# ЛАБОРАТОРНАЯ РАБОТА №2

# Тема лабораторной работы: работа с классификацией видов тестирования.

**Описание программного проекта.**

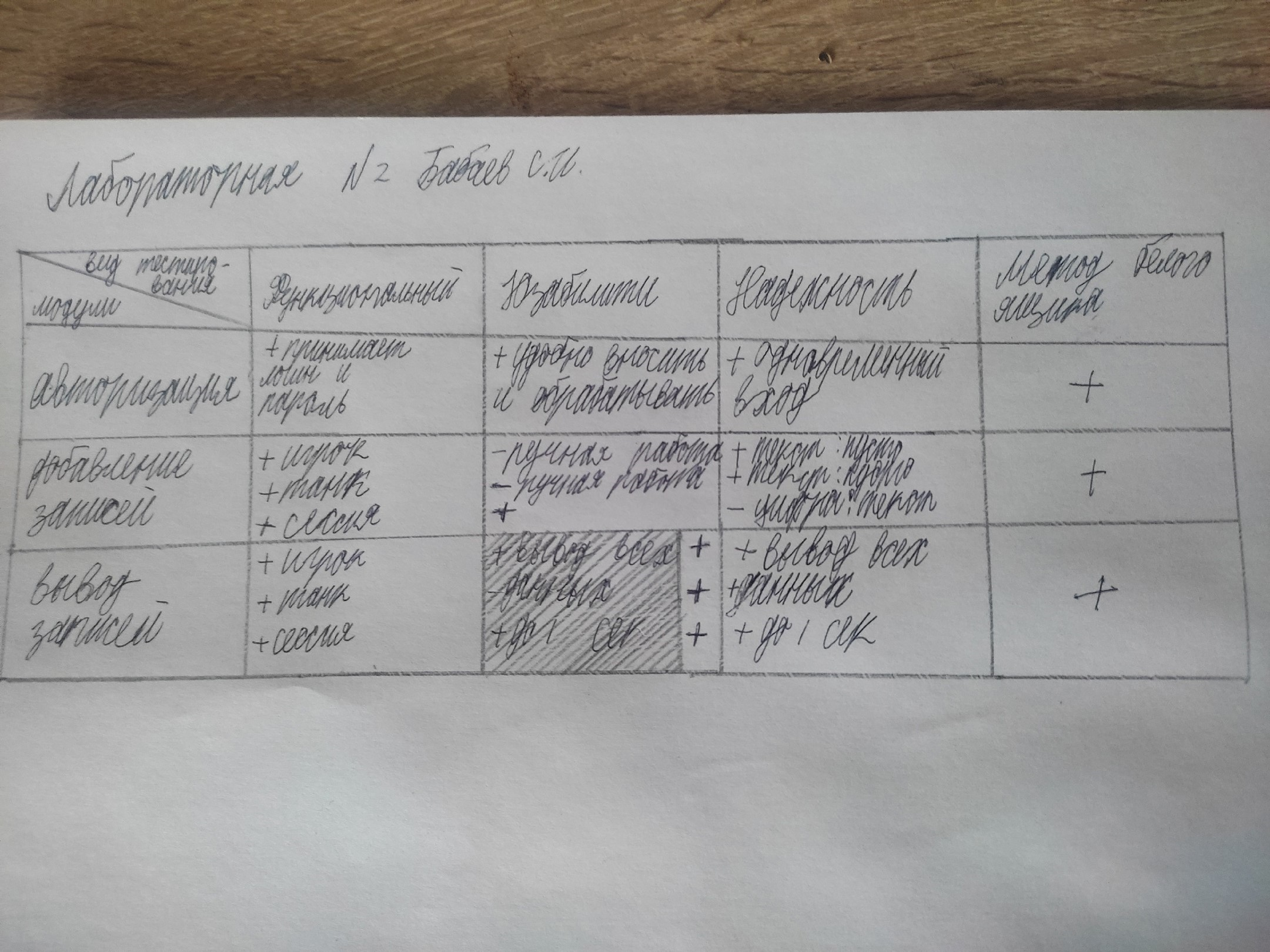
В данной работе мы будет тестировать программный проект игры «Танки». Он создан для хранения данных о пользователях игры и игровых сессиях и составления отчётов. Проект представляет из себя приложение для работы с базой данных. В основе приложения лежит сама БД, документы MS Word и MS Excel для создания отчетов. Проект был создан в среде разработки SQL, а также для работы с базой данных.

