МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ,
МИНИСТЕРСТВО НАУКИ, ВЫСШЕГО ОБРАЗОВАНИЯ И ИННОВАЦИЙ
КЫРГЫЗСКОЙ РЕСПУБЛИКИ

МОО ВО Кыргызско-Российский Славянский университет
имени первого Президента Российской Федерации Б.Н. Ельцина

УТВЕРЖДАЮ
директор Политехнического института
_____ Т.В. Яцев
3 сентября 2020 г.

МОДУЛЬ: ВВЕДЕНИЕ В ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ
Современные языки программирования

рабочая программа дисциплины (модуля)

Разработано **Высшая школа Цифровых технологий**

Учебный план b150303_26_1 мех.plx
Квалификация Направление 15.03.03 - РФ, 650500 - КР Прикладная механика
Профиль "Вычислительная механика и компьютерный инжиниринг"

бакалавр

Форма обучения **очная**

Общая трудоемкость **3 ЗЕТ**
Часов по учебному плану 108
в том числе:
аудиторные занятия 54
самостоятельная работа 53,9

Виды контроля в семестрах:

зачет с оценкой 6

Распределение часов дисциплины по семестрам

Семестр (<Курс>.<Семестр на курсе>)	**6 (3.2)**		Итого	
Недель	16			
Вид занятий	УП	РП	УП	РП
Лекции	18	18	18	18
Практические	36	36	36	36
Контактная работа в период теоретического обучения	0,1	0,1	0,1	0,1
В том числе инт.	2	2	2	2
Итого ауд.	54	54	54	54
Контактная работа	54,1	54,1	54,1	54,1
Сам. работа	53,9	53,9	53,9	53,9
Итого	108	108	108	108

Программу составил(и):
ст. преподаватель, Джалилова Т.Я.; к. т. н., доцент, Хмелёва И.В. _____

Рецензент(ы):
д.т.н., профессор, Верзунов С.Н. _____

Согласовано РОП:
канд. физ.-мат. наук, доцент, Герман К. А. _____

Рабочая программа дисциплины
Современные языки программирования

разработана в соответствии с ФГОС 3++:
Федеральный государственный образовательный стандарт высшего образования - бакалавриат по направлению подготовки 15.03.03 Прикладная механика (приказ Минобрнауки России от 09.08.2021 г. № 729)

составлена на основании учебного плана:

Направление 15.03.03 - РФ, 650500 - КР Прикладная механика
Профиль "Вычислительная механика и компьютерный инжиниринг"
утвержденного учёным советом вуза от 26.06.2026 протокол № 11.

Рабочая программа одобрена на заседании *Высшей школы цифровых технологий*

Протокол от 02.09.2026 г. № 1
Срок действия программы: 2026-2030 уч.г.
Руководитель подразделения: к.т.н., доцент Хмелёва И.В.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ
к рабочей программе дисциплины (модуля)

Современные языки программирования

_____ *наименование дисциплины*

№ п/п	Основание внесения изменения	Раздел/ пункт	Содержание изменения
1.			
2.			
3.			
4.			
5.			

1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ	
1.1	Формирование базовых навыков программирования: Овладение синтаксисом и возможностями языка Python как универсального инструмента для решения научно-технических задач в области прикладной механики.
1.2	Автоматизация инженерных вычислений: Обучение методам обработки больших массивов данных, автоматизации рутинных расчетов и разработки прикладного программного обеспечения для нужд компьютерного инжиниринга.
1.3	Освоение методов объектно-ориентированного программирования (ООП): Использование объектного подхода для создания сложных механических моделей и иерархий расчетных сущностей.
1.4	Разработка пользовательских интерфейсов (GUI): Создание программных комплексов с графическим интерфейсом для визуализации результатов моделирования и управления расчетными модулями.
1.5	Интеграция с инженерным ПО: Подготовка базы для использования Python в качестве скриптового языка в CAE-системах (ANSYS, Abaqus и др.) и работы с библиотеками численного анализа (NumPy, SciPy).

