МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе №2

Обработка событий и рисование в Tkinter

По дисциплине «Объектно-ориентированное программирование»

Выполнил студент группы ИВТ-б-о-20-1

Галяс Д. И. « »_____20__г.

Подпись студента_____

Работа защищена « »_____20__г.

Проверил Воронкин Р. А. _____

(подпись)

Ставрополь 2022

Цель работы: приобретение навыков улучшения графического интерфейса пользователя GUI с помощью обработки событий и рисования, реализованных в пакете Tkinter языка программирования Python версии 3.x.

Ссылка на репозиторий: https://github.com/DIMITRY-GALYAS1/4.7-PySide2-.git

Ход работы:

1. Создал новый репозиторий на github, после клонировал его и создал в папке репозитория новый проект PyCharm.

2. Выполнил первое задание.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

"""
Напишите программу, состоящую из двух списков Listbox. В первом будет,
например, перечень товаров, заданный программно. Второй изначально пуст, пусть это
будет перечень покупок. При клике на одну кнопку товар должен переходить из одного
списка в другой. При клике на вторую кнопку - возвращаться (человек передумал покупать).
Предусмотрите возможность множественного выбора элементов списка и их перемещения
"""

import sys
from PySide2.QtWidgets import QWidget, QApplication, QListWidget, QPushButton, QAbstractItemView, QVBoxLayout, \
    QHBoxLayout


class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.list1 = QListWidget()
        self.list2 = QListWidget()
        self.list1.setSelectionMode(QAbstractItemView.ExtendedSelection)
        self.list2.setSelectionMode(QAbstractItemView.ExtendedSelection)
        self.list1.addItems(shopping_list)
        self.butn1 = QPushButton("Добавить")
        self.butn2 = QPushButton("Удалить")
```

Рисунок 1. Код первого задания

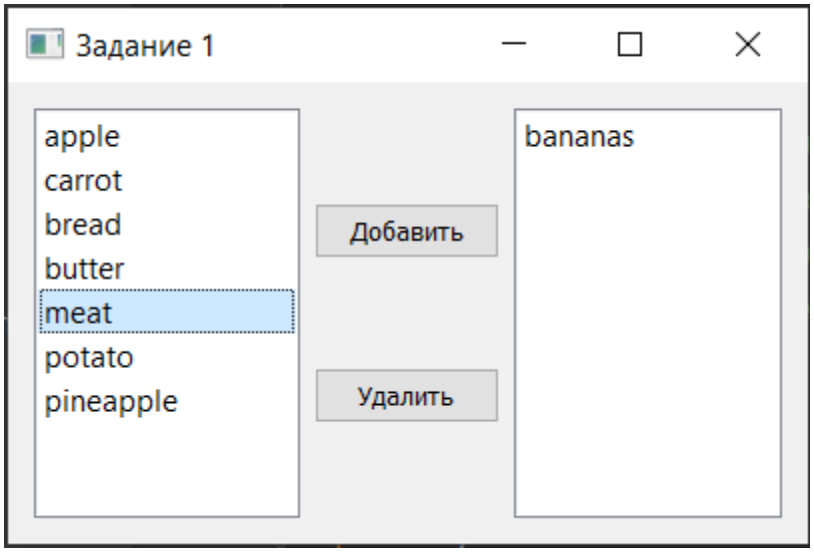

Рисунок 2. Результат работы кода

3. Выполнил второе задание.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

"""
Напишите программу по следующему описанию. Нажатие Enter в однострочном текстовом поле приводит к перемещению текста из
него в список. При двойном клике по элементу-строке списка, она должна копироваться в текстовое поле.
"""

import sys
from PySide2.QtWidgets import QApplication, QWidget, QLineEdit,  QVBoxLayout, QListWidget


class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.list_write = QListWidget()
        self.line_edit = QLineEdit()
        self.line_edit.returnPressed.connect(self.move_txt)
        self.list_write.itemDoubleClicked.connect(self.copy_item)
        self.initialization()

    def initialization(self):
        self.setGeometry(100, 100, 400, 230)
        self.setWindowTitle("Задание 2")
        self.display_widgets()

    def display_widgets(self):
        v_box = QVBoxLayout()
        v_box.addWidget(self.line_edit)
        v_box.addWidget(self.list_write)
```

