ПРИЛОЖЕНИЕ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ В.Ф. УТКИНА»

Кафедра «Систем автоматизированного проектирования
вычислительных средств»

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ

Архитектуры промышленных программных систем

Направление подготовки
09.03.01 «Информатика и вычислительная техника»

Направленность (профиль) подготовки
«Интеллектуальные системы и технологии»

Квалификация выпускника – бакалавр

Форма обучения – очная

Рязань 2026

1. ОБЩИЕ ПОЛОЖЕНИЯ

Оценочные материалы – это совокупность учебно-методических материалов (контрольных заданий, описаний форм и процедур), предназначенных для оценки качества освоения обучающимися данной дисциплины как части основной профессиональной образовательной программы.

Цель – оценить соответствие знаний, умений и уровня приобретенных компетенций, обучающихся целям и требованиям основной профессиональной образовательной программы в ходе проведения текущего контроля и промежуточной аттестации.

Основная задача – обеспечить оценку уровня сформированности общепрофессиональных и профессиональных компетенций, приобретаемых обучающимся в соответствии с этими требованиями.

Контроль знаний проводится в форме текущего контроля и промежуточной аттестации.

Текущий контроль успеваемости проводится с целью определения степени усвоения учебного материала, своевременного выявления и устранения недостатков в подготовке обучающихся и принятия необходимых мер по совершенствованию методики преподавания учебной дисциплины (модуля), организации работы обучающихся в ходе учебных занятий и оказания им индивидуальной помощи.

К контролю текущей успеваемости относятся проверка знаний, умений и навыков, приобретенных обучающимися в ходе выполнения индивидуальных заданий и лабораторных работ. При оценивании результатов освоения практических занятий и лабораторных работ применяется шкала оценки «зачтено – не зачтено». Количество лабораторных работ и их тематика определены рабочей программой дисциплины, утвержденной заведующим кафедрой.

Результат выполнения каждого индивидуального задания должен соответствовать всем критериям оценки в соответствии с компетенциями, установленными для заданного раздела дисциплины.

Промежуточный контроль по дисциплине осуществляется проведением теоретического зачета.

Форма проведения теоретического зачета – устный ответ по вопросам, сформулированным с учетом содержания учебной дисциплины и утвержденным на заседании кафедры. При подготовке к устному ответу обучаемый может составить в письменном виде план ответа, включающий в себя основные понятия и определения, выводы формул, схемы алгоритмов, фрагменты программ т.п.

2. ПАСПОРТ ОЦЕНОЧНЫХ МАТЕРИАЛОВ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

Контролируемые разделы (темы) дисциплины (результаты по разделам)	Код контролируемой компетенции (или её части)	Наименование оценочного средства
Раздел 1. Основы архитектуры промышленных программных систем.	ОПК–1	Зачёт
Раздел 2. Принципы проектирования архитектур программных систем.	ОПК–1	Зачёт
Раздел 3. Классические архитектуры программных систем.	ОПК–1	Зачёт
Раздел 4. Современные архитектуры программных систем.	ОПК–1	Зачёт
Раздел 5. Интеграция корпоративных приложений.	ОПК–1	Зачёт
Раздел 6. Качество программных архитектур.	ОПК–1	Зачёт

Показатели и критерии обобщенных результатов обучения

Результаты обучения по дисциплине	**Показатели оценки результата**
Компетенция ОПК-1. Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности	Осуществляет управление проектом на всех этапах жизненного цикла" подразумевает, что специалист должен быть способен организовать, планировать, контролировать и завершать проекты в соответствии с установленными целями, сроками, ресурсами и качественными требованиями.
Компетенция ОПК-1.1. Демонстрирует естественнонаучные и общеинженерные знания, знания методов математического анализа и моделирования, теоретического и экспериментального исследования.	Демонстрирует способность специалиста анализировать требования проекта, оценивать доступные технологии и инструменты, и принимать обоснованные решения по их выбору.
Компетенция ОПК-1.2. Применяет естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности.	Компетенция предполагает способность специалиста участвовать в формировании общей IT-инфраструктуры организации.
Компетенция ОПК-1.3. Использует современные информационные технологии в профессиональной деятельности.	Компетенция направлена на обеспечение целостности и согласованности ПО в рамках общей ИТ-инфраструктуры.

