Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический
университет имени П. О. Сухого»

Кафедра «Информатика»

ТЕСТИРОВАНИЕ И ВЕРИФИКАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

ПРАКТИКУМ
по выполнению лабораторных работ для студентов специальности 1-40 04 01 «Информатика и технологии программирования» дневной формы обучения

Гомель 2020

УДК 004.4(075.8)
ББК 32.972я73
Т36

Рекомендовано научно-методическим советом
факультета автоматизированных и информационных систем
ГГТУ им. П. О. Сухого
(*протокол № 7 от 04.03.2019 г.*)

Составитель *В. Ф. Велесницкий*

Рецензент: доц. каф. «Информационные технологии» ГГТУ им. П. О. Сухого
канд. техн. наук, доц. *В. В. Комраков*

Тестирование и верификация программного обеспечения : практикум по выпол-
Т36 нению лаборатор. работ для студентов специальности 1-40 04 01 «Информатика и технологии программирования» днев. формы обучения / сост. В. Ф. Велесницкий. – Гомель : ГГТУ им. П. О. Сухого, 2020. – 32 с. – Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: https://elib.gstu.by. – Загл. с титул. экрана.

Составлен в соответствии с учебной программой по одноименной дисциплине. Включает шесть лабораторных работ для закрепления учебного материала по данной дисциплине. Рассмотрены современные подходы к тестированию программных продуктов и требования к ним.

Для студентов специальности 1-40 04 01 «Информатика и технологии программирования» дневной формы обучения.

УДК 004.4(075.8)
ББК 32.972я73

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В производстве любого продукта не обходится без этапа его тестирования. Это же касается и программирования. Как раз дисциплина «Тестирование и верификация программного обеспечения» и освещает этот вопрос.

Курс направлен на изучение задач и подходов их решения в тестировании программных продуктов на различных этапах их жизненного цикла. Данный практикум предназначен для ознакомления с типовыми задачами тестировщиков. Поскольку не последнее место уделяется модульному тестированию, то это касается непосредственно и разработчиков. В большинстве лабораторных работ нет строгого требования к инструментальным средствам выполнения работ, что не ограничивает учащихся в выборе подхода к решению задачи.

Практикум предлагает для выполнения шесть лабораторных работ. В каждой работе есть небольшая теоретическая часть, которая позволит студентам лучше справиться с поставленными задачами.

ЛАБОРАТОРНАЯ РАБОТА №1

ВИДЫ ТЕСТИРОВАНИЯ. ПЛАНИРОВАНИЕ ТЕСТИРОВАНИЯ

Цель работы: Изучить классификацию видов тестирования, практически закрепить эти знания путем генерации тестов различных видов, научиться планировать тестовые активности в зависимости от специфики поставляемой на тестирование функциональности.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Не существует стандартного перечня видов тестирования, однако некоторые из них настолько очевидны, что стали общепринятыми.

Инсталляционное тестирование (installation testing) – проверка всего того, что связано с инсталляцией продукта в систему и удалением продукта из системы.

Регрессионное тестирование (regression testing) – проверка того, что внесённые в приложение изменения не привели к потере работоспособности того, что ранее работало, и/или привели к работоспособности того, что ранее не работало.

Тестирование нового функционала (new feature testing) – проверка того, что заявленный в данном билде новый функционал работает должным образом.

Конфигурационное тестирование (configuration testing) – проверка того, как приложение работает с различным оборудованием и/или собственными настройками.

Тестирование совместимости (compatibility testing) – проверка того, как приложение взаимодействует с другими приложениями и операционной системой. В случае веб-ориентированных приложений особое внимание уделяется совместимости с различными браузерами.

Тестирование удобства использования (usability testing) – проверка того, насколько пользователю удобно и приятно работать с приложением.

Тестирование интернационализации (internationalisation testing) – проверка готовности продукта к переводу на различные языки.

Тестирование локализации (localisation testing) – проверка качества перевода продукта на конкретный язык.

Позитивное тестирование (positive testing) – проверка того, как приложение работает в заведомо “тепличных условиях” (корректные данные, условия работы и т.п.)

Негативное тестирование (negative testing) – проверка того, как приложение реагирует на различные “неприятности” (пропала сеть, повреждён файл и т.п.)

Исследовательское тестирование (exploratory testing) – редкий вид тестирования, основанный на профессиональной интуиции тестировщика, которая может подсказать опасные и малоисследованные способы работы с приложением.

Покажем на примере полотенца возможный перечень вопросов по каждому из перечисленных выше видов тестирования.

