

Программирование

Лекторы:

Григорий Французов

Ассистент:

Дарья Радзивил

**Язык:**

Русский

Трудоемкость:

6 з.е.

Форма контроля:

Дифф. зачет в 1 семестре

Экзамен во 2 семестре

Образовательная программа:

Теоретическая и экспериментальная физика

1, 2 семестры

Беспроводные технологии

1,2 семестры

Лекции (ак.час)*	Практические занятия (ак.час)	Лабораторные занятия (ак.час)
64	64	
*1 академический час = 45 минутам		

В курсе программирования мы познакомимся с современным языком программирования Python, научимся решать как алгоритмические задачи, так и задачи физического моделирования, линейной алгебры и математического анализа. Изучим основные модули (библиотеки) языка, трюковые структуры данных.

Содержание курса

1 семестр

Программирование

Структура курса

Разделы	Лекции (ак.ч.)	Лабы (ак.ч.)
1. Вводная: История и становление компьютеров	2	2
<ol style="list-style-type: none">1. Правила курса, рекомендуемые материалы.2. Давид Гильберт, унификация математики. Курт Гедель, теорема о неполноте.3. Машина Тьюринга, Лямбда Исчисление.4. Декларативные (Функциональные) и Императивные языки.5. Как построить исполнительное устройство, Архитектуры компьютеров: "код это данные" архитектура Фон Неймана "код отдельно от данных" Гарвардская архитектура6. История компьютеров.7. Устройство и физические ограничения современных компьютеров.		
2. Данные и программирование: Переменные	2	2
<ol style="list-style-type: none">1. Программирование - преобразование данных из одной форму в другую.2. Откуда они берутся, как и куда их можно передать, как их хранить.3. Язык программирования Python, почему именно он.4. Переменные и типы: числа, строки, коллекции5. Хранение простых переменных в памяти.		
3. Данные и программирование: Циклы и ветвления	2	2
<ol style="list-style-type: none">1. Булева алгебра2. Ветвления в Python, нелинейные программы, оператор If3. Сложные и комбинированные условия4. Циклы (For и While)5. Вычислительная сложность и циклы		
4. Данные и программирование: Функции и рекурсия	2	2
<ol style="list-style-type: none">1. Что такое функции (абстракции), разбиение на функции2. Аргументы функций3. Возврат значений4. Вызов функции из функции5. Рекурсию6. Передача функции в функцию (коллбэк, first class citizen) map, reduce7. Стек вызовов, хранение функций в памяти		
5. Данные и программирование: Словари и множества	2	2
<ol style="list-style-type: none">1. Словари, хранение в памяти2. Расширение словаря3. Перебор элементов словаря4. Множества5. Операции с множествами		
6. Данные и программирование: Структурирование кода. Библиотеки и модули	2	2
<ol style="list-style-type: none">1. Пространство имен, модуль, пакет2. Менеджер пакетов Pip, репозиторий пакетов3. Управление зависимостями, циклические зависимости4. Sandboxing, Dll hell		
7. Процесс разработки ПО. Введение	2	2

<ol style="list-style-type: none"> 1. Общее представление рабочем процессе 2. Роли в команде разработки, варианты карьерного пути в ИТ 3. Управление и планирование в командах 4. Инструменты командной работы: <ul style="list-style-type: none"> - таск трекаеры - система контроля версий, - CI/CD - мессенджеры и система коммуникаций 5. Вопросы безопасности 6. Жизненный цикл ПО 		
8. Процесс разработки ПО. Аналитика и постановка задач	2	2
<ol style="list-style-type: none"> 1. Задача продуктовой аналитики 2. Продуктовый дизайн 3. Как построить диалог с заказчиком 4. Как проверять идеи и гипотезы 5. Customer Development 		
9. Процесс разработки ПО. Профессия программиста и тестировщика	2	2
<ol style="list-style-type: none"> 1. Уровни компетенций программиста 2. Не программистские задачи (код ревью, менторинг) 3. Отладка и дебаг (устранение неисправностей) 4. Зачем нужно тестирование 5. Виды тестирования, Пирамида тестирования 6. Автоматизация тестирования, Unit-тесты 		
10. Данные и программирование. Файлы	2	2
<ol style="list-style-type: none"> 1. Работа с файлами в Python 2. Подходы к хранению данных в файлах 3. Сериализация и десериализация 4. Распространенные форматы (CSV, JSON, XML) 5. Библиотеки для работы с этими форматами 		
11. Данные и программирование: Сети и сетевые протоколы	2	2
<ol style="list-style-type: none"> 1. Компьютерные сети и их история 2. Модели организации сетей. Модель OSI. TCP/IP, UDP 3. Работа с сетевыми запросами в Python 4. Интернет, DNS 5. Сетевые протоколы, HTTP 6. Работа с HTTP, библиотека requests 		
12. Данные и программирование: Устройство современного веб приложения	2	2
<ol style="list-style-type: none"> 1. Типы клиентов веб приложений 2. Браузеры и их работа 3. Подходы к проектированию веб приложений (MVC\MVVM) 4. Разделение обработки и представления (бэкенд и фронтенд) 5. Адаптивная верстка, mobile first 6. Хостинг и облака 		
13. Данные и программирование: API	2	2
<ol style="list-style-type: none"> 1. Введение в паттерны проектирования веб приложений 2. Подходы к проектированию API 3. RPC, REST, GraphQL 4. паттерн CQRS 5. Веб API в Python. Flask/FastAPI 		
14. Данные и программирование: Базы данных	2	2

<ol style="list-style-type: none"> 1. Зачем нужны базы данных 2. Типы баз данных 3. Целостность данных ACID, транзакции 4. CAP теорема 5. Популярные базы данных и их особенности 		
--	--	--

