Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Максимов Алексей Борисович
Должность: директор департамента по образовательной политике
Дата подписания: 02.11.2023 11:11:06
Уникальный программный ключ:
8db180d1a3f02ac9e60521a5672742735c18b1d6

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
Факультет информационных технологий

УТВЕРЖДАЮ
Декан факультета
«Информационные технологии»
_____/Д.Г.Демидов/
«16» мая 2022

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Рефакторинг»

Направление подготовки/специальность
09.03.02 Информационные системы и технологии

Профиль/специализация
Автоматизированные системы обработки информации и управления

Квалификация
Бакалавр

Формы обучения
Очная

Москва, 2022 г.

Разработчик(и):
ст. преподаватель  / М.В. Алпатова /

Согласовано:
Заведующий кафедрой
«Информатики и информационных технологий»,
к.т.н.  / Е.В. Булатников/

Содержание

1 Цели, задачи и планируемые результаты обучения по дисциплине 4
2 Место дисциплины в структуре образовательной программы 5
3 Структура и содержание дисциплины 5
3.1 Виды учебной работы и трудоемкость 5
3.2 Тематический план изучения дисциплины 5
3.3 Содержание дисциплины 7
3.4 Тематика семинарских/практических и лабораторных занятий 8
3.5 Тематика курсовых проектов (курсовых работ) 9
4 Учебно-методическое и информационное обеспечение 9
4.1 Нормативные документы и ГОСТы 9
4.2 Основная литература 9
4.3 Дополнительная литература 10
4.4 Электронные образовательные ресурсы 10
4.5 Лицензионное и свободно распространяемое программное обеспечение 10
4.6 Современные профессиональные базы данных и информационные справочные системы 10
5 Материально-техническое обеспечение 10
6 Методические рекомендации 10
6.1 Методические рекомендации для преподавателя по организации обучения 10
6.2 Методические указания для обучающихся по освоению дисциплины 11
7 Фонд оценочных средств 11
7.1 Методы контроля и оценивания результатов обучения 11
7.2 Шкала и критерии оценивания результатов обучения 11
7.3 Оценочные средства 12

1 Цели, задачи и планируемые результаты обучения по дисциплине

Целью дисциплины "Рефакторинг" является формирование у студентов способности к концептуальному, функциональному и логическому проектированию информационных систем, с акцентом на оптимизацию и улучшение существующего программного кода. Студенты научатся прибегать к рефакторингу как инструменту для повышения эффективности и надежности информационных систем среднего и крупного масштаба.

В рамках дисциплины ставятся следующие задачи, соответствующие задачам профессиональной деятельности, указанным в ФГОС ВО:

- Освоение концепций и методологий рефакторинга, направленных на оптимизацию и улучшение программного кода.
- Приобретение навыков идентификации "запахов кода" и выбора соответствующих методов рефакторинга.
- Изучение принципов чистого кода и методов написания эффективного и читаемого кода.
- Применение полученных навыков в практической деятельности с акцентом на проектирование и оптимизацию информационных систем среднего и крупного масштаба, как предусмотрено ФГОС ВО.

По завершении изучения дисциплины студент:

- Знает принципы рефакторинга и чистого кода, понимает их важность для проектирования информационных и автоматизированных систем, особенно на предприятиях среднего и крупного масштаба.
- Умеет применять методы и техники рефакторинга для оптимизации и улучшения программного кода, производить концептуальное, функциональное и логическое проектирование информационных систем с учетом принципов чистого кода.
- Имеет навыки применения инструментальных средств рефакторинга, тестирования и контроля версий для эффективного проектирования информационных и автоматизированных систем.

Обучение по дисциплине «Рефакторинг» направлено на формирование у обучающихся следующих компетенций:

Код и наименование компетенций	Индикаторы достижения компетенции
ПК-6. Способен предотвращать потери и повреждения данных	ИПК-6.1. Знает способы и методы резервного копирования и восстановления данных ИПК-6.2. Умеет производить резервное копирование и восстановление данных ИПК-6.3. Имеет навыки применения программного обеспечения для резервного копирования и восстановления данных

2 Место дисциплины в структуре образовательной программы

Дисциплина относится к категории элективных дисциплин блока Б1 «Дисциплины (модули)».

Дисциплина взаимосвязана логически и содержательно-методически со следующими дисциплинами и практиками ОПОП:

- Базы данных;
- Объектно-ориентированное программирование;
- Тестирование программного обеспечения;
- Шаблоны проектирования;
- Анализ программного кода.

