МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ

УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения

вычислительной техники и автоматизированных

систем

**Лабораторная работа №13**

по дисциплине: ООП

тема: «**Знакомство с библиотеками языка Python. PyQT.**»

Выполнил: студент группы ВТ-231

Масленников Д. А.

Проверили:  
Буханов Д. Г.

Белгород 2025

**Цель работы:** приобретение практических навыков создания приложений на языке Python, QT приложения.

**Задание**:

Для выполнения лабораторной работы требуется установить интерпретатор Python версии 3.6+. Выполнить написание программы-сценария в соответствии с вариантом задания (табл. 1). Провести тестирование. Оформить отчет.

**Вариант 14**

|  |  |
| --- | --- |
| 4 | QT-Библиотека (с поискам по тексту) |

Код программы:

import sys

from PyQt5.QtWidgets import (QApplication, QMainWindow, QWidget, QVBoxLayout,

QHBoxLayout, QLabel, QLineEdit, QPushButton,

QListWidget, QTextEdit, QMessageBox, QDialog,

QDialogButtonBox)

class Book:

"""Класс для представления книги в библиотеке"""

def \_\_init\_\_(self, title, author, year, content):

self.title = title

self.author = author

self.year = year

self.content = content

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

return f"{self.title} ({self.author}, {self.year})"

def contains\_text(self, text):

"""Проверяет, содержится ли текст в любом поле книги (без учета регистра)"""

text\_lower = text.lower()

return (text\_lower in self.title.lower() or

text\_lower in self.author.lower() or

text\_lower in str(self.year).lower() or

text\_lower in self.content.lower())

class Library:

"""Класс библиотеки с полной бизнес-логикой"""

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

self.books = []

self.init\_sample\_books()

def init\_sample\_books(self):

"""Инициализация тестовыми данными"""

sample\_books = [

("Clean Code", "Роберт Мартин", 2002, "пишем чистый код"),

("Па́ттерны проектирования", "Банда четырех", 1994, "стоит знать"),

("ООП", "Гради Буч", 1991, "БАЗА")

]

for title, author, year, content in sample\_books:

self.add\_book(title, author, year, content)

def add\_book(self, title, author, year, content):

"""Добавление новой книги с валидацией"""

if not all([title, author, year, content]):

raise ValueError("Все поля должны быть заполнены")

try:

year = int(year)

except ValueError:

raise ValueError("Год должен быть числом")

self.books.append(Book(title, author, year, content))

return True

def search\_books(self, text):

"""Поиск книг по тексту (без учета регистра)"""

if not text:

return self.books

return [book for book in self.books if book.contains\_text(text)]

def get\_all\_books(self):

"""Получение всех книг"""

return self.books

def get\_book\_by\_title(self, title):

"""Получение книги по названию"""

for book in self.books:

if book.title == title:

return book

return None

class LibraryApp(QMainWindow):

"""Класс приложения, отвечающий только за интерфейс"""

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

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

self.setWindowTitle("QT-Библиотека")

self.setGeometry(100, 100, 800, 600)

self.library = Library()

self.init\_ui()

def init\_ui(self):

"""Инициализация пользовательского интерфейса"""

main\_widget = QWidget()

self.setCentralWidget(main\_widget)

# Создание элементов интерфейса

self.search\_input = QLineEdit()

self.search\_button = QPushButton("Поиск")

self.add\_book\_btn = QPushButton("Добавить книгу")

self.books\_list = QListWidget()

self.book\_details = QTextEdit()

# Настройка элементов

self.search\_input.setPlaceholderText("Введите текст для поиска...")

self.book\_details.setReadOnly(True)

# Подключение сигналов

self.search\_button.clicked.connect(self.handle\_search)

self.add\_book\_btn.clicked.connect(self.show\_add\_dialog)

self.books\_list.itemClicked.connect(self.show\_book\_details)

# Компоновка интерфейса

search\_layout = QHBoxLayout()

search\_layout.addWidget(self.search\_input)

search\_layout.addWidget(self.search\_button)

search\_layout.addWidget(self.add\_book\_btn)

main\_layout = QVBoxLayout()

main\_layout.addLayout(search\_layout)

main\_layout.addWidget(QLabel("Список книг:"))

main\_layout.addWidget(self.books\_list)

main\_layout.addWidget(QLabel("Содержание:"))

main\_layout.addWidget(self.book\_details)

main\_widget.setLayout(main\_layout)

self.update\_books\_list()

# Методы работы с интерфейсом

def update\_books\_list(self, books=None):

"""Обновление списка книг в интерфейсе"""

self.books\_list.clear()

books\_to\_show = books if books is not None else self.library.get\_all\_books()

for book in books\_to\_show:

self.books\_list.addItem(str(book))

def show\_book\_details(self, item):

"""Отображение деталей книги в интерфейсе"""

title = item.text().split(" (")[0]

book = self.library.get\_book\_by\_title(title)

if book:

details = f"Название: {book.title}\nАвтор: {book.author}\nГод: {book.year}\n\n{book.content}"

self.book\_details.setPlainText(details)

def handle\_search(self):

"""Обработка поиска (только интерфейс)"""

search\_text = self.search\_input.text()

found\_books = self.library.search\_books(search\_text)

if search\_text and not found\_books:

QMessageBox.information(self, "Поиск", "Книги не найдены.")

self.update\_books\_list(found\_books)

def show\_add\_dialog(self):

