

ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

Старший преподаватель

должность, уч. степень, звание

подпись, дата

С.Ю. Гуков

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3

СИСТЕМЫ КОНТРОЛЯ ВЕРСИЙ (VCS)

по курсу: ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4217

подпись, дата

И.М. Полозков

инициалы, фамилия

Санкт-Петербург 2025

Цель работы

Изучить предназначение и различные способы организаций систем контроля версий (Version Control System, VCS) Git. Познакомиться с операциями над файлами в репозитории и с приемами командной работы над проектом.

Задание

Задание можно выполнять в любой платформе, основанной на git (GitHub, GitLab, GitFlic, BitBucket и др.). Необходимо объединиться в команды по 2-4 человека. У каждого участника команды должен быть свой зарегистрированный аккаунт на выбранной платформе. Один из участников команды создает репозиторий и присоединяет к нему остальных участников. Необходимо придумать общий интерфейс программы (один из участников делает коммит набросков интерфейса в репозиторий, остальные обновляют у себя локальную копию репозитория). Далее каждый из участников обязательно в своей отдельной ветке выполняет свое задание по варианту (задания в команде должны различаться), периодически делая коммиты своих классов и изменений в коде в репозиторий и делая pull request в ветку dev, при этом обновляя (дополняя) свой локальный проект кодом этой ветки (с обновлениями коллег по команде). Ветки после слияния удалять не нужно. Также при pull request каждому необходимо сделать проверку (review) кода коллег – оставить несколько комментариев с советами и замечаниями по различным местам в коде. После того, как все участники команды сделают свое задание, ветка dev сливается в главную ветку master (main), и оформляется файл README.md с пояснениями о выполненных заданиях. В итоге должен получиться проект с единым интерфейсом, выполняющий несколько различных задач (по количеству участников команды). У каждого участника команды на компьютере должен находиться полный общий локальный проект (содержащий свое реализованное задание и код коллег по команде). В качестве проверки задания преподаватель также будет смотреть в онлайн репозитории созданные ветки и список коммитов – кто из участников,

когда и какие сделал изменения в проекте. Проект обязательно должен иметь графический пользовательский интерфейс (User Interface, UI), а также может быть написан на любом языке программирования. Замечание: разрешается выполнять проект в одиночку, при условии имитации работы в команде (регистрации 2-3 аккаунтов и выполнении различного задания с каждого аккаунта).

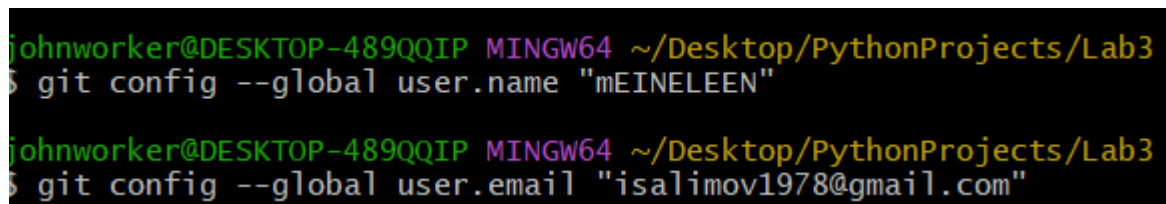
Вариант

5. Пользователь открывает файл с данными о численности населения России за последние 15 лет. Вывести эту информацию на экран в удобном табличном формате. По этим 6 данным построить графики зависимости от года. Вычислить максимальный процент прироста и убыли населения за год. Реализовать статистическое прогнозирование методом экстраполяции по скользящей средней на последующие N лет, вывести эту информацию на отдельном графике либо закрасить другим цветом на том же.

17. Пользователь открывает файл с данными о проценте детей, рожденных вне брака за последние 15 лет. Вывести эту информацию на экран в удобном табличном формате. По этим данным построить графики зависимости от года. Вычислить максимальный и минимальный процент изменения этих данных за год. Реализовать статистическое прогнозирование методом экстраполяции по скользящей средней на последующие N лет, вывести эту информацию на отдельном графике либо закрасить другим цветом на том же.

Решение

Сначала был произведён вход в основной аккаунт. См. рис. 1.



```
johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3
$ git config --global user.name "mEINELEEN"

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3
$ git config --global user.email "isalimov1978@gmail.com"
```

Рисунок 1 – Вход в основной аккаунт

Далее была произведена инициализация локального репозитория и

переименование ветки в main, так как это рекомендовано GitHub. См. рис. 2.

```
johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3
$ git init
Initialized empty Git repository in C:/Users/johnworker/Desktop/PythonPro
ab3/.git/

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (master)
$ git branch -m main

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (main)
$
```

Рисунок 2 – Инициализация локального репозитория и переименование ветки

Далее локальный репозиторий был подключён к удалённому.

```
johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (main)
$ git remote add origin https://github.com/mEINELEEN/TPLr3.git
```

Рисунок 3 – Подключение локального репозитория к удалённому

Далее был создан и запущен первый коммит. См. рис. 4, 5.

```
johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (main)
$ git add .

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   .gitignore
    new file:   README.md

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (main)
$ git commit -m "Первый коммит в main"
[main (root-commit) 7204db4] Первый коммит в main
 2 files changed, 472 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 README.md

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 2.58 KiB | 2.58 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/mEINELEEN/TPLr3.git
 * [new branch]      main -> main
```

