:

```
l1 = Label(text = 'Метка', bg = 'gainsboro', fg = 'yellow', font="Arial 16", width=6,
          height=1)
l1.place(x = 150, y = 250),
```

т. е. большинство свойств метки аналогичны свойствам кнопки. Ниже перечислены свойства, присущие только метке:

- **justify:** устанавливает выравнивание текста относительно метки. Значение LEFT выравнивает текст по левому краю, CENTER - по центру, RIGHT - по правому краю
- **relief:** определяет тип границы метки, по умолчанию значение FLAT, т. е. метка сливается фоном формы. Другие значения свойства:
  - RAISED
  - SUNKEN
  - GROOVE

- RIDGE
- **wrplength**: при положительном значении (соответствует логическому значению TRUE) текст при достижении правой границы метки будет переноситься на следующую строку.

### **entry (поле ввода)**

Горизонтальное поле (однострочный редактор), в которое можно ввести или отобразить строку текста. Большинство основных свойств редактора аналогично свойствам предыдущих виджетов. К специфическим для редактора свойствам относятся:

- **bd()** – ширина границы в пикселях;
- **show()** – символ-заменитель для сокрытия вводимой в редактор информации.

Класс **Entry** включает широкий набор методов, позволяющих оперировать занесенной в него информацией:

- **get()** - возвращает текст виджета;
- **delete(first, last=None)** - очищает поле ввода от *first* и **до** *last* позиции. Если указан только 1-й параметр, то удаляется один символ на указанной позиции. Для удаления до конца можно указать в качестве 2-го параметра значение **END**. Индексация значений начинается с нуля;
- **icursor(index)** - перемещает курсор на указанную позицию в редакторе;
- **focus()** - устанавливает курсор (фокус) в редактор, т. е. устанавливает связь редактора с клавиатурой компьютера;
- **index(index)** - возвращает текущую позицию курсора в редакторе;
- **insert(index, string)** - вставляет текст в поле редактора перед указанной позицией;

### **text (форматированный текст)**

Этот прямоугольный виджет позволяет редактировать и форматировать текст с использованием различных стилей, внедрять в текст рисунки и даже окна.

По умолчанию размер виджета равен 80 знакоместам по горизонтали и 24-м по вертикали. Однако эти значения можно изменять с помощью свойств **width** и **height**. Есть также возможность конфигурировать шрифт, цвета и другие параметры текста.

Значение **WORD** свойства **wrap** позволяет переносить слова на новую строку целиком, а не по буквам. (**wrap = CHAR**) – режим по умолчанию. **NONE** – запрет переноса, пока не будет нажата клавиша **Enter**.

Основные методы у виджета **Text** такие же, как у виджета **Entry**: **get()**, **insert()** и **delete()**. Однако если в случае однострочного текстового поля было достаточно указать один индекс элемента при вставке или удалении, то в случае многострочного надо указывать два – номер строки и номер символа в этой строке (другими словами, номер столбца). При этом *нумерация строк начинается с единицы, а столбцов – с нуля*.

```
text.insert(1.0, s)
```

Методы **get()** и **delete()** могут принимать не два, а один аргумент. В этом случае будет обрабатываться только один символ в указанной позиции.

Особенностью текстового поля библиотеки **Тк** является возможность форматировать в нём текст, т.е. придавать его разным частям разное оформление. Делается это с помощью методов **tag\_add()** и **tag\_config()**. Первый из них добавляет тег, при этом надо указать его произвольное имя и отрезок текста, к которому он будет применяться. Метод **tag\_config()** настраивает тегу стили оформления. Пример:

```
text.tag_add('title', 1.0, '1.end')
```

```
text.tag_config('title', foreground="red", background = "yellow", font=("Verdana",  
24, 'bold'), justify=CENTER)
```

При этом для цветов текста и фона надо использовать зарезервированные слова **foreground** (вместо **fg**) и **background** (вместо **bg**).

С помощью метода **search** можно находить программно требуемое содержимое в редакторе. При этом надо задавать координаты начала и конца фрагмента текста, в котором реализуется поиск:

```
...  
start = 1.0  
pos = text.search("o", start, stopindex="end")  
if not pos:  
    root.title("Ничего не найдено")  
else:  
    root.title(pos)
```

Таже можно конкретные позиции в редакторе фиксировать с помощью меток:

```
text.mark_set("here", '1.8')
```

В дальнейшем вместо позиции 1.8 в программе можно использовать ее метку "here".

Установка курсора на определенную позицию производится следующими методами:

```
...
def Ust(event):
    text.mark_set(INSERT, "2.5")

b1 = Button(text = "Установка")

b1.bind("<1>", Ust)
```

Для обнаружения текущей позиции курсора в редакторе используется тот же метод с соответствующим параметром INSERT ("insert"):

```
<позиция курсора> = <редактор>.index('insert')
```

```
...
def Ust(event):
    p = text.index(INSERT)
    root.title(p)

b1 = Button(text = "Установка")

b1.bind("<1>", Ust)
```

Также можно программно считывать выделенный в редакторе текст:

```
...
def Copy(event):
    s = text.selection_get()
    root.title(s)

b1 = Button(text = "Копировать")

b1.bind("<1>", Copy)
```

При выделении можно зафиксировать координаты начала и конца выделения:

```
...
def Copy(event):
    s = text.index(SEL_FIRST)
    root.title(s)
```

```
b1 = Button(text = "Копировать")
```

```
b1.bind("<1>", Copy)
```

```
...
```

Аналогично определяются координаты и конца выделения.

В виджет `Text` можно вставлять другие виджеты с помощью метода `window_creat()`. Ниже приведен пример вставки в текст метки, на которой размещен "смайлик". Метка расположится в конце введенного в окно текста.

```
label = Label(text=":)", bg="yellow")
```

```
text.window_create(INSERT, window=label)
```

Если в текстовое поле вводится больше строк текста, чем его высота, то оно само будет прокручиваться вниз. При просмотре прокручивать вверх-вниз можно с помощью колеса мыши и стрелками на клавиатуре. Однако бывает удобнее пользоваться скроллером – полосой прокрутки (`ScrollBar`). Её можно создать как отдельный виджет и привязать к компоненту `text`. Однако существует отдельный вариант этого виджета с уже привязанной вертикальной линейкой прокрутки. Он является объектом класса `ScrolledText`. Поэтому этот класс предварительно надо импортировать

```
from tkinter import scrolledtext,
```

а далее создается окно этого класса с указанием его параметров:

```
text = scrolledtext.ScrolledText(root, width=40, height=10
```

### **checkboxbutton (флажок)**

Данный виджет умеет переключаться между двумя состояниями при нажатии на него мышью, т. е. используется для бинарного выбора.

У каждого флажка должна быть своя переменная `Tkinter`, которая может принимать только два значения, соответствующие включенному и выключенному состояниям флажка. Она может быть логического (`BooleanVar`) или целого (`IntVar`) типа. В первом случае переменная может принимать значения `True` или `False`. Во втором случае соответствующими значениями являются 1 и 0. Какому значению переменной соответствует установка или сброс флажка, задается параметрами `onvalue` и `offvalue`. С помощью опции `onvalue` устанавливается значение, которое принимает связанная переменная при включенном флажке, с помощью свойства `offvalue` – при выключенном.

```
c1 = Checkbutton(text="Выбор", variable=cvar1, onvalue=1, offvalue=0)
```

Параметр **text** представляет комментарий, поясняющий назначение флажка.

С помощью методов **select()** и **deselect()** флажок можно программно включать и выключать. Например:

```
c1.deselect() или c1.select()
```

### **radiobutton (селекторная кнопка)**

Данная кнопка используется для представления одного из альтернативных значений. Такие кнопки, как правило, действует в группе, т. е. они не создаются в приложении по одной. При нажатии на одну из них кнопка группы, выбранная ранее, "отскакивает".

Объединение кнопок в одну группу устанавливается через общую переменную, разные значения которой соответствуют включению разных радиокнопок группы. У всех кнопок одной группы свойство **variable** устанавливается в одно и то же значение – связанную с группой переменную. А свойству **value** присваиваются разные значения этой переменной.

```
r_var = BooleanVar()
r_var.set(0)

r1 = Radiobutton(text='Первая', variable=r_var, value=0)
r2 = Radiobutton(text='Вторая', variable=r_var, value=1)
```

В данном случае объединяет обе кнопки в одну группу логическая переменная **r\_var**, которая будет принимать значение 0 или 1.

В программном коде обычно требуется "снять" данные о том, какая из двух кнопок включена. Делается это с помощью метода **get()** для связывающей переменной Tkinter, например:

```
vybor = r_var.get()
```

### **listbox (список)**

От класса **Listbox** создаются списки – виджеты, внутри которых в столбик перечисляются опции. При этом можно выбирать один или несколько элементов списка. Заполняется **Listbox** с помощью метода **insert()**.

```
lbox = Listbox(width=15, height=7)
lbox.place(x = 70, y = 50)
```

```
for i in ('один', 'два', 'три', 'четыре', 'пять', 'шесть', 'семь'):
    lbox.insert(0, i)
```

Можно изменить порядок заполнения списка командой

```
lbox.insert(END, i)
```

По умолчанию в `Listbox`, кликая мышкой, можно выбирать только один элемент. Вариант выбора задается с помощью атрибута `selectmode = (SINGLE/BROWSE/MULTIPLE/EXTENDED)`. По умолчанию устанавливается вариант одиночного выбора `BROWSE`. Такой же выбор также задается вариантом `SINGLE`. Два других варианта используют множественный выбор.

Если для `Listbox` необходим скроллер, то он настраивается так же, как для текстового поля, т.е. в программу добавляется виджет `Scrollbar`, который связывается с экземпляром `Listbox`.

С помощью метода `get()` из списка можно получить один элемент (текст опции) по индексу, или срез, если указать два индекса.

```
a = lbox.get(1)
```

Метод `delete()` удаляет один элемент списка или сразу срез уже в ходе работы программы, а с помощью методов `insert(<позиция>, <текст опции>)` этот список может пополняться.

Метод `curselection()` позволяет получить в виде кортежа индексы выбранных опций экземпляра `Listbox`. При одиночном выборе кортеж хранит индекс только одной опции, причем в 0-м элементе кортежа.

Если для считывания установленного значения (значений) списка используется событие, не связанное с самим списком, то приложение работает корректно. Сложнее ситуация, если используется событие, связанное с самим `листбоксом`.

```
lbox.bind('<Button-1>', Opr)
```

В этом случае имеет место запаздывание на один шаг с фиксацией выделенной опции. Для корректной работы виджета требуется двойной клик мыши по опции или вариант события вида

```
lbox.bind('<Double-Button-1>', Opr)
```

Список опций можно сохранять в виде отдельного текстового файла или формировать его на основе содержимого такого файла:

```
def saveList():
```

```
f = open('list000.txt', 'w')
f.writelines("\n".join(lbox.get(0, END)))
f.close()
```

### **scale (шкала)**

Виджет используется для задания числового значения путем перемещения движка в определенном диапазоне. Свойства:

- **orient** - как расположена шкала на окне. Возможные значения: HORIZONTAL, VERTICAL (горизонтально, вертикально).
- **length** - длина шкалы.
- **from\_** - с какого значения начинается шкала.
- **to** - каким значением заканчивается шкала.
- **tickinterval** - интервал, через который отображаются метки шкалы.
- **resolution** - шаг передвижения (минимальная длина, на которую можно передвинуть движок).

Пример создания шкалы:

```
s1 = Scale(root, orient = HORIZONTAL, length = 200, from_ = 50, to = 80, tickinterval = 5, resolution=5)
```

Текущее значение линейки можно получить посредством метода **get()**:

```
a = s1.get()
```

#### *Постановка задачи*

Создать программное приложение в соответствии с заданием, отладить его, предоставить преподавателю возможность проверки корректности выполнения задания, оформить отчет по выполненному заданию.

#### *Методика (алгоритм) выполнения задания*

I. Из раздела «Варианты задания» в соответствии с вариантом (номером в списке группы: первая цифра-номер группы, далее идет номер студента в группе) выбрать задание на разработку программного проекта:

II. В «Теоретическом разделе» изучить необходимые средства модуля tkinter для выполнения определенного вариантом задания.

III. Разработать графический интерфейс средствами языка Python для выполнения определенного вариантом задания.

IV. Разработать программные модули на языке Python для реализации требуемого в задании функционала.

V. Привязать разработанный функционал к программному интерфейсу.

VI. Произвести отладку полученного программного приложения.

VII. Предоставить преподавателю возможность проверки корректности выполнения задания в дистанционном или очном варианте.

VIII. Оформить отчет по выполненному заданию, содержащий:

- 1) задание;
- 2) структуру проекта;
- 3) блок-схему начальной установки приложения, блок-схемы процедур.
- 4) листинг программы.

#### *Варианты задания*

1-1. На форме располагаются компоненты: на середине формы метка **Label** с указанием фамилии студента; **Listbox** с опциями-цифрами 1, 2, 3, 4; четыре кнопки со стрелками ←, ↑, →, ↓; редактор **Entry**; **Checkbutton** и кнопка «Закреть». При нажатии мышью кнопки со стрелкой метка с фамилией смещается в соответствующем направлении на количество пикселей, указанных в **Listbox**. Текущий шаг перемещения отображается в редакторе **Entry**, если он виден. Режим его видимости задается флагом **Checkbutton**.

1-2. На форме располагаются компоненты: на середине формы метка **Label** с указанием номера группы; **Listbox** с опциями ←, ↑, →, ↓; радиогруппа для выбора шага перемещения на 1, 2 или 3 пикселя; **Scale**, задающая размер шрифта на метке в диапазоне 8-16 пикселей; кнопка «Сдвиг» и кнопка «Закреть». При выборе мышью соответствующей опции с направлением и нажатии кнопки «Сдвиг» форма смещается в соответствующем направлении на заданное число пикселей.

- 1-3. На форме располагаются компоненты: четыре кнопки с номерами, редактор **Entry**, метка **Label**, флажок **Checkbutton**, кнопки «Очистить» и «Заккрыть». Исходно на метке пусто. Нажатие кнопки с номером приводит к появлению в окне редактора или на метке ее номера, добавляемого в конец строки. Переключение вывода производится флажком **Checkbutton**. Кнопка «Очистить» приводит окно редактора и поверхность метки в исходное состояние.
- 1-4. На форме располагаются компоненты: метка **Label**, четыре кнопки с номерами, **Scale** с разметкой от 1 до 4 и кнопка «Заккрыть». При нажатии мышью какой-либо кнопки она становится невидимой, но появляется ранее скрытая кнопка. Номер невидимой кнопки появляется на метке. Синхронно перемещается ползунок линейки **Scale**. Перемещение ползунка мышью также приводит к вышеописанным манипуляциям с кнопками.
- 1-5. На форме располагаются компоненты: редактор **Entry**, кнопки + и -, список **Listbox** с опциями "Показать" и "Скрыть", кнопка "Очистить" и кнопка **Close**. Исходно в редакторе находится число 20. Нажатие кнопки + приводит к увеличению содержимого редактора на 1, а кнопки -, к уменьшению на 1. При выборе опции "Скрыть" кнопки исчезают и управление переходит к клавишам + и - клавиатуры. Опция "Показать" возвращает проект в исходное состояние. Нажатие кнопки «Очистить» приводит к установке редактора в исходное состояние. Кнопка **Close** заканчивает программу.
- 1-6. На форме располагаются компоненты: редактор **Text** размером 10x30, занимающий центр окна; четыре кнопки ←, ↑, →, ↓; линейка **Scale** с опциями 1, 2, 3 и кнопка **Close**. Нажатие кнопки со стрелкой приводит к перемещению правой или нижней границы окна редактора в соответствующем направлении на количество символов, заданных в компоненте **ComboBox**. При этом в окне редактора отображается его текущий размер. Если редактор уменьшается до размера текста, то кнопка уменьшения для соответствующего направления исчезает. При увеличении данного размера она снова появляется. Кнопка **Close** заканчивает программу.
- 1-7. На форме располагаются компоненты: редактор **Text** размером 20x20, занимающий центр окна; компонент **Listbox** с четырьмя опциями ←, ↑, →, ↓; радионабор с вариантами "Левая и верхняя" и "Правая и нижняя"; кнопка «Пуск». Нажатие последней

приводит к перемещению на 1 символ/строку (правый нижний угол – редактор меняет размер) или 1 пиксель (левый верхний угол – редактор меняет положение) в соответствии с опцией, указанной в компоненте **Listbox**. Выбор дублируется в редакторе **Text**. Если опция не выбрана, то в окне редактора выводится соответствующее сообщение.

1-8. На форме располагаются: редактор **Entry**, занимающий центр окна; компонент **Listbox** с четырьмя опциями  $\leftarrow$ ,  $\uparrow$ ,  $\rightarrow$ ,  $\downarrow$ ; линейка **Scale**, используемая для задания пяти разных цветов, и кнопка «Пуск». Нажатие этой кнопки приводит к отображению в окне редактора слов «Влево», «Вправо», «Вверх», «Вниз» в соответствии с опцией, указанной в компоненте **Listbox**, и перемещению редактора на один пиксель в соответствующем направлении. Цвет надписи задается компонентом **Scale**. Если опция направления не выбрана, то в окне редактора выводится соответствующее сообщение.

1-9. На форме располагаются компоненты: четыре метки **Label** с названиями цветов, список **Listbox** с опциями-цветами, **Radiogroup** с теми же цветами, флажок **Checkbutton**, кнопка «Выбор» и кнопка «Закрыть». При выборе названия цветка и нажатии мышью кнопки «Выбор» или клавиши "Enter" на клавиатуре соответствующая метка исчезает, а скрытая появляется. Вариант выбора: через список или радиогруппу задается флажком. Кнопка **Close** заканчивает программу.

1-10. На форме располагаются компоненты: восемь меток **Label** с названиями городов, список **Listbox** с теми же городами-опциями, линейка **Scale** для изменения количества городов, доступных для выбора, в диапазоне от 4 до 8, кнопка «Пуск» и кнопка «Закрыть». Исходно все метки-города скрыты. При выборе названия города в **Listbox** и нажатии мышью кнопки «Пуск» соответствующая метка появляется, а ранее видимая исчезает. Кнопка **Close** заканчивает программу.

1-11. На форме располагаются: семь разных кнопок с надписями - номерами или названиями дней недели; список **Listbox**, задающий количество доступных кнопок ( $4 \div 7$ ); радиогруппа из двух компонентов; кнопка **Close** и редактор **Entry**. Нажатие каждой кнопки приводит к ее сокрытию и показу ранее скрытой кнопки. При этом номер

(название) ранее скрытой кнопки отображается в окне редактора. Вид надписи на кнопках задается радиогруппой. Кнопка **Close** заканчивает программу.

1-12. На форме располагаются: шесть разных кнопок с номерами-надписями, линейка **Scale** с разметкой от 1 до 6, кнопка **Close**, редакторы **Entry** и **Text**, флажок **Checkbutton**. Движение ползунка линейки приводит к последовательному сокрытию соответствующих кнопок и показу ранее скрытой кнопки. При этом номер ранее скрытой кнопки добавляется к строке в окне редактора. Вид текущего редактора устанавливается флажком **CheckBox**. Кнопка **Close** заканчивает программу.

1-13. На форме располагаются: семь редакторов **Entry** с вписанными названиями дней недели, кнопка **Close**, список **Listbox** с названиями шести цветов и метка. Нажатие одной из клавиш клавиатуры в диапазоне  $1 \div 7$  приводит сокрытию соответствующего редактора и отображению соответствующего дня недели на метке, а также показу ранее скрытого редактора. При этом текст в нем отображается цветом, заданным списком **Listbox**. Кнопка **Close** заканчивает программу.

1-14. На форме располагаются: кнопка с надписью "Да", кнопка **Close**, линейка **Scale** с разметкой от 0 до 30, флажок **Checkbutton**, редакторы **Entry** длиной 30 символов и **Text** размером 40x10. В редакторе **Entry** вводится строка символов. Нажатие кнопки "Да" приводит к дублированию содержимого данного редактора в окне **Text** и показу, через дефис, числа символов в строке **Entry**. Если установлен флажок, то ползунок линейки устанавливается в позицию, равную длине строки. Кнопка **Close** заканчивает программу.

1-15. На форме располагаются: две метки для вывода размера формы (или положения формы на экране) и две метки с соответствующими подписями «Ширина» и «Высота» (или X и Y); четыре кнопки  $\leftarrow$ ,  $\uparrow$ ,  $\rightarrow$ ,  $\downarrow$ ; радиопереключатель на два положения; редактор **Entry** и кнопка **Close**. Нажатие кнопок со стрелками приводит к соответствующему изменению размера формы или перемещению формы по пространству экрана, что отображается на панелях. Вид манипуляции с формой задается радиопереключателем и отображается в редакторе. Кнопка **Close** заканчивает программу.

