

БИЛЕТ 1

Что такое модульное (unit) тестирование? В чём его отличие от интеграционного? Назовите принципы F.I.R.R.S.T.

Модульное тестирование это проверка минимально изолированных компонентов программы (функций, методов, классов) в отрыве от остального кода. Отличие от интеграционного: Модульное проверяет конкретную функцию изолированно (используя моки/заглушки), а интеграционное проверяет совместную работу и взаимодействие нескольких модулей или внешних систем. Принципы F.I.R.R.S.T.: Fast (быстрые), Independent (независимые), Repeatable (повторяемые в любой среде), Self-Validating (самовалидирующиеся, выдают четкий результат успех/провал), Timely (своевременные, пишутся до или во время кодинга).

БИЛЕТ 2

Что такое рефакторинг программного кода? Назовите 4-5 «запахов кода» (code smells) с кратким описанием каждого.

Рефакторинг это изменение внутренней структуры исходного кода для улучшения его читаемости, понимания и поддержки без изменения внешнего поведения системы. Запахи кода (Code Smells): 1. Дублирование кода (Duplicate Code) одинаковые фрагменты в нескольких местах. 2. Длинный метод (Long Method) — функция выполняет слишком много действий и сложна для восприятия. 3. Большой класс (Large Class) класс нарушает SRP (принцип единственной ответственности) и перегружен логикой. 4. Длинный список параметров (Long Parameter List) в функцию передается слишком много аргументов, что затрудняет вызов. 5. Завистливые функции (Feature Envy) метод класса обращается к данным другого класса чаще, чем к собственным.

БИЛЕТ 3

Что такое оптимизация программного кода? Назовите виды оптимизации: по скорости, памяти, читаемости. Когда стоит применять оптимизацию? Оптимизация это модификация кода с целью повышения эффективности его работы (уменьшение потребления ресурсов). Виды оптимизации: • По скорости: уменьшение времени выполнения алгоритмов (замена вложенных циклов, кэширование). • По памяти: снижение потребления ОЗУ или диска (использование генераторов вместо списков, очистка объектов). • По читаемости: приведение кода к стандартам (PEP8, рефакторинг) для облегчения поддержки человеком. Когда применять: Только после профилирования и выявления реальных «узких мест» (bottlenecks). Руководствоваться правилом: «Преждевременная оптимизация корень всех зол».

БИЛЕТ 4

Что такое управление качеством в IT-проекте? Назовите методы обеспечения качества (QA) на разных этапах разработки.

Управление качеством (QA) это системный комплекс процессов, гарантирующий, что разрабатываемое ПО соответствует установленным требованиям, стандартам и ожиданиям пользователей. Методы QA по этапам: • Анализ требований: верификация и ревью технического задания (ТЗ) на полноту и непротиворечивость. • Проектирование: архитектурное ревью, проверка UX/UI макетов. • Разработка (Кодинг): статический анализ кода (линтеры), проведение Code Review, написание Unit-тестов. • Тестирование: функциональное, интеграционное, нагрузочное и регрессионное тестирование готовых сборок. • Внедрение (Релиз): приёмочные тесты (UAT), дымовое тестирование (Smoke testing) на продакшене.

БИЛЕТ 5

Что такое прогнозная стоимость завершения проекта (EAC)? Запишите формулу EAC=BAC / CPI. Что такое CPI и как он рассчитывается?

EAC (Estimate at Completion) ожидаемая общая стоимость проекта по его окончании, рассчитываемая на основе текущих показателей эффективности. Формула: EAC=BAC/CPI, где BAC — утвержденный общий бюджет проекта. CPI (Cost Performance Index) индекс выполнения стоимости. Показывает финансовую эффективность проекта. Рассчитывается как: CPI=EV/AC, где EV (Earned Value) освоённый объём (плановая стоимость фактически выполненных задач), а AC (Actual Cost) — фактические затраты на эти задачи. Если CPI < 1 идет перерасход бюджета, если CPI > 1 экономия.

БИЛЕТ 6

Что такое плотность дефектов (Defect Density)? Запишите формулу расчёта. Как она используется для оценки качества ПО?

