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

Кафедра инфокоммуникаций

Отчет по лабораторной работе №4.8

Обработка событий и рисование в PySide2

По дисциплине «Объектно-ориентированное программирование»

Выполнил студент группы ИВТ-б-о-20-1
Новикова В.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 PySide2 языка программирования Python версии 3.x

Ход работы:

1. Создала общедоступный репозиторий и клонировала на его на локальный сервер.

https://github.com/Valentina1502/4_8_PySide2.git

2. После изучения теоретического материала и методических рекомендаций приступила к выполнению заданий.

Задание 1. 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

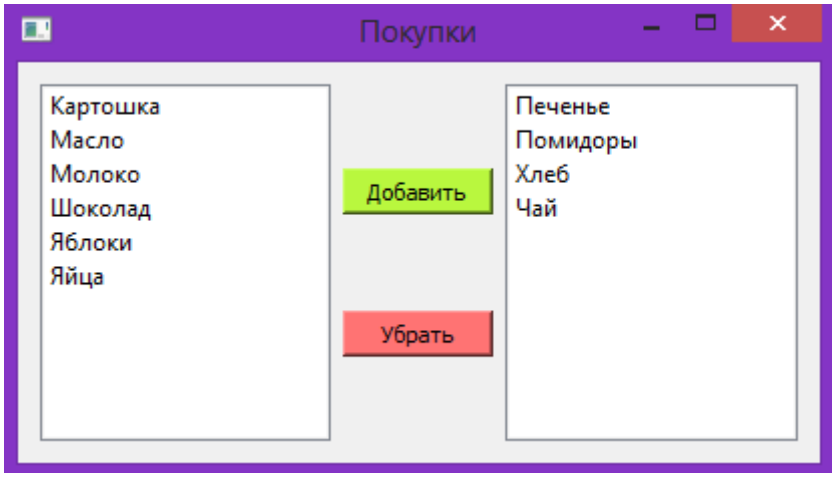

Рисунок 1 – Задание 1

Код:

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


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


class MainWindow(QWidget):
```

```
    def __init__(self):
        super().__init__()  # вызываем конструктор базового класса
        self.list1 = QListWidget()
        self.list1.setSelectionMode(QAbstractItemView.ExtendedSelection)
        self.list1.addItems(products)
        self.button1 = QPushButton("Добавить", self)
        self.button2 = QPushButton("Убрать", self)
        self.list2 = QListWidget()
        self.list2.setSelectionMode(QAbstractItemView.ExtendedSelection)
        self.initialize_ui()



    def initialize_ui(self):
        self.setGeometry(110, 100, 400, 200)
        self.setWindowTitle("Покупки")
        self.button1.setStyleSheet("background: #B9F73E")
        self.button2.setStyleSheet("background: #FF7373")
        self.button1.clicked.connect(self.add_product)
        self.button2.clicked.connect(self.delete_product)

        self.show()


    def add_product(self):
        listItems = self.list1.selectedItems()
        for item in listItems:
            self.list1.takeItem(self.list1.row(item))
            self.list2.addItem(item)

    def delete_product(self):
        listItems = self.list2.selectedItems()
        for item in listItems:
            self.list2.takeItem(self.list2.row(item))
            self.list1.addItem(item)


    def align(self):
        hbox = QHBoxLayout()
        vbox = QVBoxLayout()
        hbox.addWidget(self.list1)
        hbox.addLayout(vbox)
        vbox.addWidget(self.button1)
        vbox.addWidget(self.button2)
        hbox.addWidget(self.list2)
        self.setLayout(hbox)

if __name__ == '__main__':
    products = ["Хлеб", "Молоко", "Чай", "Картошка", "Яйца",\
        "Яблоки", "Масло", "Помидоры", "Шоколад", "Печенье"]
    products.sort()
    app = QApplication(sys.argv)
    window = MainWindow()
    window.align()
```