- 1-16. На форме располагаются: редактор **Text**; две метки для вывода размера редактора и две метки с соответствующими подписями «Ширина» и «Высота»; четыре кнопки ←, ↑, →, ↓; список **Listbox** с опциями "Левый верхний" и "Правый нижний" и кнопка **Close**. Нажатие кнопок со стрелками приводит к соответствующему изменению размера редактора через соответствующее перемещение угла, что отображается на панелях. Размер редактора находится в диапазоне от 10x40 до 25x100. При достижении граничного значения соответствующая кнопка исчезает. Кнопка **Close** заканчивает программу.
- 1-17. На форме располагаются компоненты: редактор **Entry**, занимающий центр окна, у которого размер задан в пикселях; компонент **Listbox** с четырьмя опциями ←, ↑, →, ↓; радионабор с двумя опциями: "Форма" и "Редактор", - и кнопка «Пуск». Нажатие этой кнопки приводит к изменению размера окна редактора или формы на 1 пиксель в направлении опции, указанной в компоненте **Listbox**. Объект изменения задается компонентом **Combobox**. Текущий размер редактора отображается в самом редакторе, а формы – в ее заголовке. Если опция не выбрана, то в окне редактора выводится соответствующее сообщение.
- 1-18. На форме располагаются: редактор **Text**; линейка **Scale** с разметкой от 10 до 20; метка с названием группы; три кнопки с надписями «Влево», «Вправо», «Центр»; радиогруппа с аналогичными функциями и кнопка **Close**. Нажатие одной из кнопок или переключение радионабора приводит к изменению выравнивания надписи на метке. Название режима выравнивания дублируется в редакторе **Text**. Линейка **Scale** задает размер шрифта надписи. Кнопка **Close** заканчивает программу.
- 1-19. На форме располагаются: семь кнопок с номерами и кнопка **Close**. Нажатие кнопки с номером приводит к тому, что рядом с кнопкой появляется подсказка в виде названия соответствующего дня недели, а у других кнопок она исчезает. Подсказки появляются в окнах редакторов **Entry** или на метках, вариант реализации меняется флажком **Checkbutton**. Список **Listbox** позволяет задавать цвет подсказки (9 вариантов). Кнопка **Close** заканчивает программу.
- 1-20. На форме располагаются компоненты: два редактора **Entry**, **Label**; радиогруппа из двух опций "Левый" и "Правый"; список **Listbox** с опциями "Русские буквы", "Ла-

тинские буквы", "Цифры", "Знаки препинания"; кнопка "Копирование" и кнопка Close. В редакторы могут вводиться строки из букв, цифр, знаков препинания. При нажатии кнопки "Копирование" соответствующие символы, задаваемые списком Listbox, копируются на Label. Источник копирования указывается радиогруппой. Кнопка Close заканчивает программу.

1-21. На форме располагаются: семь кнопок с названиями-днями недели; каждой кнопке соответствует флажок, переключающий русские/английские названия дней недели; список Listbox с названиями дней и кнопка Close. Выбор дня в списке Listbox приводит к сокрытию флажка данного дня и отображению другого скрытого флажка. Кнопка Close заканчивает программу

1-22. На форме располагаются компоненты: редакторы Entry и Text, Label, Listbox, кнопка "Да" и кнопка Close. При запуске программы курсор находится в редакторе Entry. В него можно вводить только буквы, преобразуемые к верхнему регистру. При нажатии кнопки "Да" введенная строка копируется в Text или на Label. Приемник копирования выбирается с помощью Listbox. После четырех попыток ввести цифры редактор Entry деактивируется. Кнопка Close заканчивает программу.

1-23. На форме располагаются компоненты: два списка Listbox с подписями А и В, метка Label, редактор Entry, кнопка типа Button и кнопка Close. Каждый компонент Listbox содержит шесть разных чисел. Их выбор и нажатие кнопки Button приводят к отображению на метке результата  $A \otimes B$ . Редактор Entry, задает тип операции (+, -, \*), Линейка Scale определяет цвет результата (не менее 6 цветов). Кнопка Close заканчивает программу.

1-24. На форме располагаются компоненты: два компонента Listbox с названиями ИМЕНА, которые содержат женские и мужские имена, два компонента ComboBox с названиями ФАМИЛИИ, содержащими женские и мужские фамилии; радиогруппа с опциями "Мужчины" и "Женщины"; метка Label, кнопка Button и кнопка Close. Компоненты ИМЕНА содержат по шесть разных имен, а ФАМИЛИИ – по восемь разных фамилий. Их выбор и нажатие кнопки Button приводят к отображению на метке конкатенации имени и фамилии. В каждый момент на форме отображаются только два списка с именами и фамилиями (или женские, или мужские). Два других

списка в этот момент невидимы. Женские сочетания отображаются красным цветом, а мужские - синим. Кнопка **Close** заканчивает программу.

1-25. На форме располагаются компоненты: два радионабора: **A** и **B**, - по 8 разных чисел, редактор **Text**, список **ListBox**, линейка **Scale**, кнопка **Button** и кнопка **Close**. Выбор чисел и нажатие кнопки **Button** приводят к отображению в новой строке редактора результата  $A \otimes B$ . Список **ListBox** задает тип операции (+, -, \*, /). **Scale** задает цвет результата (не менее 5 цветов). Кнопка **Close** заканчивает программу.

1-26. На форме располагаются компоненты: четыре кнопки с числами 3, -7, 304, 3691; **Entry**; метка, кнопки «Дублирование» и «Закреть». Исходно окно редактора пусто. Нажатие кнопки с числом приводит к появлению в окне редактора числа, добавляемого в начало строки. Нажатие кнопки "Дублирование" дублирует содержимое редактора на метку. Кнопка «Очистить» приводит окно редактора в исходное состояние. При достижении в редакторе строкой длины 25 символов кнопки с номерами исчезают и появляются только после нажатия кнопки «Очистить». Её нажатие возвращает систему в исходное состояние.

1-27. На форме располагаются: семь разных кнопок с номерами или названиями дней недели сверху; линейка **Scale**, задающая количество доступных кнопок ( $4 \div 7$ ); радиогруппа из двух компонентов; кнопка **Close** и редактор **Entry**. Нажатие каждой кнопки приводит к ее сокрытию и показу ранее скрытой кнопки. При этом номер (название) ранее скрытой кнопки дублируется в редакторе **Entry**. Вид надписи на кнопках задается радиогруппой и дублируется в заголовке формы. Кнопка **Close** заканчивает программу.

1-28. На форме располагаются компоненты: два редактора **Entry**, поименованные «Ширина» и «Высота», в которые можно вводить текст; кнопка «Изменение» и кнопка «Закреть». При вводе в один из редакторов текста соответствующий размер формы меняется при нажатии кнопки «Изменение» на длину введенной строки: если введены только цифры, то в большую сторону, если только буквы – то в меньшую сторону. Изменения реализуются только после изменения в редакторе. Величина изменения отображается на заголовке формы.

- 1-29. На форме располагаются компоненты: метка **Label** на середине формы с номером группы; **RadioGroup** 1 исходно с опциями ←, ↑, →, ↓; **RadioGroup** 2 из трех кнопок, кнопка «Сдвиг» и кнопка «Закреть». Радионабор 2 задает шаг перемещения (1, 2, 3 пикселя). При выборе мышью опции с направлением и нажатии кнопки «Сдвиг» метка смещается в соответствующем направлении на выбранное число пикселей. При достижении края формы соответствующая опция с направлением в **RadioGroup** исчезает и появляется при отходе от края. Если направление смещения не выбрано, кнопка «Сдвиг» не видна.
- 1-30. На форме располагаются компоненты: два редактора **Text**, **Label**; радиогруппа из двух опций "Левый" и "Правый"; список **Listbox** с опциями "Русские буквы", "Латинские буквы", "Цифры", "Знаки препинания"; кнопка "Копирование" и кнопка **Close**. В редакторы могут вводиться строки из букв, цифр, знаков препинания. При нажатии кнопки "Копирование" соответствующие символы, задаваемые списком **Listbox**, копируются на **Label**. Источник копирования указывается радиогруппой. Кнопка **Close** заканчивает программу.
- 1-31. На форме располагаются компоненты: редактор **Text** размером 11x11, в центре которого находится символ **0**; четыре кнопки со стрелками ↑, ↓, →, ←; два виджета **Scale** с подписями «Строка» и «Столбец», диапазон их значений от -5 до 5, а начальное значение равно 0. Нажатие соответствующей стрелки приводят к движению символа в заданном направлении. Движение вверх увеличивает на 1 значение виджета «Столбец», вниз - уменьшает на это же значение. Соответственно, движение влево-вправо меняет значение виджета «Строка». Когда символ доходит до соответствующей границы редактора происходит деактивизация соответствующей стрелки, при отходе от границы – активизация. Двойной щелчок мыши закрывает форму.
- 1-32. На форме располагаются компоненты: два редактора **Entry**, поименованные «Ширина» и «Высота», в которые можно вводить текст; кнопка «Изменение» и кнопка «Закреть». При вводе в соответствующий редактор текста размер формы меняется при нажатии кнопки «Изменение» на длину введенной строки: если во введенном тексте есть три символа 0, то в большую сторону, если ни одного такого символа – то

в меньшую сторону. Изменения реализуются только после изменения в редакторе. Величина изменения отображается в редакторе Text.

1-33. На форме располагаются компоненты: метка Label на середине формы с номером группы; четыре кнопки Button, сдвигающие в соответствующем направлении ←, ↑, →, ↓; RadioGroup из трех опций, кнопка «Копирование» и кнопка «Закреть». Радионабор задает шаг перемещения (1, 2, 3 пикселя). При нажатии каждой кнопки Button метка смещается в соответствующем направлении на выбранное число пикселей. При достижении края формы соответствующая кнопка с направлением исчезает и появляется при отходе от края. Кнопка «Копирование» выводит на заголовок формы текущий шаг перемещения.

1-34. На форме располагаются компоненты: кнопка Button на середине формы с надписью-номером группы; RadioGroup исходно с опциями ←, ↑, →, ↓; редактор Entry, в который можно вводить только цифры 1, 2, 3; список Listbox и кнопка «Закреть». Редактор Entry задает шаг перемещения (1, 2, 3 пикселя). При выборе мышью опции с направлением и нажатии кнопки «Сдвиг» панель смещается в соответствующем направлении на выбранное число пикселей. При достижении края формы соответствующая опция с направлением в RadioGroup исчезает и появляется при отходе от края. Если направление смещения не выбрано, кнопка «Сдвиг» не видна.

1-35. На форме располагаются компоненты: на середине формы редактор Text размером 10x20; Listbox с опциями 1 ÷ 10 для выбора номера строки; Scale, задающая номер позиции в строке в диапазоне 0 ÷ 19; кнопка «Занести» и редактор Entry. При нажатии кнопки «Занести» текст из редактора Entry вставляется в редактор Text на позицию, заданную Listbox'ом и шкалой Scale. Вставленный фрагмент окрашивается цветом, номер которого определяется произведением заданной строки на столбец вставки.

2-1. На форме располагаются: шесть разных кнопок с номерами сверху, линейка Scale с разметкой от 1 до 6, кнопка Close, редакторы Entry и Text, флажок Checkbutton. Движение ползунка линейки приводит к последовательному сокрытию соответствующих кнопок и показу ранее скрытой кнопки. При этом номер ранее скрытой кнопки добавляется к строке в окне редактора. Вид текущего редактора устанавливается флажком CheckButton. Кнопка Close заканчивает программу.

2-2. На форме располагаются: семь редакторов **Entry** с вписанными названиями дней недели, кнопка **Close**, список **ListBox** с названиями шести цветов и метка. Нажатие клавиш клавиатуры (п, в, с, ч, я, у, о или П, В, С, Ч, Я, О) приводит сокрытию соответствующего редактора и отображению соответствующего дня недели на панели, а также показу ранее скрытого редактора. После появления ранее скрытого редактора текст в нем отображается цветом, заданным списком **ListBox**. Регистр символов определяется линейкой **Scale** на две позиции. Кнопка **Close** заканчивает программу.

2-3 На форме располагаются компоненты: на середине формы заполненный символами редактор **Text** размером 10x10; две линейки **Scale**, задающая номер позиции символа в редакторе и расположенные соответственно вертикально и горизонтально; кнопки со стрелками ← и → (движение по часовой стрелке, движение против часовой стрелки). Исходно символ в левом верхнем углу выделен жирным красным цветом. При нажатии соответствующих стрелок этот вариант форматирования передается поочередно символам по периметру редактора.

2-4. На форме располагаются: редактор **Text**; линейка **Scale** с разметкой от 10 до 20; метка с названием группы; три кнопки с надписями «Влево», «Вправо», «Центр»; радиогруппа с аналогичными функциями и кнопка **Close**. Нажатие одной из кнопок или переключение радионабора приводит к изменению выравнивания надписи на метке. Название режима выравнивания дублируется в редакторе **Text**. Линейка **Scale** задает размер шрифта надписи. Кнопка **Close** заканчивает программу.

2-5. На форме располагаются компоненты: два редактора **Entry**, **Label**; радиогруппа из двух опций "Левый" и "Правый"; линейка **Scale**, размеченная вариантами "Русские буквы", "Латинские буквы", "Цифры", "Знаки препинания"; кнопка "Копирование" и кнопка **Close**. В редакторы могут вводиться строки из букв, цифр, знаков препинания. При нажатии кнопки "Копирование" соответствующие символы, задаваемые списком **Listbox**, копируются на **Label**. Источник копирования указывается радиогруппой. Кнопка **Close** заканчивает программу.

2-6. На форме располагаются компоненты: два списка **ListBox** с подписями А и В, метка **Label**, две линейки **Scale**, кнопка **Button** и кнопка **Close**. Каждый компонент **ListBox** содержит шесть разных чисел. Их выбор и нажатие кнопки **Button** приводят к отоб-

ражению на метке результата  $A \otimes B$ . Одна линейка **Scale** задает тип операции (+, -, \*), а вторая - цвет результата (не менее 6 цветов). Кнопка **Close** заканчивает программу.

2-7. На форме располагаются компоненты: два радионабора **A** и **B** по 8 разных чисел, редактор **Text**, список **Listbox**, линейка **Scale**, кнопка **Button** и кнопка **Close**. Выбор чисел и нажатие кнопки **Button** приводят к отображению в новой строке редактора результата  $A \otimes B$ . Список **Listbox** задает тип операции (+, -, \*, /). **Scale** задает цвет результат (не менее 5 цветов). Кнопка **Close** заканчивает программу.

2-8. На форме располагаются компоненты: четыре кнопки с числами 3, -7, 304, 3691; **Entry**; метка, кнопки «Дублирование» и «Закреть». Исходно окно редактора пусто. Нажатие кнопки с числом приводит к появлению в окне редактора числа, добавляемого в начало строки. Нажатие кнопки «Дублирование» дублирует содержимое редактора на метку. Кнопка «Очистить» приводит окно редактора в исходное состояние. При достижении в редакторе строкой длины более 25 символов кнопки с номерами исчезают и появляются только после нажатия кнопки «Очистить». Её нажатие возвращает систему в исходное состояние.

2-9. На форме располагаются: семь разных кнопок с номерами или названиями дней недели на них; линейка **Scale**, задающий количество доступных кнопок ( $4 \div 7$ ); радиогруппа из двух компонентов; кнопка **Close** и редактор **Entry**. Нажатие каждой кнопки с номером/названием приводит к ее сокрытию и показу ранее скрытой кнопки. При этом номер (название) ранее скрытой кнопки дублируется в редакторе **Entry**. Вид надписи на кнопках задается радиогруппой. Кнопка **Close** заканчивает программу.

2-10. На форме располагаются компоненты: метка **Label** на середине формы с номером группы; **Radiogroup 1** исходно с опциями  $\leftarrow$ ,  $\uparrow$ ,  $\rightarrow$ ,  $\downarrow$ ; **Radiogroup 2** из трех кнопок, кнопка «Сдвиг» и кнопка «Закреть». Радионабор 2 задает шаг перемещения (1, 2, 3 пикселя). При выборе мышью опции с направлением и нажатии кнопки «Сдвиг» правой клавишей мыши панель смещается в соответствующем направлении на выбранное число пикселей. При достижении края формы соответствующая опция с направлением в **Radiogroup** исчезает и появляется при отходе от края. Если направление смещения не выбрано, кнопка «Сдвиг» не видна.

2-11. На форме располагаются компоненты: два редактора **Entry**, поименованные «Ширина» и «Высота», в которые можно вводить текст; кнопка «Изменение» и кнопка «Закреть». При вводе в соответствующий редактор текста размер формы меняется при нажатии кнопки «Изменение» на длину введенной строки: если введены только цифры, то в большую сторону, если только буквы – то в меньшую сторону. Изменения реализуются только после изменения в редакторе. Величина изменения отображается на заголовке формы.

2-12. На форме располагаются компоненты: на середине формы метка **Label** с указанием номера группы; **ListBox** с опциями ←, ↑, →, ↓; радиогруппа для выбора шага перемещения на 1, 2 или 3 пикселя; **Scale**, задающая размер шрифта на метке в диапазоне 8-16 пикселей; кнопка «Сдвиг» и кнопка «Закреть». При выборе мышью соответствующей опции с направлением и нажатии кнопки «Сдвиг» форма смещается в соответствующем направлении на заданное число пикселей.

2-13. На форме располагаются компоненты: редактор **Text** размером 10x20, занимающий центр окна и полностью заполненный текстом; еще один редактор **Text** размером 20x10, исходно пустой; редактор **Entry**; кнопка «Поиск». Ввод в редактор **Entry** некоторого символа и нажатие кнопки «Поиск» приводит к тому, что во втором редакторе **Text** в столбик выводятся координаты заданного символа в первом редакторе **Text**. Если указанный символ отсутствует в первом редакторе, то в окне второго редактора выводится соответствующее сообщение.

2-14. На форме располагаются компоненты: редактор **Entry**, занимающий центр окна в который можно вводить только числа от 2 до 5; компонент **Listbox** с четырьмя опциями ←, ↑, →, ↓; линейка **Scale**, также размеченная от 2 до 5. Редактор **Entry** задает шаг перемещения. Выбор одной из опций списка **Listbox** приводит к перемещению этого виджета в выбранном направлении на число пикселей, заданных в редакторе **Entry**. При этом заданный шаг дублируется на линейке **Scale**, которая устанавливает вертикальную ориентацию, если список перемещается вертикально, и горизонтальную – при горизонтальном перемещении списка.

2-15. На форме располагаются компоненты: редактор **Entry**, две метки **Label** с названиями "Левая" и "Правая"; радиогруппа из двух опций с теми же вариантами "Левая",

"Правая"; список **Listbox** с опциями "Русские заглавные буквы", "Русские прописные буквы", "Латинские заглавные буквы", "Латинские прописные буквы", "Цифры"; кнопка "Копирование" и кнопка **Close**. В редактор могут вводиться строки из букв, цифр, знаков препинания. При нажатии кнопки "Копирование" соответствующие символы, задаваемые списком **Listbox**, копируются на одну из **Label**. Адресат копирования указывается радиогруппой. Кнопка **Close** заканчивает программу.

2-16. На форме располагаются компоненты: два списка **Listbox** с названиями «Столица» и «Страна», заполненные 8 названиями столиц и 8 названиями соответствующих им стран; редактор **Text**, список **Listbox** с вариантами 5 размеров шрифта, кнопка **Button** и кнопка **Close**. Выбор столицы и страны, нажатие кнопки **Button** приводят к отображению их конкатенации в строке редактора **Text** в новой строке. При этом если сочетание корректно, то текст отображается зеленым цветом, в противном случае красным. Список **Listbox** задает размер шрифта у очередной строки. Кнопка **Close** заканчивает программу.

2-17. На форме располагаются: семь разных кнопок с номерами или названиями дней недели сверху; линейка **Scale** с диапазоном значений  $1 \div 7$ ; радиогруппа из двух компонентов; кнопка **Close** и редактор **Entry**. Нажатие каждой кнопки приводит к ее сокрытию и показу ранее скрытой кнопки. При этом название ранее скрытой кнопки дублируется в редакторе **Entry**, а номер переводит в соответствующее значение линейку **Scale**. Вид надписи на кнопках задается радиогруппой и дублируется в заголовке формы. Кнопка **Close** заканчивает программу.

2-18. На форме располагаются компоненты: два редактора **Entry**, поименованные «Нижняя граница» и «Верхняя граница», в которые можно вводить текст; линейка **Scale** с диапазоном  $0 \div 50$ ; кнопка «Изменение» и кнопка «Закреть». При вводе в один из редакторов текста соответствующая граница линейки меняется при нажатии кнопки «Изменение» на длину введенной строки: если введены только цифры, то в большую сторону, если только буквы – то в меньшую сторону. При этом нижняя граница диапазона всегда должна оставаться меньше верхней. Изменения границы реализуются только после изменения в редакторе. Величина изменения отображается на заголовке формы.

2-19. На форме располагаются компоненты: редактор **Text** размером 33x27, в котором по периметру движется символ русского или английского алфавита, вид которого определяется его позицией в редакторе как номером в соответствующем алфавите. Две кнопки со стрелками ←, → задают движение по часовой или против часовой стрелки. При движении по часовой стрелке символы находятся в нижнем регистре, против – в верхнем. Номер символа в алфавите отображается на заголовке формы.

2-20. На форме располагаются компоненты: два редактора **Text**, **Label**; радиогруппа из двух опций "Левый" и "Правый"; список **Listbox** с опциями "Русские буквы", "Латинские буквы", "Цифры", "Знаки препинания"; кнопка "Копирование" и кнопка **Close**. В редакторы могут вводиться строки из букв, цифр, знаков препинания. При нажатии кнопки "Копирование" соответствующие символы, задаваемые списком **Listbox**, копируются на **Label**. Источник копирования указывается радиогруппой. Кнопка **Close** заканчивает программу.

2-21. На форме располагаются компоненты: редактор **Text** размером 10x10, в левом верхнем углу которого находится начальный символ русского алфавита; по кнопке **Button** с каждой стороны редактора с соответствующими стрелками →, ↓, ←, ↑, задающими движение по часовой стрелке символа по периметру редактора. При движении происходит смена символов в соответствии с их последовательностью в русском алфавите. Смена происходит по кольцу. Номер символа в русском алфавите отображается в заголовке формы.

