

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Кубанский государственный университет»

Факультет компьютерных технологий и прикладной математики
Кафедра вычислительных технологий

УТВЕРЖДАЮ

Проректор по учебной работе,
качеству образования – первый
проректор

Хагуров Т.А.

подпись

«_30» _мая_ 2025г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Б1.В.06 «ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ»

Направление
подготовки/специальность 02.03.02 Фундаментальная информатика и
информационные технологии
(код и наименование направления подготовки/специальности)

Направленность (профиль) /специализация
Математическое и программное обеспечение компьютерных технологий

Программа подготовки академический бакалавриат

Форма обучения очная

Квалификация выпускника бакалавр

Краснодар 2025

Рабочая программа дисциплины «ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования (ФГОС ВО) по направлению подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Программу составил(а):

Жук Арсений Сергеевич, ст. пр.

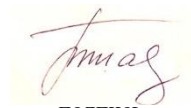
Ф.И.О. , должность, ученая степень, ученое звание



подпись

Приходько Татьяна Александровна, доцент, к. т. н.

Ф.И.О. , должность, ученая степень, ученое звание

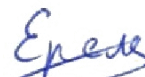


подпись

Рабочая программа дисциплины «ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ» утверждена на заседании кафедры Вычислительных технологий протокол № 7 «07» мая 2025 г.

И.о. заведующего кафедрой (разработчика) Еремин.А.А.

(фамилия, инициалы)



подпись

Утверждена на заседании учебно-методической комиссии факультета Компьютерных Технологий и Прикладной Математики протокол № 4 от «23» мая 2025г

Председатель УМК факультета

Коваленко А.В.

фамилия, инициалы



подпись

Рецензенты:

Гаркуша О.В., доцент кафедры информационных технологий ФБГОУ ВО «Кубанский государственный университет», кандидат физико-математических наук.

Схаляхо Ч.А., доцент КВВУ им.С.М.Штеменко, к.ф.-м.н., доцент

1. ЦЕЛИ И ЗАДАЧИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

1.1 Цель освоения дисциплины

Целью преподавания и изучения дисциплины «Паттерны проектирования» является изучение теоретических и практических основ разработки ООП приложений в рамках заданной архитектуры.

1.2 Задачи дисциплины

Результатом освоения дисциплины «ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ» является приобретение следующих знаний и умений:

студент должен **знать** теоретические принципы программирования в ООП парадигме, классификацию и базовые принципы применения паттернов проектирования, основные признаки плохого кода и методы рефакторинга, принципы реализации базовых архитектур ООП приложений в рамках заданных фреймворков, принципы SOLID ООП разработки; **уметь** создавать приложения в рамках архитектурного паттерна MVC и производных от него паттернов без использования фреймворков, проводить рефакторинг кода, составлять ООП архитектуру приложения, разбирать аспекты построения заданного архитектурного паттерна в рамках конкретного фреймворка на примере MVC паттерна в фреймворке ruby-on-rails, применять принципы SOLID ООП разработки там, где это необходимо; **владеть** методами рефакторинга, принципами построения ООП архитектуры без антипаттернов; паттернами проектирования и аспектами построения современных архитектурных паттернов.

1.3 Место дисциплины в структуре образовательной программы

Дисциплина «ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ» относится к вариативной части обязательных дисциплин учебного курса бакалавриата.

Для изучения дисциплины необходимо знание основ программирования, технологий баз данных, конструирования алгоритмов и структур данных, разработки в компьютерных сетях. Знания, получаемые при изучении дисциплины «ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ», используются при изучении других дисциплин учебного плана бакалавра (кроссплатформенное программирование, программирование для мобильных платформ, бек-разработка, распределенные задачи и алгоритмы, принципы командной разработки ПО), а также при работе над курсовыми работами и выпускной работой.

1.4 Перечень планируемых результатов обучения по дисциплине (модулю),

соотнесенных с планируемыми результатами освоения образовательной программы.

Изучение данной учебной дисциплины направлено на формирование у обучающихся следующими **профессиональными компетенциями и соотнесенные с ними индикаторы достижения компетенций:**

Код и наименование индикатора*	Результаты обучения по дисциплине (<i>знает, умеет, владеет (навыки и/или опыт деятельности)</i>)
ПК-1 Способен понимать и применять в научно-исследовательской и прикладной деятельности современный математический аппарат, основные законы естествознания, современные языки программирования и программное обеспечение; операционные системы и сетевые технологии	
ПК-1.1. Знает основы научно-исследовательской деятельности в области	принципы программирования в ООП парадигме, классификацию и базовые принципы применения

Код и наименование индикатора*	Результаты обучения по дисциплине (знает, умеет, владеет (навыки и/или опыт деятельности))
информационных технологий, имеет научные знания в теории информационных систем	паттернов проектирования, принципы реализации базовых архитектур ООП приложений в рамках заданных фреймворков, принципы SOLID ООП разработки
ПК-1.2. Умеет применять полученные знания в области фундаментальных научных основ теории информации и решать стандартные задачи в собственной научно-исследовательской деятельности	создавать приложения в рамках архитектурного паттерна MVC и производных от него паттернов без использования фреймворков, разбирать аспекты построения заданного архитектурного паттерна в рамках конкретного фреймворка на примере MVC паттерна в фреймворке ruby-on-rails, применять принципы SOLID ООП разработки там, где это необходимо
ПК-1.3. Имеет практический опыт научно-исследовательской деятельности в области информационных технологий	паттернами проектирования и аспектами построения современных архитектурных паттернов
УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	
УК-2.1. Знает необходимые для осуществления профессиональной деятельности правовые нормы	основные признаки плохого кода и методы рефакторинга
УК-2.2. Умеет определять круг задач в рамках избранных видов профессиональной деятельности, 46 планировать собственную деятельность исходя из имеющихся ресурсов; соотносить главное и второстепенное, решать поставленные задачи в рамках избранных видов профессиональной деятельности	проводить рефакторинг кода, составлять ООП архитектуру приложения
УК-2.3. Имеет практический опыт применения нормативной базы и решения задач в области избранных видов профессиональной деятельности	методами рефакторинга, принципами построения ООП архитектуры без антипаттернов

