Методы тестирования ПО

МЕТОДЫ ПОИСКА ОШИБОК

Содержание лекции

Определимся, кто несет ответственность за качество системы

Изучим методы поиска ошибок

Задания

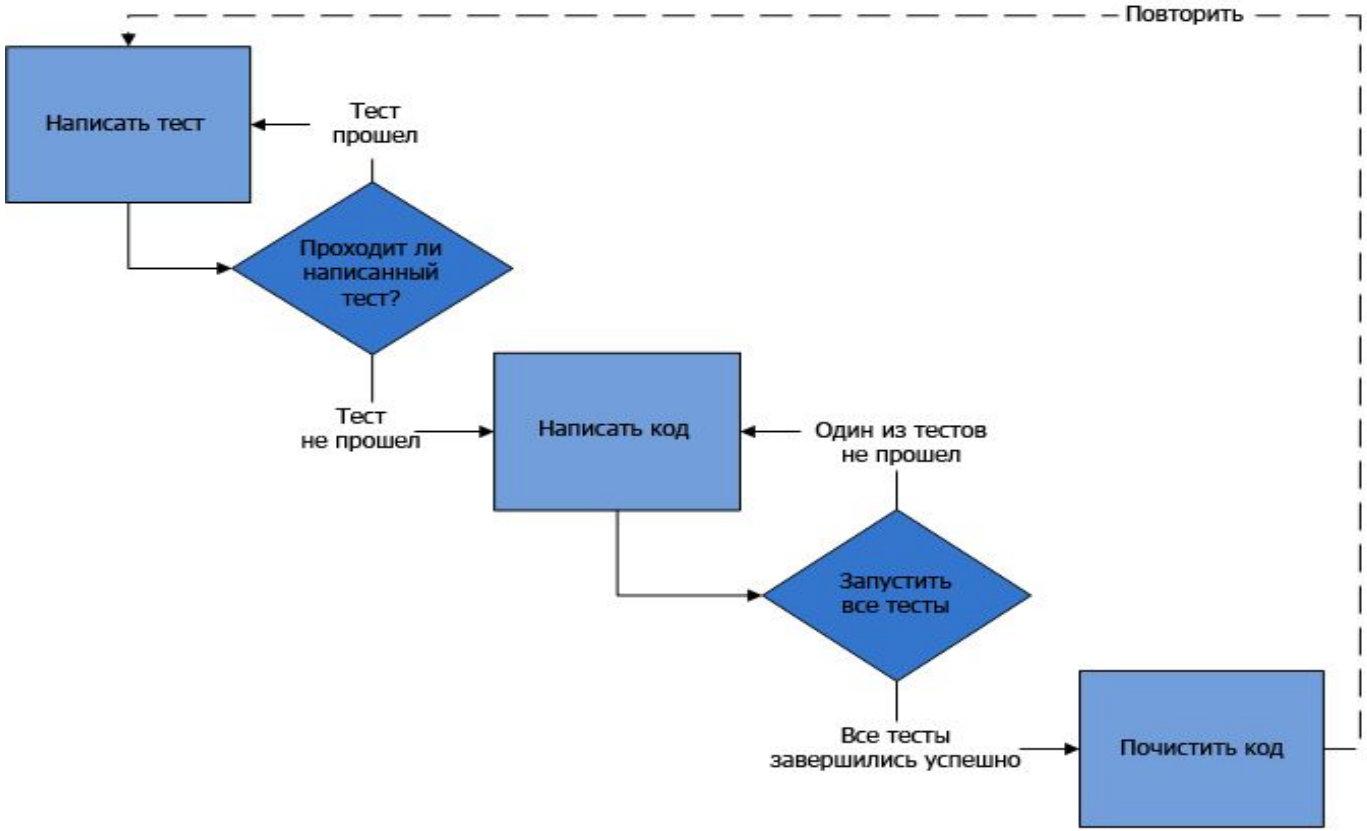
Качество vs ответственность

От кого зависит качество системы?

