ФЕДЕРАЛЬНОЕ АГЕНСТВО ПО ОБРАЗОВАНИЮ
РОССИИ

УДМУРТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет информационных технологий и вычислительной техники

Кафедра Теории и методологии информатики

РАБОЧАЯ ПРОГРАММА

по дисциплине
«Языки программирования»

для направаления 511900 — Информационные технологии

Курс 1
Семестр 1–2
Всего аудиторных часов 102
Лекции час. 68
Лабораторные работы, час. 0
Практические (семинарские) занятия, час. 34
Самостоятельная работа, час. 51
Зачет (семестр) 1
Экзамен (семестр) 2

г. Ижевск
2007

Рабочая программа составлена на опыте 20-летнего опыта работы и на базе научной квалификации лектора, подкрепленной существующим учебником; никакие стандарты во внимание не принимались, поскольку по закону они являются лишь рекомендацией, а в реальности они не подкреплены системным обоснованием. Ставится цель максимально способствовать развитию фундаментальных навыков, необходимых для практической работы на должностях выше рядовых программистов в современной информатике.

Автор рабочей программы
д.ф.-м.н. профессор, зав. каф. ТиМИ Непейвода Н. Н.

Рабочая программа утверждена на заседании кафедры теории и методологии информатики
«9 сентября» 2007 г.

Решение методической комиссии ФИТВТ
« » 2007 г.
Председатель методической комиссии

Согласовано с библиотекой УдГУ « » 2007 г.
Директор библиотеки УдГУ

ОПД.Ф.05	Языки программирования и методы трансляции Основные понятия языков программирования; синтаксис, семантика, формальные способы описания языков программирования; типы данных, способы и механизмы управления данными; методы и основные этапы трансляции; конструкции распределенного и параллельного программирования.	153
	Стандарт в данном случае отражает лучшую практику работы конкретной кафедры конкретного университета, имеющего конкретных специалистов. Курса нетрадиционного программирования и стилей программирования в нем вообще нет. А это гораздо более фундаментальный курс, тем более для будущих программистов в индустриальном программировании. Стандарт не является законом, и поэтому не принимается во внимание в данной программе, основанной на критически осмысленном и многократно защищенном 20-летнем опыте успешной работы. **АВТОМАТИЗИРОВАННАЯ ПРОВЕРКА РАБОЧИХ ПРОГРАММ НЕЗАКОННА, А В ДАННОМ СЛУЧАЕ И НЕВОЗМОЖНА**	

Специалист по информационным технологиям должен знать теоретические основы и теоретические ограничения различных стилей программирования, владеть основами всех пяти основных стилей (структурный, сентенциальный, автоматный, функциональный, событийный), понимать, в каких областях каждый из этих стилей имеет преимущество. В курсе программирования закладываются основы структурного и автоматного стилей на базе традиционного языка. В этом курсе закладывается вся система взаимосвязей стилей программирования и даются основы сентенциального, функционального, событийного программирования. В конкретной реализации данного курса эти стили осваиваются на примере языков Рефал, Пролог, Лисп, языка скриптов событийного программирования Парадокс.

2. Принципы построения курса «Языки программирования»

Курс является фундаментальным прикладным курсом, и должен ввести студентов в круг важнейших понятий программирования и показать их взаимосвязь с практикой и теорией.

Основное внимание уделяется выработке системного понимания соотношений между различными стилями. Поэтому в качестве конкретных систем, на которых ведется обучение стилям, взяты реализации, наиболее ярко подчеркивающие особенности каждого из стилей и обладающие наивысшей сте-

пенью концептуального единства. В неразрывной связи с практикой показывается, насколько необходимо хорошее знание теории.

Как показывает опыт, сложная практика, тем более демонстративно дистанционирующаяся от конкретных инструментов воспринимается большинством студентов лишь в том случае, когда они видят, насколько принципиальный выигрыш дает применение подходящего орудия в подходящем месте, даже несмотря на то, что данное орудие не является общепринятым. Поэтому пришлось разрабатывать принципиально новый набор задач для данного курса.

Ввиду того, что необходимо достичь баланса между концептуальной важностью и трудностью курса, с одной стороны, и возможностью его понимания для большинства студентов, с другой стороны, а также ввиду того, что год от года уровень студентов колеблется, часть часов оставлена как резервные, а объем проходимого материала может также различаться. Материал, который является наиболее нетрадиционным и трудным, помечен красным цветом (он, как правило, не опускается; опускаются при слабом потоке или при возникновении неожиданных технических трудностей более технические части, помеченные зеленым цветом).

Основным учебником по курсу является книга [1], а основным задачником книга [2].

Практические занятия курса «Языки программирования» направлены на отработку работы в различных стилях программирования и на организацию взаимодействия программ. написапнных в разных стилях.

