Тестирование программного обеспечения

Типы тестирования: функциональное тестирование

- 90% рабочего времени занимает проверка функциональных требований: логики и бизнес-правил приложения или системы.
- Как правило, полноценное системное/функциональное тестирование является самым трудоёмким процессом
- Обращайте внимание:
- На невозможность полного покрытия – всегда надо выбирать
- На необходимость постоянно отслеживать приоритетность требований от версии к версии: требования меняются, приоритеты тоже.

Типы тестирования: бизнес циклы

- Бизнес-циклы: служат для проверки корректности работы алгоритмов и механизмов, автоматизирующих не столько пользовательские операции, сколько естественную цикличность любого бизнеса, завершающуюся отчётами, архивированием данных, выполнением нетипичных для системы операций.
- Закрытие месяца, закрытие года, получение очередной крупнооптовой партии товаров, расчёт пени, реструктуризации долгов и т.п.

- Когда покрыты все:
  - строки кода программы
  - ветви кода в программе
  - пути в коде
  - входные данные и все их возможные комбинации
  - последовательности комбинаций входных данных
  - …

Полное тестирование это –

Почему нельзя полностью протестировать программу

- Количество всех возможных комбинаций входных данных слишком велико, чтоб его можно было проверить полностью
- Количество всех возможных последовательностей выполнения кода программы также слишком велико, чтобы его можно было протестировать полностью
- Пользовательский интерфейс программы (включающий все возможные комбинации действий пользователя и его перемещений по программе) обычно слишком сложен для полного тестирования

Виды тестовых сценариев

- Позитивные сценарии
- Негативные сценарии
- Граничные сценарии
- Исследовательские сценарии:
- «А что должно быть если…»

Два основных вопроса в тестировании:

1. Что подать на вход?
2. Чего ожидать на выходе?

Техники тестирования. Эквивалентное разбиение

- Чтобы избежать ненужного тестирования, разбейте область входных значений на группы эквивалентных тестов.
- Два теста считаются эквивалентными если они настолько похожи, что проводить оба бессмысленно.

Рассмотрим **пример**

- Программа складывает два целых числа
- Каждое из слагаемых – не более чем целое двузначное число
- Программа запрашивает у пользователя два числа и выводит результат

Классы эквивалентности

	Классы корректных данных	Классы некорректных данных	Граничные и специальные значения
Первое слагаемое	от -99 до -10 от -9 до -1 0 от 1 до 9 от 10 до 99	> 99 < -99	0, 1, -1, 9, -9 10, -10 99, -99 100, -100
Второе слагаемое	-”-”-	-”-”-	-”-”-
Сумма	от -198 до -100 от -99 до -1 0 от 1 до 99 от 100 до 198	> 198 < -198	(-99, -99) (-49, -51) (99, 99) (49, 51)

Порядок действий

- Перечисляются все переменные (как входные, **так и выходные**)
- Для каждой переменной определяется разбиение на классы
- Строятся все возможные комбинации классов
- В качестве представителей берутся граничные, приграничные или специальные значения

Практические примеры

«**Треугольник**»

- Программа запрашивает три числа
- Определяется тип треугольника, имеющего стороны введенной длины: равносторонний, равнобедренный, разносторонний

Техники тестирования. Эквивалентное разбиение

- Корректный разносторонний треугольник
- Корректный равносторонний треугольник
- Три корректных равнобедренных треугольника (a=b, b=c, a=c)
- Одна, две или три стороны равны нулю (5 тестов)
- Одна сторона отрицательная
- «Почти» выполняется правило треугольника (три варианта a+b=c, a+c=b, b+c=a)
- Не выполняется правило треугольника (три варианта a+b<c, a+c<b, b+c<a)
- Нецелое число или не число
- Неправильное количество аргументов

