

Федеральное государственное бюджетное образовательное учреждение
высшего образования
"Дальневосточный государственный университет путей сообщения"
(ДВГУПС)

УТВЕРЖДАЮ

Зав.кафедрой

(к202) Информационные технологии и
системы

Попов М.А., канд. техн.
наук, доцент



27.05.2022

РАБОЧАЯ ПРОГРАММА

дисциплины **Функционально-логическое программирование**

09.03.04 Программная инженерия

Составитель(и): ст. преподаватель, Рыбкина О.В.; канд. техн. наук, доцент, Попов М.А.

Обсуждена на заседании кафедры: (к202) Информационные технологии и системы

Протокол от 18.05.2022г. № 5

Обсуждена на заседании методической комиссии учебно-структурного подразделения: Протокол от 27.05.2022 г. № 7

г. Хабаровск
2022 г.

Визирование РПД для исполнения в очередном учебном году

Председатель МК РНС

__ _____ 2023 г.

Рабочая программа пересмотрена, обсуждена и одобрена для исполнения в 2023-2024 учебном году на заседании кафедры (к202) Информационные технологии и системы

Протокол от __ _____ 2023 г. № __
Зав. кафедрой Попов М.А., канд. техн. наук, доцент

Визирование РПД для исполнения в очередном учебном году

Председатель МК РНС

__ _____ 2024 г.

Рабочая программа пересмотрена, обсуждена и одобрена для исполнения в 2024-2025 учебном году на заседании кафедры (к202) Информационные технологии и системы

Протокол от __ _____ 2024 г. № __
Зав. кафедрой Попов М.А., канд. техн. наук, доцент

Визирование РПД для исполнения в очередном учебном году

Председатель МК РНС

__ _____ 2025 г.

Рабочая программа пересмотрена, обсуждена и одобрена для исполнения в 2025-2026 учебном году на заседании кафедры (к202) Информационные технологии и системы

Протокол от __ _____ 2025 г. № __
Зав. кафедрой Попов М.А., канд. техн. наук, доцент

Визирование РПД для исполнения в очередном учебном году

Председатель МК РНС

__ _____ 2026 г.

Рабочая программа пересмотрена, обсуждена и одобрена для исполнения в 2026-2027 учебном году на заседании кафедры (к202) Информационные технологии и системы

Протокол от __ _____ 2026 г. № __
Зав. кафедрой Попов М.А., канд. техн. наук, доцент

Рабочая программа дисциплины **Функционально-логическое программирование**
разработана в соответствии с ФГОС, утвержденным приказом Министерства образования и науки Российской Федерации от 19.09.2017 № 920

Квалификация **бакалавр**

Форма обучения **очная**

ОБЪЕМ ДИСЦИПЛИНЫ (МОДУЛЯ) В ЗАЧЕТНЫХ ЕДИНИЦАХ С УКАЗАНИЕМ КОЛИЧЕСТВА АКАДЕМИЧЕСКИХ ЧАСОВ, ВЫДЕЛЕННЫХ НА КОНТАКТНУЮ РАБОТУ ОБУЧАЮЩИХСЯ С ПРЕПОДАВАТЕЛЕМ (ПО ВИДАМ УЧЕБНЫХ ЗАНЯТИЙ) И НА САМОСТОЯТЕЛЬНУЮ РАБОТУ ОБУЧАЮЩИХСЯ

Общая трудоемкость **4 ЗЕТ**

Часов по учебному плану	144	Виды контроля в семестрах:
в том числе:		зачёты с оценкой 8
контактная работа	36	
самостоятельная работа	108	

Распределение часов дисциплины по семестрам (курсам)

Семестр (<Курс>.<Семес тр на курсе>)	8 (4.2)		Итого	
	8			
Неделя	8			
Вид занятий	УП	РП	УП	РП
Лекции	16	16	16	16
Лабораторные	16	16	16	16
Контроль самостоятельной работы	4	4	4	4
В том числе инт.	4	4	4	4
Итого ауд.	32	32	32	32
Контактная работа	36	36	36	36
Сам. работа	108	108	108	108
Итого	144	144	144	144

1. АННОТАЦИЯ ДИСЦИПЛИНЫ (МОДУЛЯ)

1.1	<p>Задачи искусственного интеллекта, проблемы и особенности решения. Современные парадигмы программирования. Представление знаний. Методологии программирования, методологии логического и функционального программирования. Процедурные и декларативные языки программирования. Языки функционального программирования. Преимущества и недостатки функционального подхода в программировании. Классификация функций. Абстракция в функциональном программировании. Язык Lisp, его диалекты. Основные элементы языка Lisp. Атомы, точечные пары, S-выражения, списки. Атомы и константы. Правила записи идентификаторов. Правила записи списков. Вычисляемые S-выражения. Внутреннее представление списков. Функции в Lisp-e. Селекторы, конструкторы, предикаты. Особенность работы предикатов сравнения. Функции работы со списками. Использование функционалов и рекурсии. Использование параметров при определении функций. Функции более высокого порядка. Языки логического программирования. Отношение, как способ фиксации знаний. Исчисление высказываний и исчисление предикатов. Кванторные операции. Принцип резолюции. Язык Пролог. Простейшие логические программы. Понятие домена. Стандартные домены. Структура программы на языке Пролог. Основные элементы и основные структуры языка Пролог (факты, правила и вопросы). Простейший интерпретатор логических программ. Протокол интерпретатора. Виды импликации. Декларативная и процедурная семантика логических программ. Понятие процедуры в логической программе. Иллюстрация различия декларативной и процедурной семантики программ. Отождествление и сопоставление. Правила отождествления. Механизм унификации. Механизм возврата. Построение дерева вывода. Порядок утверждений в логической программе и поиск решений. Конъюнкция и дизъюнкция целей. Рекурсивные процедуры и их применение. Граничные условия. Стандартные предикаты <code>cat</code> и <code>fail</code>. Список. Формы записи списков в Пролог. Работа со списками и рекурсивные процедуры их обработки средствами Пролога. Примеры программ.</p>
-----	--

