

УДК 681.3

Л.Л. Смолякова, А.Г. Юркин

## Информатика и программирование как самостоятельные науки и дифференциация их преподавания

Преподаваемая четырехсеместровая дисциплина многие годы носила название «ЭВМ и программирование» вкупе с такими, как «Практикум на ЭВМ», «Вычислительный практикум» и «Вычислительная практика», однако в последнее время, в связи с развитием новых информационных технологий, стала называться «Информатика». Прежде чем касаться технологических и методологических аспектов, проведем водораздел между понятиями «информатика» и «программирование», выделив их цели, методы, задачи и инструментарий, заметив, что в обоих направлениях выделяются составляющие и науки, и искусства, и ремесла. Для этого сформулируем в сегодняшнем понимании основные этапы решения задач, связанных со сбором, преобразованием информации, решением прикладных задач и внедрением результатов (многие вопросы относятся и к «бумажной» информатике).

1. Дискретизация информации – отображение четырехмерного континуального информационного пространства реального мира на одномерное дискретное конечное информационное пространство компьютерной памяти. Четырехмерный континуум преобразуется в одномерный, а пространство всей мировой компьютерной памяти (включая сетевые объединения) дискретно, конечно и преобразуется в одномерное со сквозной адресацией как конечное множество конечных множеств. Известные методы отображения – сканирование, оконтуривание, распознавание образов, векторизация, квантование. Это есть чистая задача информатики, и программирование может предоставлять лишь некоторые инструментальные средства для ее решения.

2. В структуризации и упаковке дискретной информации с целью ее обозримости и экономии ресурсов применяются методы декомпозиции, агрегирования, сжатия, проектирования баз данных разной структуры, разграничения доступа, поисковые системы, индексация, рубрикация и каталогизация. Заказчиком опять-таки выступает информатика, методы – математические, инструментальные средства – задача программирования.

3. Хранение и передача информации осуществляются организацией сетевого взаимодействия, распределенного хранения, преобразования сигналов,

обеспечения помехоустойчивости, надежности и защиты. Это задача информатики с подключением методов физики и отчасти программирования.

4. Математические модели реального мира применяются для преобразования информации с целью воздействия на объект в интересах индивида. Большинство моделей разработаны в математике, кибернетике и прикладных науках на протяжении десятилетий и уже стали классическими. Здесь для информатики почти нет места, все решается алгоритмизацией и программированием.

5. На внедрение результатов влияет трудно решаемая задача обратного отображения дискретных результатов на континуум мира в виде управляющих воздействий с предсказанием поведения объекта. Это задача кибернетики, математики, других наук о мире, но не информатики и программирования.

С другой стороны, в последнее время наблюдается ярко выраженная дифференциация подготовки абитуриентов в области информационных технологий. Тому есть как объективные (существенное техническое и методическое различие в оснащении средних учебных заведений, разный уровень подготовленности преподавателей), так и субъективные (различные интересы преподавателей, отсутствие единого стандарта по предмету) причины. В результате на одном полюсе – школьники, порой одаренные, но не аттестованные по предмету, на другом – имеющие подготовку на уровне первого курса вуза, а где-то посередине – лишь знакомые с офисными приложениями и любители компьютерных игр. А в вузах – единый стандарт и единая программа подготовки, которая есть развитие стандарта с учетом особенностей вуза, и наблюдается большой отсев способных, но неподготовленных студентов.

Для ликвидации второго противоречия в Алтайском госуниверситете функционирует Математический лицей при АГУ, который посещают около 400 учащихся из 10 школ, в основном из тех, где информатика поставлена слабо; планируется открытие лицеев в районах края. Естественно, программа лицея частично пересекается с вузовской, но в последнее время упор делается на информатику, а не на программирование. Цель подготовки очевидна – выявить способных

школьников и помочь им в начальной адаптации к вузовским требованиям.

Тем не менее дифференциация поступивших остается, и в этом (1999/2000) учебном году удалось коллектив студентов-первокурсников разделить на два потока – «начинающие» и «продвинутые», с дальнейшим разделением на пользователей (преподавателей информатики, начиная от школьников и кончая курсами компьютерной грамотности) и разработчиков с рекомендацией специализироваться по программированию (с правом перехода с одного потока на другой с учетом интересов и успехов). Основой разделения служит тестирование и собеседование при зачислении, с учетом пожеланий обучаемых. Такой дидактический подход известен как метод «внешней» дифференциации, при котором студенты специально организуются в учебные группы по какому-либо признаку.