3. Цели курса и его место

— Дать первоначальные навыки программирования на всех основных стилях

— Показать тесную взаимосвязь теории и практики информатики

— Показать сложность проблемы семантики и разнообразие семантик алгоритмических языков. Познакомить с примерами точных семантик.

— Сформировать систему важнейших фундаментальных практических понятий программирования: стили программирования, семантики, синтаксис, прагматика, призраки, подпорки, концептуальная несовместимость, концептуальное единство.

— Выработать реалистическое отношение к новизне, универсальности (как к безусловно отрицательному качеству, которое иногда приходится терпеть за неимением хорошо реализованных специализированных орудий), показать, как новые возможности портят хорошие системы.

— Теорема Янова о полноте системы преобразований схем Янова как пример полной алгебраической семантики.

— Теорема Карри-Шейнфинкеля как пример полной трансляционной семантики.

— Взаимосвязь конструктивных логик и стилей программирования.

Курс взаимосвязан с курсами дискретной математики и логики, основ алгоритмизации и всем циклом курсов по информатике. Курс преподается параллельно и в системном взаимодействии с курсами Основ алгоритмизации и Логики.

4. Распределение (примерное) часов по видам и темам работ

№	Тема	Часы	Лекции	Практ	Самост
1	О вашей будущей специальности. Цели курса.	2	2	0	0
2	Понятие о синтаксисе, семантике и прагматике	2	2	0	0
3	Модель вычислительной машины Цузе и традиционные языки	2	2	0	0
4	Сентенциальное программирование на примере языка Рефал	22	6	8	8
5	Теоретическая основа функционального программирования: комбинаторная логика и λ-конверсии.	8	6	0	5
6	Вторая теоретическая основа: рекурсивные схемы. Трансформационная и алгебраическая семантика на их примере.	6	4	0	2
7	Функциональное программирование на примере языка Лисп	20	4	8	8
8	Сентенциальное программирование: вторая ипостась (Пролог)	26	6	12	8
9	Схемы программ. Их семантика как пример интерпретационной семантики. Структурные схемы и теорема Бема—Джакопини. Схемы Янова и теорема Янова.	12	8	0	4

10	Автоматное программирование как часть общей системы стилей. Его взаимосвязь со схемами Янова.	2	2	0	0
11	Структурное программирование как часть общей системы стилей. Рекурсивный и циклический варианты. Призраки и подпорки. Инварианты. Сеть данных и сеть управления. Невидимые подпорки и совместное исполнение.	8	6	0	2
12	Логическая семантика на примере структурного программирования, пред- и постусловий, инвариантов	10	8	0	2
13	Событийное программирование	12	2	2	8
14	Параллелизм. Различные его виды и их совместимость с различными стилями.	4	4	0	0
15	Резерв	12	6	4	4
16	Итого	153	68	34	51

3. Формы контроля.

Текущий контроль осуществляется в виде сдачи программ по прохождении каждой программистской темы и мини-контрольных на 10–15 мин. по каждому разделу теоретических тем.

Итоговый контроль состоит из зачета в конце 1 семестра и экзамена в конце 2 семестра.

4 Содержание курса.

4.1. Темы и их краткое содержание

4.1.1. О вашей будущей специальности. Цели курса. Что необходимо от информатика, не собирающегося оставаться чернорабочим? Что умалчивается в нынешней литературе? Вы думаете, что вы уже много знаете, но вы еще ничего не знаете и не умеете как следует. В голове нужно иметь систему и интеллект (естественный).

4.1.2. Понятие о синтаксисе, семантике и прагматике. Синтаксис (контекстно-свободный и полный; конкретный и абстрактный). Виды семантик: интерпретационная, трансляционная, трансформационная, алгебраическая, логическая, естественно-языковая. Что такое прагматика?

4.1.3. Модель вычислительной машины Цузе и традиционные языки. Машина типа машины Цузе (память, процессор, команды, данные). Интер-

претация присваиваний как действий с памятью. Понятие о языке традиционного типа.

4.1.4. Сентенциальное программирование на примере языка Рефал. Отождествление и подстановка. Поле зрения Рефал-программы. Структура Рефал-программы. некоторые приемы программирования. Дополнительные возможности для ввода-вывода и организации глобальной памяти.

4.1.5. Теоретическая основа функционального программирования: комбинаторная логика и λ-конверсии. Функция как фундаментальное понятие. Три комбинатора Шейнфинкеля. Сведение многоместных функций к одноместным функционалам. Задачи на построение комбинаторов. Зацикливающийся комбинатор. λ-язык. λ-конверсии. Теорема Карри-Шейнфинкеля и преимущество человеческого ума перед автоматом на ее примере. Теорема о неподвижной точке. Теорема Черча-Россера (формулировка и значение).

4.1.6. Вторая теоретическая основа: рекурсивные схемы. Трансформационная и алгебраическая семантика. Рекурсия как один из главных инструментов информатика. Рекурсивные схемы по Маккарти. Задачи на установление соотношений между рекурсивно определенными программами. Задачи на построение простейших функций обработки списков.

