МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ГРОЗНЕНСКИЙ ГОСУДАРСТВЕННЫЙ НЕФТЯНОЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ имени академика М.Д. Миллионщикова

«УТВЕРЖДАЮ»
Первый проректор
И.Г. Гайрабеков
«02» 09 2026 г.

РАБОЧАЯ ПРОГРАММА

дисциплины

«Разработка ПО на языке Python»

Направление подготовки/специальность

09.03.03 Прикладная информатика

Направленность/специализация

«Разработка мобильных и Web–приложений»

Квалификация

Бакалавр

Год начала подготовки

2026

Грозный 2026

1. Цели и задачи освоения дисциплины

Целью освоения дисциплины является подготовка обучающихся к организационно–управленческому виду деятельности по направлению подготовки 09.03.03 Прикладная информатика (профиль подготовки: Разработка мобильных и Web–приложений) посредством обеспечения этапов формирования компетенций, предусмотренных ФГОС, в части представленных ниже знаний, умений и навыков.

Задачами дисциплины является изучение понятийного аппарата дисциплины, основных теоретических положений и методов, формирование умений и привитие навыков применения теоретических знаний для решения практических и прикладных задач.

2. Место дисциплины в структуре образовательной программы

Дисциплина относится к «Часть, формируемая участниками образовательных отношений» блока 1.

В свою очередь, данный курс, помимо самостоятельного значения, является последующей дисциплиной после курсов: «Программирование», «Языки и методы разработки Web-приложений», «Вычислительные системы, сети и телекоммуникации»

А также предшествующей дисциплиной для курсов: «Разработка Web-приложений с применением фреймворков», «Серверная разработка Web-приложений», «Продвинутая разработка мобильных приложений: код, архитектура, алгоритмы»,

3. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с индикаторами достижения компетенций

Таблица 1

Компетенции дисциплины

Код по ФГОС	Индикаторы достижения	Планируемые результаты обучения по дисциплине (ЗУВ)
Профессиональные		
ПК – 5. Способен разрабатывать и адаптировать прикладное программное обеспечение	ПК – 5.1 Проектирует и разрабатывает прототипы ИС ПК – 5.2 Применяет технологии разработки и управления базами данных ИС ПК – 5.3 Обеспечивает соответствие разработанного кода и процесса кодирования на языках программирования принятым в организации или проекте стандартам и технологиям	**Знает** методы и технологии проектирования и разработки прототипов информационных систем, включая управление базами данных и стандарты кодирования. Осведомлён о различных языках программирования и подходах к адаптации прикладного программного обеспечения под специфические требования проекта или организации. **Умеет** проектировать и разрабатывать прототипы информационных систем, применять современные технологии для разработки и управления базами данных. Способен обеспечивать соответствие разработанного кода принятым стандартам и технологиям, а также адаптировать процессы кодирования для оптимизации производительности и соблюдения качества. **Владеет** навыками эффективной разработки и адаптации прикладного программного обеспечения в соответствии с требованиями и стандартами проекта. Использует передовые методы программирования и управления данными для создания стабильных, эффективных и легко адаптируемых приложений.
Общепрофессиональные		
ОПК-2. Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач	ОПК-2.1. Выбирает современные информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности.	***Знает*** современные информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности. ***Умеет*** применять современные информационные технологии и

профессиональной деятельности	ОПК-2.2. Применяет современные информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности.	программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности. ***Владеет*** навыками взаимодействия с современными информационными технологиями и программными средствами, в том числе отечественного производства, при решении задач профессиональной деятельности.

4. Объем дисциплины и виды учебной работы

Таблица 2

Вид учебной работы		Всего часов/ зач.ед. ОФО 4 семестр	Всего часов/ зач.ед. ОФО 5 семестр	Всего часов/ зач.ед. ЗФО 4 семестр	Всего часов/ зач.ед. ЗФО 5 семестр
Контактная работа (всего)		64/1.8	51/1.4	16	14
В том числе:					
Лекции		32/0.9	17/0.5	8	4
Практические занятия					
Семинары					
Лабораторные работы		32/0.9	34/0.9	8	10
Самостоятельная работа (всего)		72/2	101/2.8	114	144
В том числе:					
Курсовая работа (проект)					
Расчетно–графические работы					
Индивидуальное задание		18/0.5	36/1	36	72
Рефераты					
Доклады					
Презентации					
И (или) другие виды самостоятельной работы:					
Подготовка к лабораторным работам		36/1	36/1	36	36
Подготовка к практическим занятиям					
Подготовка к экзамену					
Подготовка к зачету		18/0.5	29/0.8	42	36
Вид отчетности		Зачет	Экзамен	Зачет	Экзамен
Общая трудоемкость дисциплины	ОФО 288/8	136/3.7	136/3.7	130	158
	ЗФО				

5. Содержание дисциплины

5.1. Разделы дисциплины и виды занятий

Таблица 3

№ п/п	Наименование раздела дисциплины по семестрам	Часы лекц. занятий ОФО	Часы лаборат. занятий ОФО	Часы лекц. занятий ЗФО	Часы лаборат. занятий ЗФО
4 семестр					
1.	Тема 1: Введение в программирование и Python	**4**	**4**	2	2
2.	Тема 2: Переменные и типы данных	**4**	**4**	2	2
3.	Тема 3: Условные конструкции и Циклы	**4**	**4**	2	2
4.	Тема 4: Списки и кортежи	**4**	**4**	2	2
5.	Тема 5: Функции	**4**	**4**		
6.	Тема 6: Работа с файлами	**4**	**4**		
7.	Тема 7: Итераторы и генераторы	**4**	**4**		
8.	Тема 8: Регулярные выражения	**4**	**4**		
5 семестр					
9.	Тема 9: Введение в ООП	**2**	**4**	2	2
10.	Тема 10: Атрибуты, свойства и методы	**4**	**8**	2	2
11.	Тема 11: Магические методы	**4**	**8**		2
12.	Тема 12: Наследование, инкапсуляция и полиморфизм	**4**	**8**		2
13.	Тема 13: Модули и пакеты	**3**	**6**		2

5.2. Лекционные занятия

Таблица 4

Раздел	Наименование раздела дисциплины	Содержание раздела
1.	Лекция 1: Введение в программирование и Python	• Что такое программирование и зачем это нужно. • История и особенности Python. • Установка Python и настройка среды разработки. • Запуск первой программы.
2.	Лекция 2: Переменные и типы данных	• Основы переменных и их объявление. • Типы данных: целые числа, вещественные числа, строки, булевы значения. • Преобразование типов и операции над данными.
3.	Лекция 3: Условные конструкции и	• Операторы сравнения и