Плотность дефектов (Defect Density) метрика, отражающая количество обнаруженных ошибок, приходящихся на единицу размера программного продукта. Формула: Defect Density = Кол-во дефектов / Размер ПО (в KLOC — тыс. строк кода или Story Points) Использование для оценки качества: Метрика позволяет сравнивать качество разных модулей проекта независимо от их объема. Высокая плотность указывает на критические нестабильные компоненты кода, требующие рефакторинга или более глубокого покрытия тестами. Низкая плотность перед релизом свидетельствует о готовности системы.

БИЛЕТ 7

Что такое приёмочное тестирование (UAT)? Чем оно отличается от других видов тестирования? Какова роль заказчика в UAT?

UAT (User Acceptance Testing) это финальный этап тестирование ПО, выполняемый конечными пользователями или заказчиком для проверки соответствия системы бизнес-требованиям. Отличие от других видов: UAT проверяет не техническую корректность кода (как модульные или системные тесты), а удобство и применимость продукта в реальных бизнес-сценариях. Роль заказчика: Заказчик является ключевым участником: он формирует критерии приёмки, выполняет тесты на реальных данных и подписывает акт приёмки, утверждая готовность ПО к эксплуатации.

БИЛЕТ 8

Что такое диаграмма последовательности (Sequence Diagram)? Опишите её элементы: актёры, линии жизни, сообщения, фреймы.

Диаграмма последовательности поведенческая UML-диаграмма, отображающая детальное взаимодействие объектов и компонентов системы во временной последовательности. Элементы: • Актёры (Actors): внешние сущности (пользователи, роли), инициирующие сценарий. • Линии жизни (Lifelines): вертикальные пунктирные линии, обозначающие время существования объекта в процессе. • Сообщения (Messages): горизонтальные стрелки (синхронные, асинхронные, ответы), показывающие обмен данными. • Фреймы (Frames): логические блоки со спецификаторами условий (alt ветвление, loop циклы, opt - опция).

БИЛЕТ 9

Что такое тестирование методом «чёрного ящика»? Опишите его характеристики, преимущества и ограничения.

Тестирование «чёрного ящика» (Black-box) метод тестирования функционала ПО, при котором внутренняя структура, архитектура и код программы полностью неизвестны тестирующим. Характеристики: Тесты базируются исключительно на спецификациях и требованиях. Проверяется только реакция системы на входные данные. Преимущества: Тестировщику не нужно знать языки программирования; объективность (тесты независимы от разработчика); раннее нахождение пробелов в логике ТЗ. Ограничения: Невозможно протестировать скрытые пути выполнения кода; риск дублирования тест-кейсов; сложность локализации конкретной строчки ошибки в коде.

БИЛЕТ 10

Что такое техническое задание (ТЗ)? Назовите обязательные разделы ТЗ на разработку программного продукта.

Техническое задание (ТЗ) основной юридический и технический документ, детально определяющий требования к ПО, цели разработки, критерии качества и порядок приемки готового продукта. Обязательные разделы (по ГОСТ): 1. Общие сведения о проекте. 2. Назначение и цели создания (разработки) системы. 3. Характеристика объектов автоматизации. 4. Требования к системе (функциональные, нефункциональные, технические, эргономические). 5. Состав и содержание работ по созданию системы (этапы и сроки). 6. Порядок контроля и приемки системы.

БИЛЕТ 11

Назовите основные ограничения проекта («треугольник проекта»: содержание, сроки, стоимость). Объясните их взаимосвязь.

Треугольник ограничений включает три ключевых параметра: Содержание (объем функций/задач), Сроки (время на разработку) и Стоимость (бюджет и ресурсы). В центре треугольника находится Качество. Взаимосвязь: Эти параметры жестко взаимосвязаны. Изменение одного параметра неизбежно влечет деформацию остальных. Например, если заказчик требует увеличить содержание (добавить фичи), проектная команда обязана либо соразмерно увеличить сроки, либо поднять стоимость для привлечения новых ресурсов. Попытка зафиксировать все три вершины при рост требований ведет к критическому падению качества.

БИЛЕТ 12

Что такое производительность команды в разработке ПО? Как её измеряют? Назовите факторы, влияющие на производительность.

