Задание 2 Отладка отдельных модулей программного проекта

Теоретическая часть

Рассмотрим два основных подхода к проектированию тестов.

Первый подход ориентируется только на стратегию тестирования, называемую стратегией "черного ящика", тестированием с управлением по данным или тестированием с управлением по входу-выходу. При использовании этой стратегии программа рассматривается как черный ящик. Тестовые данные используются только в соответствии со спецификацией программы (т. е. без учета знаний о ее внутренней структуре). Недостижимый идеал сторонника первого подхода — проверить все возможные комбинации и значения на входе. Обычно их слишком много даже для простейших алгоритмов. Так, для программы расчета среднего арифметического четырех чисел надо готовить 10^7 тестовых данных.

При первом подходе обнаружение всех ошибок в программе является критерием исчерпывающего входного тестирования. Последнее может быть достигнуто, если в качестве тестовых наборов использовать все возможные наборы входных данных. Следовательно, приходим к выводу, что для исчерпывающего тестирования программы требуется бесконечное число тестов, а значит, построение исчерпывающего входного теста невозможно. Это подтверждается двумя аргументами: во-первых, нельзя создать тест, гарантирующий отсутствие ошибок; во-вторых, разработка таких тестов противоречит экономическим требованиям. Поскольку исчерпывающее тестирование исключается, нашей целью должна стать максимизация результативности капиталовложений в тестирование (максимизация числа ошибок, обнаруживаемых одним тестом). Для этого необходимо рассматривать внутреннюю структуру программы и делать некоторые разумные, но, конечно, не обладающие полной гарантией достоверности предположения.

Второй подход использует стратегию "белого ящика", или стратегию тестирования, управляемую логикой программы, которая позволяет исследовать внутреннюю структуру программы. В этом случае тестировщик получает тестовые данные путем анализа только логики программы; стремится, чтобы каждая команда была выполнена хотя бы один раз. При достаточной квалификации добивается, чтобы каждая команда условного перехода выполнялась бы в каждом направлении хотя бы один раз. Цикл должен выполняться один раз, ни разу, максимальное число раз. Цель тестирования всех путей извне также недостижима. В программе из двух последовательных циклов внутри каждого из них включено ветвление на десять путей, имеется 10^{18} путей расчета. Причем выполнение всех путей расчета не гарантирует выполнения всех спецификаций.

Сравним способ построения тестов при данной стратегии с исчерпывающим входным тестированием стратегии "черного ящика". Неверно предположение, что достаточно построить такой набор тестов, в котором каждый оператор исполняется хотя бы один раз. Исчерпывающему входному тестированию может быть поставлено в соответствие исчерпывающее тестирование маршрутов. Подразумевается, что программа проверена полностью, если с помощью тестов удается осуществить выполнение этой программы по всем возможным маршрутам ее потока (графа) передач управления.

Последнее утверждение имеет два слабых пункта: во-первых, число не повторяющих друг друга маршрутов — астрономическое; во-вторых, даже если каждый маршрут может быть проверен, сама программа может содержать ошибки (например, некоторые маршруты пропущены).

Свойство пути выполняться правильно для одних данных и неправильно для других — называемое чувствительностью к данным, наиболее часто проявляется за счет численных погрешностей и погрешностей усечения методов. Тестирование каждого из всех маршрутов одним тестом не гарантирует выявление чувствительности к данным.

В результате всех изложенных выше замечаний отметим, что ни исчерпывающее входное тестирование, ни исчерпывающее тестирование маршрутов не могут стать полезными стратегиями, потому что оба они нереализуемы. Поэтому реальным путем, который позволит создать хорошую, но, конечно, не абсолютную стратегию, является сочетание тестирования программы несколькими методами.

Рассмотрим пример тестирования оператора:

if A and B then…

при использовании разных критериев полноты тестирования.

При критерии покрытия условий требовались бы два теста: ***A = true***, ***B = false*** и ***A = false***, ***B = true***. Но в этом случае не выполняется then-предложение оператора if.

Существует еще один критерий, названный покрытием решений/условий. Он требует такого достаточного набора тестов, чтобы все возможные результаты каждого условия в решении выполнялись, по крайней мере, один раз; все результаты каждого решения выполнялись тоже один раз и каждой точке входа передавалось управление, по крайней мере, один раз.

Недостатком критерия покрытия решений/условий является невозможность его применения для выполнения всех результатов всех условий. Часто подобное выполнение имеет место вследствие того, что определенные условия скрыты другими условиями. Например, если условие AND есть ложь, то никакое из последующих условий в выражении не будет выполнено. Аналогично, если условие OR есть истина, то никакое из последующих условий не будет выполнено. Следовательно, критерии покрытия условий и покрытия решений/условий недостаточно чувствительны к ошибкам в логических выражениях.

Критерием, который решает эти и некоторые другие проблемы, является комбинаторное покрытие условий. Он требует создания такого числа тестов, чтобы все возможные комбинации результатов условия в каждом решении и все точки входа выполнялись, по крайней мере, один раз.

