Компьютерные инструменты в образовании, 2015
№ 6: 43–66
УДК: 004.9, 378.016, 378.146, 378.147, 378.4
http://ipo.spb.ru/journal

ОПЫТ ПРОВЕДЕНИЯ ПРАКТИКУМОВ ПО ПРОГРАММИРОВАНИЮ НА МАТЕМАТИКО-МЕХАНИЧЕСКОМ ФАКУЛЬТЕТЕ СПбГУ

Чернышев Георгий Алексеевич

Аннотация

В данной работе рассказывается об опыте проведения практикумов по программированию на математико-механическом факультете СПбГУ у студентов специальностей, ориентированных на подготовку программистов. Описывается формат занятий, методы оценивания успеваемости, освещенные темы, разработанные модули учебных программ и используемая в курсе литература. Подробно описывается опыт применения методики домашнего чтения с проверкой усвоенного материала. Рассматривается входное тестирование, по результатам которого программа курса адаптируется для каждой группы. Наконец, в работе представлен обзор подобных методик как с практической стороны — указываются примеры подобного устройства занятий из Computer science curriculum 2013, так и с теоретической — разбираются научные работы, исследующие роль домашнего чтения по специальности в обучении студентов. Данная статья является попыткой поделиться удачным опытом построения программ.

Ключевые слова: *практикум по программированию, математико-механический факультет, СПбГУ, опрос, организация занятий, учебная программа.*

1. ВВЕДЕНИЕ

Автор ведет практикумы по программированию на кафедре информатики математико-механического факультета СПбГУ уже более пяти лет. Данная работа представляет собой попытку описать методику преподавания, которую он выработал за это время. Также будет проанализирован опыт, полученный за это время.

Автор начал свою работу на кафедре в 2010 году, когда язык Паскаль еще был достаточно популярным языком для преподавания программирования. Проведя два семестра с использованием языка Паскаль, автор поставил для себя вопрос: продолжать ли использовать его или модернизировать учебную программу путем смены языка. Основной тезис в пользу сохранения языка был следующий: в этом, базовом, курсе преподаются не конкретные языки или технологии, а фундаментальные концепции — алгоритмы и структуры данных. Промышленные и специализированные языки входят в программу старших курсов. Кроме того, студенты могут их изучить на дополнительных занятиях, проходящих на нашем факультете [20].

Другие соображения в пользу сохранения были следующие:

1. Язык Паскаль — это язык, специально созданный для обучения [63], он прост и понятен. Для «первого» языка программирования это существенный плюс.
2. Поступившие студенты уже обычно знакомы с данным языком программирования из школьных занятий. Среди заданий первичного тестирования (разбираются в третьей части) присутствует специальный вопрос, проверяющий знание языка Паскаль. Согласно этому тестированию, с Паскалем не были знакомы всего несколько студентов за пять лет.
3. Для этого языка существует много обучающих курсов, программ и других материалов. Имеется большое количество качественной и, что самое главное, недорогой литературы.
4. Достоинства языка и окружения: простой и понятный синтаксис, типобезопасность, короткая спецификация (90 страниц против 500 у Java) [48], ясные сообщения об ошибках и т. д.

Автор считает что, исходя из этих соображений, можно использовать язык Паскаль в течение одного семестра у непрофильных или слабых групп. Впрочем, как будет показано дальше, консенсуса по вопросу, какой язык использовать в качестве первого, в мире до сих пор нет. Однако у языка Паскаль и окружения есть следующие серьезные недостатки:

1. Паскаль не может продемонстрировать концепции современных языков программирования [41]:
   (a) Из крупных диалектов перегрузка операторов присутствует только у PascalABC.NET [53] и Free Pascal [34]. Однако у них разный синтаксис и набор предоставляемых возможностей.
   (b) Средства обобщенного программирования, такие как шаблоны и дженерики, либо отсутствуют, либо их синтаксис диалекто-специфичен. Например, можно сравнить Free Pascal [45, 66] и PascalABC.NET [1].
   (c) Слабая поддержка элементов функционального программирования. Например, лямбда-выражения отсутствуют в диалекте Free Pascal [35, 65], а в PascalABC.NET появились только в середине 2010 года [57], причем на момент написания статьи работа над ними продолжается.
   (d) Отсутствие единой библиотеки стандартных контейнеров. Имеющиеся библиотеки диалекто-специфичны, при этом даже для одного диалекта существует множество библиотек [40]. Кроме того, эти библиотеки проигрывают по функционалу библиотекам индустриальных языков программирования. Это серьезный недостаток, так как одним из навыков современного программиста является умение выбрать подходящие средства, которые им предоставляет платформа.
2. Фрагментированность инструментария IDE: различные реализации предоставляют лишь часть из такого важного функционала, как интеграции с системой контроля версий, юнит-тестирования, мощного редактора и т. д. Полного набора ни для одного из диалектов автор найти не сумел. Например, в Lazarus есть модуль юнит-тестирования FPCUnit [42], однако нет законченного плагина для Subversion [50].
3. Малое сообщество. Например, на сайте вопросов и ответов Stackoverflow [55] в разделе, посвященном Free Pascal, 680 вопросов, а в разделе по Java [56] почти миллион. Конечно, нельзя сравнивать один диалект учебного языка и индустриальный язык, но общую картину это передает.

4. Отрицательная репутация языка Pascal среди студентов, что приводит к падению мотивации к обучению. Поступая в вуз, студент ожидает преподавания современных языков программирования, таких как C# или Java. По опыту автора, встреча с языком Паскаль приводит к падению посещаемости и утрате интереса к курсу.

В начале нулевых язык Паскаль еще преподавался в вузах по всему миру: около 2003 в Новой Зеландии (School of Information Technology and Electrotechnology Otago Polytechnic) [46], около 2004 на Кипре (Near East University) [38], в 2004 году в США (Denver University) [39], в 2008 в США (University of Houston-Downtown [69] и Texas A&M [67]).

Более того, в некоторых университетах США его продолжают преподавать и на момент написания статьи: Lee University [51], University of Houston-Clear Lake [70], Louisiana State University-Eunice [52].

Но, конечно, Паскаль однозначно выбыл из гонки: в настоящее время он остался в отдельных вузах и на непрофильных специальностях. Например, уже на момент 2001 года из 39 аккредитованных австралийских вузов Паскаль (Delphi) преподавался всего в одном [41].

Проанализировав аргументы, автор принял решение о серьезной модернизации тех программ, по которым он вел курсы. Опыт создания подобных программ и опыт проведения занятий по ним описывается в настоящей статье.

В следующей части мы рассмотрим предпосылки и историю появления метода в его исходном виде. Затем, в третьей части будет представлена анкета и разобрано тестовое задание. Данное задание используется на первом занятии курса для определения уровня каждого студента и группы в целом. В четвертой части будут рассмотрены четыре класса навыков, которые необходимо выработать при подготовке программистов. В этой части будет приведена общая структура занятия и будет описано, как именно эти классы навыков преподаются в курсах автора данной работы. Далее, в пятой части будет подробно разобран основной компонент данного метода — домашнее чтение и опросы. Будут разобраны вопросы подготовки и проверки результатов опросов, имеющиеся модули и освещенные в них темы, а также будут описаны сложности данного метода. В шестой части будут приведены результаты подхода в целом. В седьмой части будет приведен обзор подобных подходов как использующихся на практике, так и экспериментальных, представленных в научных публикациях по данной тематике. Наконец, в восьмой части подводятся итоги и обсуждаются возможные направления развития метода.

2. ПРЕДПОСЫЛКИ ВЫБОРА МЕТОДА

В 2011 году автор начал работать с профессором математико-механического факультета СПбГУ Александром Львовичем Тулупьевым и его группой. А.Л. Тулупьев познакомил автора с основами метода, о котором рассказывается в данной статье. В новом учебном году для новой группы Александр Львович предложил использовать следующую методику: отобрать доступную литературу и скомпоновать главы выбранных книг для выдачи в качестве домашнего чтения. Затем, сделать упражнения по темам для работы в классе [6]. Для контроля чтения было совместно подготовлено несколько вариантов опросов, из 5–6 заданий, приблизительно на четверть часа письменной работы. Опросы планировалось проводить каждое занятие. Несмотря на то, что группа была самая слабая за всю практику автора, уровень группы стал заметно расти.

На весенний семестр 2011/12 учебного года автору выделили новую профильную группу. Преподавание у этой группы студентов велось на языке программирования Java.

Во время подготовки программы выяснилось, что на занятиях затруднительно освещать современные языки программирования ввиду их сложности. Уходит слишком много времени на то, чтобы обучить языку на приемлемом уровне, и не остается времени на прием или разбор задач. Для решения этой проблемы пригодились уже подготовленные опросы по домашнему чтению. После этого семестра автор стал самостоятельно создавать содержание опросов по новым темам, начал искать и готовить литературу. За это время метод был серьезно переработан и видоизменен.

Другой источник «вдохновения» — это сертификационные экзамены IT компаний, прежде всего компании Sun Microsystems. Автор сдал несколько экзаменов в 2007–2008 годах и был приятно поражен самим подходом и качеством материалов для подготовки.

