

МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ
ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ЛГПУ»)**

Структурное подразделение Институт физико-математического
образования, информационных и обслуживающих технологий
Кафедра информационных образовательных технологий и систем

УТВЕРЖДАЮ

Врио директора ИФМОИОТ

Е.А. Журавлева

«15»  2025 г.



Приложение к рабочей программе учебной дисциплины

**ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
для проведения текущего контроля и промежуточной аттестации
обучающихся по дисциплине
«Современные инструменты разработки программного обеспечения»**

По направлению подготовки 09.04.04 Программная инженерия

Профиль подготовки Программное обеспечение систем и комплексов

Квалификация выпускника – магистр

Форма обучения очная, заочная

Курс ОФО – 1 курс, ЗФО – 1 курс

Разработчик

Швыров В.В.

канд. физ.-мат. наук, доцент, доцент
кафедры информационных
образовательных технологий и систем

Заведующий кафедрой

 Д.А. Капустин

Протокол от «14»  2025 г. № 9

Луганск, 2025

1. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ

1.1. Область применения

Фонд оценочных средств (ФОС) – неотъемлемая часть рабочей программы дисциплины (модуля) Современные инструменты разработки программного обеспечения и предназначен для контроля и оценки образовательных достижений студентов, освоивших программу дисциплины (модуля).

1.2. Цели и задачи фонда оценочных средств

Цель ФОС – установить соответствие уровня подготовки обучающегося требованиям ФГОС ВО бакалавриат / специалитет / магистратура по направлению подготовки 09.04.04 Программная инженерия, утвержденного приказом Министерства образования и науки Российской Федерации от 19 сентября 2017 г. № 932 (с изменениями и дополнениями).

1.3. Перечень компетенций, формируемых в процессе освоения основной образовательной программы

Процесс освоения дисциплины направлен на формирование следующих компетенций и индикаторов их достижения:

Код по ФГОС ВО	Индикатор достижения
Универсальные	
Общепрофессиональные	
Профессиональные	
ПК-2. Владение методами программной реализации распределенных информационных систем ПК-6. Понимание существующие подходы к верификации моделей программного обеспечения	ПК-2.1. Знать методы программной реализации распределенных информационных систем ПК-2.2. Уметь использовать методы программной реализации распределенных информационных систем ПК-2.3. Владеть навыками использования методов программной реализации распределенных информационных систем ПК-6.1. Знать методы верификации моделей программного обеспечения ПК-6.2. Уметь использовать методы верификации моделей программного обеспечения ПК-6.3. Владеть навыками использования методов верификации моделей программного обеспечения

1.4. Этапы формирования компетенций и средства оценивания уровня их сформированности

Этапы формирования компетенций	Компетенции	Контрольно-оценочные средства / способ оценивания
Тема 1. Введение. Основные методы разработки программного обеспечения.	ПК-2; ПК-6	Выполнение лабораторных работ
Тема 2. Обзор современных технологий разработки.	ПК-2; ПК-6	Выполнение лабораторных работ
Тема 3. Проектирование разработки.	ПК-2; ПК-6	Выполнение лабораторных работ
Тема 4. Использование языков высокого уровня в процессе разработки.	ПК-2; ПК-6	Выполнение лабораторных работ
Тема 5. Разработка веб-приложений.	ПК-2; ПК-6	Выполнение лабораторных работ
Тема 6. Современные методы обработки данных.	ПК-2; ПК-6	Выполнение лабораторных работ
Тема 7. Визуализация многомерных данных в веб-приложениях	ПК-2; ПК-6	Выполнение лабораторных работ
Тема 8. Тестирование, стандартизация и отладка программного кода.	ПК-2; ПК-6	Выполнение лабораторных работ
Текущая аттестация	ПК-2; ПК-6	Контрольная работа
Промежуточная аттестация	ПК-2; ПК-6	Экзамен (письменный)

1.5. Описание показателей формирования компетенций

Код компетенции	Результаты сформированности
ПК-2. Владение методами программной реализации распределенных информационных систем ПК-6. Понимание существующие подходы к верификации моделей программного обеспечения	ПК-2.1. Знает методы программной реализации распределенных информационных систем ПК-2.2. Умеет использовать методы программной реализации распределенных информационных систем ПК-2.3. Владеет навыками использования методов программной реализации распределенных информационных систем ПК-6.1. Знает методы верификации моделей программного обеспечения ПК-6.2. Умеет использовать методы верификации моделей программного обеспечения ПК-6.3. Владеет навыками использования методов верификации моделей программного обеспечения

