ФЕДЕРАЛЬНОЕ государственное БЮДЖЕТНОЕ

образовательное учреждение

высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра защиты информации

*Изображение выглядит как текст, Шрифт, снимок экрана, логотип

Автоматически созданное описание*

**ЛАБОРАТОРНАЯ РАБОТА №3**

**«**Объектно-ориентированное программирование**»**

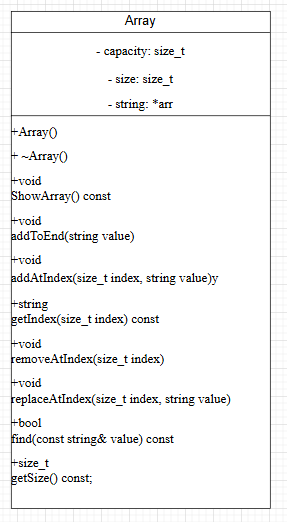
**по дисциплине: «*Программирование*»**

|  |  |
| --- | --- |
| Выполнил:  Студент гр. «АБс-324», «АВТФ»  *Бурлаков Илья Евгеньевич*  «\_\_» \_\_\_\_\_\_\_\_ 2024г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | Проверил:  *Ассистент кафедры ЗИ*  *Исаев Глеб Андреевич*  «\_\_\_» \_\_\_\_\_\_ 2024г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

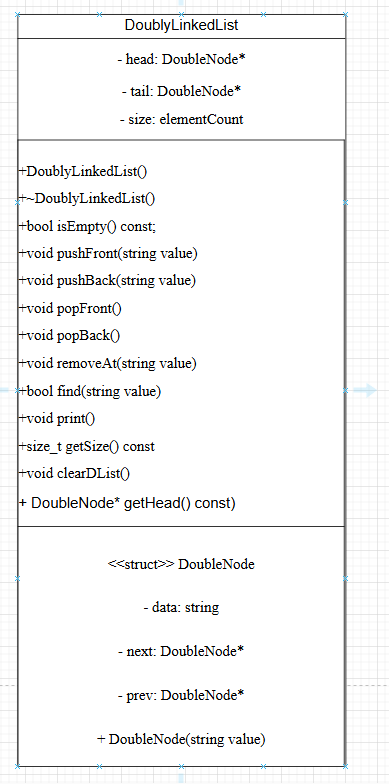
Новосибирск 2024

# **UML – Диаграммы**

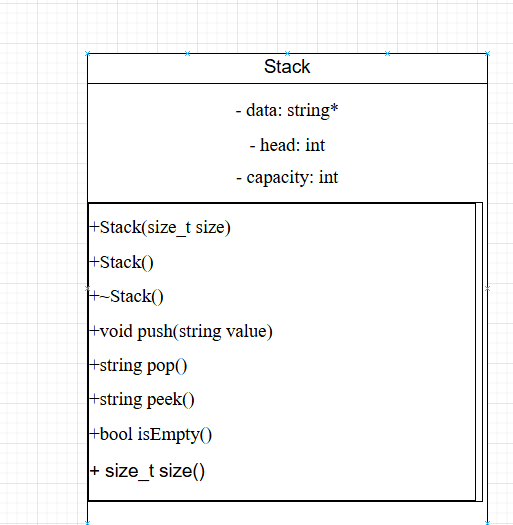
## Массив:



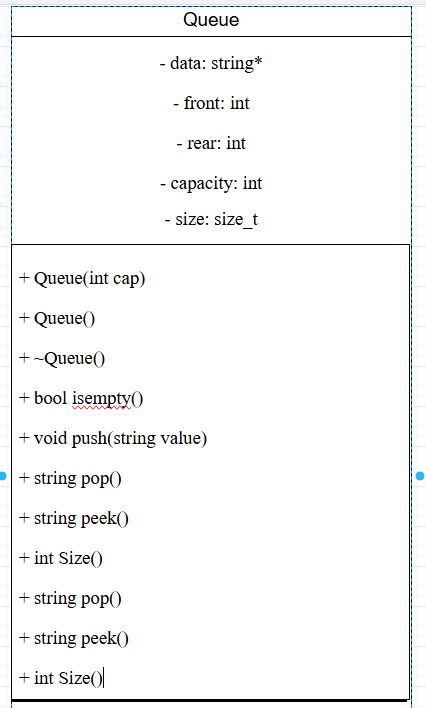
## Списки:



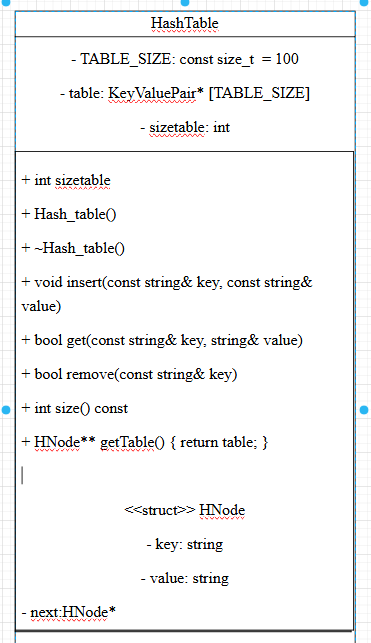
## Стек:



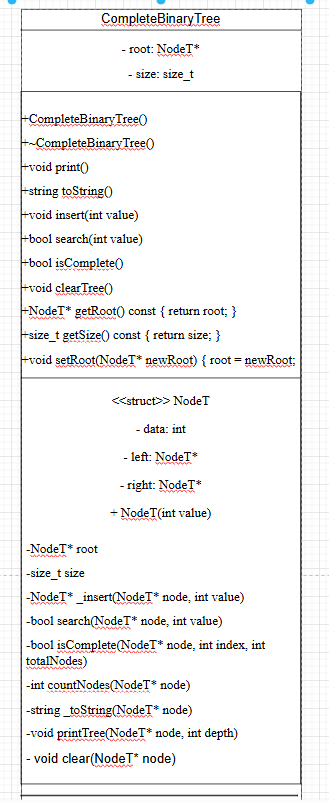
## Очередь:



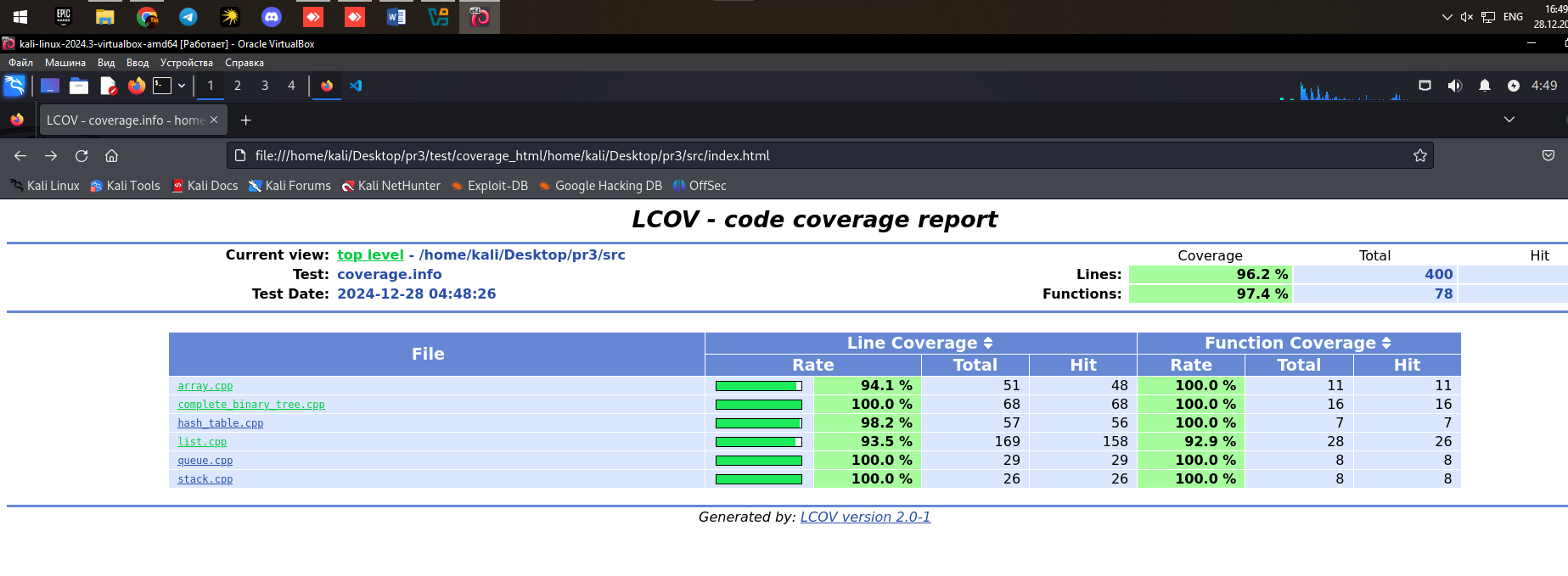
## Хеш-таблица:



## Дерево:

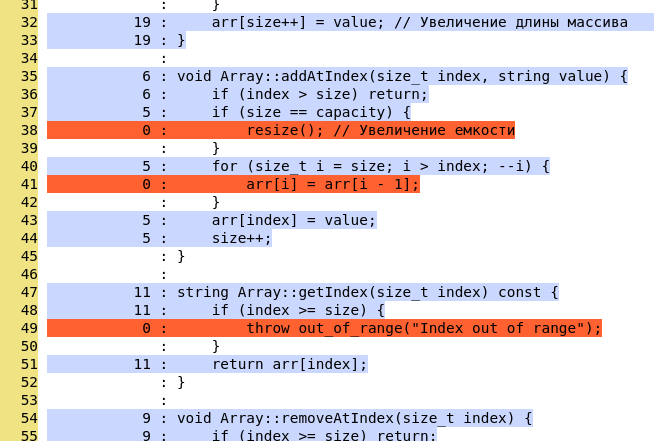
****

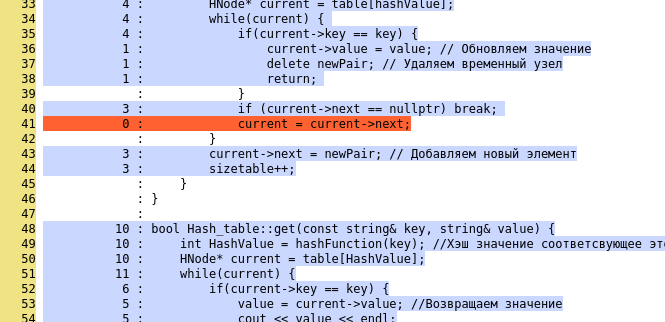
**Работа тестов**

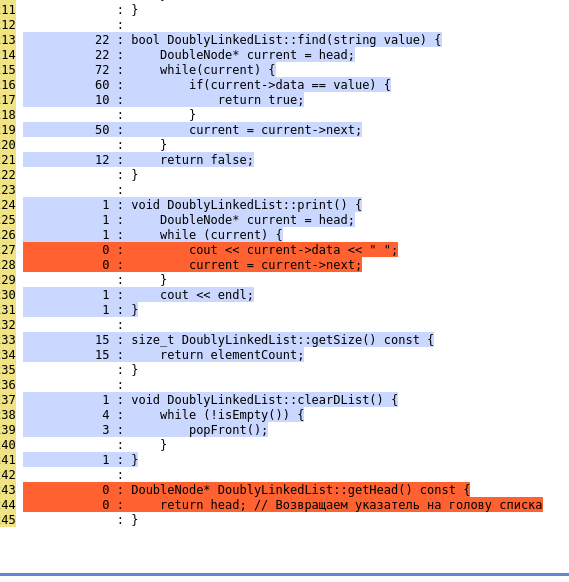
****

**Место где код не заполняется полностью**

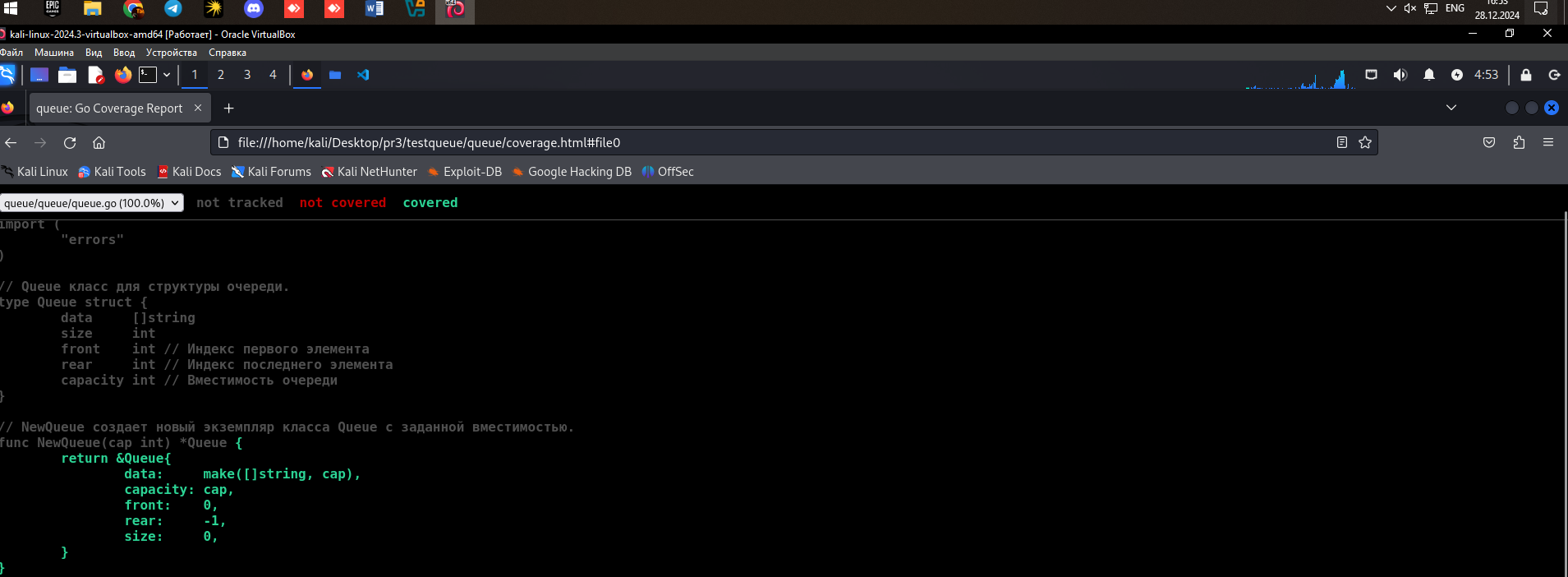
**Array**

****

**Hesh table  
**

**List  
**

**Работ тестов на GO**

****

**Введение**

Компьютеры используются во все более широком спектре областей применения, и их предназначенная и правильная работа часто имеет решающее значение для успеха бизнеса и/или безопасности человека. Поэтому разработка или выбор высококачественных систем и продукции имеет первостепенное значение. Всесторонняя спецификация и оценка качества систем и программных продуктов является ключевым фактором обеспечения надлежащего качества. Это может быть достигнуто путем определения соответствующих качественных характеристик качества с учетом предполагаемого использования систем и программного продукта. Важно, чтобы, когда это возможно, каждая соответствующая характеристика качества системы и программного продукта задавалась и оценивалась с использованием аттестованных или широко принятых показателей.