	Циклы	логические операторы. • Условные конструкции if, elif, else. • Циклы for и while, итерация по последовательностям. • Ключевые слова break, continue и else в циклах.
4.	Лекция 4: Списки и кортежи	• Создание и использование списков и кортежей. • Методы и функции для работы со списками. • Индексация и срезы.
5.	Лекция 5: Функции	• Определение и вызов функций. • Параметры и аргументы функций. • Возвращаемые значения. • Области видимости переменных. • Анонимные функции (лямбды).
6.	Лекция 6: Работа с файлами	• Открытие и закрытие файлов. • Чтение и запись данных в файлы. • Работа с файлами через менеджер контекста with.
7.	Лекция 7: Итераторы и генераторы	• Понятие и использование итераторов. • Создание генераторов с помощью функций и выражений. • Преимущества генераторов в Python.
8.	Лекция 8: Регулярные выражения	• Основы синтаксиса регулярных выражений. • Функции поиска, замены и разделения строк с использованием модуля re. • Примеры практического применения.
9.	Лекция 9: Введение в ООП	• Концепции объектно-ориентированного программирования. • Классы и объекты в Python. • Конструкторы и деструкторы.
10.	Лекция 10: Атрибуты, свойства и методы	• Атрибуты классов и экземпляров. • Геттеры, сеттеры и декоратор property. • Статические методы и методы класса.
11.	Лекция 11: Магические методы	• Основные магические

		методы (__init__, __str__, __repr__). • Методы для операций сравнения. • Магические методы для контейнеров.
12.	Лекция 12: Наследование, инкапсуляция и полиморфизм	• Основы и принципы наследования. • Примеры множественного наследования. • Инкапсуляция данных. • Полиморфизм и его реализация в Python.
13.	Лекция 13: Модули и пакеты	• Импорт стандартных и пользовательских модулей. • Создание и структурирование модулей. • Пакеты и пространства имен.

5.3. Практические (семинарские) занятия

Практические занятия учебным планом не предусмотрены.

5.4 Лабораторные занятия

Таблица 5

№ п/п	Наименование работы	Содержание лабораторной работы
1.	Лабораторная работа 1. Основы программирования на Python	Название: Основы программирования на Python Цель: познакомить студентов с основами программирования и языком Python, научить устанавливать Python и запускать базовые программы. Задания: 1. Установка Python 2. Настройка рабочей среды 3. Запуск первой программы
2.	Лабораторная работа 2. Работа с переменными и типами данных в Python	Название: Работа с переменными и типами данных в Python Цель: научить студентов создавать переменные, использовать различные типы данных и осуществлять базовые операции над данными в Python. Задания: 1. Создание переменных 2. Операции с числами 3. Работа со строками 4. Преобразование типов 5. Исследование операций

3.	Лабораторная работа 3. Управление потоком программы в Python	Название: Управление потоком программы в Python Цель: научить студентов использовать условные конструкции и циклы для контроля за выполнением программ, а также освоить итерацию по различным последовательностям. Задания: 1. Использование условных конструкций 2. Итерация с использованием цикла for 3. Использование цикла while 4. Управление выполнением циклов 5. Продвинутое использование циклов
4.	Лабораторная работа 4. Изучение списков и кортежей в Python	Название: Изучение списков и кортежей в Python Цель: познакомить студентов с основами работы со списками и кортежами, их методами и функциями, а также научить их использовать индексацию и срезы для доступа и манипуляции данными. Задания: 1. Создание списков и кортежей 2. Работа с методами списков 3. Индексация и срезы 4. Преобразование списков в кортежи и обратно 5. Продвинутое манипулирование списками
5.	Лабораторная работа 5. Основы работы с функциями в Python	Название: Основы работы с функциями в Python Цель: научить студентов создавать и использовать функции в Python, включая понимание параметров, аргументов, возвращаемых значений, областей видимости переменных и анонимных функций. Задания: 1. Создание и вызов функций 2. Параметры и аргументы функций 3. Возвращаемые значения 4. Области видимости переменных 5. Анонимные функции (лямбды)
6.	Лабораторная работа 6. Управление файлами в Python	Название: Управление файлами в Python Цель: научить студентов основам работы с файлами в Python, включая открытие, чтение, запись и использование менеджера контекста для эффективного управления ресурсами. Задания: 1. Открытие и закрытие файлов 2. Чтение данных из файла 3. Запись данных в файл 4. Работа с файлами через менеджера контекста with 5. Расширенное задание: работа с двоичными файлами

7.	Лабораторная работа 7. Итераторы и генераторы в Python	Название: Итераторы и генераторы в Python Цель: обучить студентов основам работы с итераторами и генераторами в Python, показать их преимущества и научить создавать генераторы с помощью функций и выражений. Задания: 1. Использование итераторов 2. Создание генератора с помощью функции 3. Создание генератора с помощью генераторного выражения 4. Преимущества использования генераторов 5. Продвинутое задание: Генератор для чтения файлов
8.	Лабораторная работа 8. Освоение регулярных выражений в Python	Название: Освоение регулярных выражений в Python Цель: познакомить студентов с синтаксисом и функциями модуля re для работы с регулярными выражениями в Python, научить их использовать эти инструменты для поиска, замены и разделения текстовых данных. Задания: 1. Основы синтаксиса регулярных выражений 2. Функция поиска 3. Функция замены 4. Функция разделения 5. Продвинутое использование регулярных выражений
9.	Лабораторная работа 9. Основы объектно-ориентированного программирования в Python	Название: Основы объектно-ориентированного программирования в Python Цель: познакомить студентов с основными концепциями объектно-ориентированного программирования, включая создание классов, объектов, и использование конструкторов и деструкторов в Python. Задания: 1. Основы ООП и создание классов 2. Создание объектов 3. Конструкторы 4. Деструкторы 5. Понимание концепций ООП
10.	Лабораторная работа 10. Изучение атрибутов, свойств и методов в объектно-ориентированном программировании Python	Название: Изучение атрибутов, свойств и методов в объектно-ориентированном программировании Python Цель: научить студентов различать атрибуты классов и экземпляров, использовать геттеры и сеттеры с декоратором property, а также применять статические методы и методы класса для управления и обработки данных класса. Задания: 1. Атрибуты классов и экземпляров 2. Геттеры, сеттеры и декоратор property 3. Статические методы и методы класса 4. Продвинутое задание: Использование всех типов методов

