# **ФГБОУ ВО «Московский политехнический университет»**

## **Лабораторная работа №7**

GUI, классы, модуль Tkinter

**Задание 2 Вариант №19**

По дисциплине:

Программирование

Выполнил

студент 1 курса

группы 201-321

Сметанина А.А.

Проверил:

\_\_\_\_\_\_\_\_\_\_\_\_ Никишина И.Н.

# **МОСКВА 2020**

**Цель работы**

Получить практический навык в написании программ, использующих графический

интерфейс пользователя. Познакомиться с такими понятиями, как класс, виджеты, массивы записей, а также с атрибутами класса и методами передачи данных между классами.

**Постановка задачи**

Автоматизированная информационная система на железнодорожном вокзале содержит сведения об отправлении поездов дальнего следования. Для каждого поезда указывается:

* номер поезда;
* станция назначения;
* время отправления.

Данные в информационной системе организованы в виде линейного списка. Составить программу, которая:

* обеспечивает первоначальный ввод данных в информационную систему и формирование линейного списка;
* производит вывод всего списка;
* получает номер поезда и выводит данные об этом поезде;
* получает название станции назначения и выводит данные обо всех поездах, следующих до этой станции.

**Теоретическая часть**

Для решения задачи используется графический модуль PyQt5.

**Структура**

**Класс Train**

Описывает поезд, хранит его номер, время отправления и станцию назначения.

**Класс Trains**

Описывает список поездов. Выполняет поиск поезда по номеру, поиск поездов по станции назначения.

**Класс MainWindow**

Обеспечивает работу главного окна. Представлен вывод поездов по

мере их ввода, кнопки «Добавить» и «Найти», вызывающие соответствующие окна, и кнопка «Выйти», завершающая выполнение программы.

**Класс InputWindow**

Обеспечивает ввод поездов, контролирует правильность введённых данных: не допускается ввод пустых полей, ввод нецифровых символов в строке ввода номера поезда, ввод некорректного времени, то есть есть контроль за соблюдением формата «часы:минуты», часы в полуинтервале [0; 24), минуты – [0; 60). Пополняет список поездов.

**Класс SearchWindow**

Выполняет поиск поезда по номеру и поиск поездов по станции назначения. Принимает на вход номер поезда (число) либо название станции. Поиск происходит в соответствии с нажатой кнопкой.

**Класс ErrorWindow**

Обеспечивает вывод сообщений об ошибках в отдельном окне.

Сообщение выводится, например, если нажата кнопка «Найти» в главном окне, когда список поездов пуст.

**Описание программы**

Программа написана на алгоритмическом языке Python 3.6, реализована в среде macOS

Catalina и состоит из частей, отвечающих за ввод данных, вычисление и представление данных на экране монитора.

**Описание алгоритма**

1. Получить на вход информацию о поездах.
2. Отобразить их в главном окне.
3. Ожидать от пользователя нажатие какой-либо кнопки, то есть ожидать некоторое событие, которое будет передано соответствующему обработчику.

**Описание входных и выходных данных**

Программа принимает на вход числа и строки, выводит GUI.

**Листинг программы**

import sys

import PyQt5.QtWidgets as W

from PyQt5 import uic, QtCore

from datetime import datetime as dt

class Train:

def \_\_init\_\_(self, a, b, c):

self.number = a

self.dest = b

self.time = c

def \_\_str\_\_(self):

time = self.time.strftime("%H:%M")

num = self.number

return f"Поезд {num}\nОтправляется в {time}\nИдёт до {self.dest}"

class Trains:

def \_\_init\_\_(self, a):

self.trains = a

def search\_by\_num(self, num):

for train in self.trains:

if train.number == num:

return train

def search\_by\_dest(self, dest):

res = []

dest = dest.strip()

for train in self.trains:

if train.dest.strip() == dest:

res.append(train)

if res:

return Trains(res)

def append(self, train):

if train not in self:

self.trains.append(train)

def \_\_contains\_\_(self, a):

return any(i == a for i in self.trains)

def \_\_str\_\_(self):

if len(self.trains) == 0:

return ''

return '\n\n'.join(str(t) for t in self.trains)

trains = Trains([])

class MainWindow(W.QMainWindow):

def \_\_init\_\_(self, \*args):

super().\_\_init\_\_()

uic.loadUi('./windows/MainWindow.ui', self)

self.InputWindow = InputWindow(self, [])

self.ErrorWindow = ErrorWindow(self, [])

self.SearchWindow = SearchWindow(self, [])