Кто несет ответственность за качество системы?

Классификация по знанию внутренней системы

Разработка через тестирование



Преимущества

1. Возможность убедиться, что приложение пригодно для тестирования
2. Автоматизированные тесты покрывают все пути исполнения
3. Разработчик продумывает детали интерфейса до реализации
4. Тесты заставляют делать свой код более приспособленным для тестирования
5. Модульное тестирование способствует формированию четких и небольших интерфейсов
6. Несмотря на то, что при разработке через тестирование требуется написать большее количество кода, общее время, затраченное на разработку, обычно оказывается меньше
7. Тесты защищают от ошибок. Поэтому время, затрачиваемое на отладку, снижается многократно
8. Устранение дефектов на более раннем этапе разработки, препятствует появлению хронических и дорогостоящих ошибок, приводящих к длительной и утомительной отладке в дальнейшем
9. Тесты позволяют производить рефакторинг кода без риска его испортить. При внесении изменений в хорошо протестированный код риск появления новых ошибок значительно ниже
10. Уверенность в том, что изменения не нарушит существующую функциональность, придает уверенность разработчикам и увеличивает эффективность их работы
11. Разработка через тестирование способствует более модульному, гибкому и расширяемому коду.
12. Тесты могут использоваться в качестве документации. Хороший код расскажет о том, как он работает, лучше любой документации.

Слабые места

1. Существуют задачи, которые невозможно решить
2. Прохождение функциональных тестов
3. Поддержка от руководства
4. Модульные тесты обычно пишутся теми же, кто пишет тестируемый код
5. Ложное ощущение надежности
6. Тесты сами по себе являются источником накладных расходов
7. Сложно определить покрытие тестами: неудачные архитектура, дизайн или стратегия тестирования приводят к большому количеству непройденных тестов, важно их все исправить в индивидуальном порядке. Простое удаление, отключение или поспешное изменение их может привести к необнаруживаемым пробелам в покрытии тестами.

Разработка тестов

МЕТОДЫ ПОИСКА ОШИБОК

Где искать тесты?

- Тщательное изучение и анализ требований (описания функции, модуля, спецификации, и т.д.).
- Декомпозиция требований\функций.
- Выявление всех условий, входных и выходных данных (что)
- Анализ поведения (как)
- Использование различных техник для выделения определенных тестов
- Использование накопленных знаний о выполненных проектах (оттестированных продуктах)
- Интуиция
- Анализ\просмотр выявленных тестов и добавление новых

Проблемы, которые придется решить

- Искать все ошибки или грубейшие?
- Если не все, то как установить порог допустимости ошибки?
- Когда завершать тестирование?
- Что делать, если сроки поджимают и/или нет ресурсов на дальнейшее тестирование?
- Где остановиться в документировании тестов?
- Изменять тест или следовать первоначальной инструкции?

Классы эквивалентности (partitioning, equivalent analysis)

- Анализируем входные и выходные данные:
  - правильные классы эквивалентности (корректные входные данные)
  - неправильные классы эквивалентности (ошибочные входные данные)

Лучшие представители

- Какие значения тестировать внутри класса эквивалентности?
- Используем предположения:
  - Множество возможных значений непрерывно
  - Значения могут быть спроецированы на числовую ось, мы всегда можем определить, что из двух значений одно больше, а другое меньше или они одинаковы

Разберем пример

Программа:

INPUT < 10

⇒ Результат: сообщение об ошибке

10 <= INPUT < 25

⇒ Результат: печать “Hello”

25 <= INPUT

⇒ Результат: сообщение об ошибке

Типы ошибок:

- Программа не принимает числовые значения как факт
  - Проверяется любым числом
- Написали <= 25 вместо < 25
  - Определяется только на границе
- Опечатка: написали 52 вместо 25
  - Определяется и на границе в том числе

Выводы:

Типы ошибок:

- Программа не принимает числовые значения как факт
  - Проверяется любым числом
- Написали <= 25 вместо < 25
  - Определяется только на границе
- Опечатка: написали 52 вместо 25
  - Определяется и на границе в том числе

Граничное значение проверяет все 3 типа ошибок

Значение не на границе проверяет только 1 тип ошибок

Анализ граничных значений (Boundary Value Testing)

- Идентифицировать граничные значения для каждого входного значения (класса эквивалентности)
  - на границе
  - значение, меньшее граничного («у границы»\'below point')
  - значение, большее граничного («за границей» \'above point')

Примеры:

- *Область корректных значений: [-1.0, 1.0]*
  -> тесты для -1.0, 1.0, -1.001, 1.001
- *Максимальная длина слова – 5 символов*
  - > тесты для 4,5,6
- *Область выходных значений: минимум расхода 0.00, максимум 2000*
  -> подбираем входные данные для того, чтобы получить на выходе 0.00, 2000.00, 2000.01, -0.01

В задаче:

INPUT < 10 Результат: сообщение об ошибке
10 <= INPUT < 25 Результат: печать “Hello”
25 <= INPUT Результат: сообщение об ошибке

Проецируем на числовую ось:

Граничные значения

Для точки: Z

-> Z-1, Z, Z+1

Для отрезка: [x, y]

-> x-1, x, y, y+1

А для такого интервала значений: (x, y) ?

Выбор значений

- Значение в пределах класса является лучшим представителем
- Граничные значения часто будут лучшими представителями
- Могут быть лучшие представители, которые не будут граничными значениями
- Могут быть выделены лучшие представители в классах, значения которых не будут очевидно сравнимы (больше-меньше)

Как тестируем?

- Классы эквивалентности: некорректные значения
  - Тестировать одно некорректное значение за раз для того, чтобы проверить, что система идентифицирует его корректно.

- Классы эквивалентности: корректные значения
  - Как ?

Стратегии работы с новым кодом

ТАБЛИЦЫ РЕШЕНИЙ (DECISION TABLE TESTING)

Этапы построения таблицы

1. Определить/записать все условия
2. Посчитать количество возможных комбинаций условий
3. Запомнить комбинации
4. Записать действия
5. Убрать лишние комбинации (схлопывание таблицы)

Пример: светофор

(SQA DAYS 14 - ЕЛЕНА СТАШЕНКО)

Автомобиль находится перед светофором, определить его дальнейшие действия

Условия:

- Горит ли красный? Y N
- Горит ли желтый? Y N
- Горит ли зеленый? Y N

Количество комбинаций:

- 2*2*2 = 8

1. Определить список возможных условий и их значения

Условия	Значения
Горит красный?	Y, N
Горит желтый?	Y, N
Горит зеленый?	Y, N

2. Определить список всех возможных действий (ожидаемых результатов для условий)

Действия
Ехать
Стоять
Готовиться
Специальное действие

3. Определить все значения для условий («да»\«нет» или более 2х значений) и их уникальные комбинации, которые приводят к выполнению ожидаемых результатов связанных с этим правилом

Условия	Значения	1	2	3	4	5	6	7	8
Горит красный?	Y, N	Y	Y	Y	Y	N	N	N	N
Горит желтый?	Y, N	Y	Y	N	N	Y	Y	N	N
Горит зеленый?	Y, N	Y	N	Y	N	Y	N	Y	N

Действия								
Ехать							X	
Стоять		X		X				
Готовиться		X						
Специальное действие	X		X		X	X		X

4. Создать тест кейс для каждого правила (столбца) – как минимум один, если условия бинарные и если условие – интервал значений, рассмотреть тесты как для нижней так и для верхней границы интервала

Дополнительные условия

Желтый может мигать? А зеленый? А красный?

Специальное действие – это что?

Все ли светофоры одинаковые?

Желтый может мигать?