2-22. На форме располагаются: редактор **Text**; линейка **Scale** с разметкой от 10 до 20; метка с названием группы шириной 15 символов; три кнопки с надписями «Влево», «Вправо», «Центр»; радиогруппа с аналогичными функциями и кнопка **Close**. Нажатие одной из кнопок или переключение радионабора приводит к изменению выравнивания надписи на метке. Название режима выравнивания добавляется в конец редактора **Text**. Линейка **Scale** задает размер шрифта надписи. Кнопка **Close** заканчивает программу.

2-23. На форме располагаются компоненты: последовательно расположенные четыре редактора **Entry**, в которые можно вводить не более десяти символов. Исходно текстовый курсор находится в левом редакторе. При вводе информации сначала запол-

няется первый редактор, после чего курсор переходит во второй и т.д. Стирание информации клавишей **Backspace** происходит в обратном порядке. Линейка **Scale** показывает сквозную позицию курсора. Кнопка **Clear** возвращает системы в исходное состояние.

2-24. На форме располагаются: семь кнопок **Button** с номерами или названиями дней недели сверху; линейка **Scale** с диапазоном значений 1 - 7; радиогруппа из двух компонентов; кнопки **Clear** и **Close**, редактор **Entry** с значением 0. Исходно 1-я кнопка скрыта. Движение ползунка линейки **Scale** приводит к сокрытию очередной кнопки и показу ранее скрытой. При этом номер (если на кнопках отображаются номера) ранее скрытой кнопки добавляется к числу в редакторе **Entry**. Если число в редакторе превысило 30, то линейка **Scale** деактивируется. Вид надписи на кнопках задается радиогруппой. Кнопки **Clear** возвращает систему исходное состояние. Кнопка **Close** заканчивает программу.

2-25. На форме располагаются компоненты: на середине формы метка **Label** с указанием номера группы; **ListBox** с опциями ←, ↑, →, ↓; радиогруппа для выбора шага перемещения на 1, 2 или 3 пикселя; радионабор, задающая размер шрифта на метке в диапазоне 8-16 пикселей; кнопка «Сдвиг» и кнопка «Закреть». При выборе мышью некоторой опции с направлением и нажатии кнопки «Сдвиг» форма смещается в соответствующем направлении на заданное число пикселей, а метка остается на месте. Если метка касается нижней границы, то программа возвращается в исходное состояние.

2-26. На форме располагаются компоненты: редактор **Text** размером 20x20, занимающий центр окна; компонент **ListBox** с четырьмя опциями ←, ↑, →, ↓; компонент **ComboBox** с двумя опциями: "Высота" и "Ширина", - и кнопка «Пуск». Выбор опции "Высота" приводит к сокрытию кнопок ←, →, а опции "Ширина" - ↑, ↓. Нажатие кнопки «Пуск» приводит к изменению размера окна редактора (правого нижнего угла) на 1 символ в направлении, указанной в компоненте **ListBox**. Текущий размер редактора отображается в самом редакторе. Если опция не выбрана, то в окне редактора выводится соответствующее сообщение.

2-27. На форме располагаются: две панели для вывода размера формы (положения формы на экране) и две панели с соответствующими надписями «Ширина» и «Высота» (X и Y); четыре кнопки  $\leftarrow$ ,  $\uparrow$ ,  $\rightarrow$ ,  $\downarrow$ ; радиопереключатель на два положения; редактор Entry и кнопка Close. Нажатие кнопок со стрелками приводит к соответствующему изменению размера формы или перемещению формы по пространству экрана на 1 пиксель, что отображается на панелях. Вид перемещения задается радиопереключателем и отображается в редакторе. Кнопка Close заканчивает программу.

2-28. На форме располагаются: редактор Text; линейка Scale с разметкой от 10 до 20; метка с названием группы; три кнопки с надписями «Влево», «Вправо», «Центр»; радиогруппа с аналогичными функциями и кнопка Close. Нажатие одной из кнопок или переключение радионабора приводит к изменению выравнивания надписи на метке. Название режима выравнивания дублируется в редакторе Text. Линейка Scale задает размер шрифта надписи. Кнопка Close заканчивает программу.

2-29. На форме располагаются компоненты: два редактора Entry, Label; радиогруппа из двух опций "Левый" и "Правый"; список Listbox с опциями "Русские буквы", "Латинские буквы", "Цифры", "Знаки препинания"; кнопка "Копирование" и кнопка Close. В редакторы могут вводиться строки из букв, цифр, знаков препинания. При нажатии кнопки "Копирование" соответствующие символы, задаваемые списком Listbox, копируются на Label. Источник копирования указывается радиогруппой. Кнопка Close заканчивает программу.

2-30. На форме располагаются компоненты: два списка ListBox с подписями A и B, метка Label, два списка ComboBox, кнопка типа Button и кнопка Close. Каждый компонент ListBox содержит шесть разных чисел. Их выбор и нажатие кнопки Button приводят к отображению на метке результата  $A \otimes B$ . Один список ComboBox задает тип операции (+, -, \*), а второй - цвет результата (не менее 6 цветов). Кнопка Close заканчивает программу.

2-31. На форме располагаются компоненты: два радионабора A и B по 8 разных чисел, редактор Text, список ListBox, список ComboButton, кнопка Button и кнопка Close. Выбор чисел и нажатие кнопки Button приводят к отображению в новой строке редактора результата  $A \otimes B$ . Список ListBox задает тип операции (+, -, \*, /).

**ComboButton** задает цвет результат (не менее 5 цветов). Кнопка **Close** заканчивает программу.

2-32. На форме располагаются компоненты: четыре кнопки с числами 3, -7, 304, 3691; **Entry**; метка, кнопки «Дублирование» и «Заккрыть». Исходно окно редактора пусто. Нажатие кнопки с числом приводит к появлению в окне редактора числа, добавляемого в начало строки. Нажатие кнопки "Дублирование" дублирует содержимое редактора на метку. Кнопка «Очистить» приводит окно редактора в исходное состояние. При достижении в редакторе строкой длины 25 символов кнопки с номерами исчезают и появляются только после нажатия кнопки «Очистить». Её нажатие возвращает систему в исходное состояние.

2-33. На форме располагаются: семь разных кнопок с номерами или названиями дней недели сверху; линейка **Scale** задающий количество доступных кнопок ( $4 \div 7$ ); радиогруппа из двух компонентов; кнопка **Close** и редактор **Entry**. Нажатие каждой кнопки приводит к ее сокрытию и показу ранее скрытой кнопки. При этом номер (название) ранее скрытой кнопки дублируется в редакторе **Entry**. Вид надписи на кнопках задается радиогруппой. Кнопка **Close** заканчивает программу.

2-34. На форме располагаются компоненты: метка **Label** на середине формы с номером группы; **RadioGroup** 1 исходно с опциями  $\leftarrow$ ,  $\uparrow$ ,  $\rightarrow$ ,  $\downarrow$ ; **RadioGroup** 2 из трех кнопок, кнопка «Сдвиг» и кнопка «Заккрыть». Радионабор 2 задает шаг перемещения (1, 2, 3 пикселя). При выборе мышью опции с направлением и нажатии кнопки «Сдвиг» панель смещается в соответствующем направлении на выбранное число пикселей. При достижении края формы соответствующая опция с направлением в **RadioGroup** исчезает и появляется при отходе от края. Если направление смещения не выбрано, кнопка «Сдвиг» не видна.

2-35. На форме располагаются: две панели для вывода размера формы (положения формы на экране) и две панели с соответствующими надписями «Ширина» и «Высота» (X и Y); четыре кнопки  $\leftarrow$ ,  $\uparrow$ ,  $\rightarrow$ ,  $\downarrow$ ; радиопереключатель на два положения; редактор **Entry** и кнопка **Close**. Нажатие кнопок со стрелками приводит к соответствующему изменению размера формы или перемещению формы по пространству экрана на 1 пиксель, что отображается на панелях. Вид перемещения задается радиопереключателем и отображается в редакторе. Кнопка **Close** заканчивает программу

## Задание № 2 Разработка меню

### *Теоретический раздел*

Меню стало стандартным элементом Windows-программ, поэтому язык Python содержит эффективные средства их создания и использования в составе проектов. Находится оно под строкой заголовка и представляет собой выпадающие списки под словами-пунктами меню. Пункты конечных списков представляют собой команды, обычно выполняющие какое-либо действие или открывающие диалоговые окна.

В `tkinter` экземпляр меню создается от класса `Menu`, далее его надо привязать к виджету, на котором оно будет расположено. Обычно таковым выступает главное окно приложения. Его свойству `menu` присваивается экземпляр `Menu` через имя связанной с экземпляром переменной. В приложении может быть несколько главных меню, но активным является всегда одно. Переключение между меню производится командой `root.config(menu=<название_главного_меню>)`.

```
from tkinter import *

root = Tk()
mainmenu = Menu(root)
root.config(menu=mainmenu)

root.mainloop()
```

Если выполнить данный код, то никакого меню пока не видно. Появится только тонкая полоска под заголовком окна, ведь ни одного пункта меню не было создано. Метод `add_command()` добавляет пункт меню:

```
...
mainmenu.add_command(label='Файл')
mainmenu.add_command(label='Справка')
...
```

В данном случае "Файл" и "Справка" – это опции главного меню. К ним можно добавить свойство `command`, связав тем самым опцию с какой-либо функцией-обработчиком клика. Хотя такой вариант меню имеет право на существование, в большинстве приложений панель главного меню содержит выпадающие списки команд, а сами опции на панели командами по сути не являются. Клик по ним приводит лишь к раскрытию соответствующего списка.

В `tkinter` проблема создания выпадающих списков и подвязывания их к главному меню решается с помощью метода `add_cascade()`.

### Листинг 2.1

```
from tkinter import *

root = Tk()

mainmenu = Menu(root)
root.config(menu=mainmenu)

filemenu = Menu(mainmenu, tearoff=0)
filemenu.add_command(label="Открыть...")
filemenu.add_command(label="Новый")
filemenu.add_command(label="Сохранить...")
filemenu.add_command(label="Выход")

helpmenu = Menu(mainmenu, tearoff=0)
helpmenu.add_command(label="Помощь")
helpmenu.add_command(label="О программе")

mainmenu.add_cascade(label="Файл", menu=filemenu)
mainmenu.add_cascade(label="Справка", menu=helpmenu)

root.mainloop()
```

В результате будет создано следующее дерево меню (рис. 2.1):

<b>Файл</b>	<b>Справка</b>
Открыть...	Помощь
Новый	О программе
Сохранить...	
Выход	

Рис. 2.1

Анализ листинга 2.1 и рис. 2.1 показывает, что дерево меню в модуле `tkinter` строится снизу вверх, поэтому целесообразно перед началом конструирования его построить графически целиком, а потом переходить к реализации.

На основное меню (`mainmenu`), добавляются не команды, а другие меню. У выпадающих меню `filemenu` и `helpmenu` в качестве родительского виджета указывается не `root`, а `mainmenu`. Команды обычно уже к дочерним меню. Значение 0 параметра `tearoff` отключает возможность открепления подменю, иначе его можно было бы делать плавающим кликом мыши по специальной пунктирной линии (в случае `tearoff=0` она отсутствует).

Точно также можно подвязывать дочерние меню к `filemenu` и `helpmenu`, создавая многоуровневые списки пунктов меню.

```

...
helpmenu = Menu(mainmenu, tearoff=0)

helpmenu2 = Menu(helpmenu, tearoff=0)
helpmenu2.add_command(label="Локальная справка")
helpmenu2.add_command(label="На сайте")

helpmenu.add_cascade(label="Помощь", menu=helpmenu2)

helpmenu.add_command(label="О программе")
...

```

Теперь меню будет трехуровневым (рис. 2.2):

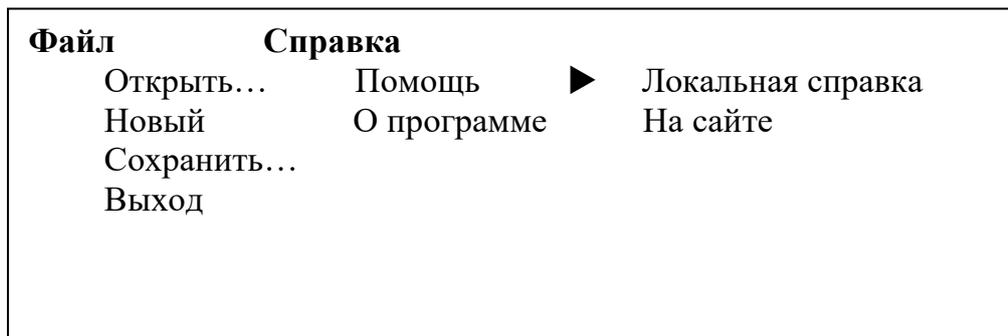


Рис. 2.2

Метод `add_separator()` добавляет линию разделитель в меню. Используется для визуального разделения групп команд.

Для связи опции меню с действиями используется её свойство:

**command = <функция>**

Наиболее часто привязка горячих клавиш к опции меню имитируется следующим образом: дублируется функция, запускаемая опцией меню, и создается событие для ее запуска в виде нажатия горячих клавиш, которое привязывается к форме.

```

...
def zan():
    b1["text"] = 'Самолет'
...
sgrmenu.add_command(label="Транспорт: <Control-q>", command = zan)
...
def zan1(event):
    b1["text"] = 'Самолет'

root.bind('<Control-q>', zan1)
...

```

В примере предполагается, что функция `zap` запускается опцией меню "Транспорт". Эта же функция дублируется и запускается сочетанием горячих клавиш `<Control-q>`, привязанных к форме.

Привязка пиктограмм к опциям меню реализуется по следующему алгоритму:

1. Создается объект, в который загружается файл пиктограммы. Для этого можно использовать графический элемент `PhotoImage`.

```
im = PhotoImage(file = 'smiley.gif')
```

2. Объект привязывается к требуемой опции меню

```
sgrmenu.add_command(label="Пассажирские", image = im, compound="left" )
```

Анализ этой последовательности показывает, что она аналогична загрузке изображения на кнопку `Button`, рассмотренной в лабораторной работе №1.

Еще один важный аспект формирования меню программного приложения - создание контекстного меню. В этом случае экземпляр меню привязывается не к родительскому виджету, а к меню применяется метод `post()`, аргументами которого являются координаты того места, где должно появляться меню.

В примере контекстное меню используется для смены размера шрифта на кнопке. Для этого:

1. Создается набор функций, каждая из которых устанавливает свое значение шрифта:

```
def f12():  
    b1["font"] = "Arial 12"
```

```
def f15():  
    b1["font"] = "Arial 15"
```

```
def f18():  
    b1["font"] = "Arial 18"
```

```
def f21():  
    b1["font"] = "Arial 21"
```

2. Создается набор опций контекстного меню, каждая из которых активизирует одну из этих функций.

```
v = IntVar()  
pm = Menu(tearoff=0)  
pm.add_checkbutton(label="12", variable = v, onvalue = 0, offvalue=0, command=f12)
```

```
pm.add_checkbutton(label="15", variable = v, onvalue = 1, offvalue=0, com-  
mand=f15)  
pm.add_checkbutton(label="18", variable = v, onvalue = 2, offvalue=0, com-  
mand=f18)  
pm.add_checkbutton(label="18", variable = v, onvalue = 3, offvalue=0, com-  
mand=f21)
```

Метод `add_checkbutton` добавляет к опции свойство `checked`, т. е. возможность отображения текущего выбора в виде 'галочки'.

Именно переменная `v = IntVar()` объединяет все четыре опции в одну группу, т. е. можно создавать несколько разных контекстных меню с разными связующими переменными. У всех опций разное значение переменной `onvalue`.

3. Производится привязка контекстного меню к конкретному месту интерфейса.

Для этого используются методы `event.x` и `event.y`

```
x1 = event.x
```

```
y1 = event.y,
```

которые сообщают текущие координаты в локальной системе того виджета, над которым находится мышь. А далее, если выполняется требуемое условие, с помощью метода `post()` происходит отображение контекстного меню.

```
def pom(event):  
    x1 = event.x  
    y1 = event.y  
    if (x1 > 0 and x1 < 111) and (y1 > 0 and y1 < 24):  
        pm.post(event.x_root, event.y_root)
```

```
b1.bind("<Button-3>", pom)
```

Но место меню уже указывается в абсолютных координатах всего экрана `event.x_root` и `event.y_root`.

При этом надо учитывать, что размер некоторых виджетов может меняться в ходе работы программы, т. е. для корректности надо адаптивно менять размеры области, доступной для вызова контекстного меню.

Для временной дезактивации некоторой опции меню в ее свойствах указывается `state = DISABLED`.

### *Постановка задачи*

1. На основании иерархии объектов предметной области, заданной вариантом в табл. 2.1, сформировать главное меню программного проекта. В меню должно быть не

менее 20 опций и четырех уровней, причем две любые опции должны быть дополнены графическими пиктограммами.

2. У двух конечных опций должны быть горячие клавиши. Выбор одной из таких опций приводит к:

- для четных номеров варианта - появлению в центре метки с фамилией студента;
- для нечетных – появлению в окне однострочного редактора фамилии студента.

Выбор другой такой опции приводит к исчезновению вышеуказанного текста.

3. Создать контекстное меню, привязанное к редактору или метке, в зависимости от варианта. В меню пять опций: для четных номеров – для выбора цвета текста, для нечетных – для задания размера шрифта текста. Выбор опции сопровождается установкой рядом с ней метки  $\surd$  и стирания предыдущей метки.

#### *Методика (алгоритм) выполнения задания*

- I. В «Теоретическом разделе» изучить необходимые средства и методы модуля tkinter для формирования главного и контекстного меню программного приложения.
- II. Из раздела «Варианты задания» в соответствии с вариантом (номером в списке группы: первая цифра-номер группы, далее идет номер студента в группе) выбрать предметную область для разработки главного меню:
- III. Сформировать дерево главного меню и реализовать его средствами языка Python.
- IV. Дополнить две опции главного меню графическими пиктограммами.
- V. Ввести в состав программного проекта редактор или метку с фамилией в зависимости от варианта.
- VI. Разработать процедуры управления визуализацией виджета с фамилией.
- VII. Связать процедуры управления визуализацией виджета с фамилией с опциями главного меню.
- VIII. Создать для с опций главного меню, управляющих визуализацией виджета с фамилией, горячие клавиши.
- IX. Разработать контекстное меню для управления параметров текста на виджете с фамилией и привязать его к данному виджету.
- X. Произвести отладку полученного программного приложения.
- XI. Предоставить преподавателю возможность проверки корректности выполнения задания в дистанционном или очном варианте.

ХП. Оформить отчет по выполненному заданию, содержащий:

- 1) задание;
- 2) структуру проекта;
- 3) блок-схему начальной установки приложения, блок-схемы процедур.
- 4) листинг программы.

*Варианты задания*

Таблица 2.1

<b>Номер варианта</b>	<b>Предметная область</b>
1	2
1-1	Животный мир
1-2	Улицы города
1-3	Автомобили
1-4	Летательные аппараты
1-5	Средства вычислительной техники
1-6	Периодические издания
1-7	Книги
1-8	Сотрудники вуза
1-9	Студенты
1-10	Учебные курсы
1-11	Помещения вуза
1-12	Принтеры
1-13	Фирмы
1-14	Программные пакеты
1-15	Города
1-16	Водоемы
1-17	Медицинские учреждения
1-18	Страны мира
1-19	Учебные заведения
1-20	Плавательные средства
1-21	Птицы
1-22	Одежда
1-23	Виды спорта
1-24	Бытовые электроприборы
1-25	Магазины
1-26	Небесные тела
1-27	Детские игрушки
1-28	Кухонная мебель
1-29	Часы
1-30	Развлекательные заведения

1	2
1-31	Места отдыха
1-32	Жилища
1-33	Столовые приборы
1-34	Спортивный инвентарь
1-35	Источник света
2-1	Цветы
2-2	Фильмы
2-3	Музыкальные инструменты
2-4	Музыкальные произведения
2-5	Телевизионные передачи
2-6	Компьютерные игры
2-7	Мебель
2-8	Продукты питания
2-9	Товары магазина
2-10	Игрушки
2-11	Виды транспорта
2-12	Художники
2-13	Композиторы
2-14	Писатели
2-15	Сотовые телефоны
2-16	Головные уборы
2-17	Обувь
2-18	Учебные дисциплины вуза
2-19	Растительный мир
2-20	Оружие
2-21	Самолеты
2-22	Городской транспорт
2-23	Сотрудники завода
2-24	Кухонная утварь
2-25	Осветительные приборы
2-26	Школьные принадлежности
2-27	Сайты
2-28	Породы собак
2-29	Породы кошек
2-30	Водоплавающие птицы
2-31	Рыбы
2-32	Произведения живописи
2-33	Печатная продукция
2-34	Животный мир
2-35	Сайты

## Контрольная работа № 2

### Задание № 3 Разработка усложненного Python-проекта

#### *Теоретический раздел*

К настоящему времени средства модуля `tkinter` расширены модулем `tkinter.ttk` (`themed tk`), представляющим дополнительные возможности для конструирования графических пользовательских интерфейсов. Этот модуль включает средства стилизации уже рассмотренных виджетов (`Button`, `Checkbutton`, `Entry`, `Label`, `Menubutton`, `Radiobutton`, `Scale`), а также вводит в использование ряд новых виджетов:

### **SpinBox**

Данный виджет используется для задания конкретного численного значения из заданного перечня. Для создания спинбокса используется класс `Spinbox`:

```
spin = Spinbox(root, from_=0, to=100)
```

Перечень значений можно задавать двумя способами:

1. указанием граничных значений диапазона;
2. перечислением возможных значений.

