

**Федеральное государственное автономное образовательное
учреждение высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»**

УТВЕРЖДЕНО
Директор

А.В. Малеев

	Рабочая программа дисциплины (модуля)
по дисциплине:	Программирование на Python
по направлению:	Программная инженерия
профиль подготовки:	Разработка программно-информационных систем высшая школа программной инженерии высшая школа программной инженерии
курс:	1
квалификация:	бакалавр

Семестр, формы промежуточной аттестации: 2 (весенний) - Дифференцированный зачет

Аудиторных часов: 60 всего, в том числе:

лекции: 30 час.

семинары: 30 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 48 час.

Всего часов: 108, всего зач. ед.: 3

Программу составил: А.В. Созыкин, канд. техн. наук

Программа обсуждена на заседании высшей школы программной инженерии 19.03.2025

Аннотация

Данный курс разработан для углубленного изучения Python, в том числе его применения в анализе данных, разработке web-приложений.

1. Цели и задачи

Цель дисциплины

- овладение практическими навыками использования языка программирования Python для подготовки и анализа данных, решения аналитических и статистических задач, веб-разработки.

Задачи дисциплины

- освоение языка программирования и платформы Python;
- изучение системы контроля версий Git;
- изучение особенностей объектно-ориентированного программирования в Python;
- развитие навыков создания функциональных веб-приложений;
- развитие навыков применения языка программирования Python для работы с базами данных.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Формулирует совокупность взаимосвязанных задач в рамках поставленной цели работы, обеспечивающих ее достижение. Определяет ожидаемые результаты решения поставленных задач
	УК-2.2 Проектирует решение конкретной задачи проекта, выбирая оптимальный способ ее решения, исходя из действующих правовых норм и имеющихся ресурсов и ограничений
ОПК-2 Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности	ОПК-2.1 Способен использовать информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности
ПК-2 Способен формализовать и алгоритмизировать поставленную задачу	ПК-2.1 Способен формализовать и алгоритмизировать поставленную задачу
	ПК-2.2 Владеет методами и приемами формализации и алгоритмизации задач

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны знать:

- основы парсинга и работы с API;
- основы визуализации данных в Python;
- систему контроля версий Git и особенности работы с ней;
- особенности командной работы в Git и GitHub;
- принципы объектно-ориентированного программирования;
- особенности работы с базами данных из Python;
- модули, пакеты, импорты в Python;
- возможности веб-скрапинга;
- веб-фреймворк Django;
- техники использования Python в веб-разработке.

уметь:

- использовать регулярные выражения;
- решать задачи классификации и кластеризации;
- работать с локальным репозиторием в Git;
- работать с удаленным репозиторием через GitHub;
- работать с файловой системой;
- работать с внешним API;
- использовать Select-запросы;
- применять регулярные выражения;
- работать с ORM;
- использовать Python в веб-разработке.

владеть:

- навыками работы с данными;
- навыками командной работы в Git и GitHub;
- навыками работы с разными форматами данных;
- навыками работы с классами;
- способами работы с базами данных из Python;
- навыками применения итераторов, генераторов, декораторов;
- приемами тестирования Django-приложений с использованием Pytest.

4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Создание web-приложений в Python	6	6		9
2	Python для анализа данных	6	6		9
3	Статистика в Python	6	6		10
4	Git - система контроля версий	6	6		10
5	Django: создание функциональных веб-приложений	6	6		10
Итого часов		30	30		48
Подготовка к экзамену		0 час.			
Общая трудоёмкость		108 час., 3 зач.ед.			

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 2 (Весенний)

1. Создание web-приложений в Python

Углубленный Python. Многопоточное и асинхронное программирование.

2. Python для анализа данных

Работа с файловой системой и модули. Исключения и обработка ошибок. Понятие класса. Регулярные выражения и основы синтаксического разбора. Библиотеки. Библиотека numpy. Вычислительные задачи. Библиотека Pandas. Функции и работа с данными. Продвинутый pandas. Основы парсинга и работы с API.

3. Статистика в Python

Базовые понятия статистики. Основы визуализации данных в Python (библиотеки matplotlib и seaborn). Случайные события. Случайные величины. Корреляция и корреляционный анализ. Задачи классификации и кластеризации. Доверительные интервалы. Статистическая проверка гипотез для несвязанных выборок. Статистическая проверка гипотез для связанных выборок. A/B тесты и как их проводить. Кейс-стади.

4. Git - система контроля версий

Знакомство с системой контроля версий Git. Работа с локальным репозиторием в Git. Работа с удаленным репозиторием через GitHub. Командная работа в Git и GitHub.

5. Django: создание функциональных веб-приложений

Знакомство с Django. Подготовка и запуск проекта. Обработка запросов и шаблоны. Работа с ORM. Знакомство с API на примере Django REST framework. CRUD в DRF. Разделение доступа в DRF. Тестирование Django-приложений с использованием Pytest.

