

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ПЕТРА ВЕЛИКОГО
Физико-механический институт
Высшая школа прикладной математики и вычислительной физики

Отчёт по лабораторной работе №6

по дисциплине

"Автоматизация научных исследований"

Генерация списка литературы

Выполнил студент:
Лапина Ольга Константиновна
Группа 5040102/50201

Санкт-Петербург
2025

Содержание

1. Задание.....	3
2. Промпты.....	4
2.1. Промпт 1 - P1.....	5
2.2. Промпт 2 - P2.....	5
2.3. Промпт 3 - P3.....	5
3. Результаты.....	6
3.1. Промпт 1 - P1.....	6
Статьи для анализа DeepSeek:.....	12
Статьи для анализа ChatGPT:.....	17
3.2. Промпт 2 - P2.....	18
Статьи для анализа DeepSeek:.....	23
Статьи для анализа ChatGPT:.....	28
3.3. Промпт 3 - P3.....	29
Статьи для анализа DeepSeek:.....	35
Статьи для анализа ChatGPT:.....	41
4. Сравнительный анализ результатов.....	42
Таблица 1: Оценка ответов ИИ-моделей.....	42
5. Выводы.....	45

1. Задание

1. Метрики оценки

Каждый предложенный моделью источник оценивается по 5 критериям (максимум по каждому — 10 баллов):

- Существование: Реальность источника.
 - 0–6 баллов: Ссылка «битая» или ведет на страницу с ошибкой 404/403, однако статья с похожей тематикой или названием может быть найдена вручную через поиск.
 - 7–9 баллов: Ссылка корректна и ведет на существующую публикацию, но доступ к полному тексту статьи ограничен (платная подписка, требовалась регистрация и т.д.).
 - 10 баллов: Ссылка корректна, ведет на существующую публикацию, и полный текст статьи доступен для бесплатного ознакомления или загрузки.
- Оформление: Соответствие ссылки требованиям ГОСТ.
 - Здесь включается форматирование той информации, которая была получена от AI. Проверка на соответствие гостю. (0-10)
- Полезность:
 - Оценивается релевантность найденного источника заявленной теме исследования (на основе анализа аннотации, ключевых слов, названия).
- 0–10 баллов: Субъективная оценка соответствия.
 - 0 — тема полностью не совпадает,
 - 10 — статья идеально соответствует запросу.
- Новизна:

Оценивается актуальность источника на основе года его публикации.

 - 0–4 балла: Год публикации до 1980
 - 5–7 баллов: Год публикации с 1981 по 1996
 - 7–8 баллов: Год публикации с 1997 по 2012
 - 9–10 баллов: Год публикации 2013 и новее.
- Разнообразие типа: Тип источника (статья, книга и т.д.) и его характеристики.
 - 0–8 баллов: Найденный источник — научная статья в журнале или сборнике.
 - 6–10 баллов: Найденный источник относится к другому типу: монография (книга), диссертация, патент, стандарт, материалы конференции (proceedings), препринт, отчет и т.д. Конкретный балл (в диапазоне) выставляется с

учетом редкости и ценности данного типа источников для темы исследования.

Итоговый балл за промт складывается из суммы средних по критериям по источникам.

2. План и объем эксперимента

- Модели (2): Gemini, DeepSeek.
- Сценарий (1): «Формирование списка по теме».
- Промпты (3): Три различные по сложности и детализации формулировки темы (P1, P2, P3).
- Источников на промпт (3): По три уникальных и существующих источника от каждой модели.

Общий объем анализируемых данных:

2 модели × 3 промпта × 3 источника × 1 сценарий = 18 источников.

3. Процедура проведения

- Для каждого из трех промптов (P1, P2, P3) задать идентичную задачу моделям Gemini и DeepSeek.
- Из ответа каждой модели отобрать первые три уникальные и «живые» ссылки.
- Для каждого из 18 отобранных источников провести верификацию и оценку по 5 описанным критериям.
- Занести результаты в сводную таблицу (аналогичную приведенной) и рассчитать итоговые баллы.
- Провести сравнительный анализ результатов, выявив сильные и слабые стороны каждой модели в зависимости от сложности промпта.

4. Ожидаемый результат (Анализ)

- Сводная таблица с оценками 18 источников.
- Выводы о сравнительной эффективности моделей Gemini и DeepSeek.
- Анализ влияния сложности промпта на качество результатов (например, рост баллов по критериям «Существование» и «Оформление» от P1 к P3).
- Оценка слабых мест моделей (например, недостаточное разнообразие типов источников).

2. Промпты

2.1. Промпт 1 - P1.

Сгенерируй список научных источников, которые могут быть полезны для исследования данной темы. Укажи URL. Учитывай индекс цитируемости.

{название}

{описание}

2.2. Промпт 2 - P2.

Используя предоставленное название и краткое описание статьи, составь список релевантных научных источников. Укажи только реально существующие публикации: статьи, книги, мета-обзоры или авторитетные конференционные/журнальные работы. Отбирай источники по тематической близости, теоретической значимости и актуальности. Укажи валидные URL. Учитывай индекс цитируемости.

{название}

{описание}

2.3. Промпт 3 - P3.

Проанализируй предоставленные название и краткое описание научной статьи. На основе темы сформируй точный и проверяемый список литературы, состоящий исключительно из реально существующих научных публикаций. Не изобретай источники. Обязательно проверяй все ссылки, которые выдаешь. Включай только работы из рецензируемых журналов, научных книг, конференций или авторитетных репозиториев. Отбирай литературу по следующим критериям:

- строгая релевантность теме;
- фундаментальность и/или современность исследования;
- практическая ценность для написания статьи;
- разнообразие типов источников (обзоры, эксперименты, теория).
- индекс цитируемости

Вывод: авторов, год, точное название, журнал/конференцию или издателя, и при наличии — DOI. Если тема широкая, включи ключевые базовые источники и последние значимые работы. Страйся охватить основные направления, необходимые для полноценного академического исследования. Ко всем статьям, которые ты предложишь, укажи валидные ссылки, по которым можно прочитать полные версии текстов

{название}

{описание}

3. Результаты

В качестве темы выберу скорректированную тему своей бакалаврской работы: “Разработка программных интерфейсов в современных инженерных приложениях.”

В качестве описания использовался реферат и ключевые слова к ВКР:

Данная работа посвящена исследованию принципов разработки языковых программных интерфейсов для современных инженерных приложений с фокусом на программные комплексы для моделирования многофазных потоков. Основная цель заключается в проектировании и реализации программного интерфейса на Python с использованием Python/C API, обеспечивающего дублирование функциональности графического интерфейса пользователя. Проведен анализ современных подходов к проектированию API для инженерного ПО, включающий: классификацию API; сравнительный анализ технологий связывания языков; основные принципы проектирования языкового программного интерфейса; особенности проектирования на Python/C API. В результате были разработаны модули расширения, позволяющие работать с объектами разрабатываемого программного комплекса посредством скриптовых команд Python, а не только через графический интерфейс. Проведено тестирование полученных модулей с использованием рабочих скриптов, представленных в работе. С помощью разработанного программного интерфейса на Python можно выполнять циклические действия с моделью и расчётами, а также интегрировать сторонние библиотеки (NumPy, SciPy, Matplotlib) для расширения функциональности.

Ключевые слова: ПРОГРАММНЫЕ ИНТЕРФЕЙСЫ, ЯЗЫКОВЫЕ ПРОГРАММНЫЕ ИНТЕРФЕЙСЫ, PYTHON/C API, МОДЕЛИРОВАНИЕ МНОГОФАЗНОГО ПОТОКА, ПРОГРАММНЫЙ КОМПЛЕКС

Ссылка на ВКР: <https://elib.spbstu.ru/dl/3/2025/vr/vr25-2727.pdf/info>

За невозможностью доступа к Gemini (возникли проблемы с подключением к нему), была изменена модель: вместо Gemini используется ChatGPT.

3.1. Промпт 1 - Р1.

1. DeepSeek

Для исследования по разработке программных интерфейсов в инженерных приложениях я подобрал научные источники, сгруппированные по

ключевым направлениям вашей работы. Они помогут подтвердить ваши теоретические выводы и обозначить место вашего решения в текущем научно-техническом контексте.

В таблице ниже представлены основные источники по ключевым темам.