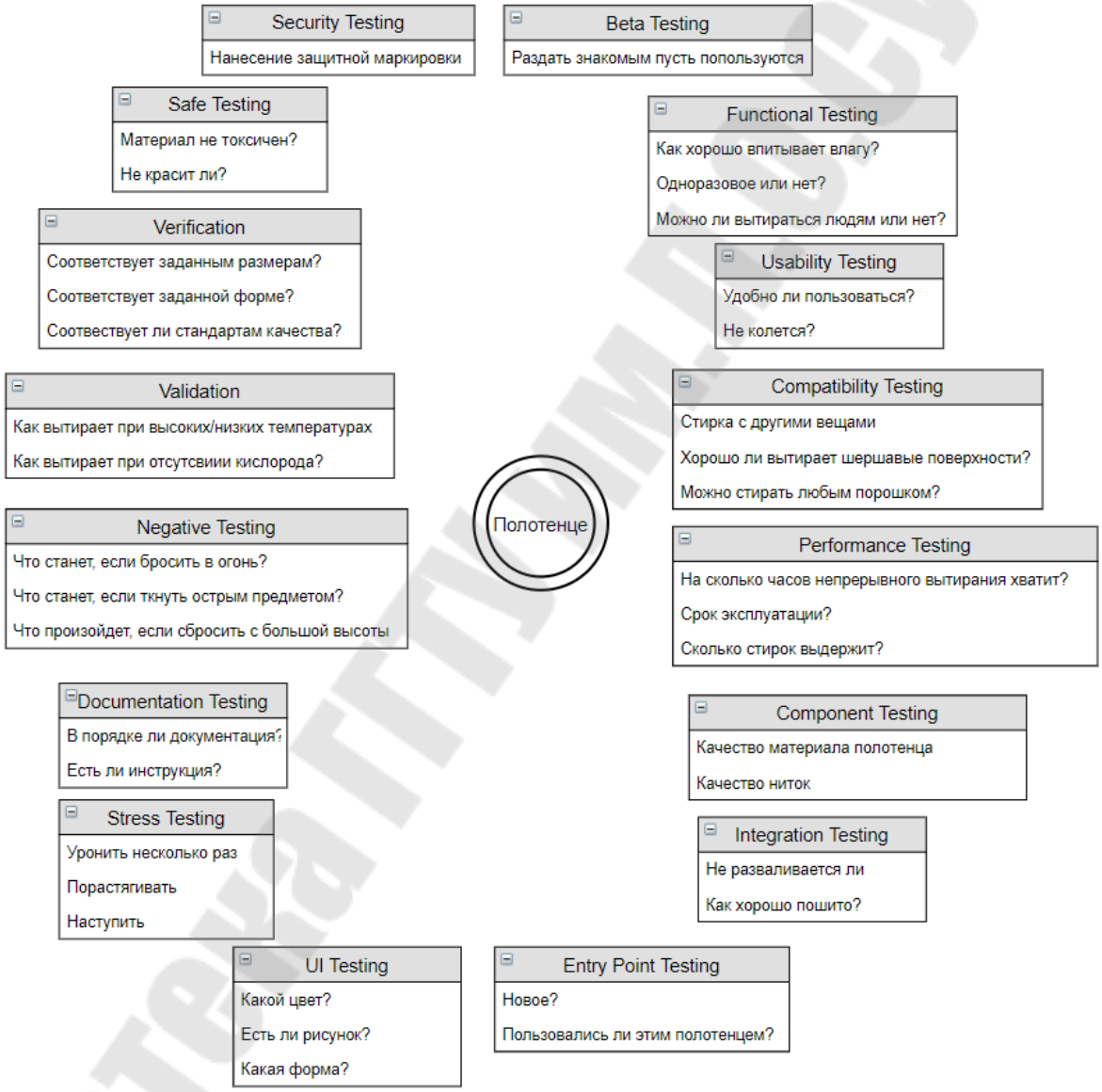


Рис. 1 Пример составления тестов

Задание

Выполнить генерацию тестов (не менее 3 на каждый вид тестирования) различных видов для конкретного объекта реального мира согласно варианту в таблице 1 ниже.

Таблица 1. Варианты заданий.

Номер варианта	Предмет
1.	Ручка
2.	Стул
3.	Стол
4.	Точилка
5.	Чашка
6.	Кухонный нож
7.	Вилка
8.	Ложка
9.	Зубная щетка
10.	Щетка для обуви
11.	Ластик
12.	Маркер
13.	Полотенце
14.	Линейка
15.	Циркуль

Требования к отчету

В отчете должны быть отображены следующие пункты:

1. Задание.
2. Краткие теоретические сведения.
3. Составить не менее 3 вопросов к каждому из перечисленных видов тестирования (см. рис. 1).
4. Выводы.

Контрольные вопросы

1. Что такое тестирование?
2. Какие виды тестирования знаете и в чем их суть?
3. Чем отличается валидация от верификации?
4. В чем суть тестирования безопасности?

ЛАБОРАТОРНАЯ РАБОТА №2
РАЗРАБОТКА ТЕСТОВ

Цель работы: Разработать рабочую тестовую документацию для тестирования приложения.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Тестирование web-приложений обладает рядом особенностей, и при проведении самого процесса тестирования необходимо обращать внимание на следующие аспекты:

1. Единство дизайна.
2. Навигация.
3. Функциональность.
4. Совместимость с браузером.
5. Совместимость с операционной системой.
6. "Дружественность".
7. "Работоспособность".

Единство дизайна. Под единством дизайна понимается не только, а точнее не столько сочетаемость цвета элементов (так как это удел дизайнера), сколько соблюдение выбранной цветовой гаммы, придающей всем страницам сайта "единство". Сюда входят цвета фона (или рисунок), ссылок (в т.ч. посещенной и активной), а также любых других элементов, расположенных на странице. Кроме того, на этом же этапе необходимо оценивать размер и вид используемого шрифта для различных уровней вложения текста (заголовки различных уровне, собственно текст, ссылки, примечания и т.п.) Здесь же имеет смысл оценивать совместимость с дизайном звуков, рисунков и анимации, а также проверять имеет ли место единство отображения при использовании других экранных расширений и глубин цвета.

Навигация. Навигация предполагает тестирование перемещения по сайту, что дает представление о возможности любого пользователя легко найти необходимый раздел, независимо от способа реализации меню (текстовые ссылки, картинки, единая картинка с картой ссылок и др.). На этом же этапе оценивается логичность перемещения между формами, кнопками и другими элементами страницы при помощи TAB, курсорных клавиш и т.п.

Функциональность. Общие подходы к тестированию функциональности веб-страниц аналогичны таковым при тестировании приложений. Ниже приведен примерный перечень основной функциональности веб-страниц:

- ссылки (работоспособность, открытие в том же или новом окне и т.п., полное отсутствие битых ссылок);
- формы (ввод текста, чисел, использование маски, работа с незаполненными полями, длина вводимых символов, корректная работа чек-боксов, комбо-боксов, radio buttons, логичность установок "по умолчанию" и т.д.);
- базы данных (поиск, добавление информации, редактирование, удаление, проверка на дублирование информации);
- доступ (различные роли и права);
- секретность (работа с паролями, передача данных, защита и т.д.);
- кеширование (проверка на установку кеширования и обновления файлов);
- проверка работы с браузером (refresh, forward/back, изменение размеров окна, выбор кодировки, скроллинг, отключение флеша, скрипта);
- фреймы (загрузка страниц, скроллинг и т.п.);
- анимация (наличие, изменение размеров, загрузка и т.д.);
- аудио и видео (наличие, размещение, качество и др.);
- печать (корректно ли печатаются страницы);
- загрузка (как на сайт, так и с сайта);
- экспорт/импорт данных (если есть такая функция);
- интеграция (например: возможность входа на сайт через социальные сети или оплата услуг через paypal account);
- рекомендации для достижения хорошего веб-приложения (в некоторых частных случаях могут не соблюдаться – например, не всегда валидационные сообщения будут красного цвета);
- как внешний вид так и наполнение сайта должны быть надлежащего уровня (с сайтом должно быть приятно и удобно работать);
- все элементы должны быть выровнены, выравнивание одних и тех же элементов в различных частях одного и того же сайта должно быть унифицировано (например: выравнивание кнопок по левому краю);
- шрифт всех ярлыков должен быть одинакового цвета;
- не должно быть битых ссылок (ссылок не ведущих на необходимый контент, несуществующих ссылок);
- каждая под-страница (так называемые "child pages" или "subpages") должна открываться в новом окне;
- на каждой странице должны присутствовать удобные элементы навигации. такие как: хлебные крошки, вперед/назад, сохранить/отменить и т.д.;
- если в таблице содержится больше 10 элементов информации, то должна присутствовать "пагинация" (навигационные ссылки: следующая/предыдущая, первая/последняя и номера страниц);

- у каждой страницы наряду с заголовком должен быть подзаголовок выдержанные в одном стиле (шрифт, цвет и т.д.);
- каждый ярлык должен начинаться с заглавной буквы;
- желательно, чтобы валидационные сообщения были красного цвета;
- все обязательные для заполнения поля должны быть жирного шрифта или помечены специальным символом «*».

