Лабораторная работа №1

Виды тестирования. Планирование тестирования

Цель: изучить классификацию видов тестирования, разработать проверки для различных видов тестирования, научиться планировать тестовые активности в зависимости от особенностей поставляемой на тестирование функциональности.

Теоретические сведения

Тестирование (Testing) – процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта.

Конечной целью тестирования является предоставление пользователю качественного программного обеспечения (ПО).

Качество (Quality) – степень, с которой компонент, система или процесс соответствует зафиксированным требованиям и/или ожиданиям и нуждам пользователя или заказчика.

Дефект (defect, bug, ошибка) – ключевой термин тестирования, означающий отклонение фактического результата от ожидаемого. Для обнаружения дефекта необходимо выполнить три условия: знать фактический результат, знать ожидаемый результат, зафиксировать факт разницы между фактическим и ожидаемым результатом.

Процесс тестирования как процесс поиска дефектов сводится к следующей последовательности действий:

1. Узнаем ожидаемый результат.
2. Узнаем фактический результат.
3. Сравниваем ожидаемый и фактический результаты.

Источником ожидаемого результата является спецификация – детальное описание того, как должно работать ПО.

В общем случае любой дефект представляет собой отклонение от спецификации. Важно обнаружить эти дефекты до того, как их найдут конечные пользователи.

Тестирование можно классифицировать по очень большому количеству признаков.

Виды тестирования в зависимости от объекта тестирования

Виды тестирования в зависимости от объекта тестирования: функциональные, пограничные, нефункциональные (рисунок 1).