Шкала оценки сформированности компетенций

В процессе оценки сформированности знаний, умений и навыков обучающегося по дисциплине, проводимой на этапе промежуточной аттестации в форме теоретического зачета, используется оценочная шкала «зачтено – не зачтено».

Оценка «зачтено» выставляется обучающемуся, который прочно усвоил предусмотренный рабочей программой материал; правильно, аргументировано ответил на все вопросы, с приведением примеров; показал глубокие систематизированные знания; владеет приемами рассуждения и сопоставляет материал из разных источников: теорию связывает с практикой, другими темами данного курса, других изучаемых предметов; без ошибок выполнил практическое задание.

Обязательным условием выставленной оценки является правильная речь в быстром или умеренном темпе. Дополнительным условием получения оценки «зачтено» могут стать хорошие успехи при выполнении самостоятельной и лабораторной работы, систематическая активная работа

на практических занятиях.

Оценка «не зачтено» выставляется обучающемуся, который не справился с 50% вопросов и заданий при прохождении тестирования, в ответах на другие вопросы допустил существенные ошибки. Не может ответить на дополнительные вопросы, предложенные преподавателем. Целостного представления о взаимосвязях элементов курса и использования предметной терминологии у обучающегося нет. Оценивается качество устной и письменной речи, как и при выставлении положительной оценки.

ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ И ВОПРОСЫ ДЛЯ ПОДГОТОВКИ К ЗАЧЕТУ И САМОКОНТРОЛЮ

Что такое архитектурный шаблон?
В чём разница между архитектурным паттерном, дизайнерским паттерном и методологией?
Какие цели преследуются при выборе архитектурного шаблона?
Назовите основные требования к архитектуре ПО.
Как влияет выбор архитектурного шаблона на масштабируемость, поддерживаемость и производительность системы?
Какие существуют основные классы архитектурных шаблонов? Приведите примеры.
Чем отличается монолитная архитектура от микросервисной?
В чём суть слоистой архитектуры (Layered Architecture)?
Какие преимущества и недостатки имеет клиент-серверная архитектура?
Что такое архитектура типа Pipe-Filter? Когда она применяется?
Что такое Model-View-Controller (MVC)? Где он используется?
Как устроена архитектура Model-View-Presenter (MVP)? Чем она отличается от MVC?
Что такое Event-Driven Architecture? Приведите примеры использования.
Как работает архитектура Microservices? Какие у неё основные компоненты?
Что такое Service-Oriented Architecture (SOA)? Как её отличить от микросервисов?
Что такое Domain-Driven Design (DDD)?
Какие элементы входят в DDD-подход: домен, сущности, агрегаты, репозитории?
Как DDD взаимодействует с архитектурными шаблонами?
Что такое Hexagonal Architecture (порт и адаптеры)? Какие плюсы и минусы у этого подхода?
Что такое Clean Architecture (Uncle Bob)? Какие слои в ней выделяются?
Что такое CQRS? В каких случаях его применяют?
Что такое Saga Pattern и как он используется в микросервисах?
Какие есть подходы к обработке транзакций в распределённых системах?
Что такое Event Sourcing? Как оно сочетается с CQRS?
Какие шаблоны используются для обеспечения отказоустойчивости в

