



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЦИФРОВОЙ ЭКОНОМИКИ

СОГЛАСОВАНО
Руководитель ОП

Р.И. Дремлюга

«17» июня 2019 г.

УТВЕРЖДАЮ

Директор Школы цифровой
экономики



И.Г. Мирин

2019 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«ЯЗЫКИ, АЛГОРИТМЫ И МЕТОДЫ ПРОГРАММИРОВАНИЯ»
направления 09.04.01 Информатика и вычислительная техника
Магистерская программа «Искусственный интеллект и большие данные»
Форма подготовки очная

Школа цифровой экономики

курс 1 семестр 1, 2

лекции 36 час.

практические занятия 0 час.

лабораторные работы 18 час.

всего часов аудиторной нагрузки 36 час.

самостоятельная работа 72 час.

контрольные работы программой не предусмотрены

курсовая работа/проект – не предусмотрено

зачет с оценкой 1, 2 семестр

экзамен – не предусмотрено учебным планом

Рабочая программа составлена в соответствии с требованиями Федерального государственного образовательного стандарта высшего образования по направлению подготовки/специальности 09.04.01 Информатика и вычислительная техника, утвержденного приказом Министерства образования и науки Российской Федерации от 19.09.2017 г. № 918.

Рассмотрена и утверждена на заседании Дирекции Школы цифровой экономики «17» июня 2019 года (протокол № 124-01-07-05).

Составитель: ст.пр. Кленин А.С.

Оборотная сторона титульного листа РПД

I. Рабочая программа пересмотрена на заседании Дирекции Школы цифровой экономики:

Протокол от «_____» _____ 20__ г. № _____

Заместитель директора ШЦЭ

по учебной и воспитательной работе _____
(подпись) (И.О. Фамилия)

II. Рабочая программа пересмотрена на заседании Дирекции Школы цифровой экономики:

Протокол от «_____» _____ 20__ г. № _____

Заместитель директора ШЦЭ

по учебной и воспитательной работе _____
(подпись) (И.О. Фамилия)

АННОТАЦИЯ

Б1.О.02.03 ЯЗЫКИ, АЛГОРИТМЫ И МЕТОДЫ ПРОГРАММИРОВАНИЯ

Рабочая программа учебной дисциплины «Языки, алгоритмы и методы программирования» предназначена для студентов, обучающихся по направлению подготовки 09.04.01 Информатика и вычислительная техника (уровень магистратуры), профиль «Искусственный интеллект и большие данные».

Дисциплина «Языки, алгоритмы и методы программирования» входит в базовую часть блока «Дисциплины (модули) Б.1» (Б1.О.02.03) учебного плана подготовки магистров.

Общая трудоемкость освоения дисциплины составляет 7 зачетных единиц или 252 часов. Дисциплина реализуется на 1 курсе в 1 семестре.

Семестр	Аудиторные занятия			Самостоятельная работа	Форма контроля	Всего по дисциплине	
	Лекции	Лабораторные работы	Практические занятия			Часы	Зачетные единицы
1 семестр	18	-	36	90	зачет с оценкой	144	4
2 семестр	18	-	36	54	зачет с оценкой	108	3
ИТОГО	36		72	144		252	7

Место дисциплины в структуре ОПОП

Дисциплина «Языки, алгоритмы и методы программирования» входит в обязательную часть Блока 1. «Дисциплины (модули)» основной профессиональной образовательной программы магистратуры, модуль «Машинное обучение и программирование».

Цель изучения дисциплины – формирование общекультурных компетенций будущих специалистов, работающих с технологиями виртуальной и дополненной реальности, через ознакомление с общими

принципами построения и использования языков программирования, а также развитие навыков проектирования и реализации алгоритмов решения практических задач на одном из языков программирования.

Задачи:

По окончании курса студент должен знать:

- терминологию дисциплины;
- основные структуры и инструментарий, которые применяются в языках программирования:
- основные структуры и типы данных;
- основные методы при разработке алгоритмов (рекурсия, отход назад, метод ветвей и границ, анализ арифметических выражений);
- базовые алгоритмы на динамических структурах данных;
- библиотеки стандартных программ.

Студент должен уметь:

- применять методы программирования при разработке информационных систем;
- определять структуры данных при проектировании алгоритмов в процессе решения задач;
- разбивать решение сложной задачи на последовательность более простых задач;
- использовать библиотеки стандартных программ, которые включены в язык программирования;
- самостоятельно освоить тот язык программирования, который необходимо использовать при решении задач.

Для успешного освоения дисциплины «Языки и методы программирования» студент должен:

Знать: основы компьютерной грамотности.

Уметь: работать с файлами, ориентироваться в интерфейсе новых программ.