Производительность команды объем полезной и качественной работы, выполняемый группой специалистов за фиксированную единицу времени (спринт, месяц). Измерение: В Agile измеряется скоростью (Velocity) количеством Story Points или закрытых задач за спринт. Традиционно может измеряться выработкой (функции/модули в месяц). Факторы влияния: Квалификация и опыт участников (Senior/Junior), четкость и стабильность требований в ТЗ, уровень автоматизации процессов (CI/CD, автотесты), внутренняя атмосфера и коммуникация в коллективе, технический долг проекта.

БИЛЕТ 13

Что такое отклонение по стоимости (CV) и индекс выполнения расписания (SPI)? Запишите формулы и объясните интерпретацию результатов.

CV (Cost Variance) отклонение по стоимости. Показывает соответствие текущих затрат плану. Формула: CV=EV-AC. • Интерпретация: CV>0 экономия бюджета; CV=0 строго в бюджете; CV<0 перерасход средств. SPI (Schedule Performance Index) индекс выполнения расписания. Отражает темпы соблюдения сроков. Формула: SPI=EV/PV (где PV плановый объем задач). • Интерпретация: SPI > 1 — опережение календарного графика; SPI=1 проект идет ровно по плану; SPI<1 отставание от графика.

БИЛЕТ 14

Что такое тестирование «белым ящиком» (white-box testing)? Чем оно отличается от тестирования «чёрным ящиком»? Что такое покрытие ветвей?

Тестирование «белым ящиком» метод тестирования ПО, основанный на полном знании внутренней структуры, логики и исходного кода программы. Отличие от «черного ящика»: Тестировщик анализирует сам код, пути его выполнения, ветвления и условия, в то время как в «черном ящике» проверяются только внешние функции по ТЗ без доступа к коду. Покрытие ветвей (Branch Coverage) — метрика тестирования «белого ящика», показывающая, какой процент возможных исходов управляющих конструкций (условий if/else) был пройден написанными тестами. Для 100% покрытия нужно составить тесты так, чтобы каждое условие сработало и в True, и в False.

БИЛЕТ 15

Что такое статический анализ кода? Назовите типичные виды ошибок, которые он выявляет. Чем отличается от динамического тестирования?

Статический анализ это автоматизированная проверка исходного кода программы специализированными инструментами (линтерами, анализаторами) без запуска самого приложения. Виды выявляемых ошибок: Синтаксические ошибки, мертвый код (неиспользуемые переменные/функции), потенциальные утечки памяти, выход за границы массивов, нарушения код-стайла (например, PEP8), использование уязвимых функций. Отличие от динамического тестирования: Статический анализ изучает «сухой» текст программы, а динамическое тестирование обязательно запускает код на исполнение с конкретными тест-кейсами для проверки поведения в реальном времени.

БИЛЕТ 16

Что такое спринт в методологии Scrum? Из каких артефактов и событий состоит Scrum?

Спринт в Scrum фиксированный, регулярно повторяющийся отрезок времени (обычно от 1 до 4 недель), в течение которого команда создает работоспособный, потенциально готовый к релизу инкремент продукта. События Scrum: Планирование спринта (Sprint Planning), Ежедневный Scrum (Daily Standup), Обзор спринта / Демо (Sprint Review), Ретроспектива (Sprint Retrospective). Артефакты Scrum: Бэклог продукта (Product Backlog), Бэклог спринта (Sprint Backlog), Инкремент продукта (Increment).

БИЛЕТ 17

Что такое бэклог продукта (Product Backlog) в Scrum? Как приоритизируются задачи в бэклоге?

Бэклог продукта (Product Backlog) — динамический, упорядоченный список всех требований, пользовательских историй (User Stories), функций, доработок и багов, которые необходимо реализовать в конечном продукте. Приоритизация задач: Задачи ранжируются Владелцем Продукта (Product Owner) сверху вниз. Чем выше ценность задачи для бизнеса/пользователя и меньше рисков/затрат на реализацию, тем выше она находится. Для объективной оценки применяются методики: MOSCOW (Must, Should, Could, Won't), матрица Эйзенхауэра или фреймворк WSJF (Weighted Shortest Job First).