2. МЕСТО ДИСЦИПЛИНЫ (МОДУЛЯ) В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Код дисциплины:	Б1.В.10
2.1	Требования к предварительной подготовке обучающегося:
2.1.1	Объектно-ориентированное программирование
2.1.2	Математическая логика и теория алгоритмов
2.2	Дисциплины и практики, для которых освоение данной дисциплины (модуля) необходимо как предшествующее:
2.2.1	Интеллектуальные системы и технологии

3. ПЕРЕЧЕНЬ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ (МОДУЛЮ), СООТНЕСЕННЫХ С ПЛАНИРУЕМЫМИ РЕЗУЛЬТАТАМИ ОСВОЕНИЯ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

УК-1: Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач
Знать:
Методики поиска, сбора и обработки информации; актуальные российские и зарубежные источники информации в сфере профессиональной деятельности; метод системного анализа.
Уметь:
Применять методики поиска, сбора и обработки информации; осуществлять критический анализ и синтез информации, полученной из разных источников; применять системный подход для решения поставленных задач.
Владеть:
Методами поиска, сбора и обработки, критического анализа и синтеза информации; методикой системного подхода для решения поставленных задач.

ПК-10: Владение навыками использования различных технологий разработки программного обеспечения

Знать:
Современные технологии разработки ПО (структурное, объектно-ориентированное)
Уметь:
Использовать современные технологии разработки ПО
Владеть:
Навыками использования современные технологии разработки ПО

4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ), СТРУКТУРИРОВАННОЕ ПО ТЕМАМ (РАЗДЕЛАМ) С УКАЗАНИЕМ ОТВЕДЕННОГО НА НИХ КОЛИЧЕСТВА АКАДЕМИЧЕСКИХ ЧАСОВ И ВИДОВ УЧЕБНЫХ ЗАНЯТИЙ

Код занятия	Наименование разделов и тем /вид занятия/	Семестр / Курс	Часов	Компетенции	Литература	Инте ракт.	Примечание
	Раздел 1. Лекции						

1.1	Введение в функциональное программирование. Задачи искусственного интеллекта, проблемы и особенности решения. Современные парадигмы программирования. Представление знаний. Методологии программирования, методологии логического и функционального программирования. /Лек/	8	2	УК-1 ПК-10	Л1.2Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
1.2	Процедурные и декларативные языки программирования. Языки функционального программирования. Преимущества и недостатки функционального подхода в программировании. Классификация функций. Абстракция в функциональном программировании. /Лек/	8	2	УК-1 ПК-10	Л1.2Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	2	Лекция-консультация
1.3	Язык Lisp, его диалекты. Основные элементы языка Lisp. Атомы, точечные пары, S-выражения, списки. Атомы и константы. Правила записи идентификаторов. Правила записи списков. Вычисляемые S-выражения. Внутреннее представление списков. Функции в Lisp-e. /Лек/	8	2	УК-1 ПК-10	Л1.2Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	2	Лекция-консультация
1.4	Языки логического программирования. Отношение, как способ фиксации знаний. Исчисление высказываний и исчисление предикатов. Кванторные операции. Принцип резолюции. /Лек/	8	2	УК-1 ПК-10	Л1.2Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
1.5	Селекторы, конструкторы, предикаты. Особенность работы предикатов сравнения. Функции работы со списками. Использование функционалов и рекурсии. Использование параметров при определении функций. Функции более высокого порядка. /Лек/	8	2	УК-1 ПК-10	Л1.2Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
1.6	Простейший интерпретатор логических программ. Протокол интерпретатора. Виды импликации. Декларативная и процедурная семантика логических программ. Понятие процедуры в логической программе. Иллюстрация различия декларативной и процедурной семантики программ. /Лек/	8	2	УК-1 ПК-10	Л1.1Л2.1 Э1 Э2 Э3 Э4 Э5 Э6	0	
1.7	Отождествление и сопоставление. Правила отождествления. Механизм унификации. Механизм возврата. /Лек/	8	2	УК-1 ПК-10	Л1.1Л2.1 Э1 Э2 Э3 Э4 Э5 Э6	0	
1.8	Построение дерева вывода. Порядок утверждений в логической программе и поиск решений. Конъюнкция и дизъюнкция целей. Рекурсивные процедуры и их применение. /Лек/	8	2	УК-1 ПК-10	Л1.1Л2.1 Э1 Э2 Э3 Э4 Э5 Э6	0	
Раздел 2. Практические занятия							
2.1	Программирование, не требующее рекурсии /Лаб/	8	2	УК-1 ПК-10	Л1.2 Л1.1Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
2.2	Простая рекурсия /Лаб/	8	2	УК-1 ПК-10	Л1.2 Л1.1Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	

2.3	Сложные формы рекурсии /Лаб/	8	2	УК-1 ПК-10	Л1.2 Л1.1Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
2.4	Программы со структурами /Лаб/	8	2	УК-1 ПК-10	Л1.2 Л1.1Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
2.5	Локальные переменные и функции. Программирование функционалов /Лаб/	8	2	УК-1 ПК-10	Л1.2 Л1.1Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
2.6	Язык Пролог. Простейшие логические программы. Понятие домена. Стандартные домены. Структура программы на языке Пролог. Основные элементы и основные структуры языка Пролог (факты, правила и вопросы). /Лаб/	8	2	УК-1 ПК-10	Л1.2 Л1.1Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
2.7	Пролог. Рекурсивное определение правил. Формирование рекурсивных предикатов. Граничные условия. Стандартные предикаты cat и fail. /Лаб/	8	2	УК-1 ПК-10	Л2.1 Э1 Э2 Э3 Э4 Э5 Э6	0	
2.8	Пролог. Список. Формы записи списков в Пролог. Работа со списками и рекурсивные процедуры их обработки средствами Пролога. Примеры программ. Программы со структурами. /Лаб/	8	2	УК-1 ПК-10	Л2.1 Э1 Э2 Э3 Э4 Э5 Э6	0	
Раздел 3. Самостоятельная работа студента							
3.1	подготовка к лекциям /Ср/	8	16	УК-1 ПК-10	Л1.2 Л1.1Л2.1 Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
3.2	подготовка к лабораторным работам /Ср/	8	48	УК-1 ПК-10	Л1.2 Л1.1Л2.1 Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
3.3	подготовка к зачёту /Ср/	8	8	УК-1 ПК-10	Л1.2 Л1.1Л2.1 Л2.2 Э1 Э2 Э3 Э4 Э5 Э6	0	
3.4	/ЗачётСОц/	8	36			0	

5. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Размещены в приложении

6. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

6.1. Рекомендуемая литература

6.1.1. Перечень основной литературы, необходимой для освоения дисциплины (модуля)

	Авторы, составители	Заглавие	Издательство, год
Л1.1	Сайбель П.	Практическое использование Common Lisp	Москва: ДМК Пресс, 2015, http://e.lanbook.com/books/element.php?p11_id=58686
Л1.2	Сошников Д. В.	Функциональное программирование на F#	Москва: ДМК Пресс, 2011, http://e.lanbook.com/books/element.php?p11_cid=25&p11_id=1274

6.1.2. Перечень дополнительной литературы, необходимой для освоения дисциплины (модуля)			
	Авторы, составители	Заглавие	Издательство, год
Л2.1	Клоксин У., Меллиш К.	Программирование на языке пролог: пер. с англ.	Москва: Мир, 1987,
Л2.2	Душкин Р. В.	Функциональное программирование на языке Haskell (+CD)	Москва: ДМК Пресс, 2008, http://e.lanbook.com/books/element.php?p11_cid=25&p11_id=1247

6.2. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)		
Э1	Журнал «Практика функционального программирования»	http://fprog.ru/
Э2	Обзор литературы о функциональном программировании	http://alexott.net/ru/fp/books/
Э3	Обзор литературы о логическом программировании.	http://fprog.ru/2009/issue1/alexott-literature-overview/
Э4	Издательство «Открытые системы»	http://www.osp.ru
Э5	Издание о высоких технологиях	http://www.cnews.ru
Э6	Сообщество IT-профессионалов	http://habrahabr.ru/

6.3 Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем (при необходимости)

6.3.1 Перечень программного обеспечения	
Scheme Racket, свободно распространяемое ПО	
SWI-Prolog, свободно распространяемое ПО	
ПО DreamSpark Premium Electronic Software Delivery - Подписка на программное обеспечение компании Microsoft. В подписку входят все продукты Microsoft за исключением Office, контракт 203	
Free Conference Call (свободная лицензия)	
Zoom (свободная лицензия)	
6.3.2 Перечень информационных справочных систем	
Профессиональная база данных, информационно-справочная система Гарант - http://www.garant.ru	
Профессиональная база данных, информационно-справочная система КонсультантПлюс - http://www.consultant.ru	

7. ОПИСАНИЕ МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЙ БАЗЫ, НЕОБХОДИМОЙ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

Аудитория	Назначение	Оснащение
201	Компьютерный класс для практических и лабораторных занятий, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также для самостоятельной работы	столы, стулья, компьютерная техника с возможностью подключения к сети Интернет, свободному доступу в ЭБС и ЭИОС, проектор
304	Учебная аудитория для проведения занятий лекционного типа	комплект учебной мебели: столы, стулья, интерактивная доска, мультимедийный проектор, компьютер, система акустическая
424	Учебная аудитория для проведения лабораторных и практических занятий, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации. Лаборатория электронных устройств регистрации и передачи	комплект учебной мебели, мультимедийный проектор, экран, компьютер преподавателя
108	Компьютерный класс для практических и лабораторных занятий, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также для самостоятельной работы	комплект учебной мебели: столы, стулья, компьютерная техника с возможностью подключения к сети Интернет, свободному доступу в ЭБС и ЭИОС: Intel(R) Core(TM) i5-4670 CPU @ 3.40GHz, 8 Gb, 1Tb, DVD+RW, ЖК 23", проектор, экран для проектора

8. МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ (МОДУЛЯ)

Занятия по дисциплине «Функционально-логическое программирование» реализуются с использованием как активных, так и интерактивных форм обучения, позволяющих взаимодействовать в процессе обучения не только преподавателю и студенту, но и студентам между собой.

В соответствии с учебным планом для слушателей дневного отделения изучение курса предполагает выполнение установленного комплекса практических работ (в аудитории), а также расчетно-графических работ (самостоятельно) в течение одного семестра.

Необходимый и достаточный для успешного выполнения практической работы объем теоретического материала изложен в методических указаниях или выдается на занятиях преподавателем. При выполнении задания должны соблюдаться все требования, изложенные в методических указаниях.

Практическая работа считается выполненной, если студент смог продемонстрировать на лабораторном стенде – ПК с соответствующим программным обеспечением правильный результат и пояснить ход выполнения работы.

При выполнении РГР студент должен руководствоваться лекционным материалом, а также обязательно использовать другие литературные источники по своему усмотрению, в частности, приведенные в РПД дисциплины. В ходе выполнения каждой РГР студент на изучаемых ранее языках и технологиях программирования должен создать несколько вариантов тематического (в соответствии с заданным вариантом) приложения, реализующего предусмотренные заданием функционал. После завершения выполнения каждой РГР слушатель допускается к защите и демонстрации приложения. Защита РГР проходит в форме собеседования по вопросам, касающимся причин применения и особенностей реализации предложенных программных решений.

Текущий контроль знаний студентов осуществляется на практических занятиях в соответствии с тематикой работ путем устного опроса, а также при защите РГР. Кроме этого в середине семестра проводится промежуточная аттестация студентов дневной формы обучения, согласно рейтинговой системе ДВГУПС.