Владеть: основными навыками работы в интегрированной среде.

В результате данной дисциплины у обучающихся формируются следующие общепрофессиональные компетенции (элементы компетенций).

Код и формулировка компетенции	Этапы формирования компетенции
<p>ОПК-2. Способен разрабатывать оригинальные алгоритмы и программные средства, в том числе с использованием современных интеллектуальных технологий, для решения профессиональных задач</p>	<p>ОПК-2.1 Знать: современные информационно-коммуникационные и интеллектуальные технологии, инструментальные среды, программно-технические платформы для решения профессиональных задач ОПК-2.2 Уметь: обосновывать выбор современных информационно-коммуникационных и интеллектуальных технологий, разрабатывать оригинальные программные средства для решения профессиональных задач ОПК-2.3 Владеть: навыками разработки оригинальных программных средств, в том числе с использованием современных информационно-коммуникационных и интеллектуальных технологий, для решения профессиональных задач</p>
<p>ОПК-5. Способен разрабатывать и модернизировать программное и аппаратное обеспечение информационных и автоматизированных систем</p>	<p>ОПК-5.1 Знать: современное программное и аппаратное обеспечение информационных и автоматизированных систем ОПК-5.2 Уметь: модернизировать программное и аппаратное обеспечение информационных и автоматизированных систем для решения профессиональных задач ОПК-5.3 Владеть: навыками разработки программного и аппаратного обеспечения информационных и автоматизированных систем для решения профессиональных задач</p>
<p>ОПК-8. Способен осуществлять эффективное управление разработкой программных средств и проектов</p>	<p>ОПК-8.1 Знать: методы и средства разработки программного обеспечения, методы управления проектами разработки программного обеспечения, способы организации проектных данных, нормативно-технические документы (стандарты и регламенты) по разработке программных средств и проектов ОПК-8.2 Уметь: выбирать средства разработки, оценивать сложность проектов, планировать ресурсы, контролировать сроки выполнения и оценивать качество полученного результата. ОПК-8.3 Владеть: навыками разработки технического задания, составления планов, распределения задач, тестирования и оценки качества программных средств</p>

Для формирования вышеуказанных компетенций в рамках дисциплины «Языки, алгоритмы и методы программирования» применяется следующий метод интерактивного обучения: метод автоматизированного обучения в

системе автоматического тестирования программ CATS, предъявляющей задания и позволяющей оценить решение.

При выполнении различных видов работ используются следующие технологии:

1. *Проблемное обучение* – стимулирование обучающихся к самостоятельному приобретению знаний, необходимых для решения конкретной проблемы.

2. *Контекстное обучение* – мотивация магистрантов к усвоению знаний путём выявления связей между конкретным знанием и его применением.

3. *Обучение на основе опыта* – активизация познавательной деятельности магистрантов за счёт ассоциации и собственного опыта с предметом обучения

I. СТРУКТУРА И СОДЕРЖАНИЕ ТЕОРЕТИЧЕСКОЙ ЧАСТИ КУРСА

Лекционные занятия (18 часов)

МОДУЛЬ 1. Принципы программирования на языках высокого уровня

Тема 1.1. Парадигмы программирования. Модели разработки программ: водопадная, итерационная. Методы разработки ПО. Способы исполнения программ.

Структурное программирование, объектно-ориентированное программирование, императивное программирование, декларативное программирование.

Сложность как основополагающая проблема программирования, а абстракция-конкретизация и прогнозирование-контроль – как основные ортогональные методы борьбы со сложностью.

Стили программирования: операционный, ситуационный, функциональный, реляционный, параллельный, объектно-ориентированный.

Тема 1.2. Аспекты программирования на языках высокого уровня

Базовые концепции и принципы с различных позиций (технологической, авторской, математической, семиотической и реализаторской).

Управление памятью, ручное и автоматическое управление памятью. Куча. Счётчик ссылок, решение проблемы подсчёта ссылок. Сборка мусора.

Компиляция, интерпретация, байт-код.

Теория тестирования методом белого ящика, тестирование методом чёрного ящика, unit-тест, стресс-тестирование.

Фундаментальные концепции и принципы, воплощенные в современных и перспективных языках программирования

Способы классификации ЯП. Статическая типизация, динамическая типизация, слабая типизация, сильная типизация.

МОДУЛЬ 2. Обзор языков программирования высокого уровня

Тема 2.1 История возникновения языков программирования высокого уровня

Докомпьютерные языки программирования. Появление компьютерных языков программирования. Машинный код, автокод, язык ассемблера.

Появление языков программирования высокого уровня. Язык FORTRAN, современное развитие языка. Язык COBOL, влияние на синтаксис SQL.