Выше представлен первый вариант задания, где параметры `from_=0`, `to=100` задают границы диапазона.

Разрешается просто перечислить для `Spinbox` диапазон возможных значений следующим образом:

```
spin = Spinbox(window, values=(3, 8, 11), width=5)
```

Тогда виджет будет отображать только эти 3 числа: 3, 8 и 11. Кроме того, можно указать ширину виджета с помощью параметра `width`.

Также можно задать значение по умолчанию для `Spinbox`. Для этого надо передать это значение параметру `textvariable` следующим образом:

```
var = IntVar()
```

```
var.set(36)
```

```
spin = Spinbox(window, from_=0, to=100, width=5, textvariable=var)
```

Параметр `textvariable` определяет переменную (в данном случае `var`), через которую считывается или устанавливается текущее значение виджета.

## **combobox (разворачивающийся список)**

Данный компонент по своим функция аналогичен предыдущему виджету, но исходно список опций свёрнут и раскрывается при щелчке по кнопке раскрытия. Поскольку виджет находится в модуле `tkinter.ttk`, т. е. требуется отдельная загрузка этого модуля в начале работы скрипта:

```
import tkinter.ttk as ttk
```

Список значений задается с помощью свойства

```
values = ["Один", "Два", "Три", "Четыре", "Пять" ]
```

в виде кортежа:

```
cb = ttk.Combobox(foreground = "#555", background = "#ffffff", values = ["Один",  
"Два", "Три", "Четыре", "Пять" ], height=5)
```

Текст указанной опции списка можно получить с помощью метода `get()`, а задать конкретную опцию – методом `current()`, например, `cb.current(1)`. Соответственно, индекс выбранной опции можно также получить через метод `current`:

```
i = cb.current()
```

Если же выбор не установлен, то значение индекса равно – 1.

Можно также сразу получить индекс выбранного значения через событие `<ComboboxSelected>`:

```
def Otb(event):  
    root.title(cb.current())  
  
cb.bind("<<ComboboxSelected>>", Otb)
```

## **Progressbar**

Данный виджет используется для визуализации динамики протекания указанных процессов. Чтобы создать данный виджет, используется класс `progressbar`:

```
from tkinter.ttk import Progressbar
```

```
...
```

```
bar = Progressbar(root, length=200, orient = HORIZONTAL, mode = 'determinate')
```

Необязательный параметр `orient` имеет два варианта значения: `HORIZONTAL` ("horizontal") и `VERTICAL` ("vertical"). По умолчанию установлен вариант `HORIZONTAL`.

Необязательный параметр `mode` имеет два варианта значения: `"determinate"` и `"indeterminate"`. По умолчанию установлен вариант `"determinate"`. В этом случае работа виджета представляет собой разворачивающуюся ленту. Во втором случае динамика отображается движением ползунка по виджету.

Установить значение `progressbar` можно таким образом:

```
bar['value'] = 70
```

## Всплывающие окна

При работе с модулем `tkinter` можно использовать всплывающее окно `messagebox`, загружаемое в приложение следующим образом:

```
from tkinter import messagebox
```

Использование варианта окна `messagebox.askyesno` приводит к появлению окна-вопроса с двумя кнопками: **Да**, **Нет**. Нажатие "Да" в диалоговом окне возвращает в программу `True`, "Нет" вернет `False` (также как закрытие окна через крестик). Таким образом в коде можно обработать выбор пользователя:

```
answer = messagebox.askyesno(title="Вопрос", message="Перенести данные?")
```

т.е. на основе ответа можно управлять работой программы.

Опции `title` и `message` являются позиционными, так что можно указывать только значения, т. е. упрощенный вариант работы программы имеет вид:

```
messagebox.askyesno("Вопрос", "Перенести данные?")
```

Подобные окна генерируются также при использовании функции `askokcancel()` с надписями на кнопках "ОК" и "Отмена", `askquestion()` (возвращает не `True` или `False`, а строки `'yes'` или `'no'`), `askretrycancel()` ("Повторить", "Отмена"), `askyesnocancel()` ("Да", "Нет", "Отмена").

Другую группу составляют окна с одной кнопкой, которые служат для вывода сообщений различного характера. Это `showerror()`, `showinfo()` и `showwarning()`.

## Кнопка **Speedbutton**

Во многих современных системах программирования средства создания GUI включают кнопку, которая может фиксироваться в двух позициях: нажата, отжата, - т.е. она может использоваться для отображения двоичных значений. При этом такие кнопки также могут образовывать радиогруппы.

В модуле `tkinter` подобные кнопки имитируются стандартной кнопкой `Radiobutton` или `Checkbutton` в режиме параметра `indicatoron = 0`.

```
r_var = BooleanVar()
r_var.set(0)

r1 = Radiobutton(text='Переключение', indicatoron = 0, variable=r_var, value=0)
```

Рассмотренный вариант подходит при формировании группы кнопок `Speedbutton`, работающих в режиме радиопереключателя. Для одиночной кнопки, работающей в режиме бинарного переключателя, необходимо переключение кнопки производить средней (колесиком) или правой кнопкой мыши.

Корректно происходит преобразование в кнопку `Speedbutton` виджета `Checkbutton`.

```
r1 = Checkbutton(text='Запуск', width=10, height=2, variable=var, indicatoron = 0)
```

## Контейнер Canvas

Как уже не раз указывалось ранее, `Canvas` является виджетом-контейнером, в который можно вставлять растровые рисунки. С помощью функций из пакетов `Image` и `ImageTk` из `PIL` формируется объект-изображение для загрузки в контейнер

```
from tkinter import *
from PIL import ImageTk, Image

FILENAME = "pos.jpg" # файл с графическим изображением
root = Tk()
root.geometry('700x500')

c = Canvas(root, width=400, height=200, bg="white")
c.place(x = 200, y = 200)
img = ImageTk.PhotoImage(file = FILENAME)
c.create_image(50, 10, image=img, anchor="nw")

root.mainloop()
```

Можно имя файла сразу загружать в объект `Image`:

```
img = Image.open("pos.jpg")
```

что сократит одну строку.

В контейнере задаются координаты опорной точки (50, 10), относительно которой производится привязка самого рисунка на основе параметра `anchor` (рис. 3.1): `N`, `NE`, `E`, `SE`, `SW`, `W`, `NW` или `CENTER`.

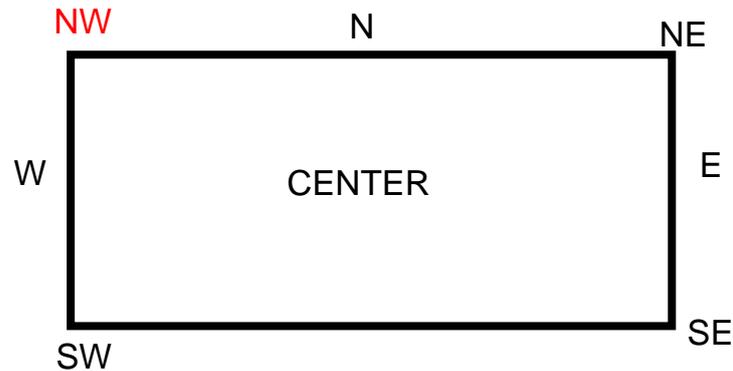


Рис. 3.1

У самого контейнера также задается положение и размер.

можно узнать размеры рисунка по индексу размера:

```
im = Image.open("image.png")
width = im.size[0]
height = im.size[1]
```

Изображение также можно масштабировать, т.е. увеличивать и/или уменьшать с помощью метода `resize`.

```
...
img = Image.open("image.png")
size=(200,200)
out = img.resize(size)

img = ImageTk.PhotoImage(out)
c.create_image(0, 0, image=img, anchor="nw")
```

Можно также с помощью фильтров минимизировать потери качества при увеличении размеров рисунка. Для этого используется набор фильтров библиотеки `PIL`. Их возможные значения: `NEAREST`, `BILINEAR`, `LANCZOS`, `BICUBIC` и `ANTIALIAS`. Ниже приведен пример использования одного из фильтров:

```
...
img = img.resize((500, 500), Image.ANTIALIAS)
...
```

Изображение также можно отображать относительно горизонтальной или вертикальной оси и вращать по часовой стрелке с дискретом  $90^{\circ}$ . Это значения метода `transpose`: `FLIP_LEFT_RIGHT`, `FLIP_TOP_BOTTOM`, `ROTATE_90`, `ROTATE_180`, `ROTATE_270`.

Пример вращения изображения на  $180^{\circ}$ :

```
img = img.transpose(Image.ROTATE_180)
```

### *Постановка задачи*

Создать программное приложение в соответствии с заданием, отладить его, предоставить преподавателю возможность проверки корректности выполнения задания, оформить отчет по выполненному заданию.

### *Методика (алгоритм) выполнения задания*

- I. Из раздела «Варианты задания» в соответствии с вариантом (номером в списке группы: первая цифра-номер группы, далее идет номер студента в группе) выбрать задание на разработку программного проекта:
- II. В «Теоретическом разделе» изучить необходимые средства модуля `tkinter.ttk` для выполнения определенного вариантом задания.
- III. Разработать графический интерфейс средствами языка Python для выполнения определенного вариантом задания.
- IV. Разработать программные модули на языке Python для реализации требуемого в задании функционала.
- V. Привязать разработанный функционал к программному интерфейсу.
- VI. Произвести отладку полученного программного приложения.
- VII. Предоставить преподавателю возможность проверки корректности выполнения задания в дистанционном или очном варианте.
- VIII. Оформить отчет по выполненному заданию, содержащий:
  - 1) задание;
  - 2) структуру проекта;
  - 3) блок-схему начальной установки приложения, блок-схемы процедур.
  - 4) листинг программы.

### *Варианты задания*

- 1-1. На форме располагаются компоненты: редактор `Entry`, группа из четырех кнопок `SpeedButton`, метка `Label`, контекстное меню для формы и две кнопки `Button` (для очистки редактора и закрытия формы) с подходящими по назначению рисунками. В

**Entry** можно ввести только дробное число как аргумент тригонометрической функции. Сама функция: **sin**, **cos**, **tg** или **ctg**, - выбирается с помощью кнопок **SpeedButton**. Запуск на вычисление производится с помощью контекстного меню. Результат помещается на **Label**.

1-2. На форме располагаются компоненты: два редактора **Entry**, **Label**, главное меню, кнопка **SpeedButton**, две кнопки **Button** (с надписями «Очистить» и «Закреть»). В один редактор вводится с проверкой дробное число, а во второе - целое как показатель возведения в степень первого числа. Запуск на вычисление производится с помощью команды главного меню. Результат помещается на **Label**. Кнопка **SpeedButton** убирает/восстанавливает главное меню. Очистка окна аргумента производится кнопкой **Button**. Вторая кнопка заканчивает программу.

1-3. На форме располагаются компоненты: редакторы **Entry** и **Text**, **Label**, **ListBox**, радиопереключатель на два положения, кнопка **Button** с надписью «Да» в виде рисунка и кнопка **Close**. В редактор **Entry** вводится строка, преобразуемая к верхнему или нижнему регистру. Вид преобразования задается радиопереключателем. При нажатии кнопки «Да» строка копируется в **Text** или на **Label**. Приемник копирования выбирается с помощью **ListBox**. Когда редактор **Text** заполнится, копирование производится только на **Label**. Очистка **Text** через контекстное меню восстанавливает исходную ситуацию. Кнопка **Close** заканчивает программу.

1-4. На форме располагаются компоненты: два компонента **ListBox**, таблица размером 4x6 из редакторов **Entry**, кнопка **SpeedButton**, две кнопки **Button** и кнопка **Close**. На одной из кнопок **Button** надпись «Очистка», на другой – «Занесение». Над одним из компонентов **ListBox** надпись «Строки», над другим – «Столбцы». С помощью первого выбирается строка таблицы, другим - столбец. При нажатии кнопки «Занесение» в соответствующую ячейку заносится произведение или сумма номеров строки и столбца, предыдущая ячейка при этом очищается. Вид операции определяется состоянием кнопки **SpeedButton**. Кнопка «Очистка» очищает данную ячейку. Кнопка **Close** заканчивает программу.

1-5. На форме располагаются компоненты: таблица размером 3x6 из компонентов **Label**, шесть кнопок **Button** с пронумерованными названиями изображений, контейнер **Canvas**, радионабор из трех положений и кнопка **Close**. Нажатие кнопки с названием

приводит к появлению на **Canvas** соответствующего изображения и дублированию названия рисунка в соответствующей ячейке строки таблицы. Номер строки задается радионабором, а столбца – номером кнопки. Предыдущая ячейка таблицы при этом очищается.

1-6. На форме располагаются: таблица размером 4x5 из компонентов **Label**, два набора радиокнопок с соответствующими надписями для выбора номеров строк и столбцов таблицы, редактор **Entry**, кнопка **Button** с надписью «Занесение» и кнопка **Close**. Выбор ячейки таблицы и нажатие кнопки «Занесение» приводит к появлению в данной ячейке текста из редактора **Entry**. При этом в другой ячейке надпись исчезает. Редактор требует ввода слов (только буквы) длиной от 4 до 8 символов, в противном случае занесение не происходит. Кнопка **Close** заканчивает программу.

1-7. На форме располагаются компоненты: таблица размером 4x6 из компонентов **Label**, в первую ячейку которого занесена фамилия студента; четыре кнопки со стрелками  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ ; редактор **Entry** для отображения координат текущей ячейки и кнопка **Close**. Нажатие кнопок со стрелками приводит к перемещению в указанном направлении фамилии. Редактор позволяет ввести новые координаты ячейки в указанном формате и диапазоне с клавиатуры, а нажатие клавиши **Enter** приводит к перемещению фамилии в новое место. Остальные ячейки таблицы при этом очищаются.

1-8. На форме располагаются: четыре кнопки **Button** с рисунками и номерами сверху, контейнер **Canvas**, две линейки **Scale**, кнопка **Close** и редактор **Entry**. Нажатие каждой кнопки приводит к дублированию на **Canvas** рисунка с кнопки, а номер кнопки отображается в окне редактора. Линейки **Scale** меняют размер **Canvas** в диапазоне 50x50 .. 150x150. Кнопка **Close** заканчивает программу.

1-9. На форме располагаются компоненты: таблица с размером 6x6 из кнопок **Button**, в первую ячейку которой занесено название группы; метка для отображения координат текущей ячейки; список **ListBox**, меняющий размер таблицы (варианты: 5x4, 5x6, 6x6, 7x6, 5x7), редактор **Entry** и кнопка **Close**. Нажатие клавиш  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$  на клавиатуре приводит к перемещению в указанном направлении названия группы. Редактор позволяет ввести новые координаты ячейки в указанном формате и диапазоне с клавиатуры, а нажатие клавиши **Enter** приводит к перемещению названия группы в новое место. Остальные ячейки таблицы при этом очищаются.

- 1-10. На форме располагаются: семь кнопок **SpeedButton** с подписями-днями недели; линейка **Scale**, размеченная днями недели; кнопка **Close**; флажок **CheckButton** и редактор **Entry**. Перемещение вдоль линейки приводит к нажатию соответствующей кнопки и отображению текущего дня недели в редакторе **Entry**, а флажок задает режим отображения: заглавными или прописными буквами. Кнопка **Close** заканчивает программу.
- 1-11. На форме располагаются: кнопка **SpeedButton**, кнопка **Close**, редакторы **Entry** и **Text**, контейнер **Canvas**. В редакторе **Text** содержится 10 Фамилий И.О. В редакторе **Entry** разрешено вводить текст только по такому же формату. Если введенная фамилия совпадает с одной из фамилий в **Text**, в **Canvas** отображается изображение данного человека. В противном случае фон **Canvas** становится прозрачным, т.е. его не видно. Проверка запускается кнопкой **SpeedButton**. Кнопка **Close** заканчивает программу.
- 1-12. На форме располагаются: таблица размером 5x5 из компонентов **Entry**; главное меню с четырьмя опциями  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ ; кнопка **SpeedButton** и кнопка **Close**. Главное меню меняет координаты активной ячейки, которые отображаются на заголовке формы. При достижении границы таблицы соответствующая опция меню деактивируется и становится рабочей при отходе от границы. В текущую ячейку можно занести дату в формате 01/07/2005 или 01/июл/05. Вид формата меняется кнопкой **SpeedButton**. Кнопка **Close** заканчивает программу.
- 1-13. На форме располагаются: таблица размером 4x7 из компонентов **Label**, в каждой строке которой свой набор элементов: "Цветы", "Деревья", "Реки", "Города"; четыре набора главных меню, соответствующие строкам таблицы; две линейки **Scale** для изменения координат текущей ячейки таблицы; кнопка **Close**. Изменение номера строки делает активным соответствующее ей меню, а изменение номера столбца деактивирует соответствующую ему опцию меню. Кнопка **Close** заканчивает программу.
- 1-14. На форме располагаются: главное меню с шестью опциями-названиями животных, контейнер **Canvas**, кнопка **Close** и радионабор для выбора животного. Выбор осуществляется мышью или клавишами клавиатуры ( $\rightarrow$ ,  $\leftarrow$  или  $\uparrow$ ,  $\downarrow$ ). Выбранное животное отображается на элементе **Canvas**, а соответствующая ему опция в меню становится невидимой. Кнопка **Close** заканчивает программу.

- 1-15. На форме располагаются: календарь на текущий месяц из компонентов **Button** с горизонтальным расположением недель, сверху группа из семи кнопок **SpeedButton**, размеченная днями недели; редактор **Entry**, используемый для ввода числа месяца. Ввод дня в редактор и активация контекстного меню приводят к нажатию кнопки, соответствующей выбранному дню недели. Этот же день дублируется в заголовке формы. Если при вводе числа произошла ошибка, выводится окно `showerror()`. Кнопка **Close** заканчивает программу.
- 1-16. На форме располагаются компоненты: таблица размером 4x4 из редакторов **Entry**, два списка **ListBox** для выбора строки и столбца таблицы, три флажка **CheckBox**. В ячейки таблицы можно в произвольном порядке заносить только текст, только число, смешанную информацию. Посредством клавиш  $\rightarrow$ ,  $\leftarrow$ ,  $\uparrow$ ,  $\downarrow$  клавиатуры можно перемещаться только по выбранным строке и столбцу. При этом активизируется **CheckBox**, соответствующий текущей информации в ячейке. Кнопка **Close** заканчивает программу.
- 1-17. На форме располагаются: таблица размером 3x3, два радионабора для выбора текущей ячейки, компонент **Scale**, редактор **Text**. Все ячейки таблицы заполнены названиями различных предметов (птиц, рыб, животных и т.п.). Компонент **Scale** имеет размер ячейки и перемещается синхронно активной ячейке. При выборе ячейки на нем отображается изображение текущего предмета, а название предмета дублируется в редакторе **Text**. Кнопка **Close** заканчивает программу.
- 1-18. На форме располагается таблица размером 6x7 из компонентов **Entry**. Столбцы таблицы с помощью меток поименованы названиями факультетов, а строки - номерами курсов. Таблица заполнена реальными названиями групп. Редактор **Entry** позволяет вводить только название группы. Если оно совпадает с существующим в таблице названием, на элементе **Scale** появляется изображение декана данного факультета, а название данного факультета отображается в заголовке формы. Кнопка **Close** заканчивает программу.
- 1-19. На форме располагаются: таблица размером 7x1 из компонентов **Entry**, в которую занесены дни недели, редактор **Text**, флажок **CheckBox**, кнопка **Close**. Стрелками клавиатуры  $\uparrow$  и  $\downarrow$  по дням недели перемещается курсор. При этом в редакторе **Text**

отображается текущий день недели. Перемещение вверх (вправо) устанавливает, вниз (влево) – сбрасывает флажок. Кнопка **Close** заканчивает программу.

1-20. На форме располагаются: таблица размером 2x12 из компонентов **Entry**, в первую строку которой занесены названия месяцев; компонент **Canvas** и кнопка **Close**. В проект введены три контекстных меню: для формы, таблицы и **Canvas**. Контекстное меню формы меняет цвет ее фона (6 цветов), таблицы - приводит к отображению во второй строчке количества дней в выбранном месяце, **Canvas** - меняет рисунок на компоненте (5 рисунков). При этом в заголовке формы указывается, какого компонента меню активизировалось. Кнопка **Close** заканчивает программу.

1-21. На форме располагаются компоненты: две таблицы размером 4x4 из компонентов **Entry**, четыре радиокнопки в виде набора, контекстное меню для левой формы и две кнопки (с надписями «Очистить» и «Закреть»). В ячейки левой таблицы заносятся: дробные числа, текст или алфавитно-цифровая информация. В одноименную ячейку правой таблицы заносятся соответственно: результат вычисления тригонометрической функции, длина текста или символ "X". Функция: **sin**, **cos**, **tg** или **ctg**, - выбирается с помощью радионабора. Запуск на вычисление производится с помощью контекстного меню. Очистка окна аргумента производится кнопкой. Вторая кнопка заканчивает программу.

1-22. На форме располагаются компоненты: таблица размером 3x3 из компонентов **Entry**, 8 кнопок **SpeedButton** по периметру таблицы и радионабор на два положения. Кнопки перемещают по периметру таблицы символ '\*'. При этом возможно нажатие только следующей по текущему направлению кнопки, которое задается радионабором.

1-23. На форме располагаются: таблица размером 5x5 из компонентов **Entry**, два радионабора для выбора текущей ячейки, кнопка **Button**, редактор **Text**. Все ячейки таблицы заполнены названиями различных предметов (птиц, рыб, животных и т.п.). Кнопка **Button** имеет размер ячейки и перемещается синхронно активной ячейке. При этом на ней отображается изображение текущего предмета, а название предмета дублируется в редакторе **Text**. Кнопка **Close** заканчивает программу.

