Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Институт прикладной математики и компьютерных наук

УТВЕРЖДЕНО:
Директор
А. В. Замятин

Оценочные материалы по дисциплине

Рефакторинг и обратное проектирование

по направлению подготовки

02.04.02 Фундаментальная информатика и информационные технологии

Направленность (профиль) подготовки:
Системная инженерия и управление IT-проектами

Форма обучения
Очная

Квалификация
Магистр

Год приема
2025

СОГЛАСОВАНО:
Руководитель ОП
А.Н. Моисеев

Председатель УМК
С.П. Сущенко

Томск – 2025

1. Компетенции и индикаторы их достижения, проверяемые данными оценочными материалами

Целью освоения дисциплины является формирование следующих компетенций:

ПК-1 Способен проектировать программное обеспечение.

ПК-2 Создает архитектурный проект программного средства.

Результатами освоения дисциплины являются следующие индикаторы достижения компетенций:

ИПК-1.1 Использует существующие типовые решения и шаблоны проектирования программного обеспечения

ИПК-1.2 Применяет методы и средства проектирования программного обеспечения, структур данных, баз данных, программных интерфейсов

ИПК-1.3 Знает методы и средства проектирования программного обеспечения, методы и средства проектирования баз данных

ИПК-2.1 Оценивает возможность создания архитектурного проекта программного средства

ИПК-2.2 Определяет цели архитектуры программного средства

ИПК-2.3 Определяет ключевые сценарии для архитектуры программного средства

2. Оценочные материалы текущего контроля и критерии оценивания

Элементы текущего контроля:

– тесты.

Тест (ИПК-1.1, ИПК-1.2)

1. Какие проблемы проектирования можно решить с помощью рефакторинга?

a) Улучшение читаемости кода
b) Оптимизация производительности
c) Снижение сложности кода
d) Все вышеперечисленное

2. Как можно устранить повторяющийся код?

a) Использование циклов
b) Выделение общих частей в функции
c) Применение шаблонов проектирования
d) Все вышеперечисленное

3. Что такое рефакторинг?

a) Изменение функциональности кода
b) Изменение структуры кода без изменения его поведения
c) Добавление новых функций
d) Оптимизация производительности

4. Когда необходимо проводить рефакторинг?

a) Перед каждым коммитом
b) При добавлении новых функций
c) При появлении технического долга
d) В любом из вышеперечисленных случаев

5. Как можно сократить длинный метод?

a) Разделить его на несколько мелких методов
b) Использовать более мощные операторы
c) Увеличить объем комментариев
d) Убрать ненужные проверки

6. Какие метрики используются для оценки сложности кода?
a) Количество строк кода
b) Условная сложность
c) Когнитивная сложность
d) Все вышеперечисленное

7. Как вычисляется условная сложность кода?
a) Подсчетом количества условных операторов
b) Анализом времени выполнения кода
c) Подсчетом количества функций
d) Оценкой объема памяти, используемой программой

8. Дайте рекомендации по корректному именованию переменных и методов.
a) Имена должны быть короткими и без пробелов
b) Имена должны отражать суть переменной или метода
c) Имена могут быть любыми, если используются комментарии
d) Используйте только заглавные буквы

9. Что такое магические константы и как их следует избегать?
a) Константы, значение которых непонятно из контекста
b) Константы, которые не изменяются в процессе выполнения программы
c) Константы, определенные в одном месте программы
d) Константы, которые задаются случайным образом

10. Приведите примеры «хороших» комментариев.
a) Объяснение сложного алгоритма
b) Пояснение причин использования определенного метода
c) Комментарии, поясняющие каждый шаг кода
d) Указание на ошибки в коде

Ключи: 1 г), 2 б), 3 б), 4 г), 5 а), 6 г), 7 а), 8 б), 9 а), 10 а).

Критерии оценивания: тест считается пройденным, если обучающий ответил правильно как минимум на половину вопросов.