Основная база данных носит название «Танки\_БД», она содержит информацию о игроках, игровых предметах, и игровых сессиях и сведения об игровых серверах.

Функционал программы состоит из:

* Добавление и удаление данных в таблицах.
* Редактирование данных в таблицах.
* Фильтрация данных.
* Поиск нужной информации по всей БД.
* Выполнение пользовательских запросов.
* Создание отчётов.

**Компонентное(модульное) тестирование.**



**Выявленные дефекты.**

В ходе модульного тестирования был выявлены следующие дефекты:

* При создании отчета в MS Word возникает проблема несоответствия строк таблицы БД и строк таблицы MS Word, а именно их нехватка. Предложенное решение: увеличить число строк в таблице MS Word. Недостатки: нарушение внешнего вида документа и его удобство в прочтении.
* Отсутствует предупреждение пользователя о некорректном вводе данных, ввиду чего может возникнуть потеря данных. Предложенное решение: добавить в исходный код алгоритм, исключающий ложные символы для каждого типа вносимых данных и добавляющий возможность для пользователя самостоятельно решить – вносить изменения или же исправить данные, т.к. бывают «внештатные» ситуации, при которых просто необходимо ввести данные не по шаблону.

**Выводы по работе.**

Мы произвели описание программного продукта, разобрали его на компоненты осуществили модульное тестирование, в ходе которого был выявлен ряд дефектов и предложения по их дальнейшему решению.

**Список использованных источников.**

Учебник по тестированию .pdf

«Тестирование программного обеспечения. Базовый курс.» Святослав Куликов 289стр.

**Контрольные вопросы**

1. Для оценки качества конкретного ГОСТа не существует, поэтому каждый разработчик, работодатель воспринимает качество программного кода по-своему, исходя из личного опыта.

2. Виды тестирования, подразделяются на группы:

- по объекту (на что обращен процесс тестирования).

Функциональное. Проверяется соответствие требованиям. Оценка производится в соответствии с ожидаемыми и полученными результатами (на основании функциональной спецификации), при условии, что функции отрабатывали на различных значениях.

Навигация. Действие кнопки «назад» браузера для сайта, ошибки: 404, 500 и т. д.

Инсталляция. Формальное тестирование программы установки (графический интерфейс пользователя, общее удобство пользования, соответствие стандартам), функциональное тестирование программы установки, тестирование механизма лицензирования и способности противостоять взлому, проверка стабильности работы приложения после установки.

Нагрузочное. Тесты производительности (количество обрабатываемых запросов в единицу времени), нагрузочные тесты (количество бизнес-пользователей приложения, генерирующих запросы в единицу времени), стресс-тесты (функционирование в условиях экстремальных уровней нагрузки и ограничения ресурсов, а также контроль за скоростью восстановления после стресса), объемное тестирование (увеличение объема хранимых, обрабатываемых данных).

Ресурсные тесты. Тестирование устойчивости в течение длительного времени.

Юзабилити. Тестирование удобства пользования приложением определяет, соответствует ли приложение потребностям целевой аудитории и отвечает ли оно требованиям пользователя (сколько шагов и времени нужно пользователям для выполнения их задач в приложении, частота возникновения проблем взаимодействия с пользовательским интерфейсом).

UI/GUI. Тестирование на соответствие стандартам графических интерфейсов, тестирование с различными разрешениями экрана, тестирование в ограниченных условиях, например, в условиях нехватки памяти, тестирование локализованных версий: точность перевода, проверка длины названий элементов интерфейса, тестирование графического интерфейса пользователя на целевых устройствах.

Локализация. Проверка правильности перевода согласно тематике данного сайта или программы. Проверка перевода раздела «Помощь» и сопроводительной документации.