Студент, своевременно выполнивший все предусмотренные программой практические работы и защитивший РГР допускается к зачету. Выходной контроль знаний слушателей осуществляется на зачете в конце семестра в форме собеседования или тестирования.

Расчетно-графическая работа №1 ФУНКЦИОНАЛЬНОЕ ПРОГРАММИРОВАНИЕ

Задание №1

Задание состоит из трех задач, в которых требуется составить программы на Лиспе. В первой задаче не требуется рекурсия, остальные две задачи требуют применения простой рекурсии. При составлении программ (если не оговорено противное) можно использовать все встроенные функции Лиспа.

Отладку программ можно осуществлять с помощью функции трассировки (`trace <имя функции>`), трассировка функции отключается - (`untrace <имя функции>`).

Варианты заданий

Вариант 1

1. Напишите функцию трех аргументов (`list3 x y z`) такую, что $(list3\ x\ y\ z) = (x\ y\ z)$ для любых символьных выражений; не используйте функцию `list`.
2. Последовательность чисел Фибоначчи 1, 1, 2, 3, 5, 8, 13... строится по следующему закону: первые два числа - единицы; любое следующее число есть сумма двух предыдущих $f(n)=f(n-1)+f(n-2)$. Напишите функцию (`f n f1 f2`) с накапливающимися параметрами `f1` и `f2`, которая вычисляет n-ое число Фибоначчи.
3. Определите умножение целых чисел (`*2 x y`) через сложение и вычитание.

Вариант 2

1. Напишите функцию, которая выдает истину, если ее аргумент удовлетворяет хотя бы одному из следующих условий:
 - а) является списком из двух элементов;
 - б) является списком из двух атомов;
 - в) является списком из трех элементов.
2. Определите возведение в целую степень (x^n) через умножение и деление.
3. Напишите функцию (`fullength x`), считающую полное количество атомов (не равных `nil`) в списке `x`.

Вариант 3

1. Напишите с помощью композиции условных выражений функции от четырех аргументов (`and4 x1 x2 x3 x4`) и (`or4 x1 x2 x3 x4`), совпадающие с встроенными функциями `and` и `or` от четырех аргументов.
2. Напишите функцию, вычисляющую последний элемент списка.
3. Напишите функцию от двух аргументов `x` и `n`, которая создает список из `n` раз повторенных элементов `x`.

Вариант 4

1. Напишите функцию, осуществляющую циклическую перестановку элементов в списке, т.е. $(f\ g\ h\ j) \rightarrow (g\ h\ j\ f)$.
2. Напишите функцию, которая из данного одноуровневого списка строит список списков его элементов, например, $(a\ b) \rightarrow ((a)\ (b))$.
3. Определите функцию, зависящую от двух аргументов `u` и `v`, являющихся списками, которая вычисляет список всех элементов `u`, не содержащихся в `v`.

Вариант 5

1. Определите функцию (`f a b c`), которая равна истине тогда и только тогда, когда из отрезков с длинами `a`, `b` и `c` можно построить треугольник.
2. Определите функцию, зависящую от двух аргументов `u` и `v`, являющихся списками, которая вычисляет список всех элементов, содержащихся либо в `u`, либо в `v`, но не одновременно в `u` и `v`.
3. Напишите функцию, осуществляющую замену элементов списка `u` на соответствующие элементы списка `x` в списке `w`, например,

$y=(a\ b), x=(1\ 2), w=((a\ b)\ a\ (c\ (a\ (a\ d)))) \rightarrow ((1\ 2)\ 1\ (c\ (1\ (1\ d))))$.

Вариант 6

1. Определите функцию $(f\ a\ b\ c)$, которая вычисляет список корней квадратного уравнения $a*x^2+b*x+c=0$ (если корней нет, то список пустой).
2. Напишите функцию, аналогичную встроенной функции замены `subst` в списке s выражения x на y , но производящую взаимную замену x на y , т.е. $x \rightarrow y, y \rightarrow x$.
3. Определите функцию $(f\ s)$, результатом которой является список, получающийся после удаления на всех уровнях всех положительных элементов списка чисел s .

Вариант 7

1. Определите функцию, которая меняет местами первый и последний элементы списка, оставляя остальные на своих местах.
2. Определите функцию $(\text{summa_digits}\ n)$, результатом которой является сумма цифр натурального числа n .
3. Определите функцию $(f\ s)$, которая из данного списка s удаляет последний элемент.

Задание №2

Задание состоит из трех задач, в которых требуется составить программы на Лиспе. В первых двух задачах требуется для программирования использовать локальные или вспомогательные функции. В третьей задаче требуется использовать функционалы. При составлении программ (если не оговорено противное) можно использовать все встроенные функции Лиспа.

Варианты заданий

Вариант 1

1. Определите функцию, зависящую от одного аргумента, которая по данному списку вычисляет список его элементов, встречающихся в нем более одного раза. Проверьте, как она будет работать на примере $(a\ a\ a\ b\ a)$.
2. Определите функцию, зависящую от двух аргументов u и n , которая по данному списку строит список его элементов, встречающихся в нем не менее n раз. Проверьте работу этой функции на примере $(a\ a\ b\ a\ c\ b\ c\ a\ b\ b\ d\ a\ b)$ для $n=1,2,5,0$.
3. Используя функционалы, напишите функцию, которая из данного списка строит список списков его элементов, например, $(a\ b) \rightarrow ((a)\ (b))$.

Вариант 2

1. Определите функцию, обращающую список и все его подсписки на любом уровне, например, $(a\ b\ (c\ d)\ e) \rightarrow (e\ (d\ c)\ b\ a)$.
2. Напишите функцию, заменяющую Y на число, равное глубине вложения Y в W , например, $Y=a, W=((a\ b)\ a\ (c\ (a\ (a\ d)))) \rightarrow ((2\ b)\ 1\ (c\ (3\ (4\ d))))$.
3. Напишите функцию, единственным аргументом которой являлся бы список списков, объединяющую все эти списки в один.