Категория	Название и автор(ы)	Год	Ключевая релевантность для вашей работы	URL / Источник	Уровень доверия и цитирования
API-дизайнер и архитектор ПО	<i>Designing Pythonic library APIs</i> , Ben Hoyt	2023	Практические принципы проектирования удобных «питонических» API, что напрямую относится к разработке ЯПИ для Python.	https://benhoyt.com/writings/python-api-design/	Авторитетный блог опытного разработчика, используется как практическое руководство.
API-дизайнер и архитектор ПО	<i>Интерфейсы в программировании</i> (лекционный материал)	Н/д	Теоретические основы и классификация программных интерфейсов (API/ABI), методы интеграции между языками — основа для	http://lectures.oostrovski/04-applications/21-interface-classes/	Учебный материал, полезен для структурирования теоретической главы.

аналитическо
й части.

Python/C API и связывани	<i>C/C++ из Python (C API), Habr</i>	2018	Практическое руководство с примерами кода по созданию C-расширений с использованием Python/C API (структуры, функции, аргументы).	https://habr.com/ru/article/469043/	Практическое техническое руководство, высокая релевантность для раздела реализации.
--------------------------	--------------------------------------	------	---	---	---

Python/C API и связывани	<i>Модуль объекты (официальная документация Python)</i>	Н/д	Исчерпывающее описание API для создания и управления модулями в C-расширениях ях (PyModuleDef, PyModule_Create).	https://digitology.tech/docs/python_3/c-api/module.html	Официальная документация, наиболее авторитетный источник.
--------------------------	---	-----	--	---	---

Инженерное ПО и научные	<i>toughio: Pre-and post-processing Python library for TOUGH</i>	2022	Пример реальной Python-библиотеки для работы с ПО	https://pypi.org/project/toughio/	Официальная страница открытой библиотеки, конкретный
-------------------------	--	------	---	---	--

вычисле
ния для пример в вашей
моделирован
ия предметной
многофазных
потоков области.
(TOUGH).
Демонстриру
ет подходы к
интеграции.

Инженерное ПО и научные вычисления 11 языков для DevOps..., Habr 2021 Описывает роль Python как языка для автоматизации и, интеграции и создания инструменто B, поддерживае т ваше обоснование выбора Python. <https://habr.com/ru/companies/first/articles/678994/> Популярный отраслевой ресурс, подтверждает стандартные практики.

Рекомендации по использованию источников

В контексте вашей работы источники могут быть использованы следующим образом:

- Для теоретической основы: Материал об интерфейсах в программировании послужит хорошим фундаментом для раздела с классификацией API и технологий связывания. Статья Бена Хойта поможет сформулировать принципы «питонического» дизайна **ЯПИ (Интересная опечатка от модели)**.

- Для технической реализации: Официальная документация Python и практическое руководство с Habr — ключевые источники для описания работы с Python/C API. Ссылки на них в разделе реализации придаст вашей работе техническую достоверность.
- Для контекста и обоснования: Библиотека toughio — отличный пример из вашей же предметной области (моделирование потоков), который можно использовать как аналог в анализе существующих решений. Статья про Python в DevOps поможет обосновать выбор Python как стандарта для автоматизации сложных систем.
- Для внесения научного вклада: Сформулируйте, чем ваше решение (например, специфика созданных объектов или интеграция с конкретными библиотеками) отличается от существующих подходов, подобных toughio. Это подчеркнет новизну и практическую значимость вашей работы.

Как провести дальнейший поиск

Чтобы дополнить список более формальными научными источниками, рекомендую следующие стратегии:

- Используйте ключевые термины: Поискайте научные статьи по запросам "scientific software API design", "Python/C API extension", "multiphase flow simulation scripting", "language binding for engineering software".
- Проверьте базы данных: Ищите в таких базах, как IEEE Xplore, ACM Digital Library, ScienceDirect (Elsevier), SpringerLink. Для доступа к ним может потребоваться использование сети университета.
- Изучите цитирования: Найдите ключевые статьи по вашей теме и посмотрите, какие работы на них ссылаются, а также какие источники цитируют они сами. Это можно сделать через Google Scholar.
- Смотрите доклады с конференций: Обратите внимание на материалы конференций по вычислительной гидродинамике (например, связанных с ASME), научным вычислениям или инженерии программного обеспечения.

Надеюсь, этот список будет полезен для подготовки вашего исследования. Если вам нужна помощь в формулировке запросов для поиска в конкретных научных базах данных — обращайтесь.

Доп. промпт: “Оформи как список литературы для научной статьи следуя ГОСТ:”

Ответ:

Коллега, оформил список литературы по ГОСТ Р 7.0.100-2018 (актуальная версия для библиографических описаний). Источники сгруппированы по типам документов, что соответствует стандартной практике.



Список литературы

1. toughio: Pre- and post-processing Python library for TOUGH (версия 1.3.0) [Электронный ресурс]. – URL: <https://pypi.org/project/toughio/> (дата обращения: 22.12.2025).
2. Hoyt, B. Designing Pythonic library APIs [Электронный ресурс] / B. Hoyt. – 2023. – URL: <https://benhoyt.com/writings/python-api-design/> (дата обращения: 22.12.2025).
3. Документация Python. Модуль объекты [Электронный ресурс]. – URL: https://digitology.tech/docs/python_3/c-api/module.html (дата обращения: 22.12.2025).
4. Интерфейсы в программировании: лекционный материал [Электронный ресурс]. – URL: <http://lectures.ostrov.ski/04-applications/21-interfaces/> (дата обращения: 22.12.2025).

5. C/C++ из Python (C API) [Электронный ресурс] // Habr. – 2018. – URL: <https://habr.com/ru/articles/469043/> (дата обращения: 22.12.2025).
6. 11 языков программирования для DevOps-инженеров [Электронный ресурс] // Habr / First Line Software. – 2021. – URL: <https://habr.com/ru/companies/first/articles/678994/> (дата обращения: 22.12.2025).

Примечание: В научной статье к каждому пункту из этого списка в основном тексте должна быть соответствующая квадратная ссылка-сноска (например, [1]). Для вашей работы наиболее важными источниками для цитирования являются №1 (как пример аналога в предметной области), №2 (принципы дизайна) и №3, №5 (техническая реализация).

Статьи для анализа DeepSeek:

Для дальнейшего анализа в п.4 выберем статьи:

- *Designing Pythonic library APIs*, Ben Hoyt - <https://benhoyt.com/writings/python-api-design/>
- *C/C++ из Python (C API)*, Habr - <https://habr.com/ru/articles/469043/>
- *Инженерное ПО и научные вычисления* - <https://habr.com/ru/companies/first/articles/678994/>

2. ChatGPT

Ниже приведён **список научных источников** (статьи, препринты, проекты и библиографии), которые могут быть полезны для исследования темы:

-
1. Препринты / научные статьи по API-дизайну

1. API design for machine learning software: experiences from the scikit-learn project

Поясняет принципы проектирования API в широко используемой Python-библиотеке.

 arXiv: <https://arxiv.org/abs/1309.0238> (arXiv)

2. Efficient and Accurate Automatic Python Bindings with cppy & Cling

Исследует подходы автоматической генерации привязок Python ↔ C++. Полезно при проектировании расширений.

 arXiv: <https://arxiv.org/abs/2304.02712> (arXiv)

3. Towards Lakosian Multilingual Software Design Principles

Обсуждает проектирование многоязычных интерфейсов (в т.ч. через FFI, например Pybind11).

 arXiv: <https://arxiv.org/abs/1906.08351> (arXiv)

2. Научные статьи (Engineering & API)

4. ogs5py: A Python-API for the OpenGeoSys 5 Scientific Modeling Project

Демонстрирует разработку Python API для научной моделирующей библиотеки (судьба многомодульных задач). *Полный текст доступен через Wiley.*

 <https://ngwa.onlinelibrary.wiley.com/doi/full/10.1111/gwat.13017> (Wiley Online Library)

5. swmm_api: A Python Package for Automation, Customization, and Visualization in SWMM-Based Urban Drainage Modeling

Пример современного Python API для моделирования инженерной задачи (urban drainage).

 <https://doi.org/10.3390/w17091373> (MDPI)

6. An application programming interface for multiscale shape-material modeling

Примеры проектирования API для сложных инженерных задач с указанием DOI.

 <https://doi.org/10.1016/j.advengsoft.2021.103046> (ScienceDirect)

📌 3. Методологические источники по API-дизайну

7. THE ART OF API DESIGN: BEST PRACTICES FOR MODERN SOFTWARE DEVELOPMENT

Обзор лучших практик API-дизайна (архитектурные паттерны, масштабируемость, качество интерфейса).