1-24. На форме располагается таблица размером 2x12 из компонентов **Entry**. Верхний ряд таблицы заполнен названиями месяцев, а нижний – средними температурами по

месяцам, причем значения температур не повторяются. Виджет **SpinBox** охватывает диапазон всех указанных температур. Если при изменении его значение совпадает со средней температурой некоторого месяца, то название месяца появляется в редакторе **Entry** или **Text**. Тип редактора задается окном `message-box.askyesno`, которое вызывается щелчком мыши по форме. Кнопка **Close** заканчивает программу.

1-25. На форме располагаются шесть кнопок **SpeedButton** и шесть элементов **Canvas** с разными рисунками. У каждого такого элемента есть контекстное меню, с помощью которого элемент скрывается, а кнопка **SpeedButton** переходит в нажатое состояние. При этом ранее невидимый элемент показывается на форме, а соответствующая ему кнопка **SpeedButton** отжимается. На форме также находится виджет **SpinBox** на шесть значений. Контекстное меню срабатывает только у того элемента **Canvas**, который указан виджетом **SpinBox**. Кнопка **Close** заканчивает программу.

1-26. На форме располагаются: редактор **Text**, метка **Label**, кнопка **Close**, редактор **Entry** и две линейки **Scale**, используемые для задания нового размера редактора **Text** в диапазоне от 15x20 до 40x50. Задание новой размерности приводят к ее отображению на метке. Если размер редактора меньше 20x30, то в нём копируются только заглавные буквы, введенные в редакторе **Entry**, в противном случае любой текст. Кнопка **Close** заканчивает программу.

1-27. На форме располагается главное меню из названий месяцев года, компонент **ProgressBar**, двенадцать кнопок **Speedbutton**. Выбор некоторого месяца приводит к нажатию соответствующей ему кнопки **Speedbutton**, при этом предыдущая кнопка отжимается. **ProgressBar** устанавливается на позицию, соответствующую номеру месяца. Второй выбор одного и того же месяца приводит к закрытию формы.

1-28. На форме располагаются: четыре компонента **Canvas**; три главных меню по четыре позиции с названиями «Рыбы», «Птицы», «Звери»; радионабор из трех кнопок **Speedbutton** для выбора главного меню. Имя выбранного меню отображается в заголовке формы. Его выбор приводит к загрузке в контейнеры **Canvas** соответствующих изображений фауны. Щелчок по опции главного меню приводит к сокрытию соответствующего ему **Canvas**, а ранее сокрытый становится видимым. Кнопка **Close** заканчивает программу.

- 1-29. На форме располагаются: пять редакторов **Entry**; пять компонентов **Label**; два радионабора по пять кнопок **Speedbutton** для выбора источника и приемника копирования. Выбирается источник копирования – редактор и приемник копирования - метка. Команда копирования – щелчок мыши правой кнопкой по форме. Если копируемый текст – только цифры, то метка окрашивается в желтый цвет, только буквы – зеленый, смешанный текст – коричневый. При попытке копирования текста, содержащего знаки препинания, выводится диалоговое сообщение об ошибке. Программа закрывается командой главного меню.
- 1-30. На форме располагаются: редактор **Text** размеров 25x100, две линейки **Scale** с диапазоном значений 25 и 100, **SpinBox** с диапазоном значений 33, флажок **Checkbutton**. Линейки **Scale** задают позицию символа в редакторе **Text**, а **Spinbox** номер символа в русском алфавите, **Checkbox** - регистр символа. Занесение символа на указанную позицию происходит по двойному клику мыши на форме. Кнопка **Close** заканчивает программу.
- 1-31. На форме располагаются: редактор **Text** размеров 10x20, заполненный текстом из цифр, букв и знаков препинания; компоненты **Entry**, **Progressbar**, **Combobox**, **Button** и **Label**. **Combobox** задает номер строки. Клик мыши по редактору приводит к движению вдоль выбранной строки на один символ, что иллюстрируется движением ползунка **Progressbar**. Если символ является цифрой, то он копируется в редактор **Entry**, буквой – на кнопку **Button**, знаком препинания – на метку **Label**. Смена строки приводит приложение в исходное состояние. Форма закрывается, если движение по строке доходит до конца.
- 1-32. На форме располагаются: четыре компонента **Canvas**; три главных меню по четыре позиции с названиями «Рыбы», «Птицы», «Звери»; радионабор из трех кнопок **Speedbutton** для выбора главного меню. Имя выбранного меню отображается в заголовке формы. Его выбор приводит к загрузке в контейнеры **Canvas** соответствующих изображений фауны. Щелчок по опции главного меню приводит к сокрытию соответствующего ему **Canvas**, а ранее сокрытый становится видимым. Кнопка **Close** заканчивает программу.
- 1-33. На форме располагаются: пять редакторов **Entry**; пять компонентов **Label**; два радионабора по пять кнопок **Speedbutton** для выбора источника и приемника копи-

вания. Выбирается источник копирования – редактор и приемник копирования - метка. Команда копирования – щелчок мыши правой кнопкой по форме. Если копируемый текст – только цифры, то метка окрашивается в желтый цвет, только буквы – зеленый, смешанный текст – коричневый. При попытке копирования текста, содержащего знаки препинания, выводится диалоговое сообщение об ошибке. Программа закрывается командой главного меню.

1-34. На форме располагаются компоненты: два редактора **Entry**, **Label**, главное меню, кнопка **Speedbutton**, две кнопки **Button** (с надписями «Очистить» и «Заккрыть»). В один редактор вводится с проверкой дробное число, а во второе - целое как показатель возведения в степень первого числа. Запуск на вычисление производится с помощью команды главного меню. Результат помещается на **Label**. Кнопка **Speedbutton** убирает/восстанавливает главное меню. Очистка окна аргумента производится кнопкой **Button**. Вторая кнопка заканчивает программу.

1-35. На форме располагаются: таблица размером 5x5 из компонентов **Entry**; главное меню с четырьмя опциями  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ ; кнопка **Speedbutton** и кнопка **Close**. Главное меню меняет координаты активной ячейки, которые отображаются на заголовке формы. При достижении границы таблицы соответствующая опция меню деактивируется и становится рабочей при отходе от границы. В текущую ячейку можно занести дату в формате 01/07/2005 или 01/июл/05. Вид формата меняется кнопкой **Speedbutton**. Кнопка **Close** заканчивает программу.

2-1. На форме располагается таблица размером 5x5 из компонентов **Entry** с контекстным меню. В ячейки таблицы можно заносить только положительные двузначные числа. Контекстное меню запускает сравнение суммы элементов над и под главной диагональю (на главную диагональ заносить информацию нельзя). Если сумма над главной диагональю больше, то над таблицей появляется элемент **Canvas** с размером стороны, равной сумме. На **Canvas** изображена рыба. В противном **Canvas** случае появляется внизу таблицы с рисунком птицы.

2-2. На форме располагаются компоненты: редактор **Entry**, в котором исходно находится английская буква **a**; виджеты **ProgressBar** и **Spinbox**, две кнопки **Button** со стрелками  $\leftarrow$  и  $\Rightarrow$ . Стрелки или значения **Spinbox** задают смену символов от **a** до **z**. Вид управления задается окном `messagebox.askyesno`, которое вызывается щелчком

средней (колесика) кнопки мыши по форме. Смена символов иллюстрируется виджетом **Progressbar**. Окно программы закрывается контекстным меню.

2-3. На форме располагается главное меню на четыре опции. Три первых (птицы, рыбы, звери) из них поочередно активизируют свои компоненты **Canvas**. К каждому такому компоненту привязано контекстное меню для выбора отображаемого на **Canvas** рисунка (по 6 экземпляров). При активизации одного из **Canvas** предыдущий активный виджет становится невидимым. Название отображаемого объекта указывается в заголовке формы. Опция «Закрытие окна» заканчивает программу.

2-4. На форме располагаются две таблицы размером 3x4 из компонентов **Entry** и главное меню для выбора из них активной. В левую таблицу можно вводить только текст, а в правую - отрицательные и положительные вещественные числа. При попытке неправильного ввода внизу соответствующей таблицы загорается красная лампочка (элемент **Canvas**). Активная таблица индицируется радиокнопкой. Кнопка **Speedbutton** заканчивает работу программы.

2-5. На форме располагаются компоненты: редакторы **Entry** и **Text**, **Label**, радионабор из двух кнопок, кнопки **Button** с надписью «Добавить» и **Close**. При запуске программы курсор находится в редакторе **Entry**. В него вводится строка, содержащая только цифры или только буквы. Вид фильтра определяется радионабором. При нажатии кнопки «Добавить» строка добавляется в **Text** или на **Label**. Приемник копирования выбирается с помощью контекстного меню. Кнопка **Close** заканчивает программу.

2-6. На форме располагаются: таблица размером 5x5 из компонентов **Entry**; контекстное меню таблицы с четырьмя опциями  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ ; кнопка **Speedbutton**; редактор **Entry** и кнопка **Close**. Контекстное меню меняет координаты активной ячейки, которые отображаются в заголовке таблицы. При достижении границы таблицы соответствующая опция меню деактивируется и становится рабочей при отходе от границы. В текущую ячейку можно занести через редактор **Entry** дату в формате 01/07/2005 или 01/июл/05. Занесение реализуется через контекстное меню редактора. Вид формата меняется кнопкой **Speedbutton**. Кнопка **Close** заканчивает программу.

2-7. На форме располагаются: таблица размером 2x12 из компонентов **Label**, в первую строку которой занесены названия месяцев; компонент **Canvas** и кнопка **Close**. В

проект введены три контекстных меню: для формы, таблицы и **Canvas**. Контекстное меню формы меняет цвет фона (6 цветов), таблицы - приводит к отображению во второй строчке время года выбранного месяца, **Canvas** - меняет рисунок на компоненте (4 рисунка, соответствующих времени года выбранного месяца). Кнопка **Close** заканчивает программу.

2-8. На форме располагаются компоненты: редактор **Entry**, главное меню на четыре опции, компонент **Label**, контекстное меню для формы и две кнопки **Button** (для очистки редактора и закрытия формы) с подходящими по назначению рисунками. В **Entry** вводится дробное число как аргумент тригонометрической функции. Сама функция: **sin**, **cos**, **tg** или **ctg**, - выбирается с помощью главного меню. Запуск на вычисление производится с помощью контекстного меню. Результат помещается на компонент **Label**.

2-9. На форме располагаются компоненты: два редактора **Entry**, метка **Label**, главное меню, кнопка **Speedbutton**, две кнопки **Button** (с надписями «Очистить» и «Закрыть»). В один редактор вводится с проверкой дробное число, а во второе - целое как показатель возведения в степень первого числа. Неправильный ввод приводит к появлению модальных окон с сообщением об ошибке. Запуск на вычисление производится с помощью команды главного меню. Результат помещается на метку **Label**. Кнопка **Speedbutton** активизирует/деактивирует главное меню. Очистка окон аргументов производится кнопкой **Button**. Вторая кнопка заканчивает программу.

2-10. На форме располагаются компоненты: редакторы **Entry** и **Text**, **Label**, **ListBox**, радиопереключатель в виде двух кнопок **Speedbutton**, кнопка **Button** вида «Да» с рисунком и кнопка **Close**. В редактор **Entry** вводится строка, преобразуемая к верхнему или нижнему регистру. Вид преобразования задается радиопереключателем. При нажатии кнопки «Да» строка копируется в **Text** или на **Label**. Приемник копирования выбирается с помощью **ListBox**. Когда в редакторе **Text** создается семь строк, копирование производится только на **Label**. Очистка **Text** восстанавливает исходную ситуацию. Кнопка **Close** заканчивает программу.

2-11. На форме располагаются: два компонента **ListBox**, таблица размером 6x6 из компонентов **Entry**, кнопка **Speedbutton**, две кнопки **Button** и кнопка **Close**. На одной из кнопок **Button** надпись «Очистка», на другой – «Занесение». Над одним из компонен-

тов **Listbox** надпись «Строки», над другим – «Столбцы». С помощью первого выбирается строка таблицы, другим - столбец. При нажатии кнопки «Занесение» в соответствующую ячейку заносится произведение (если номер столбца не меньше номера строки) или сумма (в противном случае) номеров строки и столбца. При занесении произведения кнопка **Speedbutton** переводится в нажатое состояние, суммы - отжимается. Кнопка «Очистка» очищает данную ячейку. Кнопка **Close** заканчивает программу.

2-12. На форме располагаются компоненты: таблица размером 2x12 из компонентов **Label**, двенадцать пронумерованных кнопок **Speedbutton**, контейнер **Canvas**, радионабор на два положения и кнопка **Close**. Верхний ряд таблицы заполнен названиями месяцев или их номерами-римскими цифрами (вид задается радионабором), нижний - среднемесячной температурой. Нажатие кнопки **Speedbutton** с номером месяца приводит к появлению на **Canvas** зимнего изображения, если температура отрицательна для выбранного месяца, осенне-весеннего – если температура в диапазоне 0÷10, и летнего для других температур.

2-13. На форме располагаются: таблица 6x6 из компонентов **Entry**, два набора радиокнопок с соответствующими надписями для выбора номеров строк и столбцов таблицы, редактор **Entry**, кнопка **Button** с надписью «Занесение» и кнопка **Close**. Выбор ячейки таблицы и нажатие кнопки «Занесение» приводит к появлению в данной ячейке текста из редактора **Entry**. При этом, если произведение номеров строки и столбца не превышает 9, то в соответствующую ячейку копируются только цифры, находится в диапазоне 10÷18 – только русские нижнего регистра, в диапазоне 19÷27 – только английские нижнего регистра, больше 27 – только верхнего регистра. Редактор требует ввода строк длиной от 8 до 18 символов, в противном случае занесение не происходит. Кнопка **Close** заканчивает программу.

2-14. На форме располагаются компоненты: таблица 5x5 из компонентов **Label**, в центральную ячейку которой занесена фамилия студента; четыре кнопки со стрелками ↑, ↓, →, ←; набор кнопок **Speedbutton**, соответствующих каждой строке и столбцу; редактор **Entry** для отображения координат текущей ячейки и кнопка **Close**. Нажатие кнопок со стрелками приводит к перемещению в указанном направлении фамилии,

при этом нажимаются кнопки **Speedbutton**, соответствующие текущим строке и столбцу.

2-15. На форме располагаются: четыре кнопки **Speedbutton** и четыре соответствующие им компоненты **Canvas** с рисунками, две линейки **Scale** и кнопка **Close**. Щелчок мышью по компоненту компоненты **Canvas** активизирует соответствующую ему кнопку **Speedbutton**, остальные кнопки деактивированы. Нажатие этой кнопки приводит к сокрытию соответствующего ей компонента **Canvas**, а отжатие - к его показу. Линейки **Scale** меняют размер текущего **Canvas** в диапазоне 50x50 .. 150x150. Кнопка **Close** заканчивает программу.

2-16. На форме располагаются: семь кнопок **Speedbutton** с надписями-названиями месяцев; линейка **Scale**, размеченная названиями месяцев; кнопка **Close**; главное меню и редактор **Entry**. Перемещение вдоль линейки приводит к нажатию соответствующей кнопки и отображению текущего месяца в редакторе **Entry**, а главное меню задает режим отображения: заглавными или прописными буквами. Кнопка **Close** заканчивает программу.

2-17. На форме располагаются компоненты: редактор **Text** размером 7x7, в левом верхнем углу которого находится латинский символ **a**; четыре кнопки со стрелками  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ , исходно активна только кнопка  $\rightarrow$ ; два виджета **Spinbox** с надписью «Строка» и «Столбец». Нажатие соответствующей стрелки приводит к движению символа по периметру редактора по часовой стрелке. Когда символ доходит до конца стороны редактора происходит активизация следующей стрелки. **Spinbox** показывают текущее значение строки и столбца (позиции символа). Контекстное меню закрывает форму.

2-18. На форме располагаются: три редактора **Entry**, соответствующие им три кнопки **Speedbutton** и три виджета **Spinbox** с максимальным значением 30. Выбор редактора для ввода задается кнопкой **SpeedButton**, а максимальная длина вводимой строки - виджетом **SpinBox**. Если длина ввода превышает установленное значение, то соответствующий редактор и кнопка деактивируются. Очистка каждого редактора происходит своим контекстным меню, после чего он возвращается в исходное состояние. Программа закрывается при установке любого **Spinbox** в нулевое состояние.

- 2-19. На форме располагаются компоненты: редактор **Text** размером 5x5, в центре которого находится символ **0**; четыре кнопки со стрелками  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ ; два виджета **Spinbox** с надписью «Строка» и «Столбец» и начальными значениями 3, назначение которых понятно из их названия. Нажатие соответствующей стрелки приводит к движению символа только по разрешенным строке и столбцу. Когда символ доходит до соответствующей границы редактора происходит деактивация соответствующей стрелки, при отходе от границы - активизация. При смене траектории символ устанавливается на пересечение разрешенных строки и столбца. Контекстное меню закрывает форму.
- 2-20. На форме располагаются: компонент **Canvas** размером 200x200, в левом верхнем углу которого находится рисунок размером 20x20; четыре кнопки со стрелками  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ ; **ListBox** со значениями N, NE, E, SE, SW, W, NW и CENTER; флажок **Combobox**. При установленном флажке кнопками-стрелками рисунок попиксельно перемещается внутри контейнера **Canvas**. При сброшенном флажке кнопки деактивируются и перемещение реализуется стрелками клавиатуры. **Listbox** устанавливает режим привязки рисунка к опорной точке. Закрывает форму двойной клик по кнопке **Close**.
- 2-21. На форме располагаются компоненты: редактор **Text** размером 11x11, в центре которого находится символ **0**; четыре кнопки со стрелками  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ ; два виджета **Spinbox** с надписью «Строка» и «Столбец» и начальными значениями 0, назначение которых понятно из их названия; радионабор на два значения. Нажатие соответствующей стрелки приводит к движению символа в заданном направлении. Движение вверх увеличивает на 1 значение виджета «Столбец», вниз - уменьшает на это же значение. Соответственно, движение влево-вправо меняет значение виджета «Строка». Когда символ доходит до соответствующей границы редактора происходит деактивация соответствующей стрелки, при отходе от границы - активизация. Радионабор переключает управление между кнопками со стрелками формы и клавиатуры. Главное меню закрывает форму.
- 2-22. На форме располагаются компоненты: последовательно семь редакторов **Entry**, линейка **Scale** на семь положений, контекстное меню у каждого редактора, главное меню для закрытия формы. Линейка **Scale** используется для указания активного редак-

тора, в который можно вводить информацию. Остальные редакторы деактивированы. Если редактор активизируется при движении вперед линейки **Scale**, то в редактор можно вводить только буквы, назад – только цифры. Контекстные меню используются для очистки редакторов.

2-23. На форме располагаются компоненты: редактор **Text** размером 11x11, в центре которого находится символ **0**; четыре кнопки со стрелками ↖, ↗, ↘, ↙; два виджета **Spinbox** с надписью «Строка» и «Столбец» и начальными значениями 0, назначение которых понятно из их названия. Нажатие стрелок ↘ и ↖ приводит к движению символа по главной диагонали, ↗ и ↙ - по вспомогательной. Переход с одной диагонали на другую возможен только в центре. Движение вверх увеличивает на 1 значение виджета «Столбец», вниз - уменьшает на это же значение. Соответственно, движение влево-вправо меняет значение виджета «Строка». Когда символ доходит до соответствующего угла редактора происходит деактивация соответствующей стрелки, при отходе от границы - активизация. Название диагонали отображается на метке. Главное меню закрывает форму.

2-24. На форме располагаются: компонент **Canvas** размером 200x200, в левом верхнем углу которого находится рисунок размером 20x20; две радиогруппы по четыре опции («Влево», «Вправо», «Вверх», «Вниз») с подписями «Окно» и «Рисунок»; линейка **Scale** на три положения. Первая радиогруппа предназначена для смещения по пространству формы контейнера **Canvas**, вторая – внутри контейнера рисунка. Линейка **Scale** задает дискрет перемещения. Название перемещаемого в данный момент компонента отображается в заголовке формы. Закрывает форму двойной клик по кнопке **Close**.

2-25. На форме располагаются компоненты: редактор **Text** размером 6x6, в левом верхнем углу которого находится латинский символ **a**; четыре кнопки со стрелками ↑, ↓, →, ←; два виджета **Scale** с надписью «Строка» и «Столбец»; две метки с индикаторами: зеленым и красным, - и надписями: «По часовой», «Против часовой». Символ может двигаться только по периметру редактора по нажатию соответствующей стрелки. **Scale** показывают текущее значение строки и столбца (позиции символа). При движении по часовой стрелке видима кнопка «По часовой», в обратном направлении – «Против часовой». Контекстное меню закрывает форму.