```
    sys.exit(app.exec_())
```

Задание 2. Напишите программу по следующему описанию. Нажатие Enter в однострочном текстовом поле приводит к перемещению текста из него в список (экземпляр Listbox). При двойном клике по элементу-строке списка, она должна копироваться в текстовое поле

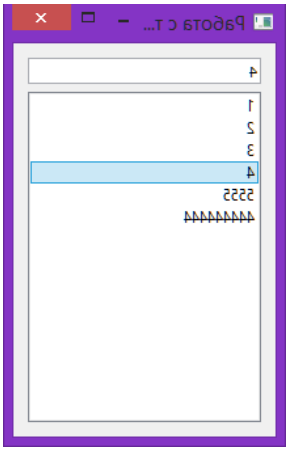

Рисунок 2 – Задание 2

Код:

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


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


class MainWindow(QWidget):
    def __init__(self):
        super().__init__()  # вызываем конструктор базового класса
        self.list =QListWidget()
        self.list.itemDoubleClicked.connect(self.in_line)
        self.line = QLineEdit()
        self.line.returnPressed.connect(self.in_list)
        self.initialize_ui()


    def initialize_ui(self):
        self.setGeometry(110, 100, 200, 300)
        self.setWindowTitle("Работа с текстом")
        v_layout = QVBoxLayout()
        v_layout.addWidget(self.line)
        v_layout.addWidget(self.list)
        self.setLayout(v_layout)
```

```
        self.show()


    def in_list(self):
        self.list.addItem(self.line.text())
        self.line.clear()


    def in_line(self):
        listItems = self.list.selectedItems()
        for item in listItems:
            self.line.setText(item.text())


if __name__ == '__main__':
    app = QApplication(sys.argv)
    window = MainWindow()
    sys.exit(app.exec_())
```

Задание 3. 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.

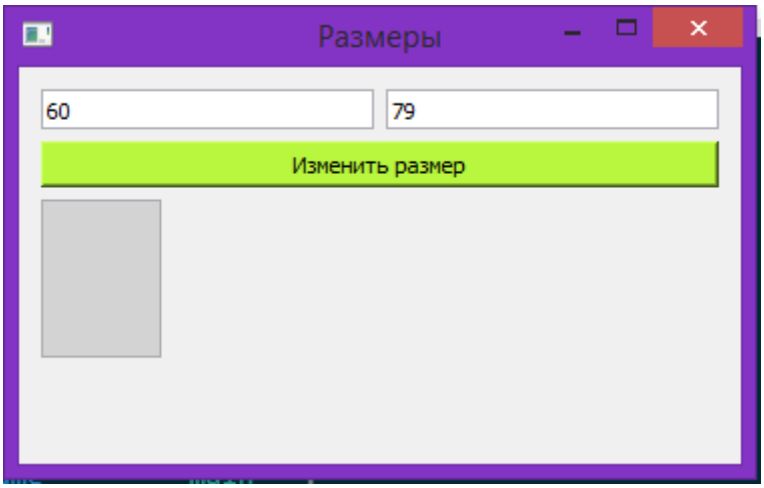

Рисунок 3 – Задание 3

Код:

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


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

```
class MainWindow(QWidget):
    def __init__(self):
        super().__init__()  # вызываем конструктор базового класса
        QApplication.instance().focusChanged.connect(self.on_focus)
        self.line_edit1 = QLineEdit()
        self.line_edit2 = QLineEdit()
        self.line_edit1.returnPressed.connect(self.edit_size)
        self.line_edit2.returnPressed.connect(self.edit_size)
        self.button1 = QPushButton("Изменить размер", self)
        self.textbox = QTextEdit()
        self.initialize_ui()


    def initialize_ui(self):
        self.setGeometry(110, 100, 400, 200)
        self.setWindowTitle("Размеры")
        self.button1.setStyleSheet("background: #B9F73E")
        self.button1.clicked.connect(self.edit_size)
        hlay = QHBoxLayout()
        vlay = QVBoxLayout()
        hlay.addWidget(self.line_edit1)
        hlay.addWidget(self.line_edit2)
        vlay.addLayout(hlay)
        vlay.addWidget(self.button1)
        vlay.addWidget(self.textbox)
        self.setLayout(vlay)
        self.show()


    def edit_size(self):
        self.textbox.resize(
            int(self.line_edit1.text()),
            int(self.line_edit2.text())
        )


    def on_focus(self, old, new):
        if self.textbox == new:
            self.textbox.setStyleSheet(f"background-color: #fff;")
        elif self.textbox == old:
            self.textbox.setStyleSheet(f"background-color: #d3d3d3;")


if __name__ == '__main__':
    app = QApplication(sys.argv)
    window = MainWindow()
    sys.exit(app.exec_())
```