Вариант 3

1. Напишите функцию, определяющую глубину первого вхождения элемента u в список w .
2. Напишите функцию, которая делает из списка множество, т.е. удаляет все повторяющиеся элементы.
3. Напишите функцию $(\text{exists}\ p\ x)$, которая проверяет 'Существует ли элемент списка x , удовлетворяющий предикату p ?' (p - функция или функциональное имя).

Вариант 4

1. Напишите функцию, которая определяет является ли данное натуральное число простым. Воспользуйтесь более общей задачей: $(\text{ispr}\ n\ m)$ - 'Число n не делится ни на одно число большее или равное m и меньшее n '. Имеем $(\text{ispr}\ n\ n)$ - истинно, во-первых, если $n = m$, и, во-вторых, если истинно $(\text{ispr}\ n\ m+1)$ и n не делится на m .
2. Напишите функцию, которая сортирует список чисел, используя алгоритм простой вставки.
3. Напишите функцию $(\text{all}\ p\ x)$, которая проверяет 'Для всех ли элементов списка x выполняется предикат p ?' (p - функция или функциональное имя).

Вариант 5

1. Напишите функцию, которая сортирует список чисел, используя алгоритм простого выбора.
2. Определите функцию $(f\ s)$, результатом которой является список, получающийся из списка списков s после удаления всех подсписков, содержащих числа.
3. Напишите функцию $(\text{filter}\ p\ x)$, которая 'фильтрует' (создает список) элементы списка x , удовлетворяющие предикату p (p - функция или функциональное имя).

Вариант 6

1. Определите функцию $(f\ a\ n)$, которая от двух числовых аргументов вычисляет величину $a+a*(a+1)+a*(a+1)*(a+2)+\dots+a*(a+1)*\dots*(a+n)$.
2. Определите функцию $(f\ s)$, которая вычисляет список $(m_1\ m_2\ m_3)$, состоящий из трех наибольших элементов списка $s: m_1 \geq m_2 \geq m_3$. Исходный список содержит не менее трех элементов.
3. Определите функцию $(f\ s\ n)$, которая из списка чисел s создает новый список, прибавляя к каждому атому число n . Исходный список не предполагается одноуровневым.

Вариант 7

1. Определите функцию $(f\ n)$, n кратное 3, вычисляющую сумму: $1*2*3+4*5*6+\dots+(n-2)*(n-1)*n$.
2. Определите функцию $(f\ s)$, которая в одноуровневом списке чисел s переставляет все отрицательные элементы в начало списка, например, $(f\ (4\ -8\ 6\ -9\ -7)) \rightarrow (-8\ -9\ -7\ 4\ 6)$.
3. Определите функцию $(f\ s)$, которая из списка чисел s создает новый список, меняя знак у каждого атома. Исходный список не предполагается одноуровневым.

- Вариант 8 1. Определите функцию $(f\ s)$, вычисляющую знакопередающую сумму $a_1 - a_2 + a_3 - a_4 + \dots + a_k * (-1)^{(k+1)}$ для списка s , имеющего вид $(a_1\ a_2\ a_3\ \dots\ a_k)$.
2. Определите функцию $(f\ n)$, которая для натурального числа n вычисляет $1!+2!+3!+\dots+n!$.

3. Напишите функцию (count p x), которая подсчитывает, сколько атомов в списке x удовлетворяет предикату p (p - функция или функциональное имя). Список x не предполагается одноуровневым.

Расчетно-графическая работа №2 ЛОГИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

Задание №1

Задание состоит из двух задач, в которых требуется составить программы на Прологе для написания простых предикатов. При составлении про-грамм (если не оговорено противное) можно использовать все встроенные предикаты Пролога. SWI-Prolog не имеет стандартного help'a для Windows, для этого используется предикат help. Вызов help(<имя предиката>) выдает на экран информацию об этом предикате. Вызов help(7) выдает на экран список всех встроенных предикатов с комментариями. Текстовый файл руководства по SWI-Prolog - pl\library\manual. Отладку предикатов можно осуществлять с помощью предиката трассировки trace(<имя предиката>), трассировка предиката отключается - trace(<имя предиката>, -all).

Варианты заданий

Вариант 1

1. Определите возведение в целую степень через умножение и деление.
2. Напишите предикат $p(+L, -N)$ - истинный тогда и только тогда, когда N - предпоследний элемент списка L, имеющего не менее двух элементов.

Вариант 2

1. Напишите предикат $p(+X, +N, -L)$ - истинный тогда и только тогда, когда L - список из N раз повторенных элементов X.
2. Напишите предикат $p(+L, -S)$ - истинный тогда и только тогда, когда S - список списков элементов списка L, например, $p([a, b, c], [[a], [b], [c]])$ - истина.

Вариант 3

1. Напишите предикат $p(+L, -S)$ - истинный тогда и только тогда, когда L - список списков, а S - список, объединяющий все эти списки в один.
2. Напишите предикат $p(+L, -S)$ - истинный тогда и только тогда, когда спи-сок S есть циклическая перестановка элементов списка L, например, $p([f, g, h, j], [g, h, j, f])$ - истина.

Вариант 4

1. Напишите предикат $p(+X, +N, +V, -L)$ - истинный тогда и только тогда, ко-гда список L получается после добавления X на N-е место в список V.
2. Напишите предикат $p(+N, +V, -L)$ - истинный тогда и только тогда, когда список L получается после удаления N-го элемента из списка V.

Вариант 5

1. Напишите предикат $p(+V, -L)$ - истинный тогда и только тогда, когда спи-сок L получается после удаления всех повторных вхождений элементов в список V, например, $p([a, b, c, d, d, a], [a, b, c, d])$ - истина.
2. Напишите предикат $p(+V, -L)$ - истинный тогда и только тогда, когда спи-сок L получается после удаления из списка V всех элементов, стоящих на четных местах, например, $p([1, 2, 3, 4, 5, 6], [1, 3, 5])$ - истина.

