ПРОБЛЕМЫ ОСВОЕНИЯ ОБЪЕКТНО ОРИЕНТИРОВАННЫХ ТЕХНОЛОГИЙ ПРОГРАММИРОВАНИЯ НА РАННИХ ЭТАПАХ ИЗУЧЕНИЯ ИНФОРМАТИКИ В ПЕДАГОГИЧЕСКОМ ВУЗЕ

А. А. Бейда

Белорусский государственный педагогический университет имени Максима Танка
Минск, Беларусь
E-mail: anatoliy_beida@mail.ru

Обосновывается необходимость в ранней пропедевтике объектно ориентированных технологий программирования при изучении информатики на профильных специальностях в педагогическом вузе. Обсуждаются сопутствующие проблемы и пути их решения.

Ключевые слова: алгоритмизация, класс, методика, объектно ориентированное программирование, педагогический вуз, пропедевтика, технология.

Объектно ориентированные технологии – основа современного программирования. Они реализованы во многих языках высокого уровня. Однако изучение объектно ориентированных технологий программирования не является задачей общего среднего образования и поэтому подавляющее большинство выпускников средних школ с этими технологиями не знакомо. С другой стороны, школьная программа углубленного уровня по информатике содержит обширный раздел алгоритмизации и программирования, ориентированный на процедурные методы решения задач алгоритмизации.

Очевидно, что педагогический вуз, осуществляя подготовку специалистов по информатике, должен сформировать у них базовые представления и основные практические навыки современных технологий программирования. При этом начальные знания студентов первого курса в области алгоритмизации и программирования могут сильно различаться – от базового до углубленного уровня. Естественно, возникают следующие стартовые проблемы: а) выравнивание начальной подготовки; б) ранняя пропедевтика объектного метода в современной информатике. Примечательно, что, правильно решая вторую проблему, мы достигаем и решения первой. Кроме того, мы получаем мощный толчок в повышении фундаментальности и научности всего цикла информационных дисциплин. Об этом и пойдет речь ниже.

Технология объектно ориентированного программирования (ООП) – это не просто набор новых методов в программировании – это новый уровень мышления и восприятия (абстрагирования) окружающего мира. А коль скоро так, то даже при очень хорошей подготовке по алгоритмике на базе процедурной техники требуется достаточно много усилий по изменению (совершенствованию) сложившейся парадигмы мышления. Здесь-то и возникает область пересечения для вчерашних школьников, имеющих разную подготовку по

информатике. Объектные технологии предоставляют им приблизительно равные стартовые условия. Стандартные офисные приложения, какими являются Word и Excel, привычны для школьника. Тем более интересно ему открыть в них более глубокий пласт, взглянув на них с точки зрения разработчика. Так хорошо знакомый материал становится опорой и источником большого количества практических задач и одновременно целью для качественно иного изучения (см. в этой связи [1–4]). Объектные модели этих приложений демонстрируют высокий профессиональный уровень абстрагирования в конкретных предметных областях, а встроенные средства VBA (Visual Basic for Applications) предоставляют богатые возможности соавторства на путях автоматизации утилитарных задач практического применения соответствующих приложений. Такой прием повышает статус первокурсника в его собственных глазах, служит источником творческого вдохновения. Здесь скрываются также широкие возможности для дифференцированного обучения, ибо спектр задач автоматизации офисных приложений не ограничен [5–6].

Какие же проблемы возникают у первокурсника при первом соприкосновении с технологией ООП? Прежде всего – *психологические*. Основные школьные предметы (особенно математика) способствуют закреплению у него преимущественно процедурного (директивного) способа мышления при решении задач. Как правило, в каждой задаче задается именно такое количество условий, какого необходимо и достаточно для ее решения. Лишнее условие или такой набор условий, в котором можно выбрать две несовпадающие комбинации данных, разными путями приводящие к решению задачи, может повергнуть ученика в легкий шок. У него, как правило, сформировано стерильное, рафинированное мышление, удаляющее его от реальной действительности в мир некого стандартизованного формализма. Это не лучший способ развить интуицию, независимость суждений, определенную смелость в анализе задач и принятии решений.

Следующий круг проблем – это проблемы *мировоззренческие*. Не зря технологию ООП называют новой парадигмой программирования. Парадигма по Томасу Куну (1970 г.) – это набор теорий, стандартов и методов, которые совместно представляют собой способ организации научного знания – иными словами, способ видения мира [7, с. 26]. «Парадоксально, но стиль решения задач, воплощенный в объектно ориентированной технике, нередко используется в повседневной жизни. Тем самым новички в информатике часто способны воспринять основные идеи объектно ориентированного программирования сравнительно легко, в то время как люди, более осведомленные в информатике, зачастую становятся в тупик из-за своих представлений. К примеру, Алан Кей обнаружил, что легче обучать языку Smalltalk детей, чем профессиональных программистов» [7, с. 27]. Технология ООП возникла в результате естественного развития языков структурного программирования (Алгол, Паскаль, С), позволявших эффективно писать умеренно сложные программы. Однако дальнейшее увеличение объема программ и широкое вовлечение событийных техник управления ими привело к востребованности идей инкапсуляции (объединения данных и кода в форме объектов для защиты от вмешательства извне и неправильного использования), наследования (когда один объект приобретает основные свойства другого за счет совместного использования его кода) и полиморфизма (позволяющего использовать один интерфейс в схожих, но технически разных ситуациях). Механизмом для создания объектов являются классы (специальные шаблоны кода, инкапсулирующие данные, свойства, события и функциональность и абстрагирующиеся как единое целое в рамках некоторой математической модели). Современное программное обеспечение создается на базе мощных библиотек классов. Эти классы по сути и есть те абстракции, которые структурируют и вбирают в себя известные методы решения задач из конкретных областей знаний.