2 семестр

Программирование

Структура курса

Разделы	Лекции (ак.ч.)	Лабы (ак.ч.)
1. Алгоритмы и структуры данных: Асимптотическая сложность простых алгоритмов	2	2
<ol style="list-style-type: none"> 1. Базовые алгоритмы сортировки 2. Сложность операций над структурами данных 		
2. Алгоритмы и структуры данных: Алгоритмы поиска	2	2
<ol style="list-style-type: none"> 1. Двоичный поиск 2. Двоичная куча 3. Деревья поиска 		
3. Алгоритмы и структуры данных: Алгоритмы на графах	2	2
<ol style="list-style-type: none"> 1. NP трудные задачи 2. Методы оптимизации: <ul style="list-style-type: none"> • монте-карло • градиентный спуск • имитация отжига 		
4. Объектно-ориентированное программирование: Введение и основные понятия	2	2
<ol style="list-style-type: none"> 1. Что такое и зачем нужно ООП 2. История ООП 3. Принципы ООП 4. SOLID 		
5. Объектно-ориентированное программирование: Принципы ООП и особенности Python	2	2
<ol style="list-style-type: none"> 1. Особенности и проявление принципов ООП в Python: <ul style="list-style-type: none"> - объекты, классы, ЖЦ объекта - перегрузка методов - множественное наследование 2. ORM в Python 		
6. Объектно-ориентированное программирование: Другие языки и подходы	2	2
<ol style="list-style-type: none"> 1. Реализация в Smalltalk 2. Реализация в JS 		
7. Функциональное программирование	2	2
<ol style="list-style-type: none"> 1. Главное отличие от ООП (ФП лучше/удобнее выражает процесс) 2. Компиляторы, 3. Автоматизированные доказательства 4. (ФП как полигон для фиц других языков) 		
8. Современные операционные системы: Взгляд программиста	2	2

<ol style="list-style-type: none"> 1. Процесс загрузки 2. Прерывания 3. Системные вызовы 4. Вытесняющая многозадачность 5. Виртуальная память 6. Кольца защиты 		
9. Компиляторы и интерпретаторы	2	2
<ol style="list-style-type: none"> 1. Подходы к архитектуре компиляторов 2. Подходы к архитектуре интерпретаторов 		
10. Concurrency и параллелизм	2	2
<ol style="list-style-type: none"> 1. Что такое параллелизм 2. Отличие concurrency от параллелизма 		
11. Многопоточность и асинхронность: Теоретические основы	2	2
<ol style="list-style-type: none"> 1. Отличие многопоточности и асинхронности 2. Паттерны работы с многопоточными приложениями: <ul style="list-style-type: none"> - семафор - mutex - замыкание 		
12. Многопоточность и асинхронность: Python	2	2
<ol style="list-style-type: none"> 1. Библиотека Asyncio 2. Паттерн Async/await 		
13. Собеседования в IT компаниях	2	2
<ol style="list-style-type: none"> 1. Выбор компании 2. Процесс собеседования 3. Подготовка к собеседованиям 4. Переговоры с потенциальным работодателем 		
14. Обработка данных в Python	2	2
<ol style="list-style-type: none"> 1. Библиотека Pandas 2. Библиотека NumPy 		
15. Нейросети, прикладной взгляд	2	2
<ol style="list-style-type: none"> 1. Введение в нейросети, какие задачи решают 2. Базовые понятия 3. Примеры и библиотеки 		

Рекомендуемые ресурсы

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. — СПб.: Питер, 2015. — 1120 с.: ил. — (Серия «Классика computer science»). ISBN 978-5-496-01395-6
2. Лутц М. Изучаем Python, том 1, 5-е изд.: Пер. с англ. — СПб.: ООО «Диалектика», 2019. — 832 с. : ил. — Парад, тит. англ. ISBN 978-5-907144-52-1 (рус., том 1)
3. Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. Алгоритмы: построение и анализ, 3-е издание = Introduction to Algorithms, Third Edition. — М.: «Вильямс», 2013. — 1328 с. — ISBN 978-5-8459-1794-2.

Политика оценивания

Оценочные средства дисциплины: домашние задания, коллоквиум, дифф. зачет (1 семестр), экзамен (2 семестр).

В целом за семестр можно набрать 100 баллов:

- домашние задания - совокупно 60 баллов
- коллоквиум - максимум 10 баллов
- дифф. зачет/экзамен - максимум 30 баллов

Домашние задания оцениваются:

- Д/з № 1 - максимум 12 баллов,
- Д/з № 2 - максимум 15 баллов
- Д/з № 3 - максимум 15 баллов
- Д/з № 4 - максимум 18 баллов

Коллоквиум состоит из двух частей:

- 1) Теоретическо-практическая часть (тестовая форма).
- 2) Практическая часть, которая сдаётся с помощью средств git и GitHub.

Дифф. зачет/экзамен состоит из трех частей:

- 1) Теоретические вопросы. Несколько вопросов по курсу в виде диалога со студентом. Обязательная и основная часть проверки знаний.
- 2) Вопросы по домашним заданиям. Часть, которая отвечает за проверку осознанного и самостоятельного выполнения практических заданий во время семестра. Для некоторых студентов эта часть необязательна - при наличии хорошего отзыва от преподавателя практики.
- 3) Часть для тех, кто плохо ответил на предыдущие вопросы, но хочет хорошую отметку или повысить её. Включает в себя решение задач на языке Python и дополнительные вопросы.

Общая итоговая оценка формируется исходя из количества баллов: от 90 до 100 - «отлично», от 74 до 90 - «хорошо», от 60 до 74 - «удовлетворительно».