

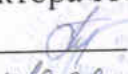

МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

**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
«Специальные языки программирования»

По направлению подготовки 44.03.05 Педагогическое образование (с двумя профилями подготовки)

Профиль подготовки Математика. Информатика

Квалификация выпускника – бакалавр

Форма обучения очная, заочная

Курс ОФО – 4 курс, ЗФО – 6 курс

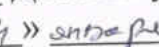
Разработчик

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) Программирование на языке Python 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 44.03.05 Педагогическое образование (с двумя профилями подготовки), утвержденным приказом Министерства образования и науки Российской Федерации от 22 февраля 2018 г. №125 (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-3. Способен осваивать и применять базовые научно-теоретические знания и практические умения по информатике в профессиональной деятельности	ПК.3.1. Способен формировать и реализовывать программы развития универсальных учебных действий по информатике ПК.3.2. Демонстрирует знание содержания образовательных программ по информатике ПК.3.3. Способен проектировать образовательные программы различных уровней и элементы образовательных программ в предметной области «Информатика»

1.4. Этапы формирования компетенций и средства оценивания уровня их сформированности

Этапы формирования компетенций	Компетенции	Контрольно-оценочные средства / способ оценивания
Тема 1. Введение. Области применения языка Python.	ПК-3	Выполнение лабораторных работ
Тема 2. Основы синтаксиса. Основные типы данных.	ПК-3	Выполнение лабораторных работ

Тема 3. Циклы.	ПК-3	Выполнение лабораторных работ
Тема 4. Функции в языке Python.	ПК-3	Выполнение лабораторных работ
Тема 5. Массивы. Пакет numpy.	ПК-3	Выполнение лабораторных работ
Текущая аттестация	ПК-3	Контрольная работа
Промежуточная аттестация	ПК-3	Экзамен

1.5. Описание показателей формирования компетенций

Код компетенции	Результаты сформированности
ПК-3. Способен осваивать и применять базовые научно-теоретические знания и практические умения по информатике в профессиональной деятельности	<p>ПК.3.1. Способен формировать и реализовывать программы развития универсальных учебных действий по информатике</p> <p>ПК.3.2. Демонстрирует знание содержания образовательных программ по информатике</p> <p>ПК.3.3. Способен проектировать образовательные программы различных уровней и элементы образовательных программ в предметной области «Информатика»</p>

1.6. Критерии оценивания компетенций на разных этапах их формирования

Вид учебной работы	Количество баллов		
8 семестр / 16 триместр			
	ОФО	О-ЗФО	ЗФО
Оформление отчетов по лабораторным работам	30 баллов		30 баллов
Работа на лабораторных занятиях	30 баллов		30 баллов
Выполнение тестовых заданий	-		-
Выполнение заданий самостоятельной работы	10 баллов		10 баллов
зачета	30 баллов		30 баллов
Итого за семестр:	100 баллов		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	
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	Зачтено
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	D – 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	E – 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. Какие особенности делают язык программирования Python привлекательным для начинающих разработчиков?
2. В каких областях программирования часто используется Python?
3. Какие преимущества языка Python делают его популярным в области научных исследований и анализа данных?
4. Как Python поддерживает разработку веб-приложений и веб-фреймворки?
5. Какие возможности языка Python предоставляют для автоматизации задач и написания скриптов?
6. В чем заключается роль Python в области машинного обучения и искусственного интеллекта?
7. Какие инструменты и библиотеки в области графического интерфейса пользователя (GUI) используют Python?
8. Как Python применяется в сфере разработки игр и виртуальной реальности?
9. Какие преимущества предоставляет использование Python в области сетевого программирования?
10. Как язык Python используется в системном администрировании и разработке сетевых приложений?
11. Какова цель использования отступов в языке Python?
12. Что такое переменная и как объявляется в Python?
13. Как создать комментарий в Python?
14. Какие ключевые слова используются для создания условий в операторе if?
15. Какие операторы используются для выполнения арифметических операций в Python?
16. Как объявить строковую переменную в Python?
17. Как осуществляется конкатенация строк в Python?
18. Как получить подстроку из строки в Python?
19. Какие методы используются для работы с числами с плавающей точкой в Python?
20. Как определить длину строки в Python?
21. Что такое индексация и как она применяется к строкам и спискам в Python?
22. Как создать список в Python и какие операции можно выполнять с ним?
23. Как определить длину списка в Python?
24. Какие методы используются для добавления и удаления элементов из списка в Python?
25. Как создать кортеж в Python и в чем отличие от списка?
26. Какие операторы используются для доступа к элементам кортежа?
27. Как создать множество (set) в Python, и какие операции поддерживаются для множеств?