Формализация описания языков программирования. Появление структурного программирования, язык Algol, его основные идеи и версии.

Тема 2.2 Классификация языков программирования высокого уровня

Возникновение языков программирования. Уровень языков программирования. Императивные и процедурные языки. Автокоды, язык Ассемблера, FORTRAN, Lisp, Algol 60, COBOL.

Тема 2.3 Поколения языков программирования высокого уровня

Второе поколение языков программирования. Структурное программирование. FORTRAN II, Algol 68, PL/1. Интерпретируемые и компилируемые ЯП. Basic.

Третье поколение ЯП. Слабая, сильная и строгая типизация. Поддержка модульности. Pascal, Modula, Modula-2, Oberon, C.

Перспективы развития в области развития языков программирования.

МОДУЛЬ 3. Основы программирования на разных языках

Тема 3.1. ЯП Python

Модель выполнения. Модель памяти. Стандартные структуры данных ЯП Python: списки, кортежи, словари, множества.

Регулярные выражения. Их использование в ЯП Python. ДКА. Реализация регулярных выражений с использованием НДКА.

Конструкции языка. Элементы ФП в ЯП Питон: функции высших порядков, замыкания. Итераторы. Генераторы.

Список в ЯП Python. Основные виды типов данных в ЯП Python, list comprehension, множественное присваивание, слайсинг, отрицательные индексы списков в ЯП Python, ассоциативный массив.

Тема 3.2 Регулярные выражения

Регулярные выражения: определение, проверка принадлежности языку РВ при помощи НДКА. Виртуальное адресное пространство. Лексикографический порядок сравнения строк, сырая строка. ДКА, НДКА. Квантификаторы, множества символов, обратные ссылки.

Тема 3.3 ЯП Perl.

Назначение языка, его классификация. Основные типы данных ЯП Perl (числа, строки, массивы, ссылки). Понятие контекста. Переменные по умолчанию. Управляющие конструкции (if, for, foreach, while, sub).

Функции ЯП Perl. Передача параметров в функции. Управляющие конструкции. Работа со строками.

II. СТРУКТУРА И СОДЕРЖАНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ КУРСА

Лабораторные занятия (18 часов)

Лабораторное задание №1 Реализация классических алгоритмов на различных языках высокого уровня, использование разных стилей программирования (операционный, ситуационный, функциональный, реляционный, параллельный, объектно-ориентированный)

Лабораторное задание №2 Особенности и базовые концепции языков Симула-67, Смолток, Рефал, Ада, Модула-2, Оберон и пр. Работа со строками на различных этих языках высокого уровня.

Лабораторное задание №3 Реализация наследуемости от модульности до объектной ориентации. Разработка алгоритмов средней сложности на различных языках высокого уровня.

Лабораторное задание №4 Концепция параллелизма в различных языках высокого уровня.

III. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

Учебно-методическое обеспечение самостоятельной работы обучающихся по дисциплине «Языки, алгоритмы и методы программирования» представлено в Приложении 1 и включает в себя:

- план-график выполнения самостоятельной работы по дисциплине, в том числе примерные нормы времени на выполнение по каждому заданию;
- характеристика заданий для самостоятельной работы обучающихся и методические рекомендации по их выполнению;
- требования к представлению и оформлению результатов самостоятельной работы;
- критерии оценки выполнения самостоятельной работы.

IV. КОНТРОЛЬ ДОСТИЖЕНИЯ ЦЕЛЕЙ КУРСА

Изучение дисциплины «Языки, алгоритмы и методы программирования» предусматривает:

- изучение теоретического материала в соответствии с программой, с использованием материала из списка литературы и информационно-методического обеспечения дисциплины;
- выполнение лабораторных работ;
- *текущий контроль* – учет посещения студентами занятий в течение периода обучения и оценка своевременности и качества изучения студентами темы и выполнения лабораторных работ.
- *итоговый контроль* – выведение итоговой оценки за семестр по результатам рейтинга без обязательной сдачи экзамена.