Безопасность. Тестирование безопасности представляет собой ряд услуг, от разработки политики безопасности до тестирования безопасности на уровне приложения, операционной системы и сетевой безопасности. Учитываются три главных критерия: конфиденциальность, целостность, доступность.

Совместимость. Проверяется функционирование на различных операционных системах, платформах разработки, в различных браузерах.

Конфигурация. Тестируется функционирование программных систем на различных аппаратных платформах и при различных конфигурациях аппаратно-зависимого программного обеспечения (драйвера, взаимодействие с периферийными устройствами).

Документация. Тестируется на этапе разработки требований к программному продукту после создания функциональных спецификаций. Помогает избежать логических дефектов и ненужных изменений в продукте до начала его фактической разработки. Позволяет улучшить пользовательскую документацию.

Прототип. Основная цель тестирования прототипа – выявить потенциальные проблемы в приложении, проверить, насколько приложение соответствует потребностям и ожиданиям пользователя, и обнаружить расхождения с требованиями к графическому интерфейсу пользователя. Проверяется:

структура приложения, формы пользовательского интерфейса, прототип бизнес-логики, логические связи между модулями, навигация, графический интерфейс пользователя;

- субъекту, степени автоматизации, признаку позитивности, времени проведения. Данная группа видов тестирования характеризует как выбор объекта тестирования и роль участника команды разработки, так и особенности условий реализации программной системы и жизненного цикла разработки:

Субъект тестирования – лицо или группа специалистов, проводящая тестирование (штатные работники, потенциальные заказчики на стороне разработчика, потенциальные пользователи готовой версии продукта с ограничениями).

По степени автоматизации: ручное тестирование, автоматизированное тестирование, полуавтоматизированное тестирование.

По признаку позитивности. Позитивное тестирование – это тестирование на данных или сценариях, которые соответствуют нормальному (штатному, ожидаемому) поведению системы. Основной целью позитивного тестирования является проверка того, что при помощи системы можно делать то, для чего она создавалась. Негативное тестирование – это тестирование на данных или сценариях, которые соответствуют нештатному поведению тестируемой системы – различные сообщения об ошибках, исключительные ситуации, запредельные состояния. Основной целью негативного тестирования является проверка устойчивости системы к воздействиям различного рода, валидация неверного набора данных, проверка обработки исключительных ситуаций (как в реализации самих программных алгоритмов, так и в логике бизнес-правил).

По времени проведения. Альфа-тестирование проводится на ранней стадии разработки продукта, но в некоторых случаях может применяться для законченного продукта в качестве внутреннего приемочного тестирования. Иногда альфа- тестирование выполняется под отладчиком или с использованием окружения, которое помогает быстро выявлять найденные ошибки. Тестирование новой функциональности проверяет корректность реализации новых задач. Тестирование при приемке (Smoke-testing, дымное тестирование) – поверхностная проверка всех модулей приложения на предмет работоспособности и наличия быстро находимых критических и блокирующих дефектов. Тестирование сборки направлено на определение соответствия выпущенной версии критериям качества для начала тестирования. Приемочное тестирование – формальный процесс тестирования, который проверяет соответствие системы требованиям и проводится с целью определения, удовлетворяет ли система приемочным критериям; вынесения решения заказчиком или другим уполномоченным лицом о приемке приложения. Регрессионное тестирование – это вид тестирования, направленный на проверку изменений, сделанных в приложении или окружающей среде (починка дефекта, слияние кода, миграция на другую операционную систему, базу данных, веб-сервер или сервер приложения), для подтверждения того факта, что существующая ранее функциональность работает, как и

прежде. Регрессионными могут быть как функциональные, так и нефункциональные тесты.

3. Типы тестирования сосредоточены на делении видов тестирования, проверяющих наличие требуемой функциональности (функциональный тип тестирования, что система делает) и проверяющих режимы работы системы при выполнении пользователями своих задач (нефункциональный тип тестирования, как система это делает).