3 Структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 3 зачетные единицы, т.е. 108 академических часов.

3.1 Виды учебной работы и трудоемкость (по формам обучения)

3.1.1 Очная форма обучения

№ п/п	Вид учебной работы	Количество часов	Семестр
1	**Аудиторные занятия**	**54**	7
	В том числе:		
1.1	Лекции	18	
1.2	Семинарские/практические занятия	-	
1.3	Лабораторные занятия	36	
2	**Самостоятельная работа**	**54**	7
3	**Промежуточная аттестация**		7
	Зачет/экзамен/диф.зачет	**зачет**	зачет
	Итого:	**108**	

3.2 Тематический план изучения дисциплины (по формам обучения)

3.2.1 Очная форма обучения

№	Разделы/темы дисциплины	Трудоемкость, час					
		Всего	Аудиторная работа				
п/п			Лекции	Семинарские/ практические занятия	Лабораторные занятия	Практическая подготовка	Самостоятельная работа
1	Введение в рефакторинг	10	2				6
2	Принципы чистого кода	10	2				6

3	Запахи кода и их идентификация	14	2		6		6
4	Техники рефакторинга	14	2		6		6
5	Рефакторинг функций и методов	14	2		6		6
6	Рефакторинг классов и модулей	14	2		6		6
7	Рефакторинг систем данных	12	2		4		6
8	Тестирование и рефакторинг	12	2		4		6
9	Рефакторинг в командной работе	12	2		4		6
Итого		**108**	**18**		**36**		**54**

3.3 Содержание дисциплины

3.3.1 Очная форма обучения

1. **Введение в рефакторинг**: Основные понятия и цели рефакторинга, акцент на сохранении поведения программы при изменении её структуры, а также основные причины и мотивация к проведению рефакторинга.
2. **Принципы чистого кода**: Обсуждение концепции "чистого кода", его важности для эффективности программирования, изучение основных принципов SOLID и обоснование значения читаемости и понятности кода.
3. **Запахи кода и их идентификация**: Введение в "запахи кода" как индикаторы потребности в рефакторинге, методы их распознавания и классификация, а также анализ причин их появления.
4. **Техники рефакторинга**: Осмотр основных этапов рефакторинга, введение в разнообразные техники и методы улучшения кода с использованием шаблонов рефакторинга.
5. **Рефакторинг функций и методов**: Анализ функций и методов с целью их оптимизации, рассмотрение методов разделения и улучшения структуры функций, а также методы борьбы с дублированием кода.
6. **Рефакторинг классов и модулей**: Обзор принципов работы с классами и модулями, рефакторинг иерархии наследования, а также введение в композицию и агрегацию как альтернативные методы структурирования кода.
7. **Рефакторинг систем данных**: Изучение принципов рефакторинга в контексте систем данных, оптимизация и нормализация схем данных, а также методы эффективной работы с базами данных.
8. **Тестирование и рефакторинг**: Важность тестирования в процессе рефакторинга, введение в тесты на регресс и методы обеспечения качества кода через тестовое покрытие.
9. **Рефакторинг в командной работе**: Рассмотрение особенностей проведения рефакторинга в командном взаимодействии, введение в системы контроля версий и методы эффективной коммуникации при изменении структуры кода.

3.3.2 Заочная форма обучения

1. **Введение в рефакторинг**: Основные понятия и цели рефакторинга, акцент на сохранении поведения программы при изменении её структуры, а также основные причины и мотивация к проведению рефакторинга.
2. **Принципы чистого кода**: Обсуждение концепции "чистого кода", его важности для эффективности программирования, изучение основных принципов SOLID и обоснование значения читаемости и понятности кода.

3. **Запахи кода и их идентификация**: Введение в "запахи кода" как индикаторы потребности в рефакторинге, методы их распознавания и классификация, а также анализ причин их появления.
4. **Техники рефакторинга**: Осмотр основных этапов рефакторинга, введение в разнообразные техники и методы улучшения кода с использованием шаблонов рефакторинга.
5. **Рефакторинг функций и методов**: Анализ функций и методов с целью их оптимизации, рассмотрение методов разделения и улучшения структуры функций, а также методы борьбы с дублированием кода.
6. **Рефакторинг классов и модулей**: Обзор принципов работы с классами и модулями, рефакторинг иерархии наследования, а также введение в композицию и агрегацию как альтернативные методы структурирования кода.
7. **Рефакторинг систем данных**: Изучение принципов рефакторинга в контексте систем данных, оптимизация и нормализация схем данных, а также методы эффективной работы с базами данных.
8. **Тестирование и рефакторинг**: Важность тестирования в процессе рефакторинга, введение в тесты на регресс и методы обеспечения качества кода через тестовое покрытие.
9. **Рефакторинг в командной работе**: Рассмотрение особенностей проведения рефакторинга в командном взаимодействии, введение в системы контроля версий и методы эффективной коммуникации при изменении структуры кода.