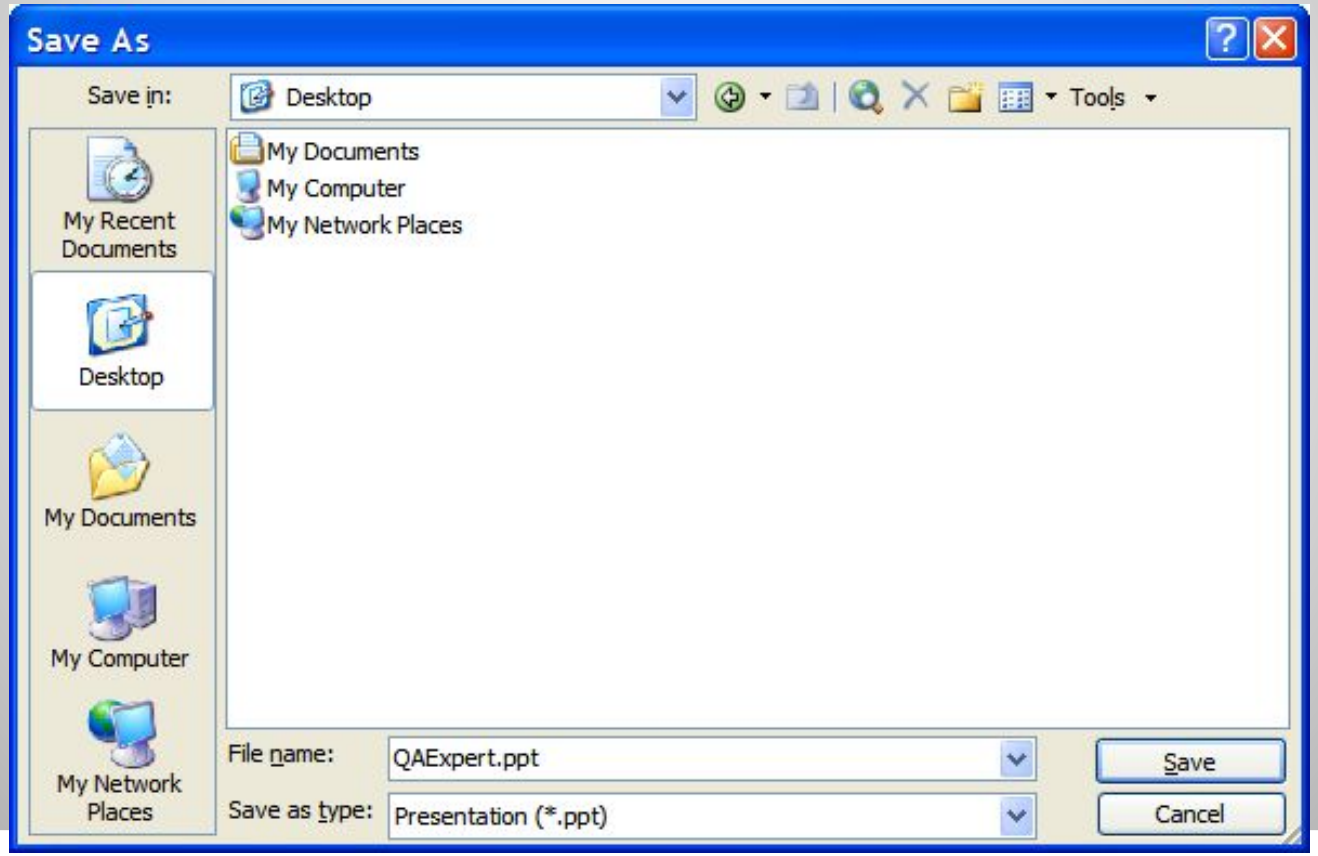
Практические примеры

Описание тестируемого функционала:

- Поле для ввода названия папки
- Кнопка «Сохранить»
- Название папки не должно превышать 64 символа

