

Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Выборнова Любовь Алексеевна
Должность: Ректор
Дата подписания: 31.08.2026 09:30:57
Уникальный программный ключ:
c3b3b9c625f6c113afa2a2c42baff9e05a38b76e

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования «Поволжский государственный университет сервиса»
(ФГБОУ ВО «ПВГУС»)

УТВЕРЖДЕНО

на заседании Высшей школы
интеллектуальных систем и
кибертехнологий

Протокол от 06.09.2025 № 2

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ
(фонд оценочных средств)

для проведения промежуточной аттестации по дисциплине

«Языки программирования»

наименование дисциплины

по образовательной программе высшего образования –
программе

бакалавриат

«Информационные системы и технологии»

наименование образовательной программы

09.03.02 «Информационные системы и технологии»

шифр, наименование направления подготовки / специальности

Составители

Коновалова Анжелика Денисовна, старший
преподаватель, Высшая школа интеллектуальных
систем и кибертехнологий

ФИО, должность, структурное подразделение,
ученая степень, ученое звание

Тольятти
2025

1. Паспорт фонда оценочных средств (далее – ФОС)

1.1. Планируемые результаты обучения по дисциплине

Код компетенции	Наименование компетенции	Код и наименование индикаторов достижения компетенции	Основание (ПС) *для профессиональных компетенций
ОПК-6	Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий	ИОПК-6.2. Использует современные языки программирования для разработки алгоритмов и программ ИОПК-6.3. Владеет методами отладки и тестирования программ	-
ПК-2	Способен к выполнению работ по сопровождению и разработке прототипов ИС	ИПК-2.2. Разрабатывает прототип ИС в соответствии с требованиями ИПК-2.3. Осуществляет тестирование прототипа ИС на проверку корректности архитектурных решений	06.015 Профессиональный стандарт «Специалист по информационным системам».

В результате изучения дисциплины обучающийся должен:

владеть навыками: основными методами, способами и средствами переработки информации на основе парадигмы императивного программирования в соответствии с технологией нисходящего структурного проектирования.

уметь: разрабатывать и записывать алгоритмы и программы на языках Бейсик и Паскаль в соответствие с технологией нисходящего структурного проектирования; записывать алгоритмы на языке схем

знать: этапы решения задачи на компьютере на основе парадигмы императивного программирования; понятие алгоритма и его основные свойства; основные понятия алгоритмической системы; основные средства записи и типы алгоритмов; цели и составные части технологии нисходящего структурного проектирования алгоритмов и программ

1.2. Содержание дисциплины

№	Тема (раздел дисциплины) (в соответствии с РПД)	Код компетенции
1.	ВВЕДЕНИЕ В C# И .NET. ПАРАДИГМЫ ПРОГРАММИРОВАНИЯ	ОПК-6 ПК-2
2.	СИСТЕМА ТИПОВ C# И УПРАВЛЕНИЕ ПАМЯТЬЮ	ОПК-6 ПК-2
3.	ПРОЦЕДУРНЫЕ СРЕДСТВА C# И МОДУЛЬНОСТЬ	ОПК-6 ПК-2
4.	ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ В C#	ОПК-6 ПК-2
5.	ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ C#	ОПК-6 ПК-2
6.	ДИНАМИЧЕСКИЕ ВОЗМОЖНОСТИ И РЕФЛЕКСИЯ В C#	ОПК-6

		ПК-2
7.	БИБЛИОТЕКИ, ПАКЕТЫ И ВЗАИМОДЕЙСТВИЕ В .NET	ОПК-6 ПК-2
8.	РАЗРАБОТКА ПРОТОТИПА ИС НА C#	ОПК-6 ПК-2
9.	ТЕСТИРОВАНИЕ ПРОТОТИПА НА C#	ОПК-6 ПК-2

1.3. Система оценивания по дисциплине

Дисциплина изучается в течение двух семестров.

Форма промежуточной аттестации по дисциплине – экзамен.

Форма промежуточной аттестации	Уровень сформированности	100-балльная шкала, %	Шкала освоения, %	5-балльная дифференцированная оценка/балл
экзамен	допороговый	ниже 61	ниже 61	«неудовлетворительно» / 2
	пороговый	61–85,9	61–69,9	«удовлетворительно» / 3
	пороговый	61–85,9	70–85,9	«хорошо» / 4
	повышенный	86–100	86–100	«отлично» / 5

2. Перечень оценочных материалов для проведения промежуточной аттестации

Контроль и оценка результатов освоения дисциплины осуществляется преподавателем в ходе текущего контроля успеваемости (в процессе проведения практических занятий, тестирования, опросов).

В ходе проведения промежуточной аттестации осуществляется контроль и оценка результатов освоения компетенций.

Перечень вопросов и заданий для подготовки к зачету (3 семестр)

ОПК-6. Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий

1. Понятие парадигмы программирования. Мультипарадигмальность C# как основное преимущество языка.
2. Архитектура .NET: CLR, CTS, CLS. Жизненный цикл выполнения программы на C# (компиляция Roslyn, JIT, выполнение).
3. Единая система типов .NET: значимые и ссылочные типы. Упаковка и распаковка. Влияние на производительность.
4. Управление памятью в .NET: стек и управляемая куча. Принципы работы сборщика мусора (поколения, алгоритмы).
5. Работа с неуправляемыми ресурсами: IDisposable, using, финализаторы.
6. Передача параметров в методы C#: ref, out, in. Возвращаемые ссылки (ref return).
7. Методы в C#: перегрузка, именованные и опциональные аргументы. Ключевое слово params.
8. Пространства имен и сборки. Модификаторы доступа и их роль в инкапсуляции.
9. Обработка исключений в C#. Иерархия исключений. Создание собственных исключений.
10. Объектно-ориентированное программирование в C#: классы и объекты. Инкапсуляция, наследование, полиморфизм.
11. Абстрактные классы и интерфейсы в C#: сравнение, области применения. Множественное наследование интерфейсов.
12. Композиция vs наследование: сравнение подходов, выбор стратегии проектирования.