Задание 4. Создайте на холсте изображение

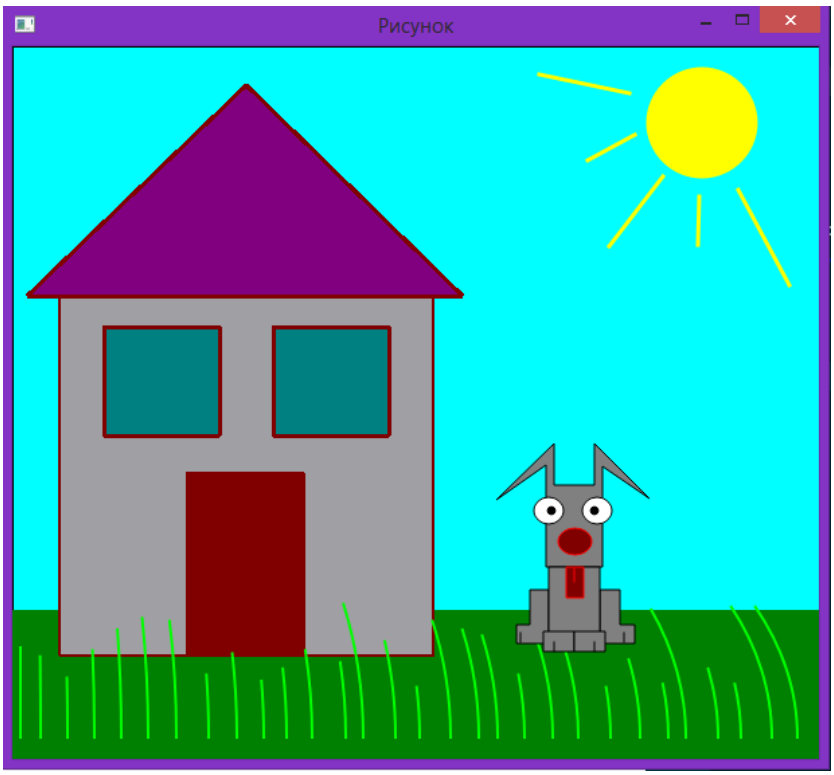

Рисунок 4 – Задание 4

Код:

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

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


class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Рисунок")
        self.setGeometry(150, 50, 600, 550)


    def paintEvent(self, event):
        painter = QPainter(self)
        painter.setBrush(QBrush(Qt.cyan))
        painter.drawRect(0, 0, 600, 500)
        painter.setPen(QPen(Qt.darkGreen))
        painter.setBrush(QBrush(Qt.darkGreen))
        painter.drawRect(0, 420, 600, 130)

        painter.setPen(QPen(Qt.darkRed, 2, Qt.SolidLine))
        painter.setBrush(Qt.gray)
        painter.drawRect(35, 186, 278, 268)
```

```
        painter.setPen(QPen(Qt.darkRed, 3, Qt.SolidLine))
        painter.setBrush((QBrush(Qt.darkCyan)))
        painter.drawRect(68, 209, 86, 81)
        painter.drawRect(194, 209, 86, 81)
        painter.setBrush((QBrush(Qt.darkRed)))
        painter.drawRect(130, 318, 86, 135)
        painter.setBrush((QBrush(Qt.darkMagenta)))
        points = QPolygon([
            QPoint(11, 186),
            QPoint(174, 29),
            QPoint(335,186)
        ])
        painter.drawPolygon(points)
        painter.begin(self)
        painter.setRenderHint(QPainter.Antialiasing)
        painter.setBrush(QBrush(Qt.yellow, Qt.SolidPattern))
        painter.setPen(QPen(Qt.yellow, 3, Qt.SolidLine))
        painter.drawEllipse(473, 17, 80, 80)
        painter.drawLine(459, 35, 392, 21)
        painter.drawLine(484, 97, 444, 149)
        painter.drawLine(540, 107, 578, 178)
        painter.drawLine(463, 66, 428, 85)
        painter.drawLine(511, 112, 510, 148)
        self.drawGrass(painter)
        self.drawCat(painter)


    def drawGrass(self, painter):
        painter.begin(self)
        painter.setRenderHint(QPainter.Antialiasing)
        painter.setPen(QPen(Qt.green, 2, Qt.SolidLine))
        painter.setBrush(Qt.green)
        for i in range(30):
            painter.drawArc(random.randint(1, 8), 300, i * 20, 430, 0 * 200,
random.randint(15, 45) * 10)


    def drawCat(self, painter):
        painter.begin(self)
        points1 = QPolygon([
            QPoint(360, 338),
            QPoint(403, 296),
            QPoint(403, 327),
            QPoint(433, 327),
            QPoint(433, 296),
            QPoint(474, 337),
            QPoint(439, 313),
            QPoint(439, 388),
            QPoint(397, 388),
            QPoint(397, 312)
        ])
        points2 = QPolygon([
            QPoint(399, 405),
            QPoint(385, 405),
```