распределённых системах?
Как определить, какой архитектурный шаблон подходит для конкретного проекта?
Какие факторы влияют на выбор архитектурного шаблона?
Какие ошибки чаще всего возникают при проектировании архитектуры?
Как протестировать соответствие реализации выбранному архитектурному шаблону?
Как документировать архитектуру системы?
Что такое bounded context в контексте DDD?
Какие инструменты используются для моделирования архитектуры?
Что такое hexagonal architecture и как она обеспечивает тестирование?
Какие есть шаблоны для управления состоянием в распределённых системах?
Как работают event brokers и message queues в архитектуре?
Основы версионного контроля
Что такое система версионного контроля (VCS)?
В чём разница между локальным и распределённым VCS?
Какие основные задачи решает система версионного контроля?
Что такое репозиторий?
Перечислите популярные системы версионного контроля.
Что такое команда git init? Для чего она используется?
Как работает команда git add?
Что делает команда git commit?
В чём смысл команды git status?
Что такое ветка (branch) в Git?
Как клонировать удалённый репозиторий?
Что такое origin в Git?
Что означают команды git push и git pull?
Что такое merge и rebase?
В чём разница между git merge и git rebase?
Как создать новую ветку в Git?
Как переключиться между ветками?
Что такое feature branch workflow?
Что такое pull request и зачем он нужен?
Что такое protected branches и где они применяются?
Что такое commit hash?
Как откатить изменения в Git?
Что такое stash и когда его используют?
Что такое cherry-pick?
Как работает git log?
Что такое конфликт слияния?
Как решаются конфликты в Git?
Что такое fast-forward merge?
Что такое squash commit и зачем он нужен?
Как можно объединить несколько коммитов в один?
Что такое detached HEAD?

Что такое tag в Git?
Что такое hook в Git?
Как работает git blame?
Что такое reflog и как он помогает восстановить изменения?
Как Git взаимодействует с CI/CD-системами?
Что такое GitHub Actions, GitLab CI, Jenkins?
Как Git поддерживает автоматическую сборку и тестирование?
Что такое feature toggle и как он связан с Git?
Как Git используется для управления выпусками (releases)?
Что такое submodule в Git?
Что такое fork и clone?
Что такое diff и как его использовать?
Что такое bare repository?
Как Git хранит данные внутри себя?
Что такое SOLID? Перечислите его пять принципов.
Зачем нужны архитектурные паттерны SOLID?
Как SOLID влияет на читаемость и поддерживаемость кода?
В чём разница между архитектурными шаблонами и принципами SOLID?
Почему принципы SOLID особенно важны при работе с крупными системами?
Сформулируйте принцип SRP.
Приведите пример нарушения SRP.
Как SRP связан с модульностью и тестированием?
Можно ли применить SRP к классам и функциям?
Как улучшить архитектуру с помощью SRP?
Сформулируйте принцип OCP.
Что значит быть "открытым для расширения, но закрытым для модификации"?
Приведите пример реализации OCP.
Как использовать интерфейсы или абстракции для соблюдения OCP?
Как OCP помогает в поддержке кода?
Сформулируйте принцип LSP.
Приведите пример нарушения LSP.
Что такое безопасная подстановка?
Как LSP связан с полиморфизмом?
Как проверить, соблюдается ли LSP в коде?
Сформулируйте принцип ISP.
Что такое "толстый" интерфейс и почему он плох?
Приведите пример разбиения интерфейса согласно ISP.
Как ISP влияет на дизайн классов?
Как ISP помогает избежать зависимости от ненужных методов?
Сформулируйте принцип DIP.
Что такое "зависимость от абстракций, а не от деталей"?
Приведите пример реализации DIP.
Как DIP связана с внедрением зависимостей (DI)?

