

# Тестирование программных средств

Сафронов Сергей,  
2009 год

# Оглавление

- Проектирование тестов
  - Методики проектирования тестов
- Эквивалентное разбиение
  - Правила выделения классов
  - Правила составления тестов
- Анализ граничных значений
  - Правила составления тестов

# Проектирование тестов

Ресурсы всегда ограничены, поэтому самым важным является следующий вопрос:

Какое подмножество всех возможных тестов имеет наибольшую вероятность обнаружения большинства ошибок?

Понятно, что худший вариант – это стохастическое (как придется), но как надо?

# Методики проектирования тестов

## Черный ящик:

- Эквивалентное разбиение
- Анализ граничных значений
- Метод функциональных диаграмм

## Стеклянный ящик:

- Покрытие строк
- Покрытие условий
- Покрытие условных операторов

# Эквивалентное разбиение

- Технология проектирования тестов, ориентированная на снижение общего числа тестов
- Основная мысль – разбить все данные для программы на классы
- Если проектировать тесты для каждого класса, а не для каждого члена класса – то общее число тестов уменьшится

# Эквивалентное разбиение (2)

## Правильный тест

- Уменьшает более чем на единицу число других тестов, которые нужно разработать для достижения приемлемого уровня тестирования
- Покрывать значительную часть других ВОЗМОЖНЫХ ТЕСТОВ

# Эквивалентное разбиение

Разработка тестов идет в два этапа:

- Выделение классов эквивалентности
- Построение тестов

<b>Входные условия</b>	<b>Правильные классы эквивалентности</b>	<b>Неправильные классы эквивалентности</b>
------------------------	--	--

# Правила выделения классов эквивалентности

- Если входное условие описывает диапазон, то выделяют один правильный класс эквивалентности и два неправильных
- Если входное условие описывает множество значений, каждое из которых трактуется особо, то определяется правильный класс эквивалентности для каждого из значений и один неправильный класс значений
- Если входное условие трактуется как «должно быть», то делается один правильный класс эквивалентности и один неправильный
- Если есть подозрение, что различные элементы класса эквивалентности могут трактоваться программой по разному, следует разбить класс на несколько подклассов



# Правила составления тестов

- Каждому классу эквивалентности назначается уникальный номер
- Проектирование новых тестов, каждый из которых покрывает как можно большее число непокрытых правильных классов эквивалентности до тех пор, пока не будут покрыты все правильные классы эквивалентности
- Проектирование тестов, каждый из которых покрывает один и только один из непокрытых неправильных классов эквивалентности пока все неправильные классы эквивалентности не будут покрыты тестами

# Практика

Программа получается на вход 3 целых значения. Эти числа являются длинами сторон треугольника. Результатом работы программы является сообщение – каким является этот треугольник: прямоугольным, равносторонним, равнобедренным, неравносторонний.

«?» : расписать классы эквивалентности

# Решение

## Корректные классы

- $(2,2,2)$  – равносторонний
- $(3,2,2)$  – равнобедренный
- $(3,4,5)$  – прямоугольный
- $(2,3,4)$  – неравносторонний

## Некорректные классы

- $(2,2,-1)$  – отрицательная сторона
- $(3,2,1)$  – вырожденный
- $(2,2,0)$  – нулевая длина
- $(5,3,1)$  – не треугольник
- $(2,3,A)$  – не число

# Практика №2

Есть поле ввода времени: формат  
ввода «ЧЧ»:«ММ»:«СС»

«?»: Расписать классы  
эквивалентности

# Решение

Входные условия	Правильные классы эквивалентности	Неправильные классы эквивалентности
Часы	$[0;23]$	$(-\infty,0)$ $[24,+\infty)$ (NaN)
Минуты	$[0;59]$	$(-\infty,0)$ $[60,+\infty)$ (NaN)
Секунды	$[0;59]$	$(-\infty,0)$ $[60,+\infty)$ (NaN)

# Анализ граничных значений

Анализ граничных значений отличается от эквивалентного разбиение в двух моментах:

- Выбор элемента в классе эквивалентности идет таким образом, чтобы проверить тестом каждую границу этого класса
- При разработке тестов рассматривается не только пространство условий, но и пространство результатов

# Правила составления тестов

1. Построить тесты для границ области и тесты с неправильными данными для случаев незначительного выхода за границы области, если входное значение описывает диапазон значений
2. Построить тесты для минимального и максимального значения условий и тесты, большие и меньшие этих значений, если входное условие удовлетворяет дискретному ряду значений
3. Использовать правило 1 для каждого выходного условия
4. Использовать правило 2 для каждого выходного условия
5. Если вход или выход программы есть упорядоченное множество, то сделать тесты на первый и последний элементы
6. Попробовать найти другие граничные значения

# Практика

- Поле ввода: возраст сотрудника.
- Допустимые значения: от 14 до 80.

«?»: список тестов?



# Решение

Входные условия	Правильные классы эквивалентности	Неправильные классы эквивалентности
Возраст	[14] [15] (16;78) [79] [80]	$(-\infty, 12]$ [0] [] [13] [81] $[82, +\infty)$ (NaN)

# Домашнее задание 😊

Программа:

- Входные данные:

- Число
- Месяц
- День недели

- Выходные данные:

- Список лет от 2000 до 2100 года, когда эта дата попадает на этот день недели

«?»: расписать классы эквивалентности