№ п/п	Контролируемые разделы / темы дисциплины	Коды и этапы формирования компетенций		Оценочные средства	
				текущий контроль	промежуточная аттестация
1	Принципы программирования на языках высокого уровня	ОПК-2 ОПК-5 ОПК-8	знает парадигмы программирования, методы разработки ПО, способы исполнения программ. описания языков программирования, особенности структурного программирования.	УО	зачет
			умеет применять на практике водопадную модель разработки программ, итерационную модель разработки программ	ПР-1	Отчет по Лабораторной работе
			владеет технологиями структурного программирования, объектно-ориентированного программирования, императивного программирования, декларативного программирования	ТС	Отчет по Лабораторной работе
2	Обзор языков программирования высокого уровня	ОПК-2 ОПК-5 ОПК-8	знает докомпьютерные и компьютерные языки программирования, машинный код, автокод,	УО-1.	зачет

			язык ассемблера, современное развитие, взаимное влияние языков программирования высокого уровня, принципы формализации описания языков программирования, особенности структурного программирования.		
			умеет определять вид и уровень языка программирования, приводить примеры современных языков программирования указанного класса	ПР-6	Отчет по лабораторной работе
			владеет языком формализации описания языков программирования	ТС	Отчет по лабораторной работе
3	Основы программирования на языке Python	ОПК-2 ОПК-5 ОПК-8	Знает модель выполнения, модель памяти ЯП Python, стандартные структуры данных ЯП Python, конструкции языка, элементы ФП: функции высших порядков, замыкания, итераторы, генераторы, стандартные структуры данных, основные виды типов данных в ЯП Python, определение регулярного выражения, понятие виртуального адресного пространства	УО-2.	зачет
			Умеет использовать в ЯП Python ДКА, стандартные структуры данных ЯП Python, реализовать регулярные выражения с использованием НДКА, квантификаторы,	ПР-2	Отчет по лабораторной работе

			множества символов, обратные ссылки средствами работы со строками		
			Владеет навыками работы со списками, кортежами, list comprehension, множественным присваиванием, слайсингом, отрицательными индексами списков, ассоциативным массивом словарями, множествами	ТС	Отчет по лабораторной работе
4	Основы программирования на языке Perl	ОПК-2 ОПК-5 ОПК-8	знает назначение языка ЯП Perl, его классификацию, основные типы данных, понятие контекста, переменные по умолчанию, управляющие конструкции	УО-2.	зачет
			умеет использовать в программе на ЯП Perl управляющие конструкции (if, for, foreach, while, sub), основные типы данных (числа, строки, массивы, ссылки), функции	ПР-11	Отчет по лабораторной работе
			владеет навыками программирования на языке Perl, механизмом передачи параметров в функциях, навыками работы со строками.	ТС	Отчет по лабораторной работе

- устный опрос (УО): собеседование (УО-1), коллоквиум (УО-2); итоговая презентация (УО-3); круглый стол (УО-4);
- технические средства контроля (ТС);
- письменные работы (ПР): тесты (ПР-1), контрольные работы (ПР-2), эссе (ПР-3), рефераты (ПР-4), курсовые работы (ПР-5), научно-учебные отчеты по практикам (ПР-6), конспект (ПР-7), проект (ПР-9). Разноуровневые задачи и задания (ПР-11) и т.п.

Типовые контрольные задания, методические материалы, определяющие процедуры оценивания знаний, умений и навыков и (или) опыта деятельности, а также критерии и показатели, необходимые для оценки знаний, умений, навыков и характеризующие этапы формирования компетенций в процессе освоения образовательной программы, представлены в Приложении 2.

V. СПИСОК УЧЕБНОЙ ЛИТЕРАТУРЫ И ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература (электронные и печатные издания)

1. Головин И. Г., Волкова И. А. Языки и методы программирования: учебник для вузов/ И. Г Головин. Москва: Академия, 2012 – 304 с.
2. Молдованова, О. В. Языки программирования и методы трансляции [Электронный ресурс]: учебное пособие / О. В. Молдованова. — Электрон. текстовые данные. — Новосибирск: Сибирский государственный университет телекоммуникаций и информатики, 2012. — 134 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/54809.html>
3. Кауфман, В.Ш. Языки программирования. Концепции и принципы [Электронный ресурс] / В.Ш. Кауфман. — Электрон. дан. — Москва : ДМК Пресс, 2010. — 464 с. — Режим доступа: <https://e.lanbook.com/book/1270>
4. Серебряков, В.А. Теория и реализация языков программирования [Электронный ресурс] : учебное пособие / В.А. Серебряков. — Электрон. дан. — Москва : Физматлит, 2012. — 236 с. — Режим доступа: <https://e.lanbook.com/book/5294>. — Загл. с экрана.
5. Языки программирования: Учебное пособие / О.Л. Голицына, Т.Л. Партыка, И.И. Попов. - 3-е изд., перераб. и доп. - М.: Форум: ИНФРА-М, 2015. - 400 с.: 60x90 1/16. - (Профессиональное образование). (переплет) ISBN 978-5-91134-744-4 - Режим доступа: <http://znanium.com/catalog/product/493421>

6. Алгоритмизация и языки программирования [Электронный ресурс] : учебно-методический комплекс / сост. О. С. Ахметова, Б. К. Тульбасова, А. Ж. Абишева. — Электрон. текстовые данные. — Алматы : Нур-Принт, Казахский национальный педагогический университет имени Абая, 2012. — 165 с. — 9965-894-95-7. — Режим доступа: <http://www.iprbookshop.ru/67008.html> ISO/IEC 9075-2:2011 Information technology — Database languages — SQL — Part 2: Foundation (SQL/Foundation). — ISO/IEC, 2011 — 1483 с.

