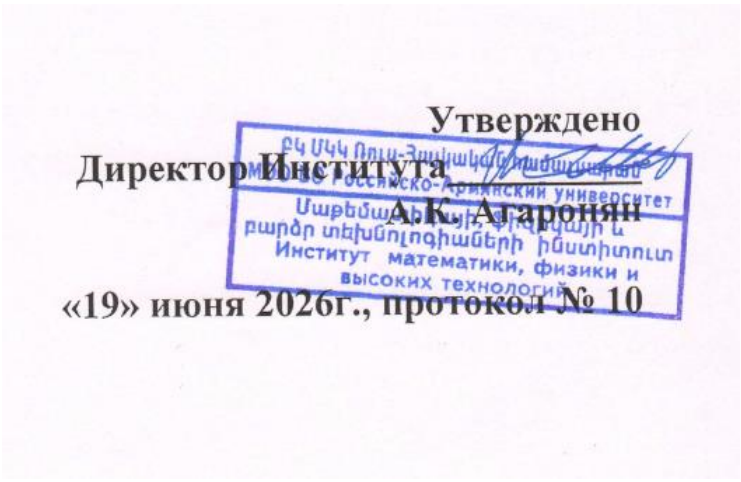
Межгосударственная образовательная организация
высшего образования
«Российско-Армянский университет»



УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ДИСЦИПЛИНЫ

Наименование дисциплины: ***Б1. В.09 Программирование в физике***

Автор (ы): ***старший преподаватель, к.ф.-м.н. Э.А. Газазян***

Направление подготовки: 11.03.04 Электроника и наноэлектроника
Наименование образовательной программы: Квантовая информатика

Согласовано:

Заведующий Кафедрой общей физики и квантовых наноструктур

Айрапетян Д.Б.



(подпись)

1. АННОТАЦИЯ

1.1. Краткое описание содержания данной дисциплины;

Программирование в физике играет важную роль в современной науке. Оно почти так же важно, как экспериментальные и теоретические методы. Поэтому будущие ученые, инженеры и преподаватели должны уметь использовать компьютерное моделирование для исследования различных физических явлений и процессов.

В Python можно использовать численные методы для решения уравнений, задач линейной алгебры, интерполяции и визуализации двумерных и трёхмерных данных.

1.2. Трудоемкость в академических кредитах и часах, формы итогового контроля (экзамен);

8 з.е. (288ч.), экзамен

1.3. Взаимосвязь дисциплины с другими дисциплинами учебного плана специальности (направления)

Численные методы в физике, Квантовая информатика, Квантовое программирование (QISKET), Структуры данных и алгоритмы (Python), Функциональное программирование (Python & Wolfram).

1.4. Результаты освоения программы дисциплины:

Код компетенции *(в соответствии рабочим с учебным планом)*	**Наименование компетенции** *(в соответствии рабочим с учебным планом)*	**Код индикатора достижения компетенций** *(в соответствии рабочим с учебным планом)*	**Наименование индикатора достижений компетенций***(в соответствии рабочим с учебным планом)*
ПК-2	Способен разрабатывать эффективные алгоритмы решения сформулированных задач с использованием современных языков программирования и обеспечивать их программную реализацию	ПК-2.1 ПК-2.2 ПК-2.3	Знает методы разработки эффективных алгоритмов решения научно-исследовательских задач Умеет использовать алгоритмы решения исследовательских задач с использованием современных языков программирования Владеет навыками разработки стратегии и методологии исследования изделий микро- и наноэлектроники

2. УЧЕБНАЯ ПРОГРАММА

2.1. Цели и задачи дисциплины

Основная цель изучаемой дисциплины — ознакомление студентов с основами программирования в физике, и ее применению в решении физических задач.