Как DIP помогает в тестировании и модульности?
Какие проблемы может решить применение SOLID-принципов?
Какие ошибки возникают при игнорировании SOLID?
Может ли система работать без соблюдения SOLID? В каких случаях?
Какие паттерны проектирования тесно связаны с SOLID?
Как проверить, соответствует ли ваш проект принципам SOLID?
Как SOLID влияет на CI/CD и автоматизацию?
Как SOLID помогает в рефакторинге кода?
Можно ли применить SOLID в фронтенд-разработке?
Как SOLID работает в микросервисной архитектуре?
Какие инструменты и практики помогают соблюдать SOLID?
Что такое программная архитектура?
В чём разница между архитектурой и проектированием?
Перечислите основные цели выбора архитектуры ПО.
Что такое монолитная архитектура?
Что такое клиент-серверная архитектура?
Как устроена монолитная архитектура? Опишите её компоненты.
Какие преимущества имеет монолитная архитектура?
Какие недостатки характерны для монолита?
В каких случаях применяется монолитная архитектура?
Как монолит влияет на скорость разработки и деплоя?
Что такое клиент и сервер в контексте архитектуры?
Как устроена клиент-серверная модель?
В чём суть разделения логики между клиентом и сервером?
Какие типы клиент-серверной архитектуры существуют (например: 2-tier, 3-tier)?
Какие плюсы и минусы у клиент-серверной архитектуры?
В чём отличие монолита от клиент-серверной архитектуры?
Как масштабируются монолитная и клиент-серверная архитектуры?
Как архитектура влияет на производительность системы?
Какие проблемы возникают при переходе от монолита к клиент-серверной архитектуре?
Как выбрать между монолитом и клиент-серверной архитектурой?
Приведите примеры продуктов с монолитной архитектурой.
Приведите примеры продуктов с клиент-серверной архитектурой.
Как клиент-серверная архитектура используется в веб-приложениях?
Какие протоколы используются в клиент-серверной архитектуре?
Как обеспечивается безопасность в клиент-серверной архитектуре?
Как монолитная архитектура связана с микросервисами?
Может ли клиент-серверная архитектура быть распределённой?
Какие технологии поддерживают клиент-серверную архитектуру?
Как архитектура влияет на CI/CD-процессы?
Как развивается клиент-серверная архитектура в условиях облачных вычислений?
Какие паттерны проектирования чаще всего используются в клиент-

серверной архитектуре?
Как работают сессии и авторизация в клиент-серверной архитектуре?
Что такое REST API и как он связан с клиент-серверной архитектурой?
Какие инструменты используются для управления состоянием клиента и сервера?
Как тестировать клиент-серверное приложение?
Что такое программная архитектура?
В чём разница между архитектурой и проектированием?
Перечислите основные цели выбора архитектуры ПО.
Что такое монолитная архитектура?
Что такое клиент-серверная архитектура?
Как устроена монолитная архитектура? Опишите её компоненты.
Какие преимущества имеет монолитная архитектура?
Какие недостатки характерны для монолита?
В каких случаях применяется монолитная архитектура?
Как монолит влияет на скорость разработки и деплоя?
Что такое клиент и сервер в контексте архитектуры?
Как устроена клиент-серверная модель?
В чём суть разделения логики между клиентом и сервером?
Какие типы клиент-серверной архитектуры существуют (например: 2-tier, 3-tier)?
Какие плюсы и минусы у клиент-серверной архитектуры?
В чём отличие монолита от клиент-серверной архитектуры?
Как масштабируются монолитная и клиент-серверная архитектуры?
Как архитектура влияет на производительность системы?
Какие проблемы возникают при переходе от монолита к клиент-серверной архитектуре?
Как выбрать между монолитом и клиент-серверной архитектурой?
Приведите примеры продуктов с монолитной архитектурой.
Приведите примеры продуктов с клиент-серверной архитектурой.
Как клиент-серверная архитектура используется в веб-приложениях?
Какие протоколы используются в клиент-серверной архитектуре?
Как обеспечивается безопасность в клиент-серверной архитектуре?
Как монолитная архитектура связана с микросервисами?
Может ли клиент-серверная архитектура быть распределённой?
Какие технологии поддерживают клиент-серверную архитектуру?
Как архитектура влияет на CI/CD-процессы?
Как развивается клиент-серверная архитектура в условиях облачных вычислений?
Какие паттерны проектирования чаще всего используются в клиент-серверной архитектуре?
Как работают сессии и авторизация в клиент-серверной архитектуре?
Что такое REST API и как он связан с клиент-серверной архитектурой?
Какие инструменты используются для управления состоянием клиента и сервера?