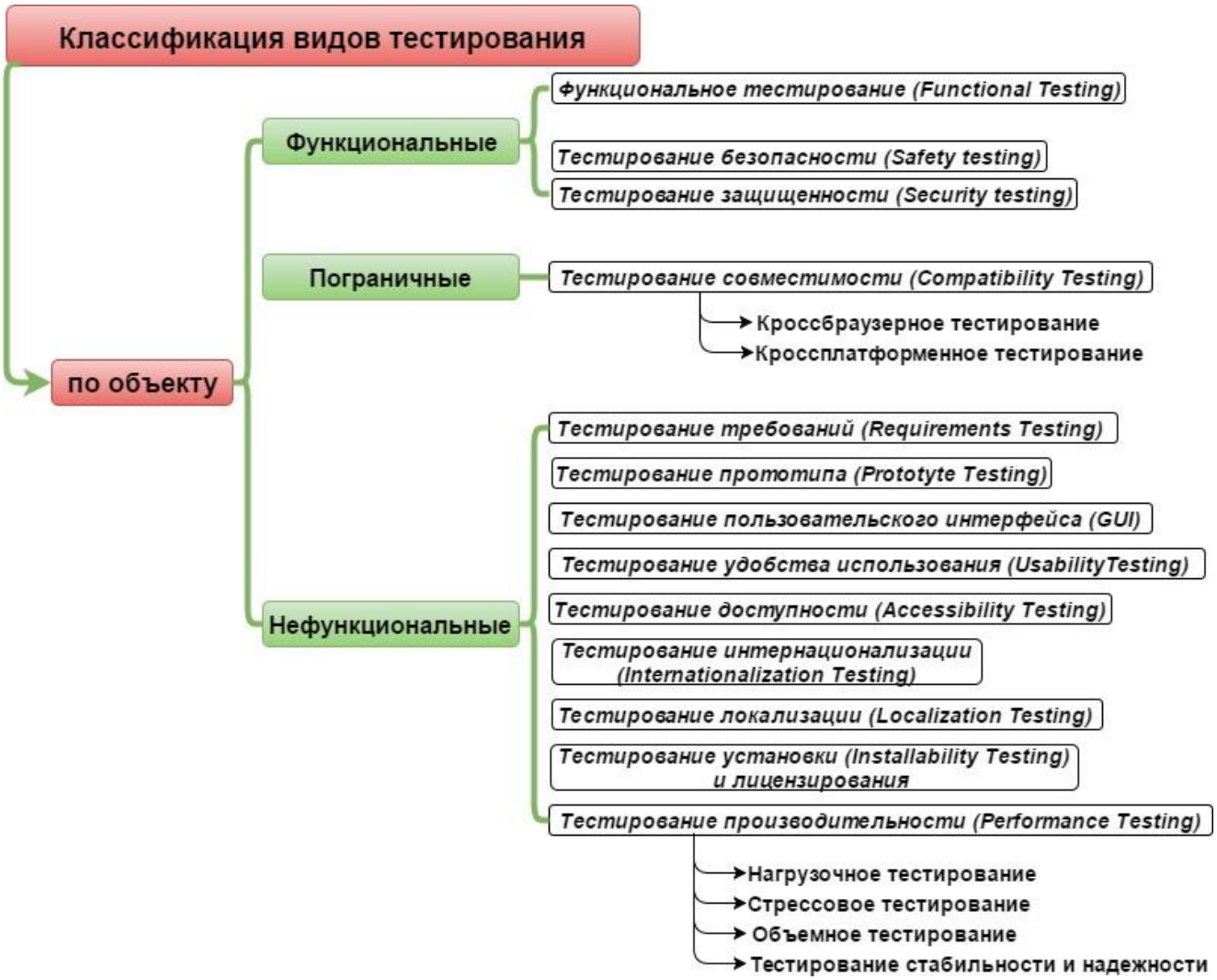

Рисунок 1 – Классификация видов тестирования в зависимости

от объекта Рассмотрим ***функциональные*** виды тестирования.

Функциональное тестирование (Functional Testing) – тестирование, основанное на сравнительном анализе спецификации и функциональности компонента или системы. **Обязательное тестирование.**

Тестирование безопасности (Safety Testing) – тестирование программного продукта с целью определить его способность при использовании оговоренным образом оставаться в рамках приемлемого риска причинения вреда здоровью, бизнесу, программам, собственности или окружающей среде. **В этом месте хочется сказать забейте (т.к. слышал только один случай жалобы на ПО, это на компанию Фольцваген), но напишу: Обязательное тестирование**.

Тестирование защищенности (Security Testing) – тестирование с целью оценить защищенность программного продукта от внешних воздействий (от проникновений). **Обязательное тестирование, как**

минимум всегда проверяйте поля ввода на возможность написать и выполнить в них скрипт.

Рассмотрим ***пограничные*** виды тестирования.

Тестирование совместимости (Compatibility Testing) **–** проверка работоспособности приложения в различных средах (браузеры и их версии, операционные системы, их типа, версии и разрядность). Виды тестирования совместимости: кроссбраузерное тестирование (различные браузеры или версии браузеров), кроссплатформенное тестирование (различные операционные системы или версии операционных систем). **Обязательное тестирование. Как вы видели на лекции в разных браузерах ваши страницы могут отражаться по разному.**

Рассмотрим нефункциональные виды тестирования, направленные на проверку характеристик или свойств программы (внешний вид, удобство использования, скорость работы и т.п.).

Тестирование требований (Requirements Testing) – проверка требований на соответствие основным атрибутам качества. **Обязательное тестирование, иначе вы просто не сдадите программное средство заказчику.**

Тестирование прототипа (Prototyte Testing) – метод выявления структурных, логических ошибок и ошибок проектирования на ранней стадии развития продукта до начала фактической разработки. **Обязательное тестирование, которое позволит вам уменьшить количество костылей в процессе работы и в конечном программном продукте.**

Тестирование пользовательского интерфейса (GUI Testing) – тестирование, выполняемое путем взаимодействия с системой через графический интерфейс пользователя (правописание выводимой информации; расположение и выравнивание элементов GUI; соответствие названий форм / элементов GUI их назначению; унификация стиля, цвета, шрифта; окна сообщений; изменение размеров окна, поведение курсора и горячие клавиши) **Обязательное тестирование.**

Тестирование удобства использования (Usability Testing) – тестирование с целью определения степени понятности, легкости в изучении и использовании, привлекательности программного продукта для пользователя при условии использования в заданных условиях эксплуатации (на этом уровне обращают внимание на визуальное оформление, навигацию, логичность, наличие обратной связи и др.). **Пользовательский интерфейс один из важных компонентов удачной продажи программного продукта, ну если честно говорить, то не только программного, для примера вспомните салон автомобиля и количество средств, которые тратят компании на эргономику. В отличие от предыдущего теста, данный вид тестирования проверяет**

эргономику. Хотя, предвкушая Ваш вопрос: «А как я пойму кому удобно, а кому нет?» отвечу, в больших компаниях этим вопросом занимаются целые отделы, а в нашем случае попросите попользоваться ваших родственников разного поколения, когда вы увидите их мучения, вы поймете, что нужно исправить.

