МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«КОСТРОМСКАЯ ГОСУДАРСТВЕННАЯ СЕЛЬСКОХОЗЯЙСТВЕННАЯ АКАДЕМИЯ»

Утверждаю:
И.о. декана электроэнергетического факультета

Николай Александрович Климов
Подписано цифровой подписью: Николай Александрович Климов
Дата: 2024.09.11 16:10:01 +03'00'
/Климов Н.А./
11 сентября 2024 года

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по дисциплине

РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ

Специальность 09.02.07 Информационные системы и программирование

Квалификация выпускника программист

Форма обучения очная

Срок освоения ППССЗ 3 года 10 месяцев

На базе основного общего образования

Караваево 2024

Фонд оценочных средств предназначен для оценивания сформированности компетенций по дисциплине «Разработка программных модулей»

Разработчик:
Преподаватель А.А. Лобачев Андрей Александрович Лобачев
Подписано цифровой подписью: Андрей Александрович Лобачев
Дата: 2024.09.05 14:03:45 +03'00'

Утвержден на заседании кафедры СПО-Тракторы и автомобили, протокол № 1 от 05.09.2024

Заведующий кафедрой А.М. Молодов Александр Михайлович Молодов
Подписано цифровой подписью: Александр Михайлович Молодов
Дата: 2024.09.05 14:31:42 +03'00'

Согласовано:
Председатель методической комиссии электроэнергетического факультета

А.С. Яблоков Алексей Сергеевич Яблоков
Подписано цифровой подписью: Алексей Сергеевич Яблоков
Дата: 2024.09.10 15:10:46 +03'00'
протокол № 7 от 10.09.2024

Результаты освоения дисциплины

«Разработка мобильных приложений»

ППССЗ (СПО) по специальности:

09.02.07 Информационные системы и программирование

Коды компетенций по ФГОС	Компетенции	Результат освоения
Общие компетенции		
ОК 01	Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам	**Знать** способы решения задач профессиональной деятельности применительно к различным контекстам. **Уметь** находить решения задач профессиональной деятельности применительно к различным контекстам проводить геометрические измерения, читать информацию, представленную в виде таблиц, графиков, схем. **Владеть** навыками выбора способа решения задач профессиональной деятельности и приемами геометрических измерений, чтения информации, представленную в виде таблиц, графиков, схем.
ОК 02	Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности	**Знать** формат оформления результатов поиска информации, порядок применения современных средств и устройств информатизации, как применять программное обеспечение в профессиональной деятельности, в том числе с использованием цифровых средств. **Уметь** оформлять результаты поиска, применять средства информационных технологий для решения профессиональных задач; планировать процесс поиска; структурировать получаемую информацию; оформлять результаты поиска информации, пользоваться современными средствами поиска информатизации. **Владеть** навыками оформления результатов поиска информации; навыками планирования процесса поиска и структурирования полученной информации.
Профессиональные компетенции		
ПК 1.1	Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием	**Знать** основные этапы разработки программного обеспечения. **Уметь** осуществлять разработку кода программного модуля на языках низкого и высокого уровней; создавать программу по разработанному алгоритму как отдельный модуль. **Владеть** навыками разработки алгоритма решения поставленной задачи и реализации его средствами автоматизированного проектирования.
ПК 1.2	Разрабатывать программные модули в	**Знать** основные этапы разработки программного обеспечения основные принципы технологии структурного и объектно-ориентированного

	соответствии с техническим заданием	программирования **Уметь** осуществлять разработку кода программного модуля на языках низкого и высокого уровней, создавать программу по разработанному алгоритму как отдельный модуль. **Владеть** навыками разработки алгоритма решения поставленной задачи и реализации его средствами автоматизированного проектирования
ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств	**Знать** основные принципы отладки программных продуктов **Уметь** выполнять отладку программы на уровне модуля **Владеть** навыками использования инструментальных средств на этапе отладки программного продукта
ПК 1.5	Осуществлять рефакторинг и оптимизацию программного кода	**Знать** методы и средства рефакторинга, оптимизации и инспекции программного кода **Уметь** применять методы, средства рефакторинга, оптимизации и инспекции программного кода **Владеть** навыками анализа алгоритмов, в том числе с применением инструментальных средств
ПК 1.6	Разрабатывать модули программного обеспечения для мобильных платформ	**Знать** основные этапы разработки программного обеспечения **Уметь** оформлять документацию на программные средства **Владеть** навыками разработки мобильных приложений

Паспорт
фонда оценочных средств

Таблица 1

№ п/п	Контролируемые дидактические единицы	Контролируемые компетенции (или их части)	Наименование оценочных средств		
			Тесты, кол-во заданий	Другие оценочные средства	
				вид	кол-во заданий
1	Тема 1. Жизненный цикл ПО	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6	20	Опрос	5
2	Тема 2. Структурное программирование	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6	20	Опрос	5
3	Тема 3. Объектно-ориентированное программирование	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6	20	Опрос	5
4	Тема 4 Паттерны проектирования	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6	20	Опрос	5
5	Тема 5 Событийно-управляемое программирование	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6	20	Опрос	5
6	Тема 6 Оптимизация и рефакторинг кода	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6	20	Опрос	5

7	Тема 7 Разработка пользовательского интерфейса	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6	20	Опрос	5
8	Тема 8 Основы ADO.Net	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6	20	Опрос	5
9	Тема 9 Самостоятельная работа	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6			
10	Темы 1-9	ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6		Собеседование	20
Всего:			80		40

Методика проведения контроля по проверке базовых знаний по дисциплине «Разработка программных модулей»

Тема 1 Жизненный цикл ПО

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6

Вопросы для устного опроса

1. Опишите, как может быть реализован “V-образный” жизненный цикл разработки ПО с учетом требований к безопасности и качества? Какие этапы жизненного цикла необходимо модифицировать, чтобы обеспечить соответствие продукта стандартам безопасности и качества?
2. Сравните “каскадную” и “итеративную” модели жизненного цикла ПО. Какие преимущества и недостатки имеет каждая модель? В каких ситуациях предпочтительнее использовать каскадную модель, а в каких – итеративную?
3. Как реализовать “инкрементную” разработку в контексте жизненного цикла ПО? Какие этапы инкрементной разработки выделяются? Как обеспечить согласованность между инкрементами и как управлять рисками, связанными с инкрементной разработкой?
4. Опишите, как можно интегрировать методологию Agile в жизненный цикл ПО. Какие этапы Agile-разработки можно использовать в контексте жизненного цикла ПО? Какие преимущества и недостатки имеет интеграция Agile в традиционный жизненный цикл?
5. Как можно применить концепции DevOps в жизненном цикле ПО? Какие этапы жизненного цикла необходимо модифицировать для включения DevOps-практик? Какие преимущества и недостатки имеет DevOps в контексте разработки и поддержки ПО?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания

Выберите один правильный вариант ответа и нажмите кнопку «Далее»

Этап, на котором определяются требования к ПО, называется:

Разработка

Поддержка

Тестирование

+Сбор и анализ требований

Методология, предусматривающая последовательную реализацию этапов, называется:
Agile
+V-модель
Scrum
Lean
Главной целью этапа тестирования является:
Подготовка документации
Определение архитектуры
+Проверка работоспособности и соответствия требованиям
Постоянное изменение требований

Этап, на котором выполняется документирование всех процессов и результатов.
+Сбор требований
Обслуживание
Разработка
Администрация

Модель, позволяющая параллельно вести разработку и тестирование, называется:
Спиральная модель
Водопадная модель
Иерархическая модель
+V-модель

Подход, основанный на итерациях и инкрементах, называется:
Водопадная модель
+Agile
TDD
Прототипирование

Основной аспект управления проектом на этапе эксплуатации называется:
Рефакторинг кода
+Устранение неполадок и улучшение функционала
Анализ рынка
Моделирование данных

Процесс, связанный с исправлением ошибок и добавлением нового функционала, называется:
Дизайн
Миграция
+Поддержка
Оценка

Тип документации, составляемый на этапе проектирования, называется:
+Техническое задание
Отчет о тестировании
Код
Пользовательская документация

Метод, позволяющий выявить недостатки на ранних этапах разработки, называется:
Итеративное тестирование
Рефакторинг

+Код-ревью
Релиз

Основная цель стадии проектирования системы называется:
Создание конечной версии ПО
+Определение структуры и архитектуры системы
Установка ПО
Обучение пользователей

Процесс, который производится после завершения разработки, называется:
Жизненный цикл
Поддержка
Тестирование
+Деплоймент

Один из методов управления изменениями в требованиях называется:
Адаптация
Реорганизация
+Контроль версии
Прототипирование

Основная цель этапа сбора требований является:
Соответствие кода стандартам
+Определение потребностей и ожиданий пользователей
Создание Test Case
Выявление ошибок

Распространенная модель разработки ПО, использующая итерации, называется:
Водопадная модель
+Спиральная модель
Прямолинейная модель
Долговременная модель

Ответственность за поддержку ПО чаще всего лежит на:
Пользователях
Менеджерах проектов
Команде разработки
+Команде технической поддержки

Основным требованием на этапе тестирования является:
Полное соответствие действующим законодательным актам
Высокая производительность
+Выявление и устранение ошибок
Погодные условия

Важность аспектом учета отзывов пользователей происходит на этапе:
Сбор требований
Проектирование
+Эксплуатации
Разработки

Основной трудностью в управлении жизненным циклом ПО является:

+Сформулировать требования
Наличие большого количества документации
Протестировать каждую функцию
Изменчивость требований

Постоянное обновление и улучшение системы происходит на этапе:
+Эксплуатации
Проектирования
Разработки
Деплоймента

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно выполнил 9-10 тестовых заданий.

Оценка «хорошо» выставляется обучающемуся, который: правильно выполнил 7-8 тестовых заданий.

Оценка «удовлетворительно» выставляется обучающемуся, который правильно выполнил 5-6 тестовых заданий.

Оценка «неудовлетворительно» выставляется обучающемуся, который правильно выполнил менее 4 тестовых заданий.

Тема 2. Структурное программирование

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6

Вопросы для устного опроса

1. Объясните, как принцип структурного программирования “однократный вход, однократный выход” (Single Entry, Single Exit) позволяет упростить отладку и повысить читаемость кода. Приведите пример функции с неправильной структурой и покажите, как ее можно переработать, соблюдая принцип “однократный вход, однократный выход”
2. Сравните и проанализируйте преимущества и недостатки использования рекурсивных функций в структурном программировании. В каких ситуациях рекурсия является предпочтительным подходом, а в каких – более эффективным решением будет использование итеративных алгоритмов?
3. Как можно использовать принципы структурного программирования для повышения эффективности и оптимизации кода? Приведите пример неэффективного кода и покажите, как его можно переработать с использованием принципов структурного программирования для улучшения производительности и уменьшения количества операций.
4. Объясните, как можно использовать структурное программирование для разработки и тестирования программных модулей. Какие принципы структурного программирования важны для создания модульных систем, легко тестируемых и поддерживаемых?
5. Опишите, как можно использовать графические методы (например, диаграммы Hassi-Shrader) для визуализации структуры программы и повышения ее читаемости. Приведите пример программы с графическим представлением ее структуры и покажите, как графическое представление упрощает понимание и анализ кода.

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания
Выберите один правильный вариант ответа и нажмите кнопку «Далее»

Принцип, который предполагает использование простых структур вместо сложных, называется:
Классификация
+Декомпозиция
Инкапсуляция
Абстракция

Модель, которая позволяет описывать алгоритмы и структуры данных при помощи блок-схем, называется:
UML
DFD
+Nassi-Shneiderman
ER-диаграммы

Один из основных методов структурного программирования, называется:
Использование глобальных переменных
+Избежание повторяемости кода
Переопределение функций
Императивное программирование

Принцип, согласно которому любой элемент модуля должен выполнять одну задачу, называется:
+Разделение ответственности
Кросс-объектность
Открытое изменение
Женерация

В структурном программировании четкую иерархию модулей представляет собой:
Функциональное программирование
Объектно-ориентированное программирование
+Модульное программирование
Логическое программирование

Один из ключевых аспектов структурного программирования, называется:
Использование стека для управления памятью
+Последовательное выполнение инструкций
Защита данных от пользователя
Формирование классов и объектов

Средство, позволяющее отладить алгоритм путем его визуализации, называется:
Компилятор
Редактор
+Дебаггер
Протестированное окружение