БИЛЕТ 18

Что такое диаграмма Ганта? Назовите её основные элементы (задачи, вехи, зависимости). Для чего она используется в управлении проектами?

Диаграмма Ганта это горизонтальная столбчатая диаграмма, используемая для визуализации календарного плана-графика выполнения работ по проекту. Основные элементы: • Задачи: горизонтальные полосы, длина которых отражает длительность этапа. • Вехи (Milestones): маркеры (ромбы), обозначающие ключевые события или точки контроля (например, подписание этапа). • Зависимости (Связи): стрелки, соединяющие задачи и задающие строгую последовательность (например, Задача B начнется только после Задачи A). Применение: Используется для оперативного планирования, отслеживания прогресса, выявления конфликтов ресурсов и расчета критического пути проекта.

БИЛЕТ 19

Для чего используется документирование программного обеспечения? Назовите виды программной документации. Что должна содержать документация к функциям?

Документирование ПО необходимо для фиксации знаний о системе, облегчения онбординга новых разработчиков, снижения стоимости поддержки кода и предоставления руководства пользователям. Виды документации: Архитектурная (системная), техническая (API, комментарии, docstrings), пользовательская (инструкции, FAQ). Документация к функциям должна содержать: Общее описание назначения функции; детальное описание входных аргументов (Args) с указанием их типов; описание возвращаемого значения (Returns) и его типа; список возможных исключений/ошибок (Raises), которые функция может выбросить.

БИЛЕТ 20

Что такое метрики программного обеспечения? Назовите основные группы метрик (метрики кода, метрики процесса, метрики качества).

Метрики ПО это числовые или количественные показатели, позволяющие измерить и оценить характеристики создаваемого софта или эффективность процессов разработки. Основные группы метрик: 1. Метрики кода: оценивают внутреннюю структуру (кол-во строк LOC, цикломатическая сложность Маккейба, метрики Холстеда/Чеплина). 2. Метрики процесса: измеряют скорость и стабильность работы команды (Velocity в Scrum, Lead Time/Cycle Time в Kanban). 3. Метрики качества: отражают надежность продукта (плотность дефектов Defect Density, процент успешных тест-кейсов, покрытие кода тестами).

БИЛЕТ 21

Что такое управление проектами? Назовите основные цели и задачи управления ИТ-проектом.

Управление проектами это профессиональная деятельность по применению знаний, методов, навыков и инструментов для успешной реализации целей проекта в рамках установленных ограничений. Цели и задачи в IT: Главная цель выпустить качественный, работающий программный продукт, удовлетворяющий заказчика, вовремя и в рамках бюджета. Задачи включают: планирование этапов и архитектуры, распределение задач в команде, управление рисками, контроль бюджета (через метрики EVM) и организация коммуникаций.

БИЛЕТ 22

Что такое agile-методологии? Назовите принципы Agile Manifesto. Чем Scrum отличается от Kanban?

Agile семейство гибких итеративно-инкрементальных подходов к разработке ПО. Принципы Agile Manifesto: 1) Люди и взаимодействие важнее процессов и инструментов; 2) Работающий продукт важнее исчерпывающей документации; 3) Сотрудничество с заказчиком важнее согласования условий контракта; 4) Готовность к изменениям важнее следования первоначальному плану. Отличия Scrum и Kanban: Scrum делит процесс на жесткие итерации (спринты), имеет строгие роли (Scrum-master, PO) и обязательные митинги. Kanban представляет собой непрерывный поток задач без фиксированных итераций и ролей; его фокус визуализация доски и жесткое ограничение одновременно выполняемой работы (WIP limits).

БИЛЕТ 23

Что такое управление командой в IT-проекте? Назовите роли в Scrum-команде и опишите ответственность каждой.

Управление командой координация деятельности участников, создание условий для высокой продуктивности, мотивация и разрешение конфликтов в проекте. Роли в Scrum-команде: • Владелец Продукта (Product Owner): отвечает за максимизацию ценности продукта, управляет бэклогом, формулирует требования бизнеса. • Scrum-мастер (Scrum Master): фасилитатор процессов. Помогает команде соблюдать Scrum, устраняет внешние препятствия и блокеры. • Команда разработки (Developers): кросс-функциональная группа специалистов (программисты, QA, дизайнеры), непосредственно создающая инкремент продукта.

БИЛЕТ 24

Что такое управление рисками в проекте? Назовите этапы: идентификация оценка -> планирование ответных мер мониторинг.

Управление рисками процесс упреждающего обнаружения, анализа и минимизации воздействия непредвиденных негативных факторов на цели проекта. Этапы управления рисками: 1. Идентификация: составление реестра потенциальных угроз (уход разработчика, изменение ТЗ). 2. Оценка: анализ вероятности наступления риска и степени его ущерба. 3. Планирование мер: выбор стратегии реагирования (избегание, снижение ущерба, передача риска, принятие). 4. Мониторинг: непрерывное отслеживание маркеров рисков и применение планов реагирования в ходе разработки.

БИЛЕТ 25

Что такое сравнительный анализ программного обеспечения? Назовите критерии, по которым сравниваются офисные пакеты (MS Office и OpenOffice).

Сравнительный анализ по это методология сопоставления нескольких аналогичных программных продуктов для выбора оптимального решения под конкретные задачи предприятия. Критерии сравнения (MS Office vs OpenOffice): Стоимость приобретения и владения лицензией; совместимость с отраслевыми форматами файлов (.docx, .xlsx, .odt); функциональная полнота (наличие продвинутых инструментов анализа); кроссплатформенность; скорость работы и требования к аппаратному обеспечению; глубина поддержки макросов и автоматизации скриптов.

БИЛЕТ 26

Что такое ветвление (branching) в Git? Зачем используются ветки? Опишите типичный сценарий работы с feature-веткой.

Ветвление в Git механизм создания параллельных линий разработки в рамках одного репозитория. Зачем нужны ветки: Они позволяют изолированно разрабатывать новые фичи, проводить эксперименты или исправлять критические баги, не затрагивая стабильный код в основной ветке (main/master). Сценарий feature-ветки: Разработчик создает новую ветку от main ('git checkout -b feature/login') -> пишет код, фиксируя изменения коммитами -> отправляет ветку на сервер (git push origin ...) -> создает Pull Request для проверки -> после успешного Code Review ветка вливается (merge) обратно в ветку main, а локальная и удаленная feature-ветка удаляются.

БИЛЕТ 27

Что такое электронно-цифровая подпись (ЭЦП)? Каковы её основные функции? Где применяется ЭЦП в разработке ПО?

ЭЦП криптографический реквизит электронного документа, полученный в результате шифрования данных закрытым ключом автора. Основные функции: Гарантия авторства (невозможность отказаться от подписи) и контроль целостности документа (любое изменение данных после подписания делает ЭЦП невалидной). Применение в разработке ПО: Подписание коммитов и тегов в Git (GPG-подпись для подтверждения личности автора); подписание исполняемых файлов (.exe, .apk, .dmg) сертификатами разработчика для исключения предупреждений операционных систем об опасном ПО при установке.

БИЛЕТ 28

Что такое функциональные и нефункциональные требования к ПО? Приведите по 3 примера каждого вида.

Функциональные требования определяют действия, которые система должна выполнять (её функции и поведение).
• Примеры: 1. Система должна отправлять email при успешной регистрации. 2. Администратор может удалять заблокированных пользователей. 3. Кнопка «Экспорт» должна выгружать отчет в Excel.
Нефункциональные требования описывают качественные характеристики, свойства и ограничения, в рамках которых функции должны работать.
• Примеры: 1. Время отклика страницы не должно превышать 1.5 секунды. 2. Система должна быть локализована на русский и английский языки. 3. База данных должна шифровать персональные данные пользователя по стандарту AES-256.

БИЛЕТ 29

Что такое ревьюирование кода? Каковы его цели и основные этапы? Назовите виды ревью.

Ревьюирование кода (Code Review) процесс совместной проверки исходного кода программы другими разработчиками (коллегами автора). Цели: Обнаружение багов на ранних стадиях, улучшение читаемости кода, контроль соблюдения стандартов (код-стайла), обмен опытом внутри команды. Этапы: Создание Pull Request -> Назначение ревьюеров -> Анализ кода и написание замечаний -> Исправление замечаний автором -> Одобрение (Approve) и слияние кода. Виды ревью: Парное программирование, формальная инспекция (инспекция кода по чек-листам), легковесное ревью (через инструменты Git PR/MR).

БИЛЕТ 30

Что такое система контроля версий (СКВ)? Назовите основные типы СКВ и приведите по одному примеру каждого типа.

Система контроля версий (СКВ) - специализированное ПО, предназначенное для фиксации изменений в файлах проекта, хранения истории правок и организации совместной командной работы. Типы СКВ: 1. Централизованные (CVCS): Все версии хранятся на одном выделенном сервере. Разработчик скачивает только рабочий файл. Пример: Subversion (SVN), Perforce. 2. Распределенные (DVCS): Каждый разработчик копирует к себе на компьютер весь репозиторий целиком вместе со всей историей изменений. Пример: Git, Mercurial.

БИЛЕТ 31

Что такое ретроспектива в Scrum? Какова её цель? Опишите формат проведения ретроспективы.

Ретроспектива обязательное финальное событие Scrum-спринта, на котором команда анализирует свою деятельность и рабочие процессы за прошедший цикл. Цель: Выявление проблем во взаимодействии и процессах, поиск путей их решения и составление конкретного плана улучшений на следующий спринт. Формат проведения: Команда собирается вместе с Scrum-мастером и отвечает на три классических вопроса: 1. Что прошло хорошо и должно быть сохранено? 2. Что пошло не так и вызвало трудности? 3. Что мы исправим/улучшим в следующем спринте? По итогам фиксируются ответственные и конкретные шаги.

БИЛЕТ 32

Что такое критический путь проекта? Как он связан с диаграммой Ганта? Почему задержка задач на критическом пути приводит к задержке всего проекта?

Критический путь наиболее длинная по времени непрерывная последовательность взаимосвязанных задач от начала до полного завершения проекта. Связь с диаграммой Ганта: На диаграмме Ганта критический путь наглядно выделяется сквозной цепочкой взаимосвязанных полос (обычно красного цвета), не имеющих временного резерва. Причина задержки проекта: Задачи критического пути имеют нулевой временной резерв (люфт). Так как они жестко связаны последовательно, увеличение длительности любой из них автоматически сдвигает дату финального релиза проекта.

БИЛЕТ 33

Что такое отладка программного обеспечения? Назовите основные методы отладки и инструменты отладки в Python.

Отладка (Debugging) это целенаправленный этап разработки, заключающийся в поиске, локализации, анализе причин и устранении уже обнаруженных ошибок (багов) в коде. Основные методы: Трассировка (вывод значений в консоль с помощью print), логирование (ведение файлов логов), пошаговое выполнение программы, использование интерактивных точек останова (Breakpoints). Инструменты в Python: Встроенный стандартный консольный модуль pdb; визуальные интерактивные отладчики, встроенные в современные среды разработки (PyCharm, VS Code).

БИЛЕТ 34

Что такое UML? Перечислите основные виды UML-диаграмм. Разделите их на структурные и поведенческие.

UML (Unified Modeling Language) унифицированный графический язык моделирования для описания, визуализации и документирования архитектуры и структуры программных систем. Разделение видов диаграмм:
• Структурные (описывают статическую архитектуру): Диаграмма классов (Class), Диаграмма компонентов (Component), Диаграмма объектов (Object), Диаграмма развертывания (Deployment).
• Поведенческие (описывают динамику во времени): Диаграмма вариантов использования (Use Case), Диаграмма последовательности (Sequence), Диаграмма деятельности (Activity), Диаграмма состояний (State Machine).

БИЛЕТ 35

Что такое диаграмма классов UML? Объясните типы отношений: ассоциация, агрегация, композиция, наследование.

Диаграмма классов основная структурная UML-диаграмма, отражающая структуру системы, её классы, их внутренние поля, методы и связи между собой. Типы отношений:
• Ассоциация: общая логическая связь объектов (например, Преподаватель консультирует Студента).
• Агрегация: слабая связь «часть-целое». Часть может существовать без целого (Студент в Группе; если группу расформировать, студент останется). Обозначается пустым ромбом.
• Композиция: жесткая связь «часть-целое». Часть уничтожается вместе с целым (Комната в Доме; без дома комната существовать не может). Закрашенный ромб.
• Наследование: отношение обобщения «is-a» (Кот наследует класс Животное, получая его свойства). Пустая стрелка.