3.4 Тематика семинарских/практических и лабораторных занятий

3.4.1 Очная форма обучения

1. **Анализ кода на наличие "запахов" и выбор стратегии рефакторинга:**
   Студенты знакомятся с различными "запахами" кода, учатся их определять и анализировать. После идентификации проблемных мест в коде студенты разрабатывают стратегии рефакторинга, направленные на устранение обнаруженных проблем.
2. **Применение основных техник рефакторинга на практике:**
   На практических примерах студенты применяют базовые техники и шаблоны рефакторинга, обучаясь эффективному и безопасному переосмыслению структуры кода.
3. **Рефакторинг функций и методов: улучшение структуры и читаемости кода:**
   Сосредоточиваясь на функциях и методах, студенты улучшают их структуру, разделяют ответственности и повышают уровень читаемости и понимания кода.
4. **Работа с классами: рефакторинг иерархии и отношений между объектами:**
   Студенты работают над классами и объектами, оптимизируя иерархии наследования, улучшая отношения между классами и применяя принципы SOLID в практической работе.
5. **Рефакторинг системы данных: оптимизация структуры и запросов:**
   На практических заданиях студенты изучают основы рефакторинга в контексте работы с данными, оптимизируют структуры данных и запросы для повышения производительности и читаемости кода.
6. **Тестирование до и после рефакторинга: гарантирование сохранения функционала:**
   Лабораторная работа акцентирует внимание на важности тестирования в процессе рефакторинга. Студенты учатся создавать тесты, гарантирующие сохранение функционала системы после изменений.
7. **Командная работа над рефакторингом: использование систем контроля версий:**

Студенты практикуются в командной работе над кодом, используя системы контроля версий для безопасного и эффективного рефакторинга, а также учатся решать конфликты и обеспечивать качественное взаимодействие в команде.

3.4.2 Заочная форма обучения

1. **Анализ кода на наличие "запахов" и выбор стратегии рефакторинга:**
   Студенты знакомятся с различными "запахами" кода, учатся их определять и анализировать. После идентификации проблемных мест в коде студенты разрабатывают стратегии рефакторинга, направленные на устранение обнаруженных проблем.
2. **Применение основных техник рефакторинга на практике:**
   На практических примерах студенты применяют базовые техники и шаблоны рефакторинга, обучаясь эффективному и безопасному переосмыслению структуры кода.
3. **Рефакторинг функций и методов: улучшение структуры и читаемости кода:**
   Сосредоточиваясь на функциях и методах, студенты улучшают их структуру, разделяют ответственности и повышают уровень читаемости и понимания кода.
4. **Работа с классами: рефакторинг иерархии и отношений между объектами:**
   Студенты работают над классами и объектами, оптимизируя иерархии наследования, улучшая отношения между классами и применяя принципы SOLID в практической работе.
5. **Рефакторинг системы данных: оптимизация структуры и запросов:**
   На практических заданиях студенты изучают основы рефакторинга в контексте работы сданными, оптимизируют структуры данных и запросы для повышения производительности и читаемости кода.
6. **Тестирование до и после рефакторинга: гарантирование сохранения функционала:**
   Лабораторная работа акцентирует внимание на важности тестирования в процессе рефакторинга. Студенты учатся создавать тесты, гарантирующие сохранение функционала системы после изменений.
7. **Командная работа над рефакторингом: использование систем контроля версий:**
   Студенты практикуются в командной работе над кодом, используя системы контроля версий для безопасного и эффективного рефакторинга, а также учатся решать конфликты и обеспечивать качественное взаимодействие в команде.

3.5 Тематика курсовых проектов (курсовых работ)

Курсовой проект дисциплиной не предусмотрен.

4 Учебно-методическое и информационное обеспечение

4.1 Нормативные документы и ГОСТы

1. Федеральный закон от 29 декабря 2012 года No 273-ФЗ «Об образовании в Российской Федерации» (с изменениями и дополнениями);

2. Федеральный государственный образовательный стандарт высшего образования - бакалавриат по направлению подготовки 09.03.02 Информационные системы и технологии, утвержденный Приказом Министерства образования и науки РФ от 19 сентября 2017 г. N 929 "Об утверждении федерального... Редакция с изменениями N 1456 от 26.11.2020;

3. Приказ Министерства образования и науки РФ от 05 апреля 2017 г. No 301 «Об утверждении Порядка организации и осуществления образовательной деятельности по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры.

4.2 Основная литература

1. Згуральская, Е. Н. Технологии программирования : учебное пособие / Е. Н. Згуральская. — Ульяновск : УлГТУ, 2020. — 71 с. — ISBN 978-5-9795-1995-1. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/165011 (дата обращения: 27.09.2022). — Режим доступа: для авториз. пользователей.

2. Калентьев, А. А. Новые технологии в программировании : учебное пособие / А. А. Калентьев. — Москва : ТУСУР, 2014. — 176 с. — ISBN 978-5-4332-0185-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/110361 (дата обращения: 27.09.2022). — Режим доступа: для авториз. пользователей.

3. Остроух, А. В. Интеллектуальные информационные системы и технологии / А. В. Остроух, А. Б. Николаев. — 3-е изд., стер. — Санкт-Петербург : Лань, 2022. — 308 с. — ISBN 978-5-507-48511-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/354536 (дата обращения: 27.09.2022). — Режим доступа: для авториз. пользователей.

4.3 Дополнительная литература

1. Мартин Роберт С. Чистый код. Создание, анализ и рефакторинг. – Питер, 2020 – 464 с.

2. Рефакторинг // Refactoring.Guru URL: https://refactoring.guru/ru/refactoring (дата обращения: 27.09.2022).

4.4 Электронные образовательные ресурсы

1. Рефакторинг. LMS Московского политеха. ЭОР разрабатывается.

4.5 Лицензионное и свободно распространяемое программное обеспечение

1. VS Code
2. SourceTree

4.6 Современные профессиональные базы данных и информационные справочные системы

1. https://urait.ru/
2. https://www.iprbookshop.ru/
3. https://e.lanbook.com/

5 Материально-техническое обеспечение

Лабораторные работы и самостоятельная работа студентов должны проводиться в специализированной аудитории, оснащенной современной оргтехникой и персональными компьютерами с программным обеспечением в соответствии с тематикой изучаемого материала. Число рабочих мест в аудитории должно быть достаточным для обеспечения индивидуальной работы студентов. Рабочее место преподавателя должно быть оснащены современным компьютером с подключенным к нему проектором на настенный экран, или иным аналогичным по функциональному назначению оборудованием.

6 Методические рекомендации

6.1 Методические рекомендации для преподавателя по организации обучения

1. При подготовке к занятиям следует предварительно проработать материал занятия, предусмотрев его подачу точно в отведенное для этого время занятия. Следует подготовить необходимые материалы – теоретические сведения, задачи и др. При проведении занятия следует контролировать подачу материала и решение заданий с учетом учебного времени, отведенного для занятия.

2. При проверке работ и отчетов следует учитывать не только правильность выполнения заданий, но и оптимальность выбранных методов решения, правильность выполнения всех его шагов.

6.2 Методические указания для обучающихся по освоению дисциплины

В процессе самостоятельной работы студенты закрепляют и углубляют знания, полученные во время аудиторных занятий, дорабатывают конспекты и записи, готовятся к промежуточной аттестации, а также самостоятельно изучают отдельные темы учебной программы.

На занятиях студентов, в том числе предполагающих практическую деятельность, осуществляется закрепление полученных, в том числе и в процессе самостоятельной работы, знаний. Особое внимание обращается на развитие умений и навыков установления связи положений теории с профессиональной деятельностью будущего специалиста.

Самостоятельная работа осуществляется индивидуально. Контроль самостоятельной работы организуется в двух формах:

- самоконтроль и самооценка студента;
- контроль со стороны преподавателей (текущий и промежуточный).

Текущий контроль осуществляется на аудиторных занятиях, промежуточный контроль осуществляется на зачете в письменной (устной) форме.

