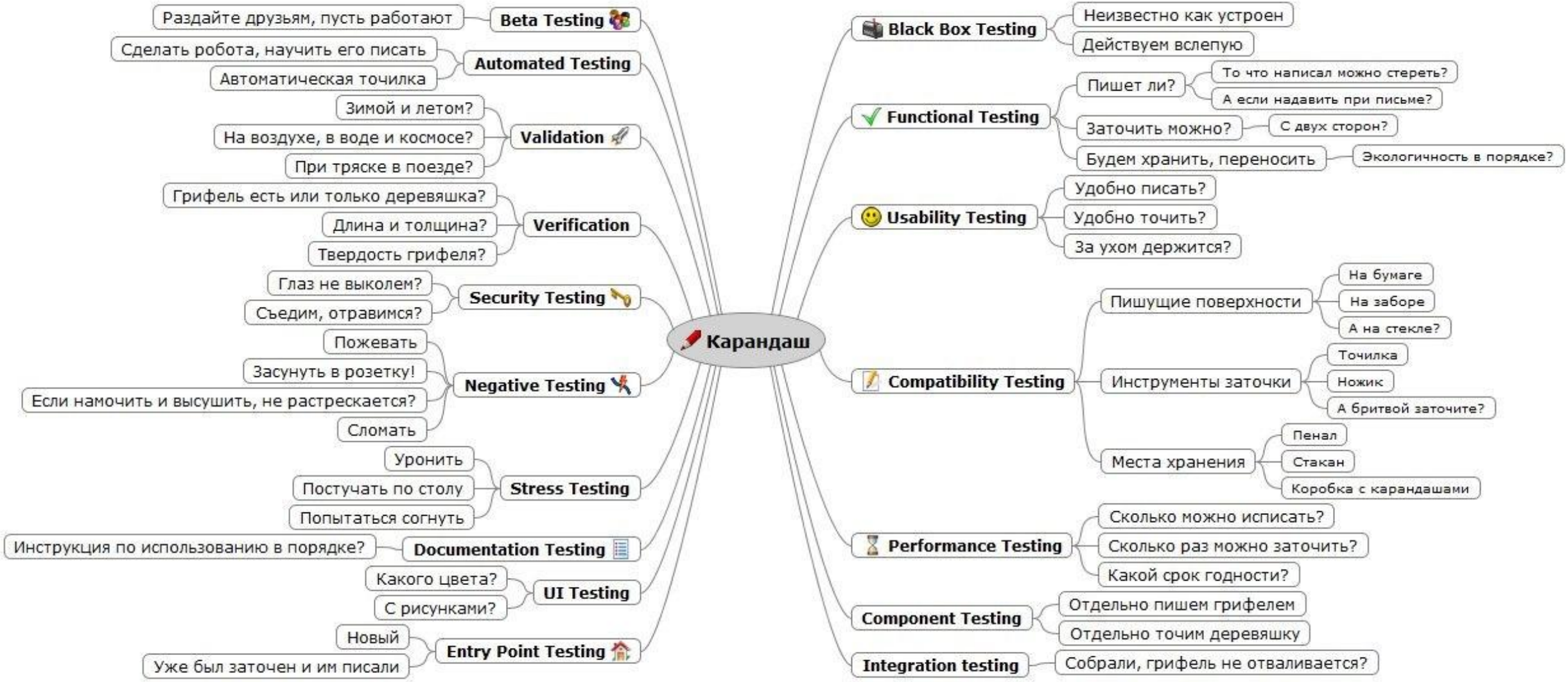
Разработка тестов

УЧИМСЯ РАЗРАБАТЫВАТЬ ТЕСТЫ

Как можно протестировать карандаш?

Как можно протестировать карандаш?



Тестовая документация

Зачем?

- для выполнения тестов
- для возможности увидеть работу команды QA
- для обеспечения контроля качества всего проекта

Кому?

- PM (работа команды, кач-во проекта)
- Dev
- Заказчик
- самим QA
- новичкам QA

Кто делает?

- QA

Виды тестов

- Тесты на основе требований (Requirements based tests)
- Функциональные тесты (functional test)
- Сравнительные («параллельные») тесты (parallel testing)
- Сценарные тесты (scenario tests)
- Тесты ошибочных ситуаций (fault injection tests)
- Тесты интерфейса (interface tests, GUI tests)
- Тесты удобства использования (usability tests)
- Тесты документации (documentation tests)
- Стрессовые тесты (stress tests)
- Тесты производительности (performance tests)
- Конфигурационные тесты (configuration tests)
- Законодательные тесты (regulation tests)

С чего начать?

- Информация для старта: браузер, скоуп, дизайн и доки, время на тест, тип тестовой документации.
- Подход к тестированию. Непонятно – спроси.
- Маркировка спецификации, понимание приоритетов спецификации и дизайна
- Дефекты вносить по ходу тестирования (можно скрин для начала)

Рекомендации по разработке тестов

- Начинайте с простых и очевидных тестов
- Если программа хорошо справляется с очевидными задачами, поставьте перед ней неочевидную
- Если остается время, занимайтесь исследовательским тестированием
- Учитесь на своём и чужом опыте