Рисунок 4 – Пуш коммита

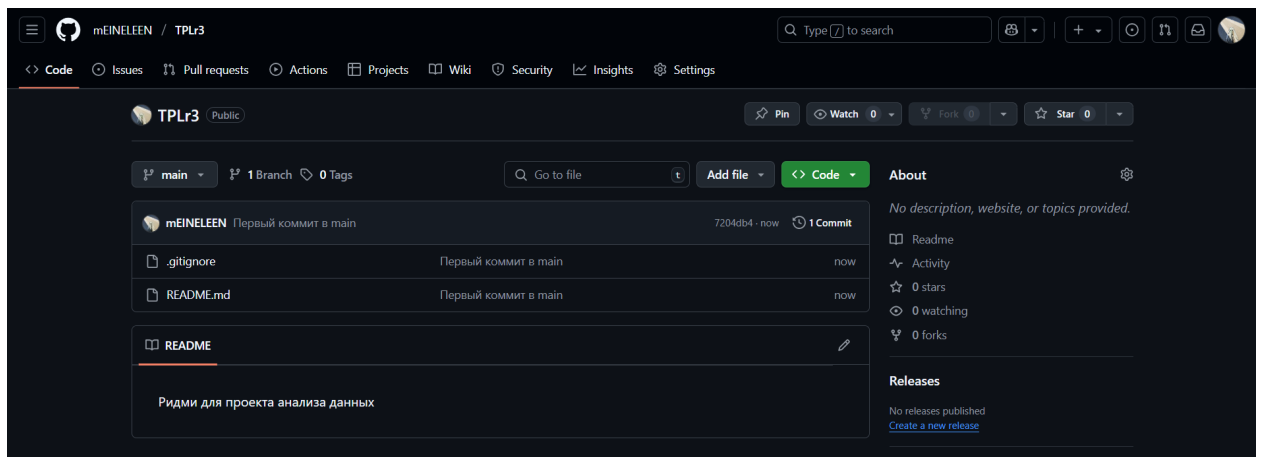


Рисунок 5 – Репозиторий после пуша

Далее были созданы ветки, необходимые для разработки по 17 варианту. А также произведён первый пуш, чтобы ветки появились на GitHub. См. рис. 5, 6.

```
johnworker@DESKTOP-4
$ git branch dev_1

johnworker@DESKTOP-4
$ git branch
* dev
  dev_1
  main
```

Рисунок 5 – Список созданных веток

```
johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (dev)
$ git push origin dev
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'dev' on GitHub by visiting:
remote:   https://github.com/mEINELEEN/TPLr3/pull/new/dev
remote:
To https://github.com/mEINELEEN/TPLr3.git
* [new branch]      dev -> dev

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (dev)
$ git checkout dev_1
Switched to branch 'dev_1'

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (dev_1)
$ git push origin dev_1
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'dev_1' on GitHub by visiting:
remote:   https://github.com/mEINELEEN/TPLr3/pull/new/dev_1
remote:
To https://github.com/mEINELEEN/TPLr3.git
* [new branch]      dev_1 -> dev_1
```

Рисунок 6 – Первый пуш на GitHub

Ветки отобразились на сайте GitHub. См. рис. 7.

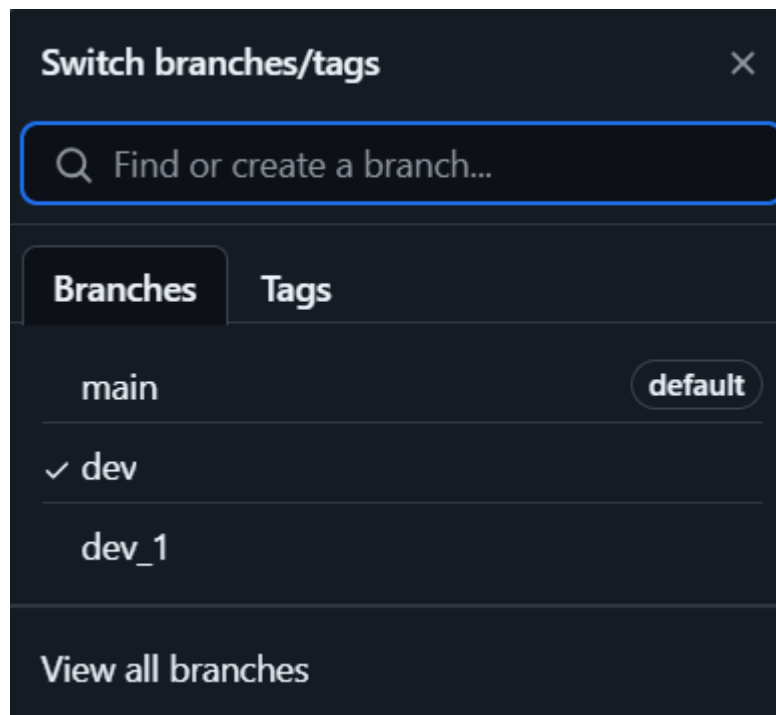


Рисунок 7 – Отображение ветки на GitHub

Для разработки интерфейса была использована библиотека PyQt5, для прорисовки графиков matplotlib, для работы с данными pandas. Программа в целом написана на языке Python. Полный код программы доступен в приложениях А, Б и В.

Сначала был разработан макет для лицевой страницы приложения. См. рис. 8.



Рисунок 8 – Макет лицевой страницы приложения

Далее была добавлен вкладка со статистикой по детям, рождённым вне брака, вкладка показана на рисунках 9, 10.

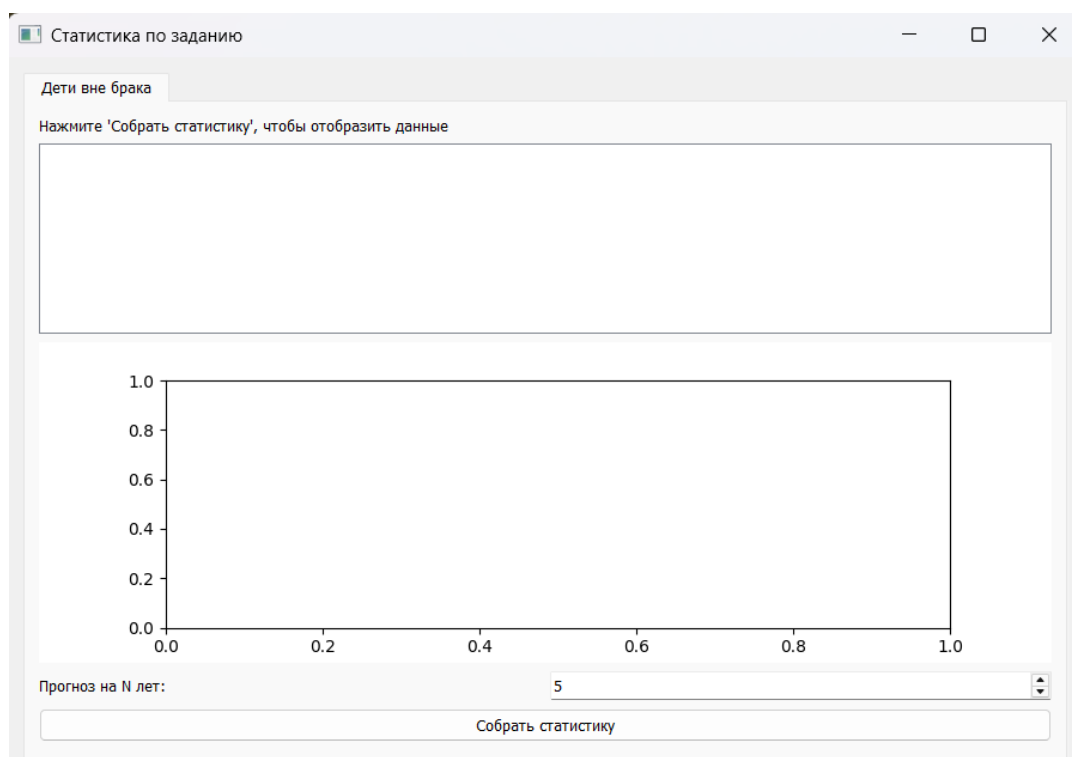


Рисунок 9 – Добавленная вкладка до нажатия на кнопку «Собрать статистику»