Наконец, необходимо выделить проблемы, которые лучше всего именовать как *поведенческие*. Как сформировать нужную парадигму мышления? Только в результате выработки устойчивых навыков поведения. Привыкнув к новым способам решения задач и доведя эти способы до автоматизма, студент начинает воспринимать их как часть своей сущности, как основной стереотип поведения (именно стереотип поведения, на наш взгляд, важнейшее отличительное свойство восприятия индивидуумом некоторой парадигмы). Это тот случай, когда количество постепенно перерастает в новое качество. И важнейшей предпосылкой успешного решения этих проблем (как следует из предыдущего) является ранняя пропедевтика ООП уже в курсе «Введение в информатику». Именно здесь имеются все необходимые составляющие – неограниченное количество целесообразных в практической деятельности задач, быстрый в освоении и одновременно мощный язык программирования VBA, наглядный и привычный графический интерфейс приложений Word и Excel, который можно использовать при вводе данных, выводе результатов и визуальной демонстрации решений.

На практике формирование соответствующих поведенческих стереотипов реализуется в три этапа. *Этап 1* представляет собой период ознакомления и привыкания, сравнения новых и старых идей. Здесь реализуется первое знакомство с объектными моделями офисных приложений, однако методы решения задач представляют собой продолжение обычных процедурных техник, но использующих новые объектные типы данных из объектных моделей Word и Excel. На этом этапе эффективно используется метод механической записи макроса с последующим анализом его кода на предмет определения основных объектов приложения, затронутых нашими механическими действиями. Лучшим приложением для решения такого рода задач является Word, который предоставляет массу коротких и эффектных задач по форматированию, анализу и редактированию текстов, манипулированию с фрагментами текста и т. п. Эти задачи обычно упрощаются в первом приближении с тем, чтобы позволить для их решения соответствующую механическую интерпретацию. На этом же этапе, естественно, вводятся и осваиваются основы VBA.

Этап 2 – это период первых самостоятельных проб, формирование понимания основ нового метода. Здесь рассматриваются задачи, в ходе решения которых удается вычленить некую самодостаточную абстракцию, приводящую к описанию класса (шаблона для объектов). Так создаются первые пользовательские типы, позволяющие упростить логику решения отдельных задач, гибко адаптируя к ним используемый язык программирования. Это и есть первые шаги в компьютерном моделировании. Приветствуется особо, если возникают новые задачи, в решении которых достигается существенный эффект в силу повторного использования кода уже существующего класса.

Этап 3 – это период проникновения в сущность нового метода, требующий создания достаточно развитых объектных моделей. Здесь происходит многократное использование некоторых из них для решения различных практических задач, как следствие – их развитие и совершенствование, выделение из их ряда наиболее употребимой и превращение ее в особый проект, преследующий достаточно дальние цели. Наиболее соответствующим новым задачам в этом периоде является Excel в силу его хорошей согласованности с компьютерным моделированием в математике. Следовательно, с переходом к изучению Excel объем и сложность создаваемого кода увеличиваются. Если при изучении Word доминируют этапы 1 и 2, то при изучении Excel – этап 3. Учебный проект «Плоскость», реализованный здесь, представляет собой некоторую компьютерную модель координатной плоскости и позволяет решать достаточно широкий спектр задач планиметрии.

Примечание 1. Особо необходимо отметить тот факт, что описываемые выше этапы реализуются в контексте углубленного изучения самих офисных приложений Word и Excel как изучение встроенных механизмов автоматизации при решении практических задач средствами этих приложений. При этом взгляд изнутри, глазами разработчика позволяет глубже понять их структуру и принципы работы в них. Подбор задач и методика их решения играют здесь исключительную роль, поскольку несут, в силу сказанного, двойную нагрузку.

Примечание 2. Еще раз о пропедевтике. Проект «Плоскость» воссоздается позже в курсе «Основы алгоритмизации», инструментом изучения алгоритмики в котором служит язык программирования C#. В то время как в VBA нет, например, механизмов классического наследования и мы использовали механизм включения-делегирования (построения объектов одних классов в других с целью использования их функциональности), C# – полнофункциональный, полностью типизированный объектно ориентированный язык, поддерживающий, в частности, классическое наследование, предоставляющий разнообразные и мощные механизмы для реализации полиморфизма и многое другое. Но такое развитие по спирали играет при обучении важную роль – это повторение однажды освоенных понятий, решение старых задач на новом уровне и другими средствами, расширение и развитие фундаментальных идей.