Последовательность выполнения и разработки тестов

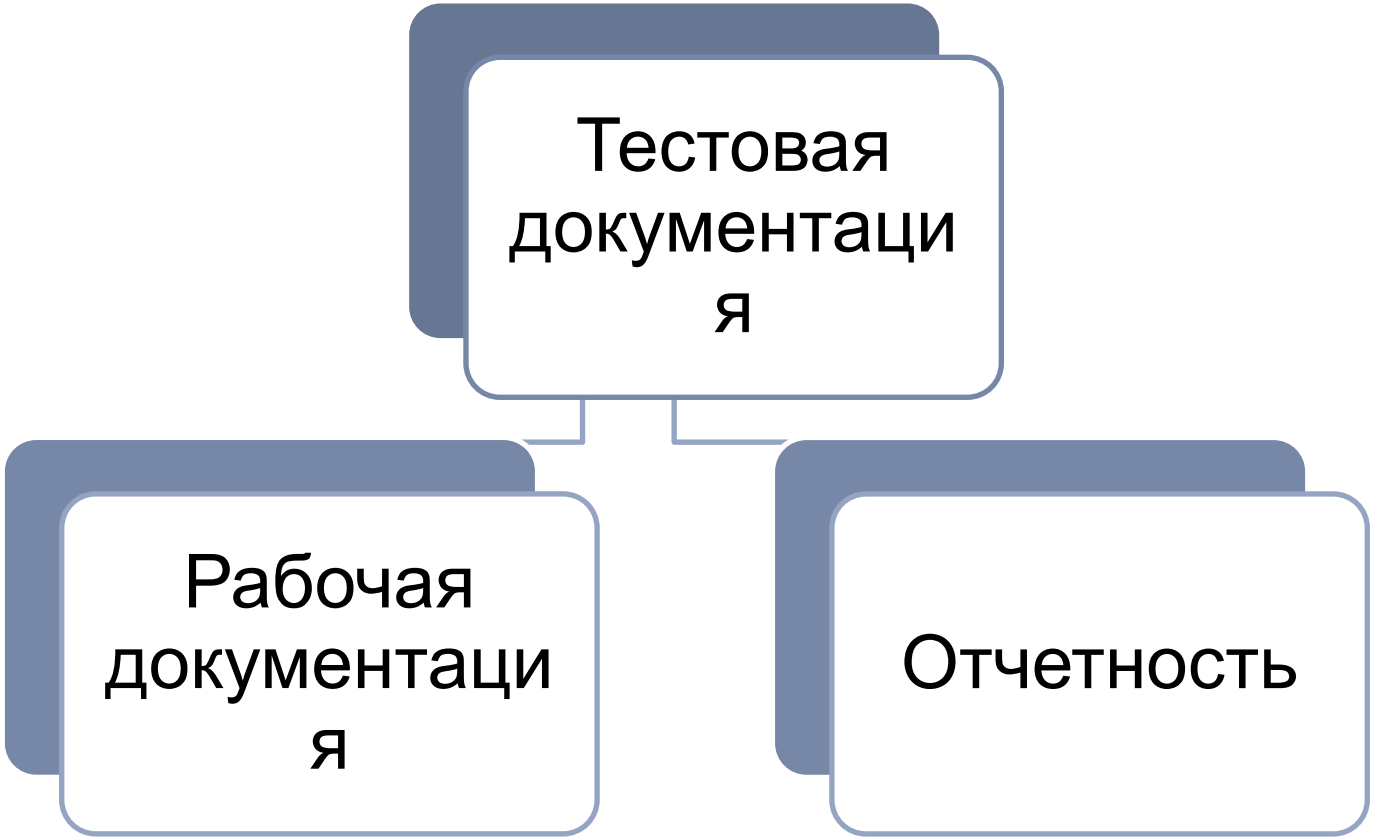
Простые позитивные

Простые негативные

Сложные позитивные

Сложные негативные

Тестовая документация



Рабочая документация

Чек лист – это список проверок, которые необходимо выполнить для тестирования приложения или какого-либо функционала приложения

Тест кейс – это набор тестовых входных данных, условий выполнения и ожидаемых результатов, разработанных с конкретной целью, такой как проверка некоторого пути выполнения программы или проверка соответствию некоторому требованию

Рабочая документация

Тип	Что описываем	Когда используем	Пример
Чеклист (Checklist)	Основные проверки. Может содержать ожидаемый результат.	Для типовой функциональности	Протестировать форму входа в почту
Тесткейсы (Test Cases)	Пошаговое описание, инструкции по тестированию. Всегда содержит ожидаемый результат.	Большие и долгосрочные проекты, требующие глубоких знаний в предметной области	Форма входа на сайт: 1. Откройте форму входа 2. Введите имя пользователя test1 3. Введите пароль test1 4. Нажмите кнопку «Войти» Ожидаемый результат: пользователь переходит на домашнюю страницу

Чек-листы

Преимущества чек-листов:

- расширение тестового покрытия за счёт отличий при прохождении
- сокращение затрат на содержание и поддержку тестов: не надо много писать
- отсутствие рутины
- возможность проходить и комбинировать тесты по-разному

Минусы чек-листов:

- начинающие тестировщики не всегда эффективно проводят тесты без подробной документации
- чек-листы невозможно использовать для обучения начинающих сотрудников
- заказчику или руководству может быть недостаточно того уровня детализации, который предлагают чек-листы

Создание чек-листов

Логически разделить приложение на отдельные, независимые друг от друга модули.

В каждом модуле перечислить доступные функции.

Для каждой функции написать списки вероятных проверок (в виде заголовков тест-кейсов, не более).

Проверять каждый пункт последовательно.

Страница "Настройки"		
GUI		Minor
Кнопка "О программе", по нажатию отображается окно "О программе" с информацией		OK
Кнопка "Обратная связь", по нажатию отображается окно "Обратная связь"		OK
Кнопка "Управление каналами", по нажатию отображается окно "Управление каналами"		OK
О программе		
GUI		OK
Проверка информации о программе, по нажатию кнопки "О программе"		OK
Кнопка "Назад", при нажатии отображается страница "Настройки"		OK
Обратная связь		
GUI		Minor
Кнопка "◄", при нажатии отображается страница "Настройки"		OK
Кнопка "Сообщить о неисправности", при нажатии отображается форма отправки обратной связи		OK
	Форма отправки обратной связи	
	Поле "Имя", доступное для редактирования	OK
	Поле "Эл. Адрес", доступное для редактирования	OK
	Текстовое поле "Описание неисправности", доступное для редактирования	OK
	Кнопка "Сообщить о неисправности", при нажатии отображается подтверждающее сообщение	OK
	Подтверждающее сообщение, отображается после заполнения всех полей и нажатия кнопки "Сообщить о неисправности"	Minor
	После нажатия кнопки "Сообщить о неисправности", на указанный эл.адрес высылается подтверждающий e-mail	OK

Зачем нужны тест-кейсы?

- Тест-кейсы дают нам структурируемый системный подход, что снижает вероятность пропуска ошибки
- Тест-кейсы – хороший способ хранения части проектной информации
- При написании тест-кейсов тестируется проектная документация
- Наличие тест-кейсов ускоряет регрессионное тестирование
- Тест-кейсы – быстрый способ ввести в курс дела новичка
- Имея тест-кейсы мы можем быстро вспомнить, что мы делали некоторое время назад
- Тест-кейсы позволяют легко отслеживать прогресс покрытого тестами функционала

При документировании тест-кейса необходимо указать:

- Идентификатор теста (id)
- Связанное с тестом требование (related requirement)
- Краткое заглавие теста (title)
- Модуль и подмодуль приложения, к которому относится тест
- Приоритет теста (priority: smoke, critical path, extended)
- Исходные данные, необходимые для теста, предусловие (initial data, precondition)
- Шаги для выполнения тестов (steps)
- Ожидаемый результат (expected result)
- Прошел тест или нет (passed or failed)
- Связанный с тестом баг (related bug)

UG_U 1.12	A	R9	Галер ея	Загрузка файла	**Галерея, загрузка файла, имя со спецсимволами** Приготовления: создать непустой файл с именем #$%^&.jpg 1. Нажать кноп картинку» 2. Нажать кноп 3. Выбрать из с приготовленны 4. Нажать кноп 5. Нажать кноп в гал ерею»	1. Появля ется окно загрузки картинки 2. Появля ется диалоговое окно бра зера выбора файла дл я загрузки 3. Имя выбранного файла я ляется в поле «Файл» 4. Д алоговое окно файла ывается, в поле «Файл» я ляется полное имя па 5. Вы бранный файл появляется в списке файлов галереи

Тест-кейсы могут быть

- Специфичными или общими
- Простыми или сложными
- Независимыми или связанными друг с другом
- Позитивными или негативными

Сравним 2 тест-кейса. Какой из них лучше?