Поскольку характеристики и показатели качества могут быть полезны не только для оценки систем и программной продукции, но и для определения требований к ним, предшественник SQuaRE - ИСО/МЭК 9126 - был заменен двумя связанными многоуровневыми стандартами: ИСО/МЭК 9126 (качество программной продукции) и ИСО/МЭК 14598 (оценка программной продукции). Накопленный опыт использования двух данных стандартов сформировал следующие основные предпосылки для создания новой серии стандартов SQuaRE:

**Область применения**

Настоящий стандарт представляет собой руководство по использованию серии стандартов под названием "Требования и оценка качества систем и программных средств" (SQuaRE). Цель настоящего стандарта состоит в том, чтобы дать общий обзор содержания серии стандартов SQuaRE, общих эталонных моделей и определений, а также взаимосвязи между документами, что позволит пользователям SQuaRE уяснить положения настоящей серии стандартов и применять ее в соответствии с назначением. Кроме того, настоящий стандарт объясняет переход от устаревших стандартов серий ИСО/МЭК 9126 и ИСО/МЭК 14598 к серии стандартов SQuaRE.

Серия стандартов SQuaRE предназначена для разработчиков, потребителей и независимых оценщиков систем и программной продукции, в частности для тех, кто отвечает за определение требований к качеству систем и программной продукции, а также за оценку качества систем и программной продукции. Рекомендуется, чтобы пользователи стандартов серии SQuaRE, а также ИСО/МЭК 9126 и ИСО/МЭК 14598 использовали настоящий стандарт в качестве руководства для выполнения своих задач.

Данный международный стандарт является результатом пересмотра ИСО/МЭК 9126-1. В него входят те же характеристики качества программного обеспечения с некоторыми поправками:

* область применения моделей качества была расширена, с тем чтобы включить в себя вычислительные системы и качество при использовании с системной точки зрения;
* в качестве характеристики качества при использовании было добавлено "Покрытие контекста" с подхарактеристиками, "Полнота контекста" и "Гибкость";
* как характеристика, а не подхарактеристика функциональности была добавлена "Безопасность", с подхарактеристиками "Конфиденциальность", "Целостность", "Безотказность", "Отслеживаемость" и "Подлинность";
* была добавлена как характеристика "Совместимость" (включая функциональную совместимость и сосуществование);
* были добавлены следующие подхарактеристики: "Функциональная полнота", "Емкость", "Защищенность от ошибки пользователя", "Доступность", "Готовность", "Модульность" и "Возможность многократного использования";
* подхарактеристики соответствия были удалены, поскольку они являются в соответствии с законами и правилами частью общих требований к системе, а не частью характеристики качества;
* модели внутреннего и внешнего качества были объединены в составе модели качества продукта;
* там, где это представилось возможным, специфичные для программного обеспечения определения были заменены на универсальные;
* нескольким характеристикам и подхарактеристикам были даны более точные названия.

Полный перечень изменений приводится в приложении A.

Данный международный стандарт предназначен для применения в сочетании с другими частями международных стандартов серии SQuaRE (ИСО/МЭК 25000 - ИСО/МЭК 25099) и ИСО/МЭК 14598 до тех пор, пока он не заменен серией международных стандартов ИСО/МЭК 2504n.

На рисунке 1 (адаптирован из ИСО/МЭК 25000) показана организация серии международных стандартов SQuaRE, которая представлена семействами стандартов, называемых также разделами.

|  |
| --- |
|  |
| Image |

Рисунок 1 - Организация серии международных стандартов SQuaRE

Серия стандартов SQuaRE состоит из следующих разделов стандартов:

Область применения моделей качества включает в себя спецификацию поддержки и оценку программного обеспечения и преимущественно программных вычислительных систем с разных точек зрения, которые связаны с их приобретением, требованиями, разработкой, использованием, оценкой, поддержкой, обслуживанием, обеспечением качества и управлением им, а также менеджментом и аудитом. Модели могут, к примеру, использоваться разработчиками, приобретателями, персоналом обеспечения качества и управления им, а также независимыми оценщиками, в особенности ответственными за спецификацию и оценку качества программного продукта. Деятельность во время разработки продукции, при которой могут быть использованы модели качества, включает в себя:

1. определение требований к программному обеспечению и системе;
2. подтверждения полноты определения требований;
3. определение целей проектирования программного обеспечения и системы;
4. определение целей тестирования программного обеспечения и системы;
5. идентификацию критериев контроля качества в рамках обеспечения качества;
6. определение критериев приемки программного продукта и/или преимущественно программной вычислительной системы;
7. установление необходимых для этого показателей характеристик качества.