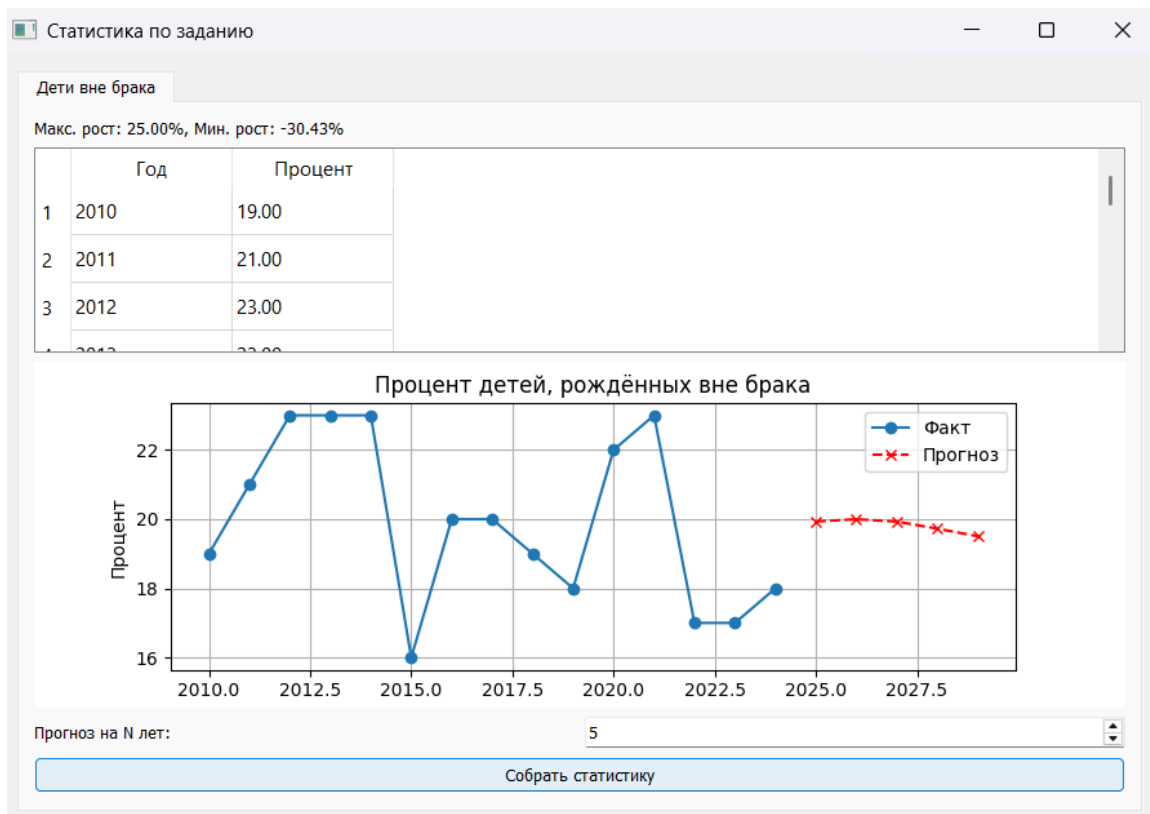


Рисунок 10 – Добавленная вкладка после нажатия на кнопку «Собрать статистику»

Далее показана работа скользящей средней. См. рис. 11.

```

15
[19, 21, 23, 23, 23, 16, 20, 20, 19, 18, 22, 23, 17, 17, 18]
15
[21, 23, 23, 23, 16, 20, 20, 19, 18, 22, 23, 17, 17, 18, 19.933333333333334]
15
[23, 23, 23, 16, 20, 20, 19, 18, 22, 23, 17, 17, 18, 19.933333333333334, 19.995555555555555]
15
[23, 23, 16, 20, 20, 19, 18, 22, 23, 17, 17, 18, 19.933333333333334, 19.995555555555555, 19.92859259259259]
15
[23, 16, 20, 20, 19, 18, 22, 23, 17, 17, 18, 19.933333333333334, 19.995555555555555, 19.92859259259259, 19.723832098765435]

```

Рисунок 11 – Скользящая средняя в работе

После окончания разработки был произведён Pull Request и merge. См. рис. 12, 13.