Тестирование доступности (Accessibility Testing) – тестирование, которое определяет степень легкости, с которой пользователи с ограниченными способностями могут использовать систему или ее компоненты. **Современный тренд говорит об обязательности данного вида тестировнаия.**

Тестирование интернационализации (Internationalization Testing) – тестирование адаптации продукта к языковым и культурным особенностям целого ряда регионов, в которых потенциально может использоваться продукт. ***Большинство отечественных компаний практически не применяет, так как разработчик не ориентирован на внешние рынки. Но знать про данный вид теста полезно. На случай «А вдруг…»***

Тестирование локализации (Localization Testing) – тестирование адаптации продукта к языковым и культурным особенностям конкретного региона, отличного от того, в котором разрабатывался продукт. ***Большинство отечественных компаний практически не применяет, так как разработчик не ориентирован на внешние рынки. Но знать про данный вид теста полезно. На случай «А вдруг…»***

Тестирование производительности (Performance Testing) – процесс тестирования с целью определения производительности программного продукта. В рамках тестирования производительности выделяют нагрузочное теситрование, объемное тестирование, тестирование стабильности и надежности, стрессовое тестирование. ***Обязательный тест для всех видов программных средств.***

Нагрузочное тестирование (Performance and Load Testing) – вид тестирования производительности, проводимый с целью оценки поведения компонента или системы при возрастающей нагрузке, например количестве параллельных пользователей и/или операций, а также определения какую нагрузку может выдержать компонент или система. ***Обязательный тест, в случае если компания собирается выпускать многопользовательское программное средство, web-приложение или сервис, а также мобильное приложение, которое подразумевает одновременную работу большого количества пользователей.***

Объемное тестирование (Volume Testing) – позволяет получить оценку производительности при увеличении объемов данных в базе данных приложения. ***Обязательный тест для средних и крупных***

программных средств.

Тестирование стабильности и надежности (Stability / Reliability Testing) – позволяет проверять работоспособность приложения при длительном (многочасовом) тестировании со средним уровнем нагрузки. ***Обязательный тест для всех видов программных средств.***

Стрессовое тестирование (Stress Testing) – вид тестирования производительности, оценивающий систему или компонент на граничных значениях рабочих нагрузок или за их пределами, или же в состоянии ограниченных ресурсов, таких как память или доступ к серверу. ***Обязательный тест для всех видов программных средств. Особенно для мобильных приложений с учетом специфики ОС (подразумевается жизненный цикл приложения)***

Тестирование на отказ и восстановление (Failover and Recovery Testing) – тестирование при помощи эмуляции отказов системы или реально вызываемых отказов в управляемом окружении. ***Обязательный тест для всех видов программных средств.***

Тестирование установки (Installability Testing) и лицензирования – процесс тестирования установки программного продукта. Включает формальный тест программы установки приложения (проверка пользовательского интерфейса, навигации, удобства использования, соответствия общепринятым стандартам оформления); функциональный тест программы установки; тестирование механизма лицензирования и функций защиты от пиратства; проверку стабильности приложения после установки. ***Обязательный тест для всех видов программных средств.***

Виды тестирование по доступу к коду программного средства

Как вы, наверное, догадались предложенная классификация видов тестирования не может быть единственной. Вопрос который вы можете себе задать: «А если у меня нет доступа к коду или доступ ограничен, или я имею полный доступ к коду программного продукта, могу ли я выполнить классификацию по данным критериям?». Конечно можете, правильнее наверное сказать - должны. Из вашего вопроса понимаем, что таких видов тестирование может быть три: белый ящик, серый ящик, черный ящик.

Белый ящик (White Box Testing) – тестирование, основанное на анализе внутренней структуры компонентов или системы (у тестировщика есть доступ к внутренней структуре и коду приложения).

Серый ящик (Grey Box Testing) – комбинация методов белого и черного ящика, состоящая в том, что к части кода архитектуры у тестировщика есть, а к части кода – нет.

Черный ящик (Black Box Testing) – тестирование системы без знания внутренней структуры и компонентов системы (у тестировщика нет доступа к внутренней структуре и коду приложения либо в процессе тестирования он не обращается к ним).