📄 <https://iaeme.com/Home/issue/IJETR?Volume=9&Issue=2> (ResearchGate)

8. On the Design, Development, and Testing of Modern APIs

(CMU/SEI)

Документ с принципами разработки и тестирования API высокого качества.

📄 https://www.sei.cmu.edu/documents/5955/On_the_Design_Development_and_Testing_of_Modern_APIS-1.pdf (SEI)

📌 4. Препринты / исследовательские разработки (Engineering / Python-API)

9. FluidSim: modular, object-oriented Python package for high-performance CFD simulations

Обсуждает проектирование API и архитектуру пакета для CFD-задач на Python.

📄 arXiv: <https://arxiv.org/abs/1807.01769> (arXiv)

10. Computational modeling of multiphase flow and transport with Python

Исследование, связанное с численным моделированием мультифазных потоков с помощью Python.

📄 ResearchGate (предварительная публикация):

https://www.researchgate.net/publication/253525142_Computational_modelling_of_multiphase_flow_and_transport_with_Python (ResearchGate)

📌 5. Дополнительные академические источники (API/Bindings)

11. Generating Bindings in MPICH

Исследование генерации API для MPI-библиотеки, применимо к

автоматизации создания привязок.

 arXiv: <https://arxiv.org/abs/2401.16547> (arXiv)



6. Академические/практические материалы (аспекты API)

12. **Fluent Interface (Wikipedia)** — концепция проектирования «читаемых» API.

 https://en.wikipedia.org/wiki/Fluent_interface (Wikipedia)

13. **Python/C API** — официальная документация Python (*не научная, но обязательна для практической части исследования*)

 <https://docs.python.org/3/c-api/index.html> (*URL напрямую из авторитетного источника — документации Python*)



Дополнительные материалы (контекст и примеры проектов)

Эти ресурсы **не обязательно научные публикации**, но являются полезными для практической части разработки интерфейсов:

- preCICE – библиотека API для многомодульного сопряжённого моделирования (имеет Python/Fortran/C++ интерфейсы) ([Wikipedia](#))
 <https://precice.org>
 - OpenMDAO – Python-базированный фреймворк мультидисциплинарной оптимизации (API-ориентирован) ([Wikipedia](#))
-



Советы по использованию этих источников

- ✓ Отдавайте предпочтение публикациям с **DOI** и высоким индексом цитирования;
- ✓ Используйте **arXiv** и **Wiley/ScienceDirect/MDPI** для академических статей с peer-review;
- ✓ Комбинируйте методологические работы (общие принципы API-дизайна) с **практическими реализациями и case studies**.

Если нужно, могу сформировать **библиографию в формате BibTeX** или помочь с **кратким анализом каждого источника** по теме!

Доп. промпт: “Оформи как список литературы для научной статьи следуя ГОСТ:”

Ответ:

Ниже оформлен **список литературы по теме разработки программных интерфейсов (API)** в научно-академическом стиле **по ГОСТ 7.0.5-2008** с указанием URL (для электронных публикаций).

Список литературы

1. **Buitinck L., Louppe G., Blondel M., Pedregosa F., Mueller A., Grisel O., Niculae V., Prettenhofer P., Gramfort A., Grobler J., Layton R., VanderPlas J., Joly A., Holt B., Varoquaux G.** API design for machine learning software: experiences from the scikit-learn project : ECML PKDD Workshop: Lang. for Data Mining & Machine Learning, 2013. – Режим доступа: <https://arxiv.org/abs/1309.0238> (дата обращения: 22.12.2025). ([Hugging Face](#))
2. **Python/C API Reference Manual.** Python 3.14.2 documentation. – Режим доступа: <https://docs.python.org/3/c-api/index.html> (дата обращения: 22.12.2025). ([Python documentation](#))
3. **Haryono S. A., Thung F., Lo D., Lawall J., Jiang L.** Characterization and Automatic Update of Deprecated Machine-Learning API Usages // arXiv, 2020. – Режим доступа: <https://arxiv.org/abs/2011.04962> (дата обращения: 22.12.2025). ([arXiv](#))
4. **PyART: Python API Recommendation in Real-Time / X. He, L. Xu, X. Zhang, R. Hao, Y. Feng, B. Xu** // arXiv, 2021. – Режим доступа: <https://arxiv.org/abs/2102.04706> (дата обращения: 22.12.2025). ([arXiv](#))
5. **Kundu B., Vassilev V., Lavrijsen W.** Efficient and Accurate Automatic Python Bindings with cppyy & Cling // arXiv, 2023. – Режим доступа: <https://arxiv.org/abs/2304.02712> (дата обращения: 22.12.2025). ([arXiv](#))

6. **An empirical study of the Python/C API on evolution and bug patterns** // Software: Practice and Experience, Wiley, 2022. – Режим доступа: <https://onlinelibrary.wiley.com/doi/abs/10.1002/sm.2507> (дата обращения: 22.12.2025). ([Wiley Online Library](#))
 7. **Tapis API Development with Python: Best Practices In Scientific REST API Implementation** / S. B. Cleveland, A. Jamthe, S. Padhy, J. Stubbs, M. Packard, J. Looney, S. Terry, R. Cardone, G. Jacobs // PEARC '20 Proceedings, 2020. – Режим доступа: по DOI: 10.1145/3311790.3396647 (дата обращения: 22.12.2025). ([ResearchGate](#))
-



Комментарии к списку:

- Публикации № 1–5 представляют **академические исследования проектирования API** и межъязыковых привязок (bindings) с конкретными примерами и анализом подходов, что важно для теоретической части работы.
 - Публикация № 2 — **официальная документация Python/C API**, обязательная к цитированию при описании технических аспектов разработки расширений на C/Python.
 - Публикации № 3 и № 6–7 дополняют обзор лучшими практиками API и автоматическими инструментами, включая REST API и динамические системы рекомендаций.
-

Если нужно оформить список в **Word (RTF/Markdown)** или добавить **официальные ГОСТ-ссылки для ISBN/DOI**, — напиши, я помогу!

Интересный момент: при доп.запросе сделал оформление не для всех источников.

Статьи для анализа ChatGPT:

Для дальнейшего анализа в п.4 выберем статьи:

- ogs5py: A Python-API for the OpenGeoSys 5 Scientific Modeling Project - <https://ngwa.onlinelibrary.wiley.com/doi/full/10.1111/gwat.13017> ([Wiley Online Library](#))

- THE ART OF API DESIGN: BEST PRACTICES FOR MODERN SOFTWARE DEVELOPMENT -
<https://iaeme.com/Home/issue/IJETR?Volume=9&Issue=2>
[\(ResearchGate\)](#)
- Efficient and Accurate Automatic Python Bindings with cppy & Cling
- <https://arxiv.org/abs/2304.02712> ([arXiv](#))

3.2. Промпт 2 - Р2.

1. DeepSeek

На основе предоставленного описания статьи по теме программных интерфейсов (API) в инженерных приложениях, вот список релевантных научных источников, отобранных по тематической близости, теоретической значимости и актуальности.



Ключевые источники по теме

Источник	Тип работы	Ссылка	Основная релевантность
Extending Python with C or C++	Официальная документация	https://docs.python.org/3/extending/extending.html	Каноническое руководство по Python/C API для создания модулей расширения, непосредственно соответствует части работы.
О проектировании API для инженерного ПО: практики и примеры из индустрии CAE	Статья (Rus.), журнал «Программная инженерия»	https://doi.org/10.17587/prin.12.xx-yy	Освещает специфику проектирования API для CAE-систем, включает анализ моделирования многофазных потоков

и интеграции скриптов.

Python for Scientific Computing and Engineering Applications	Методическое пособие (Springer)	https://link.springer.com/book/10.1007/978-3-030-50356-7	Раскрывает применение Python, NumPy, SciPy, Matplotlib для решения инженерных задач, что соответствует части работы по интеграции.
--	---------------------------------	---	--

Modern Software Engineering for Scientific Computing	Материалы конференции (Proc. of the Int. Conf. on Computational Science)	https://doi.org/10.1007/978-3-030-XXXXX-X_YY	Содержит доклады по архитектуре ПО для моделирования, включая разработку API и интеграцию с высокопроизводительными вычислениями.
--	--	---	---

API Design Patterns	Книга (Manning Publications)	https://www.manning.com/books/api-design-patterns	Систематизирует принципы и паттерны проектирования API, что полезно для раздела о классификации и проектировании.
---------------------	------------------------------	---	---

Научные основы и практика моделирования многофазных потоков: от алгоритмов к программным комплексам	Монография (Rus.)	https://urss.ru/cgi-bin/db.pl?lang=Ru&blang=r&u&page=Book&id=XXXX	Теоретическая база для предметной области исследования — моделирования многофазных потоков.
---	-------------------	---	---

Обзор релевантности и тематического охвата

Представленные выше источники в совокупности покрывают ключевые аспекты вашей работы.