28. Как объявить словарь в Python и какие операции можно выполнять с ним?
29. Как получить значение по ключу из словаря в Python?
30. Как осуществляется изменение значений в словаре?
31. Какие встроенные функции используются для преобразования типов данных в Python?
32. Как выполнить ввод данных с клавиатуры в Python?
33. Как использовать условные операторы if-else в Python?
34. Как использовать циклы for и while в Python?
35. Какие встроенные функции используются для работы с числами в Python (например, abs, round)?
36. Какие встроенные функции Python применяются для работы со строками (например, len, upper, lower)?
37. Какие функции используются для работы с списками в Python (например, append, remove)?
38. Какие операторы применяются для сравнения значений в Python (например, ==, <, >)?
39. Как объединить списки в Python?
40. Как создать функцию в Python, и как передать аргументы функции?
41. Какие встроенные функции Python применяются для работы с файлами (например, open, read)?
42. Как обработать исключение (использовать блок try-except) в Python?
43. Какие операторы применяются для логических операций в Python (например, and, or, not)?
44. Как создать многомерный список (список списков) в Python?
45. Как использовать генераторы списков (list comprehensions) в Python?
46. Как определить тип данных переменной в Python?
47. Как работает оператор "in" для проверки наличия элемента в списке или другой коллекции?
48. Как сортировать список в Python?
49. Как создать условие для выполнения действий только при выполнении определенного условия (например, if-else)?
50. Какие встроенные функции Python используются для поиска минимального и максимального значения в списке (например, min, max)?
51. Какие типы циклов поддерживаются в Python?
52. Что такое цикл "for" в Python и как он используется?
53. Какой синтаксис применяется для создания цикла "for" с использованием функции range()?
54. Как работает цикл "for" с использованием списков и других итерируемых объектов?
55. Как выход из цикла "for" до завершения всех итераций (break, continue)?
56. Как использовать цикл "for" для перебора элементов словаря в Python?
57. Как создать бесконечный цикл с использованием ключевого слова "while"?
58. Как организовать цикл "while" с использованием условного оператора?

59. Как выходить из цикла "while" при выполнении определенного условия (break)?
 60. Как использовать цикл "while" для создания простого меню с вариантами выбора?
 61. Какие операторы используются для работы с итерациями внутри циклов (например, enumerate, zip)?
 62. Как перебирать элементы двумерного списка с помощью вложенных циклов?
 63. Как создать цикл "for" с использованием генератора списков (list comprehensions)?
 64. Какие встроенные функции Python могут быть использованы в циклах для обработки данных (например, sum, len)?
 65. Как работает цикл "while" с использованием оператора else?
 66. Как использовать ключевое слово pass в циклах?
 67. Какие методы могут быть применены к строкам в циклах для обработки символов?
 68. Как можно использовать цикл "for" для генерации последовательности чисел в обратном порядке?
 69. Как создать бесконечный цикл с использованием itertools?
 70. Какие типы данных могут быть использованы в условиях циклов для определения продолжения или завершения выполнения цикла?
- 2.2. Оценочные средства для промежуточной аттестации**