Дополнительная литература
(печатные и электронные издания)

1. Формальные языки и компиляторы/МалявкоА.А. - Новосиб.: НГТУ, 2014. - 431 с.: ISBN 978-5-7782-2318-9 - Режим доступа: <http://znanium.com/catalog/product/548152> MariaDB Documentation.
2. Языки программирования : учеб. пособие / О.Л. Голицына, Т.Л. Партыка, И.И. Попов. — 3-е изд., перераб. и доп. — М. : ФОРУМ : ИНФРА-М, 2018. — 399 с. — (Среднее профессиональное образование). - Режим доступа: <http://znanium.com/catalog/product/973007>
3. Фомичев В. С. Формальные языки, грамматики и автоматы, СПбГЭУ "ЛЭТИ". – 2013.

Перечень ресурсов информационно-телекоммуникационной сети
«Интернет»

1. Электронный журнал успеваемости система AWorks//
2. Система автоматического тестирования программ CATS
3. Сообщество пользователей SQL [<http://sql.ru>].
4. Утилита для администрирования данных FlameRobin [<http://flamerobin.org>]

VI. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

На изучение дисциплины отводится 36 часов аудиторных (лекционных и лабораторных) занятий. На занятиях перед выдачей индивидуальных заданий преподаватель объясняет теоретический материал по заданной теме. Вводит основные требования к его выполнению. Приводит примеры. Необходимо поддерживать непрерывный контакт с аудиторией, отвечать на возникающие у студентов вопросы. На лабораторных занятиях преподаватель разбирает на примерах принципы и аспекты реализации задания по заданной теме.

По ряду тем магистрантам предлагается работать самостоятельно, выполняя полный обзор по теме. Преподаватель контролирует работу студентов, отвечает на возникающие вопросы, предоставляет список литературных источников для освоения темы, а также перечень вопросов для самопроверки. Если знаний полученных в аудитории оказалось недостаточно, магистрант может самостоятельно повторно просмотреть методические указания.

После выполнения задания, магистрант оформляет материал в форме программного кода и отправляет его на проверку преподавателю по электронной почте, либо предъявляет на компьютере во время занятия. Магистрант отвечает устно во время занятия по заданной теме.

По данному курсу разработаны учебные материалы. Для успешного достижения учебных целей занятий должны выполняться следующие основные требования:

- соответствие действий обучающихся ранее изученным на лекционных и семинарских занятиях методикам и методам.
- максимальное приближение действий магистрантов к реальным, соответствующим будущим функциональным обязанностям.

- поэтапное формирование умений и навыков, т.е. движение от знаний к умениям и навыкам, от простого к сложному и т.д..

- использование при работе на тренажерах или действующей технике фактических документов, технологических карт, бланков и т.п.

- выработка соответствующих индивидуальных и коллективных умений и навыков.

Магистрант должен:

- научиться работать с книгой, документацией и схемами, пользоваться справочной и научной литературой.

- научиться работать с электронными литературными источниками.

- формировать умение учиться самостоятельно, т.е. овладевать методами, способами и приемами самообучения, саморазвития и самоконтроля.

Рекомендации по подготовке к зачету:

Рекомендуется еще раз самостоятельно ответить на вопросы для самопроверки, приведенные в методических материалах.

При ответе на каждый вопрос экзамена магистрант должен продемонстрировать знание определения указанного понятия, связанных с ним особенностей реализации и применения, умение реализовать указанную операцию, а также навыки иллюстрации теоретических принципов на предложенных простых примерах.

VII. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

<p>Компьютерный класс: Проектор DLP, 3000 ANSI Lm, WXGA 1280x800, 2000:1 EW330U Mitsubishi,; Системный блок с монитором. Процессор: Intel I5-8600k 3.6Ghz, оперативная память: 32gb, жесткий диск: 1ТБ, графический ускоритель: Nvidia GTX 1080 Беспроводные ЛВС для обучающихся обеспечены системой на базе точек доступа 802.11a/b/g/n 2x2 MIMO(2SS).</p> <p>Специализированное ПО: Visual Studio 2019, Eclipse, Anaconda, Система автоматического тестирования программ CATS/</p>	<p>690922, Приморский край, г. Владивосток, о. Русский, п. Аякс, 10, г. Владивосток, о. Русский, п. Аякс, корпус G, ауд. G468</p>
--	---