```
            QPoint(385, 431),
            QPoint(375, 431),
            QPoint(375, 445),
            QPoint(399, 445)
        ])
        points3 = QPolygon([
            QPoint(437, 405),
            QPoint(452, 405),
            QPoint(452, 431),
            QPoint(463, 431),
            QPoint(463, 445),
            QPoint(437, 445)
        ])
        painter.setPen(QPen(Qt.black))
        painter.setBrush(QBrush(Qt.darkGray))
        painter.drawPolygon(points1)
        painter.drawPolygon(points2)
        painter.drawPolygon(points3)
        painter.drawRect(399, 388, 38, 55)
        painter.drawRect(395, 436, 23, 15)
        painter.drawRect(418, 436, 23, 15)
        painter.drawLine(378, 437, 378, 444)
        painter.drawLine(403, 444, 403, 451)
        painter.drawLine(433, 444, 433, 451)
        painter.drawLine(455, 437, 455, 444)
        painter.setPen(QPen(Qt.red))
        painter.setBrush(QBrush(Qt.darkRed))
        painter.drawRect(412, 388, 13, 23)
        painter.drawLine(418, 389, 418, 399)
        painter.drawEllipse(406, 359, 25, 20)
        painter.setPen(QPen(Qt.black))
        painter.setBrush(QBrush(Qt.white))
        painter.drawEllipse(388, 336, 22, 20)
        painter.drawEllipse(424, 336, 22, 20)
        painter.setPen(QPen(Qt.black))
        painter.setBrush(QBrush(Qt.black))
        painter.drawEllipse(398, 343, 6, 6)
        painter.drawEllipse(430, 343, 6, 6)


if __name__ == "__main__":
    app = QApplication(sys.argv)
    window = MainWindow()
    window.show()
    sys.exit(app.exec_())
```

Задание 5. 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)

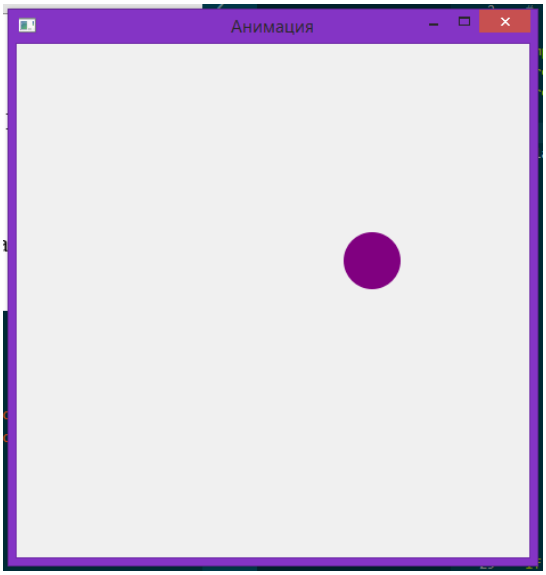

Рисунок 5 – Задание 5

Код:

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

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


class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(450, 450)
        self.setWindowTitle("Анимация")
        self.boll = QWidget(self)
        self.boll.setStyleSheet(
            "background-color: purple;\
            border-radius: 25%;"
            )
        self.boll.resize(50, 50)
        self.anim = QPropertyAnimation(self.boll, b"pos")
        self.anim.setDuration(1500)

    def mousePressEvent(self, event):
        self.anim.setEndValue(QPoint(event.x(), event.y()))
        self.anim.start()


if __name__ == "__main__":
    app = QApplication(sys.argv)
    window = MainWindow()
```

```
    window.show()
    sys.exit(app.exec_())
```

Вывод: в ходе выполнения лабораторной работы были приобретены навыки построения графического интерфейса пользователя GUI с помощью пакета PySide2 языка программирования Python версии 3.x.