Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»

Домашнее задание №3 по дисциплине
«Технологии разработки программного обеспечения»

ИСПОЛНИТЕЛЬ:

Ван Чаочао
Группа ИУ5И-12М

Москва 2021

Домашнее задание №3 по курсу Технологии разработки ПО

Выполняется на основе программного проекта из ДЗ2 (далее по тексту - **проекта)**.

1. (**базовое**) Для основного прецедента (с учетом альтернативных путей) составить тестовые варианты для функционального тестирования (по варианту):
   - разбиение на классы эквивалентности;
   - анализ граничных значений.

2. (**расширенное**) Для одной из функций составить тестовые варианты для структурного тестирования (по варианту):
   - метод тестирования базового пути;
   - метод тестирования потока данных.

3. (**дополнительное**) Составить диаграмму классов подсистемы/модуля проекта. Вычислить метрики (по варианту):
   - Абреу (по всем классам);
   - Лоренца-Кидда (для каждого класса, среднее по всем и для проекта);
   - Чидамбера-Кемерера (для каждого класса и среднее по всем).

В отчет:

1. Описание прецедента. Описания классов эквивалентности или граничных условий. Тестовые варианты для функционального тестирования.
2. Исходный код функции для тестирования; управляющий граф, информационный граф, цикломатическую сложность; независимые пути, DU-цепочки и маршруты для них (по варианту); тестовые варианты.
3. Диаграмма классов подсистемы/модуля проекта. Расчет метрик классов (по варианту).
4. (в электронном виде) Исходный код проекта.

Веб-сервис по продажам автомобилей	Ван Чаочао	ИУ5И-12М	1 (фу,рп)	М5	10	**67**	table module	active record	Базового пути	Классы эквив.	Чидамбера-Кемерера

Прецедент – «Проверьте свой уровень»

Предусловия:

1. Устройство пользователя может быть корректно подключено к сети.
2. Пользователь авторизован для входа в систему.

Главный поток:

Пользователь вводит марку, объем двигателя, объем продаж и количество мест в автомобиле в качестве условий отправки запроса.

Подпотоки:

Система возвращает пользователю релевантную информацию об автомобиле. Система проверит правильность введенных данных и наличие подходящего автомобиля.

Альтернативный поток:

Если введенное условие запроса транспортного средства неверно, система вернет сообщение об ошибке и отобразит сообщение об ошибке в окне подсказки.

Постусловия:

1. Если ошибки ввода нет, в окне будет отображаться необходимая пользователю информация об автомобиле.
2. Если при наборе текста произошла ошибка, во всплывающем окне отобразится сообщение об ошибке.

Существуют некоторые особые случаи, когда следующие граничные значения являются выполнимыми:

1. Для некоторых профессиональных брендов вводить все параметры необязательно.

Анализ ребер граничных значений:

1. Обозначенная марка автомобиля - BMW.
2. Водозабор 1,6 л.
3. Заявленная квота-8.
4. Старт продаж 0
5. Не вводите никакой информации