БИЛЕТ 36

Что такое VPN? Назовите 3-4 популярных VPN-протокола (OpenVPN, WireGuard, IPSec, L2TP) и укажите основные отличия между ними.

VPN (Virtual Private Network) виртуальная частная сеть, обеспечивающая защищенное, зашифрованное сетевое соединение поверх публичного Интернета. Отличия протоколов:
• OpenVPN: классический, крайне надежный, использует SSL/TLS, обходит брандмауэры, но создает относительно высокую нагрузку на процессор.
• WireGuard: современный протокол. Имеет компактную кодовую базу, обеспечивает максимальную скорость работы и мгновенное переподключение.
• IPSec: мощный набор протоколов защиты данных на сетевом уровне, часто применяется для жесткого объединения корпоративных офисов (Site-to-Site).
• L2TP: туннельный протокол, сам по себе не шифрует данные, поэтому практически всегда применяется исключительно в связке с IPSec (L2TP/IPSec).

БИЛЕТ 37

Что такое оптимизация программного кода по скорости? Приведите 3 конкретных приёма оптимизации с примерами на Python.

Оптимизация по скорости целенаправленное изменение исходного кода для сокращения времени выполнения алгоритма и снижения процессорных тактов. Приёмы на Python:
1. Использование генераторов списков вместо циклов с функцией '.append()': [x*2 for x in arr] выполняется быстрее, чем цикл for за счет оптимизации на уровне С.
2. Выбор правильных структур данных: использование set для проверки вхождения элемента через if x in my_set вместо list, что снижает сложность с O(n) до O(1).
3. Кэширование вычислений: подключение стандартного декоратора from functools import lru_cache для исключения повторных тяжелых расчетов функций с одинаковыми аргументами.

БИЛЕТ 38

Что такое тест-кейс? Из каких элементов он состоит? Что такое граничные значения и классы эквивалентности?

Тест-кейс это структурированный документ, содержащий набор условий, шагов выполнения и входных параметров, необходимых для проверки реализации конкретной фичи ПО. Элементы: ID, название, предусловия (Preconditions), пошаговые действия, входные данные, ожидаемый результат (Expected Result), фактический результат. Классы эквивалентности метод тест-дизайна, объединяющий входные данные в группы, для которых программа ведет себя абсолютно идентично (выделяют валидные и невалидные классы). Граничные значения тестирование значений, находящихся строго на границах классов эквивалентности (например, если лимит от 1 до 100, проверяются точки: 0, 1, 2, 99, 100, 101).

БИЛЕТ 39

В чём отличие централизованной СКВ от распределённой? Назовите достоинства и недостатки каждого типа.

Централизованная СКВ (SVN): Хранит историю исключительно на одном сервере.
• Плюсы: жесткий централизованный контроль прав доступа, простота понимания структуры.
• Минусы: при потере связи с сервером невозможно фиксировать версии; единая точка отказа (крах сервера уничтожает историю). Распределенная СКВ (Git): Каждый разработчик имеет локально полноценную копию репозитория со всей историей.
• Плюсы: автономность работы без сети, высочайшая скорость коммитов, высокая надежность резервных копий.
• Минусы: копия проекта занимает больше дискового пространства; сложнее организовать запрет на чтение отдельных папок репозитория.

БИЛЕТ 40

Что такое диаграмма вариантов использования (Use Case Diagram)? Назовите все её элементы и объясните их назначение.

Диаграмма вариантов использования поведенческая UML-диаграмма высокого уровня, концептуально описывающая функционал системы через роли пользователей и их возможные действия. Элементы и назначение:
• Актёры (Actors): фигурки людей, обозначающие роли пользователей или внешних систем, взаимодействующих с программой.
• Варианты использования (Use Cases): овалы, обозначающие конкретные функции системы (например, «Авторизоваться»).• Связи (Relationships): ассоциация (простая линия связи), <> (обязательное включение подфункции), << (расширение функционала при определенных условиях).