**Соответствие**

Любое требование к качеству, спецификация качества или оценка качества соответствуют настоящему стандарту только в тех случаях, если:

* используются модели качества, определенные в 4.1 и 4.2; или
* используется адаптированная модель качества, все изменения которой обоснованы и для которой обеспечивается отображение на стандартную модель.

Модель качества при использовании

Модель качества при использовании определяет в 4.1 пять характеристик, связанных с результатами взаимодействия с системой: результативность, производительность, удовлетворенность, свободу от риска и покрытие контекста (см. рисунок 3 и таблицу 3). Каждая характеристика применима для различных видов деятельности заинтересованных лиц, например, для взаимодействия оператора или поддержки разработчика.

|  |
| --- |
|  |
| Image |

Рисунок 3 - Модель качества при использовании

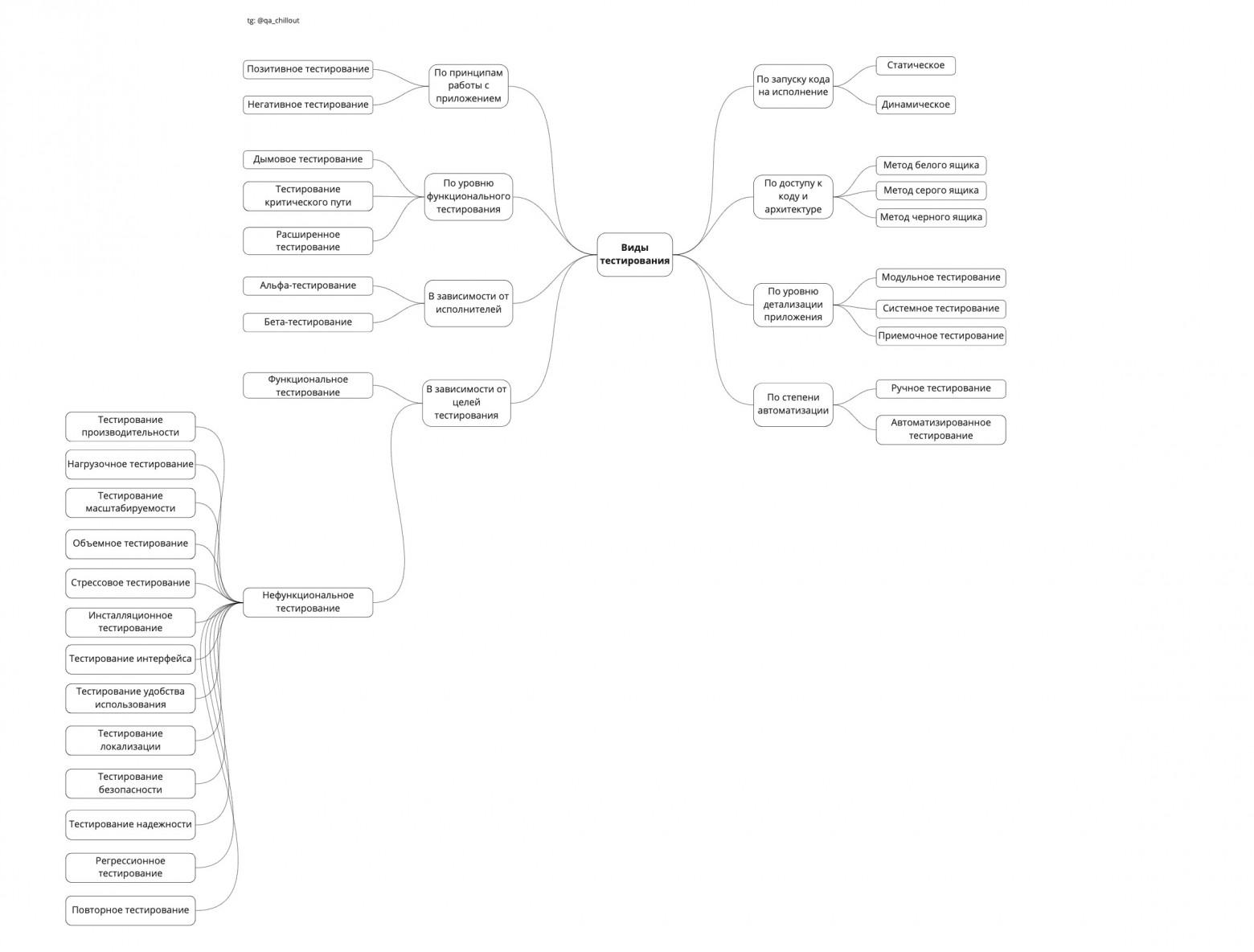
Качество при использовании системы характеризует воздействие продукции (система или программный продукт) на заинтересованную сторону. Оно определяется качествами программного обеспечения, аппаратных средств, операционной среды, а также характеристиками пользователей, задач и социальной среды. Все эти факторы вносят свой вклад в качество системы при использовании.