Подход состоял в следующем: Sun Microsystems предлагал подготовиться и сдать экзамен на знание языка программирования Java или тематических библиотек. Успешно сданный экзамен давал звание Sun Certified Professional в зависимости от области. Это звание достаточно ценилось как вклад в резюме, в то же время несколько компаний материально поощряли сотрудников, сдавших экзамен. Экзамен состоял из нескольких десятков вопросов, на которые надо было ответить в течение заданного промежутка времени, обычно нескольких часов. Во время экзамена ничем нельзя пользоваться. Экзамен сдается на компьютере, на вопросы нужно отвечать, выбирая варианты или вписывая ответы. Впрочем, экзамены высоких ступеней предполагали работу с живым экзаменатором.

Материалы для подготовки [64] очень хорошо систематизированы, изложение покрывает множество деталей языка, имеется большое количество примеров. Книга содержит два полных теста по всему материалу и набор вопросов с ответами в конце каждой главы. Автор до настоящего времени не встречал материалов по Java такого высокого качества. К сожалению, книга написана на английском языке, в российских университетских курсах ее использовать не представляется возможным.

Наконец, последним хоть и косвенным источником «вдохновения» был тот объем внимания, уделяемый чтению в ведущих мировых университетах, а именно кружки чтения текстов по специальности (reading groups) [61] [59] [60]. Такие кружки — добровольные сообщества заинтересованных студентов и преподавателей, регулярно встречающихся в неформальной обстановке для обсуждения каких-либо текстов по специальности.

3. ТЕСТИРОВАНИЕ И АНКЕТА

Приступая к работе с новой группой, автор начинает со знакомства, состоящего из заполнения небольшой анкеты и тестирования. Этому полностью посвящается первое занятие. Знакомство необходимо для подстройки дальнейших занятий под группу.

Анкета содержит несколько групп вопросов. Первая группа — вопросы общего характера, такие как ФИО, номер группы и прочее. Во второй группе находятся вопросы о способах связи: адрес электронной почты, телефон, а также логин в университетской системе. Последнее позволяет настроить удаленный доступ к папке с заданиями или другими материалами, которые предполагается выдавать в ходе работы. Это весьма важная возможность, так как позволяет не выдавать лично каждому студенту материал и тем самым экономит время на выдачу материалов. Также логин необходим для выдачи доступа к университетской системе контроля версий, используемой в курсе. Наконец, по-

следняя группа вопросов выявляет уровень профессиональной подготовки. В этой группе вопросов раздельно по каждой категории выясняется:

1. Изученный в школе материал. Он может весьма различаться, часто бывает представлен весь спектр, начиная от отсутствия информатики в школьной программе и изучения офисных пакетов, кончая достаточно глубокими знаниями какого-либо объектного или даже функционального языка. Например, у автора были студенты, занимавшиеся в школе по книге «Структура и интерпретация компьютерных программ» [7].
2. Навыки, приобретенные вне школы. К данным навыкам относятся навыки, приобретенные в каких-либо кружках программирования, курсах или же самостоятельно. Там также может быть довольно пестрый набор навыков. В последние годы все чаще появляются студенты, самостоятельно изучившие веб-технологии. Данный класс технологий довольно легок в освоении и при этом предоставляет широкий выбор возможностей для их монетизации.
3. Навыки работы в операционных системах, отличных от Windows. Этот вопрос позволяет выявить студентов, с одной стороны с подготовкой, а с другой стороны — действительно интересующихся вопросами, связанными с программированием.
4. Участие и успехи в олимпиадах, причем не только по информатике. Это тоже является весьма важным фактором для выявления подготовленных и мотивированных студентов.
5. Знание иностранных языков. Ни для кого не является секретом, что основная часть качественной литературы по нашей специальности — переводная, то же касается и многочисленных руководств, справочников, инструкций и других материалов. Незнание английского языка «отрезает» студента от огромного пласта материала. Ему придется пользоваться более узким набором литературы, часто устаревшим. Таким образом, будет полезно установить, имеет ли смысл факультативно рекомендовать те или иные источники на иностранных языках. У автора были как группы, которые практически в полном составе имели достаточный уровень подготовки для чтения любой технической литературы на английском языке, так и группы без такой возможности.

В целом, анкета рассчитана на полчаса работы. Остальное время из двух часов отдается на тестирование.

Материал тестирования полностью приведен в приложении А. Всего имеется десять заданий, из них пять практических на реализацию, еще пять — теоретического характера. Задания составлены таким образом, чтобы можно было проверить знания по темам, преподаваемым в университетском курсе программирования в течение первых двух-трех семестров. Задания размечены звездочками в зависимости от уровня сложности.

Первое задание позволяет проверить как базовые знания языка программирования, так и умение писать программный код и понимать его. Паскаль был выбран как наиболее часто преподаваемый в школе язык программирования. Второе задание проверяет умение думать, знание побитовых операций, а также косвенно характеризует олимпиадных программистов. Задание номер три специально неясно сформулировано и снабжено примером без иллюстрации. Цель этого задания[1] — выявить тех, кто будет спраши-

[1] В задании имеются в виду бусы, в которых соединение достигается не с помощью нити пропущенной через отверстие в середине, а с помощью нити вложенной в кайму, выбитую по окружности бусины, так чтобы бусину можно было «обвязать». Такой способ крепления позволяет «крутить» бусы, фиксируя сверху одну бусину и меняя местами правую и левую части. То есть фактически имеется в виду просто переворот обычных бус.

вать, попытается разобраться, прежде чем писать программный код. Кроме того, это задание практически безошибочно выявляет всех «спорщиков». Как правило, пара таких студентов бывает каждый год. Четвертое задание требует умения работать с указателями, а также аккуратности в разборе частных случаев. Наконец, пятое задание проверяет практическое владение основами ООП.

Теоретические задания проверяют знание наиболее важных понятий из теоретического курса, которые можно успеть спросить за половину занятия. Кроме того, проверяется умение систематически излагать материал, например, на некоторые вопросы[2] можно ответить кратко, а можно дать весьма объемный ответ. В целом, после ответа на теоретические вопросы будет приблизительно ясен уровень тестируемого, его «семестр» по теоретическому курсу университета.

По итогам анкеты и тестирования выбирается учебная программа на последующие три–четыре семестра. В зависимости от уровня группы учебные модули могут быть пропущены, заменены или растянуты во времени. Также возможно увеличение количества однотипных задач по определенной теме. Таким образом, происходит подстройка программы под группу.

4. ОПИСАНИЕ МЕТОДА

4.1. Общие соображения

Автор выделяет четыре основных класса навыков, которые необходимы современному программисту:

1. Навыки программирования: алгоритмическая составляющая.
2. Навыки программирования: языковая составляющая.
3. Различные технологические практики.
4. Навыки работы с источниками информации по предметной области.

Первоочередная задача вводного практического курса — развитие навыков программирования у студентов. Эти навыки можно разделить на алгоритмическую и языковую группы. В алгоритмическую группу входят следующие навыки: выбор алгоритма из возможных альтернатив, составление алгоритма, сравнение алгоритмов, границы применимости изучаемого алгоритма и так далее. В языковую группу входят: выбор языка для выражения алгоритма, правильное выражение алгоритма средствами изучаемого языка и библиотеки, взаимосвязь выразительных средств одного языка, парадигмы программирования и так далее. В статье «Как готовить системных программистов» [18] А.Н. Терехов отмечает, что необходимы оба компонента, просто знать язык программирования недостаточно. При этом изучение языка программирования тоже достаточно сложная задача.

Алгоритмическая подготовка дается в теоретическом курсе, а навыки компоновки существующих и разработки новых алгоритмов преподаются на практических занятиях. Языковая часть также преподается в теоретическом курсе, однако больший упор на это следует делать на практике.

Касательно языковой части, автор сторонник идеи, что надо преподавать не языки, а парадигмы на примере языков. Языки достаточно быстро меняются, зависят от пользовательской популярности, разработчиков и коммерческого успеха. В свою очередь, по-

[2] Вопросы два и пять, см. Приложение А.

нимание парадигм позволяет быстро изучить новый язык за счет знания принципов устройства языков данного класса.

Для демонстрации различных парадигм студенты знакомятся с качественно различными языками. В своем основном четырехсеместровом курсе автор начинает с языка C#, затем переходит на C и, наконец, рассматривает C++ (включая стандарт C++11). Таким образом, разбираются процедурное программирование, объектно-ориентированное, обобщенное программирование и присутствуют элементы функционального программирования. К сожалению, в настоящее время у автора отсутствует отдельный модуль функционального программирования.

По мнению автора, одна из задач преподавателя вводного курса практики программирования — не только научить алгоритмам и парадигмам языков, но и максимально познакомить студентов со спецификой современного программирования. Конечно, ввиду временных ограничений, полномасштабное знакомство на младших курсах не осуществить. Однако можно показать отдельные элементы и заложить понимание основ. Дальше студент доучится самостоятельно на старших курсах, на производственной практике или уже на работе. Кроме того, технологические практики эволюционируют быстрее, чем, например, алгоритмическая часть.