№ п.п.	Индекс компетенции	Содержание компетенции (или ее части)	В результате изучения учебной дисциплины обучающиеся должны		
			знать	уметь	владеть
1.	УК-2	Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся	основные признаки плохого кода и методы рефакторинга	проводить рефакторинг кода, составлять ООП архитектуру приложения	методами рефакторинга, принципами построения ООП архитектуры без антипаттернов

		ресурсов и ограничений			
2.	ПК-1	Способен понимать и применять в научно-исследовательской и прикладной деятельности современный математический аппарат, основные законы естествознания, современные языки программирования и программное обеспечение; операционные системы и сетевые технологии;	теоретические принципы программирования в ООП парадигме, классификацию и базовые принципы применения паттернов проектирования, принципы реализации базовых архитектур ООП приложений в рамках заданных фреймворков, принципы SOLID ООП разработки.	создавать приложения в рамках архитектурного паттерна MVC и производных от него паттернов без использования фреймворков, разбирать аспекты построения заданного архитектурного паттерна в рамках конкретного фреймворка на примере MVC паттерна в фреймворке ruby-on-rails, применять принципы SOLID ООП разработки там, где это необходимо	паттернами проектирования и аспектами построения современных архитектурных паттернов

2. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

2.1 Распределение трудоемкости дисциплины по видам работ

Общая трудоемкость дисциплины составляет 4 зач.ед. (144 часа), их распределение по видам работ представлено в таблице (для студентов ОФО)

Вид учебной работы	Всего часов	Семестры (часы)			
		4			
Контактная работа в том числе:	82.3	82.3			
Аудиторные занятия (всего):	68	68			
В том числе:					
Занятия лекционного типа	34	34			
Занятия семинарского типа (семинары, практ. занятия)					
Лабораторные занятия	34	34			
Иная контрольная работа					
Курсовое проектирование	10	10			
Контроль самостоятельной работы	4	4			

Промежуточная аттестация (ИКР)	0,3	0,3			
Самостоятельная работа (всего)	26	26			
В том числе:					
Курсовая работа					
<i>Проработка учебного (теоретического) материала</i>	10	10			
<i>Выполнение индивидуальных заданий (подготовка сообщений, презентаций)</i>	10	10			
<i>Реферат</i>	-	-			
<i>Подготовка к текущему контролю</i>	6	6			
Контроль:					
Подготовка к экзамену:	35,7	35,7			
Общая трудоемкость в т.ч. контактная работа зач. ед.	144	144			
	82,3	82,3			
	4	4			

2.2 Структура дисциплины:

Распределение видов учебной работы и их трудоемкости по разделам дисциплины. Разделы дисциплины, изучаемые в _5_ семестре (очная форма)

№	Наименование разделов	Количество часов				
		Всего	Аудиторная работа			Внеаудиторная работа
			Л	КСР	ЛР	СРС
1	2	3	4	5	6	7
1	Введение в ООП разработку на языке Ruby	20	6		10	4
2	Базовые паттерны проектирования.	18	4	2	6	6
3	Построение ООП приложений.	24	2	12	6	4
4	Рефакторинг.	14	6		4	4
5	Архитектурные паттерны и принципы SOLID	14	4		6	4
6	Ruby-on-Rails.	18	12		2	4
	Итого по разделам дисциплины	108	34	14	34	26
	Подготовка к экзамену	35,7				
	ИКР	0,3				
	<i>Итого по дисциплине:</i>	144				

2.3 Содержание разделов дисциплины:

2.3.1 Занятия лекционного типа

№ раздела	Наименование раздела	Содержание раздела	Форма текущего контроля
1	2	3	4

1	Введение в ООП разработку на языке Ruby	ООП, инкапсуляция, наследование, полиморфизм, методы объекта, методы класса, идентификаторы доступа, абстрактные методы, циклы в руби. Список в руби, примеси и модули в руби, enumerable, comparable, функциональные интерфейсы в руби, наследование, создание объекта, методы рефакторинга, антипаттерны повтор кода, разные именования методов, Общие принципы паттернов, классификация, паттерн шаблонный метод, паттерн итератор. Подходы в сравнении объектов на равенство и на больше-меньше в руби	ЛР, контрольные вопросы
2	Базовые паттерны проектирования	Работа с БД в руби, паттерны стратегия, одиночка, наблюдатель, фабрика, паттерн архитектурные, MVC, связь архитектурных паттернов, паттернов и фреймворков. Диаграмма классов и обозначения на ней. Диаграмма последовательностей для обозначения взаимодействия объектов. Диаграмма классов и обозначения на ней. Диаграмма последовательностей для обозначения взаимодействия объектов	-//-
3	Построение ООП приложений.	Понятия UNI-тестов, гем пакеты и бандл, автоматизированное создание документации, логгирование событий в руби.	-//-
4	Рефакторинг	Методы рефакторинга, антипаттерны: Неуместная близость, завистливые функции, классы данных, одержимость элементарными типами, ленивый класс. антипаттерны: дублирование кода, длинный метод, длинный класс, длинный список параметров, повторы условных операторов, альтернативные классы с разными интерфейсами. стрела дробью, расходящиеся модификации, посредник, теоретическая общность, параллельные иерархии наследования. Паттерны стратегия, одиночка, фабрика, строитель, мост, команда, адаптер.	-//-
5	Архитектурные паттерны и принципы SOLID.	Принципы SOLID. MVC. Расскажите базовый подход. Реализации с активным и пассивным контроллером. Выполнение операций CRUD и архитектура классов в рамках данного паттерна. Проблематика в этих реализациях. Паттерны MVP и MVVM.	-//-
6	Ruby-on-Rails.	Структура проекта, гем пакеты, модели, контроллеры и представления. Валидация, логгирование. Построение приложения	Экзаменационный проект

2.3.2. Занятия семинарского типа.

Занятия семинарского типа – не предусмотрены.