Как тестировать клиент-серверное приложение?
Что такое многозвенная архитектура?
Какие уровни (звенья) могут присутствовать в многозвенной архитектуре?
В чём отличие двухзвенной архитектуры от трёхзвенной?
Какие компоненты участвуют в трёхзвенной архитектуре?
Какие преимущества даёт разделение на уровни?
Что такое толстый клиент? Приведите примеры.
Что такое тонкий клиент? Приведите примеры.
В чём основные различия между толстым и тонким клиентом?
Как распределяется логика между клиентом и сервером в зависимости от типа клиента?
Как выбор типа клиента влияет на производительность и масштабируемость?
Как выглядит архитектура с толстым клиентом?
Как выглядит архитектура с тонким клиентом?
В каких случаях целесообразно использовать толстый клиент?
В каких случаях предпочтителен тонкий клиент?
Какой тип клиента чаще используется в веб-приложениях?
Приведите примеры приложений с толстым клиентом.
Приведите примеры приложений с тонким клиентом.
Как толстые клиенты взаимодействуют с серверными компонентами?
Как тонкие клиенты получают данные из сервера?
Какие протоколы используются для взаимодействия между клиентом и сервером?
Какие паттерны проектирования применяются в многозвенных архитектурах?
Как работает MVC в контексте многозвенной архитектуры?
Что такое Presentation Tier, Business Tier, Data Tier?
Как реализуется разделение обязанностей в трёхзвенной архитектуре?
Как обеспечивается безопасность в многозвенной архитектуре?
Какие фреймворки поддерживают многозвенную архитектуру?
Как работают RIA (Rich Internet Applications)?
Как толстые клиенты реализуются в современных системах?
Какие инструменты используются для построения тонких клиентов?
Какие технологии обеспечивают работу клиент-серверного взаимодействия?
Как толстые клиенты связаны с desktop-приложениями?
Как тонкие клиенты связаны с веб-браузерами?
Что такое middle-tier и зачем он нужен?
Как можно оптимизировать передачу данных между клиентом и сервером?
Какие проблемы возникают при переходе от толстого клиента к тонкому?
Что такое сервис-ориентированная архитектура (SOA)?
Какие основные принципы характеризуют SOA?
Что такое сервис в контексте SOA?
Какие типы сервисов существуют?
В чём отличие SOA от монолитной архитектуры?
Что такое микросервисы?

Какие основные характеристики микросервисной архитектуры?
В чём сходство и различие между SOA и микросервисами?
Какие преимущества даёт микросервисная архитектура?
Какие сложности возникают при переходе к микросервисам?
Что такое REST API и как он используется в микросервисах?
Что такое gRPC и зачем он нужен?
Что такое API Gateway и как он работает?
Что такое Service Mesh (например, Istio)?
Что такое Discovery Service и как он участвует в работе микросервисов?
Как микросервисы общаются друг с другом?
Что такое синхронное и асинхронное взаимодействие?
Что такое message broker и какие из них популярны?
Какие паттерны используются для взаимодействия микросервисов (Event-driven, Saga и др.)?
Как обеспечивается транзакционность в распределённых системах?
Как микросервисы хранят данные?
Что такое bounded context в контексте микросервисов?
Как решается проблема дублирования данных?
Что такое CQRS и как оно применяется?
Что такое event sourcing и как оно связано с микросервисами?
Как микросервисы развертываются и масштабируются?
Что такое Docker и как он используется в микросервисах?
Что такое Kubernetes и зачем он нужен?
Как обеспечивается независимое развёртывание каждого сервиса?
Какие подходы к CI/CD используются в микросервисной архитектуре?
Как обеспечивается безопасность в микросервисной архитектуре?
Что такое OAuth 2.0, JWT и как они используются?
Как происходит аутентификация и авторизация в микросервисах?
Какие проблемы безопасности могут возникнуть в микросервисной архитектуре?
Как управлять конфигурациями и секретами в микросервисах?
Как выбрать между SOA и микросервисами?
Когда стоит использовать микросервисную архитектуру?
Какие ошибки чаще всего совершают при переходе на микросервисы?
Как тестировать микросервисы?
Как документировать API микросервисов?
Что такое Serverless Architecture и как она связана с микросервисами?
Как работают observability tools (Prometheus, Grafana, Jaeger)?
Какие инструменты помогают в управлении микросервисами?
Что такое resilience engineering и как он применяется?
Как микросервисы влияют на культуру DevOps?
Что такое облачные технологии?
В чём отличие между IaaS, PaaS и SaaS?
Какие основные модели доставки облачных сервисов?
Перечислите популярные облачные провайдеры.