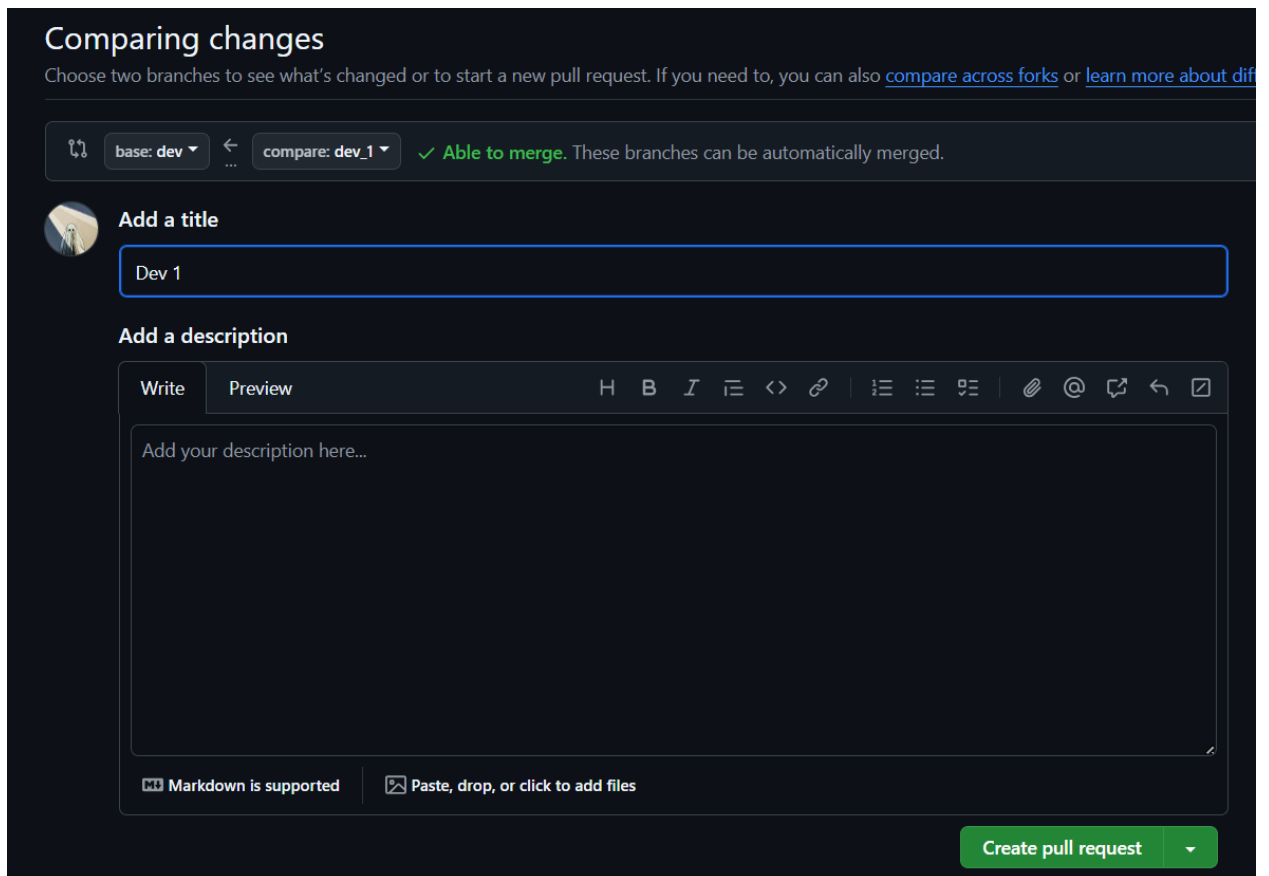


Рисунок 12 – Создание pull request

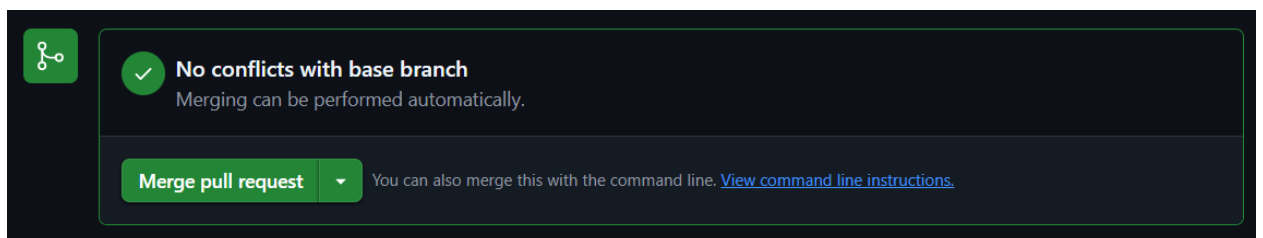


Рисунок 13 – Завершение pull request

Далее после слияния файлы были обновлены в локальном репозитории ветки main. См. рис. 14, 15.

```

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (dev_1)
$ git checkout dev
Switched to branch 'dev'

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3 (dev)
$ git pull origin dev
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (1/1), 897 bytes | 448.00 KiB/s, done.
From https://github.com/mEINELEEN/TPLeR3
* branch      dev      -> FETCH_HEAD
   7204db4..4ac1cd1 dev    -> origin/dev
Updating 7204db4..4ac1cd1
Fast-forward
 .gitignore       |    1 -
 children.py      |   94 ++++++
 main.py          |   29 ++++++
 tables/children.xlsx | Bin 0 -> 8662 bytes
4 files changed, 123 insertions(+), 1 deletion(-)
create mode 100644 children.py
create mode 100644 main.py
create mode 100644 tables/children.xlsx

```

Рисунок 14 – Обновление ветки dev в локальном репозитории

.git	01.06.2025 21:18	Папка с файлами	
.idea	01.06.2025 21:17	Папка с файлами	
.venv	01.06.2025 20:57	Папка с файлами	
__pycache__	01.06.2025 20:57	Папка с файлами	
tables	01.06.2025 21:18	Папка с файлами	
.gitignore	01.06.2025 21:18	Исходный файл G...	10 КБ
children.py	01.06.2025 21:18	JetBrains PyCharm	4 КБ
main.py	01.06.2025 21:18	JetBrains PyCharm	1 КБ
README.md	01.06.2025 17:01	Исходный файл ...	1 КБ

Рисунок 15 – Результат обновления

Далее был переключён пользователь и клонирован репозиторий, а далее создана ветка dev_2 для выполнения задания по 5 варианту. См. рис. 16, 17.

```

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3_2
$ git clone https://github.com/meINELEEN/TPLr3
Cloning into 'TPLr3'...
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 26 (delta 9), reused 20 (delta 4), pack-reused 0 (from 0)
Receiving objects: 100% (26/26), 13.20 KiB | 123.00 KiB/s, done.
Resolving deltas: 100% (9/9), done.

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3_2
$ cd TPLr3

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop/PythonProjects/Lab3_2/TPLr3 (main)
$ git config user.name
polozkow2

```

Рисунок 16 – Клонирование репозитория

```

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop
$ git checkout dev
branch 'dev' set up to track 'origin/dev'.
Switched to a new branch 'dev'

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop
$ git branch dev_2

johnworker@DESKTOP-489QQIP MINGW64 ~/Desktop
$ git checkout dev_2
Switched to branch 'dev_2'

```

Рисунок 17 – Создание необходимой ветки

Далее показано разработанное приложение по стандартному макету интерфейса. См. рис. 18.