11.	Лабораторная работа 11. Магические методы в Python	Название: Магические методы в Python Цель: изучить основные магические методы в Python, научиться использовать их для настройки поведения классов при создании экземпляров, форматировании вывода, выполнении операций сравнения и работы с контейнерами. Задания: 1. Использование методов __init__, __str__, и __repr__ 2. Методы для операций сравнения 3. Магические методы для контейнеров
12.	Лабораторная работа 12. Наследование, инкапсуляция и полиморфизм в Python	Название: Наследование, инкапсуляция и полиморфизм в Python Цель: познакомить студентов с основами наследования, инкапсуляции и полиморфизма в Python, научить их применять эти принципы для создания эффективных и гибких объектно-ориентированных программ. Задания: 1. Основы наследования 2. Множественное наследование 3. Инкапсуляция данных 4. Полиморфизм
13.	Лабораторная работа 13. Работа с модулями и пакетами в Python	Название: Работа с модулями и пакетами в Python Цель: познакомить студентов с системой модулей и пакетов в Python, научить их создавать, структурировать и использовать модули и пакеты для организации кода и повторного использования функционала. Задания: 1. Импорт стандартных модулей 2. Создание и использование пользовательского модуля 3. Структурирование модулей в пакет 4. Пространства имен и относительные импорты

6. Самостоятельная работа студентов по дисциплине

6.1. Подготовка рефератов

В рамках самостоятельной работы студент выполняет ряд работ по предложенным темам. Студент самостоятельно собирает необходимую для выполнения работ информацию, в ряде случаев дополняя ее своими обоснованными оценками и допущениями.

Темы рефератов:

1. История языка Python и его философия
2. Сравнение Python с другими языками программирования
3. Детальный анализ менеджера пакетов Python (pip)
4. Введение в виртуальные среды в Python
5. Модуль `datetime`: работа с датами и временем
6. Асинхронное программирование в Python с использованием asyncio
7. Итераторы и генераторы: подробное рассмотрение
8. Многопоточность и многопроцессорность в Python
9. Понимание и применение декораторов в Python
10. Основы машинного обучения с Python
11. Проекты с открытым исходным кодом на Python
12. Применение Python в научных исследованиях
13. Построение REST API с использованием Flask и Django
14. Обработка и анализ данных с помощью Pandas и NumPy
15. Веб-скрейпинг с использованием Beautiful Soup и Scrapy
16. Тестирование в Python: unittest, pytest и другие инструменты
17. Рефакторинг кода на Python: лучшие практики
18. Роль Python в кибербезопасности
19. Разработка игр на Python: примеры и библиотеки
20. Python во встроенных системах и IoT
21. Профилирование и оптимизация Python-программ
22. Системное администрирование с помощью Python
23. Искусственный интеллект и Python: текущее состояние и будущие тенденции
24. Python в финансовом моделировании
25. Python и квантовое программирование
26. Python и облачные технологии: AWS, Google Cloud и Azure
27. Создание и управление GUI в Python: Tkinter, PyQt

28. Python в анализе социальных медиа
29. Применение Python в графическом дизайне и анимации
30. Психология программирования: как Python помогает в обучении кодингу

7. Оценочные средства

7.1. Вопросы к первой рубежной аттестации(4-й семестр)

1. Объясните, что такое программирование и почему оно играет критически важную роль в современном мире.
2. Опишите, как развивался Python с момента его создания до наших дней, и укажите основные этапы его эволюции.
3. Объясните, как в Python используются переменные и какие правила следует соблюдать при их объявлении.
4. Сравните типы данных в Python (целые числа, вещественные числа, строки, булевы значения) с типами данных в других языках программирования.
5. Обсудите процесс и необходимость преобразования типов в Python, приведя примеры кода, где это преобразование критично.
6. Опишите, какие операции можно проводить над различными типами данных и как они влияют на результаты программы.
7. Подробно опишите, как работают условные конструкции в Python и в чем их важность при создании алгоритмов.
8. Разъясните использование циклов for и while в Python на примерах кода, объясняя, в каких случаях предпочтительнее использовать каждый из них.
9. Обсудите роль операторов break, continue и else в управлении циклами в Python, приведя примеры, когда их применение оправдано.
10. Опишите процесс создания и использования списков и кортежей в Python, указав ключевые различия и сценарии их применения.
11. Рассмотрите методы и функции для работы со списками в Python, объяснив, как каждый метод изменяет данные списка.
12. Объясните, как осуществляется индексация и работа со срезами в списках Python, и как эти возможности можно использовать для обработки данных.

Образец аттестационного билета

(за 1 рубежную аттестацию)

Грозненский государственный нефтяной технический университет им.акад. М.Д. Миллионщикова
Институт "цифровой экономики и технологического предпринимательства"
Группа "ПИ-24" Семестр "4"
Дисциплина "Разработка ПО на языке Python"
Билет № 1

1. Объясните, как в Python используются переменные и какие правила следует соблюдать при их объявлении.
2. Объясните, что такое программирование и почему оно играет критически важную роль в современном мире.

Подпись преподавателя_____ **Подпись заведующего кафедрой**_____

Вопросы к второй рубежной аттестации (4-й семестр)

1. Каковы основы определения и вызова функций в Python? Проиллюстрируйте свой ответ примерами, подчеркивая важность правильного использования функций в программировании.
2. Объясните, как в Python работают параметры и аргументы функций, включая различия между позиционными и ключевыми аргументами.
3. Расскажите о различных типах возвращаемых значений в функция, и как возвращаемые значения влияют на поток данных в программе.
4. Проанализируйте концепцию областей видимости переменных в Python, объясняя, как локальная и глобальная области видимости влияют на данные.
5. Изложите, как и почему используются анонимные функции (лямбды) в Python, приведя примеры их применения для решения программных задач.
6. Опишите процесс открытия и закрытия файлов в Python, подчеркивая важность правильного управления файловыми ресурсами.
7. Объясните методы чтения и записи данных в файлы, включая примеры кода, которые иллюстрируют эти процессы.
8. Рассмотрите использование менеджера контекста `with` при работе с файлами, объяснив, как он помогает управлять файловыми потоками и предотвращать ошибки.
9. Объясните, что такое итераторы в Python, и как они используются для управления потоками данных.
10. Расскажите, как создаются и используются генераторы в Python, включая примеры функций и выражений, которые позволяют генерировать последовательности данных.
11. Опишите преимущества использования генераторов в Python, включая сценарии, где их использование оптимально с точки зрения производительности и памяти.
12. Опишите основы синтаксиса регулярных выражений и как они применяются для обработки текстов в Python.
13. Проанализируйте функции модуля `re`, такие как поиск, замена и разделение строк, с примерами их использования.
14. Обсудите реальные сценарии использования регулярных выражений в программировании, демонстрируя, как они могут упростить и ускорить обработку данных.