1.В поле A ввести 10 2.В поле B ввести 15 3.Нажать кнопку «Сложить» 4.Проверить значение в поле C	4. Значение в поле C равно 25

1.Проверить, что программа суммирует два числа корректно	4. Суммирует корректно

Оба тест кейса – плохие

- Когда все детали прописаны до мелочей, при повторных выполнениях теста всегда будут выполняться строго одни и те же действия, что снижает вероятность обнаружить ошибку.
- Слишком общий тест-кейс сложно выполнять по многим объективным и субъективным причинам, а потому он вполне может остаться невыполненным.
- Однако интеграционные тесты, как правило, бывают более общими, чем иные. Это связано со спецификой интеграционного тестирования.
- Если в тесте прописано много мелких деталей, возрастает время его создания и поддержки.
- Однако недостаток деталей может усложнить работу новичка.

Хороший тест-кейс

Сложение A и B 1. В поле A ввести корректное целое число 2. В поле B ввести корректное целое число 3. Нажать кнопку «Сложить» 4. Проверить значение поля C 5. Повторить шаги 1-4 для значений: 0, максимального и минимального допустимого значений, 1.5, символов и спецсимволов	4. Значение поля C равно сумме A и B

Потому что:

- Здесь мы не привязаны к конкретным значениям.
- Мы знаем, как проверить результат.
- Мы сокращаем время написания и поддержки теста ссылкой на шаги 1-4.
- Мы перечислили значения, представляющие для нас особый интерес.

Хороший тест-кейс удовлетворяет следующим условиям

- Обладает высокой вероятностью обнаружения ошибок
- Исследует соответствующую область приложения
- Не выполняет ненужных действий
- Является не слишком простым, но и не слишком сложным
- Не является избыточным по отношению к другим тестам
- Делает обнаруженную ошибку очевидной
- Позволяет легко диагностировать ошибку

Простой и сложный тест кейс:

Где в ниже перечисленном простые тест-кейсы, а где – сложные?

Набор 1:

1. Откройте файл «1.txt». Файл открыт.
2. Введите слово «Дом». Появляется слово «Дом.
3. Сохраните файл. Кнопка «Сохранить» становится неактивной.

Набор 2:

1. В документе размером более 100 Мб создайте таблицу 100х100, в ячейку 50х50 вставьте картинку размером 30 Мб, применив к ней функцию «Авторасположение». Проверьте результат.

Простые тесты оперируют за раз одним объектом.

Преимущества простых и сложных тест кейсов

Каковы преимущества простых тест-кейсов?

- Их легко выполнять.
- Они понятны новичкам.
- Они упрощают диагностику ошибки.
- Они делают наличие ошибки очевидным.

Каковы преимущества сложных тест-кейсов?

- Больше шансов что-то сломать.
- Пользователи, как правило, используют сложные сценарии.
- Программисты сами редко проверяют такие варианты.

Следует постепенно повышать сложность тестов.

Советы для написания тест кейсов

Одним из важных правил оформления теста является язык написания.

Рекомендации:

- Используйте активный залог: («open», «paste», «click»). В русском языке используйте безличную форму: «открыть» (вместо «откройте»).
- Описывайте поведение системы: «появляется окно…», «приложение закрывается».
- Используйте простой технический стиль.
- ОБЯЗАТЕЛЬНО указывайте ТОЧНЫЕ названия всех элементов приложения.
- Не объясняйте базовые понятия работы с ОС.

Набор тестов (тестовый сценарий, Test suit)

Test suit - набор тестов (тест-кейсов), собранных в последовательность для достижения некоторой цели

Рекомендации по написанию тестовых сценариев:

- Пишите сценарий для отдельной части приложения
- Пишите отдельно сценарий для Smoke и Critical Path тестов
- Постепенно повышайте сложность тестов
- Организуйте сценарий логично
- Используйте один тест для одной проверки
- Помните, что заголовки тестов ограждают их суть
- Помните о необходимых приготовлениях к тесту
- Не повторяйте в один и тех же тестах одинаковые шаги
- Старайтесь избегать похожих тестов

Шаги разработки тест кейсов

1. Начинайте как можно раньше, ещё до выхода первого билда.

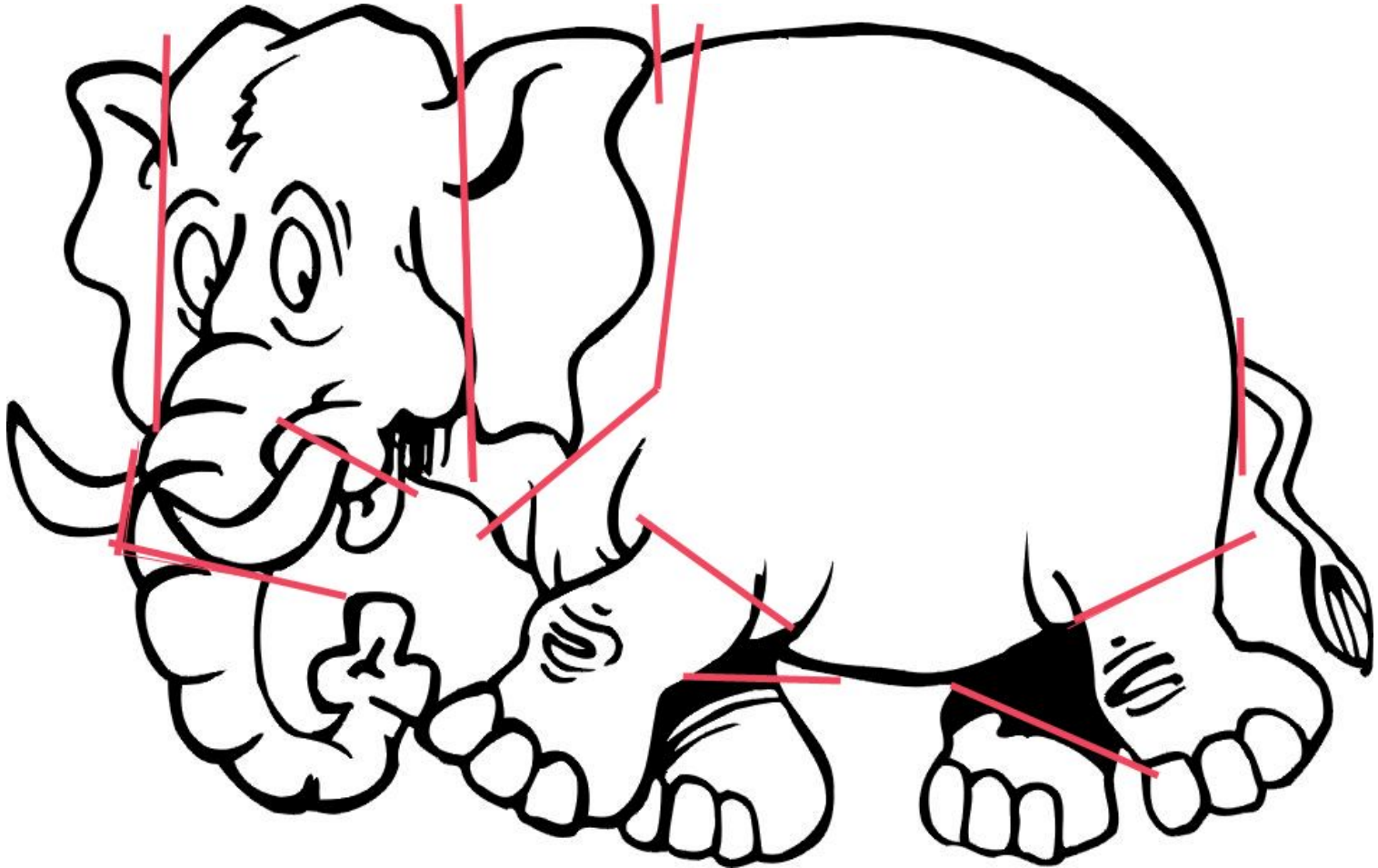
Какая информация у нас есть в это время? А какой нет?