2.2. Трудоемкость дисциплины и виды учебной работы (в академических часах и зачетных единицах) ***(удалить строки, которые не будут применены в рамках дисциплины)***

Виды учебной работы	**Всего, в акад. часах**	**Распределение по семестрам**
		1 сем
1	**2**	**3**
1. Общая трудоемкость изучения дисциплины по семестрам, в т. ч.:	**288**	**288**
1.1. Аудиторные занятия, в т. ч.:	**96**	**96**
1.1.1. Лекции	**32**	**32**
1.1.2. Практические занятия, в т. ч.	**64**	**64**
1.1.2.1. Контрольные работы		
1.1.2.2. Другое (указать)		
1.2. Самостоятельная работа	**138**	**138**
1.3. Консультации		
Итоговый контроль (Экзамен, Зачет, диф. зачет - указать)	**Экзамен 54**	**Экзамен 54**

2.3. Содержание дисциплины

2.3.1. Тематический план и трудоемкость аудиторных занятий (модули, разделы дисциплины и виды занятий) по рабочему учебному плану

Разделы и темы дисциплины	**Всего (ак. часов)**	**Лекции(ак. часов)**	**Практ. Занятия (ак. часов)**
1	**2=3+4+5+6+7**	**3**	**4**
Тема 1. Python	***21***	***4***	***14***
Тема 2. Модули и пакеты	***11***	***4***	***7***
Тема 3. Символьный расчёт	***9***	***4***	***5***
Тема 4. Численный расчёт	***15***	***4***	***10***

Тема 5. Пакеты визуализации	***13***	***4***	***8***
Тема 6.Основы работы с массивами	***7***	***2***	***4***
Тема 7. Статистика	***5***	***2***	***3***
Тема 8. Регрессия	***5***	***2***	***3***
Тема 9. Интерполяция	***11***	***4***	***7***
Тема 10. Основы линейной алгебры	***5***	***2***	***3***
Итого	***96***	***32***	***64***

2.3.2. Краткое содержание разделов дисциплины в виде тематического плана

Тема 1. Python
Python - это высокоуровневый язык программирования, известный своей простотой и широкими возможностями. В этой теме студенты познакомятся с основами Python, включая его синтаксис, базовые конструкции и операции с данными. Мы рассмотрим, как писать и запускать программы, работать с переменными, условиями, циклами и функциями.

Тема 2. Модули и пакеты
Python поддерживает модульную структуру, что позволяет организовывать код в более управляемые и повторно используемые блоки. В этой теме студенты изучат, что такое модули и пакеты, как их создавать и использовать. Мы научимся импортировать стандартные библиотеки Python и устанавливать внешние пакеты с помощью менеджера пакетов pip.

Тема 3. Символьный расчёт
Символьный расчёт позволяет работать с математическими выражениями в аналитической форме. В этой теме студенты познакомятся с библиотекой SymPy, которая предоставляет инструменты для символьной математики. Мы научимся упрощать выражения, дифференцировать и интегрировать функции, решать уравнения и системы уравнений.

Тема 4. Численный расчёт
Численные методы играют важную роль в физике для решения задач, которые не поддаются аналитическим методам. В этой теме студенты изучат основные численные алгоритмы, такие как решение уравнений и задачи линейной алгебры. Мы будем использовать библиотеку NumPy для эффективных вычислений и применения численных методов к различным физическим задачам.

Тема 5. Пакеты визуализации
Визуализация данных позволяет наглядно представлять результаты расчетов и анализировать их. В этой теме студенты познакомятся с основными инструментами для визуализации данных в Python, таким как Matplotlib . Мы научимся строить графики, создавать двумерные и трёхмерные визуализации и оформлять их для публикации и презентаций.

Тема 6. Основы работы с массивами данных
Работа с большими объемами данных является важной частью многих физических исследований. В этой теме студенты изучат основы работы с массивами данных с помощью библиотеки NumPy. Мы научимся создавать, изменять и обрабатывать массивы данных, а также использовать их для эффективных вычислений.

Тема 7. Статистика
Статистика является важным инструментом для анализа и интерпретации данных в физике. В этой теме студенты рассмотрят основные статистические методы, включая вычисление среднего значения, медианы, стандартного отклонения и анализ распределения данных. Мы также изучим нормальное распределение и его применение в физике.

Тема 8. Регрессия
Регрессия используется для моделирования и анализа отношений между переменными. В этой теме студенты изучат различные виды регрессий, включая линейную, полиномиальную и множественную регрессию. Мы научимся строить регрессионные модели, оценивать их точность и применять их для прогнозирования и интерпретации физических данных.

2.3.3. Краткое содержание практических занятий

Тема 1. Python На практических занятиях по этой теме студенты научатся:

- Устанавливать и настраивать среду разработки для работы с Python.
- Писать простые программы, используя основные конструкции языка: переменные, условия, циклы.
- Использовать функции и создавать собственные функции для решения задач.