Как облачные технологии влияют на разработку и эксплуатацию ПО?
Что такое контейнеризация?
В чём отличие виртуальной машины от контейнера?
Что такое Docker и зачем он нужен?
Как устроена архитектура Docker?
Что такое образ (image) и контейнер (container) в Docker?
Как создать Docker-образ?
Как запустить контейнер из образа?
Что такое Dockerfile и для чего он используется?
Как работает система слоёв (layers) в Docker?
Что такое Docker Hub и как он используется?
Что такое docker-compose и зачем он нужен?
Как использовать docker-compose.yml?
Как остановить и удалить контейнер?
Что такое volumes и bind mounts в Docker?
Как обеспечивается изоляция между контейнерами?
Что такое Kubernetes и зачем он нужен?
Как Kubernetes управляет контейнерами?
Что такое pod, deployment, service в Kubernetes?
Какие есть инструменты для управления Kubernetes?
Как Kubernetes взаимодействует с облачными провайдерами?
Как контейнеризация упрощает CI/CD-процессы?
Что такое CI/CD pipeline и как он связан с контейнерами?
Как автоматизировать сборку и деплой контейнеров?
Что такое immutable infrastructure?
Как контейнеры помогают в тестировании и отладке?
Как обеспечивается безопасность контейнеров?
Что такое scanning for vulnerabilities в контейнерах?
Как защитить образы Docker?
Что такое secrets management в контейнерной среде?
Как работать с сертификатами и ключами в контейнерах?
Что такое Serverless Architecture и как она связана с контейнерами?
Что такое Cloud Native и какие его компоненты?
Какие есть альтернативы Docker?
Что такое OCI (Open Container Initiative)?
Как работают sidecar-контейнеры?
Как оптимизировать размер Docker-образов?
Что такое multi-stage builds в Docker?
Как работать с microservices в контейнерной среде?
Что такое observability в контейнерной среде?
Как обеспечить high availability при работе с контейнерами?
Основы потоковой обработки данных
Что такое потоковая обработка данных (stream processing)?
В чём отличие пакетной и потоковой обработки?
Какие основные характеристики данных в потоковых системах?

Что такое event time, processing time и ingestion time?
Что такое windowing и зачем он нужен?
Что такое batch vs. stream processing?
Что такое event-driven architecture?
Что такое Kappa Architecture и как она отличается от Lambda?
Что такое real-time analytics и как он реализуется?
Какие уровни обработки событий могут быть в потоковой архитектуре?
Что такое message broker и какие из них популярны?
Что такое stream processor и какие из них существуют?
Что такое stateful processing и где он применяется?
Что такое checkpointing и state management?
Как работают sources и sinks в потоковой обработке?
Что такое Apache Kafka и как он используется в потоковых системах?
Что такое Apache Flink и как он отличается от Spark Streaming?
Что такое Apache Storm и где он применяется?
Что такое Apache Pulsar и чем он хорош?
Что такое Kafka Streams и когда его использовать?
Что такое event sourcing и как он связан с потоковой обработкой?
Что такое CQRS и как он применяется в потоковых системах?
Что такое event-driven microservices?
Что такое fan-out pattern?
Что такое aggregation over time windows?
Как обеспечивается отказоустойчивость в потоковых системах?
Что такое exactly-once semantics и как он достигается?
Как работает state checkpointing в Apache Flink?
Что такое backpressure и как он управляется?
Как системы обрабатывают пропущенные или дублированные события?
Как потоковые данные интегрируются с OLAP-системами?
Как происходит интеграция с базами данных (RDBMS, NoSQL)?
Как используются materialized views в потоковой обработке?
Как работать с real-time dashboards?
Как интегрировать потоковые данные с ML-моделями?
Как обеспечивается безопасность при передаче потоковых данных?
Как управлять доступом к потоковым системам?
Как логгируются и мониторятся потоки данных?
Как контролировать качество данных в потоках?
Как обеспечивается масштабируемость потоковых систем?
Что такое stream joins и как они работают?
Что такое event time ordering и как его обеспечить?
Какие есть подходы к тестированию потоковых приложений?
Что такое stream partitioning и зачем он нужен?
Как оптимизировать производительность потоковых вычислений?
Что такое интеграция программных систем?
Какие основные задачи решает интеграция?
В чём разница между внутренней и внешней интеграцией?