2.3.3. Лабораторные занятия

№ работы	№ раздела дисциплины	Наименование лабораторных работ	Форма текущего контроля
1.	1	Введение в язык руби. Циклы и алгоритмы. ООП. Написание класса по ТЗ.	ЛР1
2.	1	Списки. Методы работы со списками. Методы примеси enumerable. Функции высших порядков в руби и функциональные интерфейсы. Примеси. Использование примесей comparable и enumerable. Сравнение объектов на равенство и неравенство. Паттерн итератор	ЛР2
3.	1	Рефакторинг класса. Наследование, как метод рефакторинга. Классы данных. Наследование и паттерн шаблонный метод.	ЛР3
4-5.	1	Защита лаб.	Гит студента
6.	2	Работа с хранилищами данных – БД и файлы JSON, YAML. Понятия UNI-тестов, гем пакеты и бандл, автоматизированное создание документации, логгирование событий в руби	ЛР4
7.	2	MVC реализация приложения с одной сущностью, операции CRUD, filter, sort.	ЛР5
8.	2	Защита лаб.	Гит студента
9.	3	Доработка MVC приложения. Архитектура приложения с двумя сущностями	ЛР6
10.	3	Доработка MVC приложения. Архитектура приложения с ООП архитектурой предметной области	ЛР7
11.	3	Защита лаб.	Гит студента
12.	4	Рефакторинг классов работы с хранилищами данных. Наследование и ассоциация. Паттерны стратегия, одиночка, адаптер.	ЛР8
13.	4	Защита лаб.	Гит студента
14.	5	Разные реализации MVC для одного приложения с одной сущностью.	ЛР9
15.	5.	MVP, построение общего гема для работы с двумя разными UI – веб и оконного приложения.	ЛР10
16	5	Защита лаб.	Гит студента
17	6	Постановка задачи на экзаменационный проект	

2.3.4. Примерная тематика курсовых работ (проектов)

Учебным планом не предусмотрены.

2.3.5. Расчетно-графические задания

Учебным планом не предусмотрены.

2.4. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине

№	Вид СРС	Перечень учебно-методического обеспечения дисциплины по выполнению самостоятельной работы
1	2	3
1.	Введение в ООП разработку на языке Ruby	Список основной и дополнительной литературы
2.	Базовые паттерны проектирования.	Список основной и дополнительной литературы
3.	Построение ООП приложений.	Список основной и дополнительной литературы
4.	Рефакторинг.	Список основной и дополнительной литературы
5.	Архитектурные паттерны и принципы SOLID	Список основной и дополнительной литературы
6.	Ruby-on-Rails.	Список основной и дополнительной литературы

Учебно-методические материалы для самостоятельной работы обучающихся из числа инвалидов и лиц с ограниченными возможностями здоровья (ОВЗ) предоставляются в формах, адаптированных к ограничениям их здоровья и восприятия информации:

Для лиц с нарушениями зрения:

- в печатной форме увеличенным шрифтом,
- в форме электронного документа, Для лиц с нарушениями слуха:
- в печатной форме, в форме электронного документа.

Для лиц с нарушениями опорно-двигательного аппарата:

- в печатной форме, в форме электронного документа,

Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

3. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

Семестр	Вид занятия (Л, ПР, ЛР)	Используемые интерактивные образовательные технологии	Количество часов
5	Л	Компьютерные презентации и обсуждение	34
	ЛР	Разбор конкретных ситуаций (задач), тренинги по решению задач	34
Итого:			68

4. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

4.1 Фонд оценочных средств для проведения текущего контроля

№ п/п	Код и наименование индикатора	Результаты обучения	Наименование оценочного средства	
			Текущий	Промежуточная

			контроль	аттестация
1	ПК-1.1. Знает основы научно-исследовательской деятельности в области информационных технологий, имеет научные знания в теории информационных систем	принципы программирования в ООП парадигме, классификацию и базовые принципы применения паттернов проектирования, принципы реализации базовых архитектур ООП приложений в рамках заданных фреймворков, принципы SOLID ООП разработки	Опрос по теме ЛР	Вопросы 1-27
2	ПК-1.2. Умеет применять полученные знания в области фундаментальных научных основ теории информации и решать стандартные задачи в собственной научно-исследовательской деятельности	создавать приложения в рамках архитектурного паттерна MVC и производных от него паттернов без использования фреймворков, разбирать аспекты построения заданного архитектурного паттерна в рамках конкретного фреймворка на примере MVC паттерна в фреймворке ruby-on-rails, применять принципы SOLID ООП разработки там, где это необходимо	Опрос по теме ЛР	Вопросы 1-27
3	ПК-1.3. Имеет практический опыт научно-исследовательской деятельности в области информационных технологий	паттернами проектирования и аспектами построения современных архитектурных паттернов	Опрос по теме ЛР	Вопросы 1-27
4	УК-2.1. Знает необходимые для осуществления профессиональной деятельности правовые нормы	основные признаки плохого кода и методы рефакторинга	Опрос по теме ЛР	Вопросы 1-27
5	УК-2.2. Умеет определять круг задач в рамках избранных видов профессиональной деятельности, 4б планировать собственную деятельность исходя из имеющихся ресурсов; соотносить главное и второстепенное, решать поставленные задачи в рамках избранных видов профессиональной деятельности	проводить рефакторинг кода, составлять ООП архитектуру приложения	Опрос по теме ЛР	Вопросы 1-27
6	УК-2.3. Имеет практический опыт применения нормативной базы и решения задач в	методами рефакторинга, принципами построения ООП архитектуры без антипаттернов	Опрос по теме ЛР	Вопросы 1-27

	области избранных видов профессиональной деятельности			
--	---	--	--	--

Фонд оценочных средств дисциплины состоит из средств текущего контроля (вопросы при защите ЛР) лабораторных работ, средств итоговой аттестации (экзамен в 5 семестре).

Оценка успеваемости осуществляется по результатам:

- выполнения лабораторных работ;
- разработки экзаменационного проекта;
- ответа на вопросы в билете.