Рисунок 3. Код второго задания

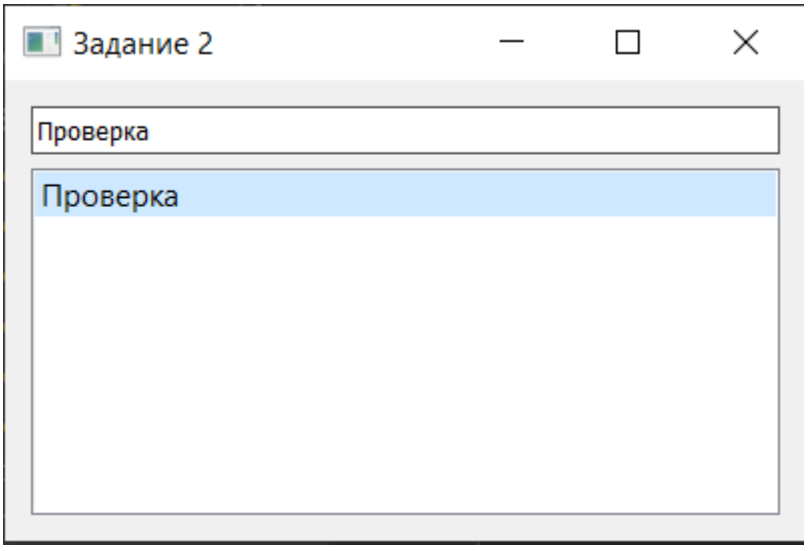

Рисунок 4. Работа кода

4. Сделал третье задание.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

"""
Напишите программу по описанию. Размеры многострочного текстового поля определяются значениями,
введенными в однострочные текстовые поля. Изменение размера происходит при нажатии мышью на кнопку,
а также при нажатии клавиши Enter. Цвет фона экземпляра Text светлосерый (lightgrey),
когда поле не в фокусе, и белый, когда имеет фокус. Для справки: фокус перемещается по виджетам при нажатии Tab,
Ctrl+Tab, Shift+Tab, а также при клике по ним мышью (к кнопкам последнее не относится).
"""


import sys
from PySide2.QtWidgets import QApplication, QWidget, QLineEdit, QHBoxLayout, QVBoxLayout, QPushButton, QTextEdit


class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        QApplication.instance().focusChanged.connect(self.on_focus)
        self.line_edit1 = QLineEdit()
        self.line_edit2 = QLineEdit()
        self.text_box = QTextEdit()
        self.btn1 = QPushButton("Edit")
        self.btn1.clicked.connect(self.edit_size)
        self.line_edit2.returnPressed.connect(self.edit_size)
        self.initialization()
```

Рисунок 5. Код третьего задания

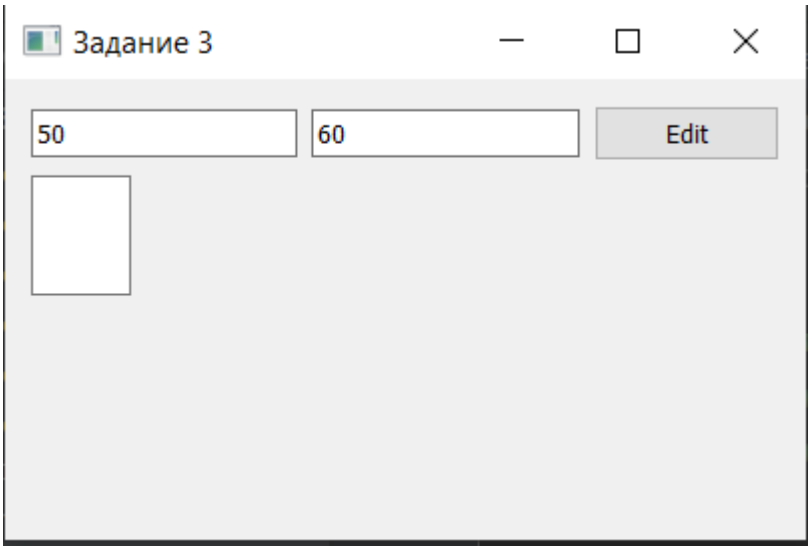

Рисунок 6. Работа кода

5. Выполнил четвертое задание.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

"""
# Создать изображение на холсте
"""

import sys
import random
from PySide2.QtCore import Qt, QPoint
from PySide2.QtGui import QPainter, QBrush, QPen, QPolygon, QColor
from PySide2.QtWidgets import QApplication, QWidget


class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Задание 4")
        self.setGeometry(300, 300, 600, 600)

    def paintEvent(self, event):
        painter = QPainter(self)
        painter.setPen(QPen(Qt.darkRed, 3, Qt.SolidLine))  # Цвет и линия обводки
        painter.setBrush(QColor(255, 0, 0, 127))
        painter.drawRect(215, 200, 253, 281)  # корпус домика
        painter.setPen(QPen(Qt.darkRed, 3, Qt.SolidLine))
        painter.setBrush((QBrush(Qt.white)))
```