Почему возникает необходимость в интеграции различных систем?
Какие типичные проблемы возникают при интеграции?
Какие основные задачи стоят перед архитектором при проектировании интеграции?
Что такое обмен данными и как он реализуется?
Как обеспечивается синхронизация данных между системами?
Что такое маршрутизация сообщений и зачем она нужна?
Какие подходы используются для преобразования данных при интеграции?
Какие технические сложности могут возникнуть при интеграции?
Что такое проблема точечных связей (point-to-point integration)?
Как избежать дублирования логики в разных системах?
Что такое проблема синхронизации состояния?
Как обрабатываются ошибки и исключения в интеграционных системах?
Что такое стиль интеграции?
Перечислите основные стили интеграции.
Что такое точечная интеграция и её недостатки?
Что такое шины сообщений (message bus) и как они работают?
Что такое ESB (Enterprise Service Bus) и зачем он нужен?
Что такое ориентированная на сервисы архитектура (SOA) и как она используется для интеграции?
Что такое микросервисная архитектура и её роль в интеграции?
Что такое event-driven architecture и как она применяется?
Что такое publish-subscribe модель и где она используется?
Что такое API-ориентированный стиль интеграции?
Какие инструменты используются для реализации шин сообщений?
Что такое Apache Kafka и как он используется в интеграции?
Что такое RabbitMQ и когда его применять?
Что такое REST API и как он используется в интеграции?
Что такое gRPC и чем он лучше REST?
Что такое Adapter Pattern и как он применяется?
Что такое Facade Pattern и его роль в интеграции?
Что такое Bridge Pattern и зачем он нужен?
Что такое Mediator Pattern и как он помогает?
Что такое Message Queue Pattern и где он применяется?
Как организуется управление конфигурациями в интеграционных системах?
Как обеспечивается надёжность интеграционных процессов?
Как работает мониторинг и логирование в интеграционных системах?
Как тестировать интеграционные сценарии?
Как документировать интеграционные процессы?
Что такое Integration Testing и как его проводить?
Что такое orchestration vs. choreography в контексте интеграции?
Как использовать API Gateways в микросервисной архитектуре?
Что такое service mesh и как он влияет на интеграцию?
Какие есть современные тренды в интеграции систем?
Что такое корпоративная интеграция?