Условия	Значения	1	2	3	4	5	6	7	8	9	10	11	12
Горит красный?	Y, N	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N
Горит желтый?	Y, N, Blinking	Y	Y	N	N	B	B	Y	Y	N	N	B	B
Горит зеленый?	Y, N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N

Действия

Ехать									X			
Стоять		X		X								
Готовиться		X										
Спец. действие	X		X		X	X	X	X		X	X	
Нерегулируется												X

Убрать лишние комбинации

Желтый может мигать?

Условия	Значения	1	2	3	4	5	6	7	8	9	10	11	12
Горит красный?	Y, N	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N
Горит желтый?	Y, N, Blinking	Y	Y	N	N	B	B	Y	Y	N	N	B	B
Горит зеленый?	Y, N	Y	N	Y	N	-	-	Y	N	Y	N	Y	N

Действия													
Ехать							X		X			
Стоять		X		X								
Готовиться		X					X					
Спец. действие	X		X		X	X		X		X	X	
Нерегулируется												X

Убрать лишние комбинации

Желтый может мигать?

Условия	Значения	1	2	3	4	5	7	8	9	10	11	12
Горит красный?	Y, N	Y	Y	Y	Y	Y	N	N	N	N	N	N
Горит желтый?	Y, N, Blinking	Y	Y	N	N	B	Y	Y	N	N	B	B
Горит зеленый?	Y, N	Y	N	Y	N	-	Y	N	Y	N	Y	N

Действия

Ехать						X		X			
Стоять		X		X							
Готовиться		X				X					
Спец. действие	X		X		X		X		X	X	
Нерегулируется											X

Стратегии работы с новым кодом

ФУНКЦИОНАЛЬНЫЕ ДИАГРАММЫ (CAUSE-EFFECT GRAPHING)

Что это и зачем?

- Предлагает способ перевода спецификаций, написанных на естественном языке, на язык формальный

- Способствует проектированию высокорезультативных тестов, не страдающих избыточностью, и обнаруживающих случаи неполноты и неоднозначности во входных спецификациях

Алгоритм действий

1. Разбить внешние спецификации на отдельные функции, которые будут тестироваться (декомпозиция функциональных требований)
2. Идентифицировать явные и неявные причины (условия на входе) и присвоить каждой из них уникальный номер
3. Идентифицировать явные и неявные эффекты (действия на выходе) и присвоить каждому из них уникальный номер
4. Перевести семантику спецификации в граф «причина-следствие» (Boolean cause-effect graphing)
5. Добавить информацию о невозможных комбинациях причин\эффектов
6. Построить таблицу решений (бинарные значения)
7. Записать тест кейс для каждого столбца

Пример

Requirements for Calculating Car Insurance Premiums:

For females less than 65 years of age, the premium is $500
For males less than 25 years of age, the premium is $3000
For males between 25 and 64 years of age, the premium is $1000
For anyone 65 years of age or more, the premium is $1500

Причины (Causes) – входные условия

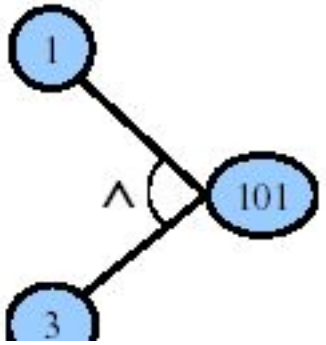
1. Пол: мужской
2. Пол: женский
3. Возраст: <25
4. Возраст: >=25 and < 65
5. Возраст: >= 65

Следствия (Effects) – выходные результаты

100. Премия = $1000
101. Премия = $3000
102. Премия = $1500
103. Премия = $500

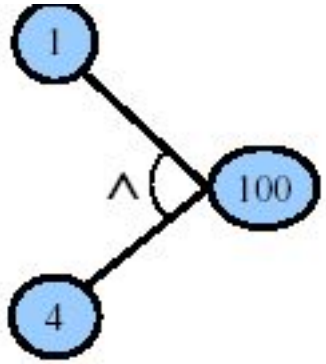
Причина:
1. Пол: мужской and
3. Возраст: < 25