Вариант 6

1. Напишите предикат $p(+V, +X, -L)$ - истинный тогда и только тогда, когда список L получается из списка V после удаления всех вхождений X на всех уровнях, например, $p([1, [2, 3, [1]], [3, 1]], 1, [[2, 3, []], [3]])$ - истина.
2. Напишите обобщение предиката member, когда ищется элемент на всех уровнях в списке.

Вариант 7

1. Определите предикат $p(+U, +V, -L)$ - истинный тогда и только тогда, когда список L есть список всех элементов списка U, не содержащихся в списке V.
2. Определите предикат $p(+U, +V, -L)$ - истинный тогда и только тогда, когда L - список всех элементов, содержащихся либо в списке U, либо в списке V, но не одновременно в U и V.

Вариант 8

1. Определите предикат $p(+V, -L)$ - истинный тогда и только тогда, когда L - список всех элементов списка V, встречающихся в нем более одного раза.
2. Напишите предикат $subst(+V, +X, +Y, -L)$ - истинный тогда и только тогда, когда список L получается после замены всех вхождений элемента X в списке V на элемент Y.

Вариант 9

1. Напишите предикат $p(+V, -L)$ - истинный тогда и только тогда, когда спи-сок L получается из списка V после удаления всех повторяющихся элементов, т. е. из списка получается множество.
2. Напишите предикат $exists(+P, +L)$, который проверяет 'Существует ли эле-мент списка L, удовлетворяющий предикату P?'

Вариант 10

1. Напишите предикат $\text{all}(+P, +L)$, который проверяет 'Для всех ли элементов списка L выполняется предикат P?'
2. Напишите предикат $\text{filter}(+V, +P, -L)$ - истинный тогда и только тогда, ко-гда список L есть список всех элементов из списка V, удовлетворяющих пре-дикату P ('фильтрация' списка).

Вариант 11

1. Используя предикаты 'родитель'(Родитель, Отпрыск), 'женщи-на'(Человек), 'мужчина'(Человек) и 'супруги'(Жена, Муж), определите от-ношения теща, шурин и зять.
2. Башня из кубиков может быть описана совокупностью фактов вида 'на'(Кубик1, Кубик2), которые истинны, если Кубик1 поставлен на Кубик2. Определите предикат 'выше'(Кубик1, Кубик2), который истинен, если Ку-бик1 расположен на башне выше, чем Кубик2. (Указание: 'выше' является транзитивным замыканием отношения 'на'.)

Вариант 12

1. Напишите предикат $\text{gcd}(+A, +B, -D)$ - истинный тогда и только тогда, когда D -наибольший общий делитель двух целых положительных чисел A и B.
2. Напишите программу для отношения $\text{double}(+List, -ListList)$, в котором ка-ждый элемент списка List удваивается в списке ListList, например, $\text{double}([1,2,3],[1,1,2,2,3,2])$ выполнено.

Вариант 13

1. Напишите новую версию предиката $\text{length}(+L, -N)$, в котором при подсчете количества элементов списка не учитывается пустой список. К при-меру, для списка [a,b,c,d,e] новая версия процедуры должна сообщить, что длина списка равна пяти, а для списка [a,[],c,d,[]] эта процедура должна да-вать длину, равную трем.
2. Пусть имеется список структур 'client': [client(a,29,3),client(b,29,6),client(c,40,2)]. Первым аргументом каждой структуры служит имя клиента, вторым - суточ-ный тариф, а третьим - количество дней, на которое взята автомашина. На-пишите правило, позволяющее вычислить итоговую сумму оплаты, объеди-няющую выплаты всех клиентов, данные о которых содержатся в списке.

Вариант 14

1. Опишите процедуру для предиката $\text{расщепить}/4$, которая берет список це-лых чисел L1 и целое число N и выдает списки L2 и L3 такие, что числа из исходного списка, меньшие, чем N, помещаются в список L2, а остальные - в список L3.
2. Напишите предикат для вычисления чисел Фибоначчи, используя метод накапливающего параметра.

Вариант 15

1. Напишите предикат $\text{digits}(+N, -L)$ - истинный тогда и только тогда, когда L - список цифр натурального числа N.
2. Напишите предикат $\text{summa_digits}(+N, -S)$ - истинный тогда и только тогда, когда S - сумма цифр натурального числа N.

Задание №2

Задание состоит из двух задач, в которых требуется составить програм-мы на Прологе для написания простых программ. При составлении программ (если не оговорено противное) можно использовать все встроенные предикаты Пролога.

Варианты заданий

Вариант 1

1. Напишите предикат, аналогичный предикату subst (см. первое контрольное задание, вариант 8, задача 2), но производящий взаимную замену X на Y, т.е. $X \rightarrow Y, Y \rightarrow X$.
2. Напишите предикат, который определяет, является ли данное натуральное число простым. Воспользуйтесь более общей задачей: $\text{ispr}(N, M)$ - 'Число N не делится ни на одно число большее или равное M и меньшее N'.

Имеем $\text{ispr}(N, M)$ -истинно, во-первых, если $N = M$, и, во-вторых, если ис-тинно $\text{ispr}(N, M+1)$ и N не делится на M.

Вариант 2

1. Сортировка списка простой вставкой (по возрастанию).
2. Сортировка списка простым выбором (по возрастанию).