Совместимость с браузером. Общеизвестно, что в силу конкуренции, тот или иной браузер имеет нередко даже существенные отличия в отображении одной и той же страницы. Для того, чтобы убедится, что любой пользователь сможет получить всю необходимую информацию требуется проводить тестирование в различных браузерах. Имеются различия и в разных версиях одного и того же браузера. Это также необходимо учитывать при тестировании.

"Дружественность". Под "дружественностью" мы понимаем то, насколько прост, легок в обращении и интуитивно понятен интерфейс сайта: легка ли навигация, доступно ли меню, не используются ли раздражающие пользователя приемы, не много ли всплывающих окон, все ли ссылки являются "рабочими", все ли необходимые данные доступны для пользователя и т.д. Например, если на сайте есть файл для скачивания, то желательно, чтобы пользователь имел возможность заранее знать его размер, мог оценить время закачки.

"Работоспособность". Проверка на "работоспособность" подразумевает оценку скорости загрузки как страниц сайта в целом, так и каждого элемента в отдельности. Сюда включается оценка размера используемых рисунков, html-файлов, аудио и видео файлов, адаптация их к различным типам соединений.

Основные проверки при проведении тестирования web-приложения.

В следующей таблице представлен перечень основных проверок необходимых при проведении тестирования web-приложения, где элементы с отметкой (B) – basic, т.е. являются основными проверками, которые необходимо сделать при тестировании Web-приложений; с отметкой (A) –advanced, т.е проверки, предоставляющие информацию для полноценного тестирования элементов Web-приложений.

Приведем пример в таблице 2.

Таблица 2. Пример чек-листа.

id	Краткое описание	Подробное описание	Шаги по воспроизведению	Ожидаемый результат	Актуальный результат	Важность
1	Грамма-тические и орфогра-фические ошибки	Отсутствие грамматических и орфографических ошибок на страницах сайта	1. Используя сервис http://spell-checker.ru/, проверяем сайт на наличие орфографических ошибок.	1. Отсутствие грамматических и орфографии-ческих ошибок	1. На сайте отсутствуют орфографические и грамматические ошибки	5

Задание

Составить чек-лист (check list) для тестирования приложения.

Варианты логики построения чек-листов:

- рассмотрение различных уровней функционального тестирования;
- рассмотрение отдельных частей (модулей и подмодулей) приложения;
- отдельных требований, групп требований, уровней и типов требований;
- типичных пользовательских сценариев;
- частей или функций приложения, наиболее подверженных рискам.

На основании разработанного чек-листа составить тест-кейсы (test cases) для тестирования приложения. Шаблон оформления тест-кейса произвольный в виде таблицы.

Варианты заданий приведены ниже в таблице 3. Если адрес сайта устарел, следует воспользоваться собственным вариантом, предварительно согласовав с преподавателем. Разработанную тестовую документацию представить в виде отчета и защитить лабораторную работу.

Таблица 3. Варианты заданий.

Номер варианта	Адрес сайта
1.	tut.by
2.	lili.by
3.	chi2b.ru
4.	arabic-calligraphy.ru
5.	aliflailavalaila.ru
6.	somatics-yoga.by
7.	kanrinru.com
8.	slovo-center.ru
9.	fitness-land.ru
10.	bodyboom.ru
11.	sharmanka.by
12.	cafe-favorit.by
13.	fizkult-nn.ru
14.	kruiz-tour.by
15.	gstu.by

Требования к отчету

В отчете должны быть отображены следующие пункты:

1. Задание.
2. Краткие теоретические сведения.
3. Составить чек-листы для тестирования в виде таблице (см. пример в таблице 2) согласно варианту.
4. Выводы.

Контрольные вопросы

1. Назовите основные аспекты в тестировании web-приложений
2. Для чего нужно тестировать web-приложений?
3. Что такое чек-лист?
4. Что такое тест-кейсы?

ЛАБОРАТОРНАЯ РАБОТА №3

ПОИСК И ДОКУМЕНТИРОВАНИЕ ДЕФЕКТОВ

Цель работы. Протестировать приложение и описать найденные дефекты.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Чек-лист – это просто набор идей: идей по тестированию, идей по разработке, идей по планированию и управлению – любых идей.

Тест-кейс – набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства.

Тест – набор из одного или нескольких тест-кейсов (таблица 4).

Таблица 4. Шаблон оформления тест-кейса.

Идентификатор (id)	Приоритет (priority)	Модуль (module)	Подмодуль (submodule)	Заглавие (суть) тест-кейса. Исходные данные, необходимые для выполнения тест-кейса. Шаги тест-кейса. (steps)	Ожидаемый результат по каждому шагу тест-кейса (expected result)

I. Тестирования удобства использования.

A. Ссылка на домашнюю страницу должна быть на каждой странице сайта.

B. Содержание веб-страницы верное, без грамматических и орфографических ошибок.

C. Проверьте, что на сайте нет битых ссылок и изображений.

D. Заголовок должен отображаться на каждой странице.

E. Все поля (текстовые, выпадающие меню, радио-кнопки и т. д.) и кнопки должны быть доступны с клавиатуры, и пользователь должен

быть в состоянии пользоваться сайтом, используя только клавиатуру.

II. Тестирования совместимости.

A. Протестируйте сайт в различных браузерах (IE, Firefox, Chrome, Safari, Opera) и убедитесь, что сайт правильно отображается.

B. Убедитесь, что используемая версия HTML совместима с соответствующими версиями браузеров.

C. Убедитесь, что картинки корректно отображаются в разных браузерах.

D. Убедитесь, что шрифты верно отображаются в разных браузерах.

E. Убедитесь, что Java Script код работает в разных браузерах.

F. Проверьте анимированные GIF в разных браузерах.

Таблица 5. Пример тест-кейсов.