- 2-26. На форме располагаются: компонент **Canvas** размером 200x200, в левом верхнем углу которого находится рисунок размером 40x40; редактор **Text** размером 5x5, в левом верхнем углу которого находится латинский символ **a**; четыре кнопки со стрелками  $\uparrow$ ,  $\downarrow$ ,  $\rightarrow$ ,  $\leftarrow$ ; две линейки **Scale** на пять положений. Кнопки со стрелками синхронно перемещают символ в редакторе и рисунок в контейнере **Canvas** строго по периметру, а линейки **Scale** показывают текущее положение фигур по горизонтали и вертикали. Закрывает приложение двойной клик по форме.
- 2-27. На форме располагаются: таблица размером 4x7 из компонентов **Label**, в каждой строке которой свой набор элементов: "Цветы", "Деревья", "Реки", "Города"; четыре набора главных меню, соответствующие строкам таблицы; две линейки **Scale** для изменения координат текущей ячейки таблицы; кнопка **Close**. Изменение номера строки делает активным соответствующее ей меню, а изменение номера столбца деактивирует соответствующую ему опцию меню. Кнопка **Close** заканчивает программу.
- 2-28. На форме располагаются: календарь на текущий месяц из компонентов **Button** с горизонтальным расположением недель, сверху группа из семи кнопок **Speedbutton**, размеченная днями недели; редактор **Entry**, используемый для ввода числа месяца. Ввод дня в редактор и активация контекстного меню приводят к нажатию кнопки **Speedbutton**, соответствующей выбранному дню недели. Этот же день дублируется в заголовке формы. Если при вводе числа произошла ошибка, выводится окно `showerror()`. Кнопка **Close** заканчивает программу.
- 2-29. На форме располагается таблица размером 6x7 из компонентов **Entry**. Столбцы таблицы с помощью меток поименованы названиями факультетов, а строки - номерами курсов. Таблица заполнена реальными названиями групп. Редактор **Entry** позволяет вводить только название группы. Если оно совпадает с существующим в таблице названием, на элементе **Canvas** появляется изображение декана данного факультета, а название данного факультета отображается в заголовке формы. Кнопка **Close** заканчивает программу.
- 2-30. На форме располагаются: таблица размером 2x12 из компонентов **Entry**, в первую строку которой занесены названия месяцев; компонент **Canvas** и кнопка **Close**. В проект введены три контекстных меню: для формы, таблицы и **Canvas**. Контекстное меню формы меняет цвет ее фона (6 цветов), таблицы - приводит к отображению во

второй строчке количества дней в выбранном месяце, **Canvas** - меняет рисунок на компоненте (5 рисунков). При этом в заголовке формы указывается, какого компонента меню активизировалось. Кнопка **Close** заканчивает программу.

2-31. На форме располагаются: таблица размером 3x3 из компонентов **Entry**, два радионабора для выбора текущей ячейки, кнопка **Button**, редактор **Text**. Все ячейки таблицы заполнены названиями различных предметов (птиц, рыб, животных и т.п.). Кнопка **Button** имеет размер ячейки и перемещается синхронно активной ячейке. При этом на ней отображается изображение текущего предмета, а название предмета дублируется в редакторе **Text**. Кнопка **Close** заканчивает программу.

2-32. На форме располагаются шесть кнопок **Speedbutton** и шесть элементов **Canvas** с разными рисунками. У каждого такого элемента есть контекстное меню, с помощью которого элемент скрывается, а кнопка **Speedbutton** переходит в нажатое состояние. При этом ранее невидимый элемент показывается на форме, а соответствующая ему кнопка **Speedbutton** отжимается. На форме также находится виджет **Spinbox** на шесть значений. Контекстное меню срабатывает только у того элемента **Canvas**, который указан виджетом **Spinbox**. Кнопка **Close** заканчивает программу.

2-33. На форме располагаются: редактор **Text**, метка **Label**, кнопка **Close**, редактор **Entry** и две линейки **Scale**, используемые для задания нового размера редактора **Text** в диапазоне от 15x20 до 40x50. Задание новой размерности приводят к ее отображению на метке. Если размер редактора меньше 20x30, то в нём копируются только заглавные буквы, введенные в редакторе **Entry**, в противном случае любой текст. Кнопка **Close** заканчивает программу.

2-34. На форме располагается главное меню из названий месяцев года, компонент **Progressbar**, двенадцать кнопок **Speedbutton**. Выбор некоторого месяца приводит к нажатию соответствующей ему кнопки **SpeedButton**, при этом предыдущая кнопка отжимается. **Progressbar** устанавливается на позицию, соответствующую номеру месяца. Второй выбор одного и того же месяца приводит к закрытию формы.

2-35. На форме располагаются: таблица размером 4x7 из компонентов **Label**, в каждой строке которой свой набор элементов: "Цветы", "Деревья", "Реки", "Города"; четыре набора главных меню, соответствующие строкам таблицы; две линейки **Scale** для изменения координат текущей ячейки таблицы; кнопка **Close**. Изменение номера строки делает активным соответствующее ей меню, а изменение номера столбца дезактивирует соответствующую ему опцию меню. Кнопка **Close** заканчивает программу.

## Задание № 4 Управление временем в Python-проектах

### Теоретический раздел

При создании динамического изображения в программных проектах требуется механизм управления временем. Модуль `tkinter` позволяет реализовать это управление двумя способами:

1. Метод `after()` откладывает выполнение какого-либо кода на заданный промежуток времени. Его синтаксис имеет вид:

`<окно>.after(<временная задержка в мс>[, <функция>])`,

т.е. выполнение `<функции>` задерживается на время, указанное в `(<временной задержке в мс>`.

Периодическую задержку можно организовать:

а) через цикл с заранее известным числом повторений. Здесь также возможны два варианта.

Обобщенный алгоритм первого варианта (рис. 4.1):

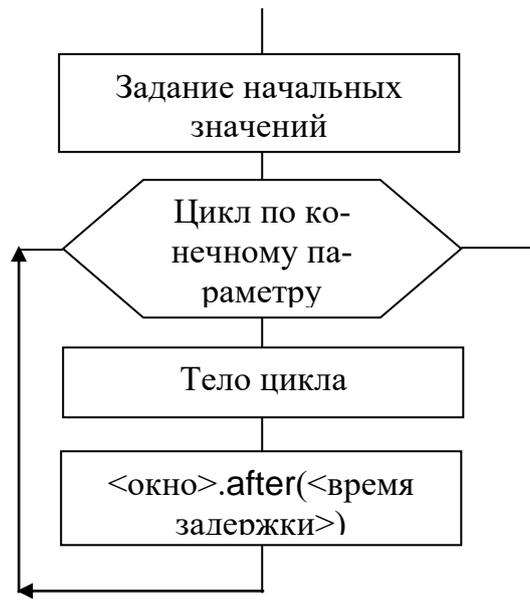


Рис. 4.1

Для того чтобы изображение реконфигурировалось на каждом шаге цикла, необходимо после очередного перестроения активизировать метод `<окно>.update_idletasks()`.

Обобщенный алгоритм второго варианта представлен на рис. 4.2:

При этом предполагается, что аргумент `<команда>` метода `after` запускает некоторую функцию, находящуюся вне цикла.



Рис. 4.2

б) цикл на основе рекурсивной структуры. Для его запуска и остановки требуется глобальная переменная-переключатель, которая устанавливается в соответствующие значения отдельными процедурами (рис. 4.3а и 4.б). А переключатель управляет работой процедуры таймера (рис. 4.3в).

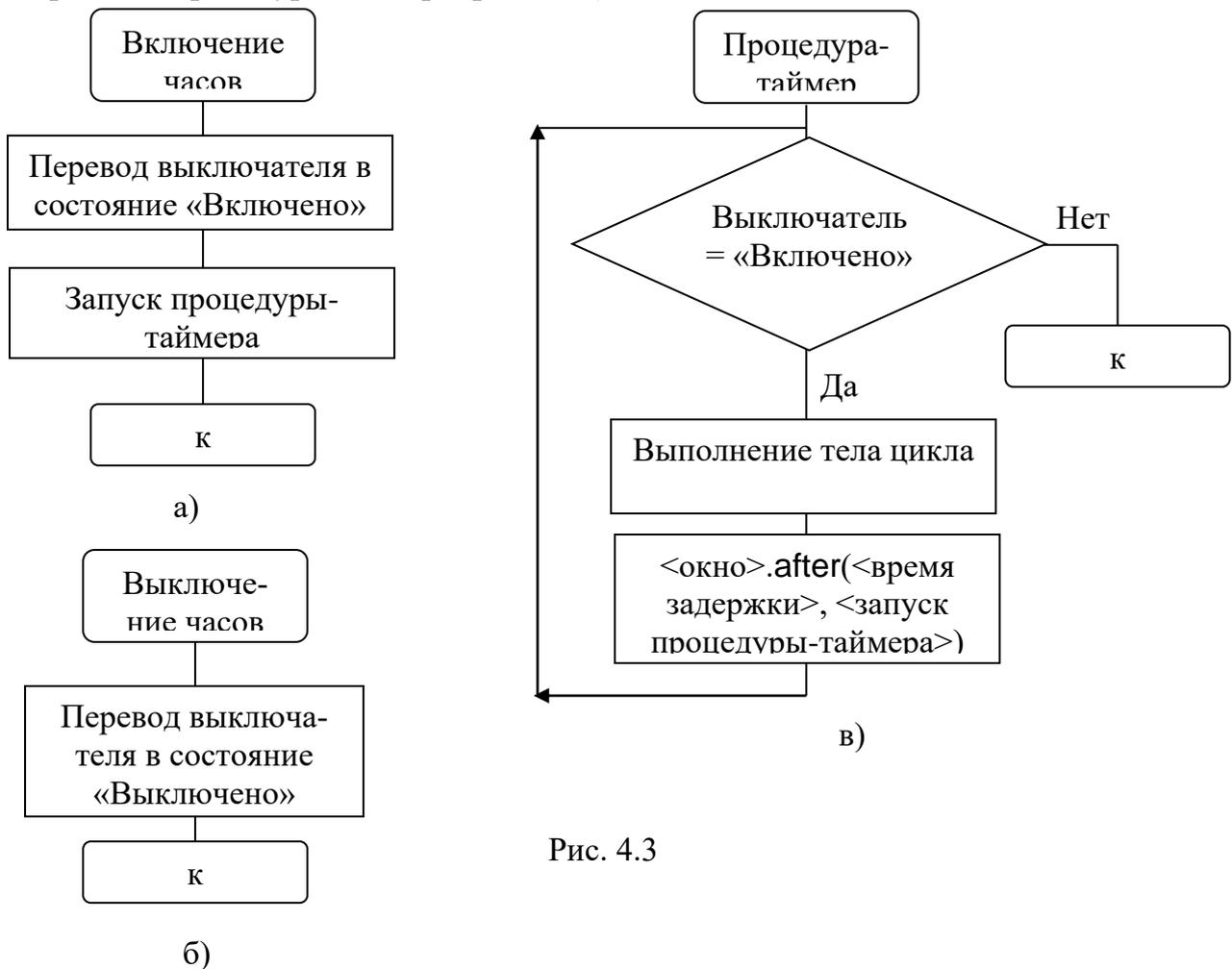


Рис. 4.3

Этот вариант не требует использования метода `update_idletasks()` для корректного отображения динамики

2. Метод `sleep()` также позволяет создавать временные задержки требуемой длительности. Его синтаксис имеет вид:

```
time.sleep(<задержка в сек.>)
```

Из записи видно, этот метод принадлежит модулю `time`, т.е. это модуль должен быть импортирован в начале скрипта.

Метод `sleep()` также требует для корректной работы поддержки методом `update_idletasks()`.

Изучение методов работы со временем эффективней всего проводить на примере динамической графики, поэтому ниже рассматриваются средства создания простейших изображений в рамках модуля `tkinter`.

Для создания подобных изображений в `tkinter` используется контейнер `Canvas` (канва), основы создания которого рассмотрены в предыдущей лабораторной работе.

На канву можно выводить текст командой, синтаксис которой имеет вид:

```
<имя канвы>.create_text(<координаты опорной точки>, text="<отображаемый текст>",  
                        [<параметры шрифта>])
```

Ниже приведен пример размещения на канве трех строк шрифтом `Verdana`, размер которого равен 14 пунктам. Текст выравнивается по центру относительно точки привязки с координатами `x=100, y=100`.

```
c.create_text(100, 100, text="Hello World,\nPython\nand Tk", justify=CENTER,  
              font="Verdana 14 italic")
```

Основные параметры текста:

- `fill` – цвет текста;
- `font` – шрифт, включает название шрифта, его размер в пунктах и стиль;
- `anchor` – выравнивание текста относительно точки привязки (см. предыдущую лабораторную работу);
- `activefill` – цвет текста при наведении на него мыши.

Для того, чтобы в дальнейшем можно было менять параметры данного фрагмента текста, ему должно быть присвоено имя переменной.

Прямоугольник в простейшем случае формируется командой:

```
<имя канвы>. create_rectangle(<координаты левого верхнего угла>, < координаты пра-  
                               вого нижнего угла >, [<параметры прямоугольника>])
```

Основные параметры текста:

- ❑ **outline** - цвет контура прямоугольника;
- ❑ **fill** - цвет заливки;
- ❑ **activefill** - цвет заливки при наведении на прямоугольник мыши;
- ❑ **activeoutline** - цвет контура прямоугольника при заходе на него мыши;
- ❑ **width** - ширина контура фигуры в пикселях.

Графические средства **tkinter** позволяют создавать ломаную линию произвольной конфигурации. Одиночный отрезок создается командой `c.create_line(10, 10, 190, 50)`, т.е. по синтаксису:

```
<имя канвы>.create_line(<координаты конца 1>, <координаты конца 2>)
```

Ломаная линия произвольно длины создается по синтаксису:

```
<имя канвы>.create_line(<массив точек>)
```

Основные параметры линии:

- ❑ **fill** - цвет линии;
- ❑ **activefill** - цвет линии при наведении на прямоугольник мыши;
- ❑ **width** - ширина линии фигуры в пикселях;
- ❑ **smooth** – сглаживание линии.

Ниже приведен пример создания линии, когда выделяется отдельно каждая точка излома:

```
c.create_line([300,80],[400,80],[450,75],[450,200],[300,180],[330,160])
```

Но такой же результат будет получен и при следующем задании линии:

```
c.create_line(300,80,400,80,450,75,450,200,300,180,330,160)
```

Параметр **smooth** по умолчанию установлен в значение 0, что соответствует излому линии в каждой точке перегиба. При **smooth=1** в точках перегиба линия сглаживается.

Метод **create\_polygon** контейнера **Canvas** близок по своему действию к предыдущему методу, но здесь первая и последняя точки линии автоматически соединяются, а замкнутые поверхности закрашиваются установленным цветом. Поэтому у данного метода параметр **fill** определяет цвет заливки замкнутого контура, а **outline** – цвет самого контура. Актуален также параметр **smooth**.

Окружности на канве создаются методом **create\_oval**, у которого в начале параметров указываются координаты левого верхнего и правого нижнего углов описывающего окружность прямоугольника:

`c.create_oval(50, 10, 150, 110, width=2)`

Все параметры окружности аналогичны по своему назначению параметрам прямоугольника или полигона.

Рассмотренные фигуры можно перемещать по пространству канвы с помощью метода `move`:

`<канва>, move(<имя фигуры>, <смещение по оси X>, <смещение по оси Y>)`

Из синтаксиса метода следует, что для реализации такого преобразования у каждой фигуры необходим идентификатор, который должен присваиваться ей в момент создания.

### *Постановка задачи*

В соответствии с заданием выбрать вариант реализации программного проекта. Во всех вариантах задания графический объект создается на базе контейнера `Canvas`. На канве создается статическое изображение с динамическим компонентом-рамкой. Толщина рамки три пикселя.

### *Методика (алгоритм) выполнения задания*

- I. Из раздела «Варианты задания» в соответствии с вариантом (номером в списке группы: первая цифра-номер группы, далее идет номер студента в группе) выбрать задание на разработку программного проекта:
- II. В «Теоретическом разделе» изучить необходимые средства модуля `tkinter.ttk` для выполнения определенного вариантом задания.
- III. Разработать графический интерфейс средствами языка Python для выполнения определенного вариантом задания.
- IV. Разработать программные модули на языке Python для реализации требуемого в задании функционала.
- V. Привязать разработанный функционал к программному интерфейсу.
- VI. Произвести отладку полученного программного приложения.
- VII. Предоставить преподавателю возможность проверки корректности выполнения задания в дистанционном или очном варианте.
- VIII. Оформить отчет по выполненному заданию, содержащий:
  - 1) задание;
  - 2) структуру проекта;

- 3) блок-схему начальной установки приложения, блок-схемы процедур.
- 4) листинг программы.

### *Варианты задания*

- 1-1. На пространстве формы изображен календарь за январь текущего года с горизонтальным расположением недель. Дни недели подписаны. Квадратная рамка движется циклически по датам с дискретом времени 0.4 сек. Цвет рамки – “black”. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп». Кнопка «Пуск» продолжает движение.
- 1-2. На пространстве формы изображен календарь за февраль текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется по датам с дискретом времени 0.5 сек. Запуск движения – контекстное меню, остановка – кнопка «Стоп», что приводит к установке рамки на первую дату.
- 1-3. На пространстве формы изображен календарь за март текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде пятиугольника циклически движется по датам с дискретом времени 0.6 сек. Запуск движения – кнопка «Пуск». Кнопка «Пуск» продолжает движение.
- 1-4. На пространстве формы изображен календарь за апрель текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам с дискретом времени 0.7 сек. Запуск движения – кнопка «Пуск», остановка – главное меню, что приводит к установке рамки на первую дату.
- 1-5. На пространстве формы изображен календарь за май текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата с закругленными углами циклически движется по датам с дискретом времени 0.4 сек. Запуск движения – главное меню, остановка – кнопка «Стоп». Кнопка «Пуск» продолжает движение.
- 1-6. На пространстве формы изображен календарь за июнь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам с дискретом времени 0.5 сек. Запуск движения и остановка движения – команды главного меню.

- 1-7. На пространстве формы изображен календарь за июль текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется по датам с дискретом времени 0.6 сек. Запуск движения и остановка движения – команды контекстного меню.
- 1-8. На пространстве формы изображен календарь за август текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде пятиугольника циклически движется по датам с дискретом времени 0.7 сек. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп», что приводит к установке рамки на первую дату.
- 1-9. На пространстве формы изображен календарь за сентябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам с дискретом времени 0.8 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню. Кнопка «Пуск» продолжает движение.
- 1-10. На пространстве формы изображен календарь за октябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам с дискретом времени 0.4 сек. Запуск движения – команда главного меню, остановка – кнопка «Стоп», что приводит к установке рамки на первую дату.
- 1-11. На пространстве формы изображен календарь за ноябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам с дискретом времени 0.5 сек. Запуск движения – кнопка «Пуск», остановка – команда контекстного меню. Кнопка «Пуск» продолжает движение.
- 1-12. На пространстве формы изображен календарь за декабрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам с дискретом времени 0.6 сек. Запуск движения – команда контекстного меню, остановка – кнопка «Стоп», что приводит к установке рамки на первую дату.
- 1-13. На пространстве формы изображен календарь за январь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата с за-

круглыми углами циклически движется по датам с дискретом времени 0.7 сек. Запуск движения и остановка движения – команды контекстного меню.

1-14. На пространстве формы изображен календарь за февраль текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется по датам с дискретом времени 0.8 сек. Запуск движения – команда главного меню, остановка – команда контекстного меню, что приводит к установке рамки на первую дату.

1-15. На пространстве формы изображен календарь за март текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам с дискретом времени 0.4 сек. Запуск движения – команда контекстного меню, остановка – команда главного меню.

1-16. На пространстве формы изображен календарь за апрель текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде пятиугольника циклически движется по датам с дискретом времени 0.5 сек. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп», что приводит к установке рамки на первую дату.

1-17. На пространстве формы изображен календарь за май текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам с дискретом времени 0.6 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню.

1-18. На пространстве формы изображен календарь за июнь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам с дискретом времени 0.7 сек. Запуск движения – команда главного меню, остановка – кнопка «Стоп», что приводит к установке рамки на первую дату.

1-19. На пространстве формы изображен календарь за июль текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата с закругленными углами циклически движется по датам с дискретом времени 0.8 сек. Запуск движения – команда главного меню, остановка – кнопка.

1-20. На пространстве формы изображен календарь за август текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется по датам с дискретом времени 0.4 сек. Запуск движения – кнопка.

ка «Пуск», остановка – команда контекстного меню, что приводит к установке рамки на первую дату.

1-21. На пространстве формы изображен календарь за сентябрь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам с дискретом времени 0.5 сек. Запуск движения – команда контекстного меню, остановка – кнопка «Стоп».

1-22. На пространстве формы изображен календарь за октябрь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде пятиугольника циклически движется по датам с дискретом времени 0.6 сек. Запуск движения и остановка движения – команды контекстного меню, что приводит к установке рамки на первую дату.

1-23. На пространстве формы изображен календарь за ноябрь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам с дискретом времени 0.7 сек. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп». Кнопка «Пуск» продолжает движение.

1-24. На пространстве формы изображен календарь за декабрь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам с дискретом времени 0.8 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню, что приводит к установке рамки на первую дату.

1-25. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 6x6. Метка в виде квадрата с закругленными углами исходно находится в левом верхнем углу, а после запуска циклически движется по траектории вида  с дискретом времени 0.8 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню..

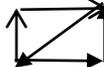
1-26. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 6x6. Метка в виде овала исходно находится в левом верхнем углу, а после запуска циклически движется по периметру матрицы по часовой стрелке с дискретом времени 0.8 сек. После окончания полного цикла рамка движется против часовой стрелки, а затем снова по часовой. Запуск движения – двойной щелчок левой клавиши мыши по форме, остановка – команда главного меню, что приводит к установке рамки в исходное положение.

1-27. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 6x6. Метка в виде квадрата с закругленными углами исходно находится в левом верхнем углу, а после запуска циклически движется по траектории вида  с дискретом времени 0.8 сек. Запуск движения – щелчок правой клавиши мыши по контейнеру Canvas, остановка – команда главного меню.