Перечень вопросов, которые выносятся на экзамен

1. Распишите основные принципы ООП. Приведите пример работающего кода для иллюстрации выполнения или невыполнения этих принципов.
2. Какие отношения между классами могут быть? Опишите, что они из себя представляют в предметной области, в коде и на диаграмме классов. Покажите пример кода с реализацией этих отношений.
3. Расскажите, что такое ассоциация между классами, приведите частные случаи, покажите примеры. Расскажите, какие паттерны проектирования используют данное отношение. Покажите примеры работающего кода с реализацией данного отношения.
4. Расскажите, как Вы понимаете принцип - предпочитайте делегацию наследованию. Покажите примеры кода, покажите, в каком случае имеет смысл использовать любое из отношений. Расскажите, какие паттерны иллюстрируют данный подход.
5. Расскажите, какие Вам известны способы создания экземпляров классов, покажите примеры некачественного и качественного подхода(работающий код). Расскажите, какие паттерны проектирования, описывающие эти подходы, Вам известны.
6. Расскажите ООП структуру Вашего языка программирования. Расскажите, какие методы принято переопределять и в каком случае.
7. Какие паттерны проектирования Вам известны, что это такое, как их принято классифицировать? Раскройте взаимосвязь паттернов проектирования, архитектурных паттернов и фреймворков. Покажите на примерах.
8. Расскажите, как составлять диаграмму классов, что на ней принято отмечать и каким образом. Раскройте все составляющие, покажите взаимосвязь диаграммы классов и кода.
9. Какие структурные паттерны проектирования Вам известны? Перечислите и расскажите базовые принципы всех паттернов. Раскройте подробно два паттерна, покажите проблему, покажите предлагаемое решение на диаграммах и в коде. Укажите связь этих паттернов с другими известными Вам паттернами.
10. Какие порождающие паттерны проектирования Вам известны? Перечислите и расскажите базовые принципы всех паттернов. Раскройте подробно два паттерна, покажите проблему, покажите предлагаемое решение на диаграммах и в коде. Укажите связь этих паттернов с другими известными Вам паттернами.
11. Какие поведенческие паттерны проектирования Вам известны? Перечислите и расскажите базовые принципы всех паттернов. Раскройте подробно два поведенческих

- паттерна, покажите проблему, покажите предлагаемое решение на диаграммах и в коде. Укажите связь этих паттернов с другими известными Вам паттернами.
12. Расскажите про подходы в сравнении объектов на равенство и на больше-меньше в руби. Расскажите, какие принципы ООП и паттерны проектирования используются в реализации данных подходов.
13. Расскажите, как иллюстрируется взаимодействие объектов на диаграмме последовательности. Покажите тривиальный пример, далее покажите пример диаграммы последовательностей, иллюстрирующий взаимодействие объектов в рамках любого архитектурного паттерна.
14. Архитектурный паттерн MVC. Расскажите базовый подход, расскажите, какие паттерны проектирования используются при его реализации. Покажите взаимодействие объектов в рамках реализации некоторых операций CRUD на диаграмме последовательностей. Покажите, в каких фреймворках он применяется.
15. Архитектурный паттерн MVP. Расскажите проблематику в двух подходах в MVC, которая привела к появлению данного архитектурного паттерна. Расскажите базовый подход. Покажите взаимодействие объектов в рамках реализации некоторых операций CRUD на диаграмме последовательностей. Покажите, в каких фреймворках он применяется.
16. Архитектурный паттерн MVVM. Расскажите проблематику, которая привела к появлению данного архитектурного паттерна. Расскажите базовый подход. Покажите взаимодействие объектов в рамках реализации некоторых операций CRUD на диаграмме последовательностей. Покажите, в каких фреймворках он применяется.
17. Архитектурные паттерны. Расскажите о всех архитектурных паттернах, которые Вам известны, базовые подходы, в каких языках и фреймворках применяются. Раскройте паттерн, применённый в фреймворке Вашей задачи. Покажите взаимодействие объектов в рамках реализации некоторых операций CRUD на диаграмме последовательностей.
18. Что такое рефакторинг? Когда его требуется применять? Приведите примеры методов рефакторинга и примеры антипаттернов, которые данные методы позволяют исправить.
19. Раскройте антипаттерны: Неуместная близость, завистливые функции, классы данных, одержимость элементарными типами, ленивый класс. Раскройте почему они считаются антипаттернами, приведите примеры кода, покажите методы рефакторинга, позволяющие их исправить. Укажите паттерны разработки, позволяющие не допускать таких ошибок.
20. Раскройте антипаттерны: дублирование кода, длинный метод, длинный класс, длинный список параметров, повторы условных операторов, альтернативные классы с разными интерфейсами. Раскройте почему они считаются антипаттернами, приведите примеры кода, покажите методы рефакторинга, позволяющие их исправить. Укажите паттерны разработки, позволяющие не допускать таких ошибок.
21. Раскройте антипаттерны: стрела дробью, расходящиеся модификации, посредник, теоретическая общность, параллельные иерархии наследования. Покажите, как они взаимосвязаны. Раскройте почему они считаются антипаттернами, приведите примеры кода, покажите методы рефакторинга, позволяющие их исправить. Укажите паттерны разработки, позволяющие не допускать таких ошибок, или приводящие к ним.
22. Раскройте принцип единой ответственности. Покажите примеры антипаттернов его нарушающих, паттернов, его реализующих. Покажите на работающих примерах.

23. Раскройте принцип открытости - закрытости. Покажите примеры антипаттернов его нарушающих, паттернов, его реализующих. Покажите на работающих примерах.
24. Раскройте принцип подстановки Барбары Лисков. Покажите примеры антипаттернов его нарушающих, паттернов, его реализующих. Покажите на работающих примерах.
25. Раскройте принцип подстановки разделения интерфейсов. Покажите примеры антипаттернов его нарушающих, паттернов, его реализующих. Покажите на работающих примерах.
26. Раскройте принцип инверсии зависимостей. Покажите примеры антипаттернов его нарушающих, паттернов, его реализующих. Покажите на работающих примерах.
27. Принципы SOLID. Зачем нужны, о чём они, как взаимосвязаны с паттернами и антипаттернами? Покажите на примерах. Как реализуются в Вашей задаче и в выбранном Вами фреймворке..

Примеры лабораторных работ.

Лабораторная работа № 3. Наследование

Цель – изучить основные принципы применения наследования, отработать методы рефакторинга кода.

Для достижения указанной цели ЗАПРЕЩЕНО пользоваться AI генераторами кода. Важно проделать самостоятельно рефакторинг. В дальнейшем в некоторых лабораторных работах использование AI генераторов может быть разрешено.

Каждое задание должно быть загружено на личный git-репозиторий отдельным коммитом. Коммиты должны иметь осмысленное название, при этом в названии содержать шифр лабы и задачи. Лабораторная работа выполняется в одной папке. Защита работы возможна на любой лабораторной работе от 5 до 16. Файлы должны иметь названия, соответствующие содержанию, а не нумерации заданий.