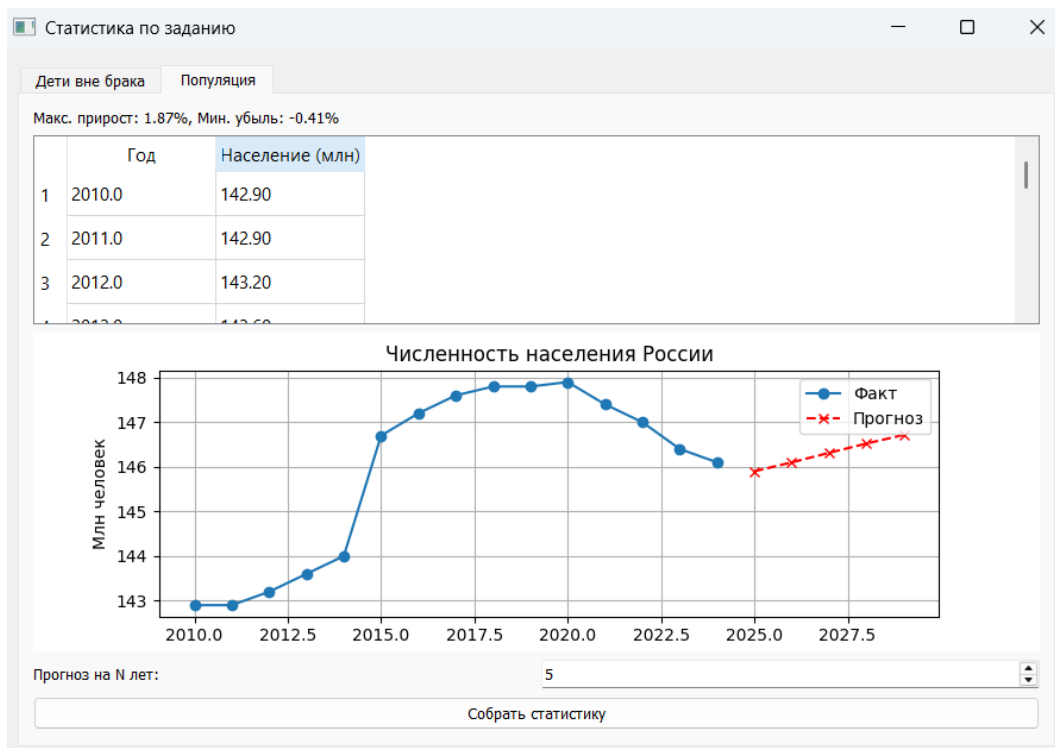


Рисунок 18 – Вторая разработанная вкладка с населением России
Далее показана работа скользящей средней во второй вкладке. См. рис. 19.

```
[142.9, 142.9, 143.2, 143.6, 144.0, 146.7, 147.2, 147.6, 147.8, 147.8, 147.9, 147.4, 147.0, 146.4, 146.1]
[142.9, 143.2, 143.6, 144.0, 146.7, 147.2, 147.6, 147.8, 147.8, 147.9, 147.4, 147.0, 146.4, 146.1, 145.9]
[143.2, 143.6, 144.0, 146.7, 147.2, 147.6, 147.8, 147.8, 147.9, 147.4, 147.0, 146.4, 146.1, 145.9, 146.1]
[143.6, 144.0, 146.7, 147.2, 147.6, 147.8, 147.8, 147.9, 147.4, 147.0, 146.4, 146.1, 145.9, 146.1, 146.31333333333333]
[144.0, 146.7, 147.2, 147.6, 147.8, 147.8, 147.9, 147.4, 147.0, 146.4, 146.1, 145.9, 146.1, 146.31333333333333, 146.5208888888889]
```

Рисунок 19 – Скользящая средняя во второй вкладке
Далее были оставлены комментарии к коду. См. рис. 20, 21.

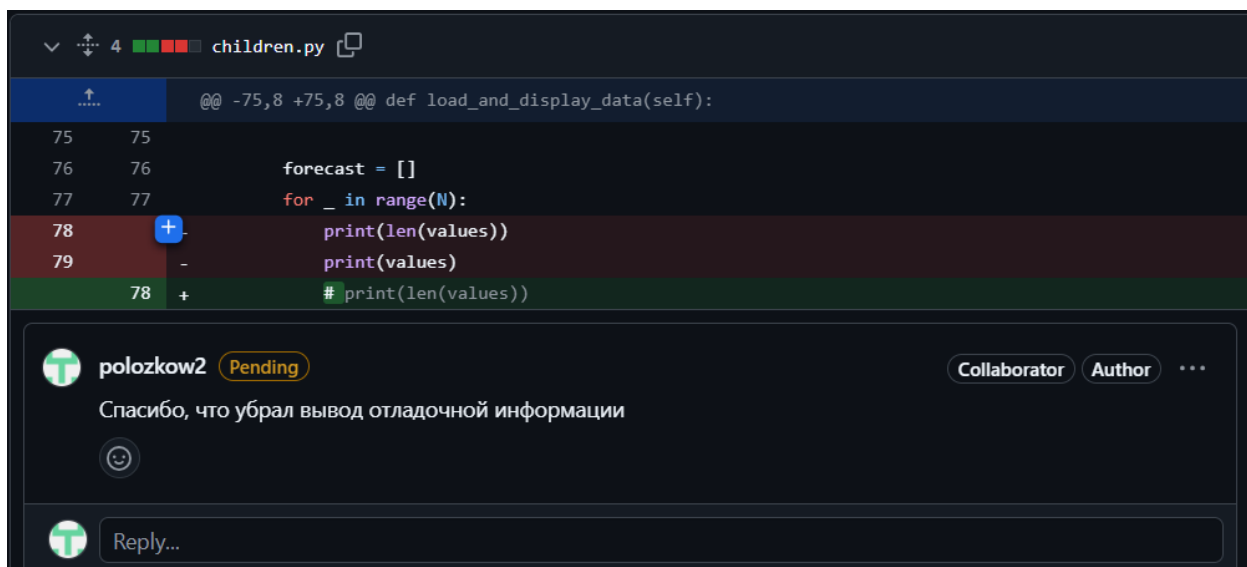


Рисунок 20 – Хвалебный комментарий к коду



Рисунок 21 – Второй комментарий к коду

Далее со второго аккаунта были оставлены комментарии к коду на код ревью. См. рис. 22, 23.