1.6. Критерии оценивания компетенций на разных этапах их формирования

Вид учебной работы	Количество баллов		
2 семестр / 2-3 триместр			
	ОФО	О-ЗФО	ЗФО
Оформление отчетов по лабораторным работам	30 баллов		
Работа на лабораторных занятиях	30 баллов		
Выполнение тестовых заданий	-		
Выполнение заданий самостоятельной работы	10 баллов		
экзамена	30 баллов		
Итого за семестр:	100 баллов		
Всего	100 баллов		

Накопительная система оценивания по 100-балльной шкале

Четырехбалльная система оценивания экзамена	100-балльная шкала	Буквенная шкала, соответствующая 100-балльной шкале	Система оценивания зачета
Отлично	90–100	А – отлично – теоретическое содержание курса освоено полностью, без пробелов; необходимые практические навыки работы с освоенным материалом сформированы; все предусмотренные программой обучения учебные задания выполнены, качество их выполнения оценено числом баллов, близким к максимальному	Зачтено
Хорошо	83–89	В – очень хорошо – теоретическое содержание курса освоено полностью, без пробелов; необходимые практические навыки работы с освоенным материалом в основном сформированы; все предусмотренные программой обучения учебные задания выполнены, качество выполнения большинства из них оценено числом баллов, близким к максимальному	
Хорошо	75–82	С – хорошо – теоретическое содержание курса освоено полностью; некоторые практические навыки работы с освоенным материалом сформированы недостаточно; все предусмотренные программой обучения учебные задания выполнены, качество выполнения ни одного из них не оценено минимальным числом баллов, некоторые виды заданий выполнены с ошибками	
Удовлетворительно	63–74	Д – удовлетворительно – теоретическое содержание дисциплины освоено частично, но пробелы не носят существенного характера; необходимые практические навыки работы с освоенным материалом в основном сформированы; большинство	

		предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий, содержат ошибки	
Удовлетворительно	50–62	Е – посредственно – теоретическое содержание курса освоено частично; некоторые практические навыки работы не сформированы, многие предусмотренные программой обучения учебные задания не выполнены либо качество выполнения некоторых из них оценено числом баллов, близким к минимальному	
Неудовлетворительно	21–49	FX – неудовлетворительно – теоретическое содержание курса освоено частично; необходимые практические навыки работы не сформированы; большинство предусмотренных программой обучения учебных заданий не выполнено либо качество их выполнения оценено числом баллов, близким к минимальному; при дополнительной самостоятельной работе над материалом курса возможно повышение качества выполнения учебных заданий	Не зачтено
Неудовлетворительно	0–20	F – неудовлетворительно – теоретическое содержание курса не освоено; необходимые практические навыки работы не сформированы; все выполненные учебные задания содержат грубые ошибки, дополнительная самостоятельная работа над материалом курса не приведет к какому-либо значимому повышению качества выполнения учебных заданий	

2. КОНТРОЛЬНО-ОЦЕНОЧНЫЕ СРЕДСТВА

2.1. Оценочные средства текущего контроля (типовые)

Вопросы для текущего контроля:

1. Что такое методология разработки программного обеспечения?
2. Какие основные этапы включает процесс разработки ПО?
3. В чем различие между классическими и гибкими методами разработки?
4. Какие существуют основные модели разработки ПО?
5. В чем суть каскадной модели разработки ПО?
6. Какие преимущества и недостатки у V-образной модели?
7. Как работает инкрементная модель разработки?
8. Чем спиральная модель отличается от каскадной?
9. В каких случаях целесообразно использовать модель RAD (Rapid Application Development)?
10. Какие основные принципы лежат в основе Agile-разработки?
11. Чем Scrum отличается от Kanban?
12. Какова роль Product Owner в Scrum?
13. Что такое спринт в Scrum?
14. Как работает метод экстремального программирования (XP)?
15. Как DevOps влияет на процесс разработки ПО?
16. Что такое CI/CD и зачем оно нужно?
17. Какие инструменты чаще всего используются в DevOps-практиках?
18. Какие критерии помогают выбрать метод разработки для конкретного проекта?
19. Какие модели лучше всего подходят для стартапов и почему?
20. Какие вызовы могут возникнуть при переходе с традиционных методов разработки на Agile?
21. Какие основные направления существуют в современных технологиях разработки программного обеспечения?
22. Чем отличается фронтенд-разработка от бэкенд-разработки?
23. Какие языки программирования наиболее востребованы сегодня?
24. Какие основные технологии используются в разработке веб-интерфейсов?
25. В чем разница между React, Angular и Vue.js?
26. Что такое адаптивный и отзывчивый дизайн?
27. Какие инструменты применяются для сборки и оптимизации фронтенд-кода?
28. Какие фреймворки наиболее популярны для серверной разработки?
29. В чем различие между монолитной и микросервисной архитектурой?
30. Что такое REST API и GraphQL, в чем их различия?
31. Какую роль играет контейнеризация (Docker, Kubernetes) в разработке ПО?
32. Чем отличаются реляционные (SQL) и нереляционные (NoSQL) базы данных?