- У нас нет работающего приложения.
- Но у нас есть документация и представители заказчика.

Смело задавайте вопросы. Помните, что на этой стадии развития проекта исправить ошибку легко и просто, а потом это будет сложно и дорого.

Шаги разработки тест кейсов

2. Разбивайте приложение на отдельные модули



Шаги разработки тест кейсов

3. Для каждой области/модуля сначала пишите чек-лист.

- Так проще проверить, всё ли нужное предусмотрено, нет ли чего лишнего.
- Удобно реорганизовывать наборы тестов.
- Легко увидеть, где можно использовать copy-paste.
- Так можно разделять интеллектуальную и рутинную работу.

Внимание!

- Просто скопированное требование – ЭТО НЕ ТЕСТ!
- Если пишете в Excel/Word – начинайте каждый новый тест в новой строке таблицы.
- Если что-то непонятно – СРАЗУ ЖЕ записывайте вопрос.

Шаги разработки тест кейсов

4. Пишите вопросы, уточняйте детали, добавляйте форматирование, используйте copy-paste.

Открытие PDF-файла 1. Нажать кнопку "Открыть". 2. Выбрать файл из диалогового окна. 3. Нажать кнопку "OK" 4. Проверить отображение открытого файла.	1. Появляется диалог выбора файлов. **Какой каталог должен открываться по умолчанию?** 2. Выбранный файл помечается выделением. 3. Диалог выбора файлов закрывается. 4. Файл отображается в рабочей области приложения.
Открытие DJV-файла 1. Нажать кнопку "Открыть". 2. Выбрать файл из диалогового окна. 3. Нажать кнопку "OK" 4. Проверить отображение открытого файла.	1. Появляется диалог выбора файлов. **Какой каталог должен открываться по умолчанию?** 2. Выбранный файл помечается выделением. 3. Диалог выбора файлов закрывается. 4. Файл отображается в рабочей области приложения.

Шаги разработки тест кейсов

5. Получите рецензию коллег-тестировщиков, разработчиков, заказчиков.

Так вы можете получить ответы на вопросы:

- Пропущено ли что-то?
- Есть ли избыточные тесты?
- Легко ли ваши тесты понять?
- Этого ли ожидает заказчик?
- Есть ли в тестах ошибки?

Кроме этого:

- Вы получаете ещё одну точку зрения на ситуацию.
- Коллеги могут заметить те ошибки, которые вы пропустили.
- У коллег может оказаться та информация, которой не было у вас.
- У разработчиков может быть своё мнение по поводу той или иной функциональности.
- Рецензирование (перепросмотр) хорошо стимулирует повышение качества разработки тестов.