Виды тестирования в зависимости от степени автоматизации: ручное, автоматизированное тестирование.

На лекции вы спрашивали про данный вид тестирования, поэтому поясняю, что тестирование бывает двух видов:

Ручное тестирование – такое тестирование, в котором тест-кейсы выполняются тестировщиком вручную без использования средств автоматизации.

Автоматизированное тестирование (Automated Testing) – набор техник, подходов и инструментальных средств, позволяющий исключить человека из выполнения некоторых задач в процессе тестирования. Тест-кейсы частично или полностью выполняет специальное инструментальное средство.

Виды тестирования в зависимости от степени изолированности тестируемых компонентов: модульное, интергационное, системное тестирование.

Модульное тестирование (Unit/Component Testing) – тестируются отдельные части (модули) системы. ***В основном разработчики пытаются пренебрегать данным тестом***. Угадайте к какому тестированию они переходят? И какие последствия? (кто это прочитал, ответ мне на почту, это пасхалка)

Интеграционное тестирование (Integration Testing) – тестируется взаимодействие между отдельными модулями. ***С этим сталкиваются разработчики при работе в команде, взаимодействие между модулями тестируется практически всегда на этапе разработки, но желательно разработать специальный тест.***

Системное тестирование (System Testing) – тестируется работоспособность системы в целом. ***Вот этот тест выполняют все разработчики, но проблема в том, что у разработчика свое видение работы системы, поэтому не всегда тест от разработчика говорит о корректности работы системы. Вот и приходится людям (разработчики тоже люди, но давайте тут их представим творцами программного средства, а остальные просто люди) разрабатывать тесты для проверки работоспособности со своей колокольни.***

Виды тестирования в зависимости от подготовленности

Тестирование по подготовленности бывает: интуитивное тестирование, исследовательское тестирование, тестирование по документации.

Интуитивное тестирование выполняется без подготовки к тестам, без определения ожидаемых результатов, проектирования тестовых сценариев. Это как вы идете на пару неподготовленные, в надежде что-то нахватать во время занятия, а может и на обум ответить во время защиты практической работы.

Исследовательское тестирование – метод проектирования тестовых сценариев во время выполнения этих сценариев. Можете запомнить это как разведка боем.

Тестирование по документации – тестирование по подготовленным тестовым сценариям, руководству по осуществлению тестов. Такой подход говорит о детальной проработке процесса разработки программного продукта, к нему нужно стремиться, но…. 85% из вас им пользоваться не будут.

Виды тестирования в зависимости от места и времени проведения тестирования

Виды тестирования в зависимости от места и времени проведения тестирования: приемочное тестирование, альфа-тестирование, бета-тестирование.

Все мы слышали про игры, операционные системы и прочее, что они находятся в альфа тестировании или в бета тестировании, давайте разберемся. Что это и когда нам чего ждать.

Приемочное тестирование (User Acceptance Testing, UAT) – формальное тестирование по отношению к потребностям, требованиям и бизнес-процессам пользователя, проводимое с целью определения соответствия системы критериям приёмки и дать возможность пользователям, заказчикам или иным авторизованным лицам определить, принимать систему.

Альфа-тестирование (Alpha Testing) – моделируемое или действительное функциональное тестирование, выполняется в организации, разрабатывающей продукт, но не проектной командой (это может быть независимая команда тестировщиков, потенциальные пользователи, заказчики). Альфа тестирование часто применяется к коробочному программному обеспечению в качестве внутреннего приемочного тестирования.

Бета-тестирование (Beta Testing) – эксплуатационное тестирование потенциальными или существующими клиентами/заказчиками на внешней стороне (в среде, где продукт будет использоваться) никак связанными с разработчиками, с целью определения действительно ли компонент или система удовлетворяет требованиям клиента/заказчика и вписывается в бизнес-процессы. Бета-тестирование часто проводится как форма внешнего приемочного тестирования готового программного обеспечения для того, чтобы получить отзывы рынка.

Одним словом, услышали про бета тестирование игры, можете смело ожидать ее на рынке в ближайшее время, скорее всего разработчики после бета-тестирования поправят баланс и уберут ошибки в картах.

Виды тестирования в зависимости от глубины тестового покрытия

В тестировании в зависимости от глубины приняты следующие виды: Smoke, MAT, AT.