13. Свойства в C#. Автоматические свойства. Вычисляемые свойства.
14. Ключевые слова `base`, `virtual`, `override`, `sealed`, `new`. Переопределение и сокрытие методов.
15. Generic-классы и Generic-методы. Ограничения параметров-типов (`where T : class, struct, new()` и др.).
16. Делегаты в C#: объявление, создание экземпляров, комбинирование. Анонимные методы.
17. События в C#. Механизм подписки и отписки. Связь событий и делегатов.
18. Лямбда-выражения в C#. Синтаксис и применение. Замыкания.
19. LINQ в C#: архитектура, синтаксис запросов, цепочки методов. Отложенное и немедленное выполнение.
20. LINQ to Objects: основные методы расширения (`Where`, `Select`, `SelectMany`, `OrderBy`, `GroupBy`, `Aggregate`, `Any`, `All`, `First`, `Single` и др.).
21. Сопоставление с образцом (pattern matching) в C#: `is`, `switch`, паттерны свойств, позиционные паттерны.
22. Неизменяемость в C#. Паттерн «Строитель» с функциональными цепочками. Создание имутабельных типов.
23. Рефлексия в C#: класс `Type`, получение информации о сборках, типах, членах.
24. Создание экземпляров и вызов методов через рефлексию. Динамическое связывание.
25. Атрибуты в C#: синтаксис создания, использование, получение через рефлексию. Сценарии применения.
26. Ключевое слово `dynamic`. Отличия от `var`. Сценарии использования в интеграции с COM и динамическими API.
27. Управление зависимостями в .NET: NuGet. Файл проекта. Семантическое версионирование.
28. Асинхронное программирование в C#: `Task`, `async/await`. Синхронизация контекста.
29. Работа с данными в .NET: `System.Text.Json`, `Newtonsoft.Json`. Сериализация в различные форматы.
30. Взаимодействие с внешними ресурсами: `HttpClient`, асинхронные запросы, обработка ответов.
31. Стандартная библиотека базовых классов .NET (BCL). Основные пространства имен (`System`, `System.Collections.Generic`, `System.Linq`, `System.IO`, `System.Threading.Tasks` и др.).
32. Модульность в .NET: проекты, решения, сборки. Типы сборок (исполняемые, библиотечные).

ПК-2. Способен к выполнению работ по сопровождению и разработке прототипов ИС

33. Прототипирование информационных систем: назначение, виды прототипов, критерии готовности.
34. Выбор технологического стека для прототипа: факторы, влияющие на решение (требования проекта, команда, инфраструктура).
35. Архитектура прототипа: слои (представление, бизнес-логика, хранение данных). Организация зависимостей.
36. Внедрение зависимостей (Dependency Injection) в .NET. Встроенный DI-контейнер.
37. Проектирование модели данных прототипа. Связи между сущностями (`one-to-many`, `many-to-many`).
38. Entity Framework Core: подходы (`Code-First`, `Database-First`). Миграции и управление схемой данных.
39. Реализация вертикального среза функциональности: разработка контроллера, сервиса и репозитория.
40. Создание Web API в .NET: атрибуты маршрутизации, привязка моделей, форматы ответов.
41. Уровни тестирования: модульное, интеграционное, системное. Роль каждого уровня в проверке прототипа.
42. Модульное тестирование в .NET: фреймворки (NUnit, xUnit), структура теста (`Arrange-Act-Assert`).
43. Тест-двойники: моки, стабы, шпионы. Использование библиотек-мокеров (`Moq`, `NSubstitute`).
44. Интеграционное тестирование: проверка взаимодействия слоев, тестирование базы данных, тестирование API.

45. Проверка архитектурной корректности. Инструменты для анализа зависимостей между модулями.
46. Покрывание кода тестами. Пороговые значения и методы увеличения покрытия.
47. Документирование архитектурных решений и результатов тестирования прототипа.
48. Сравнительная оценка прототипов на разных языках и технологических платформах.
49. Подготовка прототипа к демонстрации: оформление, инструкция по запуску, демонстрационный сценарий.
50. Практическое задание: разработка набора модульных и интеграционных тестов для проверки архитектурного решения прототипа.

Типовые практические задания при промежуточной аттестации

1. Сравнить две реализации одного алгоритма на разных языках по заданным критериям (производительность, читаемость, сопровождаемость).
2. Выбрать язык и технологический стек для прототипа ИС и обосновать решение по заданным функциональным и нефункциональным требованиям.
3. Спроектировать модульный интерфейс и показать его реализацию средствами выбранного языка.
4. Разработать тесты, проверяющие взаимодействие модулей прототипа.
5. По результатам теста определить нарушение архитектурного ограничения и предложить исправление.
6. Реализовать иерархию классов для заданной предметной области с использованием абстрактных классов и интерфейсов.
7. Создать минимальное Web API с одним контроллером, внедрением простого сервиса и подключением к локальной базе данных.
8. Разработать конвейер обработки данных с использованием LINQ и функциональных цепочек.
9. Спроектировать кастомный атрибут и реализовать утилиту для валидации объектов на основе этого атрибута с использованием рефлексии.