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

## **Лабораторная работа № 7**

GUI, классы, модуль Tkinter

**Задание 2 Вариант №11**

По дисциплине:

Программирование

Выполнил

студент 1 курса

группы 201-321

Колобов К.В.

Проверил:

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

# **МОСКВА 2020**

**Цель работы**

Получить практический навык в написании программ, использующих графический

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

**Постановка задачи**

Предметный указатель организован как линейный список. Каждая компонента указателя

содержит слово и номер страниц, на которых это слово встречается. Количество номеров страниц, относящихся к одному слову, может лежать в интервале от одного до десяти.

Составить программу, которая обеспечивает:

* начальное формирование предметного указателя;
* вывод предметного указателя;
* вывод номеров страниц для заданного слова.

**Теоретическая часть**

Для выполнения поставленной задачи используются: графический модуль PyQt5.

**Структура**

**Класс Point**

Описывает элемент предметного указателя. Хранит слово и список уникальных

страниц, на которых оно находится. Обеспечивает добавление страниц в список.

**Класс Pointer**

Описывает предметный указатель, хранит список элементов. Обеспечивает поиск,

добавление нового элемента.

**Класс MainWindow**

Обеспечивает работу главного окна. Здесь же представлен вывод предметного

указателя, кнопки «Найти» и «Добавить», вызывающие окно поиска и окно ввода соответственно.

**Класс InputWindow**

Обеспечивает ввод предметного указателя, контролируя при этом корректность

данных: не допускается ввод пустых полей, нецифровые символы в строке ввода страниц. Пополняет предметный указатель.

**Класс SearchWindow**

Выполняет поиск по предметному указателю. Принимает на вход слово, выводит

«Слово встречается на» или «Слово не найдено».

**Класс ErrorWindow**

Обеспечивает вывод сообщений об ошибках в отдельном окне. Например,

сообщение выводится, если пользователь нажимает «Найти» на главном окне при пустом предметном указателе.

**Описание программы**

Программа написана на Python 3.8, реализована в среде ОС GNU/Linux Ubuntu 20.04.

**Описание алгоритма**

1. Открыть главное окно с кнопками и ожидать действий от пользователя, реагируя на них соответствующим образом.

**Описание входных и выходных данных**

Пользователь вводит целые числа и строки, вывод осуществляется в GUI.

**Листинг программы**

import sys

from pathlib import Path

from typing import List, Set

import PyQt5.QtWidgets as Widgets

from PyQt5 import uic, QtCore

MAIN\_WINDOW\_PATH = Path('./ui/MainWindow.ui')

ERROR\_WINDOW\_PATH = Path('./ui/ErrorWindow.ui')

INPUT\_WINDOW\_PATH = Path('./ui/InputWindow.ui')

SEARCH\_WINDOW\_PATH = Path('./ui/SearchWindow.ui')

def sltext(text: str) -> str:

return text.lower().strip()

class Point:

def \_\_init\_\_(self,

word: str = '',

pages: List[int] = None) -> None:

self.\_word = word.strip()

self.\_pages = set(pages) or set()

@property

def word(self) -> str:

return self.\_word

@property

def pages(self) -> List[int]:

return list(sorted(self.\_pages))

def update(self,

pages: List[int] or Set[int]) -> None:

self.\_pages = self.\_pages.union(pages)

def \_\_contains\_\_(self,

page: int) -> bool:

return page in self.pages

def \_\_str\_\_(self) -> str:

num = f"Слово '{self.word}'"

pages = ', '.join(str(page) for page in self.pages)

pages = f"На страницах: {pages}"

return f"{num}\n{pages}"

class Pointer:

def \_\_init\_\_(self,

points: List[Point] = None) -> None:

self.\_points = points or []

@property

def points(self) -> List[Point]:

return self.\_points

def search(self,

word: str) -> Point or None:

word = sltext(word)

for point in self.points:

if word == sltext(point.word):

return point

def add(self,

point: Point) -> None:

if point.word not in self:

self.\_points += [point]

return

for i in range(len(self)):

if self.\_points[i].word == point.word:

self.\_points[i].update(point.pages)

def \_\_contains\_\_(self,

word: str) -> bool:

return any(word.strip() == point.word for point in self)

def \_\_getitem\_\_(self,

word: str) -> Point or None:

word = word.strip()

if word not in self:

return

for point in self:

if word == point.word:

return point

def \_\_iter\_\_(self) -> iter:

return iter(self.points)

def \_\_str\_\_(self) -> str:

if len(self) == 0:

return ''

corner = '-' \* 25

inside = '\n' + '-' \* 20 + '\n'

points = inside.join(str(point) for point in self.points)

return f"{corner}\n{points}\n{corner}"

def \_\_len\_\_(self) -> int:

return len(self.points)

pointer = Pointer()

class MainWindow(Widgets.QMainWindow):

def \_\_init\_\_(self, \*args) -> None:

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

uic.loadUi(MAIN\_WINDOW\_PATH, self)

self.initUI()

def initUI(self) -> None:

self.InputWindow = InputWindow(self, [])

self.ErrorWindow = ErrorWindow(self, [])

self.SearchWindow = SearchWindow(self, [])

self.setWindowTitle("Предметный указатель")

self.SearchButton.clicked.connect(self.search)

self.AddButton.clicked.connect(self.add)

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

self.checkThreadTimer = QtCore.QTimer(self)

self.checkThreadTimer.setInterval(500)

self.checkThreadTimer.start()

self.checkThreadTimer.timeout.connect(self.show)

def show(self) -> None:

self.PointsBrowser.setText(str(pointer))

super().show()

def add(self) -> None:

self.InputWindow.close()

self.InputWindow.show()

def search(self) -> None:

self.SearchWindow.close()

if len(pointer) == 0:

self.error("Предметный указатель пуст, негде искать")

else:

self.SearchWindow.show()

def error(self, msg: str) -> None:

self.ErrorWindow.close()

self.ErrorWindow.display(msg)

def close(self) -> None:

self.ErrorWindow.close()

self.InputWindow.close()

self.SearchWindow.close()

super().close()

class InputWindow(Widgets.QWidget):

def \_\_init\_\_(self, \*args) -> None:

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

uic.loadUi(INPUT\_WINDOW\_PATH, self)

self.initUI()

def initUI(self) -> None:

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

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

self.setWindowTitle("Ввод компонентов указателя")

def stext(self, obj) -> str:

return obj.text().strip()

def clear(self) -> None:

self.WordInput.clear()

self.PagesInput.clear()

def input(self) -> None:

if len(self.stext(self.WordInput)) == 0 or \

len(self.stext(self.PagesInput)) == 0:

self.clear()

return

word = self.stext(self.WordInput)

pages = self.stext(self.PagesInput)

if ',' in pages:

pages = pages.split(',')

else:

pages = pages.split()

try:

pages = [int(page) for page in pages]

except ValueError:

pass

else:

pointer.add(Point(word, pages))

finally:

self.clear()

class ErrorWindow(Widgets.QWidget):

def \_\_init\_\_(self, \*args) -> None:

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

uic.loadUi(ERROR\_WINDOW\_PATH, self)

self.initUI()

def initUI(self) -> None:

self.Box.clicked.connect(self.close)

self.setWindowTitle("Ошибка")

def display(self, message: str) -> None:

error = "<center><b>Ошибка!</b></center><br>"

self.ErrorBrowser.setText(f"{error}\n{message}")

self.show()

class SearchWindow(Widgets.QWidget):

def \_\_init\_\_(self, \*args) -> None:

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

uic.loadUi(SEARCH\_WINDOW\_PATH, self)

self.initUI()

def initUI(self) -> None:

self.SearchButton.clicked.connect(self.show\_results)

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

self.setWindowTitle("Поиск слова")

def show\_results(self) -> None:

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

self.ResultsBrowser.setText(

"<center><b>Введите запрос</b></center>")

return

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

self.clear()

point = pointer.search(query)

if point is None:

self.ResultsBrowser.setText("Слово не найдено")

else:

msg = f"Слово '{query}' встречается на: "

pages = ', '.join(str(page) for page in point.pages)

self.ResultsBrowser.setText(f"{msg} {pages} страницах")

def clear(self) -> None:

self.ResultsBrowser.clear()

self.QueryInput.clear()

def close(self) -> None:

self.clear()

super().close()

def main() -> None:

app = Widgets.QApplication(sys.argv)

RoutesWindow = MainWindow()

RoutesWindow.show()

exit(app.exec\_())

if \_\_name\_\_ == '\_\_main\_\_':

main()

**Результат работы программы**

Окна графического интерфейса пользователя.

**Список используемой литературы**

Документация языка Python: <https://docs.python.org/3/>