Образец аттестационного билета

(за 2 рубежную аттестацию)

Грозненский государственный нефтяной технический университет им.акад. М.Д. Миллионщикова
Институт "цифровой экономики и технологического предпринимательства"
Группа "ПИ-24" Семестр "4"
Дисциплина "Разработка ПО на языке Python"
Билет № 1

1. Объясните, что такое итераторы в Python, и как они используются для управления потоками данных.
2. Изложите, как и почему используются анонимные функции (лямбды) в Python, приведя примеры их применения для решения программных задач.

Подпись преподавателя_____ **Подпись заведующего кафедрой**_____

Вопросы к первой рубежной аттестации(5-й семестр)

1. Опишите основные концепции объектно-ориентированного программирования и их значение в разработке современного программного обеспечения.
2. Как классы и объекты реализуются в Python? Приведите примеры создания классов и их использования через объекты.
3. Объясните, как работают конструкторы и деструкторы в классах Python, и какую роль они играют в управлении жизненным циклом объектов.
4. Расскажите о важности и применении атрибутов классов и экземпляров в Python, давая примеры из практической разработки.
5. Как геттеры и сеттеры используются для управления доступом к данным в Python? Проиллюстрируйте использование декоратора `property` для создания управляемых свойств.
6. Опишите различие и применение статических методов и методов класса. В чем их уникальность и когда они могут быть особенно полезны?
7. Как механизм наследования в Python помогает в повторном использовании кода? Приведите примеры с множественным наследованием.
8. Объясните, как инкапсуляция данных влияет на безопасность и стабильность программы. Приведите примеры правильного использования приватных и защищенных атрибутов.
9. Как реализуется полиморфизм в Python? Дайте примеры, показывающие, как один интерфейс может использоваться для различных типов данных.
10. В чем заключается польза и методика создания пакетов в Python? Объясните процесс структурирования больших проектов с помощью пакетов и модулей.
11. Какие проблемы могут возникнуть при работе с наследованием и как Python позволяет их решать? Обсудите конкретные случаи, связанные с разрешением методов в иерархиях классов.

12. Приведите примеры, как ООП применяется для решения типичных задач программирования. Какие паттерны проектирования особенно популярны в Python и почему?

Образец аттестационного билета

(за 1 рубежную аттестацию)

Грозненский государственный нефтяной технический университет им.акад. М.Д. Миллионщикова
Институт "цифровой экономики и технологического предпринимательства"
Группа "ПИ-24" Семестр "5"
Дисциплина "Разработка ПО на языке Python"
Билет № 1

1. Как классы и объекты реализуются в Python? Приведите примеры создания классов и их использования через объекты.
2. Как механизм наследования в Python помогает в повторном использовании кода? Приведите примеры с множественным наследованием.

Подпись преподавателя_____ **Подпись заведующего кафедрой**_____

Вопросы к второй рубежной аттестации (5-й семестр)

1. Опишите, как магические методы `__init__`, `__str__`, и `__repr__` используются в Python для управления поведением объектов при создании, преобразовании в строку и официальном представлении объектов. Приведите примеры, иллюстрирующие их использование.

2. Расскажите о роли магических методов для операций сравнения в Python, таких как `__eq__`, `__lt__`, и `__gt__`. Как эти методы влияют на функциональность классов?

3. Объясните, как магические методы используются для работы с контейнерами в Python. Приведите примеры методов `__len__` и `__getitem__` и объясните, как они влияют на поведение классов, реализующих контейнеры.

4. Обсудите принципы наследования в объектно-ориентированном программировании с примерами из Python. Как наследование способствует повторному использованию кода?

5. Разъясните, как реализовано множественное наследование в Python и какие вызовы могут возникнуть при его использовании. Приведите примеры кода.

6. Изложите концепции инкапсуляции данных в Python. Как можно защитить данные в классе от нежелательного доступа?

7. Объясните, как реализуется полиморфизм в Python. Приведите примеры, показывающие, как один интерфейс может использоваться для различных типов данных.

8. Опишите процесс импорта стандартных и пользовательских модулей в Python. Какие проблемы могут возникнуть при неправильном управлении импортами?

9. Расскажите о создании и структурировании модулей в Python. Как можно организовать код, чтобы улучшить его читаемость и поддерживаемость?

10. Обсудите, как пакеты и пространства имен помогают управлять большими программными проектами в Python. Приведите примеры, как структурирование кода в пакеты улучшает модульность и предотвращает конфликты имен.

11. Как магические методы влияют на поведение объектов при взаимодействии с Python системами, такими как циклы, функции и даже с самим интерпретатором Python? Приведите примеры использования магических методов в реальных приложениях.

12. Рассмотрите процесс решения конфликтов при множественном наследовании в Python, используя алгоритм C3 линеаризации. Как этот алгоритм помогает обеспечить консистентность и предсказуемость порядка разрешения методов?

13. Обсудите преимущества и недостатки использования полиморфизма в объектно-ориентированном программировании. Как полиморфизм может улучшить или усложнить код в больших программных проектах?

Какие практические меры можно предпринять для оптимизации структуры и производительности Python-программ, используя модули и пакеты? Приведите примеры, как организация кода в модули может помочь в оптимизации и упрощении сопровождения кода.

Образец аттестационного билета

(за 2 рубежную аттестацию)

Грозненский государственный нефтяной технический университет им.акад. М.Д. Миллионщикова
Институт "цифровой экономики и технологического предпринимательства"
Группа "ПИ-24" Семестр "5"
Дисциплина "Разработка ПО на языке Python"
Билет № 1

1. Изложите концепции инкапсуляции данных в Python. Как можно защитить данные в классе от нежелательного доступа?
2. Расскажите о создании и структурировании модулей в Python. Как можно организовать код, чтобы улучшить его читаемость и поддерживаемость?

Подпись преподавателя_____ **Подпись заведующего кафедрой**_____

7.2. Вопросы на зачет(4-й семестр)

1. Объясните, что такое программирование и почему оно играет критически важную роль в современном мире.
2. Опишите, как развивался Python с момента его создания до наших дней, и укажите основные этапы его эволюции.
3. Объясните, как в Python используются переменные и какие правила следует соблюдать при их объявлении.
4. Сравните типы данных в Python (целые числа, вещественные числа, строки, булевы значения) с типами данных в других языках программирования.
5. Обсудите процесс и необходимость преобразования типов в Python, приведя примеры кода, где это преобразование критично.
6. Опишите, какие операции можно проводить над различными типами данных и как они влияют на результаты программы.
7. Подробно опишите, как работают условные конструкции в Python и в чем их важность при создании алгоритмов.
8. Разъясните использование циклов for и while в Python на примерах кода, объясняя, в каких случаях предпочтительнее использовать каждый из них.
9. Обсудите роль операторов break, continue и else в управлении циклами в Python, приведя примеры, когда их применение оправдано.
10. Опишите процесс создания и использования списков и кортежей в Python, указав ключевые различия и сценарии их применения.
11. Рассмотрите методы и функции для работы со списками в Python, объяснив, как каждый метод изменяет данные списка.
12. Объясните, как осуществляется индексация и работа со срезами в списках Python, и как эти возможности можно использовать для обработки данных.
13. Каковы основы определения и вызова функций в Python? Проиллюстрируйте свой ответ примерами, подчеркивая важность правильного использования функций в программировании.
14. Объясните, как в Python работают параметры и аргументы функций, включая различия между позиционными и ключевыми аргументами.
15. Расскажите о различных типах возвращаемых значений в функция, и как возвращаемые значения влияют на поток данных в программе.
16. Проанализируйте концепцию областей видимости переменных в Python, объясняя, как локальная и глобальная области видимости влияют на данные.
17. Изложите, как и почему используются анонимные функции (лямбды) в Python, приведя примеры их применения для решения программных задач.