33. В каких случаях лучше использовать PostgreSQL, а в каких — MongoDB?
34. Что такое кэширование и какие технологии для него применяются?
35. Что такое DevOps и как он влияет на процесс разработки?
36. В чем суть CI/CD, и какие инструменты для него используются?
37. Как работает serverless-архитектура, и в чем ее преимущества?
38. Какие технологии используются в облачной разработке (AWS, Azure, Google Cloud)?
39. Какие основные принципы обеспечения безопасности в веб-разработке?
40. Какие методы оптимизации производительности приложений наиболее популярны?
41. Что такое WebAssembly и как оно применяется в современных веб-приложениях?
42. Как работают Progressive Web Apps (PWA) и в чем их преимущества?
43. В чем разница между виртуальными машинами и контейнерами?
44. Какие технологии используются для работы с big data?
45. Как работает искусственный интеллект в программировании и какие библиотеки для него существуют?
46. Что такое блокчейн и в каких сферах он применяется?
47. Какие технологии помогают реализовать автоматическое тестирование приложений?
48. Как работает серверный рендеринг (SSR) в веб-приложениях?
49. Какие основные инструменты используются для мониторинга и логирования в современных системах?
50. Какие перспективные технологии разработки ПО могут стать популярными в ближайшие годы?
51. Что такое проектирование разработки программного обеспечения?
52. Какие основные этапы включает процесс проектирования ПО?
53. Чем логическое проектирование отличается от физического?
54. Какие основные подходы используются в проектировании программных систем?
55. Что такое архитектурные паттерны, и какие из них наиболее популярны?
56. В чем различие между многослойной (N-tier) и микросервисной архитектурой?
57. Каковы основные преимущества и недостатки монолитной архитектуры?
58. Что такое шаблоны проектирования (Design Patterns), и какие категории они включают?
59. Как работает паттерн MVC (Model-View-Controller)?
60. Чем отличается паттерн MVVM от MVC?
61. Что такое DDD (Domain-Driven Design), и в каких случаях он применяется?
62. Какую роль играет UML в проектировании программного обеспечения?

63. Какие основные диаграммы UML используются при проектировании систем?
64. Что такое ER-диаграмма, и как она помогает в проектировании баз данных?
65. Какие существуют принципы SOLID и почему они важны?
66. Что такое рефакторинг кода, и зачем он нужен в процессе проектирования?
67. Каковы основные подходы к масштабированию программных систем?
68. В чем суть принципа DRY (Don't Repeat Yourself)?
69. Что такое антипаттерны проектирования, и какие примеры можно привести?
70. Какую роль играет документация в процессе проектирования программного обеспечения?

2.2. Оценочные средства для промежуточной аттестации

Вопросы для проведения аттестации

1. Что такое языки программирования высокого уровня?
2. Какие ключевые отличия языков высокого уровня от низкоуровневых языков?
3. Какие преимущества дает использование языков высокого уровня в разработке ПО?
4. Какие наиболее популярные языки программирования высокого уровня используются сегодня?
5. Чем компилируемые языки отличаются от интерпретируемых?
6. Какие языки программирования считаются мультипарадигменными и почему?
7. Каковы основные принципы объектно-ориентированного программирования (ООП)?
8. Чем функциональное программирование отличается от ООП?
9. Какое значение имеет автоматическое управление памятью (Garbage Collection) в языках высокого уровня?
10. Почему Python считается удобным языком для начинающих программистов?
11. В чем преимущества использования Java для корпоративных решений?
12. Какие особенности языка C# делают его популярным в разработке приложений под Windows?
13. Как язык JavaScript используется в веб-разработке?
14. В чем основные различия между Java и JavaScript?
15. Почему язык Rust набирает популярность в сфере системного программирования?
16. Какие преимущества дает использование Kotlin в разработке Android-приложений?
17. В чем особенности работы со статической и динамической типизацией?