Тема 2. Модули и пакеты Практические занятия будут включать:

- Создание и использование модулей в Python.
- Импортирование стандартных библиотек Python.
- Установка и использование внешних пакетов с помощью менеджера пакетов pip.
- Организация кода в пакеты для повышения его структурированности и повторного использования.

Тема 3. Символьный расчёт В рамках этой темы студенты будут:

- Изучать библиотеку SymPy и её основные функции.
- Решать уравнения и системы уравнений с помощью символьных методов.
- Выполнять дифференцирование и интегрирование функций.
- Упрощать сложные математические выражения.

Тема 4. Численный расчёт Практические занятия будут направлены на:

- Изучение основных численных методов с использованием библиотеки NumPy.
- Решение задач линейной алгебры, таких как нахождение обратных матриц и решение систем линейных уравнений.
- Применение численных методов для решения различных физических задач.

Тема 5. Пакеты визуализации На этих занятиях студенты научатся:

- Использовать библиотеку Matplotlib для создания различных типов графиков.
- Строить двумерные и трёхмерные визуализации данных.
- Оформлять графики для публикаций и презентаций.

Тема 6. Основы работы с массивами данных Практические занятия будут включать:

- Работа с массивами данных с помощью библиотеки NumPy.
- Создание, изменение и обработка массивов данных.
- Использование методов NumPy для выполнения эффективных вычислений.
- Применение работы с массивами данных для решения физических задач.

Тема 7. Статистика На практических занятиях по статистике студенты будут:

- Вычислять основные статистические параметры, такие как среднее значение, медиана и стандартное отклонение.
- Анализировать распределение данных с использованием различных статистических методов.
- Изучать нормальное распределение и его применение к физическим данным.
- Применять статистические методы для анализа экспериментальных данных.

Тема 8. Регрессия Практические занятия по регрессии будут включать:

- Построение линейных регрессионных моделей для анализа зависимостей между переменными.
- Изучение полиномиальных и множественных регрессий.
- Оценка точности регрессионных моделей и их применение для прогнозирования.
- Применение регрессионных методов для анализа и интерпретации физических данных.

2.3.4. Материально-техническое обеспечение дисциплины

Для успешного освоения дисциплины "Программирование в физике" необходимо соответствующее материально-техническое обеспечение. Это включает в себя следующие компоненты:

1. **Компьютерные классы:**

- Оснащенные современными персональными компьютерами или ноутбуками с установленной операционной системой (Windows или Linux).
- Доступ к сети интернет для скачивания необходимых библиотек и инструментов.
- Среда разработки (IDE) для программирования на Python, такие как PyCharm, Visual Studio Code или Jupyter Notebook.

2. Программное обеспечение:

- Установленный интерпретатор Python последней версии (рекомендуется Python 3.x).

- Библиотеки для численных вычислений и символьных расчётов (NumPy, SymPy, SciPy).
- Пакеты для визуализации данных (Matplotlib).
- Менеджер пакетов pip для установки и управления библиотеками.

3. Учебные материалы:
- Учебные пособия и методические указания по каждой теме курса.
- Доступ к онлайн-ресурсам и документации по Python и его библиотекам.
- Практические задания и проекты для самостоятельной работы.

4. **Дополнительные ресурсы:**
- Электронная библиотека с доступом к книгам по программированию и физике.

Обеспечение всеми вышеперечисленными компонентами является необходимым для полного и эффективного усвоения материала дисциплины "Программирование в физике".

2.4. Модульная структура дисциплины с распределением весов по формам контролей

Формы контролей	**Вес формы (форм) текущего контроля в результирующей оценке текущего контроля (по модулям)**		**Вес формы промежуточного контроля в итоговой оценке промежуточного контроля**		**Вес итоговой оценки промежуточного контроля в результирующей оценке промежуточных контролей**		**Вес итоговой оценки промежуточного контроля в результирующей оценке промежуточных контролей (семестровой оценке)**	**Веса результирующей оценки промежуточных контролей и оценки итогового контроля в результирующей оценке итогового контроля**
Вид учебной работы/контроля	**М1** [1]	**М2**	**М1**	**М2**	**М1**	**М2**		
Контрольная работа *(при наличии)*			0.5	0.5				
Устный опрос *(при наличии)*								
Тест *(при наличии)*								
Лабораторные работы *(при наличии)*								
Письменные домашние задания *(при наличии)*	0.5	0.5						
Реферат *(при наличии)*								
Эссе *(при наличии)*								
Проект *(при наличии)*								
Другие формы (при наличии)	0.5	0.5						
Веса результирующих оценок текущих контролей в итоговых оценках промежуточных контролей					0.5	0.5		
Веса оценок промежуточных контролей в итоговых оценках промежуточных контролей								