- Технологическая основа: Официальная документация Python/C API является первоисточником для технической реализации модулей расширения, о которых говорится в описании.
- Проектирование API: Книга «API Design Patterns» предоставляет проверенные паттерны и принципы, необходимые для анализа и проектирования языковых интерфейсов для сложного инженерного ПО. Её подход дополняется статьей из русского журнала, которая фокусируется на специфике САЕ-систем.
- Предметная область: Монография по моделированию многофазных потоков задает научный контекст для разрабатываемого программного комплекса.
- Интеграция и экосистема: Пособие по Python для научных вычислений раскрывает потенциал использования NumPy, SciPy, Matplotlib для расширения функциональности, что является одной из целей работы.
- Инженерный контекст: Материалы конференции по инженерии научного ПО позволяют рассматривать разработку в более широком контексте современных практик создания исследовательского и инженерного программного обеспечения.

Рекомендации по дальнейшему поиску

Для углубленного изучения, рекомендую сконцентрироваться на следующих направлениях:

1. По конкретным САЕ-системам. Поиските документацию или научные публикации, посвященные API таких платформ, как OpenFOAM (для гидрогазодинамики), ANSYS, COMSOL или отечественной «ЛОГОС Платформе», которая, как указано в результатах поиска, также предоставляет API на Python и C.
2. По языкам связывания. Изучите темы «Cython», «PyBind11» и «cffi» — это современные альтернативы или надстройки над Python/C API, которые часто упоминаются в контексте научного ПО.

3. В академических базах данных. Используйте ключевые слова на русском и английском («программный интерфейс», «инженерное приложение», «API design», «scientific software», «multiphase flow simulation») в базах данных Scopus, Web of Science, РИНЦ, CyberLeninka, arXiv ([cs.SE](#)).

Если вас интересуют более узкие аспекты, например, паттерны проектирования API именно для задач оптимизации или конкретные кейсы интеграции с вычислительными кластерами, я могу помочь сформулировать более специфические запросы для поиска.

Доп. промпт: “Оформи как список литературы для научной статьи следуя ГОСТ:”

Ответ:

Оформляю список литературы согласно ГОСТ 7.1-2003 (Библиографическая запись. Библиографическое описание) и ГОСТ Р 7.0.100-2018 (Библиографическая запись. Сокращение слов и словосочетаний), учитывая требования к источникам на русском и английском языках.



Список литературы

1. Extending Python with C or C++ [Электронный ресурс] // Python Documentation. – Режим доступа:
<https://docs.python.org/3/extending/extending.html> (дата обращения: 22.12.2025). – Загл. с экрана.
2. Oliphant T. E. Guide to NumPy. – 2nd ed. – USA: CreateSpace Independent Publishing Platform, 2015. – 276 p.
3. Abrahams D., Grosse-Kunstleve R. W. Building Hybrid Systems with Boost.Python // C/C++ Users Journal. – 2003. – Vol. 21, № 7. – P. 29–36.
4. Virtanen P. et al. SciPy 1.0: fundamental algorithms for scientific computing in Python // Nature Methods. – 2020. – Vol. 17. – P. 261–272. – DOI: 10.1038/s41592-019-0686-2.
5. Hunter J. D. Matplotlib: A 2D Graphics Environment // Computing in Science & Engineering. – 2007. – Vol. 9, № 3. – P. 90–95. – DOI: 10.1109/MCSE.2007.55.

6. Jasak H., Jemcov A., Tukovic Z. OpenFOAM: A C++ Library for Complex Physics Simulations // International Workshop on Coupled Methods in Numerical Dynamics. – 2007. – P. 1–20.
7. Bloch J. How to design a good API and why it matters // Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications. – 2006. – P. 506–507. – DOI: 10.1145/1176617.1176622.
8. Ван Россум Г., Дрейк Ф. Л. Язык программирования Python: справочное руководство. – М.: Вильямс, 2002. – 454 с.
9. Исаев С. А., Жуков В. К. Моделирование многокомпонентных и многофазных течений: от алгоритмов к программным комплексам // Математическое моделирование. – 2015. – Т. 27, № 9. – С. 45–60.
10. Рихтер Д. Современные системы хранения данных: архитектура и API // Программирование. – 2018. – № 5. – С. 3–15. – DOI: 10.1134/S023444791805001X.
11. Гостев И. М., Смирнов А. А. Принципы проектирования скриптовых интерфейсов для САЕ-систем // Программная инженерия. – 2020. – Т. 11, № 3. – С. 124–133. – DOI: 10.17587/prin.11.124-133.
12. Классификация API инженерного программного обеспечения [Электронный ресурс] // Институт системного программирования РАН. – Режим доступа: https://www.ispras.ru/technologies/API_classification/ (дата обращения: 22.12.2025). – Загл. с экрана.



Пояснения к оформлению

Для корректного оформления согласно ГОСТ учтены следующие моменты:

- Порядок источников: Источники расположены в алфавитном порядке по фамилии первого автора или заглавию (для источников без автора), что является наиболее распространенным требованием.
- Транслитерация: Для иностранных источников и авторов использована прямая транслитерация, как это принято в российских научных публикациях (например, *Oliphant T. E., Hunter J. D.*).
- Электронные ресурсы: Для онлайн-источников (п. 1, 12) указаны обязательные элементы: общее обозначение материала ([Электронный ресурс]), режим доступа (URL) и дата обращения.

- Сокращения: Использованы стандартные сокращения согласно ГОСТ Р 7.0.100-2018 (например, Т. – том, № – номер, С. – страницы, Вол. – volume, Р. – pages).

Для завершения работы над статьей также рекомендуется проверить соответствие оформления цитирований в тексте (по системе "автор-год" или порядкового номера в квадратных скобках) представленному здесь списку литературы.

Интересный момент: на дополнительный промпт по оформлению, модель добавила ещё источников, однако не будем их брать для дальнейшего исследования, так как это может повлиять на оценку качества первоначальных промптов.

Статьи для анализа DeepSeek:

Для дальнейшего анализа в п.4 выберем статьи (тут сознательно выбираем и с битыми ссылками, так как нужно оценить можель на вариант битых ссылок):

- О проектировании API для инженерного ПО: практики и примеры из индустрии CAE - <https://doi.org/10.17587/prin.12.xx-yy>
- Python for Scientific Computing and Engineering Applications - <https://link.springer.com/book/10.1007/978-3-030-50356-7>
- API Design Patterns - <https://www.manning.com/books/api-design-patterns>

2. ChatGPT

Ниже — список реально существующих научных источников и авторитетных публикаций, релевантных к теме разработка программных интерфейсов (API) в инженерных приложениях с использованием Python/C API, с указанием валидных URL. Источники подобраны с учётом тематической близости, теоретической значимости и цитируемости в профицированных областях (API-дизайн, привязки языков, API-эволюция, связывание C/C++ с Python и инженерное ПО):

1. API-дизайн и интерфейсы в Python-софте

1. **Lars Buitinck et al.** — “**API design for machine learning software: experiences from the scikit-learn project**”.

Классическая работа по *проектированию API* в Python-пакетах, рассматривает принципы дизайна интерфейсов в научном ПО (эксперимент с реальной, широко используемой библиотекой).

📍 arXiv PDF: [API design for machine learning software — arXiv PDF \(scikit-learn\)](#)

2. **Н. И. Левонович** — **разработка многопоточного сервиса с Python и C/C++ (2024)**.

Статья из журнала *Моделирование и анализ данных* с обсуждением интеграции Python и C/C++, включая аспекты GIL и привязок языков, полезна для понимания API-связки Python/C.

📍 Журнал: [Разработка многопоточного сервиса с Python и C/C++ \(ссылка на статью\)](#)

2. Исследования Python/C API и привязок

3. “**An empirical study of the Python/C API on evolution and bug patterns**”.

Эмпирическое исследование, анализирующее C API Python: его развитие и характер ошибок — полезно для понимания стабильности интерфейса и практических подводных камней.

📍 Wiley Online Library: [Empirical study of Python/C API \(Wiley\)](#)

4. **Efficient and Accurate Automatic Python Bindings with cppyy & Cling** — Baidyanath Kundu et al. (2023).

Исследование современных автоматических привязок C++ к Python, затрагивает проблему *API-интероперабельности* и инструментальных решений.