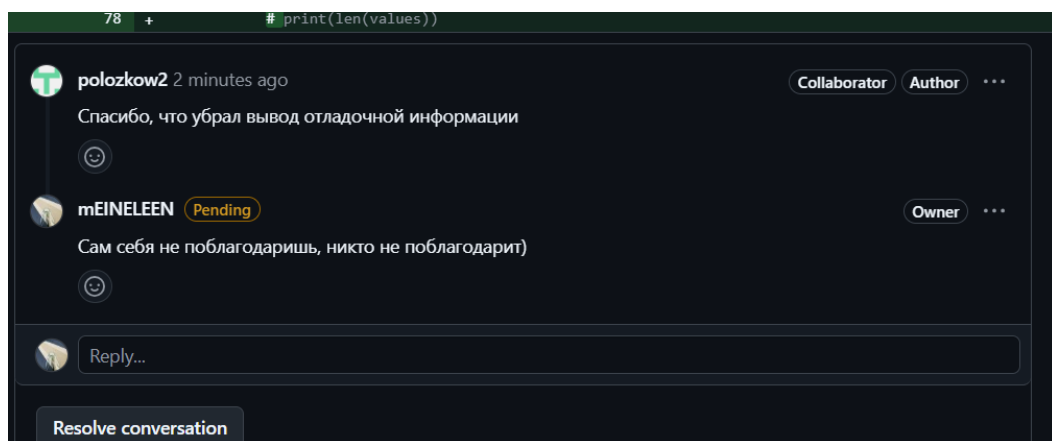


Рисунок 22 – Комментарии к коду со второго аккаунта

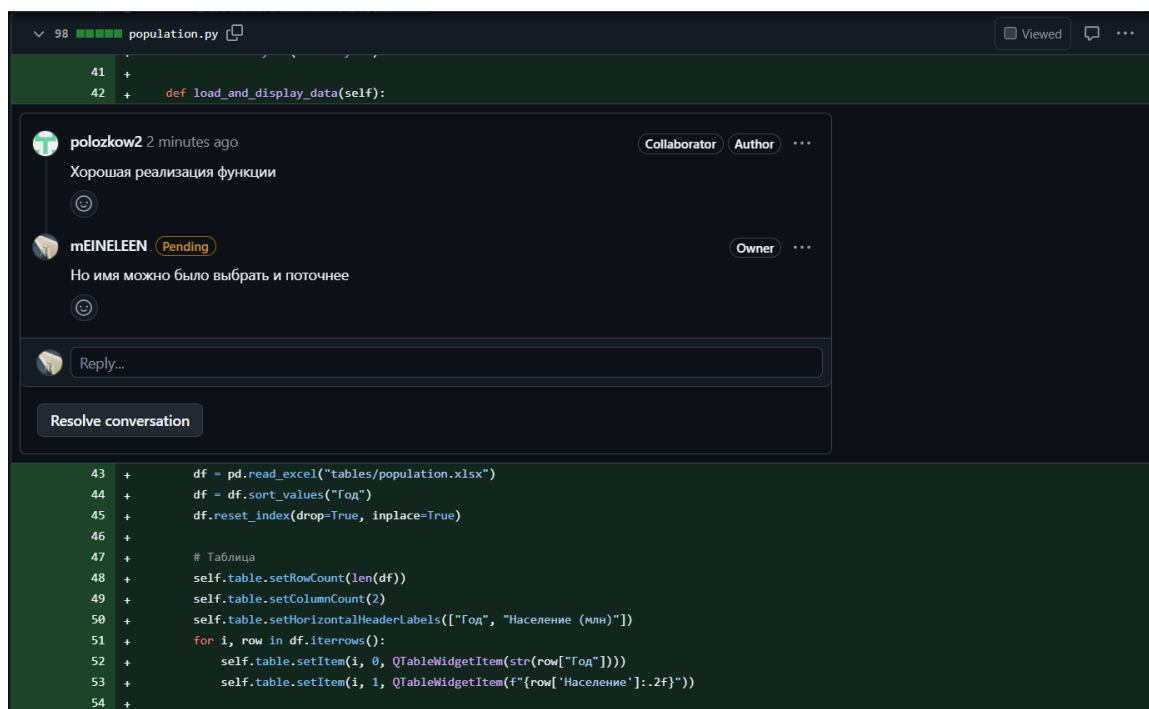


Рисунок 23 – Второй комментарий к коду со второго аккаунта

Далее показан результат объединения всех веток. См. рис. 24.

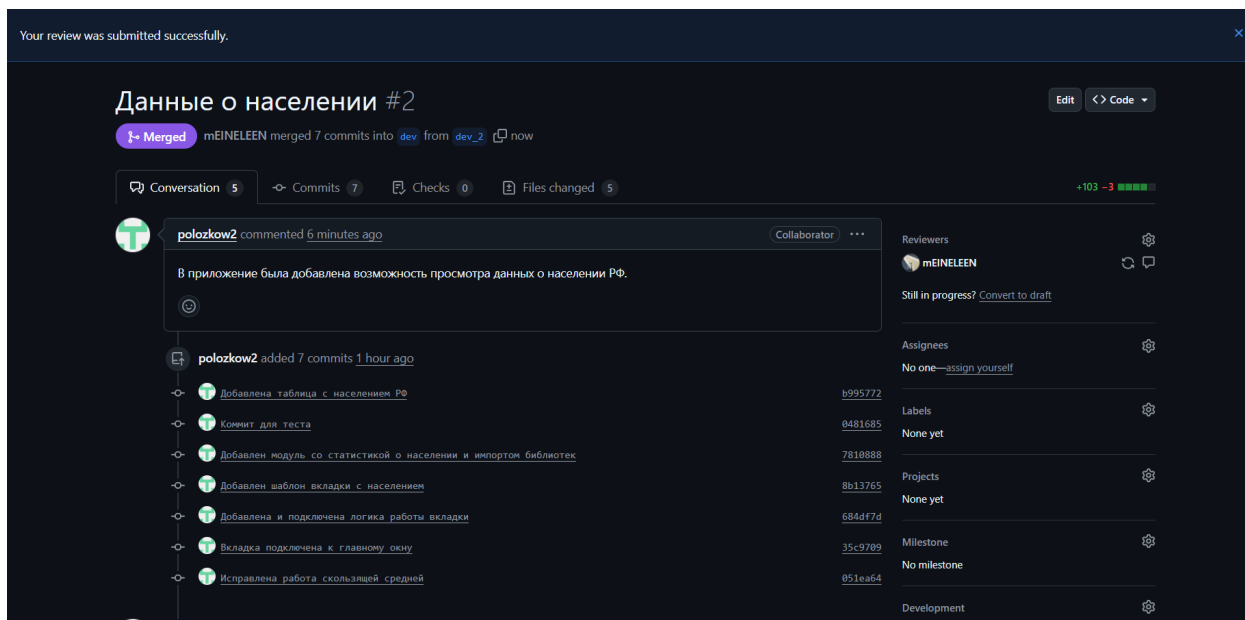


Рисунок 24 – Второе ветки dev_2 и dev

Далее показан итоговый репозиторий и история коммитов. См. рис. 25, 26.