Следствие:
101: Премия = $3000



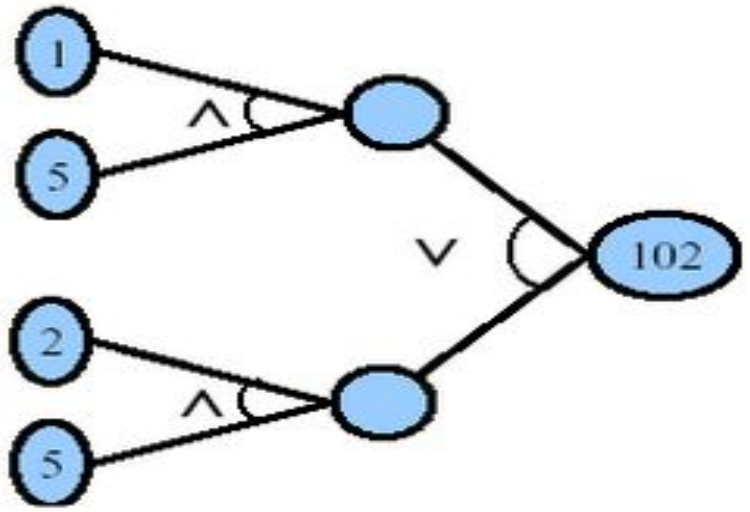
Причина:
1.Пол: мужской and
4. Возраст: >=25 and < 65

Следствие:
100: Премия = $1000



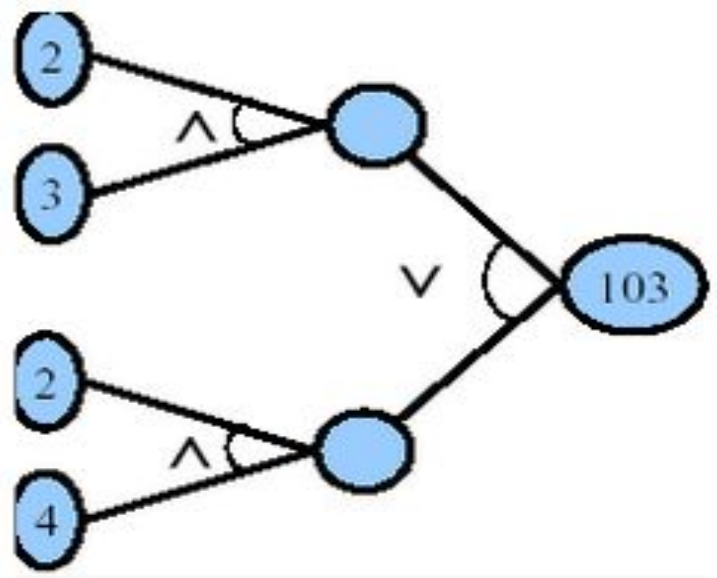
Причина:
1. Пол: мужской and
5. Возраст: >= 65
or
2. Пол: женский and
5. Возраст: >= 65