Термины и определения для каждой характеристики качества при использовании приводятся в 4.1.

Примеры показателей качества при использовании приводятся в техническом отчете ИСО/МЭК ТО 9126-4, который должен быть заменен ИСО/МЭК 25024.

**Модель качества продукта**

Модель качества продукта описана в 4.2. Она сводит свойства качества системы/программного продукта к восьми характеристикам, которыми являются: функциональная пригодность, уровень производительности, совместимость, удобство пользования, надежность, защищенность, сопровождаемость и переносимость (мобильность). Каждая характеристика, в свою очередь, состоит из ряда соответствующих подхарактеристик.

**Основные виды тестирования ПО**  
Вид тестирования — это совокупность активностей, направленных на тестирование заданных характеристик системы или её части, основанная на конкретных целях.  
  


**Классификация по запуску кода на исполнение:**

* + Статическое тестирование — процесс тестирования, который проводится для верификации практически любого артефакта разработки: программного кода компонент, требований, системных спецификаций, функциональных спецификаций, документов проектирования и архитектуры программных систем и их компонентов.
  + Динамическое тестирование — тестирование проводится на работающей системе, не может быть осуществлено без запуска программного кода приложения.

**Классификация по доступу к коду и архитектуре:**

* + Тестирование белого ящика — метод тестирования ПО, который предполагает полный доступ к коду проекта.
  + Тестирование серого ящика — метод тестирования ПО, который предполагает частичный доступ к коду проекта (комбинация White Box и Black Box методов).
  + Тестирование чёрного ящика — метод тестирования ПО, который не предполагает доступа (полного или частичного) к системе. Основывается на работе исключительно с внешним интерфейсом тестируемой системы.

**Классификация по уровню детализации приложения:**

* + Модульное тестирование — проводится для тестирования какого-либо одного логически выделенного и изолированного элемента (модуля) системы в коде. Проводится самими разработчиками, так как предполагает полный доступ к коду.
  + Интеграционное тестирование — тестирование, направленное на проверку корректности взаимодействия нескольких модулей, объединенных в единое целое.
  + Системное тестирование — процесс тестирования системы, на котором проводится не только функциональное тестирование, но и оценка характеристик качества системы — ее устойчивости, надежности, безопасности и производительности.
  + Приёмочное тестирование — проверяет соответствие системы потребностям, требованиям и бизнес-процессам пользователя.

**Классификация по степени автоматизации:**

* + Ручное тестирование.
  + Автоматизированное тестирование.

**Классификация по принципам работы с приложением**

* + Позитивное тестирование — тестирование, при котором используются только корректные данные.
  + Негативное тестирование — тестирование приложения, при котором используются некорректные данные и выполняются некорректные операции.