К технологическим практикам автор относит навыки использования систем контроля версий, навыки юнит-тестирования, навыки форматирования кода и отладки, навыки создания документации в системах наподобие Javadoc, системы сборки, системы отслеживания ошибок.

К навыкам работы с информацией относятся навыки поиска и оценки качества документации. На первый взгляд, студенты могут развить эти навыки самостоятельно, однако существенный объем документации — англоязычный, и студенты младщих курсов не всегда могут оценить ее качество. Интернет изобилует списками рассылок, пользовательскими группами, форумами и другими ресурсами, где обсуждаются вопросы предметной области курса. Данный класс навыков отвечает за способность самостоятельно развиваться, поэтому очень важно привить их как можно раньше. Это позволит студенту начать профессионально расти, снизит его зависимость от материала, дающегося на занятиях. Поэтому автор считает, что необходимо уделить этому вопросу время.

В следующем разделе представлена реализация описанного выше плана. В реализации практически полностью покрываются три из четырех разделов Software Development Fundamentals (SDF) из Computer Science Curriculum 2013 [68]. Более того, в плане автора рассматриваются отдельные области, отсутствующие в SDF, например, базовые алгоритмы на графах.

4.2. Реализация подхода

Основное содержание практических занятий составляют задания на реализацию. У автора накоплен солидный объем заданий, несколько сотен. Задания разделены по нескольким темам: применение рекурсии, списочные структуры, древовидные структуры данных, алгоритмы на графах, объекты. В каждой теме есть три уровня задач, в зависимости от сложности. Каждый уровень задач оценивается по-разному (обычно 1–3 балла). Для зачета по теме каждому студенту необходимо набрать некоторое количество баллов в зависимости от уровня группы. Таким образом, закрыть тему могут не только сильные студенты, которые могут решить любые задачи, но и слабые. При этом слабым достается больше задач, но простых, что позволяет им отработать базовые навыки по теме. Кроме того, в самых сложных задачах, предполагающих реализацию какой-то

системы, присутствует некоторый простор для интерпретации условия и возможность свободы творчества. Например, один студент изучил самостоятельно технологию WPF и решил с ее помощью задание, так как Windows Forms показалась ему достаточно скучной. Или другой пример: вместо реализации предложенной многопоточной структуры данных, студент самостоятельно нашел и изучил дипломную работу, сравнивающую несколько вариантов подобных структур. По результатам обзора была реализована структура данных, более сложная и интересная, чем исходно предложенная. Все эти изменения делались по предварительной договоренности с преподавателем и с обоснованием своего выбора.

В курсе автора преобладают домашние, а не аудиторные задания. Домашние задания студенты могут сдавать на любом занятии до конца семестра. Аудиторные задания следует сдать до конца текущего занятия. Большинство заданий носит индивидуальный характер, то есть у каждого студента будет свое задание. Иногда имеет смысл давать единые задания для всех. В таком случае это либо аудиторные задания для осуществления контроля, либо домашние задания, которые выдаются для того, чтобы слабые студенты научились решать определенный класс заданий. Групповые задания практически отсутствуют, так как сложно оценить вклад каждого участника в решение. С большой вероятностью сильный студент из группы сделает основную часть работы, так как ему обычно интереснее, чем слабым. Возможна утрата преподавательского контроля, когда сильные студенты руководствуются принципом «к сроку сделаем» и таким образом откладывают сдачу, не показывая промежуточные версии. Опыт показал, что давать такие задания не имеет особого смысла.

Все задания без исключения принимаются лично, не по электронной почте. Это позволяет проверить понимание написанного. В исключительных случаях, когда сдается большой проект, автор просит выслать код заранее, чтобы изучить и подготовиться к сдаче.

На практических занятиях автор обучает студентов не только навыкам программирования, но и знакомит с технологическим аспектом профессии программиста. В первом семестре дается простое введение в системы контроля версий, а также даются практические задания с использованием системы Subversion. К сожалению, часы и структура занятий не позволяют проводить обучение продвинутым способам использования системы. Для полноценного освоения данных инструментов требуются групповые задания и достаточно большие проекты, что невозможно в условиях младших курсов. Однако на базовом уровне эти знания преподаются, студентам объясняются понятия рабочей копии, репозитория, ревизии, конфликта. Они на практике обучаются базовым навыкам, таким как: взять код (check out), положить код (check in), сравнить ревизии, разрешить конфликт и провести поиск в репозитории.

Отладчики, анализаторы кода и профилировщики также прорабатываются в курсе. В первом семестре студенты учатся работать с графическим отладчиком, встроенным в среду разработки Visual Studio. В последнем семестре сильные группы знакомятся с линейкой инструментов Valgrind для ОС Linux, учатся работать с детектором утечек памяти и гонок в многопоточных приложениях, а также могут попробовать модуль профилирования Callgrind. Несмотря на то что основным языком при программировании в ОС Linux является C, можно отметить, что линейку инструментов Valgrind можно использовать и для языка программирования Паскаль.

Не оставлен в стороне вопрос о модульных тестах (unit-tests). Буквально на втором занятии объясняется суть модульного тестирования и показывается его применение на

практике. Данный инструмент не только необходим любому современному программисту, он еще и невероятно полезен преподавателю. Дело в том, что он позволяет снять часть нагрузки.

В большинстве случаев автор требует несколько самостоятельно написанных тестов: тесты пишутся к основному методу и проверяют корректность решения на различных примерах. Для таких системных тестов также используется фреймворк модульного тестирования. Написанные тесты дают хоть какую-то гарантию, что на первую проверку будет представлена корректно работающая программа. Хотя, конечно, тесты не избавляют преподавателя от необходимости дополнительно проверять задачу. Кроме того, наличие тестов в случае задачи, выданной всем, позволяет проводить следующую проверку: взять тесты у одного студента и проверить ими решение другого.

Форматирование кода также разбирается на одном из первых занятий. Объясняются общие моменты и смысл данной практики. Затем студентам выдаются ссылки на веб-ресурсы, описывающие различные стили форматирования кода. Студенты самостоятельно знакомятся с ними и выбирают для себя определенный стиль. При проверке заданий учитывается не только алгоритм и качество реализации, но и форматирование кода. В случае неудовлетворительного форматирования задание может быть отправлено на доработку с требованием устранить дефекты.

На занятиях также обращается внимание студентов на важность сообщества и культуры. Показываются следующие ресурсы: Stackoverflow, форумы MSDN, Github, тематические рассылки и прочее. Автор рассказывает, как ими пользоваться в режиме чтения, а иногда и рассказывает, как правильно туда писать. Показываются не только хорошие ресурсы, но и плохие, с объяснением, почему ими не следует пользоваться. Для современного программиста знание этих ресурсов и умение ими пользоваться можно считать обязательным.

Наконец, для того чтобы заинтересовать студентов, автор разбирает задачи наподобие описанных в книге Java Puzzlers [37]. Это задачи на иллюстрацию интересного поведения языка, его подводных камней. Они не только поучительны, но еще и довольно интересны. Они позволяют существенно повысить интерес к занятиям даже у большинства хорошо подготовленных студентов. К сожалению, подобной книги для других языков программирования автор не нашел, материал для этой части занятия приходится собирать самостоятельно.

4.3. Домашнее чтение и опросы

Кроме обычных практических заданий на реализацию, каждую неделю студентам выдается задание — изучить некоторый материал из книг по тематике курса. Данный материал заранее выбирается преподавателем и составляет приблизительно от 80 до 150 страниц в неделю. Проверка выполнения этих заданий выполняется путем опроса. Опрос проводится на занятии в письменной форме, без возможности пользоваться литературой и компьютерами. Длительность опроса зависит от сложности и объема материала. В начале первого семестра, когда группа учится писать опросы, дается по три минуты на вопрос. Позднее норма снижается до двух с половиной минут. Если вся группа серьезно не укладывается во время, то дается дополнительное время, но обычно не более десяти минут.

Изначально в 2011/12 учебном году автор проводил короткие опросы по относительно маленьким фрагментам, укладывающимся в 30-40 страниц. Однако было замечено,

что добросовестные студенты перед опросом читают свои рукописные конспекты. При этом эти студенты сдавали опросы лучше всех в группе. Автору понравился систематический подход, и было разрешено пользоваться рукописным конспектом на самих опросах. Это сразу же повысило качество результатов и позволило увеличить объемы опросов. Первые опросы состояли из 5-6 простых вопросов и редко занимали более 15 минут. Соответственно, их результатом была простая проверка факта прочтения материала. Рукописный конспект позволил расширить опрос до 12-15 вопросов и 45 минут. Такой длинный опрос уже давал возможность контролировать качество усвоения материала.

Кроме конспектов автор также советует использовать карты памяти (mind maps [72]). Однако они не получили популярности у студентов, хотя про них рассказывается каждой группе.