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП	
Цикл (раздел) ООП:	Б1.О.1.09
2.1	**Требования к предварительной подготовке обучающегося:**
2.1.1	Использование современного программного комплекса mat lab
2.1.2	Основы алгоритмизации и программирования
2.1.3	Информационные технологии и основы информационной безопасности
2.2	**Дисциплины и практики, для которых освоение данной дисциплины (модуля) необходимо как предшествующее:**
2.2.1	Экспериментальная механика деформируемого твердого тела
2.2.2	Компьютерный инжиниринг
2.2.3	Планирование эксперимента и методы обработки данных

3. КОМПЕТЕНЦИИ ОБУЧАЮЩЕГОСЯ, ФОРМИРУЕМЫЕ В РЕЗУЛЬТАТЕ ОСВОЕНИЯ ДИСЦИПЛИНЫ (МОДУЛЯ)	
ОПК-14: Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения.	
Знать:	
Уровень 1	Синтаксис Python и стандарты оформления кода (PEP 8). Алгоритмы численных методов, ветвления, циклы и структуры данных. Основы ООП для моделирования механических систем. Принципы построения графических интерфейсов (GUI).
Уровень 2	
Уровень 3	
Уметь:	
Уровень 1	Формализовать инженерные задачи в виде программного кода. Создавать расчетные модули с автоматизацией ввода-вывода (файлы TXT/CSV). Разрабатывать интерактивные приложения с оконным интерфейсом. Отлаживать и оптимизировать код для вычислений.
Уровень 2	
Уровень 3	
Владеть:	
Уровень 1	Навыками разработки прикладного ПО для задач вычислительной механики. Методикой модульного проектирования расчетных систем. Техникой интеграции Python-скриптов в инженерные CAE-пакеты. Культурой разработки надежного и документированного кода.
Уровень 2	
Уровень 3	

В результате освоения дисциплины обучающийся должен

3.1	**Знать:**
3.1.1	Синтаксис и библиотеки: базовые конструкции Python, стандартную библиотеку и основы научных пакетов (NumPy).
3.1.2	Алгоритмизацию: методы построения линейных, разветвляющихся и циклических алгоритмов для инженерных задач.
3.1.3	ООП и GUI: принципы объектно-ориентированного подхода и теорию построения графических интерфейсов пользователя.
3.2	**Уметь:**

3.2.1	Программировать: переводить математические модели механики в программный код.
3.2.2	Работать с данными: автоматизировать чтение и запись данных через файлы (TXT, CSV) для связи с CAE-системами.
3.2.3	Создавать интерфейсы: разрабатывать оконные приложения для визуализации и управления расчетами.
3.3	**Владеть:**
3.3.1	Разработки ПО: создания практически применимых программ для вычислительной механики и инжиниринга.
3.3.2	Отладки: поиска и устранения логических и синтаксических ошибок в расчетном коде.
3.3.3	Проектирования: использования ООП для создания расширяемых иерархий физических объектов и материалов.
3.3.4	Автоматизации инженерного анализа: написания скриптов для пакетной обработки результатов конечно-элементного моделирования.
3.3.5	Работы в IDE: использования сред разработки для контроля версий и документирования инженерного ПО.

4. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

Код занятия	Наименование разделов и тем / вид занятия/	Семестр / Курс	Часов	Компетен-ции	Литература	Инте ракт.	Пр. подг.	Примечание
	Раздел 1. Введение в Python. Типы данных, ввод/вывод, библиотека math.							
1.1	Введение в Python. Типы данных, ввод/вывод, библиотека math. /Лек/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2 Э3			
1.2	Знакомство с синтаксисом и базовые арифметические операции. Работа с вещественными числами и форматированием строк. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2 Э3			
1.3	Стандарты оформления кода и базовые алгоритмы. Изучение PEP 8. Работа с библиотекой math для реализации сложных инженерных формул. /Ср/	6	6		Л1.1 Л1.2Л2.1 Э1 Э2 Э3			
1.4	Управляющие конструкции: ветвления и сложные логические условия. Циклы for и while. Итерационные процессы и вложенные циклы. /Лек/	6	4		Л1.1 Л1.2Л2.1 Э1 Э2			
1.5	Программирование разветвляющихся алгоритмов. /Пр/	6	4		Л1.1 Л1.2Л2.1 Э1 Э2	2		
1.6	Вложенные условия и тернарный оператор. /Пр/	6	4		Л1.1 Л1.2Л2.1 Э1 Э2			
1.7	Логика и ветвления в инженерных задачах. Разработка алгоритмов проверки физической корректности данных (граничные условия, знаки напряжений). /Ср/	6	4		Л1.1 Л1.2Л2.1 Э1 Э2			
1.8	Табулирование функций и работа с циклами. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
1.9	Обработка данных в циклах: поиск экстремумов /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
1.10	Циклические вычисления и ряды. Реализация вычисления функций через ряды (Тейлора, Фурье). Работа со вложенными циклами для построения сеток. /Ср/	6	6		Л1.1 Л1.2Л2.1 Э1 Э2			

	Раздел 2. Коллекции, функции и основы ООП							
2.1	Коллекции: списки, кортежи, словари. Срезы и методы. /Лек/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2 Э3			
2.2	Работа со списками. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
2.3	Матрицы и словари. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2 Э3			
2.4	Обработка массивов и коллекций. Изучение методов глубокого и поверхностного копирования списков. Работа с матрицами через вложенные списки. /Ср/	6	6		Л1.1 Л1.2Л2.1 Э1 Э2			
2.5	Функциональное программирование. Lambda, map, filter. /Лек/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
2.6	Создание функций. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
2.7	Рекурсия и генераторы. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
2.8	Модульное программирование. Создание собственных модулей и библиотек функций. Изучение рекурсивных алгоритмов в механике. /Ср/	6	6		Л1.1 Л1.2Л2.1 Э1 Э2			
2.9	Введение в ООП. Классы, объекты, инкапсуляция и конструктор __init__. /Лек/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
2.10	Создание базового класса. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
2.11	Взаимодействие объектов класса. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
2.12	Проектирование иерархии классов. Разработка классов для физических объектов (узел, элемент, опора, сила). Изучение инкапсуляции. /Ср/	6	6		Л1.1 Л1.2Л2.1 Э1 Э2			
	Раздел 3. Продвинутое ООП и графический интерфейс (GUI)							
3.1	Продвинутое ООП. Наследование, полиморфизм и обработка исключений. /Лек/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
3.2	Иерархия классов. Обработка ошибок в ООП-коде. /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
3.3	Файловый ввод-вывод и форматы данных. Изучение парсинга CSV и JSON файлов. Интеграция данных из внешних расчетных систем. /Ср/	6	6,1		Л1.1 Л1.2Л2.1 Э1 Э2			
3.4	Основы GUI (библиотека Tkinter/PyQt). Виджеты, окна и события. /Лек/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			

3.5	Создание главного окна и кнопок. Реализация обработчиков событий (связь кнопок с функциями). /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
3.6	Обработка событий нажатия. / Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
3.7	Обработка исключений в GUI. Разработка систем защиты интерфейса от некорректного ввода пользователя. /Ср/	6	6		Л1.1 Л1.2Л2.1 Э1 Э2			
3.8	Связывание логики и интерфейса. Работа с файлами через GUI. /Лек/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
3.9	Ввод данных из GUI в файл. / Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
3.10	Разработка мини-проекта (Калькулятор/Менеджер). /Пр/	6	2		Л1.1 Л1.2Л2.1 Э1 Э2			
3.11	Подготовка итогового проекта. Связывание графической оболочки с расчетным ядром на базе ООП. Документирование кода. /Ср/	6	7,8		Л1.1 Л1.2Л2.1 Э1 Э2			
3.12	/КрТО/	6	0,1		Л1.1 Л1.2Л2.1 Э1 Э2			
3.13	/ЗачётСОц/	6			Л1.1 Л1.2Л2.1 Э1 Э2			

5. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

5.1. Контрольные вопросы и задания

Модуль 1. Основы синтаксиса и алгоритмические конструкции
Фокус: Базовые вычисления, типы данных и логика управления.
Знать:
1. В чем принципиальное различие между компилируемыми (C++) и интерпретируемыми (Python) языками?
2. Какие типы данных в Python являются изменяемыми, а какие — нет?
3. Как работают операторы деления (/), целочисленного деления (//) и остатка (%)?
4. Каков порядок приоритета арифметических и логических операций?
5. Что такое динамическая типизация и как она влияет на выделение памяти под переменные?
6. Каков синтаксис и правила использования функции range() в циклах?
7. В чем разница между операторами сравнения == и оператором идентичности is?
8. Как работают инструкции break и continue в сложных циклах?
9. Что такое «магические числа» и почему их следует заменять константами?
10. Какие стандарты оформления кода (PEP 8) являются обязательными для читаемости программы?

Уметь:
1. Реализовать ввод данных с клавиатуры и приведение их к вещественному типу (float).
2. Составить программу для решения квадратного уравнения с обработкой отрицательного дискриминанта.
3. Оформить вывод результатов расчета с использованием f-строк (округление до нужного знака).
4. Написать цикл для вычисления суммы ряда (например, разложение функции в ряд Тейлора).
5. Использовать математические функции модуля math (sin, cos, exp) в расчетах.
6. Реализовать проверку входных параметров на физическую корректность (например, площадь > 0).
7. Написать вложенный цикл для вывода таблицы значений функции двух переменных.
8. Преобразовать строковые данные, содержащие числа, в список числовых значений.
9. Создать алгоритм поиска наибольшего общего делителя для оптимизации шага сетки.
10. Использовать тернарный оператор для краткой записи простых условий.

Владеть:
1. Навыками отладки линейных программ в современных IDE (PyCharm, VS Code).
2. Методикой перевода сложных математических формул в безошибочный программный код.
3. Техникой безопасного ввода данных с защитой от некорректных типов.
4. Приемами оптимизации логических условий в разветвляющихся алгоритмах.
5. Навыками использования векторизованных операций (на базовом уровне) для замены простых циклов.

6. Способностью оценивать точность вычислений при работе с числами с плавающей точкой.
7. Методикой построения блок-схем для визуализации алгоритмов инженерных задач.
8. Навыками документирования кода через аннотации типов и комментарии.
9. Техникой работы с константами для управления глобальными параметрами модели.
10. Навыками самоконтроля кода на соответствие стандартам PEP 8.

_____

Модуль 2. Коллекции, функции и основы ООП

Фокус: Структуры данных для механики и объектный подход.

Знать:

1. В чем различие между списком (list) и кортежем (tuple) в контексте использования памяти?
2. Как работает механизм «срезов» (slices) в списках и строках?
3. Что такое словарь (dict) и почему он эффективен для хранения свойств материалов?
4. Какова роль аргумента self в методах класса Python?
5. Что такое рекурсия и в каких задачах механики (например, фракталы или деревья) она применима?
6. В чем разница между локальными и глобальными переменными (правило LEGB)?
7. Что такое Lambda-функции и когда их использование оправдано?
8. Для чего используется метод __init__ в классах?
9. Каковы принципы инкапсуляции данных в Python (публичные и приватные атрибуты)?
10. Как работают функции высшего порядка map(), filter() и zip()?