5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Учебная аудитория, оснащенная компьютерами для каждого студента.

6. Перечень рекомендуемой литературы

Основная литература

1. Изучаем Python: программирование игр, визуализация данных, веб-приложения, [руководство] / Э. Мэтиз. — Санкт-Петербург, Питер, 2020. — URL: <https://ibooks.ru/bookshelf/371712/reading> (дата обращения: 24.11.2020). - Полный текст (Режим доступа : из сети МФТИ / Удаленный доступ)

Рекомендуемая литература для самостоятельного изучения

1. Копырин, А. С. Программирование на Python: учебное пособие / А. С. Копырин, Т. Л. Салова. — Москва: ФЛИНТА, 2021. — 48 с. — ISBN 978-5-9765-4753-7. — Текст: электронный // Лань: электронно-библиотечная система. — URL: <https://e.lanbook.com/book>
2. Федоров, Д. Ю. Программирование на языке высокого уровня Python: учебное пособие для вузов / Д. Ю. Федоров. — 3-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2022. — 210 с. — (Высшее образование). — ISBN 978-5-534-14638-7. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/492920>
3. Чернышев, С. А. Основы программирования на Python: учебное пособие для вузов / С. А. Чернышев. — Москва: Издательство Юрайт, 2022. — 286 с. — (Высшее образование). — ISBN 978-5-534-14350-8. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/496893>

Дополнительная литература

Рекомендуемая литература для самостоятельного изучения

1. Гниденко, И. Г. Технологии и методы программирования: учебное пособие для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — Москва: Издательство Юрайт, 2022. — 235 с. — (Высшее образование). — ISBN 978-5-534-02816-4. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/489920>
2. Илюшечкин, В. М. Основы использования и проектирования баз данных: учебник для вузов / В. М. Илюшечкин. — Москва: Издательство Юрайт, 2022. — 213 с. — (Высшее образование). — ISBN 978-5-534-03617-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/488604>

7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

-

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)

Интерпретатор Python, среды разработки, библиотеки для анализа данных и машинного обучения, системы контроля версий, инструменты для тестирования, виртуальные окружения, облачные сервисы.

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное освоение курса требует напряжённой самостоятельной работы студента. В программе курса приведено минимально необходимое время для работы студента над темой.

Самостоятельная работа включает в себя:

- проработку учебного материала;
- подготовку к практическим занятиям, выполнение домашних заданий.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению:	Программная инженерия
профиль подготовки:	Разработка программно-информационных систем высшая школа программной инженерии высшая школа программной инженерии
курс:	<u>1</u>
квалификация:	бакалавр

Семестр, формы промежуточной аттестации: 2 (весенний) - Дифференцированный зачет

Разработчик: А.В. Созыкин, канд. техн. наук

1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Формулирует совокупность взаимосвязанных задач в рамках поставленной цели работы, обеспечивающих ее достижение. Определяет ожидаемые результаты решения поставленных задач
	УК-2.2 Проектирует решение конкретной задачи проекта, выбирая оптимальный способ ее решения, исходя из действующих правовых норм и имеющихся ресурсов и ограничений
ОПК-2 Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности	ОПК-2.1 Способен использовать информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности
ПК-2 Способен формализовать и алгоритмизировать поставленную задачу	ПК-2.1 Способен формализовать и алгоритмизировать поставленную задачу
	ПК-2.2 Владеет методами и приемами формализации и алгоритмизации задач

2. Показатели оценивания компетенций

В результате изучения дисциплины «Программирование на Python» обучающийся должен:

знать:

- основы парсинга и работы с API;
- основы визуализации данных в Python;
- систему контроля версий Git и особенности работы с ней;
- особенности командной работы в Git и GitHub;
- принципы объектно-ориентированного программирования;
- особенности работы с базами данных из Python;
- модули, пакеты, импорты в Python;
- возможности веб-скрапинга;
- веб-фреймворк Django;
- техники использования Python в веб-разработке.

уметь:

- использовать регулярные выражения;
- решать задачи классификации и кластеризации;
- работать с локальным репозиторием в Git;
- работать с удаленным репозиторием через GitHub;
- работать с файловой системой;
- работать с внешним API;
- использовать Select-запросы;
- применять регулярные выражения;
- работать с ORM;
- использовать Python в веб-разработке.