В некоторых случаях преподаватель может сделать доклад по теме будущего опроса, чтобы помочь студентам лучше подготовиться. В практике автора этот метод показал себя неэффективным — на итогах опроса он отражается слабо, однако доклад занимает от получаса до 40 минут. Кроме того, подготовка презентации так же требует нескольких часов времени. Пример такой презентации может быть найден по адресу [32].

В одном семестре возможно провести от шести до девяти опросов. Несмотря на то, что в семестре гораздо больше занятий, имеет смысл остановиться именно на этом объеме. Причин несколько:

1. Есть сложные опросы, к которым необходимо готовиться более недели.
2. Автор учитывает пожелания студентов и может перенести опрос, если на той же неделе группе предстоит какая-то серьезная работа, к которой надо готовиться — коллоквиум или контрольная.
3. Наконец, необходимо оставить время на переписывания.

Методика опросов позволяет проводить удаленный опрос (по договоренности): в назначенное время текст опроса выдается студентам (высылается на почту, выкладывается в систему контроля версий и т. д.). При проверке учитывается время отправки или коммита. В данном случае можно ожидать некоторого улучшения результатов, однако небольшого: опросы требуют умения работать головой, а не просто искать материал.

В опросах автор старается спрашивать только концепции и идеи. Опросы не содержат вопросов наподобие «перечислите сигнатуры функций из такого-то заголовочного файла». Опрос не должен сводиться к зубрежке, он должен быть содержательным и решаемым. Каждый вопрос должен предполагать короткий ответ, который можно написать за три минуты.

Для проверки работы с кодом в условиях ограниченного времени автор дает два типа вопросов:

1. Написать код. Обычно при создании вопроса автор предполагает, что достаточно будет написать 5-10 строчек кода. Как показала практика, код писать сложнее, чем текст, поэтому в опросах автор не требует писать много кода. Это исключает возможность требовать написания больших программ.
2. Покритиковать или модифицировать код. Гораздо легче критиковать и модифицировать уже написанный код, поэтому вопросы, направленные на проверку навыков работы с большим объемом кода, формулируются в виде «проанализируйте написанный код», «модифицируйте код».

Каждое задание оценивается в 1, 0.75, 0.5, 0.25 и 0 баллов. Баллы по всем заданиям суммируются, после этого производится ранжирование работ и установление порога за-

чета, то есть если студент набрал определенное количество баллов, работа считается зачтенной. Помимо этого, работы по сходным результатам группируются, что позволяет разбить студентов по уровню успеваемости. Это позволяет следить за динамикой успеваемости с течением времени.

При выдаче проверенной работы студент получает комментарии: что в ответе повлияло на оценку. В положительную сторону может повлиять материал, которого не было в литературе, или демонстрация глубокого понимания материала, каких-то неочевидных соображений. В таком случае студент может заработать дополнительные 0.5 балла.

Результаты проверки (численные) заносятся в таблицу, а пересчет в группы производится на занятии. Обычно для получения зачета за опрос достаточно набрать более половины возможных баллов. В зависимости от результатов и сложности опроса порог может быть повышен.

Разбор опросов, апелляция и обсуждение проводятся на следующем занятии. В зависимости от сложности опроса и подготовки группы, этот разбор может занимать от пяти минут до получаса.

Методика опросов по домашнему чтению предполагает переписывание, поэтому при подготовке опроса создаются два-три варианта вопросов. Эти варианты обычно выдаются по убыванию сложности, то есть каждый последующий вариант проще предыдущего. После исчерпания вариантов обычно устраивается устная проверка. Переписывания начинаются в конце второго месяца занятий, по готовности студентов.

5. ДОМАШНЕЕ ЧТЕНИЕ И ОПРОСЫ: ПОДГОТОВКА ОПРОСА, ИМЕЮЩИЕСЯ МОДУЛИ И ОСВЕЩЕННЫЕ ТЕМЫ, СЛОЖНОСТИ И СРОКИ

5.1. Подготовка к проведению опроса и его проверка

Описываемый метод предполагает, что студент должен существенную часть времени посвящать самостоятельной работе с литературой. Для того чтобы начать работу со студентами, необходимо вначале создать опросы и в целом модуль опросов.

При создании модуля опросов автор сначала выбирает список тем, которые предполагается осветить на занятиях в семестре. Учитывается количество занятий в семестре, курс и уровень подготовки. Затем формируется список литературы: подбираются имеющиеся в продаже книги на русском языке. Согласно правилам, использовать англоязычные источники запрещено, поэтому, к сожалению, выбор довольно ограничен.

Затем автор начинает готовить опросы. Опыт показывает, что подготовка одного опроса занимает один день рабочего времени, иногда два, в зависимости от сложности темы. Это серьезная затрата времени, однако она одноразовая, так как вопросы не меняются из года в год. В неделю можно формировать по одному опросу в комфортном темпе. Такой темп позволяет не «выгореть» быстро и сделать за семестр один модуль опросов.

В целях подготовки опроса автором прочитывается весь материал, по ходу чтения записываются вопросы и возможные задания. Таким образом, после прочтения набирается несколько десятков вопросов, из которых формируются два или три варианта одного опроса. Полученные варианты используются для переписывания. Автор старается делать варианты неравными по сложности: опросы для переписывания выдаются от более сложных к более простым. Это позволяет, с одной стороны, поощрить сильных, а с другой стороны, — дать возможность слабым студентам пройти опрос.

Автор старается покрыть все подразделы читаемого материала, но так, чтобы уложиться максимум в 15 вопросов в одном опросе. Если размер главы небольшой, то можно добавить вопрос с написанием кода по данной теме. При составлении опроса автор старается сделать несколько вопросов на каждый подраздел, так чтобы было равномерное покрытие. Хотя, если один подраздел важнее остальных, можно сместить акценты в опросе.

Для проверки книг и самих вопросов автор привлекает коллег из индустрии. При этом их мнение является решающим, особенно касательно языков, в которых автор не имеет индустриального опыта.

Проверка работ по опросу также занимает время. Обычно на одну группу из десяти человек уходит от 40 минут до двух часов, включая усилия по разбору почерка. Иногда у сильной группы можно получить неожиданный ответ, который выходит за рамки материала. Его тоже надо оценивать, а это требует времени.

Более того, довольно часто сильные студенты выбирают книги по заявленным темам самостоятельно. Их мотивировка следующая: неинтересно читать или тяжело пробираться через плохую грамматику и некачественный перевод. Практика показала, что знание, полученное таким «вольным» способом, также позволяет успешно сдавать опросы.

5.2. Модули опросов

За прошедшие пять лет у автора накопилось достаточно много материала по различным областям курса с использованием различных языков. В данном подразделе перечисляются имеющиеся модули и приводится их краткое описание. Ссылки на детальные программы модулей могут быть найдены на сайте автора, в разделе «teaching» [2].

Каждый из описанных модулей рассчитан на длительность в один семестр или менее. Модули даются в определенной последовательности, так как имеют различные требования к обучающимся. Например, модуль по паттернам проектирования предполагает знание языка C++. Работая с новой группой, автор обычно начинает с языка программирования C# или Java и ведет ее два семестра. В третьем семестре автор дает модуль с языками C и C++. Затем, в четвертом семестре, в зависимости от уровня группы либо изучается программирование на языке C под ОС Linux, либо глубже прорабатываются уже пройденные модули. Например, если группа слабая, то в начале третьего семестра автор может принять решение об удлинении модуля по C++ за счет увеличения интервала между опросами или за счет разбивки имеющихся опросов на части.

В настоящее время у автора имеются следующие модули:

1. Язык программирования Java (курс был сделан совместно с командой А. Л. Тулупьева) — в основном, по книгам [10, 11], а также [24]. У автора это был первый опыт создания курса с опросами. Модуль рассчитан на два семестра и включает почти все главы в первой книге [10], а также некоторые во второй (работа с базами данных, Swing). В настоящее время используется редко.
2. Алгоритмы (также совместный курс) — по книгам [15, 21, 22, 29]. Этот модуль рассчитан на два семестра и освещает следующие темы: графы, деревья поиска, хеш-таблицы, линейные списки, рекурсию и комбинаторику. Привязан к курсу по языку Java. В настоящее время используется редко.
3. Программирование на языке C#, вводный курс по книге Стиллмена и Грин [13]. Этот курс рассчитан на студентов с очень слабой подготовкой или даже на студен-

тов «с нуля», никогда не программировавших. Курс позволяет ознакомиться с технологией Windows Forms, освоить IDE и практически сразу же начать писать простой код. Автор часто сталкивается с критикой, что эта книга неудачна для этих целей, потому что полезная информация слишком «размазана» по материалу; предлагаются другие книги, например, книга Троелсена [28]. Однако автор считает, что книга Троелсена слишком скучна и не сможет заинтересовать и увлечь никогда не программировавших студентов. Впрочем, от книги Стиллмена и Грин планируется отходить, тем более что следующее издание посвящено уже не устаревшей технологии Windows Forms, а новой — WPF.