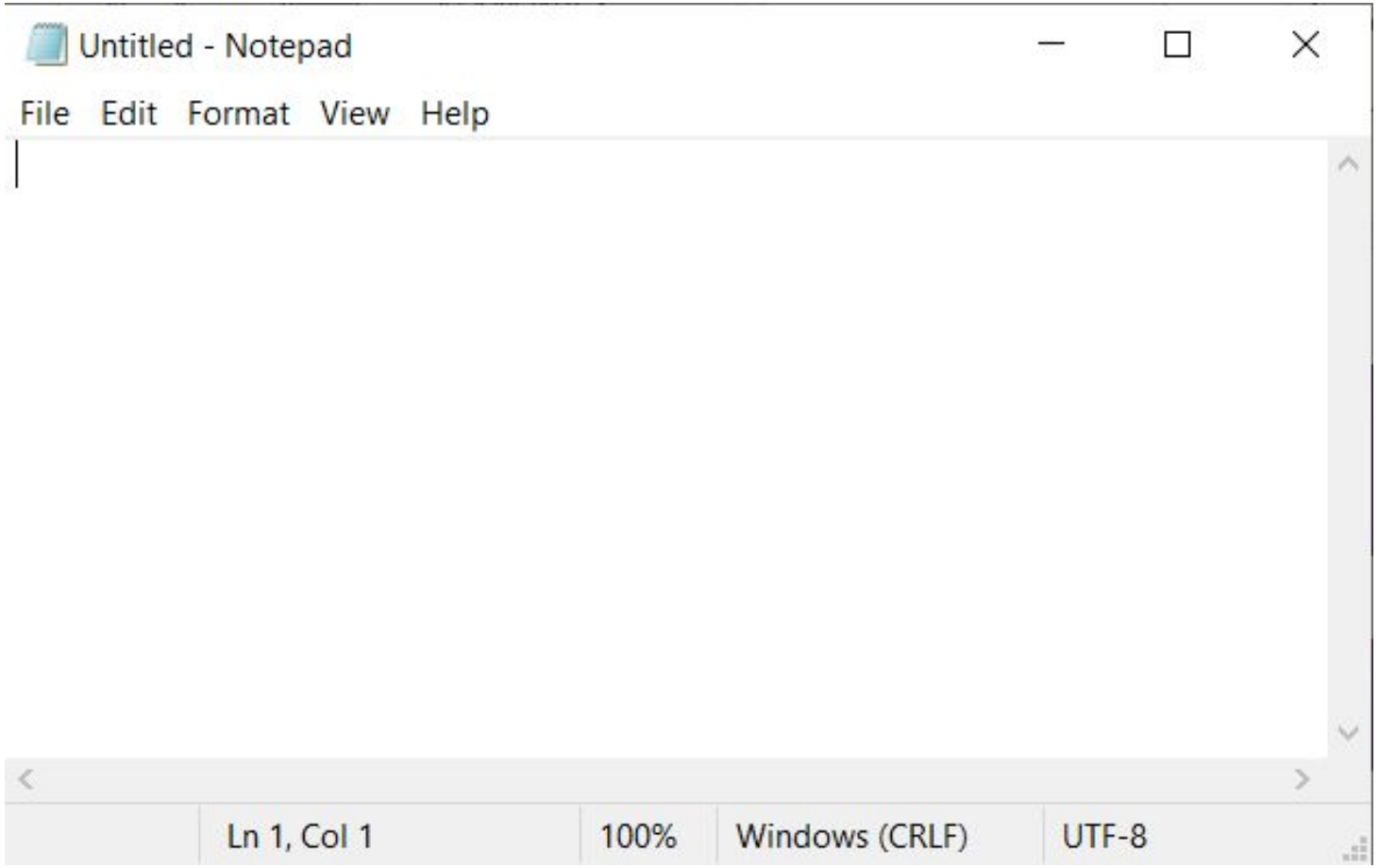
Шаги разработки тест кейсов

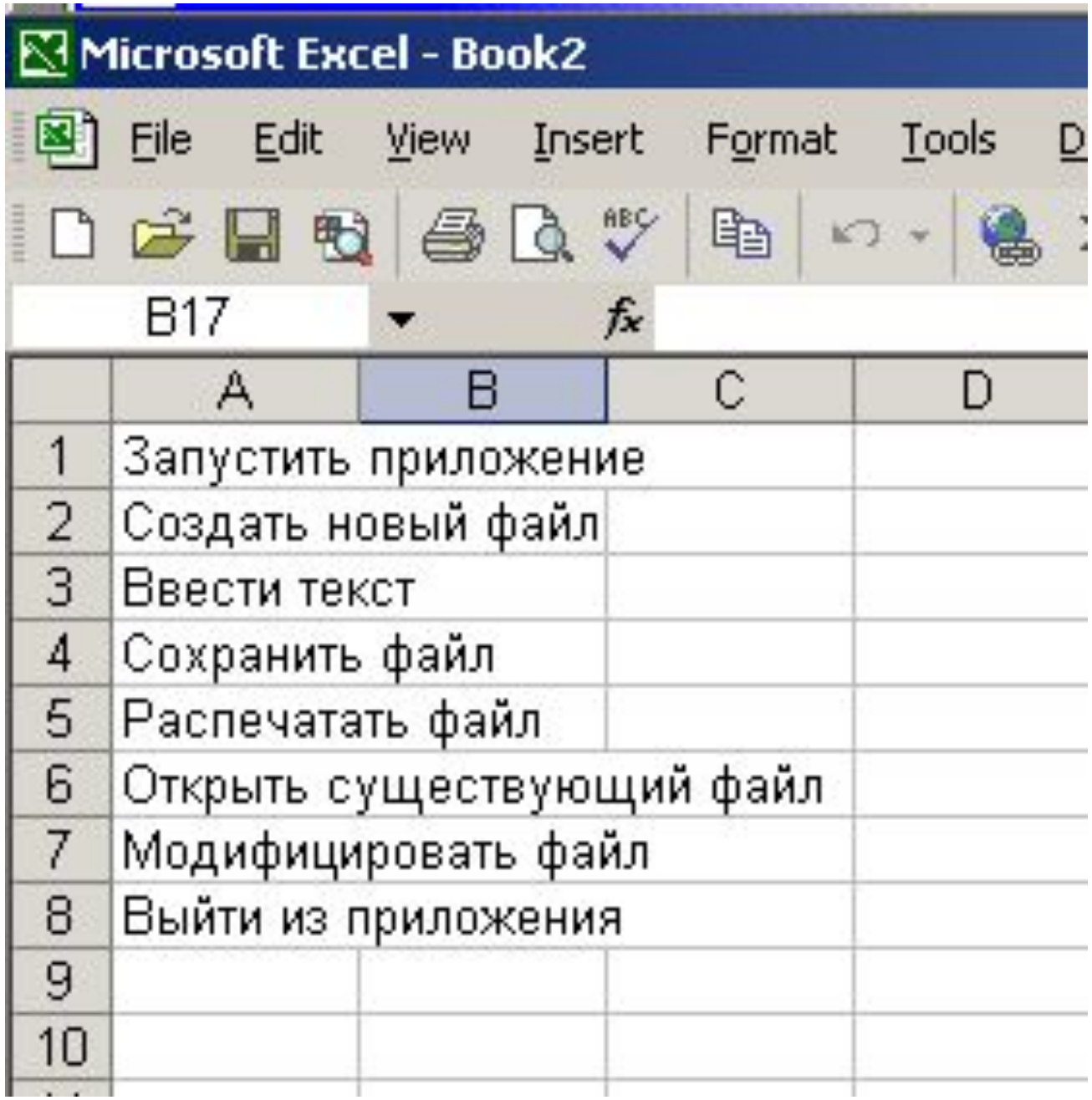
6. Обновляйте тесты, как только обнаружили ошибку или изменилась функциональность.

- Мелкие изменения вносите сразу же, как в этом возникла необходимость.
- Большие изменения можно вносить в те моменты, когда нагрузка на команду тестировщиков снижается, или когда просто появилось свободное время.

Пример разработки тест-кейсов

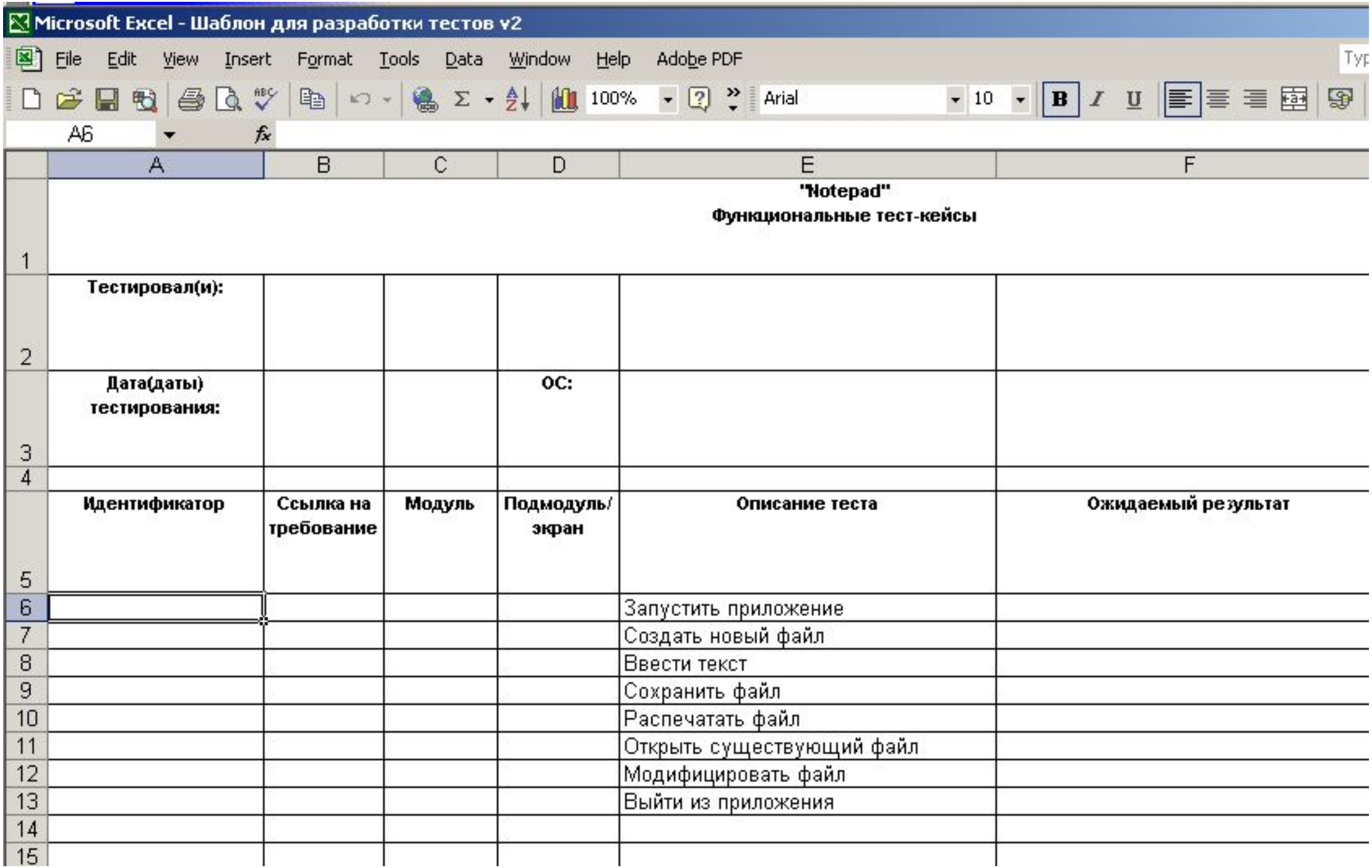
1. Что такое Notepad?
2. Какие функции для него наиболее важны?