[1] Учебный Модуль

Вес итоговой оценки 1-го промежуточного контроля в результирующей оценке промежуточных контролей							0.5	
Вес итоговой оценки 2-го промежуточного контроля в результирующей оценке промежуточных контролей							0.5	
Вес результирующей оценки промежуточных контролей в результирующей оценке итогового контроля								0.5
Вес итогового контроля (Экзамен/зачет) в результирующей оценке итогового контроля								0.5
	∑ = 1	**∑ = 1**	**∑ = 1**	**∑ = 1**	**∑ = 1**	**∑ = 1**	**∑ = 1**	**∑ = 1**

3. Теоретический блок *(указываются материалы, необходимые для освоения учебной программы дисциплины)*

3.1.1. Учебник(и)

Основные учебники для дисциплины "Программирование в физике" включают следующие издания:

- "Python для физиков" автора Алексея Смирнова. В учебнике подробно изложены основы программирования на Python, а также его применение в решении физических задач. Рассмотрены основные методы численного и символьного расчета, а также работа с библиотеками.
- "Численные методы в физике" под редакцией Юрия Иванова. Книга охватывает различные численные методы и их реализацию на Python, что делает её незаменимым ресурсом для студентов физиков.

3.1.2. Учебное(ые) пособие(я)

Учебные пособия дополняют основной учебник и предоставляют дополнительные материалы для более глубокого изучения:

- "Практикум по программированию на Python" автора Михаила Кузнецова. Пособие включает множество практических задач и проектов, направленных на развитие навыков программирования и применение их к физическим проблемам.
- "Символьные вычисления в Python" под редакцией Анны Петровой. В этом пособии рассматриваются примеры использования библиотеки SymPy для решения задач физики.

3.1.3. Курс лекций

Курс лекций по дисциплине "Программирование в физике" представляет собой серию лекций, прочитанных преподавателем на протяжении семестра. Лекции охватывают все ключевые темы курса, такие как:

- Введение в Python и основные конструкции языка.
- Использование модулей и пакетов для организации кода.
- Применение символьных и численных методов для решения физических задач.
- Методы визуализации данных и работа с большими массивами данных. Лекции записаны и доступны в текстовом и видеоформате для удобства студентов.

3.1.4. Краткие конспекты лекций

Краткие конспекты лекций представляют собой сжатые версии полного курса лекций. Они включают основные положения и ключевые моменты каждой лекции, что позволяет студентам быстро повторить материал и подготовиться к экзаменам. Конспекты содержат:

- Основные определения и теоретические концепции.
- Важные формулы и алгоритмы.
- Примеры задач и их решения. Конспекты доступны в печатном и электронном формате.

3.1.5. Электронные материалы (электронные учебники, учебные пособия, курсы и краткие конспекты лекций, презентации PPT и т.п.)

Электронные материалы предоставляют студентам дополнительные ресурсы для изучения курса:

- **Электронные учебники и учебные пособия**: интерактивные книги и пособия, доступные через университетскую электронную библиотеку. Они включают мультимедийные элементы, такие как видеоуроки и интерактивные задания.
- **Курсы и краткие конспекты лекций**: онлайн-курсы на платформах, таких как Coursera и edX, которые позволяют студентам углубленно изучать темы, обсуждаемые на лекциях.
- **Презентации PPT**: все лекции сопровождаются презентациями PowerPoint, которые включают основные моменты лекций, графики и иллюстрации. Презентации доступны для скачивания и могут быть использованы для подготовки к занятиям и экзаменам.
- **Другие электронные ресурсы**: видеозаписи лекций, подкасты, форум для обсуждения вопросов и проблем, связанные с курсом, и прочие материалы, способствующие углубленному пониманию темы.