📍 arXiv: [Efficient and Accurate Automatic Python Bindings \(cppyy & Cling\)](#)

5. **Svar: A Tiny C++ Header Brings Unified Interface for Multiple Programming Languages** — Yong Zhao et al. (2021).

Обсуждает дизайн интерфейсов, обеспечивающих доступ к C/C++

библиотекам из Python и других языков, что перекликается с задачей API-дизайна.

👉 arXiv: [Svar — unified interface for multiple languages \(arXiv\)](#)

6. **Bridging the Language Gap: An Empirical Study of Bindings for Open Source Machine Learning Libraries** — Hao Li & Cor-Paul Bezemer (2022).

Анализ существующих биндингов, демонстрирующий *практику API-интеграции* в реальных экосистемах научных библиотек.

👉 arXiv: [Bridging the Language Gap — empirical study of bindings \(arXiv\)](#)



3. Обзорные и справочные документы

7. **Python/C API Reference Manual — официальная документация Python.**

Основной авторитетный источник по Python/C API, описывает структуру, функции, управление объектами и расширение интерпретатора.

👉 Документация Python: [Python/C API Reference Manual — docs.python.org](#)

— Незаменимо при проектировании собственных расширений и модулей.

8. **SWIG — Simplified Wrapper and Interface Generator (описание в Wikipedia).**

Обзор инструмента для автоматической генерации привязок C/C++ к Python, широко используемого в инженерных приложениях (обеспечивает интерфейс между языками).

👉 Wikipedia: [SWIG — Wikipedia article](#)

9. **pybind (pybind11) — современный инструмент привязки C++/Python.**

Современная библиотека генерации биндингов C++ → Python; часто используется вместо прямого Python/C API в инженерных проектах.

👉 Wikipedia: [pybind — interface generating C++ library \(Wikipedia\)](#)

10. **Cython — язык/инструмент для упрощения разработки расширений Python.**

Обеспечивает механизм создания расширений и API-привязок,

описан в энциклопедическом источнике.

📍 Wikipedia: [Cython — Wikipedia article](#)



4. Литература по API-эволюции и проектированию

11. A Systematic Review of API Evolution Literature — ACM Digital Library.

Обзор академических работ по эволюции API, полезный для понимания изменений интерфейсов во времени и проектных подходов.

📍 ACM: [Systematic review of API evolution literature](#)



Дополнительные материалы (для теоретического обоснования)

- **SWIG Tutorial — PyTutorial98 (SWIG применение к Python/C).**

Исторический, но полезный практический материал о том, как строятся биндинги для Python через тулзы генерации.

📍 PDF: [Interfacing C/C++ and Python with SWIG \(PDF\)](#)

Если нужно, я могу сгруппировать эти источники в библиографический список по ГОСТ/АРА, добавить короткие аннотации под каждый, а также помочь составить структуру раздела «Литературный обзор» для вашей работы.

Доп. промт: “Оформи как список литературы для научной статьи следуя ГОСТ:”

Ответ:

Ниже приведён список литературы, оформленный по ГОСТ 7.0.5–2008 (система корреспонденции ISBN/ISSN, ссылки на действующие URL).

Источники релевантны теме разработка программных интерфейсов, Python/C API и привязок C/C++ в инженерных приложениях.

Список литературы

1. **Kundu B., Vassilev V., Lavrijsen W.** Efficient and Accurate Automatic Python Bindings with cppyy & Cling // *CoRR*. — 2023. — Vol. abs/2304.02712. — URL: <https://arxiv.org/abs/2304.02712> (дата обращения: 22.12.2025). ([arXiv](#))
2. **Lavrijsen W., Dutta A.** High-Performance Python-C++ Bindings with PyPy and Cling // *Proceedings of the 6th Workshop on Python for High-Performance and Scientific Computing (PyHPC '16)*, Salt Lake City, UT, USA, Nov. 14, 2016. — IEEE Press, 2016. — DOI:10.1109/PyHPC.2016.008. — URL: <https://ieeexplore.ieee.org/abstract/document/7836841/> (дата обращения: 22.12.2025). ([ResearchGate](#))
3. **Van Rossum G., Drake F. L. (ред.)** *Python/C API Reference Manual*. — Python Software Foundation, 2001–2025. — URL: <https://docs.python.org/3/c-api/index.html> (дата обращения: 22.12.2025). ([Python documentation](#))
4. **Beazley D. M.** *SWIG Reference Manual*. — Salt Lake City, UT: University of Utah, 1997. — Available at: <https://www.swig.org/Doc1.1/PDF/Reference.pdf> (дата обращения: 22.12.2025). ([Swig](#))
5. **Jakob W.** *pybind11: Seamless operability between C++11 and Python*. — GitHub repository, 2015–2025. — URL: <https://github.com/pybind/pybind11> (дата обращения: 22.12.2025). ([GitHub](#))

6. **Fernique P., Pradal C.** AutoWIG: Automatic Generation of Python Bindings for C++ Libraries // *arXiv:1705.11000*, 2017. — URL: <https://arxiv.org/abs/1705.11000> (дата обращения: 22.12.2025). ([arXiv](#))
 7. **Bacchio S., Finkenrath J., Stylianou C.** Lyncs-API: a Python API for Lattice QCD applications // *arXiv:2201.03873*, 2022. — URL: <https://arxiv.org/abs/2201.03873> (дата обращения: 22.12.2025). ([arXiv](#))
-

Дополнительные ресурсы (онлайн-документация и обзоры)

8. **Python Wiki.** Integrating Python with Other Languages: tools and bindings overview. — URL: <https://wiki.python.org/moin/IntegratingPythonWithOtherLanguages> (дата обращения: 22.12.2025). ([Python Wiki](#))
-

Если требуется, я могу помочь также **оформить библиографические ссылки в формате IEEE/APA**, подготовить **сноски под конкретные разделы текста** или **структуру раздела «Литературный обзор»** с кратким обзором каждого источника.

Интересное примечание: Не для всех сделал оформление и не практически не совпадают.

Статьи для анализа ChatGPT:

Для дальнейшего анализа в п.4 выберем статьи:

- Н. И. Левонович — разработка многопоточного сервиса с Python и C/C++ (2024) - [Разработка многопоточного сервиса с Python и C/C++ \(ссылка на статью\)](#)

- Svar: A Tiny C++ Header Brings Unified Interface for Multiple Programming Languages — Yong Zhao et al. (2021) - [Svar — unified interface for multiple languages \(arXiv\)](#)
- Bridging the Language Gap: An Empirical Study of Bindings for Open Source Machine Learning Libraries — Hao Li & Cor-Paul Bezemer (2022) - [Bridging the Language Gap — empirical study of bindings \(arXiv\)](#)

3.3. Промпт 3 - Р3.

1. DeepSeek

На основе вашего запроса я подобрал список литературы, непосредственно релевантной теме исследования. Он включает как классические работы, так и современные публикации из рецензируемых журналов и авторитетных источников.

Специфика вашей работы — разработка научного Python API для моделирования многофазных потоков — потребовала комбинирования источников из трех областей: проектирования API, научных вычислений на Python и конкретного предметного поля (CFD). Сводная таблица ключевых источников поможет вам ориентироваться в них.



Фундаментальные и актуальные источники

Автор(ы) / Организация	Год	Название	Тип источника / Журнал	Ключевая релевантность теме	DOI / Ссылка
Python Software Foundation	2025	<i>Python/C API Reference Manual</i>	Официальная документация языка	Первичный источник по Python/C API — основа для реализации вашего интерфейса.	https://docs.python.org/3/c-api/

Ван Россум, Г., Дрейк Ф.Л.	2023	<i>Python 3: Reference Manual</i>	Книга (официальное руководство)	Исчерпывающий справочник по языку Python, необходимый для понимания его внутренней семантики при связывании с C.	ISBN: 978-1-5443-9159-6
Langtangen, H.P.	2020	<i>A Primer on Scientific Programming with Python</i> (6th ed.)	Книга (Springer)	Классический учебник по использованию Python в научных целях, охватывает интеграцию с C/C++ и NumPy/SciPy.	10.1007/978-3-030-16877-3
Beazley, D.M.	2021	<i>Python Essential Reference</i> (5th ed.)	Книга (Addison-Wesley)	Подробное руководство, включающее разделы по расширению Python на C и созданию модулей.	ISBN: 978-0-13-489937-8
К.В. Рудаков и др. (ИПМ им. М.В. Келдыша РАН)	2020	<i>Инструменты и подходы к созданию программных интерфейсов</i>	Статья, Программирование	Прямая релевантность: анализ именно для научных вычислительных комплексов,	10.31857/S0556232320040067

для сложных
вычислитель
ных
комплексов

обсуждает
вопросы
производитель
ности и
связывания.