1	2	3	4	5	6
ID	Priority	Module	Submodule	Steps	Expected result
I-A	Высокий	1.Главная 2.Университет 3.Образование 4.Наука 5.Молодежная политика 6.Международное сотрудничество 7.Контакты	Header	Проверить наличие ссылки на домашнюю страницу на страницах сайта.	1. Ссылка присутствует 2. Ссылка присутствует 3. Ссылка присутствует 4. Ссылка присутствует 5. Ссылка присутствует 6. Ссылка присутствует 7. Ссылка присутствует

1	2	3	4	5	6
I-B	Высокий	Главная страница	Отсутствие грамматических ошибок	1.Проверить главную страницу на отсутствие орфографических ошибок	1.Используя сервис http://spell-checker.ru/, проверяем сайт на наличие орфографических ошибок. Грамматические и орфографические ошибки отсутствуют
I-C	Высокий	Главная страница	Отсутствие битых ссылок	1.Проверить работоспособность ссылок на главной странице сайта	1.Битые ссылки отсутствуют.

Задание

1. Протестировать приложение в соответствии с составленной ранее тестовой документацией из лабораторной работы №2.
2. Составить отчет о дефектах (bug report).

Требования к отчету

В отчете должны быть отображены следующие пункты:

1. Задание.
2. Краткие теоретические сведения.
3. Составить тест-кейсы в виде таблице (см. пример в таблице 5) согласно варианту.
4. Выводы.

Контрольные вопросы

1. Что такое чек-лист?
2. Что такое тест-кейсы?
3. Что такое тест?

ЛАБОРАТОРНАЯ РАБОТА №4

ДОКУМЕНТИРОВАНИЕ РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ

Цель работы. Составить итоговый отчет о результатах тестирования приложения.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Дефект (Defect, Bug Report) – это документ, в котором описано неправильное поведение тестируемой системы или ПО. Не нужно путать дефект с ошибкой\багом. В терминологии тестирования ПО ошибка или баг – это непосредственно проблема ПО (например, ошибка в коде), а дефект – это описание данной проблемы.

Для чего необходимо описывать дефекты?

1. Описывая ошибку, мы даем возможность другому человеку воспроизвести тот же результат и убедиться в наличии проблемы. Такое описание необходимо аналитикам, чтобы убедиться в том, что требования действительно не выполняются а также разработчикам, чтобы понять что и как им исправлять.

2. Когда мы получаем исправленное ПО для повторной проверки, детальное описание ошибки дает возможность ничего не забыть и убедиться, что все действительно исправлено.

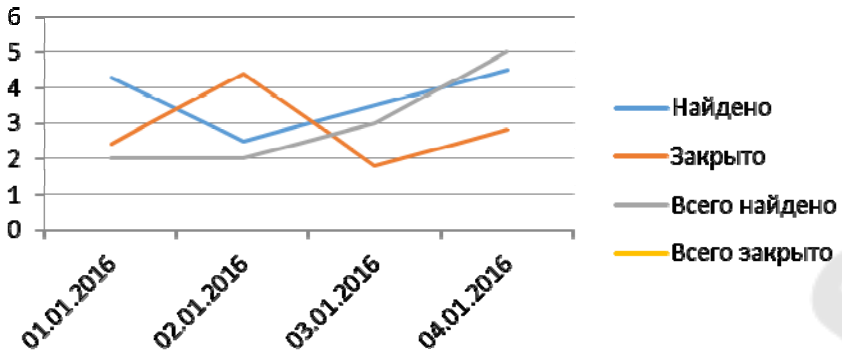
3. Имея описание ошибки (даже уже исправленной ранее) мы можем повторно проверить, что она не повторяется в новой версии ПО.

Задание.

1. Составить отчет о результатах тестирования (test result report).

Отчет должен содержать следующие разделы:

- краткое описание (summary);
- команда тестировщиков (test team);
- описание процесса тестирования (testing process description);
- расписание (timetable):
- статистика по новым дефектам (new defects statistics):
- список новых дефектов (new defects list):
- статистика по всем дефектам (overall defects statistics) (см. статистику по новым дефектам);



- рекомендации (recommendations);
- приложения (appendixes);

Метрики:

Успешное прохождение тест-кейсов:

$$T^{SP} = \frac{T^{Success}}{T^{Total}} \cdot 100\%,$$

где

T^{SP} — процентный показатель успешного прохождения тест-кейсов,

$T^{Success}$ — количество успешно выполненных тест-кейсов,

T^{Total} — общее количество выполненных тест-кейсов.

Минимальная границы значений:

1. Начальная фаза проекта: 10%.
2. Основная фаза проекта: 40%.
3. Финальная фаза проекта: 80%.

Выполнение тест-кейсов:

$$T^{E} = \frac{T^{Executed}}{T^{Planed}} \cdot 100\%,$$

где

T^{E} – процентный показатель выполнения тест-кейсов,

$T^{Executed}$ – количество выполненных тест-кейсов,

T^{Planed} – количество тест-кейсов, запланированных к выполнению.

Уровни (границы):

- Минимальный уровень: 80 %.
- Желаемый уровень: 95–100 %.

Общее устранение дефектов:

$$D_{Level}^{FTP} = \frac{D_{Level}^{Closed}}{D_{Level}^{Found}} \cdot 100\%,$$

где D_{Level}^{FTP} – процентный показатель устранения дефектов уровня важности за время существования проекта,

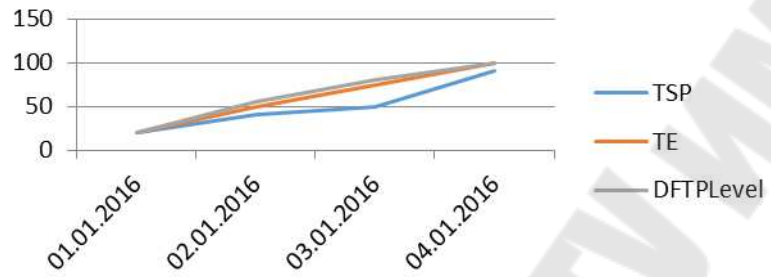
D^{Closed}_{Level} – количество устранённых за время существования проекта дефектов уровня важности,

D^{Found}_{Level} – количество обнаруженных за время существования проекта дефектов уровня важности.

Минимальные границы значений даны в таблице 6.

Таблица 6. Минимальные границы значений.

		Важность дефекта			
		низкая	средняя	высокая	критическая
Фаза проекта	Начальная	10%	40%	50%	80%
	Основная	15%	50%	75%	90%
	Финальная	20%	60%	100%	100%



Требования к отчету

В отчете должны быть отображены следующие пункты:

1. Задание.
2. Краткие теоретические сведения.
3. Провести исследования согласно заданию. По результатам составить дианграмму.
4. Выводы.

Контрольные вопросы

1. Что такое дефект?
2. Для чего необходимо описывать дефекты?
3. Что такое жизненный цикл дефекта?

ЛАБОРАТОРНАЯ РАБОТА №5

АВТОМАТИЗИРОВАННОЕ ТЕСТИРОВАНИЕ WEB-ПРИЛОЖЕНИЙ.