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЦИФРОВОЙ ЭКОНОМИКИ

**УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ
САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ**

Направление подготовки

09.04.01 «Информатика и вычислительная техника»

**Владивосток
2019**

План-график выполнения самостоятельной работы по дисциплине

№ п/п, название	Дата/сроки выполнения	Вид СРС	Примерные нормы времени на выполнение	Форма контроля
1. Принципы и методы программирования, использование разных стилей программирования (операционный, ситуационный, функциональный, реляционный, параллельный, объектно-ориентированный)	Третья неделя семестра	ИДЗ	2 недели	Коллоквиум
2. Классификация и описание языков программирования,	Четвертая неделя семестра	ИДЗ	1 неделя	Коллоквиум
3. Особенности и базовые концепции языков Симула-67, Смолток, Рефал, Ада, Модула-2, Оберон и пр. Работа со строками на различных языках высокого уровня.	Пятая неделя семестра	ИДЗ	1 неделя	Коллоквиум
4. Реализация классических алгоритмов на различных языках высокого уровня.	Шестая неделя семестра	ИДЗ	1 неделя	Проверка программы
5. Реализация алгоритмов работы со строками на различных языках высокого	Седьмая неделя семестра	ИДЗ	3 неделя	Проверка программы

уровня				
6. Реализация наследуемости от модульности до объектной ориентации. Разработка алгоритмов средней сложности на различных языках высокого уровня.	Десятая неделя семестра	ИДЗ	2 недели	Проверка программы
7. Концепция параллелизма в различных языках высокого уровня Концепция параллелизма в различных языках высокого уровня	Пятнадцатая неделя семестра	ИДЗ	2 недели	Проверка программы
8. Подготовка к зачету	Последняя неделя	ИДЗ	1 неделя	Итоговое тестирование

Критерии оценивания

В течение семестра студентам последовательно выдается набор из 4-х лабораторных работ, каждая из которых имеет вес от 15%.до 20%. Посещаемость занятий также учитывается и имеет вес 10%. Для получения зачета с оценкой «отлично» необходимо иметь итоговый балл не ниже 80%, зачета с оценкой «хорошо» – необходимо иметь итоговый балл не ниже 65%, зачета с оценкой «удовлетворительно» – необходимо иметь итоговый балл не ниже 50%,

Характеристика заданий самостоятельной работы

Самостоятельная работа направлена на развитие интеллектуальных умений, комплекса универсальных (общекультурных) и профессиональных компетенций, повышение творческого потенциала магистрантов и заключается в:

- поиске, анализе, структурировании и презентации информации;
- разработке учебного программного продукта;

– исследовательской работе и участии в научных студенческих семинарах и олимпиадах;

– анализе научных публикаций по заранее определенной преподавателем теме.

Содержание самостоятельной работы студентов по дисциплине

1. Изучение аспектов теории, не уложившихся в лекции.
2. Разработка алгоритмов и программ при выполнении лабораторных работ.
3. Подготовка к зачету.

Требования к представлению и оформлению результатов самостоятельной работы

1. Устный ответ по указанной теме.
2. Исходный код программ заданий по соответствующим темам.

КРИТЕРИИ ОЦЕНКИ ВЫПОЛНЕНИЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Оценка результатов самостоятельной работы организуется как единство двух форм: самоконтроль и контроль со стороны преподавателя.

Критерии оценки:

1. Обзор литературных источников (10 баллов)
2. Качество устного изложения содержания темы (10 баллов)
3. Выполнение каждого лабораторного задания (10 баллов)

Учебно-методические материалы для подготовки к лабораторным работам

Лабораторное задание №1 Реализация классических алгоритмов на различных языках высокого уровня, использование разных стилей программирования (операционный, ситуационный, функциональный, реляционный, параллельный, объектно-ориентированный)

Реализация каждого из заданий выполняется одновременно на двух языках: Python и Perl. Задания подразумевают использование базовых типов данных, массивов, управляющих конструкций, множественного присваивания.

Для каждого из заданий определяется использование определенного стиля программирования (операционный, ситуационный, функциональный, реляционный, параллельный, объектно-ориентированный).

Лабораторное задание №2 Реализация алгоритмов работы со строками на различных языках высокого уровня. Особенности и базовые концепции различных языков.

Реализация каждого из заданий выполняется на одном из языков: Симула-67, Смолток, Рефал, Ада, Модула-2, Оберон и пр. Работа со строками на различных этих языках высокого уровня.. В заданиях требуется использование регулярных выражений с использованием НДКА, выбор структур данных, квантификаторов, множества символов, средств работы со строками.