OpenFOAM Foundation	2025	<i>API Documentation for Programmers and C++ Guide</i>	Официальная документация языка ПО	Практический пример крупного CFD-пакета с поддержкой Python-скриптов ания, полезен для сравнения архитектур.	https://develop.openfoam.com/
---------------------	------	--	-----------------------------------	--	---

В.Е. Ампилов и др. (МФТИ)	2022	<i>Использование Python для автоматизации вычислительной гидродинамики в пакете OpenFOAM</i>	Доклад, Суперкомпьютерные дни	Пример конкретной реализации Python-скрипто для управления CFD-расчетами, близок к вашей практической задаче.	10.20948/sds-2022-221-228
---------------------------	------	--	-------------------------------	---	---

Oliphant, T.E.	2006	<i>Guide to NumPy</i>	Книга (свободно распространяющаяся)	Канонический источник по NumPy, архитектура которого построена на расширениях C и является	https://web.mit.edu/dvp/Public/numpybook.pdf
----------------	------	-----------------------	-------------------------------------	--	---

образцом для
научных
Python-библиот
ек.

Virtanen, P., et al. (SciPy community)	2020	<i>SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python</i>	Статья, Nature Methods	Фундаментальная научная статья, объясняющая архитектуру и философию ведущей научной библиотеки Python, с которой планируется интеграция.	10.1038/s41592-019-0686-2
--	------	---	------------------------	--	---

Мартин, Р. (Robert C. Martin)	2018	<i>Чистая архитектура . Искусство разработки программного обеспечения</i>	Книга (Питер)	Изложение принципов проектирования, таких как Dependency Inversion, критически важных для создания поддерживаемых API.	ISBN: 978-5-4461-0782-3
-------------------------------	------	---	---------------	--	-------------------------



Направления для дальнейшего поиска литературы

Если вы хотите углубить исследование, рекомендую обратить внимание на следующие области:

- Теория API-дизайна: Ищите работы М. Фаулера о шаблонах интеграции и Дж. Хикки о компонентно-ориентированном дизайне.
- Высокопроизводительные научные вычисления на Python: Изучите проекты Numba (ЛТ-компиляция) и mpi4py (параллельные вычисления), а также современные библиотеки для работы с данными, такие как xarray и Dask.
- Актуальные исследования в области CFD и ПО: Следите за публикациями в журналах "Computers & Fluids", "Journal of Computational Physics" и в трудах конференций AIAA SciTech. Многие современные CFD-пакеты активно развиваются Python-интерфейсы.

Для построения эффективного обзора литературы в вашей работе структурируйте его по ключевым разделам: общие принципы проектирования API, технологии связывания Python/C, использование Python в научных вычислениях и существующие подходы в ПО для моделирования потоков.

Доп. промпт: “Оформи как список литературы для научной статьи следуя ГОСТ:”

Ответ:

Вот список литературы, оформленный в соответствии с общими требованиями ГОСТ 7.0.5-2008 «Библиографическая ссылка» и ГОСТ 7.1-2003 «Библиографическая запись. Библиографическое описание».

Список литературы

1. Python/C API Reference Manual [Электронный ресурс]. – Python Software Foundation, 2025. – Режим доступа: <https://docs.python.org/3/c-api/> (дата обращения: 22.12.2025).
2. Ван Россум Г. Python 3: Reference Manual / Г. Ван Россум, Ф.Л. Дрейк. – Walnut Creek: Python Software Foundation, 2023. – 1536 с. – ISBN 978-1-5443-9159-6.
3. Лангтаген Х.П. A Primer on Scientific Programming with Python / Х.П. Лангтаген. – 6th ed. – Berlin, Heidelberg: Springer, 2020. – 942 с. –

(Texts in Computational Science and Engineering). – DOI:
[10.1007/978-3-030-16877-3](https://doi.org/10.1007/978-3-030-16877-3).

4. Бизли Д.М. Python Essential Reference / Д.М. Бизли. – 5th ed. – Boston: Addison-Wesley, 2021. – 768 с. – ISBN 978-0-13-489937-8.
5. Рудаков К.В. Инструменты и подходы к созданию программных интерфейсов для сложных вычислительных комплексов / К.В. Рудаков, А.С. Колдаев, С.А. Станкевич // Программирование. – 2020. – Т. 46, № 4. – С. 51–60. – DOI: [10.31857/S0556232320040067](https://doi.org/10.31857/S0556232320040067).
6. OpenFOAM API Documentation and Programmer's Guide [Электронный ресурс]. – OpenFOAM Foundation, 2025. – Режим доступа: <https://develop.openfoam.com/> (дата обращения: 22.12.2025).
7. Ампилов В.Е. Использование Python для автоматизации задач вычислительной гидродинамики в пакете OpenFOAM / В.Е. Ампилов, А.В. Горобцов, П.А. Михайлов // Суперкомпьютерные дни России: труды международной конференции. – Москва: Изд-во МГУ, 2022. – С. 221–228. – DOI: [10.20948/sds-2022-221-228](https://doi.org/10.20948/sds-2022-221-228).
8. Олифант Т.Э. Guide to NumPy / Т.Э. Олифант. – 2nd ed. – USA: Continuum Press, 2006. – 276 с. – Режим доступа: <https://web.mit.edu/dvp/Public/numpybook.pdf> (дата обращения: 22.12.2025).
9. Виртанен П. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python / П. Виртанен, Р. Гоммерс, Т.Э. Олифант и др. // Nature Methods. – 2020. – Vol. 17. – P. 261–272. – DOI: [10.1038/s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2).
10. Мартин Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин; пер. с англ. А. А. Слинкина. – Санкт-Петербург: Питер, 2018. – 352 с. – (Бестселлеры O'Reilly). – ISBN 978-5-4461-0782-3.

Ключевые принципы оформления, использованные в данном списке:

- Порядок расположения: Источники расположены в алфавитном порядке по фамилии первого автора или названию организации, что является стандартным требованием.
- Электронные ресурсы: Для онлайн-документов и статей обязательны указание типа ресурса ([Электронный ресурс]), режим доступа (URL) и дата обращения. Дата обращения — обязательный элемент, она указывается в скобках.

- DOI: Цифровой идентификатор объекта (DOI) является предпочтительной и устойчивой ссылкой на научную публикацию. Он указывается после библиографического описания.
- Транслитерация: Имена иностранных авторов переданы в оригинальном написании (латиницей), как это чаще всего принято в современных российских научных работах для удобства поиска.
- Описание книг: Указывается место издания (город) и издательство, общее количество страниц, серия (если есть) и ISBN.

Важное примечание: Точные требования к оформлению списка литературы (шрифт, отступы, пунктуация) могут дополнительно регламентироваться внутренними правилами вашего учебного заведения, издательства или научного журнала. Рекомендуется свериться с этими правилами перед финальной подачей статьи.

Статьи для анализа DeepSeek:

Для дальнейшего анализа в п.4 выберем статьи:

- Ван Россум Г. Python 3: Reference Manual / Г. Ван Россум, Ф.Л. Дрейк. – Walnut Creek: Python Software Foundation, 2023. – 1536 с. – ISBN 978-1-5443-9159-6.
- Бизли Д.М. Python Essential Reference / Д.М. Бизли. – 5th ed. – Boston: Addison-Wesley, 2021. – 768 с. – ISBN 978-0-13-489937-8.
- Рудаков К.В. Инструменты и подходы к созданию программных интерфейсов для сложных вычислительных комплексов / К.В. Рудаков, А.С. Колдаев, С.А. Станкевич // Программирование. – 2020. – Т. 46, № 4. – С. 51–60. – DOI: [10.31857/S0556232320040067](https://doi.org/10.31857/S0556232320040067).

2. ChatGPT

Ниже приведён **точный и проверяемый список реальной научной литературы**, релевантной теме *разработки программных интерфейсов (API) в современных инженерных приложениях*, с учётом API-дизайна, **Python/C API, связывания C/C++ и Python, а также интеграции API для научного ПО**. Все позиции — из рецензируемых журналов,

авторитетных конференций или признанных репозиториев (arXiv, Wiley, MDPI и др.), с достоверными сведениями об авторах, годах, публикациях и DOI там, где они есть.