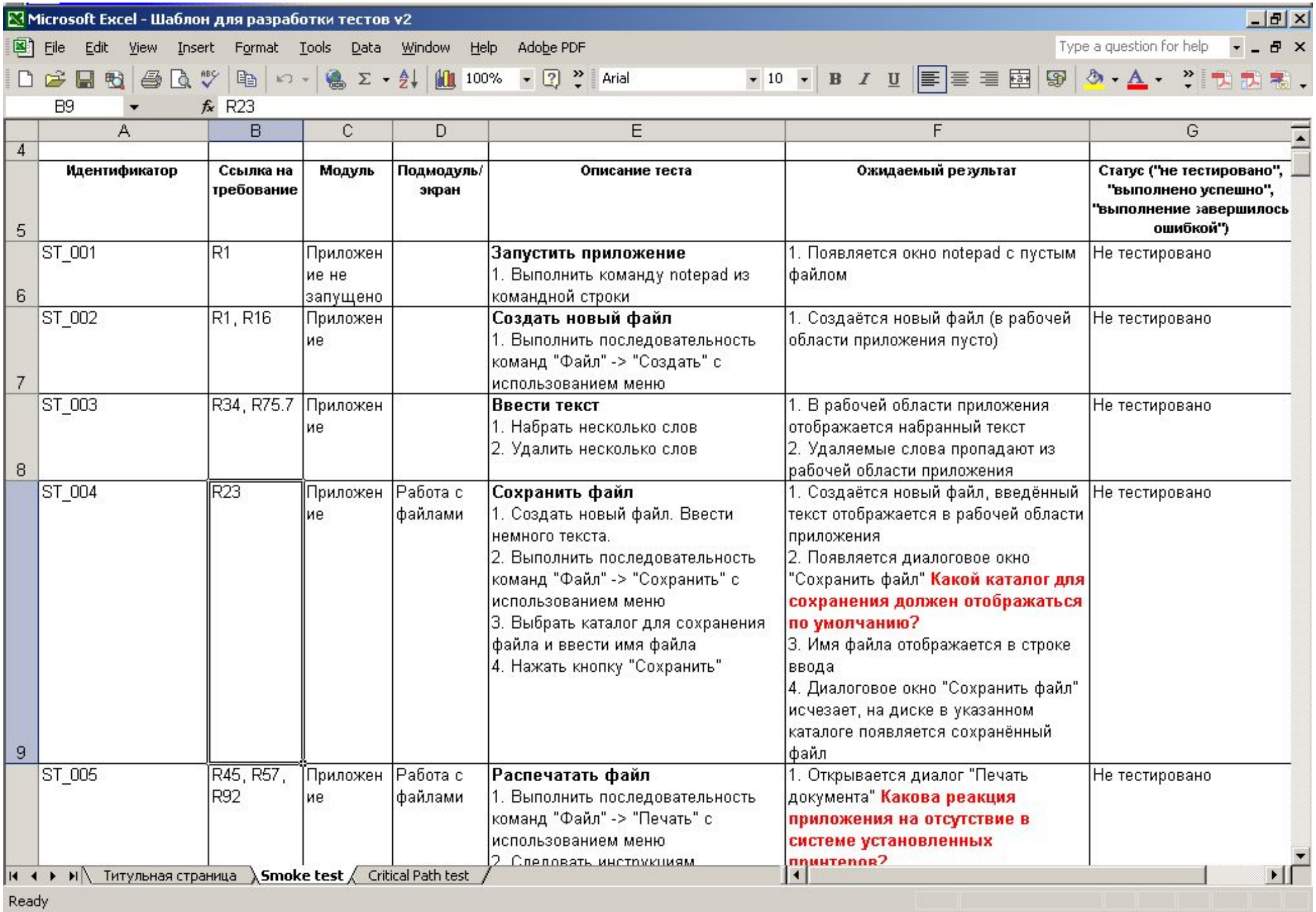


Пример разработки тест-кейсов:

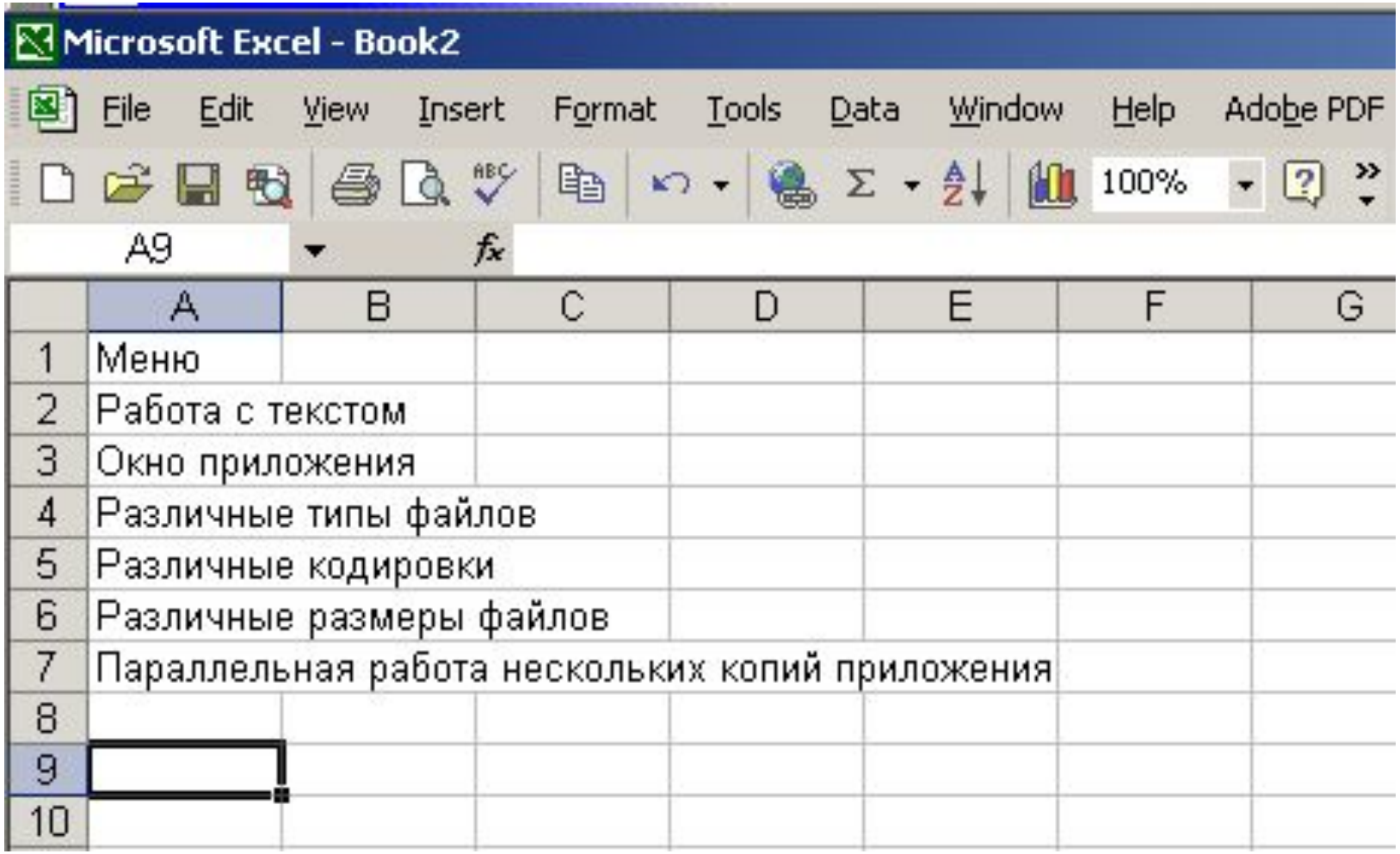
ЧЕК ЛИСТ ДЛЯ SMOKE TEST



	A	B	C	D	E	F
1					**"Notepad"** **Функциональные тест-кейсы**	
2	**Тестировал(и):**					
3	**Дата(даты) тестирования:**			**ОС:**		
4						
5	**Идентификатор**	**Ссылка на требование**	**Модуль**	**Подмодуль/ экран**	**Описание теста**	**Ожидаемый результат**
6					Запустить приложение	
7					Создать новый файл	
8					Ввести текст	
9					Сохранить файл	
10					Распечатать файл	
11					Открыть существующий файл	
12					Модифицировать файл	
13					Выйти из приложения	
14						
15						