Цель работы. Получить навыки автоматизированного тестирования web-приложений с использованием *SeleniumIDE*.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

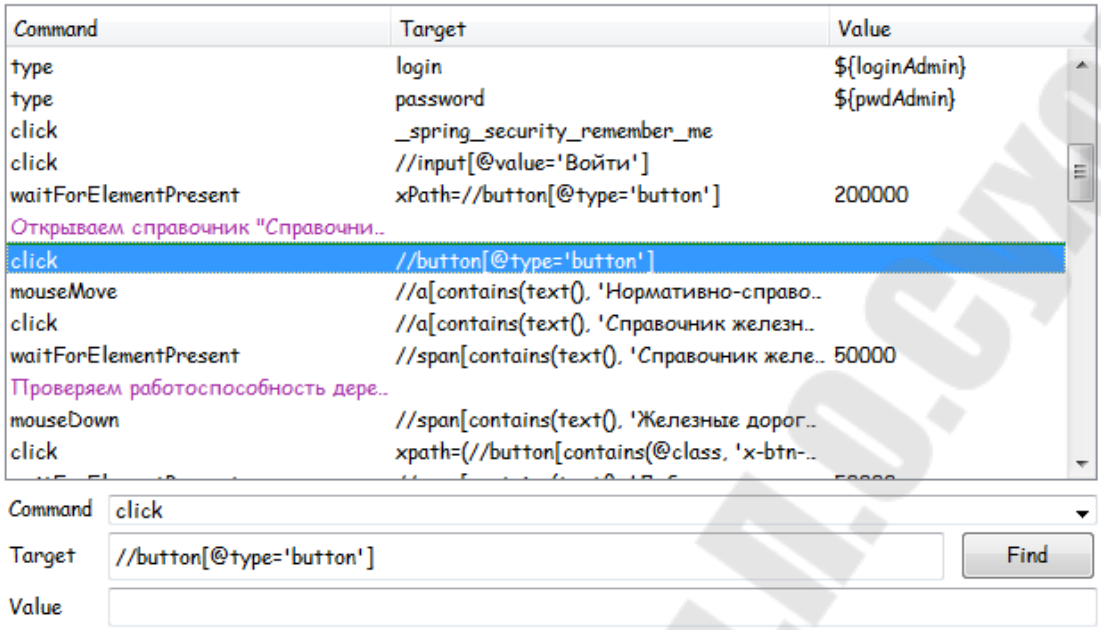
Selenium IDE (*Integrated Development Environment*, интегрированная среда разработки) – это инструмент, используемый для разработки тестовых сценариев (записи определенной последовательности действий). Он представляет собой простое в использовании дополнение к браузеру *Firefox* и, в целом, является наиболее эффективным способом разработки тестовых сценариев. Дополнение среди прочего содержит контекстное меню, позволяющее пользователю сначала выбрать любой элемент интерфейса на отображаемой браузером в данный момент странице, а затем выбрать команду из списка команд *Selenium* с параметрами, предустановленными в соответствии с выбранным элементом. Это не только экономит время, но и дает замечательную возможность для изучения языка команд *Selenium*.

Во время записи *Selenium IDE* автоматически вставляет команды в тестовый сценарий, основываясь на действиях пользователя. Обычно это команды:

- при нажатии на ссылку – команды *click* или *clickAndWait*;
- при вводе данных – команда *type*;
- при выборе опции из выпадающего списка – команда *select*;
- при нажатии на чекбокс или переключатель – команда *click.*

В тестовых сценариях бывает необходимо выполнить проверку параметров веб-страницы. Для этого необходимы команды *assert* и *verify*.

При включенном в *Selenium IDE* режиме записи, переключитесь на браузер с тестируемым веб-приложением и щелкните правой кнопкой мыши в любом месте на странице. Появится контекстное меню с командами *verify* и/или *assert*:



Задания

Вариант 1.

1. Зайти на главную страницу сайта vilka.by.
2. Проверить, что главная страница содержит текст “Календарь событий” и “Сегодня с вами работает”.
3. Проверить, что главная страница содержит ссылку с названием “Аксессуары” на страницу с аксессуарами.
4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).
5. Создать переменную со значением “золотая рыбка”. На главной странице сайта сделать активным поле ввода “Поиск”, в поле для поиска внести значение переменной и нажать кнопку найти. Посчитать и вывести количество результатов поиска (использовать команду storeXpathCount).
6. Если на главной странице присутствует кнопка/ссылка “Вход”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует кнопка “Выйти”, нажать кнопку “Обо мне” и на открытой странице выбрать аватар и сохранить изменения.
7. Проверить работу кнопки/ссылки“Выйти”.

Вариант 2.

1. Зайти на главную страницу сайта http://kartaby.by/.
2. Проверить, что главная страница содержит текст “Последнее на форуме” и “Дорогие Друзья”.

3. Проверить, что главная страница содержит ссылку с названием “Форум” на соответствующую страницу.

4. Проверить, что на главной странице есть хотя бы одна кнопка (button).

5. Перейти по ссылкам “Форум”-> “Поиск”. Создать переменную со значением “дорога”. Ввести в поле “Ключевые слова” значение переменной и нажать кнопку “Найти”. Посчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).

6. Если на главной странице присутствует кнопка/ссылка“Войти”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует кнопка “Покинуть”, перейти по ссылке “Входящие”в верхнем левом углу и проверить, что на открытой странице присутствует текст “В папке “Входящие” нет сообщений”.

7. Проверить работу кнопки/ссылки“Покинуть”.

Вариант 3.

1. Зайти на главную страницу сайта onliner.by.

2. Проверить, что главная страница содержит текст “Мобильные телефоны” и “Вакансии”.

3. Проверить, что главная страница содержит ссылку с названием “Каталог” на страницу с каталогом.

4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).

5. Создать переменную со значением “велосипед”. На главной странице сайта сделать активным поле ввода для поиска, в поле для поиска внести значение переменной.Посчитать и вывести количество результатов поиска на первой странице (использовать команду storeXpathCount).

6. Если на главной странице присутствует кнопка/ссылка“Вход”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует кнопка/ссылка “Выйти”, сохранить в новую переменную номер вашей учетной записи и вывести его в Log.

7. Проверить работу кнопки/ссылки “Выйти”.

Вариант 4.

1. Зайти на главную страницу сайта edu.gstu.by.

2. Проверить, что главная страница содержит текст “Учебный портал ГГТУ имени П.О. Сухого” и “Календарь”.

3. Проверить, что главная страница содержит ссылку с названием “edu.gstu.by” на главную страницу.

4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).