Функциональный тип тестирования включает следующие подвиды:

- функциональное тестирование;

- тестирование возможности взаимодействия (включает тестирование совместимости и интеграционное тестирование);

- тестирование безопасности.

Нефункциональный тип тестирования описывает такие подвиды, как:

- нагрузочное тестирование;

- стресс-тестирование;

- тестирование удобства пользования;

- тестирование надежности и т. д.

Примеры проведения нефункциональных тестов:

- общее исследование поведения системы при экстремальных нагрузках;

- экспертиза обработки ошибок и исключений;

- тестирование пропускной способности системы;

- изучение некоторых частей системы или ее компонентов в условиях непропорциональной нагрузки.

4. Методы подразделяются на статические (тестирующие такие объекты, как программный код, документацию, ресурсы проекта, все то, что может быть проверено без непосредственного выполнения программного кода, т. е. запуска программной системы) и динамические (тестирующие функциональность программной системы и режимы ее эксплуатации).

Подвидом статического метода тестирования является рецензирование

– вид тестирования, который может проводиться перед динамическим тестированием.

То, как он может быть проведен:

- вручную,

- с применением анализаторов кода.

Рецензирование может проводится для любого продукта, связанного с разработкой программного обеспечения, включая спецификации требований и дизайна, код, планы тестирования, руководства пользователя и т.п. Во время рецензирования могут быть найдены упущения, например в требованиях, которые маловероятно найти во время динамического тестирования. Более широким видом статического тестирования является статический анализ, проводимый при помощи специальных инструментов. При этом анализируется:

- код программы (например, потоки управления и поток данных),

- сгенерированный код, например HTML, XML.

Динамические методы тестирования позволяют проверять программную систему при ее непосредственном выполнении. Динамические методы делятся на:

- Методы белого ящика. Тестирование на соответствие программного продукта требованиям со знанием внутренней структуры реализации системы (есть в наличии исходный код и технические спецификации). Это вид тестирования позволяет проводить локализацию ошибок, анализ надежности и устойчивости и другие типы проверок, существенно повышая качество системы.

- Методы черного ящика. Тестирование на соответствие программного продукта требованиям без знания внутренней структуры реализации системы.

- Методы серого ящика. Комбинирование методов белого и черного ящика. При тестировании серого ящика разработчик теста имеет доступ к исходному коду, но при непосредственном выполнении тестов доступ к коду, как правило, не требуется.

5. Уровни тестирования предполагают оценивать программную систему с точки зрения объема программных модулей и учета соответствующих аспектов (например, внимание будет сосредоточено только на проверке кода, реализующего бизнес-логику, или важным будет являться момент интеграции подсистем или модулей). В зависимости от степени детальности компонентов подразделяются на:

- Компонентное (модульное) – тестирование программы на уровне отдельно взятых модулей, функций или классов. Цель компонентного тестирования – выявление локализованных в модуле ошибок в реализации алгоритмов, а также определение степени готовности системы к переходу на следующий уровень разработки и тестирования.

- Интеграционное тестирование предназначено для проверки связи между компонентами, а также взаимодействия с различными частями системы (операционной системой, оборудованием либо связи между различными системами). Уровнями интеграционного тестирования являются: компонентный интеграционный (проверяется взаимодействие между компонентами системы после проведения компонентного тестирования), системный интеграционный уровень (проверяется взаимодействие между разными системами после проведения системного тестирования).

- Системное тестирование проверяет как функциональные, так и нефункциональные требования к системе в целом. При этом выявляются дефекты, такие как неверное использование ресурсов системы, не предусмотренные комбинации данных пользовательского уровня, несовместимость с окружением, непредусмотренные сценарии использования, отсутствующая или неверная функциональность, неудобство использования и т. д.

При этом деление на уровни тестирования предполагает участие различных ролей участников команды разработки. Предполагается, что тесты уровня компонентного тестирования разрабатывает сам программист, контролируя самостоятельно качество своей работы или работы своих коллег. Интеграция модулей или общесистемное тестирование могут быть проведены уже другими участниками команды, знающими требования к программным интерфейсам модулей и подсистемам и имеющими возможность для манипуляции с ними.