Анализ граничных значений

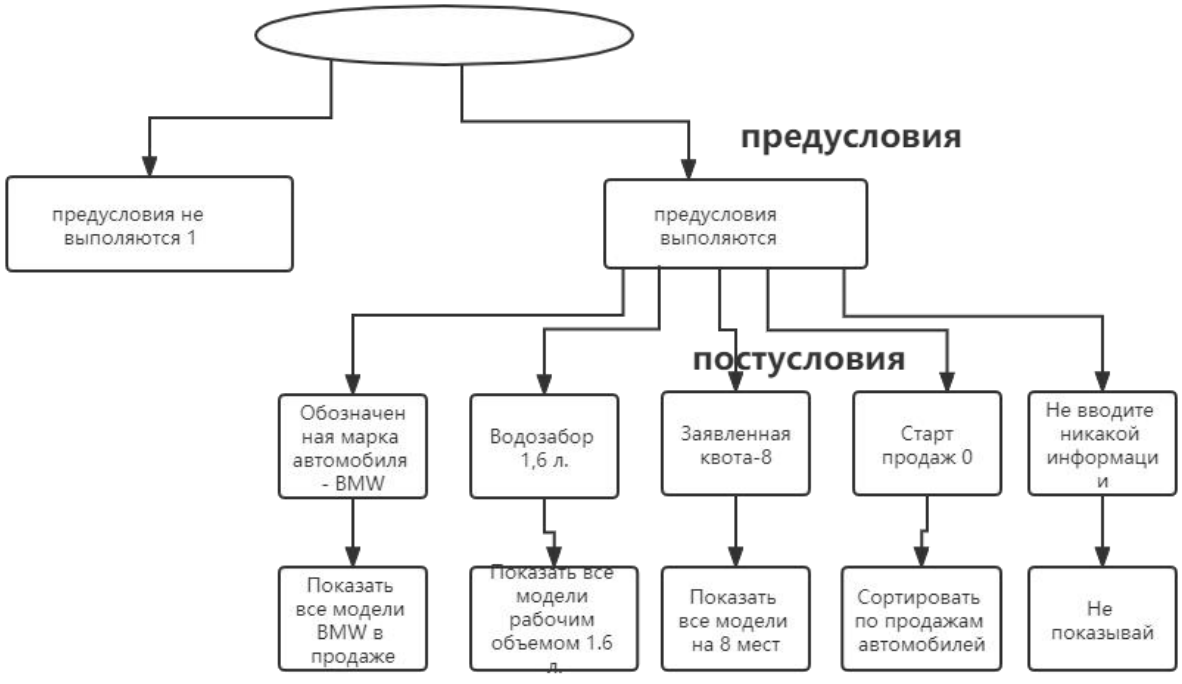
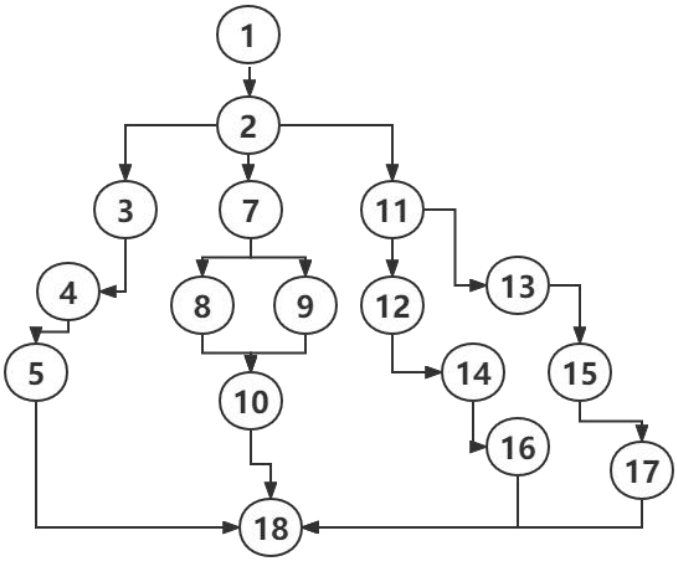


Рис1: Анализ граничных значений

- ТВ1: Пользовательский ввод - Марка автомобиля.
  - ИД: car_brand="BMW"
  - ОР: Отображение результатов всех продаваемых марок автомобилей.
  - ТВ2: Пользовательский ввод -Смещение автомобиля
  - ИД: car_ Displacement=1.6;
  - ОР: Вывод Показать все модели рабочим объемом 1.6 л.
- ТВ3: Пользовательский ввод –Количество мест в автомобиле
  - ИД: car_ numberOfSeats =8;
  - ОР: Вывод: В продаже модели автомобилей с указанием необходимого количества посадочных мест.
  - ТВ4: Сортировать выбранные модели автомобилей по объему продаж
  - ИД: car_ Sales >=1600;
  - ОР: Вывод: Показать модели с продажами более 1600
- ТВ5: имя не введено
  - ИД: car_name=《》
  - ОР: Вывод:введите название машины, которую хотите купить.

```
function fnLogin() {
    var oUname = document.getElementById("uname") (1)
    var oUpass = document.getElementById("upass")  (1)
    var oError = document.getElementById("error_box")  (1)
    var isError = true;
    (2)                          (3)
    if (oUname.value.length > 20 || oUname.value.length < 6) {
        oError.innerHTML = "Пожалуйста, введите 6-20 символов для имени пользователя."; (4)
        isError = false; (5)
        return; (6)
        (7)                                  (8)
    }else if((oUname.value.charCodeAt(0)>=48) && (oUname.value.charCodeAt(0)<=57)){
        oError.innerHTML = "Первым символом должна быть буква";  (9)
        return; (10)
        (11)                             (12)
    }else for(var i=0;i<oUname.value.charCodeAt(i);i++){
        (13)                                (14)
        if((oUname.value.charCodeAt(i)<48)||(oUname.value.charCodeAt(i)>57) &&
                (15)                                        (16)
        (oUname.value.charCodeAt(i)<97)||(oUname.value.charCodeAt(i)>122)){
        oError.innerHTML = "Должен состоять из букв и цифр"; (17)
        return; (18)
    } ^lambda
}
```