4. Фонды оценочных средств *(указываются материалы, необходимые для проверки уровня знаний в соответствии с содержанием учебной программы дисциплины).*

4.1. Планы практических и семинарских занятий

Планы практических и семинарских занятий составлены таким образом, чтобы студенты могли закрепить теоретические знания, полученные на лекциях, и развить практические навыки. Каждое занятие включает в себя выполнение конкретных задач, работу над проектами и обсуждение результатов.

Примерные темы занятий:

- Введение в Python и установка необходимых инструментов.
- Основы синтаксиса Python: переменные, условия, циклы.
- Работа с функциями и модулями.
- Символьные вычисления с использованием SymPy.
- Численные методы и их реализация с помощью NumPy.
- Визуализация данных с использованием Matplotlib и Seaborn.
- Основы работы с массивами данных.
- Применение статистических методов в Python.
- Регрессионный анализ и построение моделей.

4.2. Планы лабораторных работ и практикумов

Лабораторные работы и практикумы ориентированы на практическое применение знаний в решении физических задач. Студенты выполняют лабораторные работы, используя программирование для моделирования физических процессов и анализа данных.

Примерные темы лабораторных работ:

- Численное решение уравнений: метод Ньютона и бисекции.
- Работа с матрицами и системами линейных уравнений.
- Численное интегрирование: метод трапеций и метод Симпсона.
- Визуализация экспериментальных данных.
- Применение методов регрессии к физическим данным.
- Моделирование физических процессов с использованием программирования.

4.3. Материалы по практической части курса

- 4.3.1. Учебно-методические пособия Учебно-методические пособия включают подробные инструкции по выполнению практических и лабораторных работ, примеры задач и их решения, а также рекомендации по использованию программных инструментов.
- 4.3.2. Учебные справочники Учебные справочники содержат основные теоретические сведения, алгоритмы и методы, которые необходимы для выполнения практических и лабораторных работ. Они служат быстрым источником информации для студентов.
- 4.3.3. Задачники (практикумы) Задачники содержат разнообразные задачи и упражнения, направленные на закрепление изученного материала. В них включены задачи разного уровня сложности, от простых до более сложных, требующих творческого подхода.
- 4.3.4. Наглядно-иллюстративные материалы Наглядные материалы включают в себя графики, схемы, диаграммы и другие визуальные элементы, которые помогают лучше понять и запомнить изучаемый материал. Они используются как в печатных, так и в электронных учебных пособиях.
- 4.3.5. Другие виды материалов к другим видам материалов относятся видеозаписи лекций и практических занятий, интерактивные симуляции, онлайн-курсы и форумы для обсуждения и обмена опытом между студентами и преподавателями.

4.4. **Вопросы и задания для самостоятельной работы студентов**

Самостоятельная работа студентов включает выполнение домашних заданий, проектов и подготовку к экзаменам. Примерные вопросы и задания:

- Написание программ для решения задач из курса.
- Выполнение проектов по моделированию физических процессов.

4.5. Тематика рефератов, эссе и других форм самостоятельных работ

Тематика рефератов и эссе включает:

- История развития программирования в физике.
- Применение Python в современных физических исследованиях.
- Сравнительный анализ различных численных методов.
- Примеры успешного использования символьных вычислений в физике.
- Анализ и интерпретация данных в физических экспериментах.

4.6. Образцы вариантов контрольных работ, тестов и/или других форм текущих и промежуточных контролей

Контрольные работы и тесты проводятся для оценки текущих знаний студентов. Образцы контрольных работ включают задачи на написание программ, анализ данных и применение численных методов. Примеры тестов содержат вопросы по теоретическим аспектам курса и задания на проверку практических навыков.

4.7. Перечень экзаменационных вопросов

Экзаменационные вопросы охватывают все основные темы курса и включают:

- Основные конструкции и синтаксис Python.
- Применение модулей и пакетов.
- Символьные и численные методы.
- Визуализация данных и работа с массивами.
- Статистические методы и регрессионный анализ.