В случае циклов число тестов для удовлетворения критерию комбинаторного покрытия условий обычно больше, чем число путей.

Легко видеть, что набор тестов, удовлетворяющий критерию комбинаторного покрытия условий, удовлетворяет также и критериям покрытия решений, покрытия условий и покрытия решений/условий.

Таким образом, для программ, содержащих только одно условие на каждое решение, минимальным является критерий, набор тестов которого вызывает выполнение всех результатов каждого решения, по крайней мере, один раз; передает управление каждой точке входа (например, оператор CASE).

Для программ, содержащих решения, каждое из которых имеет более одного условия, минимальный критерий состоит из набора тестов, вызывающих все возможные комбинации результатов условий в каждом решении и передающих управление каждой точке входа программы, по крайней мере, один раз.

Деление алгоритма на типовые стандартные структуры позволяет минимизировать усилия программиста, затрачиваемые им на тестирование. Запрет на вложенные структуры как раз и объясняется излишними затратами на тестирование. Использование цепочки простых альтернатив с одним действием или структуры ВЫБОР вместо вложенных простых АЛЬТЕРНАТИВ значительно сокращает число тестов!

Задание.

1. Оформить внешнюю спецификацию.
2. Составить в виде блок-схемы алгоритм решения задачи.
3. Создать программу решения задачи на любом алгоритмическом языке программирования.
4. Составить набор тестов и провести тестирование созданной программы с помощью методов «белого ящика» (покрытия операторов, покрытия решений, покрытия условий, комбинаторного покрытия условий).
5. Оформить отчет по лабораторной работе.

Отчет по лабораторной работе должен включать:

1. Внешнюю спецификацию.
2. Алгоритм решения задачи.
3. Текст программы на языке программирования.
4. Набор тестов для отладки программы, соответствующий конкретным методам «белого ящика».

Задача. «Нахождение характерных точек функции». Составить алгоритм и написать программу последовательного вычисления значений заданной функции Y(X) до тех пор, пока не будет пройдена некоторая характерная точка графика функции. Значения аргумента X составляют возрастающую последовательность с шагом h. Начальное значение X_0 и шаг изменения аргумента h задаются пользователем.

Варианты индивидуальных заданий.

№ п/п	Вид функции *Y(X)*, характерные точки	№ п/п	Вид функции *Y(X)*, характерные точки
1	Локальный минимум функции $x^2+0.5-\sin(3*x)$	16	Точка (точки), в которой функция $\left\|\frac{2x^3-3}{x^2+1}\right\|$ равна 10.
2	Точка (точки), для которой $\sqrt{x^2+5}-1=5$.	17	Локальный минимум функции $3(x+4)^2-\sin(x+15)$
3	Пересечение графиков функций $\sqrt{x^2+1}(3+x)$ и $\frac{5}{x^2-7}$.	18	Точка (точки), в которой функция $\frac{2x^2}{\sin(x-1)}-1$ равна -500.
4	Все нули функции $x^2-\sin(3x)$.	19	Локальный максимум функции $\frac{2x+15}{3+x^2}$
5	Локальный минимум функции $\frac{5}{3x-x^2-3}$	20	Пересечение графиков функций $(x^2+1)\sin(3+x)$ и $\frac{x+5}{x^2+1}$.
6	Точка (точки), в которой функция x^2-e^x равна -10.	21	Локальный минимум функции $x^2-3x+15$
7	Пересечение графиков функций $100\sin(1+x)$ и x^3+5.	22	Все нули функции $\sqrt{x^2+0.5}-\sin(3x)$.
8	Локальный минимум функции $\sqrt{e^x}-x$	23	Локальный максимум функции $200x-e^x$
9	Все нули функции $\frac{x^2-15}{\ln x^2+3x-7}$.	24	Все нули функции $x-e^x\cos(x)$.
10	Локальный максимум функции $\frac{\sqrt{2x^2+1}}{3+x^2}$	25	Точка (точки), в которой функция $\frac{x^2+5}{x^2+7}$ равна 15.
11	Пересечение графиков функций $10\cos(x)$ и $x^3/3+5$.	26	Все нули функции $\frac{x^2-15}{2x^2+3x-7}$.
12	Локальный максимум функции $-(x-2)^2+\cos(x)$	27	Пересечение графиков функций $3x^2-15$ и $10\cos(x)+7$.
13	Точка (точки), в которой функция $3x^2-\frac{12x-5}{x^2+1}$ равна 5.	28	Локальный минимум функции $-100x+e^x$
14	Все нули функции $\frac{x+5}{x^2-7}$.	29	Локальный максимум функции $-\frac{x}{x^2+3}$
15	Локальный максимум функции $-e^{-x}-100x$	30	Пересечение графиков функций $4(x-5)^2-15$ и $\frac{100}{x^2+2}-7$.