4. Программирование на языке C#, продвинутый курс по книге Рихтера [27]. Это сложный курс, который включает в себя многое, помимо C#: основы CLR, устройство виртуальной машины, сборку мусора. В курсе рассматривается большая часть книги, за исключением многопоточного программирования. Чрезвычайно высокий уровень освещения материала и его детальная проработка не оставляют времени на изучение этих средств. Данный модуль изучают группы с хорошим уровнем подготовки или группы, предварительно прошедшие через вводный курс по книге Стиллмена и Грин [13].
5. Программирование на языках C и C++. Язык C изучается по книге Кернигана и Ритчи [8] за два опроса. Эта книга написана простым и ясным языком, содержит большое количество листингов программ. Разобрав первые шесть глав (менее 200 страниц), студент сможет начать писать и понимать относительно продвинутый код. Язык программирования C++ преподается по книге Праты [26]. Данный учебник широко известен в мировом сообществе и считается одной из лучших книг для обучения этому языку программирования. Автору предлагались и другие варианты для преподавания C++, однако все они имеют серьезные недостатки. Например, книгу Страуструпа критикуют за то, что в первых главах автор сразу же начинает с использования контейнера vector, и в целом книге не хватает последовательного изложения материала. В книге Праты этот недостаток отсутствует, материал превосходно систематизирован и снабжен большим количеством листингов. Другим достоинством является описание некоторых новых возможностей стандарта C++11. Кроме того, в этой книге есть упражнения для самостоятельной работы. К сожалению, их нельзя использовать в аудиторном занятии, так как на них приведены ответы. Однако эти упражнения могут помочь и помогают мотивированным студентам самостоятельно изучать материал.
   Наконец, книга не привязана к какой-то конкретной операционной системе и конкретному компилятору, в ней описывается процесс сборки проектов как в Windows (компиляторы Microsoft и Borland), так и в Ubuntu Linux (g++), OS X (g++) и других. Прата утверждает [26], что все примеры протестированы в данных условиях.
6. Короткий модуль по паттернам проектирования, два опроса по книге Гаммы и др. [14] с последующими докладами. Эта книга требует уверенного владения объектным инструментарием C++, поэтому автор ставит его после модуля по языку C++. Не все студенты могут работать по данной книге, поэтому автор рекомендует также и книгу Фримена и Фримен [30]. Она проще, написана более ясным языком, однако предназначена для языка программирования Java. Другой недостаток — стиль изложения не подходит для студентов второго курса СПбГУ в силу своей «нескучности». Кроме того, в плане содержания и структуры книга Гаммы является более глубокой.

7. Программирование для ОС Linux. Изучается по трем книгам [9, 23, 25]. В зависимости от уровня подготовки студентов, данный модуль может даваться или нет. В данном модуле преподаются как основы системного программирования в операционной системе Linux, так и соответствующий инструментарий. В первую очередь, это использование отладчика Valgrind для поиска гонок и утечек памяти на живых примерах. Кроме того, рассматриваются и другие полезные инструменты: strace, ltrace, make, patch, doxygen, cflow. Из тем системного программирования можно отметить работу с файлами (рассматривается как высокоуровневый интерфейс стандартной библиотеки C, так и низкоуровневый интерфейс системных вызовов), управление памятью, обзорно рассматривается ядро Linux, методы отладки в Linux. Наконец, вкратце затрагивается многопоточное программирование и командный интерпретатор bash.

5.3. Сложности при составлении опросов

Прежде всего, необходимо отметить серьезную ограниченность в выборе литературы. По правилам, автор не может использовать литературу на английском языке, более того, если бы автор мог это делать, не все студенты справились бы с такой нагрузкой. На первом-втором курсе прочитать и осмыслить такой объем литературы — это серьезный труд, требующий высокого уровня подготовки в языке.

Например, для курса по программированию под ОС Linux автор использует три книги (выше приводилось подробное описание курсов). По мнению автора, это были лучшие книги на русском языке, как минимум на момент 2011 года. Позже была обнаружена книга Майкла Керриска[3] [49], которая с успехом может заменить две из трех, однако данная книга не переведена на русский язык.

Ситуация с книгами на русском языке в целом очень тяжелая. Некоторые научные направления существуют уже десятки лет, но на русском языке учебники до сих пор отсутствуют. В этих условиях роль энтузиастов и сообщества существенно возрастает. Например, когда автор вел курс по рекомендательным системам, одной из используемых книг [16] была «Введение в информационный поиск» [5], которую помогал переводить Яндекс и сообщество специалистов по информационному поиску [3]. Другим примером может служить целая линейка книг по функциональному программированию, в том числе и «Структура и интерпретация компьютерных программ» [7]. Их тоже переводило сообщество [17, 31, 33]. Наконец, из свежих примеров можно упомянуть книгу «Цифровая схемотехника и архитектура компьютера» [12], которую также перевело сообщество [4].

К сожалению, сообщество редко переводит книги для программистов, которые необходимы в этом курсе. Кроме того, ресурсы сообщества также ограничены, поэтому покрытие даже научной литературы всё равно недостаточное.

Сложности с книгами не новы, например, в статье 2004 года [19] А.А. Терехов описывает состояние книжного рынка того времени. Отмечается перекос в ассортименте имеющейся переведенной литературы, наблюдается преобладание «массовых» книг наподобие «VB 3.0» над фундаментальными и сопоставимыми по значимости. Автор перечисляет десять важных книг, не представленных на российском рынке на тот момент. Десять лет спустя, нашлись следы перевода пяти книг из десяти.

Другая проблема при составлении модулей — низкое качество литературы, доступной на русском языке. Практически в каждой книге есть существенные дефекты перево-

[3] Майкл Керриск — нынешний сопровождающий (maintainer) системы документации man.

да. Прежде всего, необходимо отметить неудачную грамматическую структуру, не способствующую пониманию материала. К более серьезным дефектам относятся неверный перевод и пропущенные куски текста. Приведем примеры:

1. В книге [27], в главе 4 «Основы типов», в разделе «Как разные компоненты взаимодействуют во время выполнения» в тексте отсутствует какая-либо ссылка на рис. 4–11. В то же время, в английской версии книги присутствует эта ссылка и описание рисунка, она начинается с предложения “The next line of code in M3 calls Employee’s nonvirtual instance GetYearsEmployed method”. То есть переводчиками пропущен абзац из более чем 100 слов.
2. В главе 5 [27] на с. 180 написано «Пока компилятор позволяет пренебрегать явным приведением динамического типа к другому типу данных, среда CLR на этапе выполнения проверяет правильность приведения с целью обеспечения безопасности типов». Это очень путаное предложение, смысл которого довольно трудно понять. Однако если знать оборот английского языка “While X, but Y”, то всё становится на свои места. И действительно, в оригинале имеется в виду именно это: “While the compiler allows you to omit the explicit cast when casting from dynamic to some other type, the CLR will validate the cast at runtime to ensure that type safety is maintained”. Таким образом, более правильным переводом является: «Хотя компилятор позволяет не использовать явное приведение, CLR выполнит проверку приведения во время исполнения, для того чтобы обеспечить безопасность типов».
3. Бывает и просто неверный перевод. Например, в главе 12 на с. 303: «T — это имя переменной, которое применяется в исходном тексте во всех местах, где используется соответствующий тип данных». Оригинал же звучит так: “T is a variable name that can be used in source code anywhere a data type can be used”. Правильным переводом был бы следующий: «T — это имя, которое может быть использовано в коде программы в любом месте, где используется тип данных».
4. В оригинале книги Праты присутствует ошибка в программном коде [71]. При переводе ситуация усугубилась: код остался прежним, и была добавлена опечатка, серьезно запутывающая и без того сложную ситуацию. В данном пункте речь идет о странице 386, где внизу упоминается вызов функции call().
5. Доходит до курьезов, например в книге [13] в главе 11 на с. 522 рассматривается построение игры “whack-a-mole”. Это известная игра, служившая примером во многих книгах, посвященных разработке приложений для платформы Windows. В этой книге название игры переведено как «убей моль», хотя mole это вовсе не моль, а крот, да и “whack” было бы правильнее перевести как «долбани».
6. Гораздо серьезнее пострадала книга по программированию для ОС Linux [25]. Например, в пятой главе на с. 177, плашка начинается со слов «В приведенном примере», однако в разделе никакого примера нет. Обратившись к оригиналу, можно найти пропавший пример. Также становится ясно, что несоответствие довольно значительное: исчезла часть нумерации (например, в оригинале присутствует раздел 5.3.7.1), и было выброшено очень много кода с примерами.

К сожалению, при выборе книги невозможно досконально ознакомиться с содержанием, чтобы избежать подобных проблем. Многое зависит от фактора удачи.

В курсе нельзя использовать книги для подготовки к сертификационному экзамену или подобные материалы. Несмотря на то, что качество делает их привлекательным вариантом, они стоят очень дорого, не переведены и не адаптированы к формату занятий в университете. Другой вариант — использовать в преподавании дампы (базы вопросов)

реальных сертификационных экзаменов, выложенные в интернет. К сожалению, он попросту противозаконен, так как эти базы являются чужой интеллектуальной собственностью, обычно попавшей в интернет в результате противоправных действий.