Структурный подход, который помогает избежать безусловных переходов с помощью условных операторов, называется:
Линейная структура
+Структура "выбор"
Структура "цикл"
Структура "параллель"

Способ, который позволяет разбить программу на подпрограммы, называется:
Связность
Рекурсия
+Модулирование
Абстракция

Применение структурного программирования приводит к:
Увеличению сложности кода
Зависимости от платформы
+Улучшению читаемости и поддержки кода
Необходимости постоянного рефакторинга

Основной документ, который описывает стратегию и принципы выполнения алгоритма, называется:
Техническое задание
Руководство пользователя
+Описание алгоритма
Программная документация

Метод визуального представления алгоритма, который использует различные геометрические фигуры, называется:
Текстовый метод
+Блок-схема
Диаграмма классов
Псевдокод

Язык, который позволяет создавать текстовое представление алгоритмов, приближенное к программному коду, называется:
HTML
+Псевдокод
SGML
Lua

Стандарт для документирования программного обеспечения, который гарантирует единообразие, называется:
ISO 9001
+IEEE 829
GOST P 51226-98
ISO 14001

Инструмент, предназначенный для автоматизированного составления документации от кода, называется:
Git
+Doxygen
IDE
Visual Studio

Класс алгоритмов, для которого существуют решения, проверяемые за полиномиальное время, называется:
NP-полные
+P
NP-трудные
EXPTIME

Отличительная особенность Good Documentation Practices (GDP):
Описание кода большими абзацами
Неправильные примеры
+Четкая структура и доступность информации
Игнорирование обновлений документации

Блоки, которые используются в блок-схемах для обозначения начала и конца алгоритма, называется:
Прямоугольники
+Овалы
Ромбы
Параллелограммы

Инструмент для хранения версий документации и контроля изменений, называется:
Wiki
Носитель информации
Офисный пакет
+Система управления версиями

Элемент, который должен быть частью документации для настройки и развертывания программного обеспечения, называется:
+Инструкции по компиляции
Список переменных
Ссылки на источники
Содержимое комментариев в коде

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно выполнил 9-10 тестовых заданий.

Оценка «хорошо» выставляется обучающемуся, который: правильно выполнил 7-8 тестовых заданий.

Оценка «удовлетворительно» выставляется обучающемуся, который правильно выполнил 5-6 тестовых заданий.

Оценка «неудовлетворительно» выставляется обучающемуся, который правильно выполнил менее 4 тестовых заданий.

Тема 3. Объектно-ориентированное программирование

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6

Вопросы для устного опроса

1. Объясните, как работают механизмы наследования и полиморфизма в ООП. Как они взаимодействуют друг с другом? Приведите пример решения задачи с использованием наследования и полиморфизма, который демонстрирует их преимущества с точки зрения гибкости и переиспользования кода.
2. Как реализуется инкапсуляция в ООП? Почему она важна для создания модульных и гибких систем? Какие проблемы могут возникнуть, если игнорировать принципы инкапсуляции при разработке программ?
3. Опишите, как используются абстрактные классы и интерфейсы в ООП. В чем разница между ними? Когда используется абстрактный класс, а когда – интерфейс? Приведите пример, где применение абстрактного класса или интерфейса упрощает разработку и тестирование системы.
4. Как можно применить принципы ООП для создания многопоточных приложений? Какие паттерны проектирования (например, “Producer-Consumer” или “Thread-Pool”) используются для управления потоками в контексте ООП?
5. Объясните, как можно использовать ООП для разработки GUI-приложений (Graphical User Interface). Какие концепции ООП (например, “Model-View-Controller”) используются для структурирования GUI-приложений и как они взаимодействуют друг с другом?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания
Выберите один правильный вариант ответа и нажмите кнопку «Далее»

Одним из основных принципов ООП является инкапсуляция, которая подразумевает:
+скрытие внутреннего состояния объекта и предоставление доступа только через методы
защита объектов от внешних изменений
создание объектов без методов
использование глобальных переменных

Наследование позволяет:
+создавать новые классы на основе существующих
ограничить использование других классов
внедрять только статические методы
использовать только один объект

Полиморфизм в ООП означает:
+возможность использования объектов разных классов через один интерфейс
создание объектов с одним типом данных
исключительно статическую типизацию
возможность изменять только свойства объектов

Класс в ООП является:
+шаблоном для создания объектов с общими свойствами и методами
конкретной реализацией объекта
экземпляром существующего объекта
методом для обработки данных

Объект в контексте ООП - это:
+экземпляр класса, который содержит данные и методы
только структура данных
лишь набор функций
базовая единица кода

Модификаторы доступа в классах определяют:
+видимость и доступ к членам класса
только типы данных внутри класса
возможность создания новых объектов
порядок выполнения методов

Абстракция в ООП позволяет:
+выделить важные характеристики объекта, игнорируя детали
ограничить использование методов
создавать только простые классы
сосредоточиться на реализации вместо интерфейса

Из ниже перечисленных терминов с ООП связан:
+класс
массив
оператор
функция

Композиция в ООП означает:
+использование одного или нескольких объектов внутри другого объекта
создание объектов без свойства
отсутствие взаимосвязи между классами
наследование методов от родительского класса

Что такое интерфейсы в ООП?
+контракты, которые определяют, какие методы должен реализовать класс
лишь набор глобальных функций
тип данных для хранения целых чисел
аналог классов без методов

Основное требование для перегрузки методов заключается в:
+Различии списков параметров
Именах методов
Возвращаемых типах данных
Порядке вызова методов

При перегрузке метода, отличие в возвращаемом типе данных:
+Не позволяет перегружать метод
Является основным критерием выбора метода
Подразумевает использование разных имен
Обязательно требуется для выполнения перегрузки

Процесс выбора подходящего перегруженного метода для вызова осуществляется на основе:
+Списков аргументов
Возвращаемых типов
Имени метода
Количества методов в классе

Если перегруженный метод принимает параметры одного типа и различается только количеством аргументов, это называется:
+Перегрузка по количеству параметров
Перегрузка по типу параметров
Переопределение метода
Параметризация метода

Перегрузка методов может быть выполнена в:
+Одном классе
В нескольких классах, но с одним именем
Только в статических методах
Только в абстрактных классах

Структуры в C# используются для:
+Группировки связанных данных
Хранения методов и свойств
Создания динамических объектов
Обработки исключений

Делегаты в C# представляют собой:
+Тип, который может ссылаться на методы
Структуры данных для хранения значений
Классы для работы с событиями
Интерфейсы для обработки ошибок

Основной задачей регулярных выражений является:
+Поиск и манипуляция строками
Кодирование и шифрование данных
Создание и редактирование файлов
Выполнение арифметических вычислений

Какой из следующих типов данных не может быть структурой в C#:
+Класс
Массив
Перечисление
Структура

Делегаты могут быть использованы для:
+Создания событий и подписки на них
Определения структуры данных
Моделирования поведения классов
Упрощения синтаксиса языков программирования

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно выполнил 9-10 тестовых заданий.

Оценка «хорошо» выставляется обучающемуся, который: правильно выполнил 7-8 тестовых заданий.

Оценка «удовлетворительно» выставляется обучающемуся, который правильно выполнил 5-6 тестовых заданий.

Оценка «неудовлетворительно» выставляется обучающемуся, который правильно выполнил менее 4 тестовых заданий.

Тема 4. Паттерны проектирования

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6

Вопросы для устного опроса

1. Объясните, как паттерн “Стратегия” (Strategy) позволяет реализовать динамический выбор алгоритма в зависимости от контекста задачи. Приведите конкретный пример применения паттерна “Стратегия” для решения задачи обработки данных с различными алгоритмами сортировки.
2. Сравните и проанализируйте паттерны “Фасад” (Facade) и “Посредник” (Mediator). В чем их основные отличия и сходства? В каких ситуациях каждый из них является более подходящим решением? Приведите примеры использования каждого паттерна в разработке программного обеспечения.
3. Как можно использовать паттерн “Наблюдатель” (Observer) для реализации механизма уведомлений в системе, где несколько объектов должны получать уведомления о событиях, происходящих в других объектах? Приведите пример использования паттерна “Наблюдатель” в контексте GUI-приложения.
4. Объясните, как паттерн “Шаблонный метод” (Template Method) позволяет реализовать алгоритмы с вариациями в конкретных шагах. Приведите пример использования паттерна “Шаблонный метод” для разработки алгоритма сортировки с разными стратегиями сравнения элементов.
5. Как можно использовать паттерн “Прокси” (Proxy) для контроля доступа к объекту или для реализации распределенного доступа к данным? Приведите пример применения паттерна “Прокси” в контексте работы с базой данных или сетевым соединением.

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания
Выберите один правильный вариант ответа и нажмите кнопку «Далее»

Назначение паттернов в программировании:
+ Решение распространенных задач проектирования
Снижение производительности программ
Упрощение механизма работы с базами данных
Создание сложной архитектуры приложений

Порождающие паттерны ориентированы на:
+ Создание объектов
Изменение состояния объектов
Их организацию в структуре
Обработку событий

Пример порождающего паттерна:
+ Singleton
Adapter
Observer
Composite

Шаблон "Фабрика" используется для:
+ Создания объектов через интерфейс
Изменения поведения объектов
Хранения объектов в коллекции
Обработки событий между объектами

Структурные паттерны помогают:
+ Упрощать организацию классов и объектов
Программировать в многоязычных средах
Улучшать читаемость кода
Увеличивать производительность алгоритмов

Шаблон "Адаптер" позволяет:
+ Привести интерфейсы к совместимому виду
Создавать новые объекты с измененными свойствами
Хранить состояния объектов
Моделировать сложные объекты

Паттерн "Обсервер" предназначен для:
+ Реализации подписки на события
Записи данных в базу данных
Синхронизации потоков
Оптимизации памяти

Шаблон "Декоратор" используется для:
+ Добавления ответственности объектам динамически
Изменения внутреннего состояния объекта
Создания объектов с фиксированной функциональностью
Упрощения работы с потоками

Паттерн "Стратегия" позволяет:
+ Изменять алгоритмы выполнения во время работы
Зафиксировать алгоритм на этапе компиляции
Упрощать реализацию многопоточности
Увеличивать скорость выполнения программ

Применение паттерна "Команда" предполагает:
+ Инкапсуляцию запросов в виде объектов
Обработку исключений в потоках
Создание графического интерфейса
Рефакторинг кода

Какой из паттернов не является порождающим:
+ Proxy
Builder
Prototype
Factory Method

Основной целью поведенческих паттернов является:
+ Определение алгоритмов и управление их взаимодействием
Оптимизация использования памяти
Упрощение создания объектов
Изменение структуры классов

Паттерн "Фасад" позволяет:
+ Упростить взаимодействие с подсистемой
Создавать новые объекты с уникальными свойствами
Работать с многопоточностью
Обрабатывать ошибки

Паттерн "Состояние" используется для:
+ Изменения поведения объекта в зависимости от его состояния
Создания новых классов на основе существующих
Оптимизации работы с памятью
Синхронизации данных между потоками

Шаблон "Мост" помогает:
+ Изолировать абстракцию от её реализации
Создавать кроссплатформенные приложения
Работать с базами данных
Упрощать работу с сетью

Паттерн "Итератор" предназначен для:
+ Упрощения доступа к элементам коллекции
Управления состоянием разрабатываемого объекта
Создания многопоточных приложений
Оптимизации алгоритмов выполнения

Паттерн "Ленивое создание" помогает:
+ Создавать объекты по мере необходимости
Записывать данные в базу данных
Оптимизировать взаимодействие с сетевыми сервисами
Создавать сложные интерфейсы

Шаблон "Цепочка ответственности" предоставляет:
+ Передачу запроса по цепочке обработчиков
Упрощения работы с пользовательскими интерфейсами
Создание сложных объектов
Устойчивость к сбоям

Паттерн "Композиция" применяется для:
+ Создания сложных объектов на основе простых
Инкапсуляции запросов
Упрощения алгоритмов
Создания многопоточных X объектов

Паттерн "Кратка" используется для:
+ Упрощения работы с большими объемами данных
Оптимизации сетевых соединений
Создания тестов для методов
Упрощения обработки ошибок

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно выполнил 9-10 тестовых заданий.

Оценка «хорошо» выставляется обучающемуся, который: правильно выполнил 7-8 тестовых заданий.

Оценка «удовлетворительно» выставляется обучающемуся, который правильно выполнил 5-6 тестовых заданий.

Оценка «неудовлетворительно» выставляется обучающемуся, который правильно выполнил менее 4 тестовых заданий.

Тема 5. Событийно-управляемое программирование

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6

Вопросы для устного опроса

1. Объясните принцип работы “цикла событий” (event loop) в событийно-управляемом программировании. Как обрабатываются события в цикле событий? Какую роль играют “обработчики событий” (event handlers) в этом процессе? Приведите пример, иллюстрирующий работу цикла событий и обработчиков событий в контексте GUI-приложения.
2. Как можно использовать “событийные модели” (event models) для организации взаимодействия между объектами в событийно-управляемом программировании? Сравните и проанализируйте различные типы событийных моделей (например, “публикация/подписка”, “событие/обработчик”). Приведите пример использования конкретной событийной модели в разработке программы.
3. Объясните, как можно использовать “событийные цепочки” (event chains) для реализации механизма пропагации событий в иерархической структуре объектов. Приведите пример, как событие, происходящее в “дочернем” объекте, может быть передано “родительскому” объекту с использованием событийной цепочки.
4. Какие проблемы могут возникнуть при использовании событийно-управляемого программирования? Как можно минимизировать риски, связанные с использованием событийно-управляемого программирования, например, проблем с “гонка данных” (race conditions) или “тупиков” (deadlocks)?
5. Как можно использовать “событийные механизмы” (event mechanisms) для реализации асинхронного взаимодействия между объектами в разных потоках или процессах? Приведите пример, иллюстрирующий асинхронное взаимодействие с использованием событийных механизмов.

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания
Выберите один правильный вариант ответа и нажмите кнопку «Далее»

Событийно-управляемое программирование основано на:
+ Реакции на события и обработке событий
Синхронном выполнении кода
Операциях с базами данных
Украшении интерфейса

Элементы управления в графических интерфейсах предназначены для:
+ Взаимодействия пользователя с приложением
Оптимизации работы сервера
Параллельной обработки данных
Шифрования хранилища данных

Диалоговые окна предназначены для:
+ Взаимодействия с пользователем
Загруженности графического интерфейса
Хранения конфиденциальной информации
Оптимизации работы с сетью

Основные компоненты событийно-ориентированного подхода:
+ События, обработчики и источники событий
Элементы управления и пользовательский ввод
Модели и представления данных
Синхронизация потоков и использование защищенных ресурсов

Обработчик события выполняется:
+ При наступлении соответствующего события
При каждом запуске приложения
Параллельно с основным потоком
В фоновом режиме без взаимодействия с пользователем

Основные этапы работы с событиями:
+ Подписка на событие, обработка события и отписка
Генерация событий, их фильтрация и отображение
Загрузка интерфейса, взаимодействие с базой данных и сохранение
Создание объектов, тестирование и развертывание

В каком сценарии используются диалоговые окна?
+ Для вывода сообщений или запроса информации у пользователя
Для определения структуры базы данных
Для работы с файловой системой
Для оптимизации сетевых соединений

Элемент управления предназначен для ввода текста, называется:
+ Текстовое поле
Кнопка
CheckBox
ComboBox

Модель событийно-управляемого программирования включает:
+ Подписчиков и опубликованные события
Временные метки и отклики
Распараллеливание задач и асинхронные операции
Графические элементы и файлы

Наиболее распространенным тип события называется:
+ Клик мыши
Изменение текста
Загрузка окна
Закрытие приложения

Событие "MouseEnter" возникает при:
+наведении курсора на элемент управления
нажатии на элемент управления
изменении состояния элемента
перемещении окна приложения

Подход, используемый для реактивного программирования, называется:
+ Использование потоков событий
Создание классов и объектов
Программирование на основе синхронных вызовов
Использование глобальных переменных

Элемент управления, позволяющий выбирать один из множества вариантов, называется:
+ Радиокнопка
Текстовое поле
Список
Выбор цвета

Обработчики событий можно назначать через:
+ Атрибуты элемента управления или программный код
SQL-запросы и процедуры
Объекты коллекций и массивов
Графические интерфейсы пользователя

Для отображения уведомления о результате выполнения операции используется:
+ Сообщение или модальное окно
Кнопка
Текстовый ввод
Список элементов

Для обработки нажатия кнопки применяется:
+ Событие Click
Событие Change
Событие Load
Событие KeyPress

Метод, используемый для отображения диалогового окна с вводом данных, называется:
+ ShowDialog()
Display()
Run()
Load()

Для обработки событий клавиатуры используется:
+ Событие KeyDown
Событие MouseEnter
Событие Click
Событие Resize

Главная задача управление событиями в приложении — это:
+ Обеспечение интерактивности
Создание графического интерфейса
Оптимизация работы с памятью
Синхронизация данных

При создании пользовательского интерфейса важно учитывать:
+ Обработку событий и доступность элементов управления
Только визуальное оформление
Объем памяти, занимаемый приложением
Скорость работы с сетью

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно выполнил 9-10 тестовых заданий.

Оценка «хорошо» выставляется обучающемуся, который: правильно выполнил 7-8 тестовых заданий.

Оценка «удовлетворительно» выставляется обучающемуся, который правильно выполнил 5-6 тестовых заданий.

Оценка «неудовлетворительно» выставляется обучающемуся, который правильно выполнил менее 4 тестовых заданий.

Тема 6. Оптимизация и рефакторинг кода

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6

Вопросы для устного опроса

1. Объясните, как профилирование кода может помочь определить “узкие места” (bottlenecks) в программе и выбрать наиболее эффективные методы оптимизации. Приведите пример, где профилирование помогло улучшить производительность программы, и опишите методы оптимизации, которые были применены.
2. Сравните и проанализируйте различные методы оптимизации кода (например, “инлайнинг”, “развертывание циклов”, “устранение избыточных вычислений”). Какие преимущества и недостатки имеет каждый метод? В каких ситуациях каждый из методов является более эффективным решением?
3. Как рефакторинг кода может помочь улучшить структуру программы и сделать ее более читаемой и поддерживаемой? Приведите пример рефакторинга кода (например, “извлечение метода”, “перемещение класса”), который улучшил структуру программы и сделал ее более гибкой.
4. Как можно применить принципы “DRY” (Don’t Repeat Yourself) и “KISS” (Keep It Simple, Stupid) при разработке и оптимизации кода? Приведите пример кода, который нарушает эти принципы, и покажите, как его можно переработать с учетом “DRY” и “KISS”.
5. Опишите методы и инструменты, которые можно использовать для автоматизации процесса оптимизации и рефакторинга кода. Какие инструменты и технологии (например, “статический анализ кода”, “автоматизированное тестирование”) могут помочь в этом процессе?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания
Выберите один правильный вариант ответа и нажмите кнопку «Далее»

Оптимизация программного кода направлена на:
+ Повышение производительности и уменьшение затрат ресурсов
Улучшение визуального оформления
Упрощение архитектуры системы
Увеличение количества строк кода

Рефакторинг кода подразумевает:
+ Изменение внутренней структуры без изменения внешнего поведения
Полное переписывание приложения
Изменение функциональности системы
Удаление всех комментариев из кода

Одной из причин для оптимизации кода является:
+ Сложные и долгие операции
Упрощение имен переменных
Повышенное использование памяти
Избыток комментариев

В процессе рефакторинга следует избегать:
+ Вмешательства в логику или функционал программного кода
Упрощения кода
Переименования переменных для большей понятности
Удаления дублирующихся фрагментов кода

Оптимизация кода может включать в себя:
+ Упрощение алгоритмов и использование кэширования
Удаление всех циклов и условных операторов
Добавление максимально возможного количества функций
Использование сложных структур данных без необходимости

Главной целью рефакторинга является:
+ Повышение читаемости и поддерживаемости кода
Уменьшение количества строк кода
Ускорение его выполнения
Увеличение объема хранимых данных

Метод оптимизации, предполагающий анализ алгоритмов по временной сложности, называется:
+ Оптимизация по алгоритмической сложности
Оптимизация по объему кода
Оптимизация по количеству комментариев
Оптимизация по организации файлов

Принципы "Четырех правил рефакторинга" включают:
+ Коды должны быть протестированы после каждой итерации
Оптимизация кода до его полной реализации
Добавление нового функционала без рефакторинга
Упрощение процесса сборки приложения

Один из распространенных методов рефакторинга — это:
+ Извлечение метода для уменьшения дублирования
Удаление всех комментариев в коде
Комбинирование переменных в одном объекте
Увеличение количества условий в циклах

Оптимизация памяти может осуществляться через:
+ Использование более эффективных структур данных
Увеличение количества глобальных переменных
Упрощение логики программы
Удаление всех временных переменных

При рефакторинге кода важно учитывать:
+ Как изменения повлияют на существующий функционал
Только требования к новому функционалу
Как улучшить структуру без тестирования
Удаление всех неиспользуемых функций

Одним из методов оптимизации является:
+ Параллелизация вычислений
Использование только одного потока
Увеличение объема входных данных
Увеличение длины кода для упрощения понимания

Использование паттернов проектирования в рефакторинге позволяет:
+ Создавать более гибкую и поддерживаемую архитектуру
Усложнять код для большей безопасности
Минимизировать тестирование после изменений
Убрать все зависимости между модулями

Анализ производительности кода можно выполнить с помощью:
+ Профилирования
Случайного тестирования
Рефакторинга классов
Увеличения количества функций

Одной из целей рефакторинга является:
+ Улучшение тестируемости кода
Увеличение количества ошибок в коде
Создание нового интерфейса пользователя
Упрощение задач компиляции

Метод "разделяй и властвуй" применяется при:
+ Разделении больших функций на меньшие части
Объединении нескольких файлов в один
Увеличении длины переменных для понятности
Сложении функционала в одном классе

Устранение избыточности в коде может быть достигнуто путем:
+ Извлечения общих фрагментов в отдельные функции
Добавления новых фрагментов к уже существующим
Удаления всех сложных условий
Создания новых переменных для каждого использования

Эффективная оптимизация кода включает:
+ Оптимизацию алгоритмов и логики приложения
Полное игнорирование пользовательского ввода
Добавление сложных условий для обработки
Максимизацию количества классов в проекте

Ключевым аспектом рефакторинга является:
+ Поддержание функционала при изменении кода
Полное пересоздание функционала
Игнорирование тестов после изменений
Минимизация комментов для экономии места

Сложность кода может уменьшаться путем:
+ Применения принципов ясности и простоты
Увеличения вложенности и сложности
Удаления всех вспомогательных методов
Использования абстракций без явной необходимости

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно выполнил 9-10 тестовых заданий.

Оценка «хорошо» выставляется обучающемуся, который: правильно выполнил 7-8 тестовых заданий.

Оценка «удовлетворительно» выставляется обучающемуся, который правильно выполнил 5-6 тестовых заданий.

Оценка «неудовлетворительно» выставляется обучающемуся, который правильно выполнил менее 4 тестовых заданий.

Тема 7. Паттерны проектирования

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6

Вопросы для устного опроса

1. Объясните концепцию “юзабилити” (usability) в контексте разработки пользовательского интерфейса. Какие критерии используются для оценки юзабилити интерфейса? Как можно провести тестирование юзабилити и какие инструменты используются для этого?
2. Сравните и проанализируйте разные подходы к проектированию пользовательского интерфейса, например, “материальный дизайн” (Material Design), “плоский дизайн” (Flat Design), “скевоморфизм” (Skeuomorphism). В чем преимущества и недостатки каждого подхода? В каких ситуациях каждый из подходов является более подходящим решением?
3. Как применить принципы “информационной архитектуры” (information architecture) при разработке пользовательского интерфейса? Какие методы и инструменты используются для создания эффективной информационной архитектуры? Приведите пример, как правильно организованная информационная архитектура упрощает навигацию пользователя по интерфейсу.
4. Объясните, как можно использовать “паттерны проектирования пользовательского интерфейса” (UI design patterns) для создания интуитивно понятных и эффективных интерфейсов. Приведите примеры использования патернов (например, “Accordion”, “Carousel”, “Tab Bar”) в разработке веб-приложений или мобильных приложений.
5. Как можно использовать “A/B тестирование” (A/B testing) для оптимизации пользовательского интерфейса? Какие критерии используются для оценки результатов A/B тестирования? Как провести A/B тестирование и какие инструменты используются для этого?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания
Выберите один правильный вариант ответа и нажмите кнопку «Далее»

Правильный выбор шрифтов и цветов важен для:
+ Создания удобочитаемого и привлекательного интерфейса
Увеличения числа кнопок на экране
Упрощения шифрования данных
Снижения объемов трафика

Принцип "Визуального веса" подразумевает:
+ Использование элементов, которые создают визуальный баланс
Увеличение размера всех элементов интерфейса
Комбинирование текстов разных шрифтов
Удаление лишних элементов из интерфейса

Элемент, который помогает пользователю понять, где он находится в приложении, это:
+ Хлебные крошки
Слайдеры
Поп-ап окна
Логотипы

Удобство интерфейса пользователя часто измеряется с помощью:
+ Методологии юзабилити-тестирования
Количества графических элементов
Цветовой палитры
Объема текста

Избыточные элементы в интерфейсе могут привести к:
+ Увеличению когнитивной нагрузки на пользователя
Повышению скорости работы приложения
Упрощению восприятия информации
Увеличению доверия к интерфейсу

Принцип "Обратной связи" в интерфейсе включает:
+ Сообщение пользователю о результатах его действий
Отсутствие каких-либо предупреждений
Изменение графических элементов случайным образом
Упрощение логики программного обеспечения

Использование стандартных компонентов интерфейса помогает:
+ Упрощать взаимодействие пользователя с системой
Увеличивать число уникальных элементов
Избегать использования единых стилей
Разрабатывать интерфейс абсолютно с нуля

Наилучшие практики работы с формами включают:
+ Упрощение и сокращение количества полей ввода
Добавление максимально возможного числа обязательных полей
Использование сложных текстов для подсказок
Увеличение размера форм до максимума

Пользователи чаще всего ценят интерфейс за:
+ Простоту и интуитивность
Наличие сложных анимаций
Обилие графических элементов
Сложность навигации

Для повышения доступности интерфейса рекомендуется:
+ Использовать альтернативные текстовые описания для изображений
Исключать клавиатурные команды
Упрощать навигацию только для десктопных версий
Избегать использования цветовых контрастов

Информативные подсказки должны быть:
+ Краткими и лаконичными
Как можно более объемными
Сложными и понятными только специалистам
Произвольными и случайными

Проблема "засоренности" интерфейса решается путем:
+ Удаления лишних элементов и фокусирования на основном
Добавления больше информации на один экран
Увеличения размеров шрифтов и графических элементов
Искажением пропорций интерфейса

Важной частью навигации в приложении является:
+ Согласованность и предсказуемость элементов
Использование свернутого меню
Отсутствие ссылок на главные разделы
Частая смена структуры меню

При разработке мобильного интерфейса нужно учитывать:
+ Ограниченное пространство экрана
Нужно наличие одновременно всех элементов
Имеется возможность использовать любые шрифты
Доступность только для определенных устройств

При проектировании интерфейса стоит избегать:
+ Сложных архитектурных решений без необходимости
Упрощенных подходов к функционалу
Использования стандартных графических элементов
Ненужных анимаций

Мобильные интерфейсы требуют особого внимания к:
+ Элементам, которые легко нажимать
Многоуровневым меню
Большим текстовым блокам
Сложным формам для заполнения

Очевидные, интуитивно понятные действия пользователю помогают:
+ Уменьшить время на обучение и самодействие
Увеличить объемы запрашиваемого функционала
Предоставить сложные инструкции
Усложнить логику взаимодействия

Клиент-центричный подход в разработке интерфейсов включает:
+ Слушание пользователей и их обратную связь
Упрощение процессов для разработчиков
Создание интерфейса без учета пользователей
Ограничение взаимодействия с клиентами

Оптимизация загрузки страниц размещает акцент на:
+ Быстрой загрузке визуального контента
Увеличении размера изображений для детализации
Сложном коде интерфейса
Большом количестве объектов на странице

Принцип "Консистентности" в интерфейсе означает:
+ Использование одинаковых элементов и стилей по всему приложению
Постоянное изменение дизайна для привлечения внимания
Создание уникальных интерфейсов для каждой функции
Усложнение навигации для пользователей

Компоненты интерфейса, которые помогают пользователю понимать поток информации, это:
+ Визуальные индикаторы и прогресс-бары
Сложные графические иллюстрации
Звучащие сигналы уведомлений
Многоуровневые меню

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно выполнил 9-10 тестовых заданий.

Оценка «хорошо» выставляется обучающемуся, который: правильно выполнил 7-8 тестовых заданий.

Оценка «удовлетворительно» выставляется обучающемуся, который правильно выполнил 5-6 тестовых заданий.

Оценка «неудовлетворительно» выставляется обучающемуся, который правильно выполнил менее 4 тестовых заданий.

Тема 8. Основы ADO.Net

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.3 ПК 1.5 ПК 1.6

Вопросы для устного опроса

1. Объясните, как работает механизм “соединения” (connection) в ADO.NET. Какие типы соединений существуют (например, SQL Server Connection, OLE DB Connection)? Как установить соединение с базой данных и как управлять ресурсами соединения (например, открыть, закрыть, обработать ошибки)?
2. Сравните и проанализируйте преимущества и недостатки использования “DataReaders” и “DataSets” в ADO.NET. В каких ситуациях предпочтительнее использовать “DataReaders”, а в каких – “DataSets”? Приведите примеры использования каждого объекта в разработке приложения.
3. Опишите процесс использования “команд” (commands) в ADO.NET для выполнения SQL запросов. Какие типы команд существуют (например, “SqlCommand”, “OleDbCommand”)? Как указать текст запроса, параметры запроса и обработать результаты выполнения запроса?
4. Объясните, как работать с “транзакциями” (transactions) в ADO.NET. Как открыть, закрыть и управлять транзакциями? Какие механизмы используются для обеспечения атомарности, согласованности, изоляции и устойчивости (ACID) транзакций?
5. Как использовать ADO.NET для работы с разными типами баз данных? Какие провайдеры данных (например, “SqlClient”, “OleDb”) доступны в ADO.NET? Как настроить подключение к разным базам данных и как обрабатывать ошибки при работе с разными системами управления базами данных?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания
Выберите один правильный вариант ответа и нажмите кнопку «Далее»

Первый шаг при создании таблицы в SQL включает:
+ указание названия таблицы и описания её структуры
выбор типа данных для всех записей
определение названия базы данных
создание представления таблицы