В чём заключается проблема точечных связей между системами?
Какие цели преследует корпоративная интеграция?
Что такое сервисная шина (ESB)?
В чём отличие ESB от простой шины сообщений?
Как устроена архитектура ESB?
Какие основные компоненты содержит ESB?
Как ESB обеспечивает маршрутизацию сообщений?
Что такое медиаторы в ESB и зачем они нужны?
Как ESB обрабатывает ошибки и исключения?
Какие преимущества даёт использование ESB?
Какие недостатки могут быть при внедрении ESB?
В каких случаях целесообразно использовать ESB?
Как ESB влияет на производительность системы?
Может ли ESB использоваться в микросервисной архитектуре?
Что такое паттерн интеграции?
Перечислите основные паттерны интеграции.
Что такое Message Router и как он работает?
Что такое Message Translator и зачем он нужен?
Что такое Content-Based Router и где он применяется?
Что такое Message Filter и как он используется?
Что такое Splitter и Aggregator?
Что такое Pipe and Filter и как он применяется?
Что такое Content Enricher и как он работает?
Что такое Message Broker и как он связан с ESB?
Что такое Request-Reply и как он реализуется?
Что такое Publish-Subscribe и где он используется?
Что такое Event Notification и как он применяется?
Что такое Correlation Identifier и зачем он нужен?
Что такое Dead Letter Channel и как он используется?
Какие популярные ESB-инструменты существуют (например, MuleSoft, Apache Camel)?
Что такое Apache Camel и какие паттерны он поддерживает?
Что такое WSO2 ESB и его особенности?
Как ESB интегрируется с REST API и SOAP-сервисами?
Какие языки маршрутизации используются в ESB (например, DataWeave, XPath)?
Как ESB обеспечивает управление конфигурациями?
Как работает мониторинг и логирование в ESB?
Как обеспечивается безопасность в ESB?
Как ESB обрабатывает нагрузку и масштабируется?
Как тестировать интеграционные сценарии в ESB?
В чём разница между ESB и API Gateway?
Как ESB взаимодействует с микросервисами?
Что такое lightweight integration patterns и их роль?
Какие современные альтернативы ESB существуют?

Какие тренды в интеграции сейчас наиболее актуальны?
Что такое качество архитектуры программной системы?
Какие основные критерии качества архитектуры выделяют в промышленных системах?
В чём разница между функциональными и нефункциональными требованиями?
Почему важны нефункциональные требования при проектировании архитектуры?
Как качество архитектуры влияет на жизненный цикл продукта?
Что такое надёжность архитектуры?
Как обеспечивается работоспособность системы при сбоях?
Что такое высокая доступность (high availability) и как её достигнуть?
Что такое fault tolerance и как он реализуется?
Какие методы используются для обеспечения failover?
Что такое масштабируемость архитектуры?
В чём разница между вертикальной и горизонтальной масштабировостью?
Какие архитектурные решения способствуют масштабируемости?
Что такое load balancing и как он применяется?
Как обеспечить распределённую обработку нагрузки?
Что такое производительность архитектуры?
Какие метрики используются для оценки производительности?
Как оптимизировать время отклика системы?
Что такое latency и как его уменьшить?
Как обеспечить эффективное использование ресурсов?
Что такое поддерживаемость архитектуры?
Какие архитектурные паттерны способствуют поддержке?
Что такое maintainability и как её повысить?
Как документация влияет на поддерживаемость?
Что такое observability и зачем она нужна?
Как архитектура влияет на тестируемость?
Что такое testability и как её обеспечить?
Какие подходы к тестированию архитектуры существуют?
Что такое unit tests, integration tests, end-to-end tests?
Как архитектура влияет на автоматизацию тестирования?
Что такое гибкость архитектуры?
Какие архитектурные шаблоны способствуют гибкости?
Что такое адаптивная архитектура?
Как обеспечить обратную совместимость?
Как архитектура влияет на возможность модификации и расширения?
Что такое безопасность архитектуры?
Какие аспекты безопасности нужно учитывать при проектировании?
Как обеспечить аутентификацию и авторизацию?
Что такое защита от DDoS-атак и как её организовать?
Как архитектура влияет на шифрование данных и передачу информации?
Что такое простота архитектуры?

Как избежать избыточности в проектировании?
Что такое KISS и YAGNI и как они связаны с качеством?
Как обеспечить единообразие решений в проекте?
Как архитектура влияет на понятность кода и документов?
Что такое архитектурный стиль и как он влияет на качество?
Какие инструменты используются для анализа архитектуры (например, SonarQube, ArchUnit)?
Что такое архитектурные эволюционные метрики?
Как оценивать архитектуру при переходе от монолита к микросервисам?
Какие есть методики оценки качества архитектуры (например, ATAM, Architecture Tradeoff Analysis Method)?