«Внутренняя» дифференциация подразумевает индивидуальный подход к развитию креативных (творческих) способностей студентов с учетом их психологических особенностей с направленностью на научно-исследовательскую и творческую деятельность. Поскольку изначально студенты, как правило, положительно мотивированы, то основные усилия преподавателя прилагаются к поддержанию устойчивого интереса к предмету. Работая с такими студентами индивидуально, преподаватель предлагает задачи, которые подразумевают нетривиальные подходы к решению, серьезные проекты, а если студент сам определяет для себя круг интересующих его проблем, то преподаватель выступает не в качестве учителя, в качестве консультанта. Такой прием практикуется, к сожалению, недостаточно широко, так как требует привлечения значительных ресурсов преподавателя.

Теперь о содержании курса в целом и особенностях потоков. Первые два семестра посвящены классическому программированию с применением традиционного ныне языка Паскаль как наиболее пригодного для воспитания хорошего стиля программирования и привития (или развития) навыков алгоритмизации, так как, несмотря на усовершенствование инструментальных средств, классические задачи информационных технологий остаются и они должны составить основу курса подготовки будущих разработчиков программного обеспечения. Это – математизация (переход от вербальной постановки к математической), алгоритмизация (поиск или разработка алгоритмов решения, оценка их ресурсоемкости, выбор эффективных алгоритмов), кодирование алгоритма на доступном языке (задача скорее техническая, чем творческая), выбор или разработка структуры и

способа хранения данных, эффективное использование памяти с применением, при необходимости, методов сжатия информации. Попутно поднимается многокритериальная задача эффективности программ, решение которой определяется наличием ресурсов, требованиями Заказчика, и порой она может быть сведена к одному критерию в стоимостном выражении. Очевидно, здесь речь идет в основном о программировании, а не об информатике.

#### **Методическое обеспечение и дидактические технологии**

В сегодняшних условиях на рынке литературы можно встретить немало справочников (порой непомерно дорогих), небольшое количество учебников и почти полное отсутствие сборников задач, необходимых для обеспечения учебного процесса. В условиях бурного развития информационных технологий книгоиздатель, а тем более авторы не успевают за процессом, а диспаритет цен блокирует библиотечный процесс. Для покрытия дефицита учебной литературы авторами совместно с коллегами разработан и выпущен сборник задач под названием «Практикум по программированию», в котором представлены около 600 задач по всем темам: от классического программирования до разработки типичных АРМ и ИПС. Материал учебного пособия собирался в течение многих лет из разных литературных источников и практических ситуаций и прошел все этапы: от карточек с задачами через методические пособия, первого издания под грифом АГУ до второго издания под грифом УМО. В ходе подготовки условия задач неоднократно переформулировались с целью их корректности и однозначного толкования. Около половины задач оригинальны, им нет аналогов в литературе.

Задачи сборника разбиты на 18 тем: 16 посвящены классическому программированию и отражают основные методы и приемы алгоритмизации, хранения информации, эффективности использования ресурсов; последние 2 темы содержат типичные задачи, требующие применения СУБД и электронных таблиц. В них по максимуму представлены информационные среды разных отраслей, нуждающиеся в компьютеризации.

Опыт применения задач и технологии обучения показал, что основной эффект в обучении программированию состоит в решении практических задач, а отнюдь не в прослушивании лекций и прочтении учебников. В этом ключе задачи сформулированы в основном в вербальной постановке из самых различных жизненных ситуаций. В каждой теме подобрано, как правило,

более 25 задач с тем, чтобы их можно было распределить среди студенческой группы без пересечения. Несколько лет назад в университете был проведен эксперимент по «модульному» обучению: учебный год делился не на 2 семестра, а на 4 модуля, с аттестацией по окончании каждого модуля, частично эта технология закрепились и сейчас продолжается. «Практикум по программированию» удачно вписался в него: 16 задач делятся нацело на 4 части с возможностью формировать годовые несовпадающие пакеты индивидуальных заданий. Разработан и программно реализован алгоритм рандомизированной выработки таких пакетов для студентов 1 курса, в рамках которого уже в начале учебного года каждый студент получает свой список заданий. Благодаря случайному выбору, все пакеты уникальны, что снижает до минимума несамостоятельность в разработке решений. Студентам предоставляется полная свобода в планировании своего времени; роль преподавателя сводится к обсуждению условий, консультированию разработок и участию в защите решений. При защите большое внимание уделяется тестированию программ, их надежности и пользовательскому интерфейсу.