18. Опишите процесс открытия и закрытия файлов в Python, подчеркивая важность правильного управления файловыми ресурсами.

19. Объясните методы чтения и записи данных в файлы, включая примеры кода, которые иллюстрируют эти процессы.

20. Рассмотрите использование менеджера контекста with при работе с файлами, объяснив, как он помогает управлять файловыми потоками и предотвращать ошибки.

21. Объясните, что такое итераторы в Python, и как они используются для управления потоками данных.

22. Расскажите, как создаются и используются генераторы в Python, включая примеры функций и выражений, которые позволяют генерировать последовательности данных.

23. Опишите преимущества использования генераторов в Python, включая сценарии, где их использование оптимально с точки зрения производительности и памяти.

24. Опишите основы синтаксиса регулярных выражений и как они применяются для обработки текстов в Python.

25. Проанализируйте функции модуля re, такие как поиск, замена и разделение строк, с примерами их использования.

26. Обсудите реальные сценарии использования регулярных выражений в программировании, демонстрируя, как они могут упростить и ускорить обработку данных.

Образец билета на зачет

Грозненский государственный нефтяной технический университет им.акад. М.Д. Миллионщикова
Институт "цифровой экономики и технологического предпринимательства"
Группа "ПИ-24" Семестр "4"
Дисциплина "Разработка ПО на языке Python"
Билет № 1

1. Обсудите реальные сценарии использования регулярных выражений в программировании, демонстрируя, как они могут упростить и ускорить обработку данных.
2. Расскажите, как создаются и используются генераторы в Python, включая примеры функций и выражений, которые позволяют генерировать последовательности данных.

Подпись преподавателя_____ **Подпись заведующего кафедрой**_____

Вопросы на экзамен(5-й семестр

1. Опишите основные концепции объектно-ориентированного программирования и их значение в разработке современного программного обеспечения.

2. Как классы и объекты реализуются в Python? Приведите примеры создания классов и их использования через объекты.

3. Объясните, как работают конструкторы и деструкторы в классах Python, и какую роль они играют в управлении жизненным циклом объектов.

4. Расскажите о важности и применении атрибутов классов и экземпляров в Python, давая примеры из практической разработки.

5. Как геттеры и сеттеры используются для управления доступом к данным в Python? Проиллюстрируйте использование декоратора `property` для создания управляемых свойств.

6. Опишите различие и применение статических методов и методов класса. В чем их уникальность и когда они могут быть особенно полезны?

7. Как механизм наследования в Python помогает в повторном использовании кода? Приведите примеры с множественным наследованием.

8. Объясните, как инкапсуляция данных влияет на безопасность и стабильность программы. Приведите примеры правильного использования приватных и защищенных атрибутов.

9. Как реализуется полиморфизм в Python? Дайте примеры, показывающие, как один интерфейс может использоваться для различных типов данных.

10. В чем заключается польза и методика создания пакетов в Python? Объясните процесс структурирования больших проектов с помощью пакетов и модулей.

11. Какие проблемы могут возникнуть при работе с наследованием и как Python позволяет их решать? Обсудите конкретные случаи, связанные с разрешением методов в иерархиях классов.

12. Приведите примеры, как ООП применяется для решения типичных задач программирования. Какие паттерны проектирования особенно популярны в Python и почему?

13. Опишите, как магические методы `__init__`, `__str__`, и `__repr__` используются в Python для управления поведением объектов при создании, преобразовании в строку и официальном представлении объектов. Приведите примеры, иллюстрирующие их использование.

14. Расскажите о роли магических методов для операций сравнения в Python, таких как `__eq__`, `__lt__`, и `__gt__`. Как эти методы влияют на функциональность классов?

15. Объясните, как магические методы используются для работы с контейнерами в Python. Приведите примеры методов `__len__` и `__getitem__` и объясните, как они влияют на поведение классов, реализующих контейнеры.

16. Обсудите принципы наследования в объектно-ориентированном программировании с примерами из Python. Как наследование способствует повторному использованию кода?

17. Разъясните, как реализовано множественное наследование в Python и какие вызовы могут возникнуть при его использовании. Приведите примеры кода.

18. Изложите концепции инкапсуляции данных в Python. Как можно защитить данные в классе от нежелательного доступа?

19. Объясните, как реализуется полиморфизм в Python. Приведите примеры, показывающие, как один интерфейс может использоваться для различных типов данных.

20. Опишите процесс импорта стандартных и пользовательских модулей в Python. Какие проблемы могут возникнуть при неправильном управлении импортами?

21. Расскажите о создании и структурировании модулей в Python. Как можно организовать код, чтобы улучшить его читаемость и поддерживаемость?

22. Обсудите, как пакеты и пространства имен помогают управлять большими программными проектами в Python. Приведите примеры, как структурирование кода в пакеты улучшает модульность и предотвращает конфликты имен.

23. Как магические методы влияют на поведение объектов при взаимодействии с Python системами, такими как циклы, функции и даже с самим интерпретатором Python? Приведите примеры использования магических методов в реальных приложениях.

24. Рассмотрите процесс решения конфликтов при множественном наследовании в Python, используя алгоритм C3 линеаризации. Как этот алгоритм помогает обеспечить консистентность и предсказуемость порядка разрешения методов?

25. Обсудите преимущества и недостатки использования полиморфизма в объектно-ориентированном программировании. Как полиморфизм может улучшить или усложнить код в больших программных проектах?

26. Какие практические меры можно предпринять для оптимизации структуры и производительности Python-программ, используя модули и пакеты? Приведите примеры, как организация кода в модули может помочь в оптимизации и упрощении сопровождения кода.