Для добавления записи в таблицу используется команда:
+ INSERT
UPDATE
DELETE
SELECT

Команда для обновления существующих записей в базе данных:
+ UPDATE
INSERT
CREATE
ALTER

Тип данных, используемый для хранения строк, называется:
+ VARCHAR
INT
FLOAT
DATE

Удаление всех записей из таблицы, но не самой таблицы, выполняется с помощью:
+ DELETE
DROP
TRUNCATE
CLEAR

Для извлечения данных из таблицы используется команда:
+ SELECT
INSERT
DELETE
UPDATE

Команда, с помощью какой можно изменить имя существующей таблицы, называется:
+ RENAME
ALTER
EDIT
MODIFY

Для создания уникального индекса в таблице используется:
+ CREATE UNIQUE INDEX
CREATE INDEX
ALTER INDEX
SET INDEX

Для определения первичного ключа в таблице необходимо использовать:

+ PRIMARY KEY
FOREIGN KEY
UNIQUE
INDEX

Команды, которые изменяют структуру таблиц, относятся к:
+ DDL (Data Definition Language)
DML (Data Manipulation Language)
DCL (Data Control Language)
TCL (Transaction Control Language)

Основным назначением команды SELECT является:
+ извлечение данных из таблицы
изменение данных в таблице
удаление данных
создание таблицы

Фильтрация данных в команде SELECT осуществляется с помощью:
+ WHERE
ORDER BY
HAVING
GROUP BY

Агрегатные функции в SQL позволяют:
+ выполнять операции над набором данных, например, подсчет или сумму
изменять структуру таблицы
создавать новые базы данных
удалять записи

Оператор, используемый для объединения двух или более таблиц, называется:
+ JOIN
UNION
MERGE
LINK

Оператор GROUP BY используется для:
+ группировки данных по одному или нескольким полям
сортировки данных
фильтрации записей
удаления дублей

экспорт данных из базы данных в файл производится с помощью команды:
+ SELECT INTO OUTFILE
INSERT INTO
CREATE TABLE AS
ALTER TABLE EXPORT

Для установки доступа и прав на базу данных используется:
+ GRANT
SET
ALTER
CREATE

При создании новой таблицы команда CREATE TABLE требует:
+ указания названий столбцов и их типов
только названия таблицы
только названий столбцов
не требует дополнительных параметров

Оператор, используемый для определения условий соединения таблиц, называется:
+ ON
WITH
IN
AS

Подзапрос используется для:
+ выполнения запроса внутри другого запроса
управления транзакциями
изменения структуры таблицы
удаления сессий

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно выполнил 9-10 тестовых заданий.

Оценка «хорошо» выставляется обучающемуся, который: правильно выполнил 7-8 тестовых заданий.

Оценка «удовлетворительно» выставляется обучающемуся, который правильно выполнил 5-6 тестовых заданий.

Оценка «неудовлетворительно» выставляется обучающемуся, который правильно выполнил менее 4 тестовых заданий.

Вопросы собеседования к экзамену по темам 1-8

Контролируемые компетенции (знания, умения) ОК 01 ОК 02 ПК 1.1 ПК 1.2 ПК 1.6

1. Какие основные принципы объектно-ориентированного программирования применяются при разработке программных модулей?
2. Что такое абстракция данных и как она используется в разработке модулей?
3. Какие типы модулей существуют и для чего они применяются в программировании?
4. Какие инструменты и технологии используются при разработке модулей?
5. Каким образом происходит тестирование программных модулей и какие методы тестирования применяются?
6. Какие принципы SOLID применяются при проектировании модулей?
7. Каким образом происходит сопровождение и обновление программных модулей?
8. Какие методы и средства контроля версий используются при разработке модулей?
9. Каким образом происходит интеграция модулей в целостную программную систему?
10. Какие проблемы могут возникнуть при разработке и поддержке большого количества модулей?
11. Каким образом происходит оптимизация и улучшение производительности модулей?
12. Как организовать взаимодействие между разными модулями в программе?
13. Какие методы и практики разработки применяются для повышения безопасности программных модулей?
14. Какие стандарты и протоколы применяются при разработке модулей?
15. Каким образом можно оптимизировать процесс разработки модулей для ускорения проекта?
16. Какие методы обеспечения качества используются при разработке программных модулей?
17. Как оценить эффективность и эффективность работы программных модулей?
18. Как применять принцип DRY (Don't repeat yourself) при разработке модулей?
19. Каким образом можно обеспечить масштабируемость программных модулей?
20. Какую роль играют документация и комментарии при разработке программных модулей?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который правильно ответил на 5 случайно выбранных вопросов, показав достаточный уровень знаний. В случае если студент ответил на 4 вопроса правильно, но рассчитывает получить оценку «отлично» ему задаётся дополнительный вопрос ответить.

Оценка «хорошо» выставляется обучающемуся, который: правильно ответил на 4 случайно выбранных вопросов, показав достаточный уровень знаний. В случае если студент ответил на 3 вопроса правильно, но рассчитывает получить оценку «хорошо» ему задаётся дополнительный вопрос.

Оценка «удовлетворительно» выставляется обучающемуся, который: правильно ответил на 3 случайно выбранных вопросов, показав достаточный уровень знаний. В случае если студент ответил на 2 вопроса правильно, но рассчитывает получить оценку «удовлетворительно» ему задаётся дополнительный вопрос.

Оценка «неудовлетворительно» выставляется обучающемуся, который не ответил ни на один вопрос или ответил на 1 или 2 вопроса верно, но не ответил на дополнительный вопрос

Форма промежуточной аттестации по дисциплине *экзамен*.

Окончательные результаты обучения (формирования компетенций) определяются посредством перевода баллов, набранных студентом в процессе освоения дисциплины, в оценки:

– базовый уровень сформированности компетенции считается достигнутым если результат обучения соответствует оценке «удовлетворительно» (50 до 64 рейтинговых баллов);

– повышенный уровень сформированности компетенции считается достигнутым, если результат обучения соответствует оценкам «хорошо» (65-85 рейтинговых баллов) и «отлично» (86-100 рейтинговых баллов).

Дополнительные контрольные испытания

для студентов, набравших менее 50 баллов (в соответствии с Положением «О модульно-рейтинговой системе»), формируются из числа оценочных средств по темам, которые не освоены студентом.