"""Показ диалога добавления книги (только интерфейс)"""

dialog = QDialog(self)

dialog.setWindowTitle("Добавить книгу")

# Элементы формы

title\_input = QLineEdit()

author\_input = QLineEdit()

year\_input = QLineEdit()

content\_input = QTextEdit()

buttons = QDialogButtonBox(QDialogButtonBox.Ok | QDialogButtonBox.Cancel)

buttons.accepted.connect(lambda: self.try\_add\_book(

title\_input.text(),

author\_input.text(),

year\_input.text(),

content\_input.toPlainText(),

dialog

))

buttons.rejected.connect(dialog.reject)

# Компоновка

layout = QVBoxLayout()

layout.addWidget(QLabel("Название:"))

layout.addWidget(title\_input)

layout.addWidget(QLabel("Автор:"))

layout.addWidget(author\_input)

layout.addWidget(QLabel("Год:"))

layout.addWidget(year\_input)

layout.addWidget(QLabel("Содержание:"))

layout.addWidget(content\_input)

layout.addWidget(buttons)

dialog.setLayout(layout)

dialog.exec\_()

def try\_add\_book(self, title, author, year, content, dialog):

"""Попытка добавления книги (только интерфейс)"""

try:

self.library.add\_book(title, author, year, content)

self.update\_books\_list()

dialog.close()

except ValueError as e:

QMessageBox.warning(self, "Ошибка", str(e))

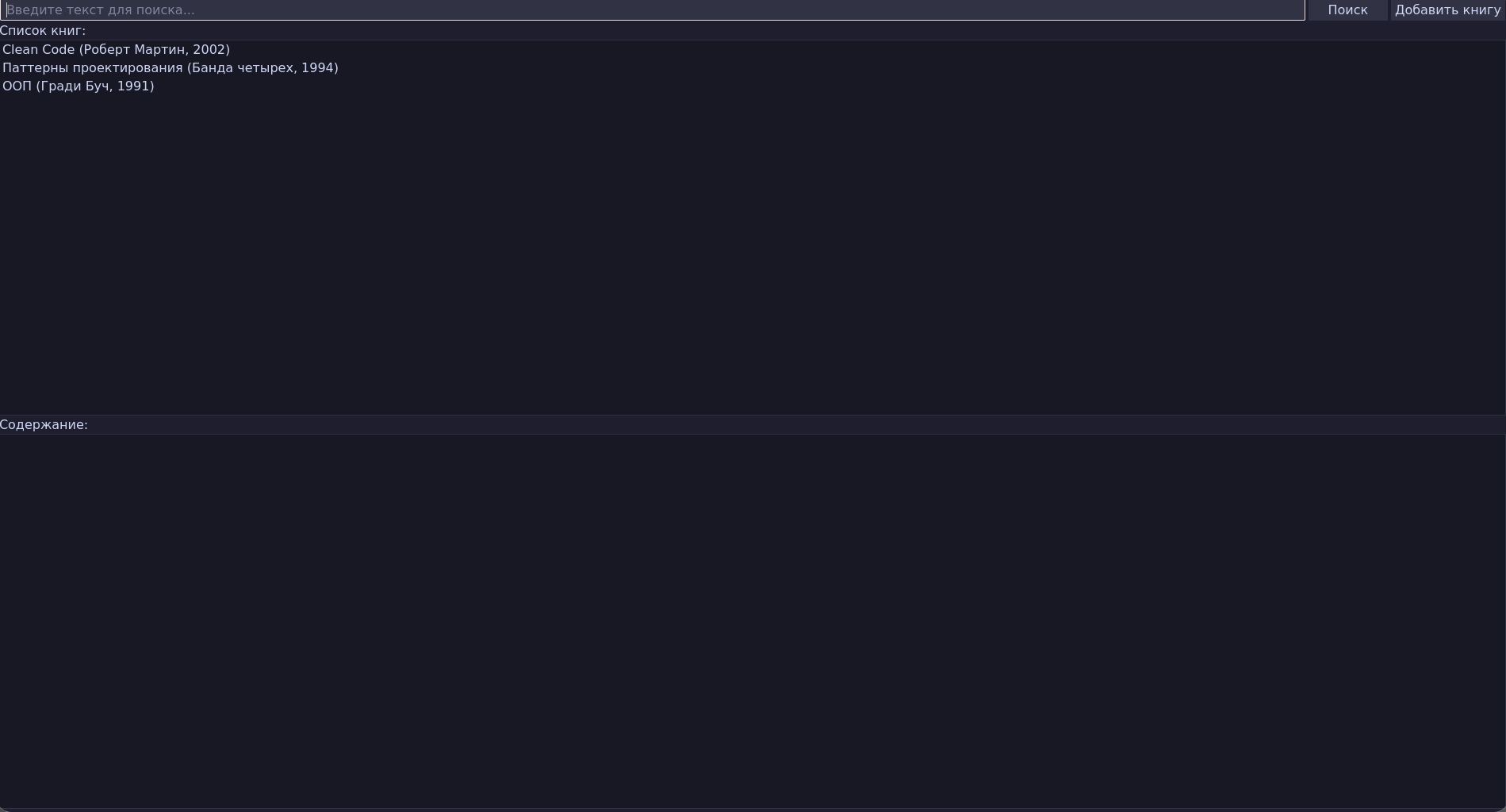
if \_\_name\_\_ == "\_\_main\_\_":

app = QApplication(sys.argv)

window = LibraryApp()

window.show()

sys.exit(app.exec\_())

Вывод программы: