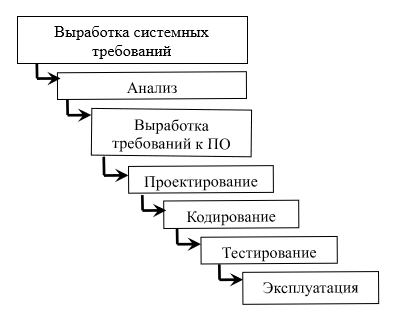
# Экзаменационный билет 1

## Понятие жизненного цикла программных средств. Модели жизненного цикла.

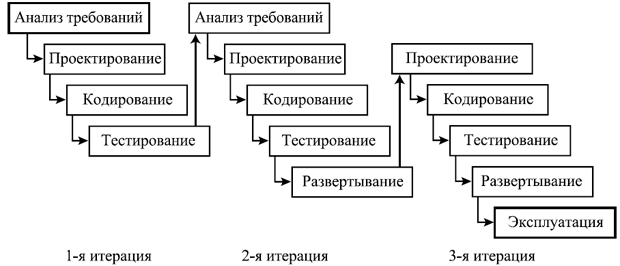
**Жизненный цикл ПО**: весь период существования ПО, с момента, когда принимается решение о его разработке до того момента, когда полностью прекращается всякое ее использование. Процессы жизненного цикла ПО по ISO 12207:

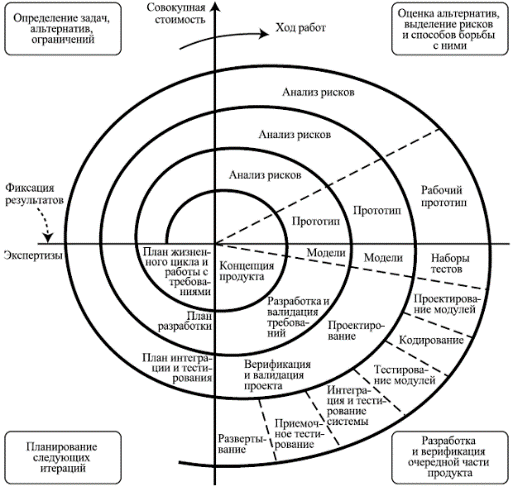
* Основные процессы: (Приобретение ПО; Поставка ПО (передача в использование); Разработка, эксплуатация; сопровождение)
* Организационные: (Управление проектом/инфраструктурой/персоналом; Усовершенствование процессов)
* Поддерживающие: (Документирование; Управление конфигурациями; Обеспечение качества; Верификация; Валидация; Аудит; Разрешение проблем)

**Модели жизненного цикла:**

**Каскадная** (водопадная): Связь между этапами показана только сверху вниз.

**Инкрементная**: Итеративная модель предполагает разбиение ЖЦ на последовательность итераций, каждая из которых напоминает «минипроект».



**Спиральная модель:** «Спиральная модель» похожа на инкрементную, но с акцентом на анализ рисков. Она предполагает 4 этапа для каждого витка: (планирование; анализ рисков; конструирование; оценка результата и при удовлетворительном качестве переход к новому витку.)

Недостаток модели состоит в том, что ошибки могут быть допущены на этапах анализа и проектирования, это может привести к дополнительным затратам или неуспеху всего проекта в целом.

## Методы тестирования программных систем. Процессы и средства тестирования программных компонентов.

Цель - обнаружить как можно больше ошибок, снижает уровень невыявленных ошибок.

Программы в процессе тестирования используются в двух формах представления:

- (символьное представление) текст на ЯП или описания, удобного для анализа человеком;

- (объектное представление) в машинном коде, пригодном для автоматической обработки исходных данных и неудобном для их анализа человеком.

**Процессы и средства тестирования программных компонентов**

**2 стратегий тестирования**: от структуры и от данных. Им соответствуют два метода тестирования программ: метод анализа потоков управления и анализа потоков данных.

При ***первой стратегии*** за основу принимается структу­ра по тексту программы в виде гра­фа. Выделяются маршруты исполнения программы и условия-предикаты. Эти условия используются для подготовки тестовых наборов. Отклонение от маршрута рассматривается как ошибка.

После устранения ошибок для каждого из маршрутов проверяется процесс обра­ботки данных, и выявляются ошибки в результатах их преобразова­ния. Затем оценивается достаточность выполненного тестирования ***по степени покрытия*** ***исходного графа программы***. Завершается тестирование при требуемом покрытии графа или при использовании ресурсов, выделенных на тестирование.

При ***второй стратегии*** за основу принимаются требования спецификаций, конкрет­ные тестовые и эталонные значения, которые подготавливаются специалистами путем анализа переменных и предикатов в тексте программы. При каждом тесте программа исполняется по определен­ному маршруту, который регистрируется. По мере тестирования отме­чаются проверенные операторы, и оценивается полнота ***покрытия тес­тами*** ***требований спецификаций*** на маршрутах тестирования. Однако, при такой стратегии трудно оценить степень покрытия и проверки всех маршрутов без использования структурного графа, построенного по ее исходному тексту.

Данная стратегия имеет преимущества при сравнительно прос­той структуре программы и при преобладании в ней вычислительных операторов. Однако возрастает риск пропустить сочетания предикатов, определяющих непроверенный маршрут. Поэтому целесообразно ***совместное применение двух стратегий***, с акцентом на одну из них в зависимос­ти от особенностей. Программы со сложной логической структуры и с ма­лой вычислительной частью целесообразно начинать тестировать по первой стратегии и только для маршрутов с вычислительными опера­торами использовать анализ потоков данных (вторую стратегию). В модулях простой структуры и со значительным объемом вычислений после первичной отладки по второй стратегии.

***Документирование процессов тестирования программных компонентов***. Должен быть описан весь процесс тестирования и результаты на каждом этапе ЖЦ: 1. исходные данные: тз и спецификации; комплекс эталонных значений; критерии качества результатов тестирования; ограничения доступных ресурсов; 2. план тестирования компонентов; 3. методы и критерии оценки достигнутого качества; 4. входные и результирующие данные тестирования; 5. распределение ответственности между специа­листами; 6. протоколы и отчеты о достигнутом качестве.

Классификация тестирования по уровням: компонентное (модульное), интеграционное, системное, приемочное.

Модульное направлено на поиск дефектов в небольших частях ПО: модулях, функциях.

Интеграционное для проверки взаимодействия между компонентами, а также взаимодействия с операционной системой, аппаратурой, внешней средой.

Системное – проверка требований в системе в целом; несогласованное или неадекватное использование ресурсов, отсутствующую или неверная функциональность, неудобство использования и т.д.

Приемочное определяет, удовлетворяет ли система приемочным критериям.

Достаточно часто используют терминологию, отражающую знание тестировщика о системе:

● Т чёрного ящика: есть доступ к ПО только через интерфейс; обычно критерий покрытия складывается из покрытия структуры входных данных и покрытия требований.

● белого ящика: разработчик теста имеет доступ к исходному коду; чаще всего применяется в случае модульного тестирования; используются метрики покрытия кода

● серого ящика: разработчик теста имеет доступ к исходному коду, но при непосредственном выполнении тестов доступ к коду, как правило, не требуется.

Также тестирование может быть классифицировано по степени автоматизации: ручное, автоматизированное, полуавтоматизированное (смесь).

# Экзаменационный билет 2

## Проблемы современной программной инженерии. Понятие сложной системы. Типы сложных систем.

**Программная инженерия** – это область компьютерной науки, которая занимается построением программных систем, настолько больших и сложных, что для этого требуется участие слаженных команд разработчиков различных специальностей и квалификаций. Она изучает различные методы и инструментальные средства разработки для достижения целей.

**Проблемы современной программной инженерии:**

* разнородность ПС
* частое отсутствие аналогов, что не позволяет использовать типовые решения
* сложность определения требований в программной системе
* разобщенность и разнородность отдельных групп разработчиков
* трудность тестирования больших программных продуктов
* сложность перехода от индивидуального программирования, к коллективному
* сложность модернизации ранее существующих программных систем.

**Понятие сложной информационной системы**

1. Система, которая разрабатывается группой разработчиков (5+ человек).
2. Если количество строк исходного кода исчисляется сотнями тысяч или даже миллионами.
3. Если сложную задачу можно декомпозировать на более простые задачи (которые будут реализованы более мелкими подсистемами).
4. Если в требованиях встречаются ***взаимоисключающие требования.*** Например, нужно обработать огромные информационные потоки, и время отклика должно быть минимальным.

## Интеграция, квалификационное тестирование и испытания комплексов программ. Подходы для интеграционного тестирования комплексов программ.

**Квалификационное тестирование:** Тестирование, выполняемое с целью убедить заказчика, что ПО соответствует заданным требованиям (ISO 12207).

**Интеграционное тестирование** ‒ объединение программного кода, соответствующих 2+ программных модулей. Это должно гарантировать, что вместе они работают, как требуется. Подходы к интеграционному тестированию:

* Снизу-вверх. Все низкоуровневые модули собираются воедино → тестируются → собирается следующий уровень. Подход полезен, если все или практически все модули, разрабатываемого уровня, готовы. Помогает определить уровень готовности приложения.
* Сверху-вниз. (модули низкого уровня симулируются заглушками).

**Испытания** компонентов и сложного комплекса программ должны включать:

* организацию и процессы испытаний: определение задач; подготовку компонентов;
* интеграцию и тестирование комплекса программ вне системы:
* испытания интерфейсов, компонентов;
* анализ корректности документации;
* приемо-сдаточные испытания программного комплекса в составе системы:
* испытания соответствия функций и качества;
* оформление акта о завершении работ и контракта на создание;
* опытную эксплуатацию;
* завершение испытаний на соответствие требованиям и утверждение готовности программного продукта для предъявления заказчику и поставки пользователям;
* внедрение версии программного продукта для применения в системе и обучение пользователей;
* изменения требований к КП, создание и сопровождение новых версий программного продукта.

Испытываться может лишь система, имеющая ТЗ. Сначала составляется общий план оценивания (сеть, ПО, оборудование, надёжность, безопасность). Затем проводятся сами испытания: проверяются функциональные характеристики (**реактивность** (способность выдерживать заданные интервалы между запуском и ответом), кол-во запросов, обрабатываемых в единицу времени), характеристики интерфейса, **отказоустойчивость**.

# Экзаменационный билет 3

## **Профили стандартов жизненного цикла систем и программных средств в области** программной инженерии.

**Профиль стандартов** – совокупность базовых стандартов с четко определенными возможностями для реализации заданной функции или группы функций.

**Цели применения профилей стандартов**:

* улучшение технико-экономических показателей проектов;
* повышение качества компонентов;
* обеспечение расширяемости и масштабируемости; переносимости между платформами.

**Профиль стандартов ЖЦ ПС** должен определять архитектуру программного комплекса.

Профиль стандартов конкретной системы развивается и конкретизируется в процессе жц и оформляется в составе документации системы. Рассматривают две группы профилей стандартов:

* ***функциональные профили***, регламентируют архитектуру и структуру объектов системы и ее компонентов; функции, интерфейсы и протоколы взаимодействия, форматы данных;
* ***технологические профили*** регламентируют процессы проектирования, разработки, применения, сопровождения и развития систем и их компонентов.

## Процессы эксплуатации и сопровождения ПС в жизненном цикле. Этапы и процедуры при сопровождении программных средств в соответствии с требованиями стандарта ISO 12207 по развитию и модификации программного продукта в жизненном цикле.

**Процесс эксплуатации ПС** ‒ использование ПС по его непосредственному предназначению. Он охватывает эксплуатацию программного продукта и поддержку пользователей в процессе эксплуатации. Работы (процедуры), обеспечивающие эксплуатацию ПС, в соответствии с требованиями ISO 12207:

1. подготовка процесса: оператор разрабатывает план эксплуатации, устанавливает процедуры получения сведений о возникающих проблемах и их решения;
2. эксплуатационные испытания (эксплуатационное тестирование): тестирование ПС в ее рабочем окружении, также обеспечение установки начального состояния компонентов системы (н-р, БД);
3. непосредственно эксплуатация системы в среде;
4. поддержка пользователя: помощь и консультации.

**Процесс сопровождения ПС** ‒ выявление и устранение дефектов, введение новых функций и компонентов, анализ состояния и корректировка документации, контроль распространения версий. По ISO 12207 работы, обеспечивающие сопровождение ПС, включают:

* **подготовку процесса**: должен быть создан план и определить процедуры, выполняемые при реализации сопровождения. Исходные данные: исходная базовая версия ПП; системные документы; предложения о модификациях и отчеты о дефектах
* **анализ проблем**
* **внесение изменений:** оператор разрабатывает и тестирует конкретные изменения ПП. Исходные данные: базовая версия; согласованные с заказчиком предложения о модификации; отчет о влиянии корректировки и выходные результаты работы по анализу изменений.
* **проверку и приемку при сопровождении;**
* **перенос;**
* **снятие с эксплуатации.**

# Экзаменационный билет 4

## Методология обеспечения качества ПС в программной инженерии. Базовые принципы стандартов качества ПО ISO 9000:2000 и ISO 15504:1-9.

**Методология обеспечения качества ПС** поддержана рядом документов и комплексом международных стандартов.

Определены в **восьми базовых принципах**, которые декларированы в стандартах **ISO 9000:2000** (Системы менеджмента качества) и **ISO 15504:1-9** (ИТ. Оценка процессов):

1. Ориентация предприятия-разработчика на потребителя-заказчика.
2. Лидерство-руководство.
3. Вовлечение персонала.
4. Процессный подход. (управлять ресурсами и деятельностью как целым).
5. Системный подход к административному управлению. (административное управление системой повышает эффективность и результативность предприятия).
6. Постоянное усовершенствование.
7. Подход к принятию решений, основанный на фактах.
8. Взаимовыгодные отношения с поставщиками.

В **ISO 15504** каждый из принципов прокомментирован комплексом действий, необходимых для их реализации в проектах.

## Управление конфигурацией в жизненном цикле программных средств. Планы, задачи и цели управления конфигурацией ПС. Отчёты о состоянии конфигурации.

**По ISO 12207** данный процесс состоит из следующих работ:

1. Подготовка процесса. «разработка плана управления конфигурацией»
2. Определение конфигурации. «определение схемы обозначения ПС и ее версий»
3. Учет состояний конфигурации.
4. Оценка конфигурации. «определение законченности и физической завершенности ПС».
5. Управление выпуском и поставка.

**Цели УК**:

Цель - документальное оформление текущей конфигурации программ и данных.

Другая задача - все лица, работающие над проектом, в любой момент его жц использовали точную информацию о всех единицах конфигурации и их взаимодействии.

**План УК** должен определять: работы по управлению конфигурацией; график их выполнения; организацию(и), ответственную(ые) за выполнение данных работ; их связь. План должен быть документально оформлен и выполнен.

**Отчеты о статусе конфигурации** должны предоставлять информацию о конфигурации и всех отклонениях. **Цель** - обеспечения информации для управления конфигурацией. Работы, связанные с отчетностью о состоянии конфигурации:

* регистрация элементов конфигурации, состояния сообщений о дефектах, хронологии изменений
* определение информации, подлежащей сопровождению, и способов регистрации и ведения отчетности.

# Экзаменационный билет 5

## ~~Профили стандартов жизненного цикла систем и программных средств в программной инженерии.~~ Профиль жизненного цикла ПС и БД на основе стандарта ISO 12207.

Билет 3.1.

В стандарте **ISO 12207** изложены общие рекомендации по преобразованию и адаптации базовой структуры процессов, ряда работ и документов.

## Базовые характеристики и субхарактеристики качества программных средств, описанные в стандарте ISO 9126:1-4:2002.

**Стандарт ISO 9126:1-4**. Модель характеристик качества ПС и компонентов состоит из 6 групп базовых показателей, каждая из которых детализирована субхарактеристиками:

* **Функциональность**:
* **пригодность** для применения по назначению;
* **правильность** реализации требований;
* **способность** к взаимодействию с компонентами и средой;
* **защищенность** – безопасность функционирования.
* **Надежность**:
* **завершенность** – отсутствие дефектов и ошибок;
* **устойчивость** при наличии дефектов и ошибок;
* **восстанавливаемость** после проявления дефектов;
* **доступность** – готовность реализации требуемых функций.
* **Эффективность**:
  + **временная эффективность**
  + **используемость** вычислительных ресурсов.
* **Практичность**:
* **понятность** функций и документации;
* **простота использования** комплекса программ;
* **изучаемость** процессов функционирования и применения.
* **Сопровождаемость**:
  + **анализируемость** – удобства для анализа предложений модификаций;
  + **изменяемость** компонентов и комплекса программ;
  + **тестируемость** изменений при сопровождении.
* **Мобильность**:
* **адаптируемость** к изменениям среды;
* **простота установки** – инсталляции после переноса;
* **замещаемость** компонентов при корректировках комплекса программ.

Характеристики, субхарактеристики можно разделить на ***три уровня детализации показателей***:

* категорийные-описательные, отражающие набор свойств и характеристики объекта – его функции, категории ответственности, защищенности и важности, которые могут быть представлены номинальной шкалой категорий-свойств;
* количественные – представляются множеством упорядоченных, числовых точек, описываемые интервальной или относительной шкалой, которые можно измерить и численно сопоставить с требованиями;
* качественные – характеризуются как (есть – нет, хорошо – плохо), устанавливаются, выбираются и оцениваются в значительной степени субъективно и экспертно.

# Экзаменационный билет 6

## Процессы сертификации в жизненном цикле программных средств. Документирование процессов и результатов сертификации программных продуктов в соответствии с рекомендациями стандартов ISO 12207, ISO 15504, ISO 9126.

**Цель сертификации ПС** - удостоверение качества продукции, гарантирование их высоких потребительских свойств.

**Результатом** положительных испытаний является сертификат соответствия ***-*** документ, удостоверяющий, что обеспечивается необходимая уверенность в том, что должным образом идентифицированная продукция соответствует конкретным стандартам.

**Документацию ПС можно разделить на:**

* технологическую - включаю­щую подробные технические описания, отражает процессы жизненного цикла.
* эксплуатационную для конечных пользователей
* Исследовательскую для руководителей, разработчиков и исследователей

В плане управления документированием каждого этапа жц целесообразно фиксировать:

* исходные данные для успешного выполнения данного этапа;
* данные о состоянии объекта и процесса разработки после завершения этапа;
* критерии оценки результатов выполненных работ и качества отчетных документов;
* состав и содержание отчетных документов, результатов завершенного этапа для использования на следующем этапе или при завершении проекта ПС.

**Документы, отражающие особенности ЖЦ конкретного программного средства:**

1. **Описание характеристик программных продуктов**, системы и внешней среды их ЖЦ, необходимых для адаптации и подготовки рабочих версий стандартов и требований проекта ПС и системы качества предприятия в соответствии с рекомендациями стандартов ISO 12207, ISO 15504 и ISO 9126.
2. **Описание целей, требований и обязательств предприятия-разработчика** в области системы качества, критериев качества процессов и продуктов разработки, поставки и поддержки всего ЖЦ ПС.
3. **Документация и средства автоматизации** проектирования, разработки, модификации, контроля и испытаний, используемых для обеспечения ЖЦ программного продукта.
4. **Планы и методики испытаний** применения и оценки эффективности процессов системы качества предприятия и программного продукта.
5. **Методики сопровождения**, идентификации компонентов программного продукта и документации, анализа и утверждения версий комплексов программ и данных.
6. **Методика конфигурационного управления**, утверждения, хранения, защиты, копирования версий программного продукта и сопровождающих документов, а также накопления и хранения, зарегистрированных в архиве предприятия данных о характеристиках качества в течение ЖЦ версий программного продукта.

## Понятие открытой системы. Стандарты POSIX.

**Открытой информационной системой** **POSIX** - система, которая реализует открытые спецификации на интерфейсы, сервисы и поддерживаемые форматы данных, достаточные для того, чтобы дать возможность

* прикладному ПО быть переносимым в широком диапазоне систем,
* взаимодействовать с другими приложениями на локальных и удаленных системах,
* и взаимодействовать с пользователями в стиле, который облегчает переход пользователей от системы к системе.

Определяющими являются **два интерфейса** между тремя базовыми компонентами:

* между прикладными программами и платформой − операционной системой (API)
* между платформой и внешним окружением (EEI).

**Свойства открытых систем**

1. Расширяемость;
2. Масштабируемость;
3. Переносимость приложений, данных и персонала;
4. Способность к интеграции;

**POSIX** регламентируют совокупность базовых унифицированных интерфейсов, специфицированных для языка С и системы UNIX. ***Основная цель*** **–** сделать программы переносимыми на уровне различных исходных языков. У каждого интерфейса существует вызывающая и вызываемая сторона, стандарты POSIX ориентированы преимущественно на формализацию вызывающей стороны.

**Задачи POSIX:**

1. содействовать облегчению и автоматизации переноса кода готовых прикладных программ на иные платформы;
2. способствовать унификации интерфейсов заранее при проектировании, а не только в процессе их реализации;
3. сохранять по возможности и учитывать все главные, созданные ранее, и используемые программные средства и компоненты;
4. определять необходимый минимум интерфейсов компонентов для ускорения создания и расширения программных продуктов

**Новую версию POSIX** составляют четыре части стандартов ISO 9945:1-4:2003 (ИТ. Интерфейсы переносимых операционных систем).

1. Базовые определения.
2. Системные интерфейсы.
3. Команды управления и сервисные программы.
4. Обоснование.

# Экзаменационный билет 7

## Управление проектами программных средств в системе СMMI

**Назначение методологии СММ/CMMI** (системы и модели оценки зрелости) - предоставить рекомендации для выбора стратегии повышения качества путем:

* анализа производственной зрелости
* оценивания факторов, влияющих на качество ЖЦ
* выделения процессов, требующих модернизации.

В методологии СММ выделены 5 уровней зрелости.

Зрелость процессов - степень управляемости, возможность поэтапной оценки качества, эффективности результатов.

**Уровни зрелости по CMM**:

* Уровень 1 - Начальный – маленький проект (несколько тысяч строк), основные процессы ЖЦ не регулируются и не выполняются должным образом, результат не предсказуем, успех зависит от руководителей и исполнителей
* Уровень 2 - Управляемый – 10-ки тысяч строк, процессы планируются заранее, внедрение контролируется для получения предсказуемого результата.
* Уровень 3 - Определенный – для проекта с высокими требованиями к качеству
* Уровень 4 - Предсказуемый – для проекта с жесткими сроками и мерами по предотвращению и выявлению дефектов, на этом уровне должна применяться система поэтапной оценки качества продукта.
* Уровень 5 - оптимизационный – все процессы на предприятии должны быть усовершенствованы

## Оценивание надежности и безопасности функционирования сложных программных средств. Оценивание эффективности использования ресурсов ЭВМ.

**Оценивание надежности** включает измерение количественных субхарактеристик и их атрибутов: устойчивости к дефектам, восстанавливаемости и доступности. Измерения проводятся при функционировании готового продукта.

● Сложность выявления редких отказов и высокая стоимость экспериментов приводят к малому количеству зарегистрированных отказов => недостоверная оценка надежности.

● Для выявления тенденции изменения показателей надежности, их зарегистрированные значения необходимо связывать во времени с моментами корректировки программ и данных.

● При заключительных приемо-сдаточных и сертификационных испытаниях для достоверного определения надежности организуются многочасовые и многосуточные прогоны в реальной и/или имитированной внешней среде в условиях широкого варьирования исходных данных с акцентом на стрессовые ситуации, стимулирующие проявления угроз надежности.

При форсированных испытаниях целесообразно выделять следующие режимы тестирования:

● полное искажение, предельные и критические значения ключевых параметров каждого типа;

● предельно большие и малые интенсивности суммарного потока;

● умышленные нарушения пользователями определенных положений инструкций и рекомендаций эксплуатационной документации.

**Оценивание ресурсной эффективности** состоит в измерении количественных субхарактеристик и их атрибутов: временной эффективности и используемости ресурсов. При этом предполагается, что в техническом задании и спецификации требований зафиксированы требуемые значения атрибутов и их приоритеты.

В стандарте ISO 9126:2 (ИТ. Оценка программной продукции. Характеристики качества и руководства по их применению) эту характеристику качества ПС рекомендуется отражать атрибутами, каждый из которых оценивать для средних и наихудших сценариев.

Для измерения атрибутов временной эффективности необходимы инструментальные средства, встроенные в операционную систему. Эти средства должны в реальном времени регистрировать: загрузку вычислительной системы; значения интенсивности потоков данных; длительность исполнения заданий; время ожидания результатов (отклика); заполнение памяти обмена. Значения этих характеристик зависят не только от ПС, но и от особенностей архитектуры и ОС.

# Экзаменационный билет 8

## Стандарт ISO 14252:1996 − Руководство по POSIX окружению открытых систем. Понятие открытой системы. Концепция стандартов POSIX, её цели и задачи.

Билет 6.2

## Основные факторы, определяющие качество сложных программных средств. Метрики характеристик качества,описанные в стандарте ISO 9126:1-4:2002.

**Метрики** общего представления о качестве:

1. внутреннее качество, проявляющееся в процессе разработки и других промежуточных этапов ЖЦ ПС;
2. внешнее качество, заданное требованиями заказчика в спецификациях и определяется характеристиками конечного проекта;
3. качество при использовании в процессе нормальной эксплуатации.

**Применение метрик:**

**Внутренние метрики** – в ходе проектирования и программирования компонентов, цель – обеспечить достижение внешнего качества.

**Внешние метрики** – в ходе испытаний, эксплуатации и наблюдения исполняемых программ, цель – обеспечить заказчикам, пользователям и разработчикам возможность анализировать качество ПС в ходе эксплуатации.

**Метрики качества в использовании** – при оценке удовлетворения потребностей пользователей, цель – интегральная оценка результатов.

Билет 5.2

# Экзаменационный билет 9

## Понятие сложной системы. Цели и принципы системного проектирования сложных программных средств (ПС). Процессы системного проектирования ПС.

Понятие сложной системы – Билет 2.1

**Проект** - комплекс организованных мероприятий по достижению единой цели для создания сложной информационной системы с требуемыми характеристиками качества при ограниченных ресурсах.

**Цель системного проектирования** - подготовить, обосновать и согласовать решения между заказчиком и потребителем и учесть все требования к качеству и ограниченные ресурсы.

Результатом этих работ должны быть системный проект, техническое задание и контракт на продолжение разработки ПС или решение о её нецелесообразности и прекращении.

**Процессы системного проектирования ПС:**

1. анализ предметной области;
2. Разработка исходных требований для тз
3. Предварительный анализ и моделирование процессов обработки данных;
4. Концепция создаваемого комплекса программ (На основе этого описания формируется предварительное тз);
5. Стратегическое планирование проекта (долгосрочные цели развития ЖЦ, отражающие безопасность функционирования, верификации и тестирования, управления конфигурацией и сопровождения);
6. Системный проект (требования к функциям и характеристикам качества комплекса программ, описание и графическое представление его архитектуры, БД и взаимодействия компонентов, предполагаемую модель ЖЦ, предварительные планы работ; проекты тз и контракта на проектирование и весь ЖЦ).

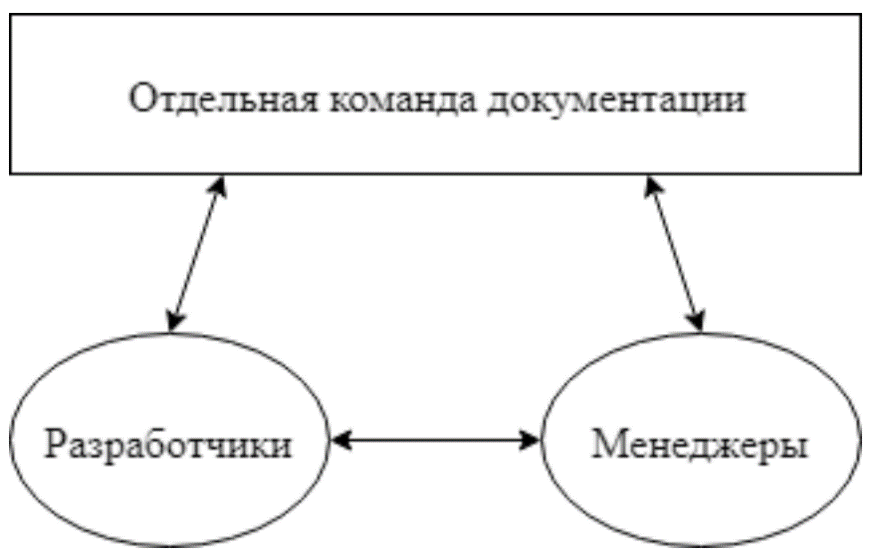
## ~~Оценивание надежности и безопасности функционирования сложных программных средств. Оценивание эффективности использования ресурсов ЭВМ.~~

Билет 7.2.

# Экзаменационный билет 10

## Организация документирования программных средств. Типы документаций.

**Документация**– частью программного продукта и требуются значительные ресурсы для ее создания и применения. Документы должны быть корректными, структурироваными и понятно изложены.

Процессы документирования программ и данных входят в жизненный цикл сложных систем. Поэтому организация и реализация работ по созданию документов должны распределяться между специалистами, ведущими создание комплексов программ и специалистами, осуществляющими разработку, контроль и издание документов.

**Типы документаций** Билет 6.1

## Задачи, решаемые с помощью машинного обучения и искусственного интеллекта. Обоснование необходимости использования *объяснимого искусственного интеллекта (XAI*). Существующие решения для XAI. Проблемы, связанные с развитием и использованием XAI.

Все задачи относятся к одной из категорий:

* + Задача регрессии – прогноз на основе выборки объектов с различными признаками. На выходе должно получиться вещественное число.
  + Задача классификации – получение категориального ответа на основе набора признаков. (в формате «да» или «нет»).
  + Задача кластеризации – распределение данных на группы.
  + Задача уменьшения размерности – сведение большого числа признаков к меньшему (обычно 2–3) для удобства их последующей визуализации.
  + Задача выявления аномалий – выявление отклонений от стандартных случаев. Схожа с задачей классификации, но более сложна в обучении.

Модели искусственного интеллекта/машинного обучения — это математические алгоритмы, которые с помощью вклада человека «обучаются» на наборах данных воспроизводить решение, которое эксперт примет при анализе такого же набора данных. Обучившись воспроизводить решение, модель функционирует самостоятельно, таким образом автоматизируя решение задач.

**Объяснимый искусственный интеллект** (Explainable AI, XAI) – модель, которая могла бы в перспективе объяснять механизмы, лежащие за алгоритмами машинного обучения.

**Обоснование необходимости использования объяснимого искусственного интеллекта (XAI)**

Модели глубокого обучения и нейронные сети работают на основе скрытых слоев, но демонстрируют результаты лучше, чем базовые алгоритмы машинного обучения. Использование таких моделей в сферах с повышенным уровнем ответственности за принятые решения привело к созданию концепции объяснимого искусственного интеллекта, в рамках которой исследуются методы анализа или дополнения моделей ИИ, позволяющие сделать внутреннюю логику и выходные данные алгоритмов прозрачными и интерпретируемыми для человека.

Предполагается, что объяснимость ИИ будет включать в себя три составляющие: симулируемость, разложимость, алгоритмическую прозрачность:

* Симулируемость - возможность анализа модели человеком
* Разложимость - способность объяснить каждую из частей модели (входные данные, параметры и выходные данные)
* Алгоритмическая прозрачность - способность пользователя понять процесс, которому следует модель ИИ, чтобы произвести любой заданный вывод из ее входных данных

**Существующие решения для XAI:** AIX360, Интерпретируемость модели в Microsoft Azure, Google Cloud’s AI Explanations, Платформа Thales XAI, ELI5, Interpret от InterpretML, Пакет DeepExplain, Пакет IML.

**Проблемы, связанные с развитием и использованием XAI:** одной из главных проблем - создание более понятной модели машинного обучения может в конечном итоге ухудшить качество принимаемых ею решений. Потому что, более сложные модели обладают гораздо большей гибкостью, чем их более простые аналоги, что позволяет аппроксимировать более сложные функции.

# Экзаменационный билет 11

## Верификация, тестирование и оценивание корректности программных компонентов. Цели, задачи и назначение. Виды и методы тестирования.

**Верификация** — процесс определения, выполняют ли программные средства требования, наложенные на них в этапах ЖЦ ПС.

**Основная цель верификации** ПС – обнаружение, регистрация и устранение дефектов и ошибок, внесённых во время разработки и модификации программ. Верификация должна быть интегрирована как можно раньше с процессами проектирования, разработки и сопровождения. Обычно проводится сверху-вниз (от общих требований до детальных).

**Назначение верификации ПС** – последовательно проверить, что:

● общие требования корректно переработаны;

● требования высокого уровня правильно переработаны в архитектуру ПС;

● требования к компонентам низкого уровня корректно переработаны в исходные тексты программных модулей;

● исполняемый объектный код удовлетворяет требованиям к исходному тексту программных компонентов.

**Тестирование**:

**Виды:**

***Тестирование для обнаружения ошибок*** имеет целью выявление отклонений в функционировании реальной программы от заданных требований. Задача состоит в обнаружении максимального числа дефектов программы. Успешным является тестирование, которое приводит к обнаружению существования ошибок при минимальных затратах.

***Тестирования для диагностики и локализации ошибок*** предназначено для того, чтобы точно установить исходное место искажения программ или данных. На этой стадии затраты оправданы и тестирование можно считать успешным, если оно привело к определению места, подлежащего непосредственной корректировке.

**Нагрузочное тестирование**

## Национальная стратегия развития искусственного интеллекта и нейротехнологий на период до 2030 года. Стандарты по защите интеллектуальной собственности в области программной инженерии.

**К 2024** - повышается уровень участия российских специалистов в международном обмене знаниями, должны быть созданы широко применяемые в мире открытые библиотеки и ПО, в которых используются ИИ.

**К 2030** - Российские организации, разрабатывающие ПО с ИИ, должны войти в группу лидеров на мировом рынке.

**К 2024** российским организациям должны быть доступны наборы данных, которые соответствуют методологиям их сбора и разметки и хранятся на общедоступных платформах. При этом данные, доступ к которым ограничен федеральными законами, должны быть надежно защищены.

**К 2030** объем опубликованных на общедоступных платформах наборов данных должен быть достаточным для решения всех актуальных задач в области ИИ.

К **2024** должны быть разработаны российские микропроцессоры, не уступающие мировым аналогам по скорости и энергоэффективности.

К **2030** на российском и международном рынках должны быть широко представлены наши микропроцессоры.

К **2024** должно увеличиться число граждан, имеющих компетенции в области ИИ, в том числе имеющих ученую степень. РФ должна стать привлекательной для их трудоустройства.

К **2030** в России должны реализовываться образовательные программы мирового уровня для подготовки специалистов. Дефицит специалистов должен быть устранен, в том числе за счет привлечения ведущих иностранных специалистов, имеющих ученую степень.

# Экзаменационный билет 12

## Цели и процессы технико-экономического обоснования проектов программных средств. Методика 1 – экспертное технико-экономическое обоснование проектов программных средств.

**ТЭ-обоснование проекта** – обоснование его рентабельности на основе рассчитанных технико-экономических показателей.

**Основные ТЭП (технико-экономические показатели):** трудозатраты;время;число специалистов.

ТЭ для ЖЦ разделяет его на две части:

* ТЭ анализа и конструирования зависит от функциональных характеристик объекта и технологий;
* ТЭ эксплуатации, сопровождения зависит от потребительских характеристик, внешних факторов.

Для прогнозирования ТЭП используются данные 2-х типов:

* Функции самого объекта;
* Экономические характеристики прототипов, пилотных проектов в той же области.

Необходимо привлекать заказчика к оценке ТЭП и производить учет затрат для:

* Корректировка оценок и учет ошибок;
* Корректировка методов оценивания;
* Обновление оценок затрат при изменении характеристик ПС;

Измерение размера ПС является ключевым фактором в оценке ТЭП, далее по вычисленным ТЭП формируется график работ.

Исходные данные для оценки ТЭП новых проектов:

* обобщенные характеристики использованных ресурсов и технико-экономические показатели завершенных разработок;
* реализованные и обобщенные перечни выполненных работ, и реальные графики проведенных ранее разработок

Группы факторов, влияющие на ТЭП при разработке программ:

* требования к его функциональным характеристикам и к качеству;
* организация процесса разработки;
* технологическая среда;
* аппаратурные вычислительные средства.

**Методика 1**

Цель – решение о продолжении разработки ПС, либо изменение, некоторых ТЭП и выделяемых ресурсов, либо прекращение проектирования данного ПС.

Если всё ок, то может реализовываться методика, состоящая из следующих шагов:

* определение класса ПС, сложности, функций;
* экспертная оценка размера – числа строк с учетом повторно-используемого кода и языка;
* оценка стоимости и длительности одно LOC (строки);
* расчет полных трудозатрат и длительности, среднего числа специалистов;
* обобщение основных ТЭП и полной стоимости, ТЭ-обоснование рентабельности.

## Организация планирования жизненного цикла сложных программных средств в соответствии со стандартами ISO 12207 и ISO 16326.

**Цель планирования ЖЦ ПС** - определить способ создания ПС, который способен удовлетворить требования тз, тс и контракта, обеспечить уровень качества. Эффективное планирование - определяющий фактор высокого качества всего ЖЦ ПС.

**В стандартах ISO 12207 и ISO 16326** рекомендуется определить администратора, который должен подготовить планы для выполнения процессов. Планы должны содержать описания соответствующих задач и охватывать **следующие задачи**:

* установление графиков решения частных задач и всего ПС;
* оценки трудозатрат на задачи и проект в целом;
* определение ресурсов, необходимых для выполнения задач и проекта;
* распределение задач по исполнителям;
* определение обязанностей исполнителей;
* определение критических ситуаций ЖЦ ПС;
* установление используемых в процессах ЖЦ ПС критериев управления качеством;
* определение затрат, связанных с реализацией каждого процесса;

Стандартами ISO 16326 и ISO 90003 рекомендуется в процессе планирования ЖЦ ПС подготовить и утвердить содержание следующих планов:

* разработки компонентов и всего ЖЦ ПС
* верификации и тестирования
* реализации процессов интеграции компонентов в версии комплекса программ;
* сопровождения и управления конфигурацией ПС
* внедрения версий ПС для конкретных пользователей, включая их подготовку и обучение;
* документирования процессов и результатов ЖЦ ПС, создания и выпуска технологической и эксплуатационной документации;
* управления и обеспечения качества ПС

# Экзаменационный билет 13

## Организация разработки требований к сложным программным средствам. Процессы разработки требований к характеристикам сложных программных систем.

**Формализация и управление требованиями** – это метод организации и документирования требований к системе, а также процесс, в ходе которого обеспечивается соглашение между заказчиком и специалистами.

**Команда должна** применить методы, чтобы понять решаемую проблему заказчика до начала разработки. Для этого следует:

- достигнуть соглашения между заказчиком и разработчиком по определению проблемы, целей и задач проекта;

- выделить проблемы-источники, стоящие за основной проблемой проекта системы;

- выявить заинтересованных лиц и пользователей;

- понять ограничения, которые будут наложены на проект, команду.

**Понимание потребностей пользователей**.

* анкетирование, формулирование популярных потребностей заказчика и пользователей;
* совещания, формулирование целей продукта; обязательно убедиться в согласии заказчиков;
* мозговой штурм и отбор идей, чтобы: выявить и/или уточнить функции проекта, определить приоритеты, риски;
* по возможности создание временных прототипов на основе первичных требований.

Целесообразно анализ и разработку требований к качеству ПС проводить в два этапа:

* общие требования,
* более конкретные требования.

**Выбор требований к характеристикам** и атрибутам качества при проектировании ПС начинается с определения исходных данных. Необходимо определить особенности проекта:

- назначение и основные функции создаваемого ПС;

- комплект стандартов и их содержание, которые целесообразно использовать при выборе характеристик качества;

- реальные ограничения всех видов ресурсов проекта.

**На этапе проектирования** последовательно должны определяться требования:

* концепции – Формализация исходных данных;
* предварительное проектирование – выбор требований к свойствам и значениям;
* детальное проектирование – подробные требования к атрибутам качества с детальным учетом и распределением реальных ограниченных ресурсов.

## ~~Верификация, тестирование и оценивание корректности программных компонентов. Цели, задачи и назначение. Виды и методы тестирования.~~

Билет 11.1

# Экзаменационный билет 14

## ~~Основные факторы, определяющие качество сложных программных средств. Метрики характеристик качества,~~~~описанные в стандарте ISO 9126:1-4:2002.~~

Билет 8.2

## Процессы переноса программных средств и баз данных,стандартизированные в ISO 14764, который детализирует требования к процессам переноса, определенным в базовом стандарте на жизненный цикл ПС (ISO 12207).

**Под переносимостью понимается**:

* процессы переноса программ и данных из одной среды в иную по архитектуре и характеристикам среду с сохранением их целостности или небольшими изменениями функций системы;
* процессы повторного использования готовых программных компонентов и информации баз данных для расширения и изменения функций системы.

**Процессы переноса** ПС и БД регламентируются документов ISO 14764, который детализирует требования к процессам переноса. Специалисты, которые проводят перенос по рекомендациям этих стандартов, должны разработать план переноса, известить пользователей, обучить персонал, выдать предупреждения о завершении переноса, оценить влияние новой версии и внешней среды и архивировать соответствующие данные.

**В содержание плана** должны быть включены:

* анализ и формирование требований к результатам переноса;
* разработка (или приобретение) инструментальных средств для выполнения переноса;
* настройка программного продукта и данных к новым условиям и среде эксплуатации;
* выполнение процессов переноса;
* верификация и тестирование результатов переноса;
* обеспечение последующей поддержки прежней среды и программного продукта.

**Сопроводитель должен также предоставить пользователям план**, процедуры и график переноса и решить следующие задачи:

* определить все компоненты, затрагиваемые переносом;
* отработать обратную связь и информацию с заказчиком и пользователями;
* определить специфику пользователей;
* опубликовать график переноса.

Сопроводитель должен выполнить следующие работы по обучению персонала.

После завершения запланированного переноса должны быть посланы соответствующие **уведомления всем заинтересованным сторонам**. Все связанные с прежней средой документы, журналы регистрации и программы следует поместить в архивы. После завершения переноса целесообразно провести итоговый анализ для оценки влияния перехода к новой среде на различные аспекты эксплуатации перенесенного программного продукта. Результаты анализа должны быть разосланы соответствующим заинтересованным.

**Отчетными результатами работ** по переносу ПС и БД являются:

* перенесенный программный продукт на новой платформе;
* план реализации переноса;
* инструментальные средства для переноса;
* извещения о намерениях по переносу;
* уведомление о завершении переноса;
* архивные данные процессов и результатов переноса.

# Экзаменационный билет 15

## Дефекты, ошибки и риски в жизненном цикле программных средств. Типы ошибок и дефектов. Риски в жизненном цикле сложных программных средств.

**Риски -** негативные события, отражающие убытки от процессов, вызванные дефектами при проектировании требований, а также при последующих этапах.

**Возможные риски в ЖЦ сложных ПС**:

● Срыв сроков проекта.

○ Провести анализ сроков;

○ Оценка стоимости

● Ошибки проектирования.

○ Провести анализ разработчиков;

○ Провести анализ ещё раз;

○ Формулировка дополнительных условий ;

○ Ранние пользовательские прототипы;

● Разработка ошибочного интерфейса (плохая связь с заказчиком).

○ Обсудить с заказчиком стилевые и функциональные особенности интерфейса.

● Потеря прибыльности (неумение заключать договор, не учли конкурентов на рынке).

○ Снижать требования к продукту (сократить функции продукта => снизится стоимость продукта).

* **Ошибка в программе** - неправильность, неумышленное искажение объекта. Любое отклонение результатов следует квалифицировать как ошибку – дефект в программе, наносящий некоторый ущерб

**Типы ошибок и дефектов**:

* Первичные в программах и данных — потенциальные угрозы проекту программного средства
* Вторичные – найденная ошибка Фиксируются при тестировании.

Вторичные ошибки являются определяющими для эффективности функционирования программ, однако не каждая первичная ошибка вносит заметный вклад в выходные результаты. Вследствие этого ряд первичных ошибок может оставаться не обнаруженным и, по существу, не влияет на функциональные характеристики ПС.

**Классификация ошибок:**

● «1-3 приоритет» - небольшие ошибки. Они не влияют отрицательно на работу программы.

● «4-7 приоритет» - умеренные, средние ошибки. Влияют конечного пользователя, но имеют малые последствия.

● «8-10 приоритет» - критические ошибки. Ошибки с высоким влиянием. Напр. сбой всей системы. Потеря важных данных. Крах системы, не помогает “restart”.

**Причины ошибок:**

* Ошибки корректности формирования и планирования выполнения требований к ПС
* Ошибки проектирования и разработки структуры ПС
* Системные ошибки в ПС
* Алгоритмические ошибки
* Ошибки реализации спецификаций компонентов
* Программные ошибки модифицированных компонентов
* Ошибки в документации модификаций
* Технологические ошибки

Первичные ошибки в ПС можно разделить на следующие группы:

● ошибки, обусловленные сложностью компонентов и ПС в целом;

● ошибки вследствие большого масштаба комплекса программ и завышенных требований качества;

● ошибки планирования и корректности требований модификаций;

● ошибки проектирования;

● системные ошибки, обусловленные отклонением функционирования ПС в реальной системе;

● алгоритмические ошибки, связанные с неполным формированием необходимых условий решения и некорректной постановкой целей функциональных задач;

● программные ошибки, в исходных кодах или при трансляции в объектный код;

● ошибки в документации;

● технологические ошибки подготовки физических носителей и документации, установки ПС на оборудование, вывода результатов работы.

## *~~Стандарт~~* ~~ISO 14252:1996 − Руководство по POSIX окружению открытых систем.~~

Билет 8.1

# Экзаменационный билет 16

## ~~Понятие сложной системы. Цели и принципы системного проектирования сложных программных средств. Процессы системного проектирования программных средств~~. ~~Разработка требований к программным средствам.~~ Создание технического задания.

First part - Билет 9.1.

Разработка требований к программным средствам – Билет 13.1

**Создание технического задания**

**ГОСТ рекомендует следующие разделы:**

* общие сведения;
* назначение и цели создания (развития) системы;
* характеристика объектов автоматизации;
* требования к системе;
* состав и содержание работ по созданию системы;
* порядок контроля и приемки системы;
* требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие;
* требования к документированию;
* источники разработки.

## Задачи и особенности объектно-ориентированного проектирования программных средств. Основные стили и парадигмы программирования.

**Объектно-ориентированное проектирование (ООП)** предназначено организовывать программные системы с большими базами данных на основе описаний объектов реального мира, важных для пользователей.

Под процессом ООП подразумевается проектирование классов объектов и взаимоотношений между этими классами. Процесс проектирования состоит из нескольких этапов:

* определение рабочего окружения и разработка моделей ее использования;
* проектирование архитектуры;
* определение основных объектов
* разработка модели архитектуры комплекса программ;
* определение и документирование интерфейсов объектов.

**Основные особенности ООП:**

* компоненты системы представляются как объекты со своими состояниями и операциями;
* объекты предоставляют методы другим объектам;
* объекты могут быть реализованы последовательно и параллельно, параллельный объект может быть пассивным, у которого состояние изменяется только через его интерфейс, или активным, который может изменять свое состояние без вмешательства извне;

Использование методов ООП строго регламентировано, поэтому:

* возрастает производительность труда разработчиков;
* компоненты системы легко изменяются и применяются повторно;
* требования проще отслеживаются;
* разработка проекта отличается непрерывностью в представлении объектов — одни и те же типы диаграмм применяются как при анализе, так и на этапе разработки;

ООП — часть объектно-ориентированного процесса разработки системы. Этот подход подразумевает выполнение трех этапов:

* объектно-ориентированный анализ
* объектно-ориентированное проектирование;
* объектно-ориентированное программирование.

Использование объектного подхода повышает уровень унификации разработки и пригодность для повторного использования не только программных компонентов, но и больших комплексов программ, что ведет к переходу к сборочному созданию программных продуктов.

**Основные парадигмы программирования:**

* процедурное: последовательно выполняемые операторы можно собрать в подпрограммы (C);
* объектно-ориентированное: (C++, C#, Smalltalk);
* функциональное: процесс вычисления трактуется как вычисление значений функций (Lisp, Haskell);
* логическое: (Пролог).

# Экзаменационный билет 17

## ~~Планирование жизненного цикла программных средств. Цели и задачи планирования~~. ~~Стандарты ISO 16326 и ISO 90003.~~ Планирование процессов управления качеством сложных программных средств.

Планирование жизненного цикла программных средств. Цели и задачи планирования. Стандарты ISO 16326 и ISO 90003 - Билет 12.2

До начала разработки в процессе формирования требований ТЗ нужно указать методики обеспечения качества.

В плане управления качеством ПС должны быть отражены:

* цель, требования к значениям характеристик качества, область действия требований и условия их применения;
* методы достижения заданных значений; действия, связанные с отчетностью об ошибках, системой корректирующих действий;
* ресурсы, базовые документы и стандарты, используемые для обеспечения качества;
* средства автоматизации разработки;

## Ключевые технические характеристики основных направлений развития технологий искусственного интеллекта, которые определяют их качество и конкурентоспособность.

1. **Производительность и скорость обработки**: СИИ должны обрабатывать большие объемы информации в реальном времени и обеспечивать высокую скорость выполнения вычислений.

2. **Точность и надежность**

3. **Адаптивность и обучаемость**: адаптация к изменяющимся условиям и обучению на новых данных.

4. **Интерпретируемость и объяснимость**: Способность объяснить и интерпретировать принимаемые решения.

5. **Этические и социальные аспекты**: Это включает вопросы приватности данных, справедливости, ответственности и безопасности.

# Экзаменационный билет 18

## **~~Методология обеспечения качества ПС в программной инженерии. Базовые принципы стандартов качества программного обеспечения в соответствии со стандартами~~** ~~ISO 9000:2000 и ISO 15504:1-9.~~

Билет 4.1

## ~~Организация документирования программных средств. Типы документаций.~~

Билет 10.1

# Экзаменационный билет 19

## **~~Методы тестирования программных систем. Процессы и средства тестирования программных компонентов~~**~~.~~

Билет 1.2.

## Управление ресурсами в жизненном цикле программных средств (ПС). Ресурсы для обеспечения функциональной пригодности при разработке сложных ПС.

**Доступные ресурсы -** финансовые, временные, кадровые и аппаратные

**Время** разработки - невосполнимый ограниченный ресурс.

**Вычислительные ресурсы**. В процессе проектирования нужно выделять ресурсы ЭВМ на обеспечение качества.

**Обобщенные ресурсы** - трудовые, временные и материальные затраты.

**Специалисты**.

**Обеспечение функциональной пригодности** – цель при использовании ресурсов в ЖЦ ПС. Необходимость выполнения ряда требований к остальным характеристикам качества приводит к тому, что использование ресурсов на их реализацию может превышать базовые затраты на обеспечение функциональной пригодности. В то же время затраты на выполнение этих требований направлены на повышение функциональной пригодности.

Для анализа затрат ресурсов их целесообразно разделить на 2 части:

● затраты на создание программных компонентов, обеспечивающих базовые свойства функциональной пригодности комплекса программ для его применения пользователями, в соответствии с требованиями контракта и тз;

● дополнительные затраты для улучшения функциональной пригодности

Наиболее универсальная единица трудозатраты специалистов в **человеко-днях** или **человеко-месяцах**.

# Экзаменационный билет 20

## ~~Дефекты, ошибки и риски в жизненном цикле программных средств. Типы ошибок и дефектов. Риски в жизненном цикле сложных программных средств.~~

Билет 15.1

## ~~Процессы управления конфигурацией программного обеспечения в соответствии со стандартами ISO 12207~~ и ISO 15846.

Билет 4.2

**Стандарт ISO 15846** детализирует положения, представленные в стандарте ISO 12207.

# Экзаменационный билет 21

## Методика оформления отчетов о выявленных дефектах, ошибках и предложениях по корректировке версий ПС.

**Методика** оформления отчетов о выявленных дефектах **должна содержать**:

* рекомендации по выявлению, регистрации условий проявления
* содержания дефектов
* содержания желательных модификаций.

Эта методика должна включаться в состав эксплуатационной документации и передаваться каждому пользователю версии программного продукта.

**Отчеты о выявленных дефектах должны содержать:**

* подробное описание сценария тестирования и исходных данных, при которых выявлен дефект;
* описание проявления дефекта и документы результатов его регистрации;
* предположение о возможной причине, вызвавшей проявление дефекта;
* предложение о возможной модификации для устранения дефекта.

**Описание выявленных дефектов** должен содержать отчеты пользователей о выявленных дефектах и дополнительно:

* оценки сложности, трудоемкости, эффективности и срочности модификаций программ;
* оценки возможного влияния предлагаемых изменений на эксплуатацию версий ПС.

Для каждого изменения должны документироваться содержательная аннотация, а также общие характеристики и достигнутые на предварительных испытаниях показатели качества очередной версии программного продукта. Результаты испытаний версии запоминаются вместе с условиями и параметрами, при которых они формировались, а также с набором тестов или указаниями места их хранения.

**Описание подготовленных** и утвержденных корректировок должно содержать:

* причину изменения программ и базы данных (ошибка, дефект, совершенствование);
* содержание изменений программ и базы данных, а также документации на версию;
* результаты квалификационного тестирования базовой версии программного продукта с предполагаемыми изменениями;
* результаты испытаний и обобщенные характеристики качества базовой версии программного продукта после внесения изменений;
* решение по распространению пользователям проведенной модификации или версии программного продукта;
* адрес хранения корректировок, документов и квалификационных тестов новой базовой версии программного продукта.

## ~~Интеграция, квалификационное тестирование и испытания комплексов программ. Подходы для интеграционного тестирования комплексов программ.~~

Билет 2.2

# Экзаменационный билет 22

## ~~Процессы эксплуатации и сопровождения ПС в жизненном цикле. Этапы и процедуры при сопровождении программных средств в соответствии с требованиями стандарта ISO 12207 по развитию и модификации программного продукта в жизненном цикле.~~

Билет 3.2.

## Характеристики качества баз данных. ~~Модели качества по стандарту ISO 9126:1-4:2002~~.

**Первым компонентом** является комплекс программ СУБД. В**ажнейшими характеристиками качества СУБД являются требования к функциональной пригодности для процессов формирования и изменения информации в** БД администраторами, а также доступа к данным и представления результатов пользователям. В зависимости от области применения СУБД, приоритет требований к качеству может отдаваться различным характеристикам: надежности и защищенности (финансовая сфера), удобству использования малоквалифицированными пользователями (социальная сфера), эффективности использования ресурсов (сфера материально-технического снабжения).

**Вторым компонентом** БД является накапливаемая и обрабатываемая информация. Характеристики качества можно разделить на функциональные и конструктивные.

**Функциональная пригодность** может отражаться:

● полнотой накопленных описаний объектов (относительное число объектов или документов)

● идентичностью данных (относительным числом описаний объектов, не содержащих ошибок, к общему числу документов об объектах)

● актуальностью данных (относительным числом устаревших данных об объектах к общему числу накопленных и обрабатываемых данных)

**К конструктивным характеристикам** качества можно отнести все стандартизированные показатели качества ПС, которые представлены в ISO 9126. Требования также должны содержать обеспечение ее надежности, эффективности использования ресурсов ЭВМ, практичности, применимости, сопровождаемости и мобильности.

**Модели качества по стандарту ISO 9126:1-4:2002:** Билет 5.2

# Экзаменационный билет 23

## ~~Основные парадигмы программирования~~. Задачи и особенности структурного, ~~объектно-ориентированного~~ и мультиагентного проектирования сложных систем.

Основные парадигмы программирования Задачи и особенности объектно-ориентированного (Билет 16.2)

Задачи и особенности структурного проектирования сложных систем

1. Разбиение на модули: Это помогает управлять сложностью системы, улучшить ее понимание и облегчить разработку и сопровождение.

2. Определение интерфейсов между модулями или компонентами системы.

3. Управление зависимостями: Сложные системы обычно имеют множество взаимосвязанных элементов, чтобы минимизировать связанность.

4. Определение иерархии помогает организовать компоненты в логически связанные группы и устанавливает их отношения по уровням или подсистемам.

5. Управление изменениями: Сложные системы часто подвержены изменениям. Задача – разработка гибкой архитектуры, которая позволяет эффективно вносить изменения и обеспечивает легкость поддержки и модификации.

Особенности структурного проектирования сложных систем:

1. Слабая связность и сильная согласованность

2. Многокритериальное проектирование

3. Масштабируемость: Архитектура должна легко адаптироваться и расширяться для удовлетворения растущих потребностей и изменений.

4. Взаимодействие с другими системами: структурное проектирование должно учитывать взаимодействия меду сложными системами, обеспечивая интерфейсы, протоколы.

Задачи и особенности мультиагентного проектирования

1. Распределение задач: Задачи или функций распределяются между агентами.

2. Координация и коммуникация: Задача состоит в разработке механизмов коммуникации и протоколов, которые позволяют агентам обмениваться информацией и координировать свои действия.

3. Разрешение конфликтов: Задача состоит в разработке алгоритмов и механизмов, которые позволяют разрешать конфликты интересов и ресурсов и достигать справедливости при принятии решений.

4. Управление ресурсами: ресурсы, такие как вычислительные мощности, память могут быть распределены между агентами. Задача - разработать алгоритм управления ресурсами, чтобы обеспечить оптимальное их использование.

Особенности мультиагентного проектирования:

1. Автономность агентов: каждый агент обладает автономностью и способностью принимать решения на основе своих собственных знаний, целей и ограничений.

2. Скорость изменений: системы могут быть подвержены быстрым изменениям в окружающей среде или сами агенты могут изменять свое состояние и стратегии. Особенность состоит в разработке алгоритмов и механизмов, которые позволяют системе адаптироваться к изменениям и обеспечивать устойчивость в динамической среде.

## История и основные направления развития искусственного интеллекта. Национальные стандарты на создание систем ИИ.

**История развития искусственного интеллекта в СССР и России**

В 1832 Корсаков опубликовал описание пяти изобретённых им механических устройств («интеллектуальных машин»), для частичной механизации умственной деятельности в задачах поиска, сравнения и классификации. В конструкции применены перфорированные карты, (роль [баз знаний](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B7%D0%BD%D0%B0%D0%BD%D0%B8%D0%B9)), а сами машины – прототипы [экспертных систем](https://ru.wikipedia.org/wiki/%D0%AD%D0%BA%D1%81%D0%BF%D0%B5%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0).

В СССР работы в области ИИ начались в 1960-х годах. В [Московском университете](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D1%81%D0%BA%D0%BE%D0%B2%D1%81%D0%BA%D0%B8%D0%B9_%D0%B3%D0%BE%D1%81%D1%83%D0%B4%D0%B0%D1%80%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D1%83%D0%BD%D0%B8%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D1%82%D0%B5%D1%82_%D0%B8%D0%BC%D0%B5%D0%BD%D0%B8_%D0%9C._%D0%92._%D0%9B%D0%BE%D0%BC%D0%BE%D0%BD%D0%BE%D1%81%D0%BE%D0%B2%D0%B0) и [Академии наук](https://ru.wikipedia.org/wiki/%D0%90%D0%BA%D0%B0%D0%B4%D0%B5%D0%BC%D0%B8%D1%8F_%D0%BD%D0%B0%D1%83%D0%BA_%D0%A1%D0%A1%D0%A1%D0%A0) был выполнен ряд исследований, рук. - Д. А. Поспелов.

Большой вклад в развитие искусственного интеллекта в СССР внес академик [Г. С. Поспелов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%BF%D0%B5%D0%BB%D0%BE%D0%B2,_%D0%93%D0%B5%D1%80%D0%BC%D0%BE%D0%B3%D0%B5%D0%BD_%D0%A1%D0%B5%D1%80%D0%B3%D0%B5%D0%B5%D0%B2%D0%B8%D1%87). Под его руководством были внедрены диалоговые и интеллектуальные системы планирования развития отраслей промышленности. По инициативе [Г. С. Поспелова](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%BF%D0%B5%D0%BB%D0%BE%D0%B2,_%D0%93%D0%B5%D1%80%D0%BC%D0%BE%D0%B3%D0%B5%D0%BD_%D0%A1%D0%B5%D1%80%D0%B3%D0%B5%D0%B5%D0%B2%D0%B8%D1%87) был создан Научный совет по проблеме «Искусственный интеллект»

**Можно выделить два направления развития ИИ:**

* решение проблем, связанных с приближением систем ИИ к возможностям человека ([Усиление интеллекта](https://ru.wikipedia.org/wiki/%D0%A3%D1%81%D0%B8%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B8%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82%D0%B0));
* создание искусственного разума, представляющего интеграцию уже созданных систем ИИ в единую систему, способную решать проблемы человечества ([Сильный и слабый искусственный интеллект](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B8_%D1%81%D0%BB%D0%B0%D0%B1%D1%8B%D0%B9_%D0%B8%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82)).

Билет 11.2

# Экзаменационный билет 24

## Описать причины возникновения рисков в жизненном цикле программных средств. Описать мероприятия по предотвращению рисков при создании любых сложных систем.

**Причины возникновения и проявления рисков** – злоумышленные действия заинтересованных лиц или случайные негативные проявления дефектов внешней среды, системы или пользователей.

**Злоумышленные, активные воздействия заинтересованных лиц:** риски могут быть обусловлены искажениями программ и информационных ресурсов, которые предназначены для применения ограниченным кругом лиц.

**Решение:** созданы и активно развиваются методы, средства и стандарты обеспечения защиты программ и данных от предумышленных негативных внешних воздействий.

**Риски при случайных**, дестабилизирующих воздействиях. Эти риски зависят от отказовых ситуаций. Основными источниками отказовых ситуаций: некорректные исходные требования, сбои и отказы в аппаратуре, дефекты или ошибки в программах и данных. При таких воздействиях, внешняя, функциональная работоспособность систем может разрушаться не полностью, однако невозможно полноценное выполнение заданных функций и требований к качеству информации для потребителей.

Эти риски могут быть обнаружены с помощью процессов анализа и сокращения рисков. Эти процессы должны сопутствовать основным этапам разработки и обеспечения ЖЦ и отражены пятью этапами:

* анализ рисков следует начинать с подготовки детальных исходных требований и характеристик проекта ПС, системы и внешней среды, для которых должны отсутствовать риски функционирования и применения;
* рекомендуется выделять три класса рисков: функциональной пригодности, конструктивных характеристик качества и нарушения ограничений ресурсов;
* в каждом классе предлагается анализировать несколько категорий наиболее важных рисков, которые упорядочивать по степени опасности
* контрмеры для сокращения рисков рекомендуется анализировать и применять последовательно, начиная с ликвидации наиболее опасных, затем проводить анализ и уменьшение уязвимости компонентов и ПС в целом
* процессы устранения рисков должны завершаться процедурами мониторинга, сопровождения и конфигурационного управления изменениями версий.

## ~~Процессы управления конфигурацией программного обеспечения в соответствии со стандартами ISO 12207 и ISO 15846.~~

Билет 20.2