Вариант 3

1. Напишите предикат $\text{p}(+L, -N)$ - истинный тогда и только тогда, когда N - количество различных элементов списка L.
2. Напишите предикат $\text{p}(+X, +Y, -Z)$ - истинный тогда и только тогда, когда Z есть 'пересечение' списков X и Y, т.е. список, содержащий их общие элемен-ты, причем кратность каждого элемента в списке Z равняется минимуму из его кратностей в списках X и Y.
- Вариант 4 1. Запрограммируйте предикат $\text{p}(+A, +B)$, распознающий, можно ли получить список элементов A из списка элементов B посредством вычеркивания неко-торых элементов. Алгоритм: Если A - пустой список, то ответом будет 'да'. В противном слу-чае нужно посмотреть, не пуст ли список B. Если это так, то ответом будет 'нет'. Иначе нужно сравнить первый элемент списка A с первым элементом списка B. Если они совпадают, то надо снова применить тот же алгоритм к остатку списка A и остатку списка B. В противном случае нужно снова при-менить тот же алгоритм к исходному списку A и остатку списка B.
2. Напишите предикат $\text{p}(+X, +Y, +L)$ - истинный тогда и только тогда, когда X и Y являются соседними элементами списка L.
- Вариант 5 1. Определите отношение $\text{sum_tree}(+TreeOfInteger, -Sum)$, выполненное, если число Sum равно сумме целых чисел являющихся вершинами дерева TreeOfInteger.
2. Определим

операторы: :- op(100, fy, ~). :- op(110, xfy, &).
:- op(120, xfy, v).

Булева формула есть терм, определяемый следующим образом: константы true и false - булевы формулы; если X и Y - булевы формулы, то и $X \vee Y$, $X \& Y$, $\sim X$ - булевы формулы, здесь \vee и $\&$ - бинарные инфиксные операторы дизъюнкции и конъюнкции, а \sim - унарный оператор отрицания. Напишите предикат p(+T), определяющий, является ли данный терм T булевой формулой.

Вариант 6 1. Встроенный предикат functor(+Term, ?Functor, ?Arity) определяет для заданного составного термина Term его функтор Functor и местность Arity. Встроенный предикат arg(+N, +Term, ?Value) определяет для целого числа N и заданного составного термина Term его N-ый аргумент Value. Определите предикаты functor1 и arg1 - аналоги предикатов functor и arg через предикат univ (=..). 2. Напишите предикат range(?M, ?N, ?L), истинный тогда и только тогда, когда L - список целых чисел, расположенных между M и N включительно (предикат должен допускать различное использование, когда не менее двух из трех аргументов конкретизованы). (Указание. Используйте предикаты var(+X) и nonvar(+X)).

Вариант 7 1. Напишите вариант программы plus(?X, ?Y, ?Z), пригодный для сложения, вычитания и разбиения чисел на слагаемые. (Указание. Используйте для порождения чисел встроенный предикат between(+Low, +High, ?Value), который порождает все целые числа от нижней границы Low до верхней границы High.)

2. Напишите программу вычисления целочисленного квадратного корня из натурального числа N, определяемого как число I, такое, что $I * I \leq N$, но $(I+1) * (I+1) > N$. Используйте определение предиката between/3 для генерирования последовательности натуральных чисел с помощью механизма возвратов.

Вариант 8

1. Напишите новую версию процедуры 'предок', которая выработывает список представителей всех промежуточных поколений, располагающихся между предком и потомком. Предположим, например, что Генри является отцом Джека, Джек - отцом Ричарда, Ричард - отцом Чарльза, а Чарльз - отцом Джейна. При запросе о том, является ли Генри предком Джейна, должен

выдаваться список, характеризующий родственную связь этих людей, конкретно: [джек, ричард, чарльз].

2. Определите предикат p(+V, +N, -L) - истинный тогда и только тогда, когда L - список элементов списка V, встречающихся в нем не менее N раз. Проверьте работу этого предиката на примере [a, a, b, a, c, b, c, a, b, b, d, a, b] для N=1,2,5,0.

Задание №3

Задание состоит из двух задач, в которых требуется составить более сложные программы на Прологе (как правило, требуется определить несколько предикатов). При составлении программы (если не оговорено противное) можно использовать все встроенные предикаты Пролога.

Варианты заданий

Вариант 1

1. Напишите предикат p(+N, +K, -L) - истинный тогда и только тогда, когда L - список всех последовательностей (списков) длины K из чисел 1,2,...,N.

2. Напишите предикат p(+N, -L) - истинный тогда и только тогда, когда список L содержит все последовательности (списки) из N нулей и единиц, в которых никакая цифра не повторяется три раза подряд (нет куска вида XXX).

Вариант 2

1. Определите отношение ordered(+Tree), выполненное, если дерево Tree является упорядоченным деревом целых чисел, т.е. число, стоящее в любой вершине дерева, больше любого элемента в левом поддереве и меньше любого элемента в правом поддереве. Определение структуры 'дерево' см. раздел 'Второе контрольное задание'. Указание. Можно использовать вспомогательные предикаты ordered_left(+X, +Tree) и ordered_right(+X, +Tree), которые проверяют, что X меньше (больше) всех чисел в вершинах левого (правого) поддерева дерева Tree и дерево Tree - упорядочено.

2. Определим операторы:

:- op(100, fy, ~).

:- op(110, xfy, &).

:- op(120, xfy, v).

Булева формула есть терм, определяемый следующим образом: константы true и false - булевы формулы; если X и Y - булевы формулы, то и $X \vee Y$, $X \& Y$, $\sim X$ - булевы формулы, здесь \vee и $\&$ - бинарные инфиксные операторы дизъюнкции и конъюнкции, а \sim - унарный оператор отрицания.

Напишите программу, распознающую логические формулы в дизъюнктивной нормальной форме, т.е. формулы, являющиеся дизъюнкцией конъюнкций литералов, где литерал - атомарная формула или ее отрицание. Вариант 3 1. Определим операторы:

:- op(100, fy, ~).

:- op(110, xfy, &).

:- op(120, xfy, v).

Булева формула есть терм, определяемый следующим образом: константы true и false - булевы формулы; если X и Y - булевы формулы, то и $X \vee Y$, $X \& Y$, $\sim X$ - булевы формулы, здесь \vee и $\&$ - бинарные инфиксные операторы дизъюнкции и

конъюнкции, а \sim - унарный оператор отрицания. Напишите программу, задающую отношение `negation_inward(+F1,-F2)`, кото-рое выполнено, если логическая формула F2 получается из логической фор-мулы F1 внесением всех операторов отрицания внутрь конъюнкций и дизъ-юнкции.