Если часть задач выполнена в один коммит, работа не проверяется. Если все коммиты сделаны в один час, работа не проверяется.

При защите обязательно написать требуемые преподавателем методы рефакторинга на листочке без использования доп материалов (не больше 3-5 минут на метод). При защите обязательно объяснение любых из использованных инструментов и принципов их работы. При защите обязательны ответы на любые вопросы по темам работы. Примеры формулировок приведены в конце работы, точные формулировки вопросов могут отличаться. При защите обязательна диаграмма классов и корректное отображение на ней базовых аспектов ООП.

При защите лабораторной работы в зависимости от уровня подготовки студента может быть необходимо доучить часть материала или дописать часть программы прямо на лабораторной работе для окончательной защиты. Если же обучающийся плохо готов: не отвечает на теоретические вопросы или не пишет код перед преподавателем или не владеет обозначениями основных аспектов ООП на диаграмме классов в своём решении, то обучающийся будет защищаться уже на следующей лабораторной работе.

Задание 1. Рефакторинг: повтор кода внутри класса.

Провести рефакторинг класса Student, убрав повтор кода везде внутри одного класса. Рефакторинг проводить в несколько этапов, каждый этап в отдельный коммит. Объяснить в описании коммита, какими методами рефакторинга и как воспользовались.

Задание 2. Рефакторинг: повтор кода внутри класса.

Провести рефакторинг класса StudentShort, убрав повтор кода везде внутри одного класса. Рефакторинг проводить в несколько этапов, каждый этап в отдельный коммит. Объяснить в описании коммита, какими методами рефакторинга и как воспользовались.

Задание 3. Наследование, как метод рефакторинга.

Составить диаграмму классов для классов Student StudentShort. Отметить ВСЕ методы и поля объекта и класса.

Провести рефакторинг, убрав следующие признаки плохого кода: методы с разным названием, повтор методов в разных классах, неиспользуемые методы.

Рефакторинг проводить в несколько этапов, каждый этап в отдельный коммит. Объяснить в описании коммита, какими методами рефакторинга и как воспользовались. На каждый этап – новая диаграмма классов.

Задание 4. Классы модели.

1. Создать класс Data_table. Данный класс имеет поле – двумерный массив с элементами любого типа, поле недоступно ни для чтения, ни для записи. Строка в таблице – значения атрибутов сущности. Запись данных только с помощью конструктора.
2. Реализовать методы, позволяющие – получить элемент по номеру строки и столбца БЕЗ возможности его редактировать, получить количество столбцов в таблице, получить количество строк в таблице.
3. Создать класс Data_list, содержащий упорядоченный массив элементов любого класса. Любые изменения данных недоступны, кроме одного метода который, будет обговорен будет позже.
4. Данный класс должен реализовывать следующий методы:
 - a. select(number) Выделить элемент по номеру
 - b. get_selected Получить массив id выделенных элементов(предполагается, что у хранимых объектов есть метод получения ID)
 - c. get_names Получить массив наименований атрибутов, кроме ID (0 столбец имеет название «№ по порядку»). данный метод не будет работать для указанного класса, так как не имеет информации об объектах внутри, реализовать этот класс необходимо в наследниках.

- d. `get_data`: Получить объект класса `Data_table`, где нулевой столбец – сгенерированный № по порядку, остальные столбцы заполнены ВСЕМИ атрибутами сущности, кроме ID, данный метод не будет работать для указанного класса, так как не имеет информации об объектах внутри, реализовать этот класс необходимо в наследниках.
- e. `clear_selected`: очистить всё выделение
- f. И никаких больше публичных методов

5. Создать наследника `Data_list_student_short`, реализующего методы `get_data` и `get_names` для класса `Student_short` и никаких больше публичных методов.

Задание 5. Паттерн Шаблонный метод. (зачтено)

- 1. Разобрать на простейшем примере паттерн Шаблон(Паттерн Паттерн) и разницу между использованием методов суперкласса в наследовании и шаблонным методом.
- 2. Использовать данный паттерн в методах `get_data` и `get_names`, реализовать дополнительные `private` методы в наследнике.
- 3. Протестировать на примере формирования сущностей класса `Student_short`. Важно, конструктор должен быть независимым от класса.
- 4. Реализовать сеттер для массива объектов, заменяющий массив объектов, при этом массив наименований столбцов НЕ МЕНЯЕТСЯ никогда, модифицировать конструкторы.
- 5. Отметить указанные классы в диаграмме классов

Лабораторная работа № 4. Классы модели

Цель – отработать принципы построения класса модели и аспекты реализации для разных хранилищ данных.

В рамках цели разрабатываются классы для работы с 5 разными хранилищами данных. Для достижения указанной цели необходимо самостоятельно реализовать класс для любого из хранилищ из заданий 1 или 2, а для остальных хранилищ воспользоваться AI генераторами кода. Обязательно указать, например, для СУБД Postgre написал класс самостоятельно, для остальных типов хранилищ воспользовался генератором. Важно понимать и ЗАПУСКАТЬ все реализованные методы. Важно реализовывать ровно указанные публичные методы и никаких больше. Важно понимать особенности реализации работы с разными хранилищами данных.

Каждое задание должно быть загружено на личный git-репозиторий отдельным коммитом. Коммиты должны иметь осмысленное название, при этом в названии содержать шифр лабы и задачи. Лабораторная работа выполняется в одной папке. Защита работы возможна на любой лабораторной работе от 6 до 16. Файлы должны иметь названия, соответствующие содержанию, а не нумерации заданий.

Если часть задач выполнена в один коммит, работа не проверяется. Если все коммиты сделаны в один час, работа не проверяется.

При защите обязательно писать скрипты работы с написанными классами по заданию преподавателя. На защите обязательно описать инструменты языка, позволяющие выполнять генерацию документации, публикацию гем-пакетов и логгирование событий программы.

При защите лабораторной работы в зависимости от уровня подготовки студента может быть необходимо дописать часть кода, тогда выставляется оценка ниже оценки 5. При защите может возникнуть ситуация, при которой студент не понимает, как пользоваться написанным инструментарием, в этом случае защита завершена, возобновится после полноценной подготовки студента и САМОСТОЯТЕЛЬНЫМ написанием всех методов класса заново.

(Зачтено) – любой из классов для работы с БД с логгированием и скриптами для установки ПО и гем-пакетов.