Тема 1. Python

1. Как объявить и использовать функции в Python?
2. Опишите основные типы данных в Python.
3. Как выполнять арифметические операции в Python?
4. Что такое условные конструкции в Python? Приведите пример.
5. Как использовать циклы for и while в Python?
6. Что такое списки в Python? Как их создавать и использовать?
7. Как работать со строками в Python?
8. Объясните использование словарей в Python.
9. Как объявить и использовать кортежи в Python?
10. Что такое множества в Python и как с ними работать?
11. Как работают логические операторы в Python?
12. Как использовать операторы сравнения в Python?
13. Что такое импорт модулей в Python и как это делается?
14. Объясните работу с файлами в Python.
15. Как обработать исключения в Python?
16. Что такое list comprehension и как его использовать?
17. Как работать с функцией range() в Python?
18. Что такое lambda-функции в Python?
19. Как создать и использовать классы в Python?
20. Как написать условный оператор if в Python?
21. Как использовать циклы for и while в Python?

Тема 2. Модули и пакеты

1. Что такое модуль в Python?
2. Как создать свой модуль в Python?.
3. Как использовать стандартные библиотеки Python?
4. Что такое файл **init**.py и зачем он нужен?
5. Как установить внешние пакеты с помощью pip?
6. Как обновить и удалить пакеты, установленные через pip?
7. Что такое виртуальная среда (virtual environment) и как её создать?

8. Как активировать и деактивировать виртуальную среду?
9. Объясните, как использовать setuptools для создания пакетов.

Тема 3. Символьный расчёт

1. Что такое символьный расчёт?
2. Как установить и импортировать библиотеку SymPy?
3. Как создать символьные переменные в SymPy?
4. Как упрощать символьные выражения в SymPy?
5. Как решать уравнения и системы уравнений в SymPy?
6. Что такое серия Тейлора и как её разложить с помощью SymPy?
7. Как выполнять матричные операции в SymPy?
8. Как работать с многочленами в SymPy?
9. Как проводить символьные вычисления с комплексными числами?
10. Как использовать функции для создания символьных выражений в SymPy?
11. Как строить графики символьных функций в SymPy?

Тема 4. Численный расчёт

1. Что такое численный расчёт?
2. Как установить и импортировать библиотеку NumPy?
3. Как создавать массивы в NumPy?
4. Как выполнять основные операции с массивами в NumPy?
5. Как решать системы линейных уравнений с помощью NumPy?
6. Как находить определители и обратные матрицы с помощью NumPy?
7. Как решать нелинейные уравнения численными методами?
8. Как находить корни многочленов с помощью NumPy?
9. Как работать с собственными значениями и векторами в NumPy?
10. Как использовать линейную регрессию в NumPy?
11. Как решать задачи линейной алгебры с использованием NumPy?
12. Как проводить численный анализ данных с помощью NumPy?
13. Как использовать векторные и матричные операции для оптимизации вычислений?

Тема 5. Пакеты визуализации

1. Что такое библиотека Matplotlib и как её установить?
2. Как создавать простые графики с помощью Matplotlib?
3. Как добавлять заголовки и подписи к осям в графиках Matplotlib?
4. Как строить несколько графиков на одном холсте в Matplotlib?
5. Как изменять стиль и цвет графиков в Matplotlib?
6. Как строить гистограммы с использованием Matplotlib?
7. Как строить трехмерные графики с помощью Matplotlib?
8. Как создавать комбинированные графики с Matplotlib ?
9. Как сохранять графики в различных форматах (PNG, PDF и т.д.)?
10. Как создавать анимации с использованием Matplotlib?
11. Как добавлять легенды и аннотации на графики в Matplotlib?
12. Как визуализировать временные ряды с помощью Matplotlib?
13. Как использовать логарифмические шкалы в Matplotlib?
14. Как строить графики рассеивания (scatter plots) в Matplotlib и Seaborn?

15. Как настраивать оси и сетку на графиках Matplotlib?

Тема 6. Основы работы с массивами данных

1. Что такое массив данных в NumPy?
2. Как создавать одномерные и многомерные массивы в NumPy?
3. Как обращаться к элементам и срезам массивов в NumPy?
4. Как изменять размер и форму массивов в NumPy?
5. Как выполнять операции над массивами (сложение, умножение и т.д.)?
6. Как использовать булевы маски для работы с массивами?
7. Как сортировать массивы и искать элементы в NumPy?
8. Как использовать функции для работы с массивами (sum, mean, max, min)?
9. Как выполнять операции по осям в многомерных массивах?
10. Как работать с структурированными массивами (структуры данных)?
11. Как загружать и сохранять массивы данных из файлов?
12. Как использовать функции для генерации случайных чисел в NumPy?
13. Как выполнять трансформации данных с использованием массивов?
14. Как вычислять статистические параметры массивов?
15. Как использовать линейную алгебру для работы с массивами?
16. Как создавать и использовать маски для обработки данных?
17. Как выполнять операцию транпонирования массивов?
18. Как применять функции для фильтрации данных в массивах?
19. Как объединять и разъединять массивы данных?
20. Как использовать функции для агрегации данных в массивах?