1. Принципы проектирования API и API-дизайн

1.1 Фундаментальные и обобщающие работы

1. **Lars Buitinck et al. (2013).**

API design for machine learning software: experiences from the scikit-learn project.

arXiv:1309.0238 (репозиторий arXiv). ([arXiv](#))

URL: <https://arxiv.org/abs/1309.0238>

Почему важно: анализ API-дизайна большой библиотеки на Python (scikit-learn). Отличный пример проектирования устойчивого, расширяемого API в инженерной библиотеке.

2. **Na Xie (2024).**

Strategic approaches to API design and management.

Applied and Computational Engineering 64(1):230–236.

DOI:10.54254/2755-2721/64/20241395. ([ResearchGate](#))

Почему важно: современные концепции проектирования API, включая именование, документацию, управление версиями — фундаментально для API-проекта.

2. Python/C API и языковая интеграция

2.1 Исследования, посвящённые Python/C API

3. **R. Holmes et al. (2021) (примерное название — см. Wiley).**

An empirical study of the Python/C API on evolution and bug patterns.

Software: Practice and Experience. Wiley (2021). ([Wiley Online Library](#))

Почему важно: эмпирический анализ эволюции и ошибок Python/C API — полезно при проектировании устойчивых расширений.

2.2 Документация и детали API

4. **Python/C API Documentation (официальное собрание)** —

актуальные сведения о функциях, стабильности ABI и ограничениях.

The Python/C API, Release 3.10.1 (внутренний документ Python).

([Simuline](#))

Полезно для: понимания возможностей и ограничений Python/C API при разработке расширений на C/C++.



3. API в инженерных и научных приложениях

3.1 Scientific APIs / bindings

5. **Francesco Rizzi et al. (2020).**

Pressio: Enabling projection-based model reduction for large-scale nonlinear dynamical systems.

arXiv:2003.07798. ([arXiv](#))

Почему важно: описывает минималистичный C++ API с

Python-binding для масштабной научной задачи — аналогично требуемому паттерну API.

6. **Ashwin V. Mohanan et al. (2018).**

FluidSim: modular, object-oriented Python package for high-performance CFD simulations.

arXiv:1807.01769. ([arXiv](#))

Почему важно: пример API инженерного ПО для численного моделирования (CFD) с акцентом на расширяемость через Python.



4. Примеры API-ориентированных инженерных пакетов (с научной значимостью)

7. **M. Pichler (2025).**

swmm_api: A Python Package for Automation, Customization, and Visualization in SWMM-Based Urban Drainage Modeling.

Water 17(9):1373. DOI:10.3390/w17091373. ([MDPI](#))

Почему важно: практический пример Python API для инженерного моделирования (urban drainage), релевантный задачам автоматизации расчётов.

8. **Sarva T. Pulla et al. (2024).**

Transforming Hydrology Python Packages into Web Application Programming Interfaces: A Comprehensive Workflow Using Modern Web Technologies.

Water 16(18):2609. DOI:10.3390/w16182609. ([MDPI](#))

Почему важно: показывает процесс превращения вычислительных моделей в API-сервисы — полезно в контексте расширения функциональности инженерного ПО.

- 🧠 5. Дополнительные работы (радикально полезные для углублённой аналитики API)

5.1 API-дизайн и usability

9. Carnegie Mellon Software Engineering Institute (2023).

On the Design, Development, and Testing of Modern APIs.

Technical report. ([SEI](#))

Почему важно: содержит эмпирические принципы API-дизайна и улучшения UX, применимые к API для инженеров.

- 📚 6. Дополнительные ресурсы (база знаний)

Эти источники не являются научными публикациями в классическом смысле, но являются актуальными техническими *reference* по Python API и проектированию:

- 👉 **SWIG** — генератор связок языков для C/C++ и Python. ([Wikipedia](#))
👉 **Zen of Python** — принципы Python-дизайна. ([Wikipedia](#))
-

🔗 Проверяемые ссылки для чтения полных текстов

Источник	Ссылка
API design for machine learning software (scikit-learn)	https://arxiv.org/abs/1309.0238
Pressio API paper	https://arxiv.org/abs/2003.07798
FluidSim modular CFD	https://arxiv.org/abs/1807.01769
swmm_api Python package	https://doi.org/10.3390/w17091373

Hydrology Python APIs to web APIs	https://doi.org/10.3390/w16182609
Strategic approaches to API design	DOI:10.54254/2755-2721/64/20 241395
Wiley empirical Python/C API study	доступ через Wiley Online Library
Python/C API reference	публичный PDF



Рекомендации по использованию литературы

- ✓ Ключевые обзоры и фундаментальные источники: #1, #2, #9.
- ✓ Технические детали Python/C API: #3, #4.
- ✓ Примеры инженерных API-реализаций: #5, #6, #7, #8.
- ✓ Практика API-дизайна и usability: #2, #9.

Если нужно, могу также **составить аннотации к каждому источнику и/или структурировать литературу по разделам вашей статьи** (теория, API-Python/C, инженерное моделирование, примеры реализации).

Доп. промт: “Оформи как список литературы для научной статье следуя ГОСТ:”

Ответ:

Ниже приведён **список литературы, оформленный по ГОСТ 7.0.5-2008/ГОСТ Р 7.0.5-2008** для вашей научной статьи по теме *разработка программных интерфейсов (API) в современных инженерных приложениях с использованием Python/C API*, с достоверными данными об авторах, годах, местах публикации и, где возможно, DOI/URIs для доступа:

-
1. Buitinck L., Louppe G., Blondel M., Pedregosa F., Mueller A., Grisel O., Niculae V., Prettenhofer P., Gramfort A., Grobler J., Layton R., VanderPlas J., Joly A., Holt B., Varoquaux G. *API design for machine*

- learning software: experiences from the scikit-learn project.* — In: **Proc. of ECML PKDD Workshop: Languages for Data Mining and Machine Learning**, 2013. — Pp. 108–122. — URL: <https://arxiv.org/abs/1309.0238> ([Scikit-learn](#))
2. **Rizzi F., Blonigan P. J., Parish E. J., Carlberg K.** *Pressio: enabling projection-based model reduction for large-scale nonlinear dynamical systems.* — [arXiv preprint](#), 2020. — URL: <https://arxiv.org/abs/2003.07798> ([arXiv](#))
 3. **Virtanen P., Gommers R., Oliphant T. E., Haberland M., Reddy T., Cournapeau D., et al.** *SciPy 1.0 — Fundamental algorithms for scientific computing in Python.* — [arXiv preprint](#), 2019. — URL: <https://arxiv.org/abs/1907.10121> ([arXiv](#))
 4. **Python Software Foundation.** *Python/C API Reference Manual. Python 3 C API.* — Online. — URL: <https://docs.python.org/3/c-api/index.html> ([Python documentation](#))
 5. **Vadlamani A., Kalicheti R., Chimalakonda S.** *APIScanner — Towards automated detection of deprecated APIs in Python libraries.* — [arXiv preprint](#), 2021. — URL: <https://arxiv.org/abs/2102.09251> ([arXiv](#))
-

 Примечания к оформлению

- ✓ Позиции 1–2 — **примеры API-дизайна и API-реализаций с Python**.
 - ✓ Позиция 3 — фундаментальная библиотека SciPy (часто интегрируется с Python API инженерных приложений).
 - ✓ Позиция 4 — официальная документация Python/C API (нормативный источник для описания API).
 - ✓ Позиция 5 — исследование актуальных проблем API в Python-сообществе (поддержка актуальности).
-

Если необходимо, могу дополнить список линеаризацией по разделам статьи, включить другие источники с **официальным DOI**, а также сформировать **примеры цитирования в тексте** по ГОСТ.

Интересное наблюдение: при доп.запросе в том же чате, что и основной запрос, он его воспринимает не как прямое указание к предыдущему ответу, а начинает заново искать источники, но уже оформлять их по ГОСТу.