- Шаг 1. Создается потоковый граф



- Шаг 2. Цикломатическая сложность

a) V(G) = 4 региона

б) 18-4 = 14

в) 3+1 = 4

- Шаг 3. Базовое множество независимых путей

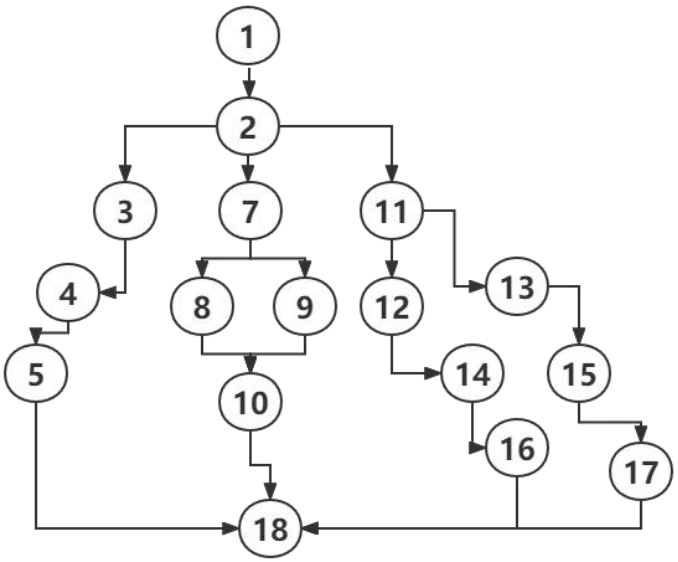
Путь 1: 1–2–3–4–5–18.

Путь 2: 1–2–7–8-9-10–18.

Путь 3: 1–2–11–12–14–16–18.

Путь 4: 1–2–13–15–17–18.

- Подготовка тестовых вариантов для каждого независимого пути:
- а)
  - ИД: carArray[i].carBrand =’ BMW’
  - ОЖ.РЕЗ.: location.assign("main.html")
- б)
  - ИД: userArray[i].userName =《wang15825》, userArray[i].passWord=《158》
  - ОЖ.РЕЗ.: alert("неправильное имя пользователя или пароль！")
- в)
  - ИД: userArray[i].userName =《zhangsan》, userArray[i].passWord=《1235》
  - ОЖ.РЕЗ.: alert("неправильное имя пользователя или пароль！")



- **DU цепочка:**

[document.getElementById("userName").value,1,4]

[document.getElementById("passWord").value,1,5]

[isLogin,1,6]

- **пути:**

[1,2,3,4] , [1,2,3,4,5] ,[1,2,3,4,5,6,7,8],[1,2,3,4,8],[1,2,3,5,8]

- **маршруты:**

[1,2,3,4,5,6,7,8]

- TB1:
  - ИД: userArray[i].userName =《zhangsan》, userArray[i].passWord=《123》
  - OP: location.assign("main.html").
- TB2:
  - ИД: userArray[i].userName =《zhangsan》, userArray[i].passWord=《1235》
  - OP: Вывод "Пожалуйста, введите номер карты заново, в номере карты слишком много цифр.
- TB3:
  - ИД: userArray[i].userName =《zhang san》, userArray[i].passWord=《123》
  - OP: Вывод: alert("неправильное имя пользователя или пароль！ ")

3.Метрики Лоренца и Кидда

Коллекция метрик Лоренца и Кидда — результат практического, промышленного подхода к оценке ОО-проектов .

М. Лоренц и Д. Кидд подразделяют метрики, ориентированные на классы, на четыре категории: метрики размера, метрики наследования, внутренние и внешние метрики

Метрика 1: Размер класса CS (Class Size)

общее количество операций (вместе с приватными и наследуемыми

экземплярными операциями), которые инкапсулируются внутри класса;

количество свойств (вместе с приватными и наследуемыми экземплярными свойствами), которые инкапсулируются классом.

Большие значения CS указывают, что класс имеет слишком много обязанностей. Они уменьшают возможность повторного использования класса, усложняют его реализацию и тестирование.

Рекомендуемое значение CS <= 20 методов

Метрика 2: Количество операций, переопределяемых подклассом, NOO (Number of Operations Overridden by a Subclass)

Большие значения NOO обычно указывают на проблемы проектирования, так же нарушается абстракция суперкласса, ослабляется иерархия классов, усложняет тестирование и модификацию программного обеспечения.