Внешняя дифференциация сводится к разнице требований при защите разработок и разным критериям оценок, разному содержанию лекционных курсов. Так, если для «начинающих» подробно читается «ядро» Паскаля с многочисленными примерами, особенности его Borland-реализации, то для второго потока – «продвинутых» это проходит поверхностно, с углубленным изучением алгоритмов, технологий надежного программирования, тонкостей реализации и обеспечения эффективности, внутримашинных процессов.

Выработан и доводится до обучаемых в начале учебного года следующий алгоритм выставления оценок за год. По учебному плану, за 1 курс предусмотрено 2 зачета и экзамен. Студентам предлагаются следующие критерии оценок (в усредненном виде):

- 4 задачи – зачет за 1-й семестр;
- 8 задач – зачет за 2-й семестр;
- 9 задач – «удовлетворительно» на экзамене;
- 12 задач – «хорошо»;
- 16 задач – «отлично».

За преподавателем остается право увеличивать или уменьшать вес каждой задачи в зависимости от сроков ее защиты (работает механизм «инфляции» при запоздалой защите) и качества решения. Однако стало очевидным, что в пределах каждой темы задачи недостаточно сбалансированы по сложности, и поэтому при подготовке второго издания они проранжированы по 10-балльной шкале с учетом сложности и трудоемкости, соответ-

ственно скорректированы критерии выставления оценок.

Плановый экзамен за первый курс проводится по классической схеме:

- билеты, в которых 2 теоретических вопроса и 1 задача (программирование в «бумажном» варианте);

- результативная оценка на экзамене не является абсолютной, а приплюсовывается к сумме баллов, накопленных на задачах лабораторных работ.

Авторы считают такой подход правомерным, ибо теоретическая часть курса не отличается строгостью, а основным критерием должно быть умение применять знания на практике. Экзамен не является обязательным, студентам предоставляется право с его помощью пополнить сумму баллов, накопленных на лабораторных работах; широко практикуются «автоматы».

Третий и четвертый семестры – знакомство с СУБД, разработка программных средств для АРМ и ИПС, знакомство с офисными приложениями и технологиями компьютерного делопроизводства. Здесь внешняя дифференциация в полной мере еще не реализована, но планируется по следующей схеме. «Разработчикам» предлагается по 2 темы – информационные среды, нуждающиеся в автоматизации (применении СУБД и ИПС), и по 1 задаче, решаемой посредством электронных таблиц. На первом этапе предполагается проектирование реляционной базы данных в виде связанных таблиц с полной защитой структуры и содержания; демонстрация умения обслуживать эти базы и исполнять запросы потенциального пользователя в режиме программиста-консультанта (как критерий зачета за третий семестр), на втором – разработка полного интерфейса по обслуживанию одной из спроектированных баз, формирование входных и выходных форм документов, удовлетворение поисковых запросов пользователя без участия разработчика. Кроме этого, предусматривается выполнение несложных расчетов с применением электронных таблиц, подготовка документации к разработанному АРМ или ИПС в среде офисных приложений и с соблюдением элементарных требований компьютерного делопроизводства. Среда разработки жестко не задается и определяется наличием инструментальных средств и опытом преподавателей. Это опять-таки из области программирования.

«Пользователи» на втором курсе в основном будут заниматься офисными приложениями с основами делопроизводства и издательских технологий, то есть ликвидацией пробелов в школьной программе, но применительно к использованию, а не к разработке программных средств. Предполагаемые разде-

лы: текстовые процессоры, правила подготовки деловой и научной литературы, электронные таблицы, математические пакеты, графические средства, знакомство с сетевыми технологиями, работа в Интернет. По проведенной классификации – это приложения чистой информатики.

После второго курса студенты выбирают специализацию в различных направлениях математики, и необходимо за два года подготовить разработчиков либо пользователей (и учителей пользователей) среднего уровня квалификации, так чтобы выпускники, независимо от специализации и уровня образования, могли надеяться на нормальное трудоустройство. Дальнейшая специализация только шлифует полученные навыки и позволяет уверенно окунуться в современные технологии.

Таким образом, проведенное разграничение между программированием и информатикой тесно увязывается с дифференциацией в подготовленности абитуриентов и программа учебного процесса позволяет снять некоторую напряженность в изучении рассматриваемого предмета, а моделирование широкого спектра реальных жизненных ситуаций, их вербальная постановка с необходимостью прохождения полного курса алгоритмизации, программирования, тестирования и защиты программных средств, повышенное внимание к надежности и уровню комфортности пользователя обеспечивают подготовку разработчиков, пользователей и преподавателей достаточно высокого уровня и предоставление обучаемым максимальной самостоятельности и объективные критерии оценок результатов, снимают лишнее напряжение во взаимоотношениях преподавателя и студента.