Лабораторное задание №3 Реализация наследуемости от модульности до объектной ориентации. Разработка алгоритмов средней сложности на различных языках высокого уровня.

Реализация каждого из заданий предполагает использование наследуемости. Задания требуют работу с моделями динамического распределения памяти. Темы Лабораторной проработки: списки, кортежи,

отрицательные индексы списков, list omprehension, слайсинг, ассоциативный массив словари, множества.

Лабораторное задание №4 Объектно-ориентированное программирование в различных языках высокого уровня. Концепция параллелизма в различных языках высокого уровня

Задания на темы: описание классов, объекты классов, функции, механизмы передачи параметров.

ГЛОССАРИЙ

Аббревиатура

OLAP (англ. online analytical processing, аналитическая обработка в реальном времени) — технология обработки информации, включающая составление и динамическую публикацию отчётов и документов.

Общие определения

Термин "Языковые конструкции" часто используется как синоним структуры управления и не следует путать с функцией. Конструкция языка является синтаксически допустимой частью программы, которая может быть сформирована из одного или нескольких лексических токенов в соответствии с правилами языка программирования.

Лексема (лингвистика) — слово как абстрактная единица естественного языка.

Лексема (информатика) — последовательность допустимых символов языка программирования, имеющая смысл для транслятора.

Стиль программирования — это набор правил, которым следует программист (осознано или потому, что "так делают другие") в процессе своей работы.

Очевидно, что хороший программист должен следовать правилам хорошего стиля.

Объектно-ориентированный стиль программирования обеспечивает наличие следующих свойств: повторная используемость;

- расширяемость;
- устойчивость к неправильным данным;
- системность.

Концептуальное проектирование компиляторов — сбор, анализ и редактирование требований к функциональности компилятора, соответствующего формальному описанию языка программирования.

Логическое проектирование компиляторов — преобразование требований к функциональности компилятора в описание архитектуры в структуры данных.

Модель данных - интегрированный набор понятий для описания данных, связей между ними и ограничений, накладываемых на данные в некоторой организации.

Транзакция — в информатике, группа последовательных операций, которая представляет собой логическую единицу работы с данными.

Транзакция представляет собой набор действий, выполняемых отдельным пользователем или прикладной программой с целью доступа или изменения содержимого.

Файл — простой набор записей, содержащих логически связанные данные.

Файловая система (картотека) — набор программ, которые выполняют для пользователей некоторые операции, каждая программа определяет свои собственные данные и управляет ими.

Физическое проектирование компиляторов — определение особенностей хранения данных, инструментальных средств, модулей, методов доступа и т.д.

Хеширование — преобразование входного массива данных произвольной длины в выходную битовую строку фиксированной длины.

Целостность программирования компиляторов — соответствие имеющейся в архитектуре информации её внутренней логике, структуре и всем явно заданным правилам.



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ШКОЛА ЦИФРОВОЙ ЭКОНОМИКИ

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по дисциплине «Языки, алгоритмы и методы программирования»

Направление подготовки
09.04.01 «Информатика и вычислительная техника»

Владивосток
2019

Сформированность каждой компетенции в рамках освоения данной дисциплины оценивается по трехуровневой шкале:

- пороговый уровень является обязательным для всех обучающихся по завершении освоения дисциплины;
- продвинутый уровень характеризуется превышением минимальных характеристик сформированности компетенции по завершении освоения дисциплины;
- эталонный уровень характеризуется максимально возможной выраженностью компетенции и является важным качественным ориентиром для самосовершенствования.

Уровень сформированности каждой компетенции на различных этапах ее формирования в процессе освоения данной дисциплины оценивается в ходе текущего контроля успеваемости, представлен различными видами оценочных средств.

Результаты обучения (компетенции из ФГОС)	Знает	Умеет	Владеет
ОПК-2 ОПК-5 ОПК-8	парадигмы программирования, методы разработки программного обеспечения, способы исполнения программ, формы описания языков программирования, особенности стилей программирования.	определять вид и уровень языка программирования, приводить примеры современных языков программирования указанного класса, реализовывать алгоритмы на различных высокоуровневых языках программирования	технологиями структурного программирования, объектно-ориентированного программирования, императивного программирования, декларативного программирования
Эталонный	Основной и дополнительный материал, предусмотренный компетенцией, без ошибок и погрешностей	умеет квалифицированно определять вид и уровень языка программирования, приводить примеры современных языков программирования указанного класса, реализовывать	всеми навыками, демонстрируя их не только в стандартных ситуациях, но и при решении нестандартных задач