Рекомендуемое значение NOO <= 3 методов.

Метрика 3: Количество операций, добавленных подклассом, NOA (Number of Operations Added by a Subclass)

С ростом NOA подкласс удаляется от абстракции суперкласса.

Обычно при увеличении высоты иерархии классов (увеличении DIT) должно уменьшаться значение NOA на нижних уровнях иерархии.

Для рекомендуемых значений CS = 20 и DIT = 6 рекомендуемое значение NOA <= 4 методов (для класса-листа).

Метрика 4: Индекс специализации SI (Specialization Index)

$$SI = (NOO \times \text{уровень}) / M_{общ},$$

где уровень — номер уровня в иерархии, на котором находится подкласс, Мобщ — общее количество методов класса.

Чем выше значение SI, тем больше вероятность того, что в иерархии классов есть классы, нарушающие абстракцию суперкласса.

Рекомендуемое значение SI <= 0,15.

Метрика 5: Средний размер операции OSAVG (Average Operation Size)

количество строк программы.

Альтернативный вариант — «количество сообщений, посланных операцией».

Рост значения означает, что обязанности размещены в классе не очень удачно.

Рекомендуемое значение OSAVG <= 9.

Метрика 6: Сложность операции OC (Operation Complexity

Сложность операции может вычисляться с помощью стандартных метрик сложности, то есть с помощью LOC- или FP-оценок, метрики цикломатической сложности, метрики Холстеда.

Метрика 7: Среднее количество параметров на операцию NPAVG (Average Number of Parameters per operation)

Чем больше параметров у операции, тем сложнее сотрудничество между объектами. Поэтому значение NPavg должно быть как можно меньшим.

Рекомендуемое значение NPavg = 0,7.

Метрика 8: Количество описаний сценариев NSS (Number of Scenario Scripts)

Рекомендуется — не менее одного сценария на публичный протокол системы, отражающий основные функциональные требования к подсистеме.

Метрика 9: Количество ключевых классов NKC (Number of Key Classes)

Ключевой класс прямо связан с проблемной областью.

20-40% от общего количества классов. Оставшиеся классы реализуют общую инфраструктуру.

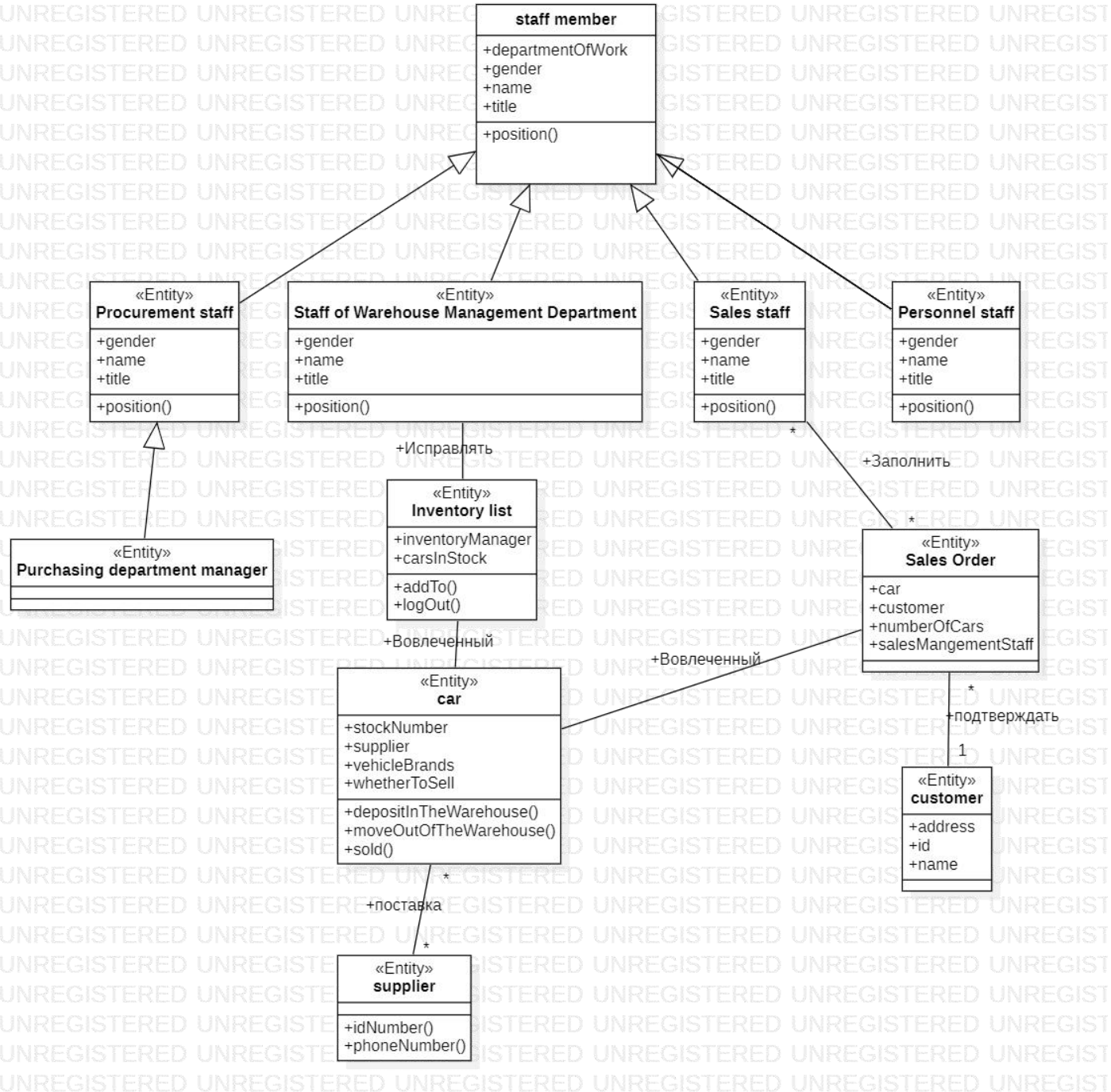
Рекомендуемое значение: если NKC < 0,2 от общего количества классов системы, следует углубить исследование проблемной области.