Задание 1. Students_list_JSON, Students_list_YAML

1. Реализовать класс Students_list_JSON, который будет работать с текстовым файлом, содержащим списки студентов. Для этого необходимо обеспечить выполнение следующих функций:
 - a. Чтение всех значений из файла;
 - b. Запись всех значений в файл;
 - c. Получить объект класса Student по ID
 - d. get_k_n_student_short_list Получить список k по счету n объектов класса Student_short (например, вторые 20 элементов, чтобы в дальнейшем можно было листать длинный список), результат вернуть в формате Data_list. Учесть в виде необязательного аргумента существующий объект класса data_list, вернуть или существующий, или измененный объект.
 - e. Сортировать элементы по набору ФамилияИнициалы.
 - f. Добавить объект класса Student в список (при добавлении сформировать новый ID).
 - g. Заменить элемент списка по ID.
 - h. Удалить элемент списка по ID.
 - i. get_student_short_count Получить количество элементов
2. Реализовать класс Students_list_YAML, который будет работать с текстовым файлом, содержащим списки студентов. Для этого необходимо обеспечить выполнение следующих функций:
 - a. Чтение всех значений из файла;
 - b. Запись всех значений в файл;
 - c. Получить объект класса Student по ID
 - d. get_k_n_student_short_list Получить список k по счету n объектов класса Student_short (например, вторые 20 элементов, чтобы дальше можно было листать длинный список), результат вернуть в формате Data_list.

- e. Сортировать элементы по набору ФамилияИнициалы.
 - f. Добавить объект класса Student в список (при добавлении сформировать новый ID).
 - g. Заменить элемент списка по ID.
 - h. Удалить элемент списка по ID.
 - i. `get_student_short_count` Получить количество элементов
3. Для класса JSON сгенерировать официальную документацию.
 4. Класс YAML опубликовать в виде гем пакета. Продемонстрировать его использование в отдельном скрипте.

Задание 2 Подключение БД к модели. Работа осуществляется с 3 различными БД на выбор студентом.

1. Создать подключение.
2. Создать таблицу student. Для этого создать отдельную директорию, в которой будут храниться файлы, реализующие изменения в структуре БД. Создать два типа файлов, в одном – изменения в структуре, в другом – автоматические скрипты заполнения данными. Сохранить выполненные Вами скрипты в гит репозиторий.
3. Заполнить таблицу данными.
4. Протестируйте выполнение `select` запроса из этой таблицы из программы на ruby.
5. Если не было сделано в предыдущем задании, то модифицируйте конструкторы или создайте альтернативные конструкторы для класса Student так, чтобы они принимали в качестве аргумента хеш, протестируйте написанные изменения.
6. Если необходимо, внесите изменения в уже существующие классы, чтобы весь написанный функционал продолжал работать.
7. Реализовать класс `Students_list_DB_1`(или 2 или 3 для каждой новой БД), который будет работать с созданной БД и таблицей, содержащей списки студентов. Для этого необходимо обеспечить выполнение следующих функций:
 - a. Получить объект класса Student по ID
 - b. `get_k_n_student_short_list` Получить список k по счету n объектов класса Student_short (например, вторые 20 элементов, чтобы в дальнейшем можно было листать длинный список), результат вернуть в формате Data_list.
 - c. Добавить объект класса Student в список (при добавлении сформировать новый ID).
 - d. Заменить элемент списка по ID.
 - e. Удалить элемент списка по ID.
 - f. Получить количество элементов
8. Подключить логгирование для каждого из трёх классов. Логги писать в отдельный файл, реализовать три разных уровня логгирования – от записи всех действий до записи исключительных ситуаций и ошибок.

Задание 3. Сформируйте отдельный скрипт, чтобы установить всё необходимое на Вашу ОС, и отдельный скрипт для установки всех необходимых гем-пакетов.

Лабораторная работа № 8. Архитектура классов работы с хранилищами данных

Каждое задание должно быть загружено на личный git-репозиторий отдельным коммитом. Лабораторная работа выполняется в одной папке. Защита работы возможна на любой лабораторной работе от 1 до 16. Каждое из 6 заданий проверяется отдельно с учетом вопросов преподавателя. Задание засчитывается отдельно, лабораторная работа зачтена в случае выполнения всех 6 заданий.

Если часть задач выполнена в один коммит, работа не проверяется. Если все коммиты сделаны в один час, работа не проверяется.

Выполняется в отдельной ветке после ЛР5.

Необходимо отметить на диаграмме классов все 5 классов из ЛР4. Далее необходимо руководствуясь указанными ниже рекомендациями объединить эти классы в общую архитектуру.

Каждый этап рефакторинга – отдельный коммит с объяснениями и новой диаграммой классов.

Задание.

1. Работу с файлами объединить через наследование. **(зачтено)**
2. Заменить наследование на стратегию.
3. Выделить классы одиночки для работы с каждой базой. **(зачтено)**
4. Классы по работе с БД объединить в иерархию наследования.
5. Если необходимо, внести изменения в одиночку.
6. Построить адаптер, чтобы класс по работе с файлами вёл себя так же, как и файл по работе с БД. Важно вносить изменения в класс по работе с файлами в этом пункте ЗАПРЕЩАЕТСЯ.
7. Должен получиться общий итоговый класс StudentList. Построить и зафиксировать обновленную диаграмму классов.
8. Проверить работу MVC приложения на каждом из 5 хранилищ данных.

Критерии оценивания к экзамену

Оценка **“отлично”** - Практические задания выполнены в срок в объеме не менее 90%. Студент разработал ООП архитектуру предметной области. Студент разобрал архитектурные паттерны для решения одной задачи и сравнил результаты. Студент демонстрирует правильные, уверенные действия по применению полученных знаний на практике, грамотное и логически стройное изложение материала при аргументации ответов на вопросы в билете, свободно владеет методами рефакторинга, паттернами и антипаттернами.

Оценка **“хорошо”** - Практические задания выполнены в срок в объеме не менее 75%. Студент разработал ООП архитектуру предметной области или Студент разобрал архитектурные паттерны для решения одной задачи и сравнил результаты. Студент демонстрирует правильные, уверенные действия по применению полученных знаний на практике, грамотное и логически стройное изложение материала при аргументации

ответов на вопросы в билете, возможно не знает некоторые паттерны проектирования или некоторые методы рефакторинга.