Уметь:

1. Создать список координат узлов расчетной сетки и выполнить его обход.
2. Реализовать функцию, принимающую произвольное число аргументов (*args, **kwargs).
3. Разработать класс «Материал» с атрибутами (модуль Юнга, коэффициент Пуассона).
4. Использовать генераторы списков (list comprehensions) для создания матриц.
5. Написать рекурсивную функцию для расчета чисел Фибоначчи или факториала.
6. Реализовать метод класса, вычисляющий напряжение по заданным силе и площади.
7. Выполнить сортировку списка объектов по заданному критерию (например, по цене или массе).
8. Создать словарь, связывающий ID узла с его координатами.
9. Написать функцию cDocstrings, описывающими входные параметры и возвращаемое значение.
10. Использовать множества (set) для поиска уникальных элементов в массиве данных.

Владеть:

1. Методикой декомпозиции инженерной задачи на отдельные функции и классы.
2. Навыками объектно-ориентированного проектирования простейших механических систем.
3. Техникой работы со вложенными структурами данных (списки списков для матриц).
4. Навыками создания модульных программ, состоящих из нескольких файлов.
5. Приемами функционального программирования для обработки массивов данных.
6. Навыками управления состоянием объектов в процессе выполнения программы.
7. Техникой реализации интерфейсов взаимодействия между объектами классов.
8. Методами профилирования функций для выявления «узких мест» в расчетах.
9. Опытом работы с итераторами и генераторами для экономии памяти.
10. Навыками применения паттернов проектирования (на базовом уровне) в ООП.

_____

Модуль 3. Графический интерфейс (GUI) и работа с файлами

Фокус: Создание инженерных приложений и интеграция с CAE.

Знать:

1. Какова иерархия обработки исключений (try-except-finally) в Python?
2. Какие режимы открытия файлов существуют (r, w, a, b)?
3. Что такое контекстный менеджер with и почему он обязателен при работе с файлами?
4. В чем разница между менеджерами геометрии pack, grid и place в Tkinter?
5. Что такое «событийно-ориентированное программирование» в контексте GUI?
6. Как реализовать механизм наследования классов для расширения функционала виджетов?
7. Как работают переменные управления (StringVar, IntVar) в графическом интерфейсе?
8. Что такое сериализация данных и форматы CSV/JSON?
9. Как организовать диалоговое окно для выбора файла на диске?
10. В чем преимущество Модульения логики расчета и кода интерфейса (паттерн MVC)?

Уметь:

1. Написать программу, считывающую матрицу жесткости из текстового файла.
2. Создать графическое окно с полями ввода для параметров балки и кнопкой расчета.
3. Реализовать обработку ошибки ZeroDivisionError при расчетах через GUI.
4. Организовать запись результатов моделирования в CSV-файл для Excel.
5. Создать выпадающий список (OptionMenu/Combobox) для выбора типа материала.
6. Написать обработчик события нажатия кнопки, который запускает расчетный модуль.
7. Создать класс-исключение для специфических инженерных ошибок.
8. Реализовать графическую метку (Label), цвет которой меняется в зависимости от результата.
9. Организовать чтение данных из JSON-файла конфигурации проекта.
10. Разместить элементы интерфейса в виде таблицы (Grid) для создания калькулятора.

Владеть:

1. Навыками проектирования пользовательских интерфейсов для инженерных расчетов.

2. Техникой создания отказоустойчивых программных комплексов.
3. Навыками интеграции Python-скриптов с внешними данными (например, экспорт в CAE).
4. Методикой тестирования интерфейса на корректность пользовательского ввода.
5. Навыками работы с бинарными файлами для хранения больших объемов данных.
6. Техникой визуализации прогресса вычислений в графическом окне.
7. Опытом разработки кроссплатформенных приложений на Python.
8. Навыками связывания сложной логики на базе ООП с элементами управления GUI.
9. Техникой обработки системных прерываний и ошибок ввода-вывода.
10. Методами подготовки программного продукта к передаче конечному пользователю.