4.1.7. Функциональное программирование на примере языка Лисп. Структура языка Лисп. Поле зрение Лисп-программы. Структура Лисп-программы. Базовые функции и функционалы. Возможность динамического определения функций. Моделирование элементов последовательного императивного языка внутри Лисп.

4.1.8. Сентенциальное программирование: вторая ипостась (Пролог). Унификация и подстановка. Структура поля зрения Пролог-программы. Структура Пролог-программы. Неудачи и возвраты. Отсечение и отрицание. Некоторые дополнительные возможности. Возможность динамического определения базы знаний и самой программы. Теорема Косовского о несовместимости унификации и конкретизации (формулировка). Характеризация сентенциального программирования в терминах «Глобальные действия, глобальные условия.»

4.1.9. Схемы программ. Их семантика как пример интерпретационной семантики. Структурные схемы и теорема Бема—Джакопини. Схемы Янова и теорема Янова. Обращается внимание на теорему Янова как на пример полной алгебраической семантики. Формулируются основные неразрешимые проблемы схем программ и на этой базе показывается, что такое «благоглупость» в практических документах и книгах. Формируется четкое понимание того, что переход к структурной схеме не всегда делает программу лучше.

4.1.10. Автоматное программирование как часть общей системы стилей. Его взаимосвязь со схемами Янова. Понятие автомата как таблицы,

графа, функции и схемы Янова. Автоматно разрешимые задачи и зависимость их от конкретной кодировки. Характеризация автоматного программирования в терминах «Глобальные действия, локальные условия.»

4.1.11. Структурное программирование как часть общей системы стилей. Рекурсивный и циклический варианты. Призраки и подпорки. Инварианты. Сеть данных и сеть управления. Невидимые подпорки и совместное исполнение. Характеризация структурного программирования в терминах «Локальные действия, локальные условия.» Неизбежность подпорок и призраков. Инварианты как основа циклов и рекурсий. Прагматическая несовместимость циклов и рекурсий. Наиболее частые призраки и наиболее редко замечаемые подпорки в структурной программе.

4.1.12. Логическая семантика на примере структурного программирования, пред- и постусловий, инвариантов. Описание оператора в виде предусловия, действия и постусловия. Слабейшие предусловия. Что можно и что трудно сделать в логической семантике?

4.1.13. Событийное программирование. События, их обработчики, приоритеты. Отличие демона от обычного события. Области. где наиболее широко применяется событийное программирование. Пример смобытийного программирования на языке скриптов. Характеризация событийного программирования в терминах «Глобальные условия, локальные действия.»

4.1.14. Параллелизм. Различные его виды и их совместимость с различными стилями. $\&$ и $\vee$ параллелизм. Параллелизм и квазипараллелизм. Совместное исполнение и распараллеливание как подпорка. Естественный и искусственный параллелизм. Распределенные системы. Виды палаллелизма, сочетающиеся с различными стилями.

4.2. Примеры задач (каждый студент получает индивидуальные задачи на каждый из четырех языков).

Болталка-0. Ответы В программах-болталках (первым и классическим примером которых была знаменитая Алиса Х. Вейценбаума) ответ генерируется переформулировкой вопросов и утверждений собеседника, чем создается иллюзия доброжелательного и понимающего компьютерного собеседника,который готов обсудить с собой все интересующее тебя. Эти программы используют английский язык.

Создать программу, которая по вопросу на русском языке генерирует ответ. Конечно же, при этом нельзя даже стремиться понять вопрос и дать ответ по существу, но он должен выглядеть как ответ именно на данный вопрос.

Дополнительные условия. Иногда можно дать ответ-джокер, подходящий почти к любому вопросу, например:

Почему все мужчины подонки, а все женщины стервы?
Только Бог и Дьявол знают это.

Подобная генерация, но имеющая несколько вариантов, должна быть оста-

влена в качестве запасного аэродрома на случай, когда Вам не удалось найти что-то более нетривиальное, за что можно зацепиться и дать ответ. Таким образом, здесь Вам предстоит, не раздувая программы. найти несколько остроумных случаев дать ответ на некоторые вопросы.

Выпрямить список Составить список из всех атомов, входящих в данный список и его подсписки. NIL атомом не считается. Одинаковые атомы должны войти в результат по одному разу.

Список литературы

[1] Непейвода Н. Н. Стили и методы программирования, М.: ИНТУИТ, 2005.

[2] А. Е. Анисимов, В. В. Пупышев. Сборник задач по основаниям программирования. М.: ИНТУИТ, 2006.

[3] Непейвода Н. Н., Скопин И. Н. Основания программирования. М.:— Ижевск, РХД, 2004.

[4] Городняя Л. В. Основы функционального программирования. М.: ИНТУИТ, 2004.