Оценка **“удовлетворительно”** - Практические задания выполнены в срок в объеме не менее 60%. Студент реализовал приложение в рамках паттерна MVC без использования фреймворков и попробовал строить ООП модель предметной области. Студент демонстрирует правильные, уверенные действия по применению полученных знаний на практике, грамотное и логически стройное изложение материала при аргументации ответов на вопросы в билете, знает некоторые методы рефакторинга и некоторые паттерны проектирования.

Оценка **“не удовлетворительно”** - Практические задания выполнены в срок в объеме менее 60%. Студент не в состоянии реализовать приложение в рамках паттерна MVC без использования фреймворков или не знает базовые принципы ООП.

Оценочные средства для инвалидов и лиц с ограниченными возможностями здоровья выбираются с учетом их индивидуальных психофизических особенностей.

– при необходимости инвалидам и лицам с ограниченными возможностями здоровья предоставляется дополнительное время для подготовки ответа на экзамене;

– при проведении процедуры оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья предусматривается использование технических средств, необходимых им в связи с их индивидуальными особенностями;

– при необходимости для обучающихся с ограниченными возможностями здоровья и инвалидов процедура оценивания результатов обучения по дисциплине может проводиться в несколько этапов.

Процедура оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья по дисциплине предусматривает предоставление информации в формах, адаптированных к ограничениям их здоровья и восприятия информации:

Для лиц с нарушениями зрения:

- в печатной форме увеличенным шрифтом,
- в форме электронного документа. Для

лиц с нарушениями слуха:

- в печатной форме, в форме электронного документа.

Для лиц с нарушениями опорно-двигательного аппарата:

- в печатной форме, в форме электронного документа.

Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

5. ПЕРЕЧЕНЬ ОСНОВНОЙ И ДОПОЛНИТЕЛЬНОЙ УЧЕБНОЙ ЛИТЕРАТУРЫ, НЕОБХОДИМОЙ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

5.1 Основная литература:

1. Мацяшек, Л. А. Практическая программная инженерия на основе учебного примера=Practical software engineering. A case study approach : практическое пособие : [16+] / Л. А. Мацяшек, Брюс Ли Лионг ; пер. с англ. А. М. Епанешникова, В. А. Епанешникова. – 4-е изд., электрон. – Москва : Лаборатория знаний, 2020. – 959 с. – (Программисту). – Режим доступа: по подписке. – URL: <https://biblioclub.ru/index.php?page=book&id=466728> (дата обращения: 29.05.2024). – Библиогр. в кн. – ISBN 978-5-00101-783-7. – Текст : электронный.
2. Фултон, Х. Программирование на языке Ruby : практическое пособие : [16+] / Х. Фултон ; пер. с англ. А. А. Слинкина. – 2-е изд. – Москва : ДМК Пресс, 2023. – 689 с. : ил., табл. – Режим доступа: по подписке. – URL: <https://biblioclub.ru/index.php?page=book&id=705236> (дата обращения: 29.05.2024). – ISBN 978-5-89818-531-2. – Текст : электронный.
3. Фултон, Х. Путь Ruby : практическое пособие : [16+] / Х. Фултон ; совм. с А. Арко ; пер. с англ. А. А. Слинкина. – 4-е изд. – Москва : ДМК Пресс, 2023. – 665 с. : ил. – Режим доступа: по

подписке. – URL: <https://biblioclub.ru/index.php?page=book&id=704224> (дата обращения: 29.05.2024). – ISBN 978-5-89818-334-9. – Текст : электронный.

5.2. Дополнительная литература:

1. Бабушкина, Ирина Анатольевна. Практикум по объектно-ориентированному программированию / И. Бабушкина, С. Окулов. - 2-е изд. - М. : БИНОМ. Лаборатория знаний, 2009. - 366 с. : ил. - Библиогр. : с. 358. - ISBN 9785996302192 : 189.75..
2. Орлов, С. А. Технологии разработки программного обеспечения [Текст] : учебник С.А. Орлов. - СПб. : ПИТЕР, 2002. - 463с. - (Учебник для вузов). - Библиогр.:с.454-457 . -Алф. указ.: с. 458-463. (37 экз. в библиотеке КубГУ).
3. Громов Ю.Ю. , Иванова О.Г. , Беляев М.П. , Минин Ю.В. Технология программирования. - Министерство образования и науки Российской Федерации, Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
4. «Тамбовский государственный технический университет». - Тамбов : Издательство ФГБОУ ВПО «ТГТУ», 2013. - 173 с. [Электронный ресурс]. - URL: <http://biblioclub.ru/index.php?page=book&id=277802>.
5. Хорев П.Б. Технологии объектно-ориентированного программирования: Учебное а. пособие для студентов вузов. / П.Б. Хорев. – М.: Академия, 2004. – 448с. (51 экз. в библиотеке КубГУ).

5.3. Интернет-ресурсы, в том числе современные профессиональные базы данных и информационные справочные системы

Электронно-библиотечные системы (ЭБС):

1. ЭБС «ЮРАЙТ» <https://urait.ru/>
2. ЭБС «УНИВЕРСИТЕТСКАЯ БИБЛИОТЕКА ОНЛАЙН» <http://www.biblioclub.ru/>
3. ЭБС «BOOK.ru» <https://www.book.ru>
4. ЭБС «ZNANIUM.COM» www.znanium.com
5. ЭБС «ЛАНЬ» <https://e.lanbook.com>

Профессиональные базы данных

1. Scopus <http://www.scopus.com/>
2. ScienceDirect <https://www.sciencedirect.com/>
3. Журналы издательства Wiley <https://onlinelibrary.wiley.com/>
4. Научная электронная библиотека (НЭБ) <http://www.elibrary.ru/>
5. Полнотекстовые архивы ведущих западных научных журналов на Российской платформе научных журналов НЭИКОН <http://archive.neicon.ru>
6. Национальная электронная библиотека (доступ к Электронной библиотеке диссертаций Российской государственной библиотеки (РГБ) <https://rusneb.ru/>
7. Президентская библиотека им. Б.Н. Ельцина <https://www.prilib.ru/>
8. База данных CSD Кембриджского центра кристаллографических данных (CCDC) <https://www.ccdc.cam.ac.uk/structures/>
9. Springer Journals: <https://link.springer.com/>
10. Springer Journals Archive: <https://link.springer.com/>
11. Nature Journals: <https://www.nature.com/>
12. Springer Nature Protocols and Methods: <https://experiments.springernature.com/sources/springer-protocols>
13. Springer Materials: <http://materials.springer.com/>
14. Nano Database: <https://nano.nature.com/>