1-28. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 7x7. Метка в виде ромба исходно находится в начале четвертого столбца, а после запуска циклически движется сначала против часовой стрелки, потом по часовой, по траектории вида  с дискретом времени 0.8 сек. Запуск/остановка движения – радионабор из двух радиокнопок. Остановка приводит к установке рамки в исходное положение.

1-29. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 7x7. Метка в виде пятиугольника исходно находится в начале четвертого столбца, а после запуска циклически движется сначала по часовой стрелке, потом против часовой, по траектории вида  с дискретом времени 0.8 сек. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп». Кнопка «Пуск» продолжает движение.

1-30. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 7x7. Метка в виде ромба исходно находится в левом верхнем углу, а после запуска циклически движется по периметру верхней треугольной матрицы по часовой стрелке, а нижней треугольной – против часовой с дискретом времени 0.8 сек. Запуск движения – двойной щелчок левой клавиши мыши по форме, остановка – команда главного меню, что приводит к установке рамки в исходное положение.

1-31. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 7x7. Метка в виде квадрата с закругленными углами исходно находится в правом верхнем углу, а после запуска циклически движется по траектории вида  с дискретом времени 0.8 сек. Запуск/остановка движения – кнопка SpeedButton.

1-32. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 7x7. Метка в виде ромба исходно находится в начале четвертого столбца, а после запуска циклически движется сначала против часовой стрелки, потом по часовой, по траектории вида  с дискретом времени 0.8 сек. Запуск/остановка движения – радионабор из двух радиокнопок. Остановка приводит к установке рамки в исходное положение.

1-33. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 7x7. Метка в виде пятиугольника исходно находится в начале четвертого столбца, а после запуска циклически движется сначала по часовой стрелке, потом против часовой, по траектории вида  с дискретом времени 0.8 сек. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп». Кнопка «Пуск» продолжает движение.

1-34. На пространстве формы изображен календарь за январь текущего года с горизонтальным расположением недель. Дни недели подписаны. Квадратная рамка движется циклически по датам с дискретом времени 0.4 сек. Цвет рамки – “black”. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп», что приводит к установке рамки на первую дату.

1-35. На пространстве формы изображен календарь за февраль текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется по датам с дискретом времени 0.5 сек. Запуск движения – контекстное меню, остановка – кнопка «Стоп».

2-1. На пространстве формы изображен календарь за январь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.4 сек. Запуск движения и остановка движения – радионабор из двух кнопок. Остановка приводит к установке рамки на последнюю дату.

2-2. На пространстве формы изображен календарь за февраль текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата с закругленными углами циклически движется в обратном направлении по датам, начиная с последней, с дискретом времени 0.8 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню. Кнопка «Пуск» продолжает движение.

2-3. На пространстве формы изображен календарь за март текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется в обратном направлении по датам, начиная с последней, с дискретом 0.6 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню, что приводит к установке рамки на последнюю дату.

2-4. На пространстве формы изображен календарь за апрель текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам, начиная с последней, в обратном направлении с дискретом

времени 0.7 сек. Запуск движения – команда главного меню, остановка – кнопка «Стоп».

2-5. На пространстве формы изображен календарь за май текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде пятиугольника циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.6 сек. Запуск и остановка движения – команды главного меню.

2-6. На пространстве формы изображен календарь за июнь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.5 сек. Запуск движения – команда контекстного меню, остановка – кнопка «Стоп», что приводит к установке рамки на последнюю дату.

2-7. На пространстве формы изображен календарь за июль текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам в обратном направлении с дискретом времени 0.5 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню. Кнопка «Пуск» продолжает движение.

2-8. На пространстве формы изображен календарь за август текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата с закругленными углами циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.5 сек. Запуск и остановка движения – команды главного меню.

2-9. На пространстве формы изображен календарь за сентябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется по датам, начиная с последней, в обратном направлении с дискретом 0.4 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню, что приводит к установке рамки на последнюю дату.

2-10. На пространстве формы изображен календарь за октябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам, начиная с последней, в обратном направлении с дискретом 0.7 сек. Запуск программы и ее остановка - контекстное меню.

2-11. На пространстве формы изображен календарь за ноябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде пятиугольника

циклически движется по датам, начиная с последней, в обратном направлении с дискретом 0.9 сек. Запуск движения – щелчок мыши по форме, остановка – двойной щелчок. При остановке рамка устанавливается на последнюю дату.

2-12. На пространстве формы изображен календарь за декабрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам, начиная с последней, в обратном направлении с дискретом 0.8 сек. Запуск и остановка движения – команды главного меню.

2-13. На пространстве формы изображен календарь за январь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.4 сек. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп», что приводит к установке рамки на последнюю дату.

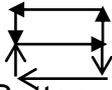
2-14. На пространстве формы изображен календарь за февраль текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата с закругленными углами циклически движется в обратном направлении по датам, начиная с последней, с дискретом времени 0.8 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню. Кнопка «Пуск» продолжает движение.

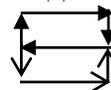
2-15. На пространстве формы изображен календарь за март текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется в обратном направлении по датам, начиная с последней, с дискретом 0.6 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню, что приводит к установке рамки на левую последнюю дату.

2-16 На пространстве формы изображен календарь за апрель текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.7 сек. Запуск движения – команда главного меню, остановка – кнопка «Стоп».

2-17. На пространстве формы изображен календарь за май текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.5 сек. Запуск движения – команда контекстного меню, остановка – кнопка «Стоп», что приводит к установке рамки на последнюю дату.

- 2-18. На пространстве формы изображен календарь за июнь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам в обратном направлении с дискретом времени 0.5 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню.
- 2-19. На пространстве формы изображен календарь за июль текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата с закругленными углами циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.5 сек. Запуск и остановка движения – команды главного меню.
- 2-20. На пространстве формы изображен календарь за август текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется по датам, начиная с последней, в обратном направлении с дискретом 0.4 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню, что приводит к установке рамки на последнюю дату.
- 2-21. На пространстве формы изображен календарь за сентябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам, начиная с последней, в обратном направлении с дискретом 0.7 сек. Запуск программы и ее остановка - контекстное меню.
- 2-22. На пространстве формы изображен календарь за октябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде пятиугольника циклически движется по датам, начиная с последней, в обратном направлении с дискретом 0.9 сек. Запуск движения – щелчок мыши по форме, остановка – двойной щелчок. При остановке рамка устанавливается на последнюю дату.
- 2-23. На пространстве формы изображен календарь за ноябрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам, начиная с последней, в обратном направлении с дискретом 0.8 сек. Запуск и остановка движения – команды главного меню.
- 2-24. На пространстве формы изображен календарь за декабрь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам, начиная с последней, в обратном направлении с дискретом времени 0.4 сек. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп», что приводит к установке рамки на последнюю дату.

2-25. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 7x7. Метка в виде квадрата с закругленными углами исходно находится в начале четвертой строки, а после запуска циклически движется сначала против часовой стрелки, потом по часовой, по траектории вида  с дискретом времени 0.8 сек. Запуск/остановка движения – кнопка **SpeedButton**.

2-26. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 7x7. Метка в виде овала исходно находится в начале четвертой строки, а после запуска циклически движется сначала против часовой стрелки, потом по часовой, по траектории вида  с дискретом времени 0.6 сек. Запуск/остановка движения – кнопка **Checkbutton**. Остановка приводит к установке рамки в исходное положение.

2-27. На пространстве формы изображен календарь за декабрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам с дискретом времени 0.6 сек. Запуск движения – команда контекстного меню, остановка – кнопка «Стоп».

2-28. На пространстве формы изображен календарь за январь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата с закругленными углами циклически движется по датам с дискретом времени 0.7 сек. Запуск движения и остановка движения – команды контекстного меню. Остановка приводит к установке рамки на первую дату.

2-29. На пространстве формы изображен календарь за март текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде ромба циклически движется по датам с дискретом времени 0.4 сек. Запуск движения – команда контекстного меню, остановка – команда главного меню.

2-30. На пространстве формы изображен календарь за июль текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата с закругленными углами циклически движется по датам с дискретом времени 0.8 сек. Запуск движения – команды главного меню, остановка – кнопка, что приводит к установке рамки на первую дату.

2-31. На пространстве формы изображен календарь за август текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде окружности циклически движется по датам с дискретом времени 0.4 сек. Запуск движения – кнопка

ка «Пуск», остановка – команда контекстного меню. Кнопка «Пуск» продолжает движение.

2-32. На пространстве формы изображен календарь за май текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде шестиугольника циклически движется по датам с дискретом времени 0.6 сек. Запуск движения – кнопка «Пуск», остановка – команда главного меню, что приводит к установке рамки на первую дату.

2-33. На пространстве формы изображен календарь за июнь текущего года с вертикальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам с дискретом времени 0.7 сек. Запуск движения – команда главного меню, остановка – кнопка «Стоп».

2-34. На пространстве формы изображена матрица случайных чисел от 0 до 99 размером 6х6. Метка в виде овала исходно находится в левом верхнем углу, а после запуска циклически движется по периметру матрицы по часовой стрелке с дискретом времени 0.8 сек. После окончания полного цикла рамка движется против часовой стрелки, а затем снова по часовой. Запуск движения – двойной щелчок левой клавиши мыши по форме, остановка – команда главного меню, что приводит к установке рамки в исходное положение.

2-35. На пространстве формы изображен календарь за декабрь текущего года с горизонтальным расположением недель. Дни недели подписаны. Метка в виде квадрата циклически движется по датам с дискретом времени 0.6 сек. Запуск движения – команда контекстного меню, остановка – кнопка «Стоп».

## Задание № 5 Графика на основе канвы

### Теоретический раздел

Графический пользовательский интерфейс предполагает широкое использование динамики, что повышает информативность взаимодействия, расширяет изобразительные возможности интерфейса, снижает требования к квалификации пользователя. Строго говоря, формирование такого инструментария должно происходить с участием специалистов по дизайну и эргономике, но в подавляющем большинстве случаев весь процесс осуществляется только силами прикладных программистов, использующих графические средства языка реализации программной системы.

Ранее уже фрагментарно рассматривались возможности канвы модуля `tkinter`, представляющей графическое полотно заданного размера, состоящее из отдельных пикселей. Это полотно является объектом класса `Canvas`. У данного класса есть свои свойства и методы, с помощью которых можно создавать графические изображения. Для этого каждый пиксель может принимать один из  $2^{24}$  возможных цветовых оттенков.

При размещении геометрических примитивов и других объектов указываются их координаты на холсте в пикселях. Точкой отсчета является верхний левый угол канвы (рис. 5.1). Поскольку единицей размера является пиксель, то при изменении разрешения меняются и размеры рисунка.

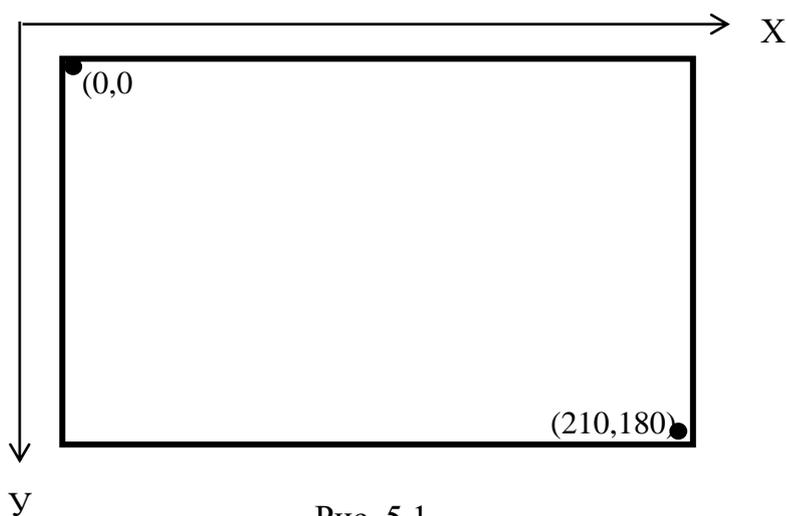


Рис. 5.1

Обычно начальными значениями координат являются (0, 0).  
К рассмотренным ранее параметрам отрезка линии необходимо также добавить свойства:

- `arrow` – указание на формирование отрезка-стрелки.

Значениями параметра `arrow` могут быть `LAST`, `FIRST`, `BOTH`. `FIRST` – стрелка направлена вперед, `LAST` – назад, `BOTH` – двунаправленная стрелка;

- `arrowshape=" 1п 2п 3п"` - параметры наконечника стрелки (рис. 5.2). Например, `arrowshape="10 20 10"`;
- `dash` – параметры пунктира, например `dash=(10,2)` - тире длиной 10 пикселей и пробел 2 пикселя. Данная установка срабатывает только при толщине линии в 1 пиксель.

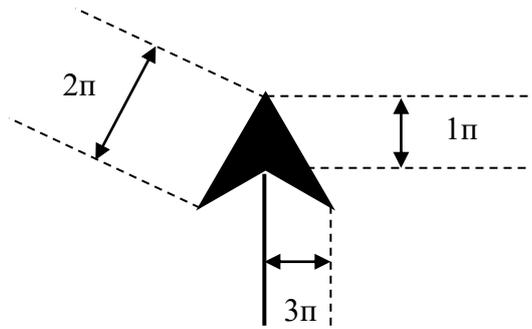


Рис. 5.2

Ниже приведен пример создания пунктирного отрезка стрелки:

```
c.create_line(100, 180, 100, 60, fill='green', width=5, arrow=LAST, dash=(10,2),
             activefill='lightgreen', arrowshape="10 20 10")
```

Новыми получаются фигуры при использовании метода `create_arc()`. В зависимости от значения опции `style` можно получить сектор (по умолчанию), сегмент (CHORD) или дугу (ARC). Также как в случае `create_oval()` координаты задают прямоугольник, в который вписана окружность (или эллипс), из которой "вырезают" сектор, сегмент или дугу. Опции `start` присваивается градус начала фигуры, `extent` определяет угол поворота.

```
c.create_oval(10, 10, 190, 190, fill='lightgrey', outline='white')
c.create_arc(10, 10, 190, 190, start=0, extent=45, fill='red')
c.create_arc(10, 10, 190, 190, start=180, extent=25, fill='orange')
c.create_arc(10, 10, 190, 190, start=240, extent=100, style=CHORD, fill='green')
c.create_arc(10, 10, 190, 190, start=160, extent=-70, style=ARC, outline='darkblue',
            width=5)
```

В Tkinter существует два способа "пометить" фигуры, размещенные на холсте, – это идентификаторы, рассмотренные еще в предыдущей лабораторной работе, и теги. Первые всегда уникальны для каждого объекта. Два объекта не могут иметь одни и тот же идентификатор. Теги не уникальны. Группа объектов на холсте может иметь один и тот же тег. Это дает возможность менять свойства всей группы. Отдельно взятая фигура на Canvas может иметь как идентификатор, так и тег.

Кроме рассмотренного ранее метода `move` модифицировать графические объекты также можно на основе методов:

- `itemconfig` изменяет указанные свойства объектов (`<канва>.itemconfig(<имя фигуры>, {<параметр> = <новое значение>}`);

- `coords` изменяет координаты (ими можно менять и размер объекта) (`<канва>.coords(<имя фигуры>, {<новая координата>})`). Если указывается только идентификатор или тег, то `coords()` возвращает текущие координаты объекта.

Предположим, что происходит перемещение прямоугольника командой

```
t = c.create_rectangle(x1, 20, x1+40, 60, outline='white', width=3)
```

Оператор `m=c.coords(t)` будет формировать на каждом шаге перемещения массив из четырех значений-координат этого прямоугольника. Доступ к каждой координате возможен по индексу в списке.

Удаление конкретной геометрической фигуры реализуется с использованием метода `delete()`. Если же требуется очистить все пространство канвы, то включается ее метод `ALL`.

### *Постановка задачи*

В соответствии с вариантами разработать проект, реализующий динамическое изображение графическими средствами канвы, отладить его, продемонстрировать его работоспособность преподавателю и оформить отчет по заданию.

### *Методика (алгоритм) выполнения задания*

- I. Из раздела «Варианты задания» в соответствии с вариантом (номером в списке группы: первая цифра-номер группы, далее идет номер студента в группе) выбрать задание на разработку программного проекта:
- II. В «Теоретическом разделе» изучить необходимые средства модуля `tkinter` для выполнения определенного вариантом задания.
- III. Разработать графический интерфейс средствами языка Python для выполнения определенного вариантом задания.
- IV. Разработать программные модули на языке Python для реализации требуемого в задании функционала.
- V. Привязать разработанный функционал к программному интерфейсу.
- VI. Произвести отладку полученного программного приложения.
- VII. Предоставить преподавателю возможность проверки корректности выполнения задания в дистанционном или очном варианте.

VIII. Оформить отчет по выполненному заданию, содержащий:

- 1) задание;
- 2) структуру проекта;
- 3) блок-схему начальной установки приложения, блок-схемы процедур.
- 4) листинг программы.

*Варианты задания*

- 1-1. Красное солнце поднимается из-за моря и перемещается вверх и вправо. Запуск движения – двойной щелчок мыши. Солнце останавливается само при достижении правого верхнего угла, а затем возвращается в исходное состояние.
- 1-2. Секундная стрелка часов совершает движение. Пуск/останов движения – кнопка SpeedButton.
- 1-3. Два пузырька в стакане всплывают за 55 сек один за другим. Запуск/останов движения – команды главного меню.
- 1-4. Солнце с 10 лучами вращается против часовой стрелки с дискретом  $10^0$ /сек с шагом в 1 градус. Запуск/останов движения – команды контекстного меню.
- 1-5. Один шар из связки поднимется вверх, а второй - с удвоенной скоростью опускается вниз. Запуск/останов движения – щелчки мыши.
- 1-6. Крюк крана двигается по стреле вправо и опускается вниз с удвоенной скоростью. Запуск движения – клавиша «Пуск», остановка – щелчок мыши.
- 1-7. Колесо с шестью разноцветными спицами вращается с дискретом 0.5 сек, поворачиваясь на каждом шаге на  $10^0$ . Запуск/останов – команды главного меню.
- 1-8. Знак дорожного движения «Кирпич» постепенно (за 20 сек с дискретом 0.4сек на каждом шаге) прячется за горизонт, а затем возвращается в исходное состояние. Пуск – двойной щелчок мыши.
- 1-9. С дискретом 1.7 сек. меняется цвет светофора. Данная процедура повторяется три раза. Пуск движения – контекстное меню.

- 1-10. Один шар на бильярдном столе движется вверх, а второй с удвоенной скоростью вправо. Движение заканчивается при достижении края стола, а затем шары возвращаются в исходное состояние. Запуск движения – двойной щелчок мыши.
- 1-11. Показания термометра изменяются за 5 сек от  $-10^0$  до  $40^0$  с шагом 1 градус, а затем в два раза медленней до  $0^0$ . Запуск движения – щелчок мыши.
- 1-12. Окна дома разного цвета движутся навстречу друг другу с шагом в 1 пиксель и в конечном итоге меняются местами за 20 сек. Запуск движения – нажатие клавиши «Пуск».
- 1-13. Вертикальная лестница с 20 перекладинами перемещается по экрану справа налево, а шарик прыгает по перекладинам в это время снизу – вверх. Запуск перемещения – команда главного меню. Остановка – двойной щелчок мыши по форме.
- 1-14. К парашюту подвешено слово «КГТУ». Все изображение опускается вниз и сносится влево с удвоенной скоростью. Запуск движения – двойной щелчок мыши. Движение останавливается при достижении левой границы окна.
- 1-15. Диск телефона поворачивается на пять цифр с дискретом 2 сек. Запуск движения – щелчок мыши.
- 1-16. На верху арки фонарь. Арка перемещается по экрану справа налево, при этом меняется цвет фонаря. Запуск движения – команда главного меню, а остановка – двойной щелчок мыши.
- 1-17. Стилизованное изображение шашки проходит туда и обратно по главной диагонали доски с дискретом 0.7 сек; Запуск движения – команда контекстного меню.
- 1-18. Горизонтальный спидометр до 200 км/час. Максимальная скорость достигается за 20 с., а затем падает до 80 км/час за 15 с. Шаг изменения индикатора - 1 пиксель Запуск – нажатием любой клавиши клавиатуры.
- 1-19. Стилизованное изображение перекрестка, через который наперерез друг другу двигаются два стилизованных автомобиля, причем скорость одного в два раза выше, чем у другого. Запуск и остановка движения – посредством кнопок.