3. Оценочные материалы итогового контроля (промежуточной аттестации) и критерии оценивания

Студент имеет право проходить промежуточную аттестацию только при успешном прохождении текущего контроля.

Экзаменационный билет состоит из двух частей.

Первая часть представляет собой два теоретических вопроса, проверяющих ИПК-1.1, ИПК-1.2, ИПК-1.3. Ответ на вопросы первой части дается в устной развернутой форме.

Вторая часть содержит практическую задачу, проверяющую ИПК-2.1, ИПК-2.2, ИПК-2.3. Ответ на задачу второй части дается в устной развернутой форме.

Перечень теоретических вопросов (ИПК-1.1, ИПК-1.2, ИПК-1.3):

Билет 1:
Какие проблемы проектирования можно решить с помощью рефакторинга?
Повторяющийся код.

Билет 2:

Что такое рефакторинг? В чем его цель? Когда его необходимо проводить?
Длинный метод.

Билет 3:
Перечислите метрики оценки сложности кода. Как они вычисляются?
Условная сложность.

Билет 4:
Дайте рекомендации по корректному именованию переменных и методов.
Одержимость элементарными типами, Магические константы.

Билет 5:
Приведите примеры “хороших” и “плохих” комментариев.
Indecent exposure.

Билет 6:
Что такое инверсия управления?
Расплывшееся решение, Стрельба дробью, Расходящиеся модификации.

Билет 7:
В чем смысл принципа единственной ответственности?
Альтернативные классы с различными интерфейсами.

Билет 8:
В чем смысл принципа открытости-закрытости?
Ленивый класс.

Билет 9:
В чем смысл принципа Лисков?
Большой класс.

Билет 10:
В чем смысл принципа разделения интерфейсов?
Завистливые функции.

Билет 11:
В чем смысл инверсии зависимостей?
Группы данных.

Билет 12:
Объясните принципы DRY и KISS.
Параллельные иерархии наследования.

Билет 13:
В чем заключается закон Деметры?
Теоретическая общность.

Билет 14:
Какие виды тестирования вы знаете? В чем они заключаются? Когда применяются?
Временное поле.

Билет 15:
Что такое FIRST?
Цепочки сообщений.

Билет 16:
Какие приемы тестирования вы знаете?
Посредник.

Билет 17:
Что такое мок? Что такое стаб? В чем разница?
Неполнота библиотечного класса.

Билет 18:
Назовите положительные и отрицательные стороны кодревью.
Классы данных.

Примеры задач (ИПК-2.1, ИПК-2.2, ИПК-2.3):

1. Проанализируйте качество представленного фрагмента исходного кода.
2. Вам предоставлен исходный код устаревшей системы управления складом. Код имеет множество проблем: дублирование, нарушение принципов SOLID, отсутствие четкой архитектуры. Опишите процесс оценки возможности рефакторинга этой системы с целью создания нового архитектурного проекта. Какие метрики и инструменты вы бы использовали для анализа текущего состояния кода?
3. Вы работаете над рефакторингом большого монолитного приложения для его преобразования в микросервисную архитектуру. Определите основные цели новой архитектуры, учитывая текущие проблемы системы и потребности в масштабировании. Как эти цели повлияют на процесс рефакторинга?
4. При рефакторинге архитектуры CRM-системы выявите 7-10 ключевых сценариев, которые наиболее критичны с точки зрения производительности и масштабируемости. Для каждого сценария предложите конкретные техники рефакторинга (например, выделение сервиса, применение кэширования, оптимизация запросов к базе данных), которые помогут улучшить архитектуру.

Критерии оценивания:

Результаты экзамена определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Оценка «отлично» выставляется, если на оба теоретических вопроса даны развернутые ответы, и задача решена без ошибок. Ответы на теоретические вопросы должны быть полными, демонстрировать глубокое понимание концепций рефакторинга и его применения. Решение задачи должно быть подробным, с обоснованием каждого шага.