владеть:

- навыками работы с данными;
- навыками командной работы в Git и GitHub;
- навыками работы с разными форматами данных;
- навыками работы с классами;
- способами работы с базами данных из Python;
- навыками применения итераторов, генераторов, декораторов;
- приемами тестирования Django-приложений с использованием Pytest.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

Во время текущего контроля студент должен уметь ответить на следующие вопросы:

1. Работа с файловой системой и модули.
2. Исключения и обработка ошибок.
3. Понятие класса.
4. Регулярные выражения и основы синтаксического разбора.
5. Библиотеки.
6. Библиотека `numpy`.
7. Вычислительные задачи.
8. Библиотека `Pandas`.
9. Функции и работа с данными.
10. Продвинутой `pandas`.
11. Основы парсинга и работы с API.
12. Базовые понятия статистики.
13. Основы визуализации данных в Python (библиотеки `matplotlib` и `seaborn`).
14. Случайные события.
15. Случайные величины.
16. Корреляция и корреляционный анализ.
17. Задачи классификации и кластеризации.
18. Доверительные интервалы.
19. Статистическая проверка гипотез для несвязанных выборок.
20. Статистическая проверка гипотез для связанных выборок.
21. А/В тесты и как их проводить.
22. Кейс-стади.
23. Работа с локальным репозиторием в `Git`.
24. Работа с удаленным репозиторием через `GitHub`.
25. Командная работа в `Git` и `GitHub`.
26. Объекты и классы.
27. Взаимодействие между ними.
28. Наследование, инкапсуляция и полиморфизм.
29. Открытие и чтение файла.
30. Запись в файл.
31. Работа с разными форматами данных.
32. Работа с библиотекой `requests`, `http`-запросы.
33. Работа с классами на примере API `VK`.
34. `Select`-запросы, выборки из одной таблицы.
35. Продвинутой выборка данных.
36. Работа с `PostgreSQL` из Python.
37. Python и базы данных. ORM.
38. Модули, пакеты, импорты в Python.
39. Регулярные выражения.
40. Веб-скрапинг.
41. Итераторы, генераторы.
42. Обработка запросов и шаблоны.
43. Работа с ORM.
44. API на примере `Django REST framework`.
45. Разделение доступа в `DRF`.
46. Тестирование `Django`-приложений с использованием `Pytest`.
47. Установка пакетов в `Ubuntu`. Маршрутизация.
48. Связывание хостинга файлов и запуска веб-приложения.
49. Использование дополнительных файлов для работы веб-приложения.

Во время занятий могут проходить интерактивные обсуждения в чатах курса, что будет являться домашним заданием. Успешное выполнение всех заданий по курсу и выполнение контрольных срезов знаний дает преимущество на экзамене.

4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

Примеры итоговых заданий:

Задание 1.

В папке лежит некоторое количество файлов. Количество и имена заранее известны.

Необходимо объединить их в один по следующим правилам:

Содержимое исходных файлов в результирующем файле должно быть отсортировано по количеству строк в них (то есть первым нужно записать файл с наименьшим количеством строк, а последним - с наибольшим).

Содержимое файла должно предваряться служебной информацией на 2-х строках: имя файла и количество строк в нем.

Задание 2.

У Яндекс.Диска есть очень удобное и простое API. Для описания всех его методов существует Полигон. Нужно написать программу, которая принимает на вход путь до файла на компьютере и сохраняет на Яндекс.Диск с таким же именем.

Все ответы приходят в формате json;

Загрузка файла по ссылке происходит с помощью метода put и передачи туда данных;

Токен можно получить, кликнув на полигоне на кнопку "Получить OAuth-токен".

Задание 3.

Доработать класс Flat Iterator в заданном коде. Должен получиться итератор, который принимает список списков и возвращает их плоское представление, т. е. последовательность, состоящую из вложенных элементов. Функция test в коде также должна отработать без ошибок.

Доработать функцию Flat_generator. Должен получиться генератор, который принимает список списков и возвращает их плоское представление. Функция test в коде также должна отработать без ошибок.

Написать итератор, обрабатывающий списки с любым уровнем вложенности.

Написать генератор, обрабатывающий списки с любым уровнем вложенности.

Задание 4.

Сделать конфигурацию docker-compose Вашего проекта.

Результатом должен быть docker-compose.yml файл с описанием конфигурации для развертывания приложения.

Для создания конфигурации необходим образ своего проекта, а значит предварительно необходимо описать Dockerfile, сделать образ и потом уже писать docker-compose.yml .

Конфигурация должна состоять из 3-х контейнеров: backend, postgres, nginx.

Контейнеры объединяются в сеть, которая работает в связке:

Nginx работает в качестве проху-http для пересылки динамических запросов к Django или возвращая статические html файлы.

PostgreSQL запускается до Django.

Django запускается через Gunicorn.

.

Критерии оценивания

отлично

10 всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений;

9 систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, правильное обоснование принятых решений;

8 глубокие знания учебной программы дисциплины и умение применять их на практике при решении конкретных задач, правильное обоснование принятых решений;

хорошо

7 твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;

6 знает материал, грамотно излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;

5 знает основной материал, грамотно излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач неточности;

удовлетворительно

4 фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;

3 характер знаний достаточен для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;

неудовлетворительно

2 не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных понятий дисциплины и не умеет правильно использовать полученные знания при решении типовых практических задач.

1 не знает формулировок основных понятий дисциплины и не умеет использовать полученные знания при решении типовых практических задач.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Дифференцированный зачет по дисциплине проводится в форме выполнения итогового задания.