ПЕРЕЧЕНЬ ВОПРОСОВ К ЗАЧЕТУ
История возникновения и философия языка Python. Преимущества и недостатки перед компилируемыми языками (C++).
Интерпретируемость и динамическая типизация: понятия и особенности реализации в Python.
Основные типы данных (int, float, bool, str). Приведение типов и встроенные функции преобразования.
Арифметические операции в Python: особенности деления нацело (//) и нахождения остатка (%). Приоритеты операций.
Логические выражения и операторы сравнения. Особенности работы операторов and, or, not.
Условный оператор if-elif-else: синтаксис и правила вложенности.
Цикл с условием while. Бесконечные циклы и способы их предотвращения.
Цикл со счетчиком for и функция range(). Итерирование по различным объектам.
Управляющие инструкции в циклах: break, continue и блок else для циклов.
Списки (list): создание, индексация и методы изменения (append, extend, remove, pop).
Срезы (slices) в списках и строках: синтаксис, шаг и использование для инверсии данных.
Кортежи (tuple): отличия от списков, преимущества использования и область применения.
Словари (dict): структура «ключ-значение», методы работы и правила формирования ключей.
Множества (set): теоретико-множественные операции (объединение, пересечение) и удаление дубликатов.
Понятие функции в Python. Определение def, передача аргументов и оператор return.
Позиционные и именованные аргументы функций. Значения аргументов по умолчанию.
Произвольное количество аргументов: использование *args и **kwargs.
Области видимости переменных: локальные, глобальные переменные и правило LEGB.
Анонимные функции (lambda): синтаксис и совместное использование с map и filter.
Генераторы списков (List Comprehensions) как эффективный инструмент создания коллекций.
Работа с текстовыми файлами: режимы открытия, методы read, write и контекстный менеджер with.
Обработка исключений: иерархия ошибок, блоки try, except, finally и else.
Понятие объектно-ориентированного программирования. Определение класса, объекта и экземпляра.
Конструктор класса __init__ и назначение параметра self.
Инкапсуляция в Python: соглашения об именовании атрибутов (публичные, защищенные, приватные).
Наследование: создание дочерних классов и использование функции super().
Полиморфизм в Python: переопределение методов и «утиная типизация».
Основы графического интерфейса (библиотека Tkinter): создание главного окна и цикл mainloop.
Основные виджеты GUI: Label, Button, Entry. Менеджеры геометрии pack и grid.
Событийно-ориентированное программирование: связывание функций с событиями виджетов.

5.2. Темы курсовых работ (проектов)

курсовой не предусмотрен

5.3. Фонд оценочных средств

5.4. Перечень видов оценочных средств

Защита проекта по итогам самостоятельной работы, практические работы, контрольные работы, промежуточная аттестация.

6. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

6.1. Рекомендуемая литература

6.1.1. Основная литература

	Авторы, составители	Заглавие	Издательство, год
Л1.1	Кольцов Д.М.	PYTHON. ПОЛНОЕ РУКОВОДСТВО.	НАУКА и ТЕХНИКА 2022
Л1.2	Васильев, А	Программирование на Python в примерах и задачах	Эксмо 2021

6.1.2. Дополнительная литература

	Авторы, составители	Заглавие	Издательство, год

	Авторы, составители	Заглавие	Издательство, год
Л2.1	Сузи Р. А.	Язык программирования Python: учебное пособие	Москва: Интернет-Университет Информационных Технологий (ИНТУИТ) 2016
6.2. Перечень ресурсов информационно-телекоммуникационной сети "Интернет"			
Э1	Официальный сайт Python и документация		https://www.python.org/
Э2	Справочник по языку Python		https://docs-python.ru/tutorial/
Э3	Python 3		https://python-scripts.com/