15. Springer eBooks (i.e. 2020 eBook collections): <https://link.springer.com/>
16. "Лекториум ТВ" <http://www.lektorium.tv/>
17. Университетская информационная система РОССИЯ <http://uisrussia.msu.ru>

Информационные справочные системы

1. **Консультант Плюс** - справочная правовая система (доступ по локальной сети с компьютеров библиотеки)

Ресурсы свободного доступа

1. КиберЛенинка <http://cyberleninka.ru/>;
2. Американская патентная база данных <http://www.uspto.gov/patft/>
3. Министерство науки и высшего образования Российской Федерации <https://www.minobrnauki.gov.ru/>;
4. Федеральный портал "Российское образование" <http://www.edu.ru/>;
5. Информационная система "Единое окно доступа к образовательным ресурсам" <http://window.edu.ru/>;
6. Единая коллекция цифровых образовательных ресурсов <http://school-collection.edu.ru/> .
7. Проект Государственного института русского языка имени А.С. Пушкина "Образование на русском" <https://pushkininstitute.ru/>;
8. Справочно-информационный портал "Русский язык" <http://gramota.ru/>;
9. Служба тематических толковых словарей <http://www.glossary.ru/>;
10. Словари и энциклопедии <http://dic.academic.ru/>;
11. Образовательный портал "Учеба" <http://www.ucheba.com/>;
12. Законопроект "Об образовании в Российской Федерации". Вопросы и ответы http://xn--273--84d1f.xn--p1ai/voprosy_i_otvety

Собственные электронные образовательные и информационные ресурсы КубГУ

1. Электронный каталог Научной библиотеки КубГУ <http://megapro.kubsu.ru/MegaPro/Web>
2. Электронная библиотека трудов ученых КубГУ <http://megapro.kubsu.ru/MegaPro/UserEntry?Action=ToDb&idb=6>
3. Среда модульного динамического обучения <http://moodle.kubsu.ru>
4. База учебных планов, учебно-методических комплексов, публикаций и конференций <http://infoneeds.kubsu.ru/>
5. Библиотека информационных ресурсов кафедры информационных образовательных технологий <http://mschool.kubsu.ru/>;
6. Электронный архив документов КубГУ <http://docspace.kubsu.ru/>
7. Электронные образовательные ресурсы кафедры информационных систем и технологий в образовании КубГУ и научно-методического журнала "ШКОЛЬНЫЕ ГОДЫ" <http://icdau.kubsu.ru/>

6. Методические указания для обучающихся по освоению дисциплины

По курсу предусмотрено проведение лекционных занятий, на которых дается основной систематизированный материал, лабораторных работ, контрольной работы, и экзамена.

Важнейшим этапом курса является самостоятельная работа по дисциплине с использованием указанных литературных источников и методических указаний автора курса.

Лабораторные работы выполняются в компьютерном классе. Отдельные работы могут выполняться в аудитории при наличии у бакалавров портативных компьютеров.

Виды и формы СР, сроки выполнения, формы контроля приведены выше в данном документе.

Для лучшего освоения дисциплины при защите ЛР студент должен ответить на несколько вопросов из лекционной части курса.

В освоении дисциплины инвалидами и лицами с ограниченными возможностями здоровья большое значение имеет индивидуальная учебная работа (консультации) – дополнительное разъяснение учебного материала.

Индивидуальные консультации по предмету являются важным фактором, способствующим индивидуализации обучения и установлению воспитательного контакта между преподавателем и обучающимся инвалидом или лицом с ограниченными возможностями здоровья.

7. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю)

7.1. Перечень информационных технологий

- Проверка домашних заданий и консультирование посредством электронной почты.
- Использование электронных презентаций при проведении лекций и практических занятий.

7.2. Перечень необходимого программного обеспечения

1. Sublime text, NodePad++.
2. Интерпретатор руби и рубигем для установки гем-пакетов
3. MySQL, Postgre.
4. Программы для демонстрации и создания презентаций («Microsoft Power Point»).

8. Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине

Наименование специальных помещений	Оснащенность специальных помещений	Перечень лицензионного программного обеспечения
Учебные аудитории для проведения занятий лекционного типа (ауд. 129, 131, А305.)	Мебель: учебная мебель Технические средства обучения: экран, проектор, компьютер	PowerPoint.
Учебные аудитории для проведения занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной	Мебель: учебная мебель Технические средства обучения: экран, проектор,	Аудитория, (кабинет) – компьютерный класс

аттестации (ауд. 147,148)	компьютер	
Учебные аудитории для проведения лабораторных работ. Лаборатория 102,105,106	Мебель: учебная мебель Технические средства обучения: компьютер	Лаборатория, укомплектованная специализированными техническими средствами обучения – компьютерный класс, с возможностью подключения к сети «Интернет», программой экранного увеличения и обеспеченный доступом в электронную информационно-образовательную среду университета

Для самостоятельной работы обучающихся предусмотрены помещения, укомплектованные специализированной мебелью, оснащенные компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду университета.

Наименование помещений для самостоятельной работы обучающихся	Оснащенность помещений для самостоятельной работы обучающихся	Перечень лицензионного программного обеспечения
Помещение для самостоятельной работы обучающихся (читальный зал Научной библиотеки)	Мебель: учебная мебель Комплект специализированной мебели: компьютерные столы Оборудование: компьютерная техника с подключением к информационно-коммуникационной сети «Интернет» и доступом в электронную информационно-образовательную среду образовательной организации, веб-камеры, коммуникационное оборудование, обеспечивающее доступ к сети интернет (проводное соединение и беспроводное соединение по технологии Wi-Fi)	1. OS UNIX - любая 2. Sublime text, NodePad++. 3. Интерпретатор руби и рубигем для установки гем-пакетов 4. MySQL, Postgre. 5. Программы для демонстрации и создания презентаций («Microsoft Power Point»).