Вариант 4

1. Определите предикат `occurrences(+Sub,+Term,-N)`, истинный, если число N равно числу вхождений подтерма Sub в терм Term. Предполагается, что терм Term не содержит переменных. 2. Разработайте программу 'Советник по транспорту'. Выберите либо сеть, состоящую из городов, либо транспортную сеть маршрутов поездов или авто-бусов в пределах одного города. Вы должны информировать систему о том, откуда и куда Вы собираетесь добраться, а система должна выдавать реко-мендации о том, какими поездами, автобусами, самолетами и т. д. Вам следу-ет воспользоваться, чтобы добраться до пункта назначения.

Вариант 5

1. Напишите предикат `p(+S, -L)`, который переводит предложение S, пред-ставленное строкой, в список атомов L. Например, `p('gfrtyre hjnki <> pi 876 h', [gfrtyre, hjnki, '<>', pi, 876,h])` выполнено. Указание. Воспользуйтесь предика-том `name/2`. 2. Множественное число большинства английских существительных получа-ется путем добавления буквы 's' к форме единственного числа. Но если су-ществительное заканчивается буквой 'y', следующей за согласной, множе-ственное число образуется путем замены буквы 'y' на сочетание 'ies'; если же существительное заканчивается буквой 'o', следующей за согласной, множе-ственное число образуется путем добавления сочетания 'es'. Напишите ут-верждения для предиката `множественное_число/2`, которые задают все эти правила. Указание. Воспользуйтесь предикатом `name/2`.

Вариант 6

1. Простейшая система кодирования сообщений заключается в замене каждой буквы сообщения на букву, находящуюся на N-й по отношению к ней пози-ции в алфавите. Например, для N=2 буква 'a' заменяется на 'c', буква 'y' на 'a' и т.д. Зная, что коды ASCII букв от 'a' до 'z' изменяются от 97 до 122, напишите процедуру для предиката `шифратор/3`, который берет шифруемое слово и целое число и выдает слово, представляющее шифр данного слова, полученный с помощью указанного метода.

Указание. Воспользуйтесь предикатом `name/2`.

2. Напишите предикат `предшествует/2`, который берет два атома в качестве своих аргументов и успешно согласуется, если первый из них в лексико-графическом порядке предшествует второму.

Указание. Воспользуйтесь предикатом `name/2`.

Вариант 7

1. Одним из примеров использования предиката `name/2` может служить гене-рация новых атомов для представления вновь вводимых объектов, например, `abc1`, `abc2`, `abc3` и т.д. Эти имена характеризуются тем, что все они состоят из корня, определяющего тип именуемого объекта, и целочисленного суффикса для различения объектов одного типа. Напишите программу

`новое_имя(+X, -Y)`. Последовательность имен создается с помощью возвра-тов. Указание. Воспользуйтесь предикатом `int_to_atom(+N,-X)`, который кон-вертирует натуральное число N в атом X.

2. Построить программу 'сжать', назначение которой - преобразование анг-лийских слов в их 'звуковой' код. Этот процесс предусматривает 'сжатие' примерно одинаково звучащих слов в одинаковый их код - своего рода, аб-бревиатуру этих слов. Слова 'сжимаются' в соответствии со следующими правилами:

- первая буква слова сохраняется;
- все последующие за ней гласные, а также буквы 'h', 'w' и 'y' удаляются;
- сдвоенные буквы заменяются одиночными;
- закодированное слово состоит не более чем из четырех букв, остальные бук-вы удаляются.

Примеры: `сжать(barrington, bmg)` и `сжать(llewellyn, ln)` - выполнено.

Указание. Воспользуйтесь предикатом `name/2`.

Отчет должен соответствовать следующим требованиям:

1. Отчет результатов РГР оформляется в текстовом редакторе MS Word на листах формата A4 (297x210).
2. Изложение материала в отчете должно быть последовательным и логичным. Отчет состоит из задания на РГР, содержания, разделов, выводов и списка литературных источников. В структуру отчета может входить Приложение.
3. Объем РГР работы должен быть – 10-15 страниц.
4. Отчет должен быть отпечатан на компьютере через 1-1,5 интервала, номер шрифта – 12-14 пт Times New Roman. Расположение текста должно обеспечивать соблюдение следующих полей:
 - левое 20 мм.
 - правое 15 мм.
 - верхнее 20 мм.
 - нижнее 25 мм.
5. Все страницы отчета, включая иллюстрации и приложения, имеют сквозную нумерацию без пропусков, повторений, литературных добавлений. Первой страницей считается титульный лист, на которой номер страницы не ставится.
6. Таблицы и диаграммы, созданные в MS Excel, вставляются в текст в виде динамической ссылки на источник через специальную вставку.
7. Основной текст делится на главы и параграфы. Главы нумеруются арабскими цифрами в пределах всей работы и начинаются с новой страницы.
8. Подчеркивать, переносить слова в заголовках и тексте нельзя. Если заголовок состоит из двух предложений, их

разделяют точкой. В конце заголовка точку не ставят.

9. Ссылки на литературный источник в тексте сопровождаются порядковым номером, под которым этот источник включен в список используемой литературы. Перекрестная ссылка заключается в квадратные скобки. Допускаются постраничные сноски с фиксированием источника в нижнем поле листа.

10. Составление библиографического списка используемой литературы осуществляется в соответствии с ГОСТ.

Оформление и защита производится в соответствии со стандартом ДВГУПС СТ 02-11-17 «Учебные студенческие работы. Общие положения»

Оценка знаний по дисциплине производится в соответствии со стандартом ДВГУПС СТ 02-28-14 «Формы, периодичность и порядок текущего контроля успеваемости и промежуточной аттестации».