Практический пример

Диалог сохранения файла



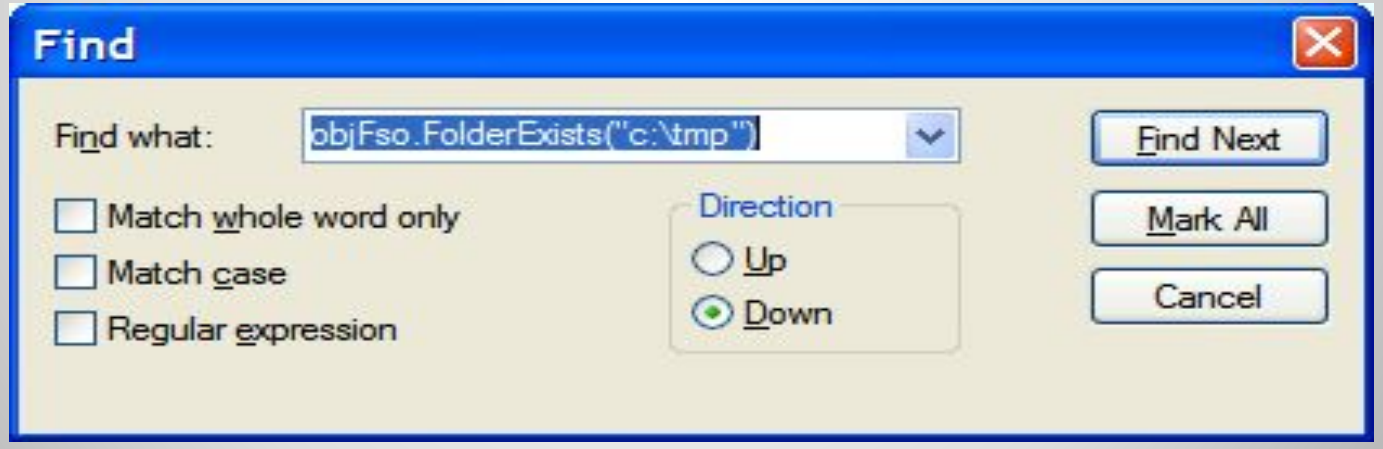
- Сначала выделяем наиболее рискованные (и важные) области – собственно сохранение , выбор нужного места, сохранение с длинным именем, с национальными символами, перезапись и т.п.
- Потом выясняем какие сценарии использования (use case)
- Выясняем классы эквивалентности
- Пишем тест-кейсы (позитивные, негативные, исследовательские)

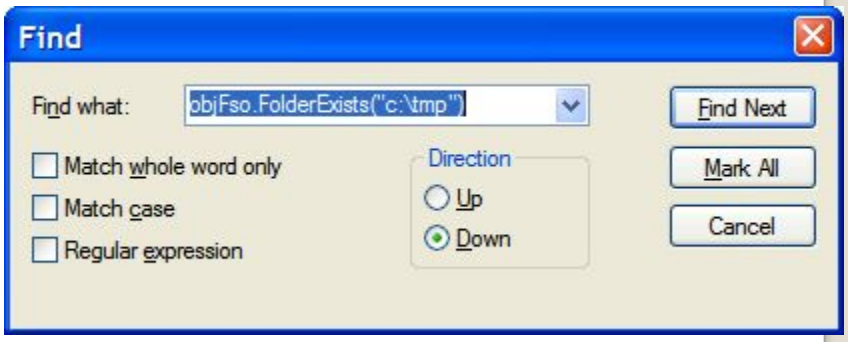
«Фиксируем шаги»

Способы снижения количества тестов

Рассмотрим пример

- Окно поиска в текстовом редакторе





- **Подсчитаем количество тестов**
- 5 переменных:
  - Find what (FW) – строка
  - Match whole words only (MW) – Boolean
  - Match case (MC) – Boolean
  - Regular expression (RE) – Boolean
  - Direction (D) – перечисляемый тип (Up, Down)
- Тестовые значения
  - FW = {‘lower’; ‘UPPER’; ‘MiXeD’}
  - MW, MC, RE = {Yes; No}
  - B = {Up; Down}
- Итого: 3 x 2 x 2 x 2 x 2 = 48 тестов

Способы снижения количества тестов

Полный перебор (все Nки)

	FW	MW	MC	RE	D
1	L	Y	Y	Y	Up
2	U	Y	Y	Y	Up
3	M	Y	Y	Y	Up
4	L	Y	Y	Y	Up
5	L	N	Y	Y	Up
…	…	…	…	…	…
47	M	N	N	N	Up
48	M	N	N	N	D

Способы снижения количества тестов

- **Выбор комбинаций**
- Для данного случая методы выбора на основе рисков и на основе сценариев малопригодны
- Оптимальнее использовать механический перебор по некоторой системе:
  - Полный перебор
  - Все пары (каждый с каждым)
  - Все значения хотя бы по разу

Способы снижения количества тестов

Все значения хотя бы по разу

	FW	MW	MC	RE	D
1	L	Y	N	Y	Up
2	U	N	Y	N	D
3	M	Y	Y	N	Up

3 теста, а не 48

Способы снижения количества тестов

Все пары (каждый с каждым)

	FW	MW	MC	RE	D
1	L	Y	N	Y	Up
2	L	N	Y	N	D
3	U	Y	Y	N	Up
4	U	N	N	Y	D
5	M	N	N	N	Up
6	M	Y	Y	Y	D

- Этот метод является «золотой серединой»
- Метод «всех пар» хорошо работает для независимых переменных
- Зачастую случайное тестирование хорошо приближается к методу «всех пар»