5. Создать переменную со значением “Тестирование и верификация ПО”. Внизу главной странице сайта сделать активным поле ввода “Поиск курса”, в поле для поиска внести значение переменной и нажать кнопку “Применить”. Убедиться, что отображаются результаты поиска. Сохранить в новую переменную категорию курса и вывести его в Log.

6. Если на главной странице присутствует кнопка/ссылка“Вход”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствуют имя и фамилия пользователя, т.е. пользователь авторизован,посчитать и вывести количество курсов у авторизованного пользователя (использовать команду storeXpathCount).

7. Проверить работу кнопки/ссылки “Выход”.

Вариант 5.

1. Зайти на главную страницу сайта mail.ru.

2. Проверить, что главная страница содержит текст “Одноклассники” и “Спорт”.

3. Проверить, что главная страница содержит ссылку с названием “Ответы” на соответствующую страницу.

4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).

5. Создать переменную со значением “велосипед”. Сделать активным поле ввода для поиска, в поле для поиска внести значение переменной, выбрать категорию “приложения” и нажать кнопку “Найти”. Подсчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).

6. Если на главной странице присутствует кнопка/ссылка “Войти”/”Вход”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным. Если на главной странице присутствует кнопка “Выход”, т.е. пользователь авторизован, вывести количество входящих писем в его почтовом ящике.

7. Проверить работу кнопки/ссылки“Выход”.

Вариант 6.

1. Зайти на главную страницу сайта rambler.ru.

2. Проверить, что главная страница содержит текст “как открыть сберегательный вклад”.

3. Проверить, что главная страница содержит ссылку с названием “Пророчества” на соответствующую страницу.

4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).

5. Создать переменную со значением “велосипед”. Сделать активным поле ввода для поиска, в поле для поиска внести значение переменной, выбрать категорию “поиск по сайту” и нажать кнопку “Найти”. Подсчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).

6. Если на главной странице присутствует кнопка/ссылка “Войти”/“Вход”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует имя пользователя, т.е. пользователь авторизован, вывести текущую температуру в городе, указанную на главной странице.

7. Проверить работу кнопки/ссылки“Выйти”.

Вариант 7.

1. Зайти на главную страницу сайта yandex.by.

2. Проверить, что главная страница содержит текст “Телепрограмма” и“Афиша”.

3. Проверить, что главная страница содержит ссылку с названием “Маркет” на соответствующую страницу.

4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).

5. Создать переменную со значением “велосипед”. Сделать активным поле ввода для поиска, в поле для поиска внести значение переменнойи нажать кнопку “Найти”. Подсчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).

6. Если на главной странице присутствует кнопка/ссылка “Войти”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует имя пользователя/почта, т.е. пользователь авторизован, вывести количество писем, отображающихся во входящих на первой странице (в почтовом ящике).

7. Проверить работу кнопки/ссылки“Выйти”.

Вариант 8.

1. Зайти на главную страницу сайта shop.by.
2. Проверить, что главная страница содержит текст “торговый портал”.
3. Проверить, что главная страница содержит ссылку с названием “Авто” на соответствующую страницу.
4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).
5. Создать переменную со значением “велосипед”. Сделать активным поле ввода для поиска, в поле для поиска внести значение переменнойи нажать кнопку “Найти”. Подсчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).
6. Если на главной странице присутствует кнопка/ссылка “Войти”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует имя пользователя/почта, т.е. пользователь авторизован, зайти в раздел “Мои отзывы на товары” и проверить, что значение заголовкаоткрытой страницы совпадает с “МойShop.by – Мои отзывы на товары”.
7. Проверить работу кнопки/ссылки“Выйти”.

Вариант 9.

1. Зайти на главную страницу сайта http://www.ikea.com/ru/ru/.
2. Проверить, что главная страница содержит текст “Добро пожаловать в ИКЕА Россия!”.
3. Проверить, что главная страница содержит ссылку с названием “Список покупок” на соответствующую страницу.
4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).
5. Создать переменную со значением “ковер”. Сделать активным поле ввода для поиска, в поле для поиска внести значение переменной и нажать кнопку “Найти”. Подсчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).
6. Если на главной странице присутствует кнопка/ссылка “Вход”, осуществить вход в систему со своими логином и паролем. Убедиться, что при навигации по сайту пользователь остается авторизованным. Если на главной странице присутствует кнопка/ссылка“Выход”, т.е. пользователь авторизован, добавить в список покупок любой товар и после этого проверить его наличие в списке покупок.
7. Проверить работу кнопки/ссылки “Выход”.

Вариант 10.

1. Зайти на главную страницу сайта https://www.21vek.by/.
2. Проверить, что главная страница содержит текст “онлайн-гипермаркет”.
3. Проверить, что главная страница содержит ссылку с названием “детям” на соответствующую страницу.
4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).
5. Создать переменную со значением “стиральная машина”. Сделать активным поле ввода для поиска, в поле для поиска внести значение переменнойи нажать кнопку “Найти”. В результатах поиска выбрать категорию “Samsung”, подсчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).
6. Если на главной странице присутствует кнопка/ссылка “Войти”, осуществить вход в систему. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует кнопка/ссылка “Мое”, т.е. пользователь авторизован, добавить в корзину любой товар и после этого проверить его наличие в корзине.
7. Проверить работу кнопки/ссылки“Выход”.

Вариант 11.

1. Зайти на главную страницу сайта http://5element.by/.
2. Проверить, что главная страница содержит текст “единый мобильный номер”.
3. Проверить, что главная страница содержит ссылку с названием “Клуб покупателей”на соответствующую страницу.
4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).
5. Создать переменную со значением “стиральная машина”. Сделать активным поле ввода для поиска, в поле для поиска внести значение переменнойи запустить поиск. Подсчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).
6. Если на главной странице присутствует кнопка/ссылка “Войти в личный кабинет”, осуществить вход в систему. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует кнопка/ссылка “Личный кабинет”, т.е. пользователь авторизован, добавить в корзину любой товар и после этого проверить его наличие в корзине.
7. Проверить работу кнопки/ссылки“Выйти”.

Вариант 12.

1. Зайти на главную страницу сайта http://zht.by/.

2. Проверить, что главная страница содержит текст “Служба поддержки”.

3. Проверить, что главная страница содержит ссылку с названием “бытовая химия” на соответствующую страницу.

4. Проверить, что на главной странице сайта есть хотя бы одно поле ввода (input) и одна кнопка (button).

5. Создать переменную со значением “шампунь”. Сделать активным поле ввода для поиска, в поле для поиска внести значение переменнойи запустить поиск. Подсчитать и вывести количество результатов на первой странице (использовать команду storeXpathCount).