Образец билета на экзамен

Грозненский государственный нефтяной технический университет им.акад. М.Д. Миллионщикова
Институт "цифровой экономики и технологического предпринимательства"
Группа "ПИ-24" Семестр "5"
Дисциплина "Разработка ПО на языке Python"
Билет № 1

1. Обсудите принципы наследования в объектно-ориентированном программировании с примерами из Python. Как наследование способствует повторному использованию кода?
2. Обсудите преимущества и недостатки использования полиморфизма в объектно-ориентированном программировании. Как полиморфизм может улучшить или усложнить код в больших программных проектах?

Подпись преподавателя_____ **Подпись заведующего кафедрой**_____

7.3. Текущий контроль

В качестве оценочных средств текущего контроля используются средства контроля выполнения и защиты лабораторных работ по дисциплине. Защита лабораторной работы – ответ на контрольные вопросы после выполнения лабораторной работы.

Лабораторная работа 1. Основы программирования на Python
Лабораторная работа 2. Работа с переменными и типами данных в Python
Лабораторная работа 3. Управление потоком программы в Python
Лабораторная работа 4. Изучение списков и кортежей в Python
Лабораторная работа 5. Основы работы с функциями в Python
Лабораторная работа 6. Управление файлами в Python
Лабораторная работа 7. Итераторы и генераторы в Python
Лабораторная работа 8. Освоение регулярных выражений в Python
Лабораторная работа 9. Основы объектно-ориентированного программирования в Python
Лабораторная работа 10. Изучение атрибутов, свойств и методов в объектно-ориентированном программировании Python
Лабораторная работа 11. Магические методы в Python

Лабораторная работа 12. Наследование, инкапсуляция и полиморфизм в Python

Лабораторная работа 13. Работа с модулями и пакетами в Python

Лабораторная работа 1.

Название: Основы программирования на Python

Цель: познакомить студентов с основами программирования и языком Python, научить устанавливать Python и запускать базовые программы.

Задания:

1. **Установка Python**
   - Установите Python последней версии с официального сайта.
   - Установите текстовый редактор или интегрированную среду разработки (IDE), например, Visual Studio Code или PyCharm.
2. **Настройка рабочей среды**
   - Настройте IDE для работы с Python.
   - Проверьте корректность установки, запустив интерпретатор Python из терминала или консоли.
3. **Запуск первой программы**
   - Создайте новый файл с расширением .py.
   - Напишите программу, которая выводит на экран фразу "Привет, мир!".
   - Запустите программу и убедитесь, что она корректно выполняет вывод в консоль.

Контрольные вопросы:

1. Что такое программирование и каковы его основные цели?
2. Какие особенности Python выделяют его среди других языков программирования?
3. Какие шаги нужно выполнить для установки Python на вашу операционную систему?
4. Какова роль текстового редактора или IDE в процессе программирования?
5. Какие ошибки могут возникнуть при запуске первой программы и как их исправить?

7.4. Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкалы оценивания.

Таблица 7

Планируемые результаты освоения компетенции	Критерии оценивания результатов обучения				Наименование оценочного средства
	менее 41 баллов (неудовлетворительно)	41–60 баллов (удовлетворительно)	61–80 баллов (хорошо)	81–100 баллов (отлично)	
ПК – 5. Способен разрабатывать и адаптировать прикладное программное обеспечение					
Знать методы и технологии проектирования и разработки прототипов информационных систем, включая управление базами данных и стандарты кодирования. Осведомлён о различных языках программирования и подходах к адаптации прикладного программного обеспечения под специфические требования проекта или организации.	Фрагментарные знания	Неполные знания	Сформированные, но содержащие отдельные пробелы знания	Сформированные систематические знания	*задания для лабораторной работы, билеты рубежных аттестаций, темы докладов*

Уметь проектировать и разрабатывать прототипы информационных систем, применять современные технологии для разработки и управления базами данных. Способен обеспечивать соответствие разработанного кода принятым стандартам и технологиям, а также адаптировать процессы кодирования для оптимизации производительности и соблюдения качества.	Фрагментарные знания	Неполные знания	Сформированные, но содержащие отдельные пробелы знания	Сформированные систематические знания	*задания для лабораторной работы, билеты рубежных аттестаций, темы докладов*
Владеть навыками эффективной разработки и адаптации прикладного программного обеспечения в соответствии с требованиями и стандартами проекта. Использует передовые методы программирования и управления данными для создания стабильных, эффективных и легко адаптируемых приложений.	Фрагментарные знания	Неполные знания	Сформированные, но содержащие отдельные пробелы знания	Сформированные систематические знания	*задания для лабораторной работы, билеты рубежных аттестаций, темы докладов*
ОПК-2. Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач					
Знать современные информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности.	Фрагментарные знания	Неполные знания	Сформированные, но содержащие отдельные пробелы знания	Сформированные систематические знания	*задания для лабораторной работы, билеты рубежных аттестаций, темы докладов*

Уметь применять современные информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности.	Фрагментарные знания	Неполные знания	Сформированные, но содержащие отдельные пробелы знания	Сформированные систематические знания	*задания для лабораторной работы, билеты рубежных аттестаций, темы докладов*
Владеть навыками взаимодействия с современными информационными технологиями и программными средствами, в том числе отечественного производства, при решении задач профессиональной деятельности.	Фрагментарные знания	Неполные знания	Сформированные, но содержащие отдельные пробелы знания	Сформированные систематические знания	*задания для лабораторной работы, билеты рубежных аттестаций, темы докладов*

8. Особенности реализации дисциплины для инвалидов и лиц с ограниченными возможностями здоровья

Для осуществления процедур текущего контроля успеваемости и промежуточной аттестации обучающихся созданы фонды оценочных средств, адаптированные для инвалидов и лиц с ограниченными возможностями здоровья и позволяющие оценить достижение ими запланированных в основной образовательной программе результатов обучения и уровень сформированности всех компетенций, заявленных в образовательной программе. Форма проведения текущей аттестации для студентов–инвалидов устанавливается с учетом индивидуальных психофизических особенностей (устно, письменно на бумаге, письменно на компьютере, в форме тестирования и т.п.). При тестировании для слабовидящих студентов используются фонды оценочных средств с укрупненным шрифтом.