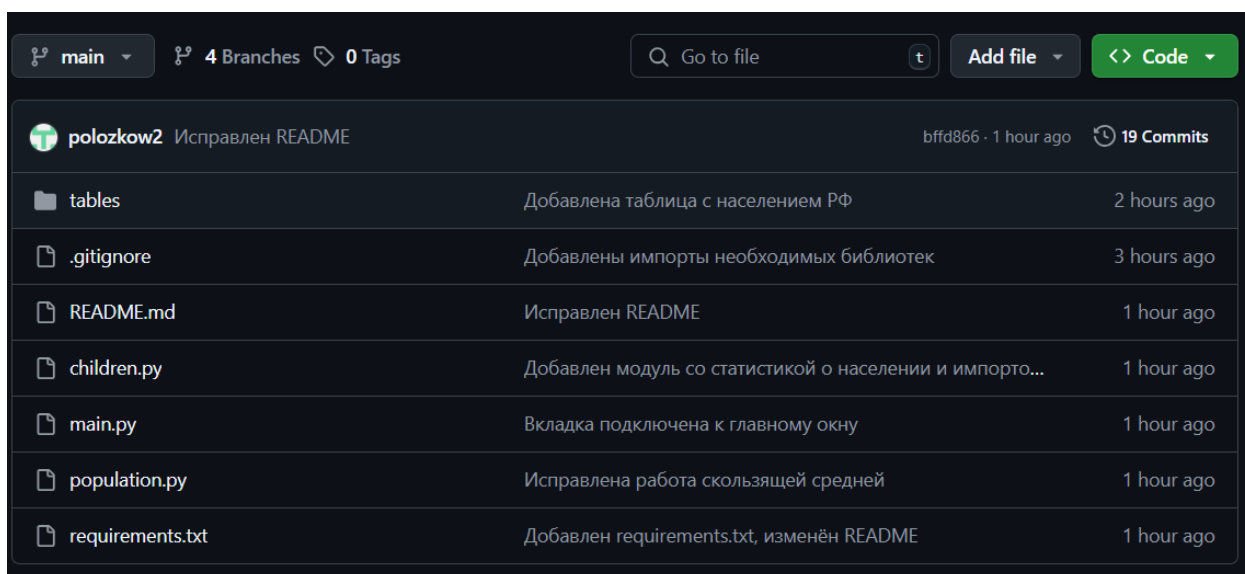


Рисунок 25 – Итоговый репозиторий

Commits on Jun 1, 2025		
Исправлен README	polozkow2 committed 1 hour ago	bffdb66 <>
Добавлен requirements.txt, изменён README	polozkow2 committed 1 hour ago	0a50b7e <>
Merge pull request #3 from mEINELEEN/dev	mEINELEEN authored 1 hour ago	Verified 7d074eb <>
Merge pull request #2 from mEINELEEN/dev_2	mEINELEEN authored 1 hour ago	Verified d866b6c <>
Исправлена работа скользящей средней	polozkow2 committed 1 hour ago	051ea64 <>
Вкладка подключена к главному окну	polozkow2 committed 1 hour ago	35c9709 <>
Добавлена и подключена логика работы вкладки	polozkow2 committed 1 hour ago	684df7d <>
Добавлен шаблон вкладки с населением	polozkow2 committed 1 hour ago	8b13765 <>
Добавлен модуль со статистикой о населении и импортом библиотек	polozkow2 committed 2 hours ago	7810888 <>
Коммит для теста	polozkow2 committed 2 hours ago	0481685 <>
Добавлена таблица с населением РФ	polozkow2 committed 3 hours ago	b995772 <>
Merge pull request #1 from mEINELEEN/dev_1	mEINELEEN authored 3 hours ago	Verified 4ac1cd1 <>
Добавлен excel, подключен модуль	mEINELEEN committed 3 hours ago	8b3ba68 <>
Добавлена реализация вкладки со статистикой	mEINELEEN committed 3 hours ago	a1b9f6e <>
Добавлен модуль children с библиотеками	mEINELEEN committed 3 hours ago	9a16b04 <>
Добавлен запуск окна	mEINELEEN committed 3 hours ago	fd50a34 <>

Рисунок 26 – Часть истории коммитов

Ссылка на GitHub

<https://github.com/mEINELEEN/TPLr3>

Для реализации задания по 17 варианту использовалась ветка dev_1 (использовалась пользователем mEINELEEN), для реализации задания по 5 варианту использовалась ветка dev_2 (использовалась пользователем polozkow2), для обобщения этих веток использовалась ветка dev и для применения финальных изменений использовалась ветка main.

Вывод

В ходе работы была реализована и протестирована информационная система, демонстрирующая основные возможности систем контроля версий, а также навыки командной разработки с использованием Git и GitHub. На основе двух заданий было создано приложение с графическим интерфейсом

на Python с использованием PyQt5, которое визуализирует и анализирует демографические данные, предоставляя прогноз методом скользящей средней. В процессе разработки были созданы отдельные ветки для каждого участника, организованы pull request'ы и проведён взаимный code review, что обеспечило совместную работу и контроль изменений. Итоговый проект представляет собой программную систему с модульной архитектурой, графиками и табличными представлениями данных, оформленную в соответствии с требованиями командной разработки.

Приложение А

«Код программы из модуля main.py»

```
import sys
from PyQt5.QtWidgets import (
    QApplication, QWidget, QVBoxLayout, QTabWidget,
)
from children import ChildrenStatsTab
from population import PopulationStatsTab

class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Статистика по заданию")
        self.resize(900, 600)

        layout = QVBoxLayout()

        self.tabs = QTabWidget()

        self.tabs.addTab(ChildrenStatsTab(), "Дети вне брака")
        self.tabs.addTab(PopulationStatsTab(), "Популяция")
        # Сюда можно добавлять другие вкладки

        layout.addWidget(self.tabs)
        self.setLayout(layout)

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

Приложение Б

«Код программы из модуля children.py»