Рисунок 7. Код четвертого задания

Рисунок 8. Работа четвертого задания

6. Приступил к выполнению пятого задания.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

"""
Изучите приведенную программу и самостоятельно запрограммируйте постепенное движение фигуры в ту точку холста,
где пользователь кликает левой кнопкой мыши. Координаты события хранятся в его атрибутах x и y (event.x , event.y)
"""

import sys
from PySide2.QtWidgets import QWidget, QApplication
from PySide2.QtCore import QPropertyAnimation, QPoint


class Window(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Задание 5")
        self.resize(500, 500)
        self.child = QWidget(self)
        self.child.setStyleSheet("background-color: green;border-radius: 25%;")
        self.child.resize(50, 50)
        self.animation = QPropertyAnimation(self.child, b"pos")
        self.animation.setDuration(1500)

    def mousePressEvent(self, event):
        self.animation.setEndValue(QPoint(event.x() - 25, event.y() - 25))
        self.animation.start()


if __name__ == "__main__":
```

Рисунок 9. Код пятого задания

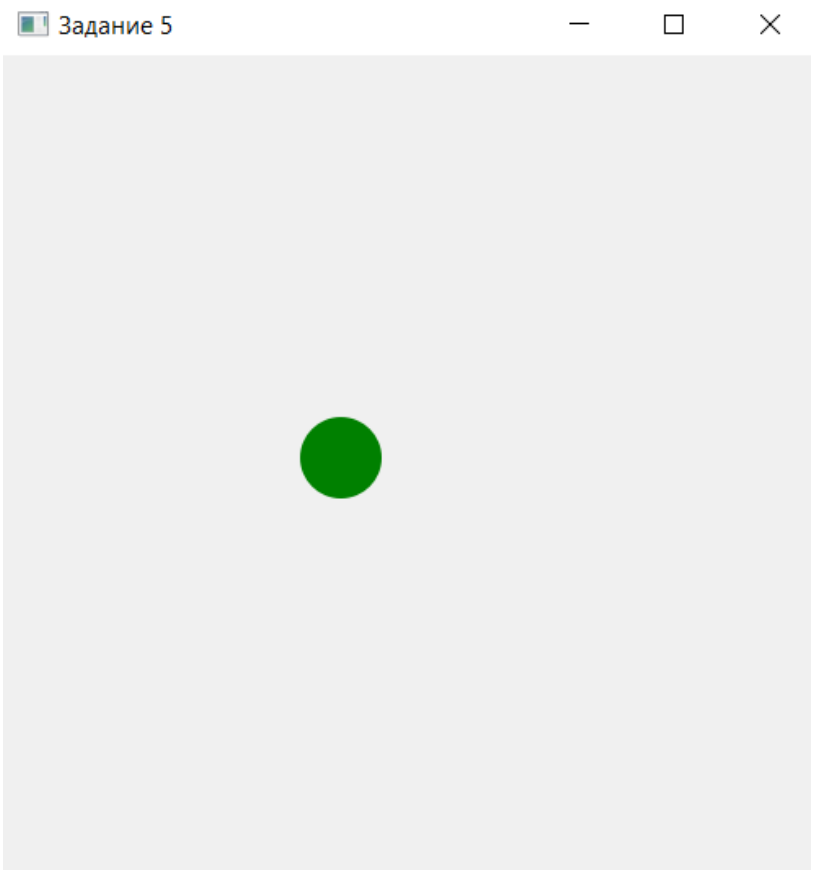

Рисунок 10. Работа пятого задания

Вывод: в ходе выполнения лабораторной работы приобрел навыки улучшения графического интерфейса пользователя GUI с помощью обработки

событий и рисования, реализованных в пакете Tkinter языка программирования Python версии 3.x.