Оценка «хорошо» выставляется, если даны правильные ответы на оба теоретических вопроса, но один из ответов недостаточно развернут или содержит незначительные неточности. Задача, может быть, не решена или решена с незначительными ошибками. Альтернативно, оценка «хорошо» может быть выставлена, если дан полный и правильный ответ на один теоретический вопрос и безошибочно решена задача, но отсутствует ответ на второй теоретический вопрос.

Оценка «удовлетворительно» выставляется, если дан развернутый и правильный ответ на один теоретический вопрос или правильное решение задачи. При этом ответ на второй теоретический вопрос может отсутствовать или содержать существенные ошибки. Альтернативно, оценка «удовлетворительно» может быть выставлена, если даны краткие, но правильные ответы на оба теоретических вопроса, даже если задача не решена.

Оценка «неудовлетворительно» выставляется, если не дан правильный развернутый ответ ни на один теоретический вопрос, решение задачи содержит грубые ошибки.

4. Оценочные материалы для проверки остаточных знаний (сформированности компетенций)

Тест

1. Какой из следующих антипаттернов характеризуется избыточной связью между модулями? (ИПК-1.1)
   a) Золотой молоток
   b) Спагетти-код
   c) Большой комок грязи
   d) Божественный объект

2. Какой антипаттерн можно устранить с помощью шаблона проектирования «Интерфейс» (Interface)? (ИПК-1.1)
   a) Большой комок грязи
   b) Хрупкое основание
   c) Альтернативные классы с различными интерфейсами
   d) Магические константы

3. Какой метод рефакторинга наиболее эффективен для устранения дублирующегося кода? (ИПК-1.2)
   a) Вынос метода
   b) Инлайн метода
   c) Разделение переменных
   d) Инкапсуляция поля

4. Какой из шаблонов проектирования может помочь устранить антипаттерн «Большой класс»? (ИПК-1.2)
   a) Делегирование
   b) Фасад
   c) Посредник
   d) Стратегия

5. Как антипаттерн «Завистливые методы» влияет на архитектуру программного средства? (ИПК-2.2)
   a) Создает избыточную зависимость между классами
   b) Усложняет поддерживаемость архитектуры
   c) Влияет на производительность системы
   d) Все вышеперечисленное

6. Какого принципа проектирования следует придерживаться для предотвращения антипаттерна «Хрупкое основание»? (ИПК-2.2)
   a) Принцип открытости-закрытости
   b) Принцип инверсии зависимостей
   c) Принцип единственной ответственности
   d) Принцип разделения интерфейсов

7. Как можно устранить антипаттерн «Ленивый класс»? (ИПК-1.3)
   a) Удалить класс

b) Объединить класс с другим классом
c) Расширить функциональность класса
d) Все вышеперечисленное

8. Какие сценарии могут привести к появлению антипаттерна «Стрельба дробью»? (ИПК-2.3)
a) Частые изменения в одном классе, которые требуют изменений в других классах
b) Изменения в одном модуле, которые не требуют изменений в других
c) Отсутствие комментариев в коде
d) Использование глобальных переменных

9. Какое действие поможет устранить антипаттерн «Магические константы» в коде? (ИПК-1.2)
a) Замена магических чисел именованными константами
b) Разделение больших методов на мелкие
c) Использование глобальных переменных для всех констант
d) Использование комментариев для объяснения значений чисел

10. Как антипаттерн «Большой комок грязи» влияет на возможность создания архитектурного проекта программного средства? (ИПК-2.1)
a) Усложняет рефакторинг и модульность системы
b) Упрощает проектирование из-за централизованной структуры
c) Делает систему более масштабируемой
d) Позволяет быстрее внедрять изменения

Ключи: 1 c), 2 c), 3 a), 4 a), 5 d), 6 a), 7 d), 8 a), 9 a), 10 a).

Информация о разработчиках

Иванова Лидия Сергеевна, кандидат технических наук, кафедра программной инженерии НИ ТГУ, старший преподаватель.