- 1-20. По экрану движется справа налево стилизованная стрелка  $\leq$  высотой в 50 пикселей, причем скорость начала в два раза выше скорости хвоста. Запуск и остановка движения – через клавиши клавиатуры.
- 1-21. На экране два прямоугольника, центры которых совпадают. Размеры первого исходно – высота 100, ширина 200 пикселей. Он зеленого цвета. Второй – желтого цвета – исходно представляет вертикальную линию высотой 200 пикселей. Первый прямоугольник сжимается со скоростью 10 пикселей/сек и шагом 1 пиксель до горизонтальной линии. Второй синхронно расширяется. Затем начинается обратный процесс. Прямоугольник – пересечение двух первых – остается все время белого цвета. Запуск и остановка движения – через клавиши клавиатуры.
- 1-22. По экрану движется справа налево со скоростью 5 пикселей/сек. и шагом 1 пиксель столбик шириной 40 пикселей и начальной высотой 10 пикселей. На каждом шаге верхняя грань столбика поднимается на 1 пиксель, а нижняя поднимется в два раза медленнее. При этом меняется случайно цвет столбика. Запуск – команда главного меню, остановка – щелчок мыши.
- 1-23. На форме нарисован квадрат со стороной 300 пикселей. Внутри квадрата по периметру движутся друг за другом против часовой стрелки со скоростью 40 пикселей/сек. и шагом в 1 пиксель три квадратика (красный, зеленый и желтый) со стороной 20 пикселей. Управление движением – команды контекстного меню.
- 1-24. На форме нарисован квадрат со стороной 400 пикселей. Внутри квадрата по периметру движется по часовой стрелке со скоростью 50 пикселей/сек. и шагом в 1 пиксель окружность синего цвета с диаметром 50 пикселей. Управление движением – команды главного меню.
- 1-25. Трехцветный мяч начальным диаметром 100 пикселей поднимается вверх со скоростью 5 пикселей/сек. и уменьшается на каждом шаге в диаметре на 1 пиксель. Запуск движения – щелчок мыши. Остановка – при уменьшении диаметра до 4 пикселей.
- 1-26. На небе серп луны и пять звезд. Луна за 2 мин. и шагом в 1 пиксель по дуге уходит за горизонт

- 1-27. На форме после щелчка мышью в случайном месте появляются квадрат со стороной 150 пикселей и окружность такого же диаметра. Программа определяет, пересекаются ли они, и выводит результат в заголовке формы. Очистка экрана – двойной щелчок мышью.
- 1-28. На канве фигура  высотой 280 пикселей. При запуске она начинается вытягиваться в вертикальную линию со скоростью 40 пикселей/сек, а затем с такой же скоростью возвращается в исходное состояние. Запуск и останов программы – контекстное и главное меню формы.
- 1-29. Исходно на канве фигура в виде квадрата со стороной 300 пикселей. При запуске горизонтальные стороны начинают изменять форму со скоростью 40 пикселей/сек, что приводит в итоге фигуру к виду . Далее фигура возвращается с такой же скоростью в исходное состояние и процесс повторяется для вертикальных сторон, но со скоростью 50 пикселей/сек. Запуск и останов программы – главное меню и двойной щелчок мыши.
- 1-30. На форме в случайных местах появляются три квадрата со сторонами в диапазоне 200÷300 пикселей и номерами в центре. Программа анализирует, какие квадраты пересекаются и выводит результат внизу формы. Управление генерацией/стиранием изображения – команды главного меню.
- 1-31. Исходно на экране находятся на одной горизонтали два квадрата. У первого сторона 300 пикселей, у второго - 2. По команде первый квадрат начинает уменьшаться в размерах со скоростью 30 пикселей/сек и шагом 1 пиксель, а второй синхронно увеличиваться. Когда они поменяются размерами, начинается обратное движение. Запуск и остановка движения – через клавиши клавиатуры.
- 1-32. На канве фигура  высотой 280 пикселей. При запуске она начинается вытягиваться в вертикальную линию со скоростью 40 пикселей/сек, а затем с такой же скоростью возвращается в исходное состояние. Запуск и останов программы – контекстное и главное меню формы.
- 1-33. Исходно на канве фигура в виде квадрата со стороной 300 пикселей. При запуске горизонтальные стороны начинают изменять форму со скоростью 40 пикселей/сек,

что приводит в итоге фигуру к виду . Далее фигура возвращается с такой же скоростью в исходное состояние и процесс повторяется для вертикальных сторон, но со скоростью 50 пикселей/сек. Запуск и останов программы – главное меню и двойной щелчок мыши.

1-34. Красное солнце поднимается из-за моря и перемещается вверх и вправо. Запуск движения – двойной щелчок мыши. Солнце останавливается само при достижении правого верхнего угла, а затем возвращается в исходное состояние.

1-35. Секундная стрелка часов совершает движение. Пуск/останов движения – кнопка SpeedButton.

2-1. На форме нарисован квадрат со стороной 400 пикселей. Два квадрата со стороной 20 пикселей и разного цвета первоначально находятся в левом и правом нижних углах. По нажатию кнопки «Старт» они начинают двигаться по диагоналям до полного перекрытия со скоростью 30 пикселей/сек. и шагом в 1 пиксель, после чего меняют направление движения: вверх и вниз по вертикали. Остановка – достижение границ квадрата.

2-2. Н – образная фигура высотой 60 пикселей и шириной 30 пикселей совершает 4 оборота против часовой стрелки со скоростью  $30^0$ /сек. и шагом в 1 градус относительно левой нижней точки. Запуск/остановка движения – кнопки на форме.

2-3.  $\emptyset$  - образная фигура перемещается сверху вниз на 400 пикселей со скоростью 25 пикселей/сек. и шагом в 1 пиксель, а прямая в это время вращается по часовой стрелке со скоростью  $20^0$ /сек. и шагом в 1 градус относительно собственного центра. Запуск/остановка движения – двойной щелчок мыши.

2-4. На форме нарисована стилизованная ёлочка высотой 300 пикселей, на которой с разной частотой зажигается семь разноцветных лампочек. Запуск движения – кнопка «Пуск», остановка – кнопка «Стоп».

2-5. На пространстве формы изображена шахматная доска. Шашка красного цвета движется по периметру доски по часовой стрелке с дискретом 0.8 сек. Запуск движения –

команда главного меню, остановка – кнопка «Пуск», что приводит к установке шашки на левую верхнюю клетку.

2-6. На пространстве формы изображена шахматная доска. Шашка зеленого цвета движется по главной диагонали доски взад-вперед с дискретом 0.5 сек. Запуск движения – команда контекстного меню, остановка – кнопка «Стоп», что приводит к установке шашки на правую нижнюю клетку.

2-7.  $\nabla$  - образная фигура вращается против часовой стрелки относительно нижнего угла со скоростью  $20^0$ /сек. и шагом в 1 пиксель. Запуск и остановка движения – радионабор на два значения.

2-8. На пространстве формы изображен квадрат размером 250x250. По часовой стрелке по периметру движется окружность диаметром 40 пикселей. По верхней стороне движение идет со скоростью 200/сек. Далее на каждой стороне скорость уменьшается в два раза. Шаг движения - 1 пиксель. Запуск движения – команда главного меню, остановка – кнопка «Пуск», что приводит к установке окружности в левый верхний угол квадрата.

2-9. В  $\diamond$  - образной фигуре внешний квадрат вращается по часовой стрелке со скоростью  $15^0$ /сек и шагом в 1 градус, а внутренний - против часовой стрелки со скоростью в два раза больше. Запуск/остановка движения – виджет **Checkbox**.

2-10. Z – образная фигура за 30 сек. распрямляется и превращается в горизонтальную линию, а затем возвращается за такое же время в исходное состояние. Запуск/остановка движения – виджет **Speedbutton**.

2-11. N – образная фигура за 30 сек. распрямляется и превращается в вертикальную линию, а затем возвращается за такое же время в исходное состояние. Запуск/остановка движения – команды главного меню.

2-12. Исходно на форме находятся друг на друге две окружности диаметром 150 пикселей: верхняя зеленого, нижняя желтого цвета. Щелчком мыши по форме запускается движение: верхняя уменьшается в диаметре со скоростью 10 пикселей/сек и шагом в 1 пиксель, а нижняя – увеличивается в два раза медленней. При уменьшении верхней

окружности до нуля начинается обратный процесс до полного совпадения окружностей. Остановка движения – двойной щелчок мыши.

2-13. Исходно на форме находится квадрат со стороной 400 пикселей. Контекстным меню запускается движение: вертикальные стороны фигуры начинают сближаться со скоростью 40 пикселей/сек и шагом в 1 пиксель. После смыкания сторон начинается обратное движение. При возврате в исходное состояние аналогичное движение повторяется для горизонтальных сторон. Остановка движения – щелчок мышью внутри фигуры.

2-14. Исходно на форме находится квадрат со стороной 500 пикселей. Главным меню запускается движение: левый верхний и правый нижний углы начинают сближаться со скоростью 40 пикселей/сек. и шагом в 1 пиксель до превращения фигуры в диагональ. После смыкания углов начинается обратное движение. При возврате в исходное состояние аналогичное движение повторяется для правого верхнего и левого нижнего углов. Остановка движения – двойной щелчок мышью внутри фигуры.

2-15. Исходно на форме находится равнобедренный треугольник с основанием внизу 400 пикселей. При запуске движения основание начинает уменьшаться со скоростью 50 пикселей/сек. и шагом в 1 пиксель. После превращения фигуры в вертикальную линию начинается обратное движение, но основание теперь находится вверху. При следующем возврате к вертикальной линии снова основание внизу. Запуск и остановка движения – клавиша **Speedbutton**.

2-16. Исходно на форме находится равнобедренный треугольник с основанием слева 500 пикселей. При запуске движения основание начинает уменьшаться со скоростью 50 пикселей/сек. и шагом в 1 пиксель. После превращения фигуры в горизонтальную линию начинается обратное движение, но основание теперь находится справа. При следующем возврате к вертикальной линии снова основание внизу. Запуск и остановка движения – кнопки **Button**.

2-17. Исходно на форме находится окружность, в которую вписан квадрат со стороной 400 пикселей. При запуске движения окружность начинает уменьшаться, пока не окажется вписанной в квадрат. Время уменьшения – 10 сек. и шаг в 1 пиксель. Далее

идет возврат к исходному состоянию, после чего начинает увеличиваться за то же время квадрат, пока окружность не окажется вписанной. Далее снова идет возврат к исходному состоянию. Запуск и остановка движения – кнопки **Button**.

2-18. Исходно на форме находится окружность диаметром 400 пикселей. При запуске движения она начинает сжиматься по горизонтали до вертикальной линии за 15 сек и шагом в 1 пиксель, затем за такое же время возвращается в исходное состояние и начинает сжиматься по вертикали. При смене направления сжатия меняется заливка окружности. Запуск и остановка движения – кнопка **Speedbutton**.

2-19. На экране первоначально находятся две окружности диаметром 400 пикселей, причем первая в виде вертикальной линии, а вторая – в исходном виде. При запуске движения первая окружность за 20 сек приобретает исходный вид, а вторая - превращается в горизонтальную линию. После этого начинаются обратные преобразования. При каждой смене меняется цвет закрашки окружностей. Запуск и остановка движения – кнопки **Button**.

2-20. На экране лесенка из 10 ступенек с дискретом 50 пикселей. На нижней ступеньке находится шарик диаметром 40 пикселей. При запуске движения шарик прыгает по ступенькам вверх с задержкой 1 сек, а затем вниз. Запуск и остановка движения – главное меню.

2-21. При запуске программы на экране в произвольном месте появляется окружность с исходным диаметром 70 пикселей и временем отображения 1 сек. Если пользователь успеет щелкнуть по окружности мышью, то диаметр ее окружности уменьшается на 1 пиксель, а время отображения – на 10 мсек. При уменьшении диаметра до 40 пикселей дальнейшие изменения останавливаются. Текущие значения диаметра и задержки отображаются в заголовке формы. Запуск и останов программы – контекстное меню формы.

2-22. По команде контекстного меню формы на экране в случайных местах появляются три окружности радиусом 100 пикселей разного цвета и с номера в центре. Программа выводит на метке результат анализа – какие окружности пересекаются. Очистка экрана – тоже контекстное меню.

- 2-23. На ночном небе семь звезд, которые мигают с разной периодичностью. Одна звезда медленно движется. Запуск и останов программы – контекстное меню формы.
- 2-24. Исходно на канве фигура в виде квадрата со стороной 200 пикселей. При запуске горизонтальные стороны начинают сжиматься со скоростью 40 пикселей/сек, что приводит фигуру к виду . Далее фигура возвращается с такой же скоростью в исходное состояние и процесс повторяется для вертикальных сторон, но со скоростью 30 пикселей/сек. Запуск и останов программы – главное меню и двойной щелчок мыши.
- 2-25. На канве фигура  высотой 250 пикселей. При запуске она начинается складываться в горизонтальную линию со скоростью 50 пикселей/сек, а затем с такой же скоростью возвращается в исходной состоянии. Запуск и останов программы – контекстное меню формы.
- 2-26. На канве фигура  высотой 300 пикселей. В левой верхней точке фигуры находится центр окружности диаметром 40 пикселей. При запуске окружность начинает двигаться так, чтобы ее центр находился все время на фигуре. Скорость движения – 60 пикселей/сек. При достижении нижней конечной точки окружность возвращается в начало и продолжает движение. Запуск и останов программы – главное меню формы.
- 2-27. Два пузырька разного диаметра в стакане высотой 200 пикселей всплывают за 55 сек. один за другим. Запуск/останов движения – команды верхнего меню.
- 2-28. Солнце с 10 лучами вращается против часовой стрелки с дискретом  $10^0$ /сек с шагом в 1 градус. Запуск/останов движения – команды контекстного меню.
- 2-29. Один шар из связки поднимется вверх, а второй - с удвоенной скоростью опускается вниз. Запуск/останов движения – щелчки мыши.
- 2-30. Крюк крана двигается по стреле вправо и опускается вниз с удвоенной скоростью. Запуск движения – клавиша «Пуск», остановка – щелчок мыши.
- 2-31. Колесо с шестью разноцветными спицами вращается с дискретом 0.5 сек, поворачиваясь на каждом шаге на  $10^0$ . Запуск/останов – команды главного меню.

- 2-32. Знак дорожного движения «Кирпич» постепенно (за 20 сек с дискретом 0.4 сек на каждом шаге) прячется за горизонт, а затем возвращается в исходное состояние. Пуск – двойной щелчок мыши.
- 2-33. С дискретом 1.7 сек меняется цвет светофора. Данная процедура повторяется три раза. Пуск движения – контекстное меню.
- 2-34. Один шар на бильярдном столе движется вверх, а второй с удвоенной скоростью вправо. Движение заканчивается при достижении края стола, а затем шары возвращаются в исходное состояние. Запуск движения – двойной щелчок мыши.
- 2-35. Два пузырька разного диаметра и цвета в стакане высотой 200 пикселей всплывают за 55 сек один за другим. Запуск/останов движения – команды верхнего меню.

## Критерии и нормы оценки контрольной работы

В соответствии с учебным планом предусмотрены две оценки результатов выполнения контрольной работы: «зачет» и «незачет». Оценка «зачет» предусматривает для каждого отдельного задания:

1. Создание и отладку работающего программного приложения, полностью реализующего задание, т. е.:
    - а) включающего графический интерфейс, в состав которого входят именно те виджеты, которые указаны в задании;
    - б) реализующего именно тот функционал, который предусмотрен заданием;
    - в) предусматривающего именно ту модель событийного управления, которая предусмотрена заданием.
  2. Оформление отчета по каждому заданию в составе контрольной работы, включающего:
    - а) структуру графического интерфейса с указанием задействованных виджетов;
    - б) оформленные в соответствии с ГОСТом блок-схемы функций в составе программного приложения;
    - в) листинг программного приложения с комментариями
- Весь текстовый материал реализуется шрифтом с размером не менее 12 пунктов.

При отсутствии хотя бы одной позиции из указанного выше перечня контрольная работа оценивается как незачтенная и возвращается студенту для доработки.

## СПИСОК РЕКОМЕНДУЕМЫХ ИСТОЧНИКОВ:

1. Хахаев, И. А. Практикум по алгоритмизации и программированию на Python: курс : учеб. пособие : [16+] / И. А. Хахаев. – 2-е изд., исправ. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 179 с. : ил. – Режим доступа: по подписке. – URL: <https://biblioclub.ru/index.php?page=book&id=429256> (дата обращения: 30.03.2022). – Библиогр. в кн. – Текст : электронный.
2. Буйначев, С. К. Основы программирования на языке Python: учеб. пособие / С. К. Буйначев, Н. Ю. Боклаг ; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург : Издательство Уральского университета, 2014. – 92 с. : табл., ил. – Режим доступа: по подписке. – URL <https://biblioclub.ru/index.php?page=book&id=275962> (дата обращения: 30.03.2022). – Библиогр. в кн. – ISBN 978-5-7996-1198-9. – Текст : электронный.
3. Сузи, Р. А. Язык программирования Python: учебное пособие : [16+] / Р. А. Сузи. – 2-е изд., испр. – Москва : Интернет-Университет Информационных Технологий (ИНТУИТ) : Бинوم. Лаборатория знаний, 2007. – 327 с. – (Основы информационных технологий). – Режим доступа: по подписке. – URL:<https://biblioclub.ru/index.php?page=book&id=233288> (дата обращения: 30.03.2022). – ISBN 978-5-9556-0109-0. – Текст : электронный.
4. Жуков Р.А. Язык программирования Python: практикум: учеб, пособие / Р.А. Жуков. — Москва: ИНФРА-М, 2019. — 216с. + Доп. материалы [Электронный ресурс]; Режим доступа: [https://codernet.ru/books/python/yazyk\\_programmirovaniya\\_python\\_praktikum/](https://codernet.ru/books/python/yazyk_programmirovaniya_python_praktikum/) (дата обращения: 27.03.2022).
5. Саммерфилд М. Программирование на Python 3. Подробное руководство. – Пер. с англ. – СПб.: Символ-Плюс, 2009. – 608 с., ил. - Электрон. дан. – Режим доступа: [http://uchcom7.botik.ru/L/prog/python/python\\_08.pdf](http://uchcom7.botik.ru/L/prog/python/python_08.pdf). (дата обращения: 27.03.2022).
6. Лекция 14 Разработка приложений с графическим интерфейсом пользователя Библиотека Tkinter – Текст: [Электронный ресурс]; Режим доступа: <http://python.inr.ru/s1914.pdf> . (дата обращения: 24.04.2022).
7. Коварцев, А. Н. Методы и технологии визуального программирования: учеб. пособие / А. Н. Коварцев, В. В. Жидченко, Д. А. Попова-Коварцева. – Самара: ООО «Офорт», 2017. - 197 с.: 83 ил материалы [Электронный ресурс; Режим доступа: [http://repo.ssau.ru/bitstream/Uchebnye-posobiya/Metody-i-tehnologii-vizualnogo-programmirovaniya-66439/3/Учебник%20визуальное%20программирование%202018\\_итог.pdf](http://repo.ssau.ru/bitstream/Uchebnye-posobiya/Metody-i-tehnologii-vizualnogo-programmirovaniya-66439/3/Учебник%20визуальное%20программирование%202018_итог.pdf) (дата обращения: 23.04.2022).
8. Карташов, Д. В. К 27 Объектно-ориентированное программирование: учебно-методическое пособие / Д. В. Карташов, Л. А. Непомнящая. – Томск : Изд-во ТГПУ, 2019. – 52 с. материалы [Электронный ресурс]; Режим доступа: <http://fulltext.tspu.edu.ru/OA/m2019-29.pdf> (дата обращения: 23.04.2022).
9. Хахаев, И. А. Практикум по алгоритмизации и программированию на Python: / И. А. Хахаев — Москва : Альт Линукс, 2010. — 126 с. : ил. — (Библиотека ALT Lin17 с. их). - Электрон. дан. – Режим доступа: <https://all-journals.com/books/program->

[ming/13066-praktikum-po-algoritmizacii-i-programmirovaniyu-na.html](http://ming/13066-praktikum-po-algoritmizacii-i-programmirovaniyu-na.html) (дата обращения: 29.03.2022).

10. Учись tkinter. – Бесплатная электронная книга, 38 с. - Электрон. дан. – Режим доступа: <https://riptutorial.com/Download/tkinter-ru.pdf> (дата обращения: 27.03.2022).
11. Python. Лекция 12. Создание приложений с GUI, 17 с. - Электрон. дан. – Режим доступа: <https://ideafix.name/wp-content/uploads/2012/08/Python-12.pdf> (дата обращения: 27.03.2022)
12. 3 вида меню в Tkinter: создание, наполнение верхнего и выпадающего меню [Электронный ресурс]; Режим доступа: <https://pythonru.com/uroki/sozdanie-menju-tkinter-10> (дата обращения: 25.04.2022).
13. Pillow обработка изображений в Python на примерах [Электронный ресурс]; Режим доступа: <https://python-scripts.com/pillow> (дата обращения: 28.04.2022).
14. Работа с датой и временем в Python [Электронный ресурс]; Режим доступа: <https://python-scripts.com/datetime-time-python> (дата обращения: 28.04.2022).

*Титульный лист контрольной работы*

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО РЫБОЛОВСТВУ**

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

**«Калининградский государственный технический университет»**

Институт цифровых технологий

Кафедра \_\_\_\_\_

наименование кафедры

Контрольная работа

допущена к защите

Руководитель: \_\_\_\_\_

(уч. степень, звание, долж-  
ность)

И.О. Фамилия

«\_\_» \_\_\_\_\_ 202\_\_ г.

Контрольная работа

защищена

Руководитель: \_\_\_\_\_

(уч. степень, звание, должность)

И.О. Фамилия

«\_\_» \_\_\_\_\_ 202\_\_ г.

**Контрольная работа № \_\_\_\_\_**

(указывается, если по дисциплине более 1 работы)

по дисциплине

«Наименование дисциплины»

Шифр студента \_\_\_\_\_

Вариант № \_\_\_\_\_

Работу выполнил:

студент гр. \_\_\_\_\_

И.О. Фамилия

«\_\_» \_\_\_\_\_ 202\_\_ г.

Калининград

202\_\_

Локальный электронный методический материал

Леонид Григорьевич Высоцкий

**ВЫСОКОУРОВНЕВЫЕ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ (ВТП)**

Редактор Г.А. Смирнова

Уч.-изд. л. 5,0. Печ. л. 6,25.

Издательство федерального государственного бюджетного образовательного  
учреждения высшего образования  
«Калининградский государственный технический университет».  
236022, Калининград, Советский проспект, 1