Следствие:
102: Премия = $1500



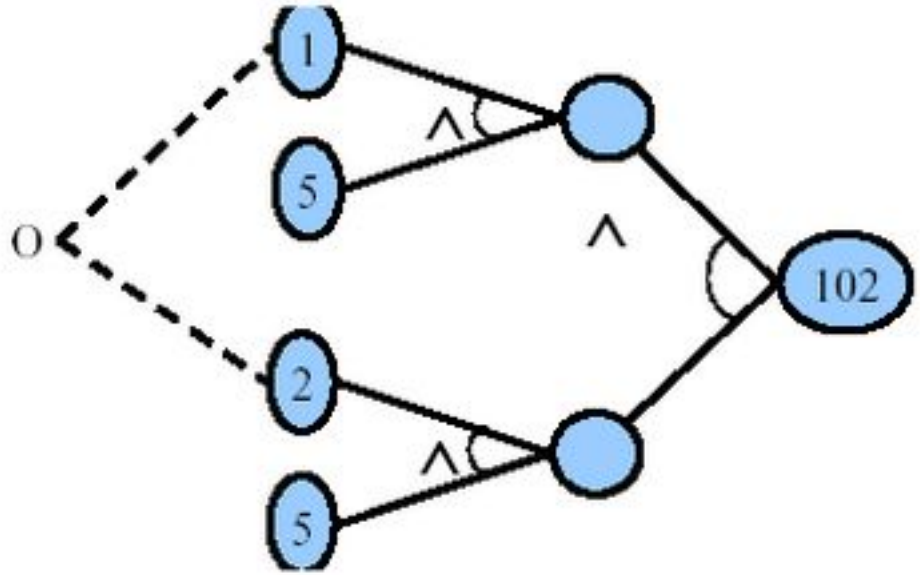
Причина:
2. Пол: женский and
3. Возраст: < 25
or
2. Пол: женский and
4. Возраст: >=25 and < 65

Следствие:
103: Премия = $500



Причина:
1. Пол: мужской and
5. Возраст: >= 65
or
2. Пол: женский and
5. Возраст: >= 65

Следствие:
102: Премия = $1500



Преобразование в таблицу решений

Test Cases	1	2	3	4	5	6
1 (мужской)	1	1	1	0	0	0
2 (женский)	0	0	0	1	1	1
3 (< 25)	1	0	0	0	1	0
4 (>= 25 and < 65)	0	1	0	0	0	1
5 (>= 65)	0	0	1	1	0	0
100 (1000$)	0	1	0	0	0	0
101 (3000$)	1	0	0	0	0	0
102 (1500$)	0	0	1	1	0	0
103 (500$)	0	0	0	0	1	1

Особенности применения ф. диаграмм

1. Требуется трансляция спецификации в булевскую логическую сеть
2. Обнаружение неполноты и неоднозначности в исходных спецификациях
3. Применение функциональных диаграмм не обеспечивает построение *всех* полезных тестов, которые могут быть определены:
   a. Как пример: метод неадекватно исследует граничные условия
4. Лучше отделять анализ граничных значений от метода функциональных диаграмм (иначе граф существенно усложняется)
5. Наиболее трудным при реализации метода является преобразование диаграммы в таблицу решений

Стратегии работы с новым кодом

ДРУГИЕ МЕТОДЫ

Метод предположение об ошибке (Error guessing)

- Этот метод в значительной степени является интуитивным
- Тест инженер использует свои знания системы и способность к интерпретации спецификации на предмет того, чтобы "предугадать" при каких входных условиях система может выдать ошибку
- Перечислить в некотором списке возможные ошибки или ситуации, в которых они могут появиться, а затем на основе этого списка написать тесты

Requirements-Driven Testing

Проверяем каждое требование\запрос, которое описано или озвучено

- анализ требований: выявление неоднозначностей, неточностей, пропущенной информации и т.п. (можно использовать функциональные диаграмма)

Отслеживаем все требования и их покрытие тестами

- список требований с идентификаторами и соответствующих тестов (Requirements Tracing Matrix)

Для каждого требования должны быть разработаны тесты

Задания

1. Классы эквивалентности
2. Граничные значения
3. Таблица решений
4. Функциональные диаграммы

1. Выполнить разбиение на классы эквивалентности

1.1 Password – длина не меньше 8 символов, максимум 16. Может состоять из латинских букв и цифр, а также могут быть использованы символы только из списка «!», «_», «?», «#». При этом пароль должен обязательно содержать, как минимум, одну заглавную букву и одну цифру.

1.2 Значение для 'Product ID' должно содержать 5 символов, первые два из них должны быть обозначениями из списка допустимых значений (A1 or A2 or B1 or B2), остальные три - уникальным числовым значением.