Примечание 3. Рассматривая технологию ООП (ее часто называют событийным программированием), необходимо помнить о месте классической алгоритмики в рамках этой технологии. В отличие от исторически ранних этапов развития программирования, современные программы – это далеко не алгоритмы в их чистом виде. Алгоритмы сопоставимы с методами, в которых реализуется функциональность классов. Однако и в целом вся деятельность по написанию современных программ, несомненно, является *алгоритмической*. Существуют методики организации такой деятельности, технологии сбора и структурирования необходимых данных, проектирования библиотек классов и графических интерфейсов, кодирования, тестирования и отладки. Вся эта деятельность определенным образом формализуется, и тем не менее реальный процесс написания современных программ нередко выходит за рамки всяких формализмов и по большому счету сопоставим с искусством. Аналогично обстоит дело и с идеями доказательного программирования (см., например, [5 с. 299, «Доказательное программирование»]), приведшими на практике лишь к возникновению ряда приемов, позволяющих достичь некоторой уверенности в надежности кода, но не дающих никаких гарантий. Естественно, что будущие преподаватели информатики должны быть ознакомлены с соответствующими идеями и технологиями. Несомненно, что при этом исключительное значение приобретает и ранняя пропедевтика ООП. С другой стороны, никак нельзя упускать всего комплекса приемов классической теории алгоритмов. *Она должна быть корректно встроена в новый контекст.* Объединяя эти две составляющие современного программирования, мы выходим за круг хоть и трудных, но рафинированных задач, приучаем студента мыслить в условиях некоторой неопределенности, эффективно оценивать проблемы и анализировать варианты решений. На этом пути студент учится накапливать собственные библиотеки классов, а это нередко приводит к многовариантности в решении задач. Тогда выбор наиболее эффективного решения осуществляется по большему количеству критериев (нежели временная или емкостная сложность алгоритма), возникает своеобразная «рыночная» стоимость алгоритмов. Аналогичные проблемы обсуждались, например, в [8]. Проблема, затронутая в этом примечании, делает актуальным накопление большого количества задач объединяющего характера (в смысле, высказанном выше). Трудность заключается еще и в том, что такие задачи должны иметь в пределах курса «сквозной» характер, ориентироваться на некоторую единую, последовательно накапливаемую библиотеку классов.

ЛИТЕРАТУРА

1. *Бейда, А. А.* Методика изучения курса «Введение в информатику» в условиях разноуровневой подготовки студентов / А. А. Бейда // БГПУ: Состояние, проблемы и перспективы теории и практики обучения математике, физике и информатике : материалы междунар. науч. конф. – Минск, 2002. – С. 167–169.
2. *Бейда, А. А.* Сістэмная інтэграцыя курсаў па інфарматыцы / А. А. Бейда // Весці БДПУ. – 2004. – № 2. – С. 23–26.
3. *Бейда, А. А.* Объектно ориентированные технологии в преподавании информатики / А. А. Бейда // Информатизация образования. – 2005. – № 2. – С. 15–27.
4. *Бейда, А. А.* Изучаем офис в контексте его объектной природы / А. А. Бейда, В. М. Бейда // Информатизация образования. – 2006. – № 1. – С. 10–25.
5. MS Office XP: разработка приложений / А. В. Матросов [и др.]. – СПб. : БХВ-Петербург, 2003. – 944 с.
6. *Гарнаев, А. Ю.* MS Excel 2002: разработка приложений / А. Ю. Гарнаев. – СПб. : БХВ-Петербург, 2004. – 768 с.
7. *Бадд, Т.* Объектно ориентированное программирование в действии / Т. Бадд. – СПб. : Питер, 1997. – 464 с.
8. *Бейда, А. А.* Задачи по информатике с элементами доказательности / А. А. Бейда, А. И. Павловский // Информатизация образования. – 2002. – № 2. – С. 28–45.

ПРОБЛЕМЫ ЧТЕНИЯ КУРСОВ МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ ДИСЦИПЛИН ДЛЯ НЕМАТЕМАТИЧЕСКИХ СПЕЦИАЛЬНОСТЕЙ

С. С. Белявский, А. Г. Горелик, Н. А. Широкова

Институт современных знаний имени А. М. Широкова
Минск, Беларусь
E-mail: alex_gorelik@km.ru

Рассматриваются проблемы интенсификации преподавания математических и компьютерных дисциплин для нематематических специальностей. Предлагаются пути их решения путем создания специализированных учебных курсов с использованием современных информационных технологий.

Ключевые слова: высшая математика, компьютерная графика, преподавание.

Современные образовательные технологии предполагают использование технических средств обучения и минимизации роли преподавателя в процессе обучения. Такого рода образовательные технологии заложены в дистанционное обучение, где одним из главных фигурантов является компьютер и, как правило, он включен в систему Интернет. Такой метод практически полностью исключает активную роль преподавателя из образо-