Тема 7. Статистика

1. Что такое статистика и как она используется в физике?
2. Как вычислить среднее значение набора данных?
3. Как найти медиану и моду набора данных?
4. Как вычислить стандартное отклонение и дисперсию?
5. Что такое нормальное распределение и как его визуализировать?
6. Как проводить корреляционный анализ данных?
7. Как проводить регрессионный анализ данных?
8. Как проверять нормальность распределения данных?
9. Как вычислять коэффициенты автокорреляции?
10. Как применять статистические методы для анализа экспериментальных данных?

Тема 8. Регрессия

1. Что такое линейная регрессия?
2. Какова цель регрессионного анализа?
3. Чем отличается простая линейная регрессия от множественной линейной регрессии?
4. Какие методы используются для оценки параметров линейной регрессии?
5. Что такое метод наименьших квадратов?
6. Что такое полиномиальная регрессия?
7. Каковы преимущества использования регрессии в физике?
8. Какие библиотеки Python используются для регрессионного анализа?
9. Что такое среднеквадратическая ошибка (MSE)?
10. Что такое средняя абсолютная ошибка (MAE)?

11. Что такое t-тест для регрессионных коэффициентов?
12. Как интерпретировать p-значение в контексте регрессионного анализа?
13. Что такое логистическая регрессия?
14. Чем логистическая регрессия отличается от линейной регрессии?
15. В каких случаях используется логистическая регрессия?
16. Как интерпретировать коэффициенты в логистической регрессии?
17. Что такое сигмоидная функция?

Тема 9. Интерполяция

1. Что такое интерполяция?
2. В каких случаях используется интерполяция?
3. Чем интерполяция отличается от экстраполяции?
4. Какие основные методы интерполяции вы знаете?
5. Что такое линейная интерполяция?
6. Как вычисляется линейная интерполяция?
7. Какие ограничения имеет линейная интерполяция?
8. Что такое полиномиальная интерполяция?
9. Какую роль играют узлы интерполяции?
10. Что такое сплайны?
11. Как работает кубическая сплайн-интерполяция?
12. В каких случаях предпочтительно использовать кубические сплайны?
13. Что такое метод наименьших квадратов в контексте интерполяции?
14. Как работает метод наименьших квадратов для интерполяции?
15. Что такое интерполяция Лагранжа?
16. Как работает интерполяция Лагранжа?
17. Какие преимущества и недостатки имеет интерполяция Лагранжа?
18. Что такое интерполяция Ньютона?
19. В чем отличие интерполяции Ньютона от интерполяции Лагранжа?
20. Какую роль играют разделенные разности в интерполяции Ньютона?

Тема 9. Основы линейной алгебры

1. Что такое вектор?
2. Как записывается вектор в компонентной форме?
3. Что такое длина (норма) вектора?
4. Как вычислить длину вектора в двумерном пространстве?
5. Что такое скалярное произведение двух векторов?
6. Как вычислить скалярное произведение двух векторов?
7. Какие свойства имеет скалярное произведение?
8. Что такое ортогональные векторы?
9. Как вычислить векторное произведение двух векторов?
10. Что такое матрица?
11. Как записывается матрица в компонентной форме?
12. Как вычислить транспонированную матрицу?

4.8. Образцы экзаменационных билетов

Экзаменационные билеты включают теоретические вопросы и практические задания. Примеры:

- Объясните основные принципы работы с массивами данных в Python.
- Напишите программу для решения системы линейных уравнений.
- Постройте график зависимости физической величины от времени и проанализируйте его.

4.9. Образцы экзаменационных практических заданий

Практические задания на экзамене могут включать:

- Написание программ для моделирования физических процессов.
- Выполнение численных расчетов с использованием библиотек NumPy и SymPy.
- Создание графиков и визуализация данных.