6.3. Перечень информационных и образовательных технологий	
6.3.1 Компетентностно-ориентированные образовательные технологии	
6.3.1.1	1. Лекции с применением мультимедийных материалов, мультимедийная аудитория.
6.3.1.2	2. Перечень образовательных технологий, используемых при осуществлении образовательного процесса по дисциплине в компетентностно-ориентированной форме.
6.3.1.3	Овладение дисциплиной предполагает использование следующих образовательных технологий (методов):
6.3.1.4	• лекция (вводная, обзорная, репродуктивно-информационная, заключительная) - целесообразность традиционной лекции состоит в решении
6.3.1.5	следующих образовательных и развивающих задач курса: показать значимость курса для профессионального становления будущего педагога; представить
6.3.1.6	логическую схему изучения представленного курса; сформировать мотивацию магистрантов на освоение учебного материала; связать теоретический материал с
6.3.1.7	практикой будущей профессиональной деятельности; представить научнопонятийную основу изучаемой дисциплины; систематизировать знания
6.3.1.8	студентов по изучаемой проблеме; расширить научный кругозор магистра как будущего специалиста высокого уровня и т.д.;
6.3.1.9	• лекция-беседа - позволяет учитывать отношение обучающегося к изучаемым вопросам, выявлять проблемы в процессе их осмысления, корректировать допускаемые ошибки и т.д.;
6.3.1.10	• лекция-дискуссия - представляет организацию диалоговой формы обучения, создающей условия для формирования оценочных знаний магистрантов, обусловливающих проявление их профессиональной позиции как будущего
6.3.1.11	специалиста высокого уровня; формируется умение высказывать и аргументировать личную точку зрения; развивается способность к толерантному восприятию иных точек зрения и т.д.;
6.3.1.12	• «мозговой штурм» - метод коллективного генерирования идей и ихконструктивная проработка при решении проблемных задач предполагает создание условий для развития умений выражать собственные взгляды, работать
6.3.1.13	во взаимодействии с другими людьми и т.д.;
6.3.1.14	• лекция с разбором конкретных ситуаций – предполагает включение конкретных ситуаций, отражающих проблемы профессиональной деятельности; создаётся ситуация, позволяющая «перевод» познавательного интереса на уровень
6.3.1.15	профессионального; активизируется возможность занять профессиональную позицию, развить умения анализа, сравнения и обобщения;
6.3.1.16	• разработка программ исследования – предполагает развитие умений системно представить программу изучения математических понятий в физике;
6.3.1.17	• тренинг по использованию методов исследования при изучении конкретных проблем математики – отрабатывается умение и навыки решения математических задач и построения математических моделей;
6.3.1.18	• рефлексия - обеспечивает самоанализ и самооценку достижения результатов познавательной деятельности.
6.3.2 Перечень информационных справочных систем и программного обеспечения	
6.3.2.1	Программное обеспечение
6.3.2.2	Среда разработки: PyCharm Community Edition или Visual Studio Code. или PyScripter
6.3.2.3	1. https://www.python.org/ – сайт, содержащий необходимые дистрибутивы и полную информацию для языка программирования Python. Интерпретатор для Python можно использовать как программируемый высокоуровневый калькулятор.
6.3.2.4	2. http://sympy.org/ – сайт, посвященный свободно распространяемому пакету Sympy, представляющему собой библиотеку Python символьных вычислений.
6.3.2.5	3. http://github.com/sympy/sympy – сайт, посвященный свободно распространяемому пакету Sympy, представляющему собой библиотеку Python символьных вычислений. Сайт свободных новинок и постоянного
6.3.2.6	пользовательского обновления данного пакета (реализованный напринципах Вики).