Еще одна проблема — с каждым годом у студентов копится база вопросов. Частичным решением данной проблемы может служить ограничение времени доступа к вопросам. Для этого автор использует проектор при выдаче опроса и после опроса забирает листочки с текстами заданий.

Есть и другое решение — если уровень группы внезапно и необоснованно подскочил, можно перемешать вопросы из разных вариантов опросов. В любом случае, даже имея всю базу вопросов — от 30 до 50 штук, студенту необходимо к ним готовится, читать и разбираться.

Однако в перспективе со студенческим коллективным разумом бороться таким методом бесполезно. Единственный способ — создавать новые вопросы по старой книге или перейти на новую.

В последние годы участились просьбы разрешить вести конспект в электронном виде, печатать его и использовать на опросе. Со стороны студентов заинтересованность понятна — с каждым годом они приходят со все более развитым навыком печати. Им проще набирать на клавиатуре, нежели писать от руки. Это мировой тренд, Россия тут не исключение. Например, в Финляндии собрались убирать из школьной программы обучение рукописному письму, а взамен ввести обучение печати [43]. Автор данной статьи противится практике набранных конспектов по вполне понятным причинам. Когда студент конспектирует, он запоминает и гарантированно выполняет какую-то работу. В случае с распечатанным документом всё не так очевидно. Например, можно скопировать материал из электронной книги.

6. РЕЗУЛЬТАТЫ

Основным результатом данного подхода является повышенная мотивация к обучению: практика показала, что студентам и слабых и, в особенности, сильных групп нравится такая система. Она предлагает им новый интересный материал и новую методику. Также можно отметить повышение посещаемости занятий и внимания к предмету. В пользу того, что студентам нравится данная система, свидетельствуют переводы в подгруппу автора, при том, что отсутствуют переводы из нее, хотя об этой возможности студентам сообщается.

7. ПОДОБНЫЕ ПРАКТИКИ, ОБЗОР

Домашнее чтение и его контроль широко используются в университетах по всему миру при проведении как лекционных курсов, так и практик. На этот метод преподавания можно взглянуть с двух позиций: научной и практической. Научный взгляд заключается в рассмотрении научных работ, исследующих использование домашнего чтения в преподавании. В этих работах разбираются примеры преподавания различных профессиональных дисциплин, не обязательно программирования, с использованием домашнего чтения и эксперименты с его контролем, не обязательно опросами. Описываются экспериментальные методики, не всегда использующиеся в преподавании регулярно. Зато в этих работах детально освещаются сами методики, способы их внедрения и анализа.

Практический взгляд состоит в указании примеров курсов по тематике программирования, взятые из Computer Science Curriculum 2013 [68], где используются опросы и домашнее чтение. В данном случае рассматривается обучение именно по сходной тематике, и эти примеры не экспериментальны, а представляют собой обычные университетские курсы.

7.1. Обзор научных публикаций

Использование домашнего чтения является популярным предметом исследований по тематике обучения. В данной части приводятся примеры подобных исследований.

В работе [44] домашнее чтение используется для вовлечения студентов в научную деятельность. В работе описывается процесс ведения небольших групп студентов по 3–5 человек, читающих научную литературу по специальности. Студентам каждой группы на неделю выдаются для ознакомления статьи. Группа студентов распределяет статьи между собой. Каждый студент читает свою статью и делает краткий одностраничный конспект, который распространяется внутри группы непосредственно перед обсуждением. Внутригрупповые обсуждения происходят на следующем занятии; после обсуждений группы делают доклады перед всей аудиторией и выбираются статьи на следующую неделю. В этом эксперименте преподаватель заранее готовил вопросы, на которые необходимо было ответить, после прочтения материала.

В конце данной работы есть очень детальное исследование эффективности предложенной методики. Исследование выполнено в виде анкетирования, содержащего различные вопросы на тему восприятия данного подхода. Опрашивались 74 студента, затем после применения методов математической статистики делался вывод, что 73% студентов оценили новый процесс как полезный.

В работе [47] разбирается вопрос, каким образом следует содействовать студентам в чтении специализированной литературы. Выяснилось, что недостаточно дать простое напутствие «иди и читай». В статье предлагаются четыре возможных вида содействия:

1) четкое указание цели: предмет чтения должен согласовываться с целями курса и критериями оценивания;
2) указание стартовой точки чтения: ключевые слова для поиска, ресурсы для поиска, примеры литературы;
3) явные инструкции и критерии: рассказать общий план статьи и указать, на какие части текста обратить внимание;
4) коллаборативное чтение и обсуждение литературы.

Далее описывается практическая проверка эффективности данных видов содействия. Для проверки рассматривается опыт курса, включавшего в себя домашнее чтение и обсуждение. Итоги оцениваются с помощью анкетирования, содержащего отзывы. В другой работе [62] исследуется, как студенты читают: какие есть проблемы чтения, какие существуют практики чтения, самостоятельно ли студенты к ним пришли или с подсказки лектора, каков уровень интереса к практикам чтения. Проводилось исследование популярности различных типов конспектов. Данные вопросы исследуются у студентов, обучающихся по трем специальностям: история, психология и биология. Исследование также проводилось с помощью анкетирования с использованием статистических методов.

Наконец, в работе [58] отмечается, что существует разница в понимании преподавателей и студентов фразы «читать статью». Поэтому появляется проблема поверхностно-

го чтения, чтения слов без понимания, которая ведет к тому, что на семинарах работает в основном преподаватель и семинар превращается в дополнительную лекцию. Таким образом, студенты становятся зависимы от преподавателя и не могут самостоятельно приобретать знания. Для решения проблемы авторы переработали курс (лекции и семинары) и сравнили его с предыдущей версией. В новой методике они создали набор групп, в каждой из которых студенты встречались и обсуждали прочитанное. Кроме того, была интенсивно задействована информационная система Blackboard [36] — для студентов были созданы группы для обсуждения. Подробное описание подхода можно найти в оригинальной статье.

В рассмотренных выше работах упор делается не только на чтение, но и на коллаборативное обсуждение — похожий механизм включается и у автора в сильных группах.

7.2. Примеры курсов

Опросы по домашнему чтению как метод контроля активно используется в университетах США (там он называется quiz). Для рассмотрения практики использования в дисциплинах, связанных с программированием, обратимся к Computer Science Curricula 2013 [68]. В этом документе есть раздел Appendix C, «примеры программ курсов», перечисляющий описания программ в различных университетах США и по всему миру. Там можно найти довольно много свидетельств об использовании метода домашнего чтения и опросов. Данный источник показывает, что метод используется как на младших, так и на старших курсах в вузах самого различного уровня. Примерами могут служить следующие курсы: CS 256 Algorithm Design and Analysis, Williams College; eScience, University of North Carolina at Charlotte; COSC/MATH 201: Modeling and Simulation for the Sciences, Wofford College; MAT 267: Discrete Mathematics, Union County College; Human-Computer Interaction, Stanford University; Computer Systems Security (CS-475), Lewis-Clark State College; Introduction to Artificial Intelligence, Case Western Reserve University; CS 144: Introduction to Computer Networking, Stanford University и другие.

Более того, в некоторых из вышеупомянутых курсов используются автоматизированные тестовые системы: например, CS 256 (используется moodle [54]) и CS 144 (используется своя online система). К сожалению, подход домашнего чтения, описанный в данной статье, полной автоматизации не поддается, так как не использует тесты, а предполагает открытые ответы на вопросы.

8. ЗАКЛЮЧЕНИЕ И ДАЛЬНЕЙШИЕ ПЛАНЫ

В данной статье было рассказано об опыте модернизации практикумов по программированию, был описан подход к преподаванию, используемый автором. В начале статьи были выделены четыре важных темы, которые следует преподавать: алгоритмические и языковые навыки программирования, документация и сообщество, а также технологические практики. Был описан подход к преподаванию этих тем в рамках курсов автора данной статьи. Основной упор в статье был сделан на описании ядра этого подхода — домашнего чтения, контролируемого посредством опросов. Была описана методика их проведения и подготовка. Также были описаны сложности, с которыми автор столкнулся в своей работе, приведены возможные методы решения некоторых проблем. Был представлен обзор подобных методов преподавания: перечислены действующие курсы в различных мировых вузах, приведены ссылки на их подробное описание в Computer

Science Curricula 2013. Кроме того, были разобраны некоторые исследовательские работы, изучавшие применение домашнего чтения и опросов.

Дальнейшие направления исследований:

- Анализ материалов вступительных анкет. Накопился солидный объем данных, который можно использовать для изучения динамики знаний абитуриентов. Можно оценить, какие технологии преподаются в школе, а какие технологии изучаются самостоятельно и на кружках программирования. Уровень теоретической подготовки и охваченные темы также поддаются оценке.
- Анализ успеваемости по накопившимся результатам опросов. В этих результатах присутствует относительная оценка каждого опроса из модуля для каждого студента. Данное исследование может показать, как быстро студенты могут научиться писать опросы, какова динамика их успеваемости, какие темы сложны и так далее.
- Оценка эффективности метода. Субъективно сейчас можно сказать, что методика приносит положительные результаты. Однако на это можно возразить множеством аргументов: у автора обычно более сильная группа, отсутствует строгое сравнение с использованием метрик, отсутствует исследование динамики и т. д. Ответить на все эти возражения можно только с помощью обстоятельного исследования, подобного тем, которые были описаны в публикациях из обзорной части данной работы.
- Исследование применимости разработанных опросов для создания сертификационного экзамена. Накоплен большой объем материала, имеется солидный опыт проведения и составления опросов, есть понимание моментов, которые трудно усваиваются обучающимися. В перспективе эти наработки можно вывести за рамки учебного курса в университете.