4.10. Банк тестовых заданий для самоконтроля

Банк тестовых заданий включает вопросы разного уровня сложности для самопроверки студентов. Примеры:

- Что такое переменная в Python и как её объявить?
- Какой метод используется для решения нелинейных уравнений?
- Как визуализировать данные с использованием Matplotlib?

4.11. Методики решения и ответы к образцам тестовых заданий

Методики решения содержат пошаговые инструкции и примеры выполнения тестовых заданий. Они помогают студентам понять логику решения задач и подготовиться к контрольным и экзаменационным испытаниям. Ответы к заданиям позволяют студентам проверить свои знания и выявить пробелы в понимании материала.

4.12. Образцы экзаменационных билетов

МОО ВО РОССИЙСКО-АРМЯНСКИЙ УНИВЕРСИТЕТ
ИНСТИТУТ МАТЕМАТИКИ, ФИЗИКИ И ВЫСОКИХ ТЕХНОЛОГИЙ
Кафедра общей физики и квантовых наноструктур

Направление: Электроника и наноэлектроника
Дисциплина: Программирование в физике
(бакалавриат I-ый курс, II-ой семестр)

Экзаменационный билет № **

1. Как работают логические операторы в Python? Как использовать операторы сравнения в Python?
2. Как выполнять операции над массивами (сложение, умножение и т.д.)? Как использовать булевы маски для работы с массивами?
3. Какие библиотеки Python используются для регрессионного анализа?

Зав. кафедройОФКН _____Д.Б. Айрапетян
20 г.

5. Методический блок

5.12. Методика преподавания

5.12.1. Методические рекомендации для студентов по подготовке к семинарским, практическим или лабораторным занятиям, по организации самостоятельной работы студентов при изучении конкретной дисциплины.

Подготовка к семинарским занятиям

1. **Изучение теоретического материала**: Перед каждым семинаром внимательно изучите соответствующий раздел учебника и дополнительные материалы, предоставленные преподавателем. Обратите внимание на основные концепции, определения и примеры.
2. **Конспектирование**: Составьте конспект ключевых моментов и вопросов, которые необходимо обсудить на семинаре. Это поможет лучше понять материал и подготовиться к активному участию в дискуссии.
3. **Просмотр видеолекций**: Если доступны видеолекции по теме семинара, обязательно посмотрите их. Видеолекции помогут лучше усвоить материал и подготовиться к обсуждению.
4. **Подготовка вопросов**: Сформулируйте вопросы по теме, которые остались непонятными или требуют дополнительного пояснения. Это позволит вам активнее участвовать в обсуждениях и получать разъяснения от преподавателя.

Подготовка к практическим занятиям

1. **Программирование на Python**: Убедитесь, что знакомы с основами языка Python. Выполните несколько простых упражнений, чтобы освежить знания.
2. **Решение задач**: Попробуйте решить несколько задач по теме предстоящего занятия самостоятельно. Это могут быть задачи из учебника или дополнительных источников.
3. **Анализ примеров кода**: Ознакомьтесь с примерами кода, которые будут обсуждаться на занятии. Попробуйте понять их работу и, если возможно, выполните их на своем компьютере.
4. **Подготовка к обсуждению**: Обсудить свои решения и подходы к задачам с преподавателем и одногруппниками. Запишите вопросы или проблемы, с которыми вы столкнулись при выполнении упражнений.

Организация самостоятельной работы

1. **Планирование времени**: Составьте расписание для самостоятельной работы. Определите время для чтения теоретического материала, выполнения практических заданий и повторения изученного материала.
2. **Использование дополнительных ресурсов**: Воспользуйтесь дополнительными ресурсами, такими как онлайн-курсы, видеолекции, статьи и блоги, чтобы углубить свои знания по теме.
3. **Регулярная практика**: Регулярно выполняйте упражнения и задачи, чтобы закрепить изученный материал. Постепенно увеличивайте сложность задач.

4. **Обратная связь**: Обсуждайте свои решения с одногруппниками и преподавателем. Не стесняйтесь задавать вопросы и просить разъяснений.
5. **Работа в группе**: Сотрудничайте с другими студентами. Обсуждайте решения задач, делитесь опытом и помогайте друг другу.