На экзамен приглашается сопровождающий, который обеспечивает техническое сопровождение студенту. При необходимости студенту–инвалиду предоставляется дополнительное время для подготовки ответа на экзамене (или экзамене). Обучающиеся с ограниченными возможностями здоровья и обучающиеся инвалиды обеспечиваются печатными и электронными образовательными ресурсами (программы, учебные пособия для самостоятельной работы и т.д.) в формах, адаптированных к ограничениям их здоровья и восприятия информации:

для инвалидов и лиц с ограниченными возможностями здоровья по зрению:

– для слепых: задания для выполнения на семинарах и практических занятиях оформляются рельефно–точечным шрифтом Брайля или в виде электронного документа, доступного с помощью компьютера со специализированным программным обеспечением для слепых, либо зачитываются ассистентом; письменные задания выполняются на бумаге рельефно–точечным шрифтом Брайля или на компьютере со специализированным программным обеспечением для слепых либо надиктовываются ассистенту;

обучающимся для выполнения задания при необходимости предоставляется комплект письменных принадлежностей и бумага для письма рельефно–точечным шрифтом Брайля, компьютер со специализированным программным обеспечением для слепых;

– для слабовидящих: обеспечивается индивидуальное равномерное освещение не менее 300 люкс; обучающимся для выполнения задания при необходимости предоставляется увеличивающее устройство; возможно также использование собственных увеличивающих устройств; задания для выполнения заданий оформляются увеличенным шрифтом;

для инвалидов и лиц с ограниченными возможностями здоровья по слуху:

– для глухих и слабослышащих: обеспечивается наличие звукоусиливающей аппаратуры коллективного пользования, при необходимости обучающимся предоставляется звукоусиливающая аппаратура индивидуального пользования; предоставляются услуги сурдопереводчика;

– для слепоглухих допускается присутствие ассистента, оказывающего услуги тифлосурдопереводчика (помимо требований, выполняемых соответственно для слепых и глухих);

для лиц с тяжелыми нарушениями речи, глухих, слабослышащих лекции и семинары, проводимые в устной форме, проводятся в письменной форме;

для инвалидов и лиц с ограниченными возможностями здоровья, имеющих нарушения опорно–двигательного аппарата:

– для лиц с нарушениями опорно–двигательного аппарата, нарушениями двигательных функций верхних конечностей или отсутствием верхних конечностей: письменные задания выполняются на компьютере со специализированным программным обеспечением или надиктовываются ассистенту; выполнение заданий (тестов, контрольных работ), проводимые

в письменной форме, проводятся в устной форме путем опроса, беседы с обучающимся.

9. Учебно–методическое и информационное обеспечение дисциплины

9.1. Литература

1. Лутц М. "Изучаем Python", 5-е издание. – Санкт-Петербург: Символ-Плюс, 2019.

2. Саммерфильд М. "Программирование на Python 3". Подробное руководство. – Санкт-Петербург: Питер, 2016.

3. Россум Г. ван, Дрейк Ф.Л. "Язык программирования Python". – Москва: ДМК Пресс, 2011.

4. Маттезинос Дж. "Python. К вершинам мастерства", 2-е издание. – Москва: ДМК Пресс, 2020.

5. Харрисон П. "Python 3. Полное руководство по изучению языка с нуля до профессионала". – Москва: Эксмо, 2021.

6. Рамальо Л. "Грокаем алгоритмы. Адаптированное изложение классической книги по программированию". – Москва: Питер, 2019.

7. Шоу З.А. "Изучайте Python на примерах". – Москва: Эксмо, 2018.

8. Свейгарт А. "Автоматизируем скучную работу с помощью Python". – Москва: ДМК Пресс, 2020.

9. Бэз Д., Роббинс А. "Python. Подробный справочник". – Санкт-Петербург: ООО "Альфа-книга", 2019.

10. Флиган Д., Маккинли Б. "Python для сложных задач: наука о данных и машинное обучение". – Санкт-Петербург: Питер, 2020.

9.2. Методические указания для обучающихся по освоению дисциплины (приложение)

10. Материально–техническое обеспечение дисциплины

10.1. Материально–техническая база

Лекционная аудитория, оснащенная компьютером, видеопроекционным оборудованием, в том числе для презентаций, средствами звуковоспроизведения, экраном.

Мультимедийные средства и другая техника для презентаций учебного материала, офисный пакет программ MSWindows (MS Excel, MSWord) для оформления расчетов экономической эффективности информационных систем, OpenOfficeGoogleChrome.

10.2. Помещения для самостоятельной работы

Помещение для самостоятельной работы (Главный учебный корпус ФГБОУ ВО «Грозненский государственный нефтяной технический университет» 364902, Чеченская республика, г. Грозный, проспект им. Х.А. Исаева, 100. Аудитория оснащена необходимой компьютерной техникой, в наличии есть необходимое ПО: WinPro 10 RUS Uprgrd OLP NL Acdmc; OfficeStd RUS OLP NL Acdmc (право на использование согласно Контракту № 267–ЭА/19 от 15.09.2019 г.).

Система ГАРАНТ (пропиетарная лицензия) Visual Studio–(Freemium) 1С Предприятие договор от 02.12.2020 регистрационные номера продуктов (9334859; 9334952) Sublime Text– (открытый доступ) Notepad++ (открытый доступ).

Приложение

Методические указания по освоению дисциплины «Разработка ПО на языке Python»

1. Методические указания для обучающихся по планированию и организации времени, необходимого для освоения дисциплины.

Изучение рекомендуется начать с ознакомления с рабочей программой дисциплины, ее структурой и содержанием разделов (модулей), фондом оценочных средств, ознакомиться с учебно–методическим и информационным обеспечением дисциплины.

Дисциплина «Разработка ПО на языке Python» состоит из 9 связанных между собою тем, обеспечивающих последовательное изучение материала.

Обучение по дисциплине «Разработка ПО на языке Python» осуществляется в следующих формах:

1. Аудиторные занятия (лекции, практические занятия).
2. Самостоятельная работа студента (подготовка к лекциям, лабораторным занятиям, рефератам и иным формам письменных работ, индивидуальная консультация с преподавателем).
3. Интерактивные формы проведения занятий (лекция–дискуссия, групповое решение кейса и др. формы).

Учебный материал структурирован и изучение дисциплины производится в тематической последовательности. Каждому практическому занятию и самостоятельному изучению материала предшествует лекция по данной теме. Обучающиеся самостоятельно проводят предварительную подготовку к занятию, принимают активное и творческое участие в обсуждении теоретических вопросов, разборе проблемных ситуаций и поисков путей их решения. Многие проблемы, изучаемые в курсе, носят

дискуссионный характер, что предполагает интерактивный характер проведения занятий на конкретных примерах.

Описание последовательности действий обучающегося:

При изучении курса следует внимательно слушать и конспектировать материал, излагаемый на аудиторных занятиях. Для его понимания и качественного усвоения рекомендуется следующая последовательность действий:

1. После окончания учебных занятий для закрепления материала просмотреть и обдумать текст лекции, прослушанной сегодня, разобрать рассмотренные примеры (10 – 15 минут).
2. При подготовке к лекции следующего дня повторить текст предыдущей лекции, подумать о том, какая может быть следующая тема (10 – 15 минут).
3. В течение недели выбрать время для работы с литературой в библиотеке (по 1 часу).
4. При подготовке к практическому занятию повторить основные понятия по теме, изучить примеры. Решая конкретную ситуацию, – предварительно понять, какой теоретический материал нужно использовать. Наметить план решения, попробовать на его основе решить лабораторные задания.