Не пугаемся абревиатур, попытаемся их понять.

В начале поймем для себя что такое тестовое покрытие.

Тестовое покрытие – одна из метрик оценки качества тестирования, представляющая из себя плотность покрытия тестами требований либо исполняемого кода.

Smoke Test – поверхностное тестирование для определения пригодности сборки для дальнейшего тестирования, должно покрывать базовые функции программного обеспечения; уровень качества: Acceptable / Unacceptable (жду перевод на почту, пасхалка номер 2).

Minimal Acceptance Test (MAT, Positive Test) – тестирование системы или ее части только на корректных данных/сценариях; уровень качества: High / Medium / Low. Это когда вы не вводите в систему того, что не надо в нее вводить и не ставите себя на место неумелого пользователя.

Acceptance Test (AT) – полное тестирование системы или ее части как на корректных (Positive Test), так и на некорректных данных/сценариях (Negative Test); уровень качества: High / Medium / Low. Тест на этом уровне покрывает все возможные сценарии тестирования: проверку работоспособности модулей при вводе корректных значений; проверку при вводе некорректных значений; использование форматов данных отличных от тех, которые указаны в требованиях; проверку исключительных ситуаций, сообщений об ошибках; тестирование на различных комбинациях входных параметров; проверку всех классов эквивалентности; тестирование граничных значений интервалов; сценарии не предусмотренные спецификацией и т.д. (к стати, а что по вашему сценарий, напишите мне на почту ваше мнение это пасхалка 3)

Виды тестирования в зависимости от тестовых активностей

Виды тестирования в зависимости от тестовых активностей: NFT, RT, DV.

Данная классификация тестирования иначе назвыется видами тестирования в зависимости от «ширины» тестового покрытия.

Тестирование новых функциональностей (New Feature Test, NFT) – определение качества поставленной на тестирование новой функциональности, которая ранее не тестировалась. Данный тип тестирования включает в себя: проведение полного теста (AT) непосредственно новой функциональности; тестирование новой функциональности на соответствие документации; проверку всевозможных взаимодействий ранее реализованной функциональности с новыми модулями и функциями.

Регрессионное тестирование (Regression Testing, RT) проводится с целью оценки качества ранее реализованной функциональности. Включает в себя проверку стабильности ранее реализованной функциональности после внесения изменений, например добавления новой функциональности, исправление дефектов, оптимизация кода, разворачивание приложения на новом окружении. Регрессионное тестирование как правило выполняется на уровне MAT.

Валидация дефектов (Defect Validation, DV) – проверка результатов исправления дефектов; может включать элементы регрессионного тестирования; уровень проверки не определяется.

Процесс тестирования программного продукта

Процесс тестирования программного продукта включает следующие этапы:

1. Изучение и анализ предмета тестирования.
2. Планирование тестирования.
3. Исполнение тестирования.

Изучение и анализ предмета тестирования начинается еще до утверждения спецификации и продолжается на стадии разработки (кодирования) программного обеспечения. Конечной целью этапа изучения и анализа предмета тестирования является получение ответов на два вопроса: какие функциональности предстоит протестировать, как эти функциональности работают.

Планирование тестирования происходит на стадии разработки (кодирования) программного обеспечения. На стадии планирования тестирования перед тестировщиком стоит задача поиска компромисса между объемом тестирования, который возможен в теории, и объемом тестирования, который возможен на практике. На данной стадии необходимо ответить на вопрос: как будем тестировать? Результатом планирования тестирования является тестовая документация.

Выполнение тестирования происходит на стадии тестирования и представляет собой практический поиск дефектов с использованием тестовой документации, составленной ранее.

Для всех программных продуктов выполняют следующие типы тестов и их композиции.

Для первой поставки программного обеспечения рекомендуется проводить Smoke + NFT_{AT} готовой функциональности: поверхностное тестирование (Smoke Test) выполняется для определения пригодности сборки для дальнейшего тестирования; полное тестирование системы или ее части как на корректных, так и на некорректных данных/сценариях (Acceptance Test, AT) позволяет обнаружить дефекты и внести запись о них в багтрэкинговую систему. Обратите внимание на слово композиция, оно вам пригодится при решении практической работы.

Для последующих поставок программного обеспечения композиции тестов могут быть следующими.