**Классификация по уровню функционального тестирования:**

* + Дымовое тестирование (smoke test) — тестирование, выполняемое на новой сборке, с целью подтверждения того, что программное обеспечение стартует и выполняет основные для бизнеса функции.
  + Тестирование критического пути (critical path) — направлено для проверки функциональности, используемой обычными пользователями во время их повседневной деятельности.
  + Расширенное тестирование (extended) — направлено на исследование всей заявленной в требованиях функциональности.

**Классификация в зависимости от исполнителей:**

* + Альфа-тестирование — является ранней версией программного продукта. Может выполняться внутри организации-разработчика с возможным частичным привлечением конечных пользователей.
  + Бета-тестирование — программное обеспечение, выпускаемое для ограниченного количества пользователей. Главная цель — получить отзывы клиентов о продукте и внести соответствующие изменения.

**Классификация в зависимости от целей тестирования:**

* + Функциональное тестирование (functional testing) — направлено на проверку корректности работы функциональности приложения.
  + Нефункциональное тестирование (non-functional testing) — тестирование атрибутов компонента или системы, не относящихся к функциональности.  
      
    1. Тестирование производительности (performance testing) — определение стабильности и потребления ресурсов в условиях различных сценариев использования и нагрузок.
    2. Нагрузочное тестирование (load testing) — определение или сбор показателей производительности и времени отклика программно-технической системы или устройства в ответ на внешний запрос с целью установления соответствия требованиям, предъявляемым к данной системе (устройству).
    3. Тестирование масштабируемости (scalability testing) — тестирование, которое измеряет производительность сети или системы, когда количество пользовательских запросов увеличивается или уменьшается.
    4. Объёмное тестирование (volume testing) — это тип тестирования программного обеспечения, которое проводится для тестирования программного приложения с определенным объемом данных.
    5. Стрессовое тестирование (stress testing) — тип тестирования направленный для проверки, как система обращается с нарастающей нагрузкой (количеством одновременных пользователей).
    6. Инсталляционное тестирование (installation testing) — тестирование, направленное на проверку успешной установки и настройки, обновления или удаления приложения.
    7. Тестирование интерфейса (GUI/UI testing) — проверка требований к пользовательскому интерфейсу.
    8. Тестирование удобства использования (usability testing) — это метод тестирования, направленный на установление степени удобства использования, понятности и привлекательности для пользователей разрабатываемого продукта в контексте заданных условий.
    9. Тестирование локализации (localization testing) — проверка адаптации программного обеспечения для определенной аудитории в соответствии с ее культурными особенностями.
    10. Тестирование безопасности (security testing) — это стратегия тестирования, используемая для проверки безопасности системы, а также для анализа рисков, связанных с обеспечением целостного подхода к защите приложения, атак хакеров, вирусов, несанкционированного доступа к конфиденциальным данным.
    11. Тестирование надёжности (reliability testing) — один из видов нефункционального тестирования ПО, целью которого является проверка работоспособности приложения при длительном тестировании с ожидаемым уровнем нагрузки.
    12. Регрессионное тестирование (regression testing) — тестирование уже проверенной ранее функциональности после внесения изменений в код приложения, для уверенности в том, что эти изменения не внесли ошибки в областях, которые не подверглись изменениям.
    13. Повторное/подтверждающее тестирование (re-testing/confirmation testing) — тестирование, во время которого исполняются тестовые сценарии, выявившие ошибки во время последнего запуска, для подтверждения успешности исправления этих ошибок.

Источники

1.Фундаментальная теория тестирования / [Электронный ресурс] // Habr : [сайт]. — URL: <https://habr.com/ru/articles/549054/>

2.СИСТЕМНАЯ И ПРОГРАММНАЯ ИНЖЕНЕРИЯ / [Электронный ресурс] // Консорциум кодекс : [сайт]. — URL: <https://docs.cntd.ru/document/1200121069>

**Вывод**

Во время выполнения лабораторной работы №3 я познакомился и подробно изучил Unit-тесты, и научился пользоваться данными текстами, а также пользоваться библиотекой Boost test и самостоятельно реализовал свои тесты.

гитхаб - https://github.com/Evlusha/lb3\_3.git