self.setWindowTitle("Поезда")

self.SearchButton.clicked.connect(self.find)

self.AddButton.clicked.connect(self.input)

self.ExitButton.clicked.connect(self.close)

self.timer = QtCore.QTimer(self)

self.timer.setInterval(500)

self.timer.start()

self.timer.timeout.connect(self.show)

def show(self):

self.TrainsBrowser.setText(str(trains))

super().show()

def input(self):

self.InputWindow.close()

self.InputWindow.show()

def find(self):

self.SearchWindow.close()

if len(trains.trains) == 0:

self.error("Список поездов пуст")

else:

self.SearchWindow.show()

def error(self, t):

self.ErrorWindow.close()

self.ErrorWindow.display(t)

def close(self):

self.ErrorWindow.close()

self.InputWindow.close()

self.SearchWindow.close()

super().close()

class InputWindow(W.QWidget):

def \_\_init\_\_(self, \*args):

super().\_\_init\_\_()

uic.loadUi('./windows/InputWindow.ui', self)

self.EnterButton.clicked.connect(self.input)

self.ClearButton.clicked.connect(self.clear)

self.setWindowTitle("Ввод поездов")

def clear(self):

self.DestInput.clear()

self.NumberInput.clear()

self.TimeInput.clear()

def input(self):

if len(self.NumberInput.text().strip()) == 0 or \

len(self.TimeInput.text().strip()) == 0 or \

len(self.DestInput.text().strip()) == 0:

self.clear()

return

num = self.NumberInput.text()

try:

num = int(num)

except ValueError:

self.clear()

return

time = self.TimeInput.text().strip()

try:

time = dt.strptime(time, "%H:%M")

except ValueError:

self.clear()

return

dest = self.DestInput.text().strip()

trains.append(Train(num, dest, time))

self.clear()

class ErrorWindow(W.QWidget):

def \_\_init\_\_(self, \*args):

super().\_\_init\_\_()

uic.loadUi('./windows/ErrorWindow.ui', self)

self.OkButton.clicked.connect(self.close)

self.setWindowTitle("Ошибка")

def display(self, t):

self.ErrorBrowser.setText(f"Ошибка!\n\n{t}")

self.show()

class SearchWindow(W.QWidget):

def \_\_init\_\_(self, \*args):

super().\_\_init\_\_()

uic.loadUi('./windows/SearchWindow.ui', self)

self.SearchByNumButton.clicked.connect(self.search\_by\_num)

self.SearchByDestButton.clicked.connect(self.search\_by\_dest)

self.ExitButton.clicked.connect(self.close)

self.setWindowTitle("Поиск")

def search\_by\_num(self):

if len(self.QueryInput.text().strip()) == 0:

self.ResultsBrowser.setText("Введите запрос")

return

query = self.QueryInput.text().strip()

self.clear()

try:

query = int(query)

except ValueError:

self.ResultsBrowser.setText("Введите число")

return

res = trains.search\_by\_num(query)

if res is None:

self.ResultsBrowser.setText("Поезд не найден")

else:

self.ResultsBrowser.setText(f"{res}")

def search\_by\_dest(self):

if len(self.QueryInput.text().strip()) == 0:

self.ResultsBrowser.setText("Введите запрос")

return

query = self.QueryInput.text().strip()

self.clear()

res = trains.search\_by\_dest(query)

if res is None:

self.ResultsBrowser.setText(f"Поезда в {query} не идут")

else:

self.ResultsBrowser.setText(f"До {query} следуют: \n{res}")

def clear(self):

self.ResultsBrowser.clear()

self.QueryInput.clear()

def close(self):

self.clear()

super().close()

a = W.QApplication(sys.argv)

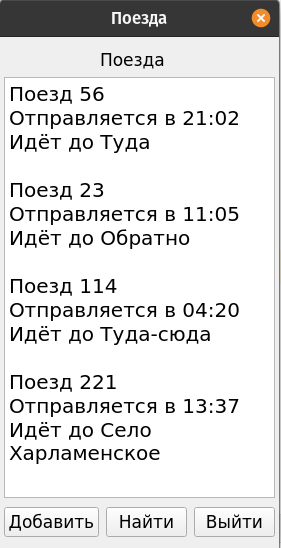
main = MainWindow()

main.show()

main.input()

exit(a.exec\_())

**Результат работы программы**



**Список используемой литературы**

1. Лутц М. Программирование на Python, том I, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.
2. Доусон М. Программируем на Python. – СПб.: Питер, 2014. – 416 с.