Метрика 10: Количество подсистем NSUB (NumberofSUBsystem)

размещение ресурсов, планирование (с акцентом на параллельную разработку), общие затраты на интеграцию.

Рекомендуемое значение: NSUB > 3.

Диаграмма Классов



Метрика 1:

user количества свойств : 3 количества операций класса: 8 средние: 5.5

database количества свойств : 0 количества операций класса: 6 средние: 3

administered количества свойств : 2 количества операций класса: 5 средние: 3.5

loadrate количества свойств : 3 количества операций класса: 0 средние: 1.5

deposit количества свойств : 3 количества операций класса: 0 средние: 1.5

loadrank количества свойств : 3 количества операций класса: 0 средние: 1.5

Метрика 4:

Уровень 1

user SI = (6*1)/8=0.75

database SI = (6*1)/6=1

administered SI = (5*1)/6=0,833

Уровень 2

loadrate SI = (0*2)/4=0

loadrank SI = (0*2)/5=0

deposit SI = (0*2)/3=0

Метрика 6:

Например user

Параметр	Вес
Вызовы функций API	3,0
Присваивания	0,5
Арифметические операции	2,0
Сообщения с параметрами	3,0

Вложенные выражения	0,5
Параметры	0,3
Простые вызовы	7,0
Временные переменные	0,5
Сообщения без параметров	1,0

Получим результаты:

Имя класса	CS	NOO	NOA	SI		OC		NSS		
user	**5.5**	**6**	**8**	**0.75**	**7**	**21**	**0,2**	**1**	**0,2**	**1**
database	**3**	**6**	**4**	**1**	**7**	**62**	**0,7**	**2**	**0,4**	**3**
administered	**3.5**	**5**	**5**	**0,833**	**5**	**43**	**0,5**	**1**	**0,4**	**3**
loadrate	**1.5**	**0**	**0**	**0**	**3**	**36**	**0,4**	**2**	**0,4**	**6**
loadrank	**1.5**	**0**	**0**	**0**	**3**	**28**	**0,35**	**3**	**0,3**	**5**
deposit	**1.5**	**0**	**0**	**0**	**3**	**30**	**0,4**	**2**	**0,3**	**3**

Список литературы

1. Технологии разработки программного обеспечения: Учебник/ С. Орлов. —СПб.: Питер, 2002 —464 с.: ил. ISBN 5-94723-145-X
2. Виноградова М.В., Белоусова В.И.Унифицированный процесс разработки программного обеспечения: учебное пособие / Виноградова М.В., Белоусова В.И. –М.: МГТУ им. Н.Э. Баумана. –2015 г. –82 с. -Режим доступа:

http://ebooks.bmstu.ru/catalog/193/book1303.html (дата обращения: 17.12.2017). —ISBN: 978-5-7038-4265-2

3. Методическое пособие к ДЗNo3 по дисциплине ТРПО