2. Методические указания по работе обучающихся во время проведения лекций.

Лекции дают обучающимся систематизированные знания по дисциплине, концентрируют их внимание на наиболее сложных и важных вопросах. Лекции обычно излагаются в традиционном или в проблемном стиле. Для студентов в большинстве случаев в проблемном стиле. Проблемный стиль позволяет стимулировать активную познавательную деятельность обучающихся и их интерес к дисциплине, формировать творческое мышление, прибегать к

противопоставлениям и сравнениям, делать обобщения, активизировать внимание обучающихся путем постановки проблемных вопросов, поощрять дискуссию.

Во время лекционных занятий рекомендуется вести конспектирование учебного материала, обращать внимание на формулировки и категории, раскрывающие суть того или иного явления, или процессов, выводы и практические рекомендации.

Конспект лекции лучше подразделять на пункты, соблюдая красную строку. Этому в большой степени будут способствовать вопросы плана лекции, предложенные преподавателям. Следует обращать внимание на акценты, выводы, которые делает преподаватель, отмечая наиболее важные моменты в лекционном материале замечаниями

«важно», «хорошо запомнить» и т.п. Можно делать это и с помощью разноцветных маркеров или ручек, подчеркивая термины и определения.

Целесообразно разработать собственную систему сокращений, аббревиатур и символов. Однако при дальнейшей работе с конспектом символы лучше заменить обычными словами для быстрого зрительного восприятия текста.

Работая над конспектом лекций, необходимо использовать не только основную литературу, но и ту литературу, которую дополнительно рекомендовал преподаватель. Именно такая серьезная, кропотливая работа с лекционным материалом позволит глубоко овладеть теоретическим материалом.

Тематика лекций дается в рабочей программе дисциплины.

3. Методические указания обучающимся по подготовке к лабораторным/семинарским занятиям.

На практических занятиях приветствуется активное участие в обсуждении конкретных ситуаций, способность на основе полученных знаний находить наиболее эффективные решения поставленных проблем, уметь находить полезный дополнительный материал по тематике семинарских занятий.

Студенту рекомендуется следующая схема подготовки к семинарскому занятию:

1. Ознакомление с планом практического занятия, который отражает содержание предложенной темы;
2. Проработать конспект лекций;
3. Прочитать основную и дополнительную литературу.

В процессе подготовки к практическим занятиям, необходимо обратить особое внимание на самостоятельное изучение рекомендованной литературы. При всей полноте конспектирования лекции в ней невозможно изложить весь материал из–за лимита аудиторных часов. Поэтому самостоятельная работа с учебниками, учебными пособиями, научной, справочной литературой, материалами периодических изданий и Интернета является наиболее эффективным методом получения дополнительных знаний, позволяет значительно активизировать процесс овладения информацией, способствует более глубокому усвоению изучаемого материала, формирует у студентов отношение к конкретной проблеме. Все новые понятия по изучаемой теме необходимо выучить наизусть и внести в глоссарий, который целесообразно вести с самого начала изучения курса;

1. Ответить на вопросы плана практического занятия;
2. Выполнить домашнее задание;
3. Проработать тестовые задания и задачи;
4. При затруднениях сформулировать вопросы к преподавателю.

Результат такой работы должен проявиться в способности студента свободно ответить на теоретические вопросы практикума, выступать и участвовать в коллективном обсуждении вопросов изучаемой темы, правильно

выполнять практические задания и иные задания, которые даются в фонде оценочных средств дисциплины.

4. Методические указания обучающимся по организации самостоятельной работы.

Цель организации самостоятельной работы по дисциплине «Разработка ПО на языке Python» — это углубление и расширение знаний в области гуманитарных наук; формирование навыка и интереса к самостоятельной познавательной деятельности.

Самостоятельная работа обучающихся является важнейшим видом освоения содержания дисциплины, подготовки к практическим занятиям и к контрольной работе. Сюда же относятся и самостоятельное углубленное изучение тем дисциплины. Самостоятельная работа представляет собой постоянно действующую систему, основу образовательного процесса и носит исследовательский характер, что послужит в будущем основанием для написания выпускной квалификационной работы, практического применения полученных знаний.

Организация самостоятельной работы обучающихся ориентируется на активные методы овладения знаниями, развитие творческих способностей, переход от поточного к индивидуализированному обучению, с учетом потребностей и возможностей личности.

Правильная организация самостоятельных учебных занятий, их систематичность, целесообразное планирование рабочего времени позволяет студентам развивать умения и навыки в усвоении и систематизации приобретаемых знаний, обеспечивать высокий уровень успеваемости в период обучения, получить навыки повышения профессионального уровня.

Подготовка к лабораторному занятию включает, кроме проработки конспекта и презентации лекции, поиск литературы (по рекомендованным спискам и самостоятельно), решение задач из перечня лабораторных работ (текущий контроль). Лабораторное занятие – выполнение поставленных перед студентом задач с использованием ПК и специального программного обеспечения.

При подготовке к контрольной работе обучающийся должен повторять пройденный материал в строгом соответствии с учебной программой, используя конспект лекций и литературу, рекомендованную преподавателем. При необходимости можно обратиться за консультацией и методической помощью к преподавателю.

Самостоятельная работа реализуется:

– непосредственно в процессе аудиторных занятий – на лекциях, практических занятиях;

– в контакте с преподавателем вне рамок расписания – на консультациях по учебным вопросам, в ходе творческих контактов, при ликвидации задолженностей, при выполнении индивидуальных заданий и т.д.

– в библиотеке, дома, на кафедре при выполнении обучающимся учебных и практических задач.

Виды СРС и критерии оценок (по балльно–рейтинговой системе ГГНТУ, СРС оценивается в 15 баллов)

1. Реферат/Доклад/Презентация
2. Индивидуальное задание

Темы для самостоятельной работы прописаны в рабочей программе дисциплины. Эффективным средством осуществления обучающимся самостоятельной работы является электронная информационно–образовательная среда университета, которая обеспечивает доступ к учебным планам, рабочим программам дисциплин (модулей), практик, к изданиям электронных библиотечных систем.

Составитель:

Ассистент каф. «ИСЭ» /Ахматсулатнов А.Ю./

СОГЛАСОВАНО:

Зав. выпускающей каф. «ИСЭ» /Магомаева Л.Р./

Директор ДУМР /Магомаева М.А./