Если не была **добавлена новая функциональность**, то: DV + RT_{MAT}. Т.е., выполняется проверка исправления дефектов программистом (Defect Validation, DV), а также проверка работоспособности остальной функциональности после исправления дефектов на позитивных сценариях (Minimal Acceptance Test, MAT).

Если **была добавлена новая функциональность**, то: Smoke + DV + NFT_{AT} + RT_{MAT}. В частности, выполняется поверхностное тестирование (Smoke Test), проверка исправления дефектов программистом (Defect Validation, DV), тестирование новых функциональностей (New Feature Testing, NFT), проверка старых функциональностей, т.е. регрессионное тестирование (Regression Test).

Если **была добавлена новая функциональность, то возможен также вариант:** DV + NFT_{AT} + RT_{MAT}, т.е. без выполнения Smoke Test.

Таким образом, **для второй и последующих поставок** обобщенная схема композиции тестов выглядит следующим образом:

(Smoke) + DV + (NFT_{AT}) + RT_{MAT}.

В **зависимости от типа и специфики приложения** (web, desktop, mobile) выполняют **специализированные тесты** (например, кроссбраузерное или кроссплатформенное тестирование, тестирование локализации и интернационализации и др.).

Практическое задание:

1. Выбрать объект реального мира (например, карандаш, стол, чашка, клавиатура, сумка и др.) с целью последующей разработки тестовых проверок для него.

2. Разработать различные проверки в соответствии с классификацией видов тестирования для выбранного объекта реального мира. Результаты внести в таблицу 1.1.

Таблица 1.1 – Тестировые проверки для различных видов тестирования

Объект тестирования: *указать*		
Вид тестирования	**Краткое определение вида тестирования**	**Тестовые проверки**
Functional Testing		
Safety Testing		
Security Testing		
Compatibility Testing		
GUI Testing		
Usability Testing		
Accessibility Testing		
Internationalization Testing		
Performance Testing		
Stress Testing		
Negative Testing		
Black Box Testing		
Automated Testing		

Unit/Component Testing		
Integration Testing		

3. Разработать композицию тестов для первой поставки программного обеспечения (build 1), состоящей из трех модулей (модуль 1, модуль 2, модуль 3).

4. Разработать композицию тестов для второй поставки программного обеспечения (build 2): исправлены заведенные дефекты, доставлена новая функциональность – модуль 4.

5. Разработать композицию тестов для третьей поставки программного обеспечения (build 3): заказчик решил расширять рынки сбыта и просит осуществить поддержку программного обеспечения на английском языке.

6. Разработать композицию тестов для четвертой поставки программного обеспечения (build 4): заказчик хочет убедиться, что программное обеспечение выдержит нагрузку в 2000 пользователей.

7. Оформить отчет и защитить лабораторную работу.

Содержание отчета:

1. Цель работы.
2. Разработанные проверки выбранного объекта реального мира для различных видов тестирования.
3. Тестовые активности для сформулированных задач.
4. Выводы по работе.

Контрольные вопросы:

1. Что такое тестирование?
2. Что такое качество программного обеспечения?
3. Что такое дефект?
4. Назовите три условия обнаружения дефекта.
5. Какие существуют виды тестирования в зависимости от объекта тестирования? Дайте характеристику каждому.
6. Какие существуют виды функционального тестирования? Дайте характеристику каждому.
7. Какие существуют виды нефункционального тестирования? Дайте характеристику каждому.
8. Какие существуют виды тестирования в зависимости от глубины покрытия? Дайте характеристику каждому.
9. Какие существуют тестовые активности? Дайте характеристику каждому.
10. Какие существуют виды тестирования в зависимости от знания кода? Дайте характеристику каждому.
11. Какие существуют виды тестирования в зависимости от

степени автоматизации? Дайте характеристику каждому.

12. Какие существуют виды тестирования в зависимости от изолированности компонентов? Дайте характеристику каждому.

13. Какие существуют виды тестирования в зависимости от подготовленности? Дайте характеристику каждому.

14. Какие существуют виды тестирования в зависимости от места и времени проведения? Дайте характеристику каждому.

15. Какие этапы составляют процесс тестирования?

16. Какая композиция тестов выполняется для первой поставки программного продукта?

17. Какие композиция тестов выполняется для последующих поставок программного продукта?

Пример тестового плана для объекта карандаш