Критериями оценки результатов самостоятельной работы студента являются:

- уровень освоения студентом учебного материала;
- умения студента использовать теоретические знания при выполнении практических задач;
- сформированность компетенций;
- оформление материала в соответствии с требованиями.

7 Фонд оценочных средств

7.1 Методы контроля и оценивания результатов обучения

В процессе обучения используются следующие оценочные формы самостоятельной работы студентов, оценочные средства текущего контроля успеваемости и промежуточных аттестаций:

- Выполнение лабораторных работ
- Промежуточное тестирование (посредством изучения теоретических материалов в системе LMS)
- Итоговое тестирование

7.2 Шкала и критерии оценивания результатов обучения

Итоговая оценка по дисциплине рассчитывается как среднее взвешенное всех оценок в соответствующем курсе LMS Московского политеха с применением весовых коэффициентов, представленных ниже:

- Лабораторные работы → 0,8

- Итоговое тестирование → 0,05
- Ознакомление с теорией → 0,15

Оценка за каждую лабораторную работу выставляется исходя из фактического выполнения всех поставленных задач с учётом сроков исполнения: за каждую 1 неделю просрочки задания из оценки вычитается 10 баллов.

Для получения положительной экзаменационной оценки студенту необходимо набрать всего минимально 55 баллов по дисциплине и завершить итоговый тест с результатом не менее 55%.

Шкала оценивания	Диапазон баллов	Описание
Не зачтено	0-54	Не достигнуто пороговое значение хотя бы для одного уровня формируемых на момент проведения аттестации компетенций. Не выполнен один или более видов учебной работы, предусмотренных учебным планом. Студент демонстрирует неполное соответствие знаний, умений, навыков приведенным в таблицах показателей, допускаются значительные ошибки, проявляется отсутствие знаний, умений, навыков по ряду показателей, студент испытывает значительные затруднения при оперировании знаниями и умениями при их переносе на новые ситуации.
Зачтено	55-69	Среднее значение для всех формируемых на момент проведения аттестации уровней компетенций – 3. Выполнены все виды учебной работы, предусмотренные учебным планом. Студент демонстрирует соответствие знаний, в котором освещена основная, наиболее важная часть материала, но при этом допущена одна значительная ошибка или неточность.
	70-84	Среднее значение для всех формируемых на момент проведения аттестации уровней компетенций – 4. Выполнены все виды учебной работы, предусмотренные учебным планом. Студент демонстрирует неполное, правильное соответствие знаний, умений, навыков приведенным в таблицах показателей, либо если при этом были допущены 2-3 несущественные ошибки.
	85-100	Среднее значение для всех формируемых на момент проведения аттестации уровней компетенций – 5. Выполнены все виды учебной работы, предусмотренные учебным планом. Студент демонстрирует соответствие знаний, умений, навыков приведенным в таблицах показателей, оперирует приобретенными знаниями, умениями, навыками, применяет их в ситуациях повышенной сложности. При этом могут быть допущены незначительные ошибки, неточности, затруднения при аналитических операциях, переносе знаний и умений на новые, нестандартные ситуации.

7.3 Оценочные средства

7.3.1 Вопросы к зачету

1. Что такое рефакторинг и каковы его основные цели?
2. В чем разница между рефакторингом и переписыванием кода?
3. Какие основные принципы лежат в основе "чистого кода"?
4. Назовите и опишите пять "запахов" кода.
5. Что такое принципы SOLID и как они связаны с рефакторингом?
6. Какие техники рефакторинга вы знаете?
7. Почему тестирование так важно в процессе рефакторинга?
8. Какова роль систем контроля версий в рефакторинге?
9. Как оптимизировать структуры данных в процессе рефакторинга?
10. В чем заключается рефакторинг функций и методов для улучшения их читаемости?
11. Почему "запахи" кода считаются индикаторами потребности в рефакторинге?
12. Какие инструменты могут помочь в процессе рефакторинга?
13. Что такое кодовая база и как ее состояние может влиять на необходимость рефакторинга?
14. Каковы основные принципы командной работы над рефакторингом?
15. Что такое "технический долг", и как рефакторинг помогает в его управлении?
16. Как можно убедиться, что рефакторинг не повлиял на функциональность системы?
17. Что такое непрерывная интеграция и как она связана с рефакторингом?
18. Какие могут быть причины для рефакторинга конкретного участка кода?
19. В чем заключается рефакторинг иерархий и отношений между объектами?
20. Каким образом можно минимизировать риски, связанные с рефакторингом в крупных проектах?