18. Какой язык программирования лучше всего подходит для научных вычислений и анализа данных?
19. Почему TypeScript становится все более популярным среди веб-разработчиков?
20. Какие инструменты помогают в разработке на языках высокого уровня (IDE, фреймворки, библиотеки)?
21. Как использование языков высокого уровня влияет на производительность программ?
22. В чем заключается роль компиляторов и интерпретаторов в разработке на языках высокого уровня?
23. Как языки высокого уровня взаимодействуют с базами данных?
24. В каких сферах чаще всего применяются языки высокого уровня?
25. Какова роль языков высокого уровня в разработке искусственного интеллекта и машинного обучения?
26. Какие особенности имеет асинхронное программирование в языках высокого уровня?
27. Как работают многопоточные и многозадачные программы на языках высокого уровня?
28. В чем преимущества и недостатки использования кроссплатформенных языков?
29. Как технологии JIT (Just-In-Time) компиляции влияют на производительность программ?
30. Как выбрать подходящий язык программирования высокого уровня для конкретного проекта?
31. Какие основные современные методы используются в разработке веб-приложений?
32. В чем разница между монолитной и микросервисной архитектурой в веб-разработке?
33. Как работает JAMstack, и в чем его преимущества?
34. Какие принципы лежат в основе методологии Progressive Web Apps (PWA)?
35. В чем различие между SSR (Server-Side Rendering) и CSR (Client-Side Rendering)?
36. Как работает статическая генерация сайтов (SSG), и какие инструменты используются для этого?
37. Какие основные фреймворки применяются в разработке современных веб-приложений?
38. Какое место занимает WebAssembly (WASM) в современной веб-разработке?
39. Что такое GraphQL, и как он отличается от REST API?
40. Какие технологии используются для создания real-time веб-приложений?
41. Как микрофронтенды помогают в организации веб-разработки?
42. Какие инструменты DevOps применяются в веб-разработке?
43. Как работает CI/CD в разработке веб-приложений?
44. Какие преимущества дает использование контейнеризации (Docker, Kubernetes) в веб-разработке?

45. В чем суть подхода Headless CMS, и какие системы используются чаще всего?
46. Как современные методы тестирования помогают в разработке веб-приложений?
47. Какие технологии используются для serverless-разработки веб-приложений?
48. Как API-first подход влияет на разработку современных веб-приложений?
49. Какие методы аутентификации и авторизации наиболее популярны в веб-приложениях?
50. Как искусственный интеллект и машинное обучение внедряются в современные веб-приложения?
51. Какие основные современные методы обработки данных существуют?
52. В чем разница между потоковой (streaming) и пакетной (batch) обработкой данных?
53. Какие технологии используются для потоковой обработки данных?
54. Как работают распределенные системы обработки данных?
55. Что такое MapReduce, и как он применяется в обработке больших данных?
56. Какие особенности имеет обработка данных в облачных сервисах?
57. В чем преимущества использования Apache Spark по сравнению с Hadoop?
58. Как работают NoSQL базы данных, и в каких случаях их лучше использовать?
59. Какие технологии используются для обработки данных в реальном времени?
60. Что такое ETL (Extract, Transform, Load), и как он применяется?
61. Какие инструменты применяются для автоматизации обработки данных?
62. Как машинное обучение помогает в обработке больших объемов данных?
63. Что такое Data Lake, и чем он отличается от традиционного хранилища данных?
64. Какие преимущества дает использование serverless-технологий для обработки данных?
65. Как работают графовые базы данных, и где они применяются?
66. Какие методы обеспечивают безопасность данных при обработке?
67. Какую роль играет DataOps в современных методах обработки данных?
68. Что такое обработка данных на периферии (Edge Computing)?
69. Как работают алгоритмы дедупликации и сжатия данных?
70. Какие перспективные технологии обработки данных могут стать популярными в ближайшие годы?