```
import pandas as pd
from PyQt5.QtWidgets import (
    QWidget, QVBoxLayout, QPushButton,
    QLabel, QTableWidgetItem,
    QSpinBox, QHBoxLayout
)
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg
as FigureCanvas
from matplotlib.figure import Figure

class ChildrenStatsTab(QWidget):
    def __init__(self):
        super().__init__()

        self.layout = QVBoxLayout()

        self.label = QLabel("Нажмите 'Собрать статистику', чтобы
отобразить данные")
        self.layout.addWidget(self.label)

        self.table = QTableWidgetItem()
        self.layout.addWidget(self.table)

        self.plot_widget = FigureCanvas(Figure(figsize=(5, 3)))
        self.ax = self.plot_widget.figure.add_subplot(111)
        self.layout.addWidget(self.plot_widget)

        control_layout = QHBoxLayout()
        self.n_label = QLabel("Прогноз на N лет:")
        control_layout.addWidget(self.n_label)
        self.n_spinbox = QSpinBox()
        self.n_spinbox.setRange(1, 20)
        self.n_spinbox.setValue(5)
        control_layout.addWidget(self.n_spinbox)
        self.layout.addLayout(control_layout)

        self.button = QPushButton("Собрать статистику")
        self.button.clicked.connect(self.load_and_display_data)
        self.layout.addWidget(self.button)

        self.setLayout(self.layout)

    def load_and_display_data(self):
        # Чтение файла
        df = pd.read_excel("tables/children.xlsx")
        df = df.sort_values("Год")
        df.reset_index(drop=True, inplace=True)
```

```

# Отображение таблицы
self.table.setRowCount(len(df))
self.table.setColumnCount(2)
self.table.setHorizontalHeaderLabels(["Год", "Процент"])
for i, row in df.iterrows():
    self.table.setItem(i, 0,
QTableWidgetItem(str(row["Год"])))
    self.table.setItem(i, 1,
QTableWidgetItem(f"{row["Процент"]:.2f}"))

# Построение графика
self.ax.clear()
self.ax.plot(df["Год"], df["Процент"], marker='o',
label="Факт")

# Расчёт изменений
df['Δ%'] = df['Процент'].pct_change() * 100
max_growth = df['Δ%'].max()
min_growth = df['Δ%'].min()

self.label.setText(
    f"Макс. рост: {max_growth:.2f}%, Мин. рост:
{min_growth:.2f}%"
)

# Прогнозирование (скользящая средняя)
N = self.n_spinbox.value()
k = 15 # длина окна скользящего среднего

# Начальные значения — последние k точек
values = df['Процент'].tolist()[-k:]

forecast = []
for _ in range(N):
    # print(len(values))
    # print(values)
    next_value = sum(values[-k:]) / k
    forecast.append(next_value)
    values = values[1:] + [next_value] # удаляем
старое, добавляем новое

last_year = df['Год'].iloc[-1]
forecast_years = [last_year + i for i in range(1, N +
1)]

self.ax.plot(forecast_years, forecast, linestyle='--',
color='red', marker='x', label='Прогноз')

self.ax.set_title("Процент детей, рождённых вне брака")
self.ax.set_xlabel("Год")
self.ax.set_ylabel("Процент")
self.ax.legend()
self.ax.grid(True)

```

```
self.plot_widget.draw()
```

Приложение В

«Код программы из модуля population.py»

```
import pandas as pd
import numpy as np
from PyQt5.QtWidgets import (
    QWidget, QVBoxLayout, QLabel, QPushButton, QTableWidgetItem,
    QTableWidgetItem, QHBoxLayout, QSpinBox
)
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg
as FigureCanvas
from matplotlib.figure import Figure

class PopulationStatsTab(QWidget):
    def __init__(self):
        super().__init__()

        self.layout = QVBoxLayout()

        self.label = QLabel("Нажмите 'Собрать статистику', чтобы
отобразить данные")
        self.layout.addWidget(self.label)

        self.table = QTableWidgetItem()
        self.layout.addWidget(self.table)

        self.plot_widget = FigureCanvas(Figure(figsize=(5, 3)))
        self.ax = self.plot_widget.figure.add_subplot(111)
        self.layout.addWidget(self.plot_widget)

        control_layout = QHBoxLayout()
        self.n_label = QLabel("Прогноз на N лет:")
        control_layout.addWidget(self.n_label)
        self.n_spinbox = QSpinBox()
        self.n_spinbox.setRange(1, 20)
        self.n_spinbox.setValue(5)
        control_layout.addWidget(self.n_spinbox)
        self.layout.addLayout(control_layout)

        self.button = QPushButton("Собрать статистику")
        self.button.clicked.connect(self.load_and_display_data)
        self.layout.addWidget(self.button)

        self.setLayout(self.layout)

    def load_and_display_data(self):
        df = pd.read_excel("tables/population.xlsx")
        df = df.sort_values("Год")
        df.reset_index(drop=True, inplace=True)

        # Таблица
```

```

self.table.setRowCount(len(df))
self.table.setColumnCount(2)
self.table.setHorizontalHeaderLabels(["Год", "Население
(млн)"])
    for i, row in df.iterrows():
        self.table.setItem(i, 0,
QTableWidgetItem(str(row["Год"])))
        self.table.setItem(i, 1,
QTableWidgetItem(f"{row['Население']:.2f}"))

    # График
    self.ax.clear()
    self.ax.plot(df["Год"], df["Население"], marker='o',
label="Факт")

    # Процентные изменения
    df["Δ%"] = df["Население"].pct_change() * 100
    max_growth = df["Δ%"].max()
    min_growth = df["Δ%"].min()

    self.label.setText(
        f"Макс. прирост: {max_growth:.2f}%, Мин. убыль:
{min_growth:.2f}%"
    )

    # Прогнозирование (скользящее среднее с движущимся
окном)
    N = self.n_spinbox.value()
    k = 15 # длина окна, можно заменить на переменную

    # Убедимся, что данных минимум k
    if len(df) < k:
        self.label.setText("Недостаточно данных для расчёта
скользящей средней.")
        return

    # последние k значений
    values = df['Население'].tolist()[-k:]
    forecast = []

    for _ in range(N):
        next_value = sum(values) / k
        forecast.append(next_value)
        print(values)
        values = values[1:] + [next_value]

    last_year = df['Год'].iloc[-1]
    forecast_years = [last_year + i for i in range(1, N +
1)]

    # Рисуем прогноз
    self.ax.plot(forecast_years, forecast, linestyle='--',
color='red', marker='x', label='Прогноз')

```

```
self.ax.set_title("Численность населения России")
self.ax.set_xlabel("Год")
self.ax.set_ylabel("Млн человек")
self.ax.grid(True)
self.ax.legend()
self.plot_widget.draw()
```