От сильных и мотивированных студентов можно получить полезную обратную связь. Если им интересно, находятся те, кто будут активно помогать в построении курсов — критикой, советом книг, материалов и задач. Интересующиеся самостоятельно знакомятся с довольно большим объемом материала, выходящего за рамки курса. Практика показала, что каждый год студенты находят что-то новое, чего автор не видел ранее. Кроме того, технологии не стоят на месте, а одному трудно держать их все в поле зрения. Наконец, некоторые сильные студенты могут взять часть обязанностей преподавателя на себя: они могут объяснять материал своим более слабым однокурсникам. Чтобы запустить все эти процессы, необходимы две вещи: подготовленная группа и поддержка достаточно высокого уровня преподавания, чтобы сильные студенты не теряли мотивацию к учебе.

Благодарности

Автор выражает благодарность Смирнову Кириллу за помощь в работе над данным текстом и А. Л. Тулупьеву за поддержку в этих начинаниях и знакомство с идеей опросов. Также автор выражает благодарность всем студентам, которые работали над улучшением модулей, за их критику, предложения и другую помощь.

Список литературы

1. Обобщенные классы. URL: http://pascalabc.net/wiki/index.php?title=Обобщенные_классы (дата обращения 06.10.2015).
2. Георгий Чернышев, личная страница. URL: www.math.spbu.ru/user/chernishev/ (дата обращения 06.10.2015).
3. Запись в журнале «Делаем русский глоссарий по информационному поиску». URL: http://ru-ir.livejournal.com/87096.html (дата обращения 06.10.2015).
4. Запись в журнале «Проектировщики айфонов учат это, а не хипстерщину». URL: http://panchul.livejournal.com/469643.html (дата обращения 06.10.2015).
5. *Маннинг Кристофер Д., Рагхаван Прабхакар, Шютце Хайнрих*. Введение в информационный поиск. М.: Вильямс, 2011.
6. *Тулупьев А.Л., Фильченков А.А., Тулупьева Т.В., Чернышев Г.А., Азаров А.А.* Информатика: рабочая программа учебной дисциплины. Направление 230700, бакалавриат, семестр 1–2. СПб.: СПбГУ, 2011.
7. *Абельсон Харольд, Сассман Джеральд Джей*. Структура и Интерпретация Компьютерных Программ. М.: Добросвет, КДУ, 2010.
8. *Керниган Б.В., Ричи Д.М.* Язык программирования Си. Третье издание. СПб.: Невский Диалект, 2001.
9. *Мэтью Нэйл, Стоунс Ричард*. Основы программирования в Linux. 4-е издание. СПб.: БХВ-Петербург, 2009.
10. *Хорстман К., Корнелл Г.* Java 2. Библиотека профессионала. Том 1. Основы, 8-е издание. М.: Вильямс, 2009.
11. *Хорстман К., Корнелл Г.* Java 2. Библиотека профессионала. Том 2. Тонкости программирования, 8-е издание. М.: Вильямс, 2009.
12. *Харрис Дэвид, Харрис Сара*. Цифровая схемотехника и архитектура компьютера, второе издание. Morgan Kaufmann, 2012.
13. *Стиллмен Э., Грин Д.* Изучаем C#. 2-е изд. / Включая .NET 4.0 и Visual Studio 2010. СПб.: Питер, 2012.
14. *Эрих Гамма, Ричард Хелм, Ральф Джонсон, Джон Влиссидес.* Приемы объектно-ориентированного проектирования. Паттерны проектирования СПб.: Питер, 2016.
15. *Кубенский А.А.* Создание и обработка структур данных в примерах на Java. СПб.: БХВ-Петербург, 2001.
16. Вопросы к зачету по курсу «Экспертные Системы», 2013 (осень), СПбГУ, Лектор Чернышев Г.А. (предварительная версия). URL: https://docs.google.com/document/d/1n2A6hlmk6TmQtz4XFuqU-fjihl5xGlUwnMuIjRab37A/edit (дата обращения 30.11.2015).
17. Переписка по теме перевода книги Криса Окасаки “Purely Functional Data Structures”. URL: http://lj.rossia.org/users/gogabr/109945.html (дата обращения 30.11.2015).
18. *Терехов А. Н.* Как готовить системных программистов // Компьютерные инструменты в образовании, 2001. № 3–4. С. 3–20.
19. *Терехов А. А.* Десять программистских книг, которые потрясли мир, но все еще неизвестны в России // Компьютерра, 2004. № 13.
20. *Терехов А. Н.* Вспоминая о статье «Как готовить системных программистов» // Компьютерные инструменты в образовании, 2007. № 4. С. 3–13.
21. *Окулов С.М.* Программирование в алгоритмах. М.: Бином. Лаборатория знаний, 2002.
22. *Шень А.* Программирование: теоремы и задачи. 2-е изд. М.: МЦНМО, 2004.
23. *Роббинс А.* Linux: программирование в примерах, 3-е издание. СПб.: КУДИЦ-ПРЕСС, 2008.
24. *Шилдт Г.* Java: руководство для начинающих. 4-е изд. М.: Вильямс, 2009.
25. *Фуско Дж.* Linux. Руководство программиста. СПб.: Питер, 2011.
26. *Прата Стивен*. Язык программирования C++ лекции и упражнения. 6 издание. М.: Вильямс, 2012.
27. *Рихтер Д.* CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд. СПб.: Питер, 2013.

28. *Троелсен Эндрю*. Язык программирования C# 5.0 и платформа .NET 4.5. М.: Вильямс, 2015.
29. *Кормен Т., Лейзерсон Ч., Ривест Р.Л., Штайн К.* Алгоритмы: построение и анализ. 2-е издание. М.: Вильямс, 2005.
30. *Эрик Фримен, Элизабет Фримен, Кэтти Сьерра, Берт Бейтс.* Паттерны проектирования. СПб.: Питер, 2011.
31. Переписка по теме перевода SICP. URL: http://lj.rossia.org/users/gogabr/99986.html?thread=556690#t556690 (дата обращения 06.10.2015).
32. *Георгий Чернышев*. Опрос 5: Окружение Linux. Интернационализация и локализация (вводная лекция). URL: www.math.spbu.ru/user/chernishev/papers/teaching/Introduction-Linux-5.pdf (дата обращения 06.10.2015).
33. Страница перевода книги “Benjamin C. Pierce, Types and Programming Languages”. URL: http://newstar.rinet.ru/~goga/tapl/ (дата обращения 06.10.2015).
34. Справка PascalABC.NET. URL: http://pascalabc.net/downloads/pabcnethelp/index.htm (дата обращения 06.10.2015).
35. Anonymous function. List of languages. URL: https://en.wikipedia.org/wiki/Anonymous_function#List_of_languages (дата обращения 06.10.2015).
36. Blackboard Learning System™ (Release 6) Product Overview White Paper. URL: http://library.blackboard.com/docs/r6/orientation/LSR6WP.pdf (дата обращения 06.10.2015).
37. *Bloch Joshua, Gafter Neal*. Java Puzzlers: Traps, Pitfalls, and Corner Cases. Addison-Wesley Professional, 2010.
38. *Cavus N., Ibrahim D.* Using learning objects to teach programming languages // Creating the Future 3rd FAE International Symposium, 2004. P. 303–308. URL: http://files.eric.ed.gov/fulltext/ED503158.pdf (дата обращения 06.10.2015).
39. Control Structures (CS 241), страница курса. URL: http://www.cs.du.edu/~cag/courses/CS/Regis/cs241/ (дата обращения 06.10.2015).
40. Data Structures, Containers, Collections. URL: http://wiki.freepascal.org/Data_Structures,_Containers,_Collections (дата обращения 06.10.2015).
41. *De Raadt Michael, Watson Richard, Toleman Mark*. Language Trends in Introductory Programming Courses // Proceedings of Informing Science + IT Education Conference. IS’2002. 2002. URL: http://proceedings.informingscience.org/IS2002Proceedings/papers/deRaa136Langu.pdf (дата обращения 06.10.2015).
42. FPCUnit. URL: http://wiki.freepascal.org/fpcunit (дата обращения 06.10.2015).
43. Finland ditches handwriting for typing lessons in school curriculums. URL: http://www.itproportal.com/2014/12/01/finland-ditches-handwriting-typing-school-curriculums/ (дата обращения 06.10.2015).
44. *Finlay Sara-Jane, Faulkner Guy*. Tête à tête: Reading groups and peer learning // Active Learning in Higher Education, 2005. № 6. P. 32–45.
45. Generics. URL: http://wiki.freepascal.org/Generics (дата обращения 06.10.2015).
46. *Haden P., Mann S.* The Trouble with Teaching Programming // Proceedings of the NACCQ, 2003. P. 63–70. URL: http://www.citrenz.ac.nz/conferences/2003/papers/63.pdf (дата обращения 06.10.2015).
47. Helping university students to ‘read’ scholarly journal articles: the benefits of a structured and collaborative approach / Yuka Fujimoto, Pauline Hagel, Paul Turner et al // Journal of University Teaching & Learning Practice, 2011. Vol. 8, № 3. URL: http://ro.uow.edu.au/jutlp/vol8/iss3/6 (дата обращения 06.10.2015).
48. *Jabłonowski Janusz*. Some remarks on teaching of programming // Proceedings of the 5th international conference on Computer systems and technologies (CompSysTech ’04). New York, NY, USA : ACM, 2004. P. IV.10–1–IV.10–6. URL: http://ecet.ecs.uni-ruse.bg/cst04/Docs/sIV/410.pdf (дата обращения 06.10.2015).
49. *Kerrisk Michael*. The Linux Programming Interface: A Linux and UNIX System Programming Handbook. No Starch Press, 2010.
50. LazSVNPkg. URL: http://wiki.freepascal.org/LazSVNPkg (дата обращения 06.10.2015).
51. Lee University Catalog 2015-2016. Каталог курсов Университета Ли, курс CSCI 260 — Introduction To Pascal Programming. URL: http://catalog.leeuniversity.edu/preview_course_nopop.php?catoid=8&