Вопросы для проведения аттестации

1. Что такое функция в Python?
2. Как объявляется функция в Python?
3. Какие элементы включают в себя заголовок функции?
4. Как передать аргументы в функцию в Python?
5. Какие типы параметров поддерживаются в Python (например, обязательные, необязательные, именованные)?
6. Как можно указать значения по умолчанию для параметров функции?
7. Как использовать произвольное количество аргументов в функции?
8. Как определить функцию с произвольным числом именованных аргументов?
9. Как работает оператор return в функции?
10. Как использовать функцию без явного возврата значения?
11. Как объявить лямбда-функцию в Python?
12. В чем отличие между функцией и лямбда-функцией?
13. Какие встроенные функции Python используются для работы с функциями (например, map, filter)?
14. Как объявить глобальную переменную внутри функции?
15. Как создать замыкание (closure) в Python?
16. Как передавать функцию как аргумент другой функции?
17. Как определить аннотации типов для параметров и возвращаемого значения функции?
18. Как использовать функции из сторонних модулей в своем коде?
19. Как создать декоратор для функции в Python?

20. Как применять декоратор к функции?
21. Как использовать ключевое слово `nonlocal` внутри функции?
22. Как определить функцию с переменным числом аргументов в Python 3.8 и выше?
23. Как использовать аргументы со значением по умолчанию вместе с аргументами переменной длины?
24. Как объявить функцию с аннотациями типов в Python?
25. Как вызвать функцию с использованием именованных аргументов?
26. Какие встроенные функции Python поддерживают работу с функциональным программированием (например, `map`, `reduce`, `filter`)?
27. Как создать и использовать генераторную функцию?
28. Как можно использовать функцию в качестве атрибута класса?
29. Каким образом можно избежать конфликта имен при импорте функций из модулей?
30. Как определить и использовать анонимные функции в Python?
31. Что такое массив в программировании?
32. Как создать одномерный массив в NumPy?
33. Как создать многомерный массив в NumPy?
34. Как узнать размерность массива в NumPy?
35. Как определить тип данных элементов в массиве NumPy?
36. Как заполнить массив определенным значением в NumPy?
37. Как создать последовательность чисел в массиве NumPy?
38. Как изменить форму (`shape`) массива в NumPy?
39. Как осуществить доступ к элементам массива в NumPy?
40. Как создать массив с равномерно распределенными значениями в NumPy?
41. Как производить операции между массивами в NumPy?
42. Как выполнить матричное умножение в NumPy?
43. Как создать массив случайных чисел в NumPy?
44. Как найти минимальное и максимальное значения в массиве NumPy?
45. Как рассчитать среднее значение и стандартное отклонение в массиве NumPy?
46. Как использовать условные выражения для манипуляций с массивами в NumPy?
47. Как создать маску (массив булевых значений) в NumPy?
48. Как применить маску для извлечения подмассива из массива NumPy?
49. Как выполнить сортировку массива в NumPy?
50. Какие функции NumPy используются для статистического анализа массивов?
51. Как создать массив с конкретными значениями на диагонали в NumPy?
52. Как производить операции с транспонированным массивом в NumPy?
53. Как использовать функции `np.sum()` и `np.prod()` для вычисления суммы и произведения значений в массиве?
54. Как осуществить индексацию с использованием массивов индексов в NumPy?
55. Как обрезать или изменять форму массива в NumPy?
56. Как работает `broadcasting` в NumPy?

57. Как применить функцию к каждому элементу массива в NumPy?
58. Как создать массив с использованием функции, передаваемой в `np.fromfunction()`?
59. Как использовать функции `np.linspace()` и `np.logspace()` для создания равномерно распределенных значений в интервале?
60. Как создать массив из файла в NumPy?
61. Как производить вычисления с использованием линейной алгебры в NumPy?
62. Какие функции NumPy предоставляют поддержку для работы с полиномами?
63. Как сохранить и загрузить массивы с использованием `np.save()` и `np.load()`?
64. Как обработать пропущенные значения в массиве NumPy?
65. Как использовать методы агрегации (например, `np.mean()`, `np.sum()`) в NumPy?
66. Как создать копию массива в NumPy?
67. Как выполнять операции с множествами с использованием массивов в NumPy?
68. Как использовать методы для работы с числовыми производными в NumPy?
69. Как осуществлять операции с комплексными числами в массивах NumPy?
70. Как измерить время выполнения операции с массивами в NumPy?