6.3.2.7	4. http://gouspo.ru/ – сайт, созданный для студентов средних и высших учебных заведений, представляющий научно-информационный ресурс по криптографии и теории кодирования, а также по связанных с ними
6.3.2.8	областями теоретической и прикладной математики.
6.3.2.9	5. http://univertv.ru/video/matematika/ Открытый образовательный видеопортал UniverTV.ru. Образовательные фильмы на различные темы. Лекции в ведущих российских и зарубежных вузах. Научная конференция или научнопопулярная лекция по интересующему вас вопросу.
6.3.2.10	6. http://www.iqlib.ru/ Электронная библиотека IQlib образовательных и просветительских изданий. Образовательный ресурс, объединяющий в себе интернет-библиотеку и пользовательские сервисы для полноценной работы с
6.3.2.11	библиотечными фондами. Свободный доступ к электронным учебникам, справочным и учебным пособиям. Аудитория электронной библиотеки IQlib – студенты, преподаватели учебных заведений, научные сотрудники и все
6.3.2.12	те, кто хочет повысить свой уровень знаний.
6.3.2.13	7. http://eqworld.ipmnet.ru/ru/library.htm EqWorld – мир математических уравнений. Учебно-образовательная физико-математическая библиотека. Электронная библиотека содержит DjVu- и PDF-файлы учебников, учебных
6.3.2.14	пособий, сборников задач и упражнений, конспектов лекций, монографий, справочников и диссертаций по математике, механике и физике. Все материалы присланы авторами и читателями или взяты из Интернета (из
6.3.2.15	www архивов открытого доступа). Основной фонд библиотеки составляют книги, издававшиеся тридцать и более лет назад.
6.3.2.16	8. http://www.edu.ru/modules.php?op=modload&name=Web_Links&file=index&l_op=viewlink&cid=1314 Федеральный портал "Российское образование". Каталог образовательных ресурсов.

7. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

7.1	При проведении лекционных и практических занятий используются мультимедийные средства, компьютерные классы, интерактивные доски, а так же классическое учебное оборудование: конференц-зал, оборудованный доской, инструментами, раздаточным материалом, учебной и методической литературой, периодической литературой по предмету.
7.2	Для проведения занятий лекционного типа предлагаются наборы демонстрационного оборудования и учебно-наглядных пособий, обеспечивающие тематические иллюстрации, соответствующие рабочей программе дисциплины.
7.3	Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с подключением к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду организации. Содержание учебной дисциплины представлено в сети Интернет.
7.4	Научная библиотека КРСУ обладает достаточным для образовательного процесса количеством экземпляров учебной литературы и необходимым минимумом периодических изданий для осуществления методического и научно - исследовательского процесса. Имеются основные академические и отраслевые научные и методические журналы, аудитории, оснащенные учебно-методической литературой и средствами обучения.
7.5	Электронно-библиотечные системы (электронная библиотека) издательства IPRBooks и университетская библиотека online, электронная информационно-образовательная среда обеспечивают одновременный доступ обучающихся

8. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ (МОДУЛЯ)

Технологическая карта дисциплины: приложение . "Технологическая карта".
Общие рекомендации к организации самостоятельной работы.
Изучение дисциплины следует начинать с проработки настоящей рабочей программы, особое внимание, уделяя целям и задачам, структуре и содержанию курса.
Основными видами аудиторной работы студента при изучении курса " Современные языки программирования" являются лекции и практические занятия в компьютерном классе.
Студент не имеет права пропускать без уважительных причин аудиторные занятия.
На лекциях излагаются и разъясняются основные понятия темы, связанные с ней теоретические и практические проблемы, даются рекомендации для самостоятельной работы. В ходе лекции студент должен внимательно слушать и конспектировать новый материал. Завершают изучение наиболее важных тем или разделов учебной дисциплины практические занятия. Они служат для приобретения навыков работы с информационными технологиями по темам дисциплины.
Важным видом работы студента при изучении дисциплины является самостоятельная работа. Самостоятельная работа должна носить творческий и планомерный характер. В процессе организации самостоятельной работы большое значение имеют консультации преподавателя.
Работа с конспектом лекций.
1. Студенту необходимо просмотреть конспект сразу после занятий, отметить материал конспекта лекций, который вызывает затруднение для понимания.
2. Попытаться найти ответы на затруднительные вопросы, используя предлагаемую литературу. Если самостоятельно не удалось разобраться в материале, то необходимо сформулировать вопросы и обратиться на ближайшей лекции за помощью к преподавателю.
3. Каждую неделю нужно отводить время для повторения, пройденного материала.
Подготовка к практическим занятиям.
1. Перед посещением лабораторного занятия изучить теорию вопроса, предполагаемого к исследованию.

2. Проверять свои знания, умения и навыки при решении задач на практических занятиях.