coid=18115 (дата обращения 06.10.2015).
52. Louisiana State University-Eunice Archived Academic Catalog 2014–2015. Курс CSC 1248 — Introduction to Pascal Programming. URL: http://catalog.lsue.edu/preview_course_nopop.php?catoid=6&coid=4228 (дата обращения 06.10.2015).
53. Michaël Van Canneyt. Reference guide for Free Pascal, version 2.6.4. Document version 2.6. March 2014. Chapter 15. Operator overloading. URL: http://www.freepascal.org/docs-html/ref/refch15.html (дата обращения 06.10.2015).
54. Moodle Manuals. URL: https://docs.moodle.org/29/en/Moodle_manuals (дата обращения 06.10.2015).
55. Nevest “freepascal” questions — Stackoverflow. URL: http://stackoverflow.com/questions/tagged/freepascal (дата обращения 06.10.2015).
56. Nevest “java” questions — stackoverflow. URL: http://stackoverflow.com/questions/tagged/java (дата обращения 06.10.2015).
57. PascalABC.NET. Что нового. URL: http://pascalabc.net/en/chto-novogo (дата обращения 06.10.2015).
58. *Railton Diane, Watson Paul*. Teaching autonomy ‘Reading groups’ and the development of autonomous learning practices // Active Learning in Higher Education, 2005. Vol. 6, № 3. P. 182–193.
59. Reading Groups — Interdisciplinary Doctoral Program at the Humanities — Princeton University. URL: https://www.princeton.edu/ihum/reading-groups/ (дата обращения 06.10.2015).
60. Reading Groups. Department of philosophy. URL: http://philosophy.columbia.edu/content/reading-groups (дата обращения 06.10.2015).
61. Reading Groups. English Graduate Student Organization @ UW. URL: https://students.washington.edu/enggso/reading-working-groups/ (дата обращения 06.10.2015).
62. *Mar Mateos, Ruth Villalón, Maria Jose De Dios, Elena Martín.* Reading and writing tasks on different university degree courses: what do the students say they do? // Studies in Higher Education, 2007. Vol. 32, № 4.
63. *Severance Charles*. The Art of Teaching Computer Science: Niklaus Wirth // Computer, 2012. Vol. 45, № 7. P. 8–10. URL: http://dx.doi.org/10.1109/MC.2012.245 (дата обращения 06.10.2015).
64. *Sierra Kathy, Bates Bert*. SCJP Sun® Certified Programmer for Java™ 6 Study Guide Exam (310-065). McGraw-Hill, 2008.
65. Stackoverflow. How can I use anonymous methods in Free Pascal? URL: http://stackoverflow.com/questions/7799077/how-can-i-use-anonymous-methods-in-free-pascal (дата обращения 06.10.2015).
66. Templates. URL: http://wiki.freepascal.org/Templates (дата обращения 06.10.2015).
67. Texas A&M University Undergraduate Catalog. Архив курсов Computer Science (CPSC) за 2008 год. Курс CPSC 110. URL: http://regcatalogarchives.tamu.edu/07-08_UG_Catalog/course_descriptions/cpsc.htm (дата обращения 06.10.2015).
68. The Joint Task Force on Computing Curricula Association for Computing Machinery (ACM) IEEE Computer Society. Computer Science Curricula 2013. Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. December 20, 2013. URL: https://www.acm.org/education/CS2013-final-report.pdf (дата обращения 06.10.2015).
69. The Texas General Education Core Web Center. URL: http://statecore.its.txstate.edu/cat.pl?cat=90 (дата обращения 06.10.2015).
70. University of Houston-Clear Lake, Undergraduate Catalog 2014-2015, Roster of Courses, CSCI — Computer Science. Курс CSCI 1318 Pascal Programming I. URL: http://uhcl.smartcatalogiq.com/en/2014-2015 (дата обращения 06.10.2015).
71. Using new to create new storage and returning a reference. URL: http://stackoverflow.com/questions/27647239/using-new-to-create-new-storage-and-returning-a-reference (дата обращения30.11.2015).
72. *Willis Cheryl L., Miertschin Susan L.* Mind maps as active learning tools // J. Comput. Sci. Coll., 2006. Vol. 21, №. 4. P. 266–272.

Приложение А: задачи с тестирования

Задание 1:

```
function mystery (i: integer):integer;
begin
  writeln('');
  repeat
    write(i mod 2);
    i := i div 2;
  until (i = 0);
end;

begin
  mystery(26);
end.
```

- На каком языке написана эта программа?
- Перепишите эту программу, заменив этот цикл на другие операторы цикла, вам известные.
- Какой результат будет напечатан при ее запуске?
- Что она делает, в чем состоит ее цель?

Задание 2:* Напишите на любом известном вам языке программирования функцию, решающую следующую задачу. Дан массив, все числа кроме одного встречаются два раза. Найти это число, используя объем памяти, не зависящий от длины массива, за один проход.

*Задание 3**:* Даны бусы (крепящиеся обвязкой), представленные в виде массива длины N. Каждый элемент массива – целое число обозначающее цвет бусинки. На вход дана пара бус (два массива), надо проверить, не одинаковые ли они, учитывая возможность того, что они могут быть перекручены. Например, бусы [1,0,0] и [0,1,0] одинаковые. Аналогично с [1, 2, 3, 4, 5] и [5, 4, 3, 2, 1].

Напишите функцию, принимающую на вход два массива длины N и определяющую, являются ли бусы, представленные массивами, одинаковыми.

Задание 4:* Напишите программу, решающую следующую задачу. Дан односвязный список целочисленных элементов (заданный с использованием указателей), из него необходимо удалить все элементы большие 25. Программу можно писать на любом языке программирования.

Задание 5:* Реализуйте объект «дробь» (объект вида X/Y, где X и Y целые числа) в соответствии с методологией объектно-ориентированного программирования. Для него должны быть реализованы: конструктор, деструктор, операция сложения, вычитания, умножения и упрощения.

Вопрос 1: В чем суть поразрядной сортировки?

Вопрос 2: Какие структуры данных Вы знаете? Опишите детали механизма функционирования, их достоинства и недостатки? Приведите примеры приложений, где было бы удобно их использовать.

Вопрос 3: Что такое двоичное дополнение и где оно используется?

Вопрос 4: Что такое O-нотация ($O(N)$, $O(logN)$ и т. д.)? Приведите примеры использования.

Вопрос 5:* Что такое полиморфизм и для чего используется? Приведите примеры использования.

TEACHING INTRODUCTORY PROGRAMMING COURSE FOR COMPUTER PROGRAMMING MAJORS AT THE DEPARTMENT OF MATHEMATICS AND MECHANICS OF SAINT-PETERSBURG UNIVERSITY

Chernishev G. A.

Abstract

In this paper we describe the experience of teaching introductory programming course for computer programming majors at the department of mathematics and mechanics of Saint-Petersburg University. We describe the course program, studied topics, a typical seminar session, assessment methods, developed units and books used to built them. The core of this approach — the home reading with control quizzes is presented in detail. We also describe the entrance test which is used to tailor course programs for each group. Finally, a short survey of similar approaches featuring home reading and quizzes is provided. This survey includes both research studies and existing course programs from the Computer Science Curriculum 2013.

Keywords: *Software Development Fundamentals, SDF, programming, quiz, lab session, course program, faculty of mathematics and mechanics, Saint-Petersburg University.*

Чернышев Георгий Алексеевич,
ассистент кафедры
Информационно-Аналитических Систем
СПбГУ,
chernishev@gmail.com