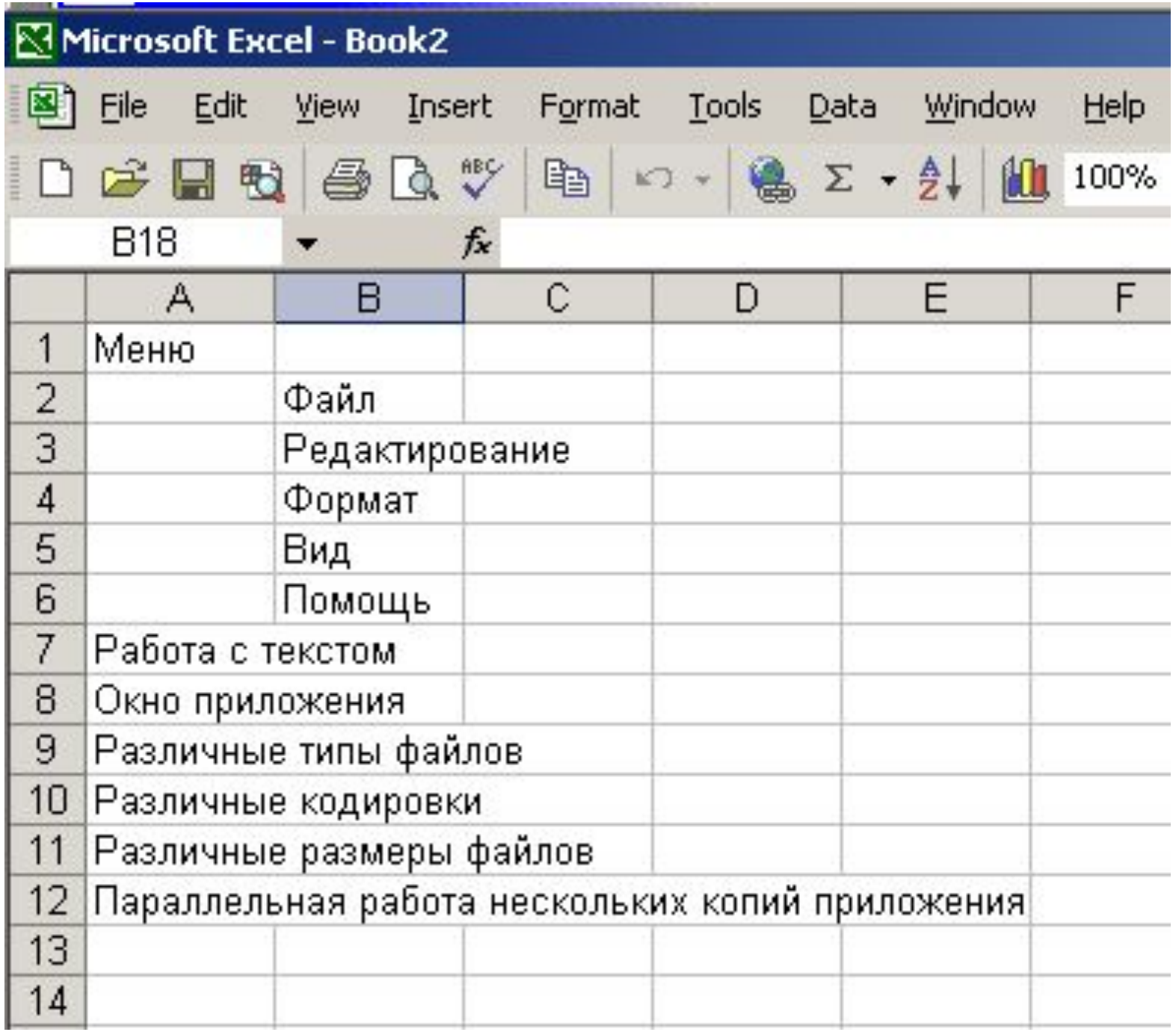


	A	B	C	D	E	F	G
4							
5	**Идентификатор**	**Ссылка на требование**	**Модуль**	**Подмодуль/ экран**	**Описание теста**	**Ожидаемый результат**	**Статус ("не тестировано", "выполнено успешно", "выполнение завершилось ошибкой")**
6	ST_001	R1	Приложение не запущено		**Запустить приложение** 1. Выполнить команду notepad из командной строки	1. Появляется окно notepad с пустым файлом	Не тестировано
7	ST_002	R1, R16	Приложение		**Создать новый файл** 1. Выполнить последовательность команд "Файл" -> "Создать" с использованием меню	1. Создаётся новый файл (в рабочей области приложения пусто)	Не тестировано
8	ST_003	R34, R75.7	Приложение		**Ввести текст** 1. Набрать несколько слов 2. Удалить несколько слов	1. В рабочей области приложения отображается набранный текст 2. Удаляемые слова пропадают из рабочей области приложения	Не тестировано
9	ST_004	R23	Приложение	Работа с файлами	**Сохранить файл** 1. Создать новый файл. Ввести немного текста. 2. Выполнить последовательность команд "Файл" -> "Сохранить" с использованием меню 3. Выбрать каталог для сохранения файла и ввести имя файла 4. Нажать кнопку "Сохранить"	1. Создаётся новый файл, введённый текст отображается в рабочей области приложения 2. Появляется диалоговое окно "Сохранить файл" **Какой каталог для сохранения должен отображаться по умолчанию?** 3. Имя файла отображается в строке ввода 4. Диалоговое окно "Сохранить файл" исчезает, на диске в указанном каталоге появляется сохранённый файл	Не тестировано
	ST_005	R45, R57, R92	Приложение	Работа с файлами	**Распечатать файл** 1. Выполнить последовательность команд "Файл" -> "Печать" с использованием меню 2. Следовать инструкциям	1. Открывается диалог "Печать документа" **Какова реакция приложения на отсутствие в системе установленных принтеров?**	Не тестировано



Пример разработки тест-кейсов:

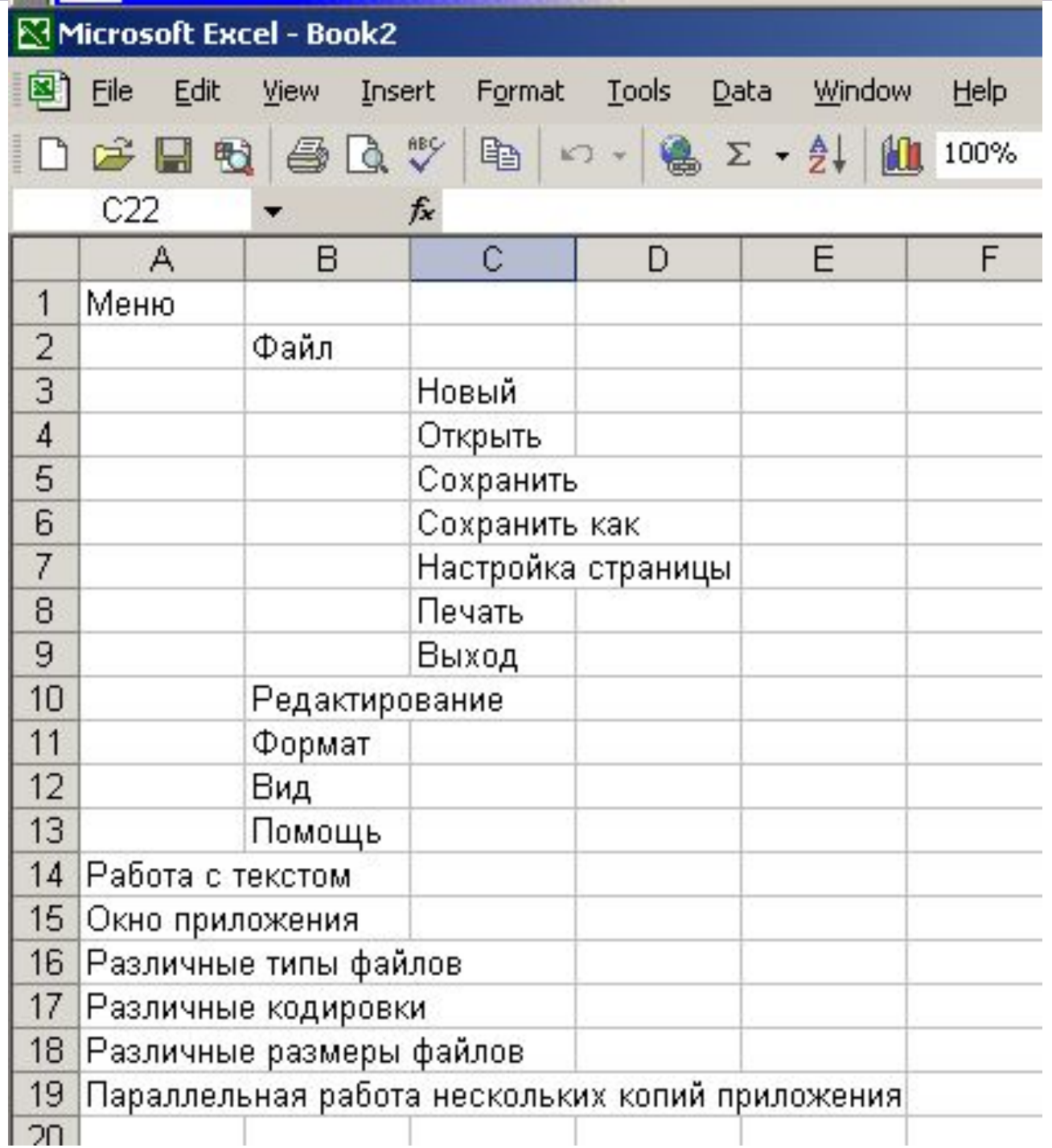
ЧЕК ЛИСТ ДЛЯ ТЕСТА КРИТИЧЕСКОГО ПУТИ



	A	B	C	D	E	F
1	Меню					
2		Файл				
3		Редактирование				
4		Формат				
5		Вид				
6		Помощь				
7	Работа с текстом					
8	Окно приложения					
9	Различные типы файлов					
10	Различные кодировки					
11	Различные размеры файлов					
12	Параллельная работа нескольких копий приложения					
13						
14						

Пример разработки тест-кейсов:

ДЕТАЛИЗИРУЕМ ЧЕК ЛИСТ



	A	B	C	D	E	F
1	Меню					
2		Файл				
3			Новый			
4			Открыть			
5			Сохранить			
6			Сохранить как			
7			Настройка страницы			
8			Печать			
9			Выход			
10		Редактирование				
11		Формат				
12		Вид				
13		Помощь				
14	Работа с текстом					
15	Окно приложения					
16	Различные типы файлов					
17	Различные кодировки					
18	Различные размеры файлов					
19	Параллельная работа нескольких копий приложения					
20						

Пример разработки тест-кейсов:

ПРОДОЛЖАЕМ ДЕТАЛИЗАЦИЮ ДО ТЕХ ПОР, ПОКА НЕ ПОЛУЧИМ ЛОГИЧНЫЙ И ДОСТАТОЧНЫЙ НАБОР ТЕСТОВ. ПОСЛЕ ЭТОГО ПЕРЕНОСИМ ЕГО В ШАБЛОН И РАБОТАЕМ АНАЛОГИЧНО ТОМУ, КАК МЫ ДЕЛАЛИ ЭТО ПРИ РАЗРАБОТКЕ SMOKE TEST.

Задание

Написать тест кейсы для меню Файл программы Notepad