Определить граничные значения

2.1 Корректные значения X - целые значения от -2 до 10

2.2 Максимальное длина вводимого значения равна 20 символам

3. Составить таблицу решений

3.1 Страховая компания предоставляет страховку клиентам, достигшим 18-ти летнего возраста. Если стаж водителя составляет от 2-х до 6-ти лет, предоставляется скидка 20%. Если стаж водителя более шести лет, скидка 30%

3.2 Любому посетителю салона красоты «Жасмин» может быть присвоена одна из категорий: «Клиент», «Клиент категории А», «Клиент категории Б», «Клиент категории С» в зависимости от количества посещение салона.

Категория «Клиент» присваивается посетителю, на счету которого 3 посещения и более.

Категория «Клиент категории А» присваивается посетителю, на счету которого 10 посещений и более.

Категория «Клиент категории Б» присваивается посетителю, на счету которого 20 посещений и более.

Категория «Клиент категории С» присваивается посетителю, на счету которого 30 посещений и более.

3. Составить таблицу решений

3.2 Система скидок магазина

Скидки предоставляются покупателям, которые приобрели накопительную карту магазина. Изначально карта имеет тип “Standard” с нулевым балансом. При покупке товара и предъявлении карты при оплате, сумма покупок зачисляется на баланс карты. Величина скидки зависит от общей суммы покупок на карте покупателя и от типа карты.

Для карты тип “Standard” скидки составляют:

- 5%, если общая сумма покупок на карте от 20000 руб до 40000 руб включительно,
- 10%, если сумма на карте больше, чем 40000 руб.

Магазин меняет карту типа “Standard” на карту типа “Silver Card”, если накопительная сумма покупателя на карте типа “Standard” становится равной или больше 50000 руб. Для карты тип “Silver Card” скидки составляют:

- 10%, если сумма на карте от 50000 руб до 70000 руб включительно,
- 20%, если сумма на карте больше, чем 70000 руб.

Магазин меняет карту типа “Silver Card” на карту типа “Gold Card”, если накопительная сумма покупателя на карте типа “Silver Card” становится равной или больше 100000 руб. Для карты типа “Gold Card” скидки составляют:

- 20%, если сумма на карте от 100000 руб до 150000 руб включительно,
- 30%, если сумма на карте больше, чем 150000 руб.

Магазин меняет карту типа “Gold Card” на карту типа “VIP Card”, если накопительная сумма покупателя на карте типа “Gold Card” становится равной или больше 200000 руб. Для карты типа “VIP Card” скидки составляют: 30%, если сумма на карте больше, чем 200000 руб

4. Применить метод функциональных диаграмм

4.1 Для банкомата банка «ТТТ» реализовано ПО, которое автоматизирует такие функции как выдача денег, выдча справки о балансе (доступные средства на карте), выдача распечатки с 10ю последними операциями по карте, оплата услуг по мобильной связи

Проанализирйте спецификацию для функции «Обработка запроса на снятие суммы с карты» и примените метод функциональных диаграмм для создания тест кейсов. Разработать и описать тест-кейсы в матрице (xls-file).

- Спецификация для функции «Обработка запроса на снятие суммы с карты»:

Если карта типа «кредитная» (K) или «дебетовая» (D), то банкомат выдает деньги клиенту при условии, что запрашиваемая сумма (X) не превышает сумму доступных средств на карте клиента (S).
Если карта типа «кредитная», то банкомат выдает деньги и в случае, если запрашиваемая сумма превышает сумму доступных средств на карте, но не выходит за рамки допустимого превышения кредита (L).
В случае, если карта не является «кредитной» или «дебетовой» или же запрашиваемая сумма превышает сумму доступных средств на карте для дебетовой карты или же запрашиваемая сумма превышает сумму доступных средств на карте и выходит за рамки допустимого превышения кредита для кредитовой карты, тогда выдается сообщение о том, что деньги не могут быть выданы и деньги не выдаются.