Статьи для анализа ChatGPT:

Для дальнейшего анализа в п.4 выберем статьи:

- Na Xie (2024).
Strategic approaches to API design and management.
Applied and Computational Engineering 64(1):230–236.
DOI:10.54254/2755-2721/64/20241395. ([ResearchGate](#))
- *Pressio: Enabling projection-based model reduction for large-scale nonlinear dynamical systems.*
arXiv:2003.07798. ([arXiv](#))
- Ashwin V. Mohanan et al. (2018).
FluidSim: modular, object-oriented Python package for high-performance CFD simulations.
arXiv:1807.01769. ([arXiv](#))

4. Сравнительный анализ результатов

Таблица 1: Оценка ответов ИИ-моделей

Статья	Существование	Оформление	Полезность	Соответствие	Новизна (Год публикации)	Индекс цитируемости	Тип источника	Средняя оценка
Промпт 1 - Р1								
DeepSeek - Итого = 4,28 + 5,57 + 1,71=11,56								
<i>Designing Pythonic library APIs</i> , Ben Hoyt	10	0	5	5	10	0	0 ¹	4,28
<i>C/C++ из Python (C API)</i> , Habr	10	0	10	10	9	0	0 ²	5,57
<i>Инженерное ПО и научные вычисления</i>	10	0	1	1	0	0	0	1,71
ChatGPT - Итого = 7 + 7 + 6,71 = 20,71								
<i>ogs5py: A Python-API for the OpenGeoSys 5 Scientific Modeling Project</i>	9 ³	0	7	7	9	9	8 ⁴	7
<i>THE ART OF API DESIGN: BEST PRACTICES FOR MODERN SOFTWARE DEVELOPMENT</i>	9 ⁵	0	9	9	10	4	8	7
<i>Efficient and Accurate Automatic Python Bindings with cppyy & Cling</i>	10	0	9	9	10	1	8	6,71

¹ Статья хорошая для простого ознакомления, но не для использования её как источника литературы в научной статье.

² Статью на Habr хорошо использовать для практической части работы, особенно если с ней есть затруднения, но не использовать в качестве источника в списке литературы в научной статье

³ Доступ только по VPN

⁴ В соответствии с разбалловкой в задании, за статью максимум - 8

⁵ Первая ссылка (перед скобками) ведёт на другой сайт и не на конкретную статью, а на номер журнала

Промпт 2 - Р2

Статья	Существование	Оформление	Полезность	Соответствие	Новизна (Год публикации)	Индекс цитируемости	Тип источника	Средняя оценка
DeepSeek - Итого = 0 + 4,57 + 5,42 = 9,99								
<i>O проектировании API для инженерного ПО: практики и примеры из индустрии CAE</i>	0	0	0	0	0	0	0	0 ⁶
<i>Python for Scientific Computing and Engineering Applications</i>	5 ⁷	0	5	5	9	2	6 ⁸	4,57
<i>API Design Patterns</i>	9 ⁹	0	9	9	9	1	9	5,42
ChatGPT - Итого = 6,71 + 6,29 + 5,71 = 18,71								
<i>Н. И. Левонович — разработка многопоточного сервиса с Python и C/C++ (2024)</i>	10	0	9	9	10	1	8	6,71
<i>Svar: A Tiny C++ Header Brings Unified Interface for Multiple Programming Languages — Yong Zhao et al. (2021)</i>	10	0	8	8	9	1	8	6,29
<i>Bridging the Language Gap: An Empirical Study of Bindings for Open Source Machine Learning Libraries — Hao Li & Cor-Paul Bezemer (2022) - Bridging the Language Gap — empirical study of bindings (arXiv)</i>	10	0	5	5	10	2	8	5,71

⁶ Не удалось найти статью (из первого ответа даже недостаточно информации, а из дополнительного есть предположительная статья, но недействительная тоже)

⁷ Не совпадает название из таблицы ответа с тем, что по ссылке

⁸ В соответствии с низшим баллом за источник не являющейся статьёй

⁹ Доступно только по VPN

ПромпТ 3 - Р3								
Статья	Существование	Оформление	Полезность	Соответствие	Новизна (Год публикации)	Индекс цитируемости	Тип источника	Средняя оценка
DeepSeek - Итого = 6,14 + 6,29 = 12,43								
<i>Ban Rossum G. Python 3: Reference Manual</i>	7	0	9	9	7 ¹⁰	4	7	6,14
<i>Бизли Д.М. Python Essential Reference</i>	7	0	9	9	7 ¹¹	4	8	6,29
<i>Рудаков К.В. Инструменты и подходы к созданию программных интерфейсов для сложных вычислительных комплексов</i>	0	0	0	0	0	0	0	0 ¹²
ChatGPT - Итого = 20,14								
<i>Na Xie (2024). Strategic approaches to API design and management. Applied and Computational Engineering</i>	10	0	7	7	10	4	8	6,57
<i>Pressio: Enabling projection-based model reduction for large-scale nonlinear dynamical systems.</i>	10	0	8	8	9	7	8	7.14
<i>Ashwin V. Mohanan et al. (2018). FluidSim: modular, object-oriented Python package for high-performance CFD simulations. arXiv:1807.01769. (arXiv)</i>	10	0	8	8	9	2	8	6.43

¹⁰ Получилось найти только за 2009 год

¹¹ Аналогично предыдущей книге - найдено только за 2009

¹² Статью не удалось найти

5. Выводы

На основе анализа таблиц с оценками можно сделать следующие выводы:

Слабая сторона промптов: Если непосредственно в промпте не указывать требование по оформлению списка по ГОСТу, то модель выдаёт списки так как она считает нужным (DeepSeek - часто в виде таблицы, ChatGPT - группируя по контексту работ).

Сравнительная эффективность моделей Chat GPT и DeepSeek:

По совокупности результатов по всем промптам, ChatGPT демонстрирует значительное преимущество в эффективности поиска и предложения научных источников.

- Общий итоговый балл:
 - ChatGPT: $20.71 \text{ (P1)} + 18.71 \text{ (P2)} + 20.14 \text{ (P3)} = 59.56$
 - DeepSeek: $11.56 \text{ (P1)} + 9.99 \text{ (P2)} + 12.43 \text{ (P3)} = 33.98$

Разрыв почти двукратный. Это преимущество ChatGPT сохраняется на всех уровнях сложности запроса (P1-P3).

Ключевые различия в эффективности:

1. Надежность и существование источников: ChatGPT стабильно предлагает ссылки на реально существующие и доступные публикации (оценка "Существование" в основном 9-10). DeepSeek же допускает критические ошибки, предлагая несуществующие (битые) ссылки или источники без ссылок (например, "О проектировании API..." с 0 баллов, "Рудаков К.В..." с 0 баллов). Это самый серьезный недостаток DeepSeek в данном контексте.
2. Полезность и релевантность: ChatGPT лучше понимает контекст запроса и подбирает источники, напрямую связанные с проектированием API для научных и инженерных вычислений (высокие баллы по критерию "Полезность"). DeepSeek чаще дает слишком общие или лишь косвенно связанные источники (например, "C/C++ из Python (C API), Habr" для P1 или справочники по Python для P3).
3. Новизна: Обе модели в основном предлагают современные источники. Однако ChatGPT чаще находит актуальные статьи последних лет (2021, 2022, 2024), в то время как DeepSeek в P3 предлагает справочники с более ранними датами (оценка новизны 7 против 10 у ChatGPT).

Вывод: Для задачи поиска академических источников по конкретной технической тематике ChatGPT показал себя как более надежный и точный инструмент. DeepSeek в текущей реализации склонен к "галлюцинациям" (придумыванию несуществующих источников) и менее точен в подборе релевантных материалов.

Влияние сложности промпта на качество результатов:

Детализация и структурирование промпта (Р3) являются критически важными для получения качественного результата от обеих моделей, особенно для ChatGPT. Однако есть пределы влияния промпта-инжиниринга: он не может компенсировать фундаментальные слабости модели (как "галлюцинации" у DeepSeek) и не гарантирует выполнения очень специфических формальных задач (оформление по ГОСТ), если они явно не заложены в обучение. Промежуточный, менее структурированный, но перегруженный требованиями промпт (Р2) может давать худший результат, чем простой (Р1) или очень детальный (Р3).

Слабые места моделей:

DeepSeek:

- "Галлюцинации" и низкая проверяемость: Склонность предлагать несуществующие или непроверяемые ссылки (0 баллов за "Существование") делает его результаты ненадежными и требует чрезмерной ручной проверки.
- Более низкое понимание контекста: Чаще предлагает общие или менее релевантные источники по сравнению с ChatGPT.

ChatGPT:

- Неидеальная работа с промежуточной сложностью: Снижение эффективности на Р2 (хотя и меньшее, чем у конкурента) указывает на возможную чувствительность к формулировкам, которые не являются ни слишком простыми, ни максимально детальными.

Общий вывод:

В ходе сравнения моделей ChatGPT и DeepSeek для задачи подбора научной литературы было выявлено, что ChatGPT является более эффективным инструментом. Эта модель демонстрирует более высокую способность находить релевантные и разнообразные источники (в плане самих научных статей), предоставляя действительные ссылки на научные, а не публицистические материалы (такие как статьи на платформе Habr).

Примечание: При выполнении лабораторной возникли проблемы с определением индекса цитируемости.