1. **CALS-технологии**

**CALS** – непрерывная информационная поддержка поставок и ЖЦ изделия. У большего числа продукции есть **электронная документация**. К высокотехнологическим изделиям предъявляется ряд жестких требований, которые выполняются CALS-технологиями: представление конструкторской и технологической документаций **в электронной форме**, эксплуатационной и ремонтной в форме **интерактивных электронных технических руководств**. Главная цель при внедрении CALS – **минимизация затрат** в ходе ЖЦ изделия, **повышение его качества**. Сокращение сроков освоения производства изделий, улучшение их качества. Повышение благодаря тому, что вся информация и характеристики хранятся в электронном виде. Преимущества: параллельное выполнение сложных проектов, планирование и управление многими предприятиями, уменьшение количества ошибок и переделок. Составляющие: **CAD** – инструментальный комплекс технических и программных средств автоматизированного проектирования изделий, **CAM** – системы автоматизации технологической подготовки производства, **CAE** – системы инженерного анализа, **EDM** – система управления проектными и инженерными данными, системы визуализации всего процесса разработки документации, средства обмена данными, средства разработки прикладного ПО.

**Анализ рисков и характеристик качества программного обеспечения при внедрении.**

Риск – проблема, которая не возникла, проблема – риск, который материализовался. Причина рисков – неопределенности. Мы не боремся с рисками – мы ими управляем. Управление рисками – процессы на определение, анализ рисков. Планирование управления рисками, идентификация рисков, оценка рисков, планирование реагирования на риски, мониторинг и контроль рисков. Основные риски: несоблюдение сроков, превышение стоимости, несоблюдение параметров качества. Для IT-проекта: отсутствие системного аналитика, сопротивление сотрудников, увеличение нагрузки, отсутствие лидера, изменение целей. Распространенные риски программного проекта: дефицит специалистов, нереалистичные сроки и бюджет, разработка неправильного пользовательского интерфейса, оттачивание деталей. Описание рисков для раннего выявления рисков и проведения комплекса мероприятий: документирование целей, повышение мотивации, привлечение сторонних специалистов. Сбор информации: опрос экспертов, мозговой штурм, метод Дельфи, карточки Кроуфорда. Риск – последствия – меры по предотвращению – меры по минимизации. Треугольники и соотношения.

2. **Актуальность определения качества ИС. Методы оценки качества ИС.**

**Оценка** проводится на всех фазах ЖЦ, что обеспечивает: Своевременное устранение недостатков. Соответствие меняющимся требованиям пользователей. Важно понять соответствует ли ПО требованиям бизнеса, и если соответствует, то насколько. Это делается для **минимизации рисков** (оценка надежности помогает выявить слабые места), **оптимизации затрат**, **соответствия требованиям пользователям**. Особенно критичны при внедрении нового ПО, модернизации и Аудита безопасности.

**Измерительный** получение информации о свойствах ИС с использованием инструментальных средств (строки, операторы и операнды, ветви программы); **Регистрационный** получение информации во время испытаний или функционирования ИС, подсчитывается число сбоев. **Органолептический** органы чувств для понимания удобства использования, эффективности. **Расчетный** теоретические и эмпирические зависимости, статистические данные, определяется длительность и точность. **Социологические** как обработка анкет-опросников.

**Целесообразность разработки модулей адаптации.**

Категории рисков: **бюджет** (+решения по мере продвижения работы), **кадры и знания** (+небольшие группы 10-12), **продуктивность** (+спринты – быстрые этапы разработки для демоверсии), **время** (+организация процесса). **IEEE 1219** сопровождение как модификация эксплуатируемого продукта. Категории сопровождения: **корректирующее** – модификация для решения сбоев; **адаптирующее** – модификация для работы в другой среде; **совершенствующее** – модификация для повышения производительности и удобства; **профилактическое** – модификация для выявления скрытых дефектов. **Специалисты по сопровождению** получают пик нагрузки во время ввода в эксплуатацию. **Инженеры по тех. поддержке** – доступ к активам, коду, создание патчей. Обязанности: проверка сценариев сбоя, выявление причин сбоя, предоставление исправлений, журналирование работ, помещение описания решения сбоя в базу знаний, передача информации разработчикам. **Работы по сопровождению**: устранение сбоев, улучшение дизайна, расширение функционала, создание интерфейсов работы с другими системами. **Деятельность персонала сопровождения**: поддержка контроля ПО, модификаций ПО, совершенствование функций, предотвращение падения производительности. **Факторы на стоимость**: тип приложения, новизна ПО, наличие и квалификация персонала по сопровождению, характеристики и специфика аппаратной части, качество дизайна. Методы адаптации: параметрическая (настройка параметров), функциональная (изменение функций), организационная (изменение организации), структурная (изменение структуры), развитие (зарождение системы, становление системой определенного качества, устойчивое функционирование, деградация, гибель).

3. **Аппаратно-программные платформы серверов и рабочих станций**

По функциональному назначению: рабочие станции и серверы. **Сервер** – компьютер для выполнения сервисной задачи без человека. Сервисные работы параллельно с пользователем – невыделенный сервер. В обычном понимании сервер – устройство выполнения команд других компьютеров. Консоль и человек нужны только при начальной настройке, обслуживании и управлении в нештатных ситуациях. В результате специализации у сервера упрощенная консоль или вообще без нее (сетевая настройка). Программный сервер, аппаратный сервер).

**Файрвол: задачи, сравнительный анализ, настройка.**

**Файрвол** – проверка и фильтрация сетевого трафика. **Межсетевой экран** – защита сетей от несанкционированного доступа. **Брандмауэр** – встроенный экран. Действия: **пропуск** - ок, **отклонение** – не ок, **отброс** – исчезни. **Задачи**: защита от несанкционированного доступа, остановка подмены трафика, защита от DDoS-атак, блокировка передачи на неизвестный ip. **Аппаратные файрволы** устанавливаются в сеть: высокая производительность, надежность, высокая цена, недостаточная гибкость. **Программные файрволы** – приложения, работают на уровне ОС: доступность, гибкая настройка, простота установки, низкая производительность, надежность. На входе смотрит пакеты и решает пустить или нет в зависимости от настроек администратора. Может блокироваться: неизвестные источники, запрещенные порты, вредоносный контент. Ограничение доступа: блокировка, ограничение портов, блокировка ip, контроль приложений. **pfSense** – дистрибутив для создания экрана. **Cisco ASA** – комплексная платформа сетевой защиты, задача в комплексной защите сетевой инфраструктуры предприятия. **Check Point Firewall** – экран детальной защиты контента, контроля доступа, аутентификации, барьер между внутренней и внешней сетью. **Fortinet Firewall** – экран защиты от внешних угроз, используется для контроля приложений, фильтрации трафика и веб-сайтов. **Sophos Firewall** – сетевая безопасность, защита организации от киберугроз. **Outpost Firewall** – защита компьютера от хакерских атак из интернета, блокировка загрузки рекламы. **Comodo Internet Security** – файрвол, защита от троянов, вирусов и др. **Kaspersky Internet Security** – комплексная защита домашних ПК.

4. **Безопасный режим Windows. Инициация безопасного режима. Использование безопасного режима Windows для выявления и устранения различного рода несовместимостей.**

**Безопасный режим** – режим запуска Windows для устранения неполадок, в котором используется ограниченный набор служб и компонентов. Загружается только база файлов и драйверов. Перезагрузить компьютер, зажать f8 до появления логотипа, выбрать систему и безопасный режим.

Если после запуска безопасного режима ошибок не происходит, значит проблема не в системных компонентах. Можно использовать другие режимы для дополнительной проверки. Можно проверить приложения из автозапуска и из других местоположений.

**Средства и протоколы шифрования сообщений.**

**Конфиденциальность** – данные не читаются неавторизованным пользователем. **Целостность информации** – 100% нетронутая информация. **Доступность информации** – доступ, когда необходимо. **Шифрование** – преобразование в целях скрытия. **Средства шифрования** – аппаратные, программные, совместные. Основные способы: **симметричное** (один ключ на шифрование и расшифрование), **асимметричное** (открытый ключ для шифровки, закрытый для расшифровки), **хеширование** (преобразование в кодовую последовательность фиксированной длины), **цифровая подпись** (обратный асимметричный). **Криптографический протокол** – набор криптографических алгоритмов. В основе набор правил для использования преобразований. Протокол устанавливает последовательность действий при передаче. Обычный протокол обеспечивает установку сеанса, выбор маршрута, обнаружение искажений и восстановление. Протокол, обеспечивающий хотя бы одну из функций безопасности – **защищенный**. Функции: аутентификация источника данных, сторон, конфиденциальность данных, невозможность отказа, целостность данных, разграничение доступа. **Классификация** по числу участников, по числу передаваемых сообщений, по целевому назначению протокола, по типу используемых криптографических систем, по способу функционирования. Групповые протоколы предполагают одновременное участие групп участников: **разделение секрета** – все группы формируют одинаковые ключи, **протокол телеконференции** – различные группы – разные ключи, **протокол групповой подписи** – нет одного – не будет ключа. **Примитивный криптографический протокол** используется как основа прикладных, решает одну задачу. **Прикладной криптографический протокол** для решения практических задач обеспечения функций безопасности (система электронного обмена данными, система электронных платежей, игровые протоколы). **Атаки**: направленные против алгоритмов, против методов реализации протоколов, против протоколов.

5. **Виртуальная машина. Назначение ВМ. Алгоритм настройки виртуальной машины в VirtualBox.**

**ВМ** – программная и/или аппаратная система для эмуляции аппаратного обеспечения. ВМ исполняет некоторый машинно-независимый код или машинный код процессора. Может **эмулировать** работу целого компьютера. Устанавливают операционные системы. Можно запустить **несколько** виртуальных машин. **Используются для**: защиты информации и ограничения возможностей программ, Эмуляции различных архитектур, оптимизации использования ресурсов, моделирования информационных систем. Кроссплатформенность, модульность, поддержка usb 2.0, поддержка 64 бит даже на 32 бит, встроенный RDP-сервер.

**Тестирование защиты программного обеспечения**

**Защита ПО** – комплекс мер на защиту ПО от несанкционированного использования, приобретения и т. д. Защита от несанкционированного **использования** – меры на противодействие нелегальному использованию. Защита от **копирования** в основном на лицензию. Средства защиты: **локальная защита** (ввод серийного номера), **сетевая** (локальная, отмена запуска «одного» приложения на двух ПК), **компакт-диски** (требование оригинала), **электронные ключи** (устройство с данными лицензии), **привязка к параметрам ПК**, **перенос в онлайн**, **защита от анализа кода** (обфускаторы запутывают код), защита на мобильных платформах (основа на невозможности чтения из ППЗУ). **Тестирование безопасности** – тест для проверки защищенности информационной системы. Меры: **конфиденциальность** (данные только «доверенным» лицам), **целостность** (точная и правильная информация), **аутентификация** (подтверждение личности), **авторизация** (дает права), **безотказность** (не отказано в получении/отправке). Типы тестирования: **сканирование уязвимостей**, **безопасности**, **тестирование на проникновение**, **оценка рисков**, **аудит безопасности**, **этический взлом**, **оценка состояния**. Протокол передачи гипертекста (**HTTP**) с 1990 года. Без установления соединения – клиент HTTP, не зависит от носителя, является безстоящим. Не полностью защищен, 80 порт, на уровне приложения, не требует шифрование, сертификаты.

6. **Динамически загружаемые библиотеки. Назначение и использование dll. Алгоритм регистрации файла \*.dll или \*.ocx в системе. Алгоритм снятия регистрации с файла \*.dll или \*.ocx.**

**DLL** – библиотека динамической компоновки. Позволяет многократное использование различными приложениями. DLL, OCX, DRV. Формат для DLL – Portable Executable. В DLL **содержатся все функции** Win32 API и много других функций. Что-то вроде библиотеки функций. DLL не связывается с процессом, процесс загружает библиотеку и использует ее.

**Многоуровневая модель качества программного обеспечения.**

**Качество** ПО – характеристика, описывающая соответствие требованиям. Определяется в стандарте ISO 9126. **Обеспечение качества** – совокупность мероприятий для обеспечения требуемого уровня качества. **Контроль качества** – совокупность действий, чтобы понять его актуальное состояние (готово, соответствует и т.д.). **Внутреннее качество** – характеристики; **внешнее качество** – поведение, качество при использовании – конкретные сценарии. Многоуровневая модель качества ПО - На верхнем уровне 6 характеристик. Каждая описывается при помощи нескольких атрибутов. **Функциональность** – способность в определенных условиях решать задачи. **Надежность** способность поддерживать определенную работоспособность в условиях. **Удобство использования** – способность быть удобным в обучении и использовании. **Производительность** – способность обеспечивать необходимую работоспособность. **Удобство сопровождения** – удобство проведения деятельности на сопровождение. **Переносимость** – способность сохранять работоспособность при переносе в окружениях. **Наборы метрик**: полнота реализации функций, корректность реализации функций. Требования определяют: что ПО должно делать, насколько должно быть надежно, удобно пользоваться, эффективно, удобно сопровождение, переносимо. Методы контроля качества: **верификация** – соответствие требованиям (так делаем или нет); **валидация** – проверка правильности продукта (то делаем или нет). Их эффективность зависит от корректности требований. Методы контроля качества только проверяют достижение характеристик. Проверка **во время работы** – в основном тестирование. **На основе симуляции** работы ПО – проверка на моделях, прототипирование.

**7. Использование виртуальной машины для выявления и устранения различного рода несовместимостей.**

7. **Аутентификация** – проверка подлинности (пароли, цифровая подпись, контрольная сумма). Односторонняя и взаимная. Криптографические способы. **ГОСТ Р ИСО/МЭК 9594-8-98** – формат информации, способ получения информации, три способа использования информации. В стандарте два вида аутентификации: **простой** – пароль, **строгий** – удостоверение личности. **Элементы**: субъект, характеристика, хозяин системы, механизм аутентификации, механизм управления доступом. **Факторы**: секретная информация, уникальный физический объект, биометрика. Способы аутентификации: электронная подпись: **неквалифицированная** – подпись с ключами, определяет лицо, изменения, **квалифицированная** – то же, что и в предыдущей + ключ в квалифицированном сертификате, соответствует требованиям закона; **по паролю**: **многоразовый** - отправил логин и пароль, получил ответ. Передается незашифрованно или с использованием SSL, TLS. Лучше использовать хеш. **Одноразовый** – генераторы чисел, случайные совместные пароли; **по SMS**: отправил логин и пароль, получил смс с кодом, отправил, получил ответ; **биометрия**: отпечатки, геометрия руки, глаза, термический образ лица, по лицу, подпись. Минэсы: Проверка идет между прошлым записанным образом и полученным сейчас. Использование муляжей. Изменение биометрических данных; **геоположение**: отправляет данные спутников, происходит отслеживание, аутентификация на местоположении выхода в интернет; **многофакторная аутентификация** – использование нескольких факторов. **Протоколы**: на ПК – шифрованное имя (логин), Password Authentication Protocol (логин-пароль), карта доступа, биометрия; в сети – Secure SNMP цифровая подпись, Security Assertion Markup Language, Cookie, Kerberos Tickets, Сертификаты X.509.

8. Групповые политики для настройки системы. Не на домашней версии. Во множестве организаций используют централизованные групповые политики на много компьютеров. Можно на сервере. Локальные групповые политики можно использовать на своем ПК. Экран входа, ограничения других пользователей. gpedit.msc.

**Риски** – негативные события и величины, отражающие потери и убытки. Оценки качества с двух позиций: положительной эффективности, негативного ущерба. Риски характеризуют возможные негативные последствия дефектов или ущерб пользователей. Характеристики дефектов и рисков помогают: оценивать реальное состояние проекта, выбирать методы и средства автоматизации тестирования и отладки, рассчитывать необходимую эффективность мер от потенциальных дефектов и невыявленных ошибок, оценивать требующиеся ресурсы ЭВМ. **Ошибка** – неправильность, искажение объекта. Предполагается, что есть эталонное состояние объекта. Эталон для ПС – спецификация требований. **Первичные ошибки** – потенциальные угрозы, **вторичные** - ошибки в результатах применения ПС. Серьезность minor, medium, major, critical. По величине ущерба: несущественные сбои, допустимые ординарные отказы, великие катастрофические отказы. Организационные дефекты из-за непонимания технологии процесса ЖЦ ПС (+лидеры). Изменения характеристик системы и внешней среды (+ моделирование с упрощением элементов). Сложности при анализе: сложность ошибок при создании корректировок; сложность проявления ошибок функционирования и получения результатов. Факторы на сложность ошибок: размер модифицируемой программы; число специалистов. Ошибки проектирования и разработки структуры ПС: пропуски, конфликты, ошибки перевода. Ошибки реализации спецификации компонентов: программные дефекты; 1 категория - дефекты ошибок пользователя при правильном наборе действий; 2 категория – ошибки неправильного графического интерфейса; 3 категория – пропущенные функции. Программные ошибки: ошибки типов операций; ошибки переменных; ошибки управления и циклов. Ошибки в документации – происходит одно, а задокументировано другое. Первичные ошибки: технологические; программные; алгоритмические; системные.

9. **Командная строка Windows. Синтаксис команд. Использование командной строки Windows для выявления и устранения различного рода несовместимостей.**

Значок «>» как указатель на то, что можно вводить команду. В командной строке текст вводятся в порядке, который описан в команде. Сначала сама команда, параметры через /, исходные файлы, конечные файлы и т. д. Повторение параметров [], необязательное {}, варианты |.

Интерфейс командной строки – текстовый интерфейс между человеком и компьютером с вводом команд. Если совсем плохо, можно в безопасном режиме открыть командную строку, там попытаться найти плохие штуки. SFC /scannow.

Файрвол в 3 билете.

10. Обновление как выполнение: обследование. Определяется **уровень обновлений системы**, подключение дополнительных функциональных возможностей. Сбор информации об **объеме внедрения** бизнес-процессов. **Подготовка плана перехода**. Подготовка тестовой системы, ее обновление, анализ и корректировка затронутого модифицированного ПО, тестирование работы системы, создание перечня мероприятий для перехода. Выполнение плана перехода. **Выполнение мероприятий** из «План перехода». **Поддержка пользователей**, решение проблем.

**Регламент обновления** – свод правил, регулирующих аспекты обновления ИС. **Структура**: Общие положения, Область действия регламента, Нормативная основа регламента, Основные используемые понятия, Порядок обновления ИС, Ответственность сторон, участвующих в обновлении ИС.

**Технические проблемы** (совместимость (устаревшее оборудование), потеря данных). **Организационные проблемы** (простой бизнес – процессов), сопротивление пользователей, юридические риски). **Финансовые проблемы** (непредвиденные траты).

**Методы предотвращения угроз надежности. Оперативные методы повышения надежности: временная, информационная, программная избыточность**

**Методы и средства**, позволяющие: создавать программные модули гарантированного качества; предотвращать дефекты проектирования за счет технологий автоматизации обеспечения ж.ц, путем систематического тестирования; удостоверять достигнутые качество и надежность; выявлять последствия дефектов. Использование **CASE-технологий** позволяет снизить уровень ошибок. **Систематическое тестирование** как основной метод измерения качества, определения корректности. **Обязательная сертификация** для удостоверения достигнутого качества и надежности функционирования ПО. **Оперативное выявление последствий дефектов** для восстановления функционирования.

**Оперативные методы повышения надежности: временная, информационная, программная избыточность:**

**Формализация** позволяет исключить ошибки и облегчить их выявление. Повышение надежности через **автоматизацию программирования**, как использование ЭВМ для составления программ. **Программная избыточность** – несколько вариантов программ для одного алгоритма. **Временная избыточность** – контроль исполнения программ и восстановление процесса. **Информационная избыточность** – резервирование исходных и промежуточных данных. Структурное программирование облегчает проектирование и повышает надежность. Структурирование данных уменьшает сложность комплекса программ и снижает вероятность ошибок.

11. **Описание основных проблем запуска компьютерных программ.**

Отсутствие NET Framework необходимой версии, Отсутствие Visual C++ и Direct X необходимой версии, Неправильная разрядность Windows, Повреждённый файл инсталляции, Отсутствие необходимой DLL-библиотеки, Некорректные сборки программ, Отсутствие прав администратора, Блокировка программами безопасности, Конфликты с не полностью удалёнными старыми версиями программ, Недостатки оборудования

**Вредоносная программа** – программа для несанкционированного доступа к информации или для воздействия на нее или ресурсы. Может: скрывать присутствие, самоуничтожиться, маскироваться, копировать себя в другие области памяти, модифицировать код других программ, искажать информацию. Основные пути: **сеть**, **внешние носители**. Основные виды: **программные закладки**. Обычно снижение быстродействия, изменение состава файлов, имитация сбоев работы; **программные вирусы**; **сетевые черви**. **Троянский конь** – дополнительная плохая часть хорошей программы. **Trojan**-: **Notifier** (оповещение об успешной атаке), **PSW** (воровство паролей), **Clicker** (несанкционированное обращение в интернет), **DDoS** (превращает ПК в атакера), **Spy** (слежка), **Backdoor** (удаленное управление), **Dropper** (инсталляторы плохие). **Вирус** – код с несанкционированным распространением и самовоспроизведением. Создает копии и внедряет их в сеть и файлы. ЖЦ вируса: попал в ПК, активировался, ищет жертву, готовит клонов, начинает вторжение. Классификация: **по среде** (файлы, загрузки, сеть), **опасность** (безвредные, малоопасные, опасные, смерть), **метод маскировки** (шифр – вирус и шифратор, метаморф –копии вируса с заменой блока команд, полиморф – шифрованный основной вирус). **Загрузочный вирус** активируется во время старта ОС, где-то в boot-секторе. **Файловые вирусы** заражают файлы. Группы: файловые, макровирусы, сетевые. Опасности: перезагрузка каналов связи, потеря данных, нарушение работы ПО, загрузка ресурсов ПК, хищение инфы. **Червь** – сетевые вредители. Крутой – **Moriss**. По протоколам: **сетевые** черви, **почтовые** черви, **IRC**-черви (Internet Relay Chat), **P2P**-черви (пиринговые сети), **IM**-черви (системы мгновенного обмена сообщениями).

12. **Тестирование ПО** – проверка соответствия между реальным и ожидаемым результатом. Одна из техник контроля качества. **Верификация** – оценка системы для определения удовлетворяют ли результаты этапа разработки его начальным условиям. **Валидация** – определение соответствия ПО ожиданиям пользователя. **План тестирования** – документ, описывающий весь объем работ по тестированию. **Тест дизайн** – этап процесса тестирования ПО. **Тестовый случай** – артефакт, описывающий совокупность шагов, необходимых для проверки тестируемой функции. **Баг репорт** – документ, описывающий ситуацию, приведшую к некорректной работе. **Тестовое покрытие** – плотность покрытия тестами требований либо кода. **Детализация тест кейсов** – уровень детализации описания тестовых шагов и требуемого результата для разумного соотношения времени прохождения к тестовому покрытию. **Время прохождения тест кейса** – от начала прохождения тест кейса до получения результата теста.

**Надежность программного обеспечения.**

**Отказ** – событие, когда выполнение функций останавливается. **Дефект** – программная аномалия. **Неисправность** – состояние, когда функции не выполняются. **Ошибка** – действие человека, с дефектом в результате. **Измерение** – определение количественного и качественного значения атрибутов. **Показатель** – значение атрибутов. **Метрика** – количественная или качественная характеристика. **Надежность ПО** – способность продукта безотказно выполнять функции. **Составляющие функциональной надежности**: **Безотказность** – свойство программы выполнять функции во время работы; **Работоспособность** – свойство программы корректно работать; **Безопасность** – свойство быть не опасной; **Защищенность** – свойство противостоять вторжениям. **Виды отказа**: **Устойчивый** – спецмероприятия по восстановлению; **Самоустраняющийся** – восстановление без вмешательства; **Перемежающийся** – многократные сбои, нужно вмешательство. **Объекты**: **Восстанавливаемые** – допускают ремонт; **Невосстанавливаемые** – не допускают ремонта. **Факторы, угрозы**: **Внутренние** (ошибки проектирования); **Внешние** (ошибки персонала при эксплуатации, искажения информации в каналах связи). Факторы на надежность: взаимодействие с внешней средой, с человеком, тестирование. Способы удостоверения: Верификация, валидация, сертификация. **Сертификация** – проверка на соответствие требованиям и нормативным документам. **Технологическая** (документы целей и методов; ссылки на руководства разработок; стандарты и нормативные документы) и **эксплуатационная** (руководства операторов, справочные, учебные руководства) документация.

13. **План внедрения** - документ, определяющий последовательность действий и временные рамки для успешного внедрения. **Включает** анализ требований предприятия, выбор подходящего ПО, обучение персонала, тестирование и внедрение. План внедрения **обеспечивает** четкость и прозрачность процесса, что помогает управлять ожиданиями и ресурсами. **Помогает** определить потенциальные риски и способы их устранения, обеспечивает координацию действий участников проекта.

**Автоматизация бизнес-процессов**: Внедрение ПО для автоматизации рутинных задач. **Управление проектами**: Использование специализированных систем для планирования, мониторинга и управления проектами. **Управление взаимоотношениями с клиентами**: Внедрение CRM-систем для улучшения взаимодействия с клиентами, повышения уровня обслуживания. **Управление ресурсами предприятия**: Интеграция ERP-систем для объединения всех бизнес-процессов в единую платформу. **Аналитика и отчетность**: Внедрение систем бизнес-аналитики (BI) для сбора, анализа и визуализации данных. **Кибербезопасность**: Разработка и внедрение решений для защиты данных и информационных систем от киберугроз. **Облачные технологии**: Переход на облачные решения для повышения гибкости, доступности и масштабируемости ИТ-инфраструктуры. **Проблемы при внедрении**: Сопротивление изменениям. Недостаток квалификации. Бюджетные ограничения. Неправильно определенные требования.

Ошибки в 8 билете.

14. **Плюсы и минусы использования консоли. Назначение и область применения консоли.**

Администрирование и конфигурация системы. Устранение разного рода несовместимостей. Консоль: тратит меньше памяти, более быстрая навигация при знании команды в сравнении с навигацией по меню, можно писать скрипты в текстовом файле те же .bat в Windows и .sh в Linux.

**Плюсы**: легкость автоматизации через скрипты и пакетные файлы; управление программами без графического интерфейса; вызов команд небольшим количеством нажатий; обращение к командам для разных исполнимых файлов; просмотр мгновенных сообщений; использование удаленного компьютера без затрат трафика; отсутствие деталей интерфейса. **Минусы**: интерфейс недружелюбный для графических пользователей; необходимость изучения синтаксиса команд и самих команд; долгий ввод без автодополнения; отсутствие аналогового ввода.

Антивирусное ПО для выявления вирусов, их удаления, лечения и профилактики. Критерии: стабильность и надежность работы, регулярное обновление базы данных, определение разнообразных типов вирусов, работа с файлами разнообразных типов, автоматическая проверка новых файлов, возможность восстановить зараженные файлы. Режимы проверки: **максимальный** (параноидальный) максимальная защита, большая трата ресурсов; **средний** (оптимальный) рекомендован; **минимальный** проверка по базе данных сигнатур (информация, если 100% вирус). **Ревизоры** – запоминают исходное и периодически сравнивают; **детекторы** – резиденты, обнаружение подозрительных действий; **доктора** – выкусывают вирус из зараженного файла; **вакцины** – модификация файла так, чтобы не ломать работу; **сторожа** – резиденты, множество операций потенциальной опасности.

15. **ЖЦ ПО** – период времени с принятия решения о необходимости создания продукта до полного изъятия из эксплуатации. Из: стадии; результат стадии; ключевые события (принятие важных решений или что-то подобное).

**Модель ЖЦ** – это структура, которая определяет последовательность выполнения действий на протяжении всего ЖЦ. Жизненный цикл включает в себя: стадии; результат стадии; ключевые события. Модели: **водопадная** предусматривает последовательное выполнение всех этапов проекта. Переход на следующий после завершения предыдущего. Требования, проектирование, реализация, тестирование, внедрение, эксплуатация и сопровождение. + полная и согласованная документация, легко определить сроки и затраты. – срань в начале минус проект; **Итерационная** – накидывать на продукт отдельные части. Итерация как плюс в возможности. – целостное понимание проекта задерживается, при итерациях удаляется часть ранней работы, добросовестность снижается; **Спиральная** – сделал фигню, ну и ладно потом доделаем. Сделал версию, потом обновил. Оценивается: риск превышения сроков и стоимости, необходимость итерации, степень понимания требований, целесообразность остановки. Особое внимание рискам. Точки: концепция, цели и содержание ЖЦ, архитектура ЖЦ, первая версия, готовое.

Анализ рисков в 1 билете.

16. **Несовместимость** – невозможность различных объектов взаимодействовать друг с другом. **Программная** – несовместимость различных видов ПО. Несколько антивирусов, устаревшие программы. В большинстве случаев устранение возможно. **Аппаратная** – несовместимость различных технических устройств. Несовместимость электропитания, сокеты. В большинстве случаев устранение отсутствует. **Информационная** – несовместимость типов данных. .doc и .docx. Много устранения. **Программно-аппаратная** – несовместимость технических устройств с программной средой. Устройство без драйвера. В большинстве возможно устранить. **Коммерческая** – захотели и сделали. Новая версия Skype, разъемы Apple. Устранение сомневается.

Разработка модулей адаптации билет 2.

17. **Совместимость** – способность различных объектов взаимодействовать друг с другом. **Аппаратная** – одно устройство работает с узлами другого. **Электромагнитная совместимость** – работа в условиях эксплуатации при воздействии помех. **Программная** – выполнение одинаковых программ с одним результатом. **Информационная** – способность двух или более систем адекватно воспринимать одинаково представленные данные.

**Обратная совместимость** – в новой версии программы есть старый интерфейс, другие объекты могут продолжать работать без изменений. Обратная совместимость позволяет сохранить предыдущий «опыт» для новых версий. Но обратная совместимость сдерживает развитие технологий. В аппаратном мире обеспечивает возможность эмулировать работу старых типов оборудования. Примеры: новый Windows и старое железо; новый «офис» и старая ОС; USB.

**Объекты уязвимости.**

**Уязвимость** – недостаток в программе, через который можно нарушить целостность. **Угрозы** – потенциальные опасности, возникающие при использовании уязвимости для атак: нарушения целостности, доступности, конфиденциальности. **Источники угрозы**: хакер; дурак сотрудник; процесс, обходящий политики безопасности. **Виды уязвимостей**: с риском – значительное воздействие, без риска – пофиг, **эксплойт** – команды для использования уязвимости; эксплуатируемые и неэксплуатируемые (без эксплойта). **Классификация** **по причине возникновения: первый тип** – уязвимости по дефектам проектирования и программирования: ошибки обработки и представления данных, некорректное использование **API** (интерфейс приложений), ошибки с **инкапсуляцией** (процесс сокрытия внутренней части, только нужное видно). Второй тип – вызванные дефектами конфигурирования и управления системой: дефекты конфигурации, настройки механизмов безопасности, аппаратного обеспечения. Распространенные типы уязвимостей: **ошибки проверки вводимых данных**: SQL-инъекция; **нарушения безопасности доступа к памяти**: переполнение буфера; **состояния гонки**: ошибки времени проверки ко времени использования; **ошибки путаницы привилегий**: подделка межсайтовых запросов в веб-приложениях. **Классификации**: **OWASP Top 10** – 10 самых опасных рисков для web-приложений; **CVE** – что-то вроде словаря известных угроз, у каждой идентификатор (год и номер); недостатки безопасности – **CWE** – перечень ошибок ПО, из-за которых возможно появление уязвимостей; **CAPEC** – как CWE, но больше контекста и дополнительной информации по работе уязвимостей; **ИСПДн** – общие описания уязвимостей систем обработки персональных данных. Ранг на основе: затронутых систем, опасности данным, угроз бизнес-функциям, простоты реализации, потенциального ущерба. Стандарт **CVSS** – общая система оценки уязвимостей. Степень опасности: низкая 0.1-3.9, средняя 4-6.9, высокая 7-8.9, критическая 9-10. Определяется по: вектору атаки (удаленность и способ), сложности атаки, взаимодействию с пользователем, требуемым привилегиям, области действия, конфиденциальности, целостности, доступности.

18. **Внедрение ПО** – настройка ПО под определенные условия использования, обучение пользователей работе с продуктом. Плоскости работ: **1** выделение критических процедур в деятельности организации. В первую очередь использовать IT-решение в этих процедурах; **2** расширение нормативной базы организации регламентами порядка выполнения процедур; **3** выполнение работ по стандартизации существующей деятельности, выделяются лучшие практики и вносятся в IT-решение по наибольшей полезности. Актуальность внедрения ПО заключается: в автоматизации, конкурентоспособности, удовлетворения требований клиента, улучшение качества принятия решений.

Результат проекта - регламентированный, документированный и автоматизированный процесс разработки и сопровождения. **Цели**: **стабильная повторяемость** (стандартизация, модификация, регламентация и документирование процессов); **снижение требований к участникам проектов** (процесс понятен настолько, что легко влиться); **сокращение сроков выполнения** (проекты становятся частью базы практик компании). **Задачи**: автоматизация ключевых процедур; создание понятных регламентов, расширение нормативной базы; внедрение лучших практик и стандартизация существующей деятельности.

Модель качества 6 билет.

19. **Тестирование ПО** – проверка соответствия между реальным и ожидаемым результатом. Одна из техник контроля качества. **Верификация** – оценка системы для определения удовлетворяют ли результаты этапа разработки его начальным условиям. **Валидация** – определение соответствия ПО ожиданиям пользователя. **План тестирования** – документ, описывающий весь объем работ по тестированию. **Тест дизайн** – этап процесса тестирования ПО. **Тестовый случай** – артефакт, описывающий совокупность шагов, необходимых для проверки тестируемой функции. **Баг репорт** – документ, описывающий ситуацию, приведшую к некорректной работе. **Тестовое покрытие** – плотность покрытия тестами требований либо кода. **Детализация тест кейсов** – уровень детализации описания тестовых шагов и требуемого результата для разумного соотношения времени прохождения к тестовому покрытию. **Время прохождения тест кейса** – от начала прохождения тест кейса до получения результата теста.

**Тестирование производительности** для определения, как быстро работает система под нагрузкой. **Нагрузочное тестирование** для оценки поведения приложения под заданной ожидаемой нагрузкой. Обычно получает время отклика важных бизнес-транзакций. Узкие места. **Стресс-тестирование** для понимания пределов приложения. Определение надежности во время экстремальных нагрузок. **Тестирование стабильности** чтобы убедиться, что приложение выдерживает ожидаемую нагрузку в течение длительного времени. Следит за потреблением памяти, проверяет деградацию скорости с течением времени. **Конфигурационное тестирование** проверяет производительность от конфигурации.

Тестирование защиты 5 билет.

20. **Программное обеспечение Веб-сервера. Клиентское программное обеспечение**

Для работы сервера нужно установить серверную часть сетевой ОС. **Сетевая защита** – ограничение доступа. Устанавливается ПО для работы сайта, обслуживания, мониторинга, защиты от взлома. На серверах для сайтов Linux, BSD. Пакеты: **веб-сервер** позволяет принимать запросы пользователей, популярный Apache; интерпретатор языка программирования; **СУБД**; **FTP-сервер** как правило для администрирования сайта; ускорители исполнения программного кода; службы работы с электронной почтой.

**ПО обеспечивает доступ к ресурсам на сервере**. **Редиректор** – сетевое ПО запросов ввода/вывода файлов, переназначает сетевым сервисам другого ПК. Есть клиентский и серверный. На представительном уровне OSI. Редиректор перехватывает запрос на ресурс, если локальный – в процессор, если удаленный – в сеть к нужному серверу. **Распределитель** – часть ПО для управления назначением букв накопителя. Создает соответствие между локальным накопителем и сетевым. При запросе на ресурс заменяет букву на сетевой адрес и отпускает запрос к редиректору.

Методы предотвращения 10 билет.

21. **Инсталляция** – процесс установки ПО на компьютер. Выполняется особой программой, входящей в ОС или в состав ПО. Большинство программ поставляются в сжатом виде. Нужно распаковать и разложить файлы в нужные места. Процессы разные – установщики свои. Варианты: **вручную** (сам), **автоматическая** (запусти и выбери), **тихая** (без окон и сообщений), **самостоятельная** (ничего не делай), **удаленная** (без монитора), **чистая** (без предыдущих версий), **непосредственная** (установка с копии на диске).

**Инсталлятор** – программа, которая устанавливает ПО на компьютер. Одни устанавливают определенное ПО, другие универсальные. Пакетный установщик обычно часть ОС. Может работать с большим количеством продуктов. Инсталлятор создается для каждого продукта отдельно. Windows – MSI, компании: InstallShield, Macrovision. UNIX – встроенные системы управления пакетами. Mac OS – пакеты. Архиваторы тоже могут создавать установщики – самораспаковывающиеся архивы.

Вредоносные программы 11 билет.

22. **Режим совместимости** нужен для запуска программ с измененными параметрами. Свойства ярлыка, совместимость. Совместимость версий системы, пониженная цветность, 640\*480, изменить параметры высокого разрешения (выбрать разрешение: исправлять размытость, использовать разрешение; режим масштабирования: приложение, система), запуск от имени администратора, изменить для всех.

Надежность ПО 12 билет.

23. **Сопровождение ПО** – процесс улучшения, оптимизации и устранения дефектов ПО. **Фаза ЖЦ**. Стандарты: **ГОСТ Р ИСО/МЭК 12207-2010** (системная и программная инженерия. процессы ЖЦ ПрСр); **ГОСТ Р ИСО/МЭК 14764-2002** (ит. сопровождение ПрСр); **IEEE 1219**. Сопровождение и поддержка не всегда синонимы. 1 – внешняя или использующая организация; 2 – только использующая. Линии сопровождения: 0 – телефонные обращения, передача тех. специалистам; 1 – консультация, настройка; 2 – функциональное сопровождение. запуск ПО на машинах заказчика; 3 – системное сопровождение, запуск ПО на оборудовании заказчика. 1-3 инженер по сопровождению.

Аутентификация 7 билет.

24. **Устранение программных несовместимостей при модульном принципе программирования**

Быстрое и эффективное создание программ: команда программистов, **1** программа разбивается на независимые части для своей функции, обеспечивается совместимость модулей. Модули: задача в модуле закончена, сложность посильна, минимум связей с модулями, проверка входа. **2** Требования: требование решение задачи=требование последний модуль; анализ, что нужно модулю для обеспечения требований; часть – входная информация и модуль, остальное в предыдущих модулях; от конца к началу, если модуль в нескольких цепочках, требования проверяются на совместимость. **3** Разработка межмодульного интерфейса. **4** После написания – комплексная отладка.

Объекты уязвимости 17 билет.

25. **Юзабилити-тест** – определение удобности объекта для применения. **Проверка эргономичности** – метод оценки удобства продукта в использовании. Говорят: иди сделай что-нибудь и скажи че как. Если проверка выявляет трудности, то разработчики дорабатывают и тестируют по новой. Для более точных данных следует: зафиксировать речь модератора и респондента, выражение лица респондента, экран компьютера респондента, события на компьютере: действия мыши, использование клавиатуры, переходы между экранами.

**Тестирование безопасности** – оценка уязвимости ПО к различным атакам. Компьютерные системы часто мишени незаконного проникновения: попытки хакеров проникнуть в систему ради интереса, месть, взлом мошенниками для наживы. Тестирование безопасности проверяет реакцию защитных механизмов. Испытатель – взломщик. Разрешено: попытки узнать пароль, атака системы с помощью утилит, подавление защиты, введение ошибок, просмотр несекретных данных. **Задача** – сделать цену взлома выше награды.

**Локализация** – адаптация к культуре какой-либо страны. Уровни: **1** обеспечение поддержки языка и национальных стандартов. Корректная работа в ОС, вывод символов языка, стандарты: форматы бумаги, времени и дробных чисел, имена, валюты, система мер. Издание документации на целевом языке; **2** перевод текстов программы. В сложном ПО не все части стоит переводить. Корректное выравнивание и размещение элементов интерфейса ввиду разного размера слов. Важен перевод терминологии. Перевод картинок и речи; **3** тонкая настройка под страну. Работа с формами слов. Дополнительные стандарты. Обеспечение возможной работы на другой локализации. Учет национального менталитета. Перерисовка графики под реалии страны. Корректировка клипарт-библиотек.

Шифрование 4 билет.