6. Если на главной странице присутствует кнопка/ссылка “Вход”, осуществить вход в систему. Убедиться, что при навигации по сайту пользователь остается авторизованным.Если на главной странице присутствует кнопка/ссылка “Личный кабинет”/”Выход”, т.е. пользователь авторизован, добавить в корзину любой товар и после этого проверить его наличие в корзине.

7. Проверить работу кнопки/ссылки “Выход”.

Требования к отчету

В отчете должны быть отображены следующие пункты:

1. Задание.
2. Краткие теоретические сведения.
3. Описать выполнение каждого пункта задания варианта с прикреплением результатов в виде скриншотов.
4. Выводы.

Контрольные вопросы

1. Что такое SeleniumIDE?
2. Для чего нужен Selenium?
3. Способы создания тестовых сценариев в SeleniumIDE.

ЛАБОРАТОРНАЯ РАБОТА №6
МОДУЛЬНОЕ ТЕСТИРОВАНИЕ

Цель работы. Изучить технологии модульного тестирования.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Unit testing **(юнит тестирование или модульное тестирование)** – заключается в изолированной проверке каждого отдельного элемента путем запуска тестов в искусственной среде. Для этого необходимо использовать драйверы и заглушки. Поэлементное тестирование – первейшая возможность реализовать исходный код. Оценивая каждый элемент изолированно и подтверждая корректность его работы, точно установить проблему значительно проще чем, если бы элемент был частью системы.

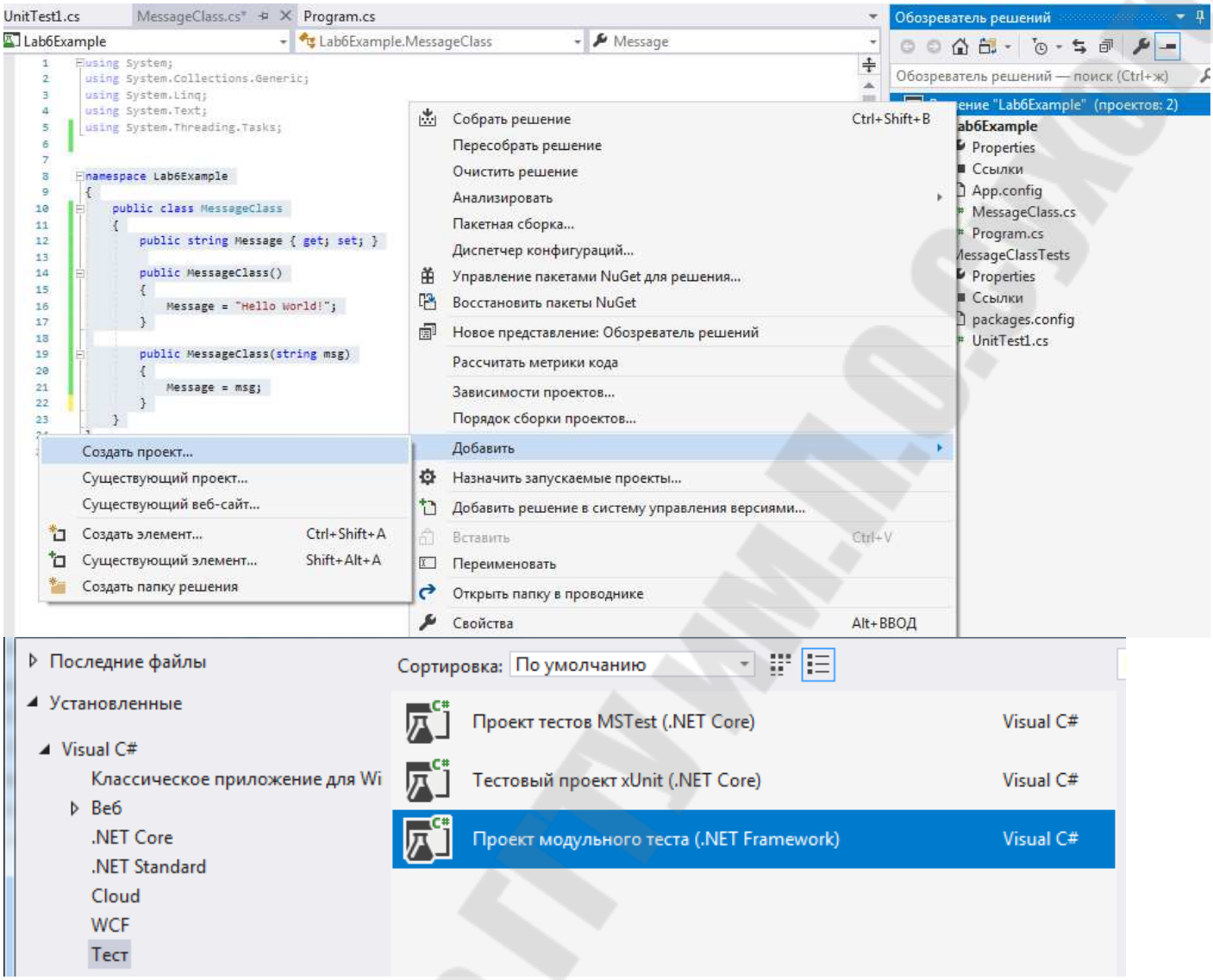
Unit (Элемент) – наименьший компонент, который можно скомпилировать.

Пример на C# в Visual Studio.

1. Создадим простенький класс *MessageClass*

```
namespace Lab6Example
{
    public class MessageClass
    {
        public string Message { get; set; }

        public MessageClass()
        {
            Message = "Hello World!";
        }

        public MessageClass(string msg)
        {
            Message = msg;
        }
    }
}
```

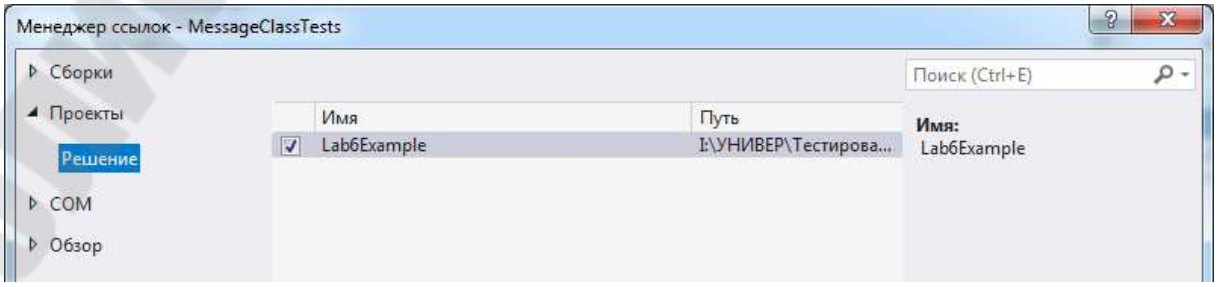
2. Создадим проект модульного тестирования:



3. Добавьте ссылку на приложение MessageClass в проект MessageClass Tests.

a. В обозревателе решений выберите проект *MessageClassTests* и в контекстном меню выберите команду *Добавить ссылку*.

b. В диалоговом окне *Добавить ссылку – MessageClassTests* разверните узел *Решение* и выберите *Проекты*. Затем выберите элемент *Lab6Example*:

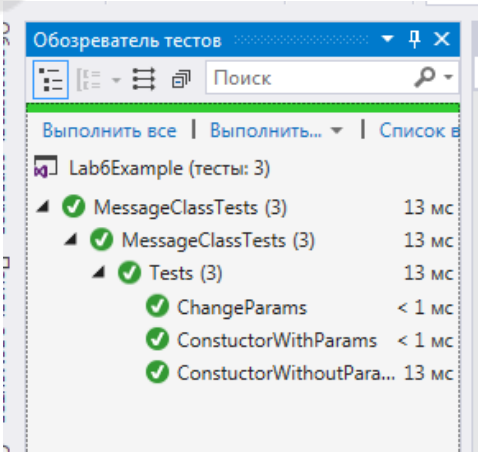


4. Код *MessageClassTests*:

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Lab6Example;

namespace MessageClassTests
{
    [TestClass]
    public class Tests
    {
        [TestMethod]
        public void ConstuctorWithoutParams()
        {
            MessageClass mc = new MessageClass();
            Assert.AreEqual("Hello World!", mc.Message);
        }

        [TestMethod]
        public void ConstuctorWithParams()
        {
            const string msg = "It's my message";
            MessageClass mc = new MessageClass(msg);
            Assert.AreEqual(msg, mc.Message);
        }

        [TestMethod]
        public void ChangeParams()
        {
            MessageClass mc = new MessageClass();
            mc.Message = "abc";
            Assert.AreEqual("abc", mc.Message);
        }
    }
}
```

5.Отобразить обозреватель

6.Выполнить тесты:



Задание

Разработать класс в соответствии с вариантом (таблица 7) (выбирается после согласования с преподавателем). Создать не менее 10 unit-тестов для тестирования методов разработанного типа. Код должен содержать поясняющие комментарии.

Таблица 7. Варианты заданий.

1.	Разработать тип для работы с матрицами. Реализовать методы, позволяющие выполнять операции сложения, вычитания матриц, предусмотрев возможность их выполнения, в противном случае должно генерироваться исключение.
2.	Разработать тип для работы с матрицами. Реализовать методы, позволяющие выполнять операции умножения матриц, умножения матрицы на число, предусмотрев возможность их выполнения, в противном случае должно генерироваться исключение.
3.	Разработать тип для вычисления определенных интегралов. Реализовать методы, позволяющие выполнять вычислять определенный интеграл методом правых прямоугольников. Предусмотрев возможность их выполнения, в противном случае должно генерироваться исключение.
4.	Разработать тип для работы с квадратными уравнениями. Реализовать методы для решения квадратных уравнений. Предусмотрев возможность их решения, в противном случае должно генерироваться исключение.
5.	Разработать тип для работы с отрезками. Реализовать методы, позволяющие определять пересечение отрезков. Предусмотреть корректность задания их координат, в противном случае должно генерироваться исключение.
6.	Разработать тип для работы с матрицами. Реализовать методы, позволяющие находить определитель матрицы, предусмотрев возможность их выполнения, в противном случае должно генерироваться исключение.
7.	Разработать тип комплексное число. Реализовать методы, позволяющие находить сумму и произведение чисел, предусмотрев возможность их выполнения, в противном случае должно генерироваться исключение.
8.	Разработать тип вектор. Реализовать методы, позволяющие находить векторное произведение между двумя векторами, в противном случае должно генерироваться исключение.
9.	Разработать тип вектор.

	Реализовать методы, позволяющие длину вектора, в противном случае должно генерироваться исключение.
10.	Разработать тип определенный интеграл. Реализовать методы, позволяющие вычислять интеграл методом трапеций, в противном случае должно генерироваться исключение.
11.	Разработать тип для нахождения площади пятиугольника. Реализовать методы, позволяющие вычислять площадь пятиугольника по заданным координатам, в противном случае должно генерироваться исключение.
12.	Разработать тип определенный интеграл. Реализовать методы, позволяющие вычислять интеграл методом Моне-Карло, в противном случае должно генерироваться исключение.
13.	Разработать тип для нахождения площади шестиугольника. Реализовать методы, позволяющие вычислять площадь шестиугольника по заданным координатам, в противном случае должно генерироваться исключение.
14.	Разработать тип для работы с матрицами. Реализовать методы, позволяющие находить обратную матрицу, предусмотрев возможность их выполнения, в противном случае должно генерироваться исключение.
15.	Разработать тип для проверки натуральных чисел. Реализовать методы, определяющие является ли натуральное число простым или совершенным. В случае некорректных данных должно генерироваться исключение.

Требования к отчету

В отчете должны быть отображены следующие пункты:

1. Задание.
2. Краткие теоретические сведения.
3. Вставить листинги классов, а также скиншоты выполненных заданий.
4. Выводы.

Контрольные вопросы

1. Что такое модульное тестирование?
2. Что такое Unit-тест?

СПИСОК ИСТОЧНИКОВ

1. Куликов, С.С. Тестирование программного обеспечения. Базовый курс / С.С. Куликов. – Минск: Четыре четверти, 2017. – 312 с.

2. Вигерс, К. Разработка требований к программному обеспечению. 3-е изд., дополненное / Вигерс К., Битти Д. Пер. с англ. – М.: Издательство «Русская редакция»; СПб.: БХВ-Петербург, 2014. – 736 ст.

3. Нильсон, Я. Web-дизайн: удобство использования Web-сайтов / Я. Нильсон, Х. Лоранжер. Пер. с англ. – М.: ООО «И.Д. Вильямс», 2007. – 368 с.

ТЕСТИРОВАНИЕ И ВЕРИФИКАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

ПРАКТИКУМ
по выполнению лабораторных работ для студентов специальности 1-40 04 01 «Информатика и технологии программирования» дневной формы обучения

Составитель **Велесницкий** Василий Федорович

Подписано к размещению в электронную библиотеку
ГГТУ им. П. О. Сухого в качестве электронного
учебно-методического документа 24.02.20.

Рег. № 35Е.
http://www.gstu.by