		алгоритмы на различных высокоуровневых языках программирования	
Продвинутый	основной материал, предусмотренный компетенцией, без ошибок и погрешностей	умеет с незначительными погрешностями определять вид и уровень языка программирования, приводить примеры современных языков программирования указанного класса, реализовывать алгоритмы на различных высокоуровневых языках программирования	основными навыками, демонстрируя их в стандартных ситуациях, в том числе при решении дополнительных задач
Пороговый	большинство основных понятий, изучаемых в рамках дисциплины	Умеет с погрешностями определять вид и уровень языка программирования, приводить примеры современных языков программирования указанного класса, реализовывать алгоритмы на различных высокоуровневых языках программирования	некоторыми основными навыками, демонстрируя их в стандартных ситуациях

**Материалы зачета по дисциплине
«ЯЗЫКИ, АЛГОРИТМЫ И МЕТОДЫ ПРОГРАММИРОВАНИЯ»**

Вопросы к зачету

1. Методы разработки ПО.
2. Способы исполнения программ. Парадигмы программирования.
3. Способы классификации ЯП.
4. Обзор ЯП, классификация некоторых из них (не менее 10).

5. Регулярные выражения - практика: квантификаторы, множества символов, обратные ссылки.
6. Регулярные выражения - теория: определение, проверка принадлежности языку РВ при помощи НДКА.
7. Ручное управления памятью. Куча. Проблемы ручного управления. Автоматическое управление памятью. Сборка мусора. Подсчёт ссылок.
8. ЯП Python. Назначение языка, его классификация. Модель данных. Модель выполнения.
9. ЯП Python. Типы данных (числа, строки, списки, словари, множества, кортежи). Индексация списков, слайсинг.
10. ЯП Python. Конструкции языка (if, for, while, def, list comprehension). Передача параметров в функции.
11. ЯП Python. Итераторы. Генераторы. Функции высших порядков. Замыкания.
12. ЯП Perl. Назначение языка, его классификация. Контекст. Переменные по умолчанию.
13. ЯП Perl. Основные типы данных (числа, строки, массивы, ссылки). Передача параметров в функции.
14. ЯП Perl. Управляющие конструкции (if, for, foreach, while, sub). Работа со строками.

Примеры дополнительных вопросов

Общие вопросы

1. что такое водопадная модель разработки?
2. что такое итерационная модель разработки?
3. что такое тестирование методом белого ящика?
4. что такое тестирование методом чёрного ящика?
5. что такое unit-тест?
6. что такое стресс-тестирование?

7. что такое компиляция?
8. что такое интерпретация?
9. что такое байт-код?
10. что такое структурное программирование?
11. что такое объектно-ориентированное программирование?
12. что такое императивное программирование?
13. что такое декларативное программирование?
14. что такое статическая типизация?
15. что такое динамическая типизация?
16. что такое слабая типизация?
17. что такое сильная типизация?
18. что такое ручное управление памятью?
19. что такое автоматическое управление памятью?

Вопросы по обработке строк и РВ

1. что такое лексикографический порядок сравнения строк?
2. что такое сырая строка?
3. что такое РВ?
4. что такое ДКА?
5. что такое НДКА?
6. что такое квантификатор в РВ?
7. как задать множество символов в РВ?
8. что означает * в РВ?
9. что означает | в РВ?
10. что означает ? в РВ?

11. что означает . в РВ?

12. как составить РВ, принимающее...?

Вопросы по управлению памятью

1. что такое виртуальное адресное пространство?

2. что такое куча?

3. что такое счётчик ссылок?

4. какую проблему подсчёт ссылок решить не может?

5. что такое сборка мусора?

Вопросы по ЯП Python

1. что представляет собой список в ЯП Python?

2. передача аргументов в функции в ЯП Python происходит по ссылке или

3. о значению?

4. какие два основных вида типов данных в ЯП Python?

5. что такое list comprehension?

6. что такое множественное присваивание?

7. что такое слайсинг?

8. что означают отрицательные индексы списков в ЯП Python?

9. что такое функции высших порядков?

10. что такое замыкание для функций?

11. что такое генератор в ЯП Python?

12. что такое итератор?

13. что такое ассоциативный массив?

14. что означает "abc"*4 в ЯП Python

Вопросы по ЯП Perl

1. как конкатенировать строки в ЯП Perl?
2. что такое строковая интерполяция?
3. почему в ЯП Perl умножение строки на число обозначается как `x` а не как более естественное `*`?
4. почему в ЯП Perl операторы сравнения строк отличаются от операторов сравнения чисел?
5. передача аргументов в функции в ЯП Perl происходит по ссылке или по значению?
6. что означает `$_` в ЯП Perl?
7. что означает `@_` в ЯП Perl?