УДК 378

О МЕТОДИКЕ ПРЕПОДАВАНИЯ ПРОГРАММИРОВАНИЯ НА СТАРШИХ КУРСАХ ВУЗОВ

Д.И. Черемисинов
Ведущий научный сотрудник ОИПИ НАН Республики Беларусь
кандидат технических наук, доцент
cher@newman.bas-net.by

Д.И. Черемисинов

Окончил Томский государственный университет, кандидат технических наук, доцент. Работает в ОИПИ НАН Беларуси в должности ведущего научного сотрудника, более 10 лет преподавал в Белорусском государственном университете информатики и радиоэлектроники курс «Интерфейсы информационных систем».

Круг научных интересов: программирование, логическое проектирование и тестирование дискретных систем управления, реализация параллельных алгоритмов управления.

Аннотация. Дисциплина «Интерфейсы информационных систем» в учебном плане для подготовки специалистов по информационным технологиям нацелена на формирования объема знаний о взаимосвязи ключевых концепции и принципов программирования. В курсе отражены современные концепции и методические подходы к обучению информатике студентов старших курсов ВУЗов. Материал курса можно считать ядром структуры понятий информатики, так как он расположен в области перекрытия всех четырёх ее частей. Одной из важных задач преподавания курса является демонстрация того, что сформированные у студентов на начальных стадиях обучения представления об основных понятиях программирования слишком поверхностны.

Ключевые слова: Преподавание информатики в университетах, информационное взаимодействие, интерфейс, кривая обучения, метод наставничества.

Введение.

Снижение посещаемости занятий отмечалась как проблема для большинства учебных дисциплин, читаемых на старших курсах нашего факультета (Факультет компьютерного проектирования, БГУИР), это наблюдается и в других высших учебных заведениях [1]. Существуют различные гипотезы и мнения, почему это происходит. В этой работе сделана попытка объяснения феномена снижения посещаемости занятий на старших курсах вузах на опыте преподавания дисциплины «Интерфейсы информационных систем», целью которого является продолжение обучения инженеров системотехников программированию на четвертом курсе бакалавриата. Дисциплина читается в БГУИР с 2010 г [1].

Область знаний, связанная с информационными технологиями, принято разделять на четыре основные дисциплины – информатика (computer science), программная инженерия (software engineering),проектирование аппаратных платформ (hardware engineering) и информационные системы (information systems) [3,4]. Профессиональными объединениями разработаны рекомендации по преподаванию программирования и информатики в университетах [3,4], которым должны следовать учебные программы вузов, чтобы квалификация выпускаемых ими специалистов соответствовала международным требованиям. В рекомендациях [3] различаются объем знаний (body of knowledge) выпускников и описание конкретных учебных курсов, посредством которых выпускники овладевают этими знаниями.

В [3] предлагается строить отдельные руководства по разработке учебных планов для подготовки специалистов в каждой из пяти дисциплин. Описание объема знаний по дисциплине «программная инженерия» организовано в виде трехуровневой иерархии. На верхнем уровне

иерархии находятся области преподаваемых знаний, представляющие собой отдельные поддисциплины программной инженерии. Каждая сфера состоит из своей группы тематических модулей, которые обозначаются аббревиатурой сферы с добавлением порядкового номера. Каждый модуль состоит из тем.

В качестве одной из целей обучения в [3] рекомендуется среди прочего формировать у студентов инженерный склад ума, основанный на математическом подходе к решению проблем. Для этого в преподавании дисциплин инженерного профиля задачей обучения является формирование целостной системы взаимосвязанных понятий, составляющих предмет дисциплины [4]. Несмотря на то, что важность математической компетенции выпускника и формирования целостной системы взаимосвязанных понятий неоднократно отмечена в [2], там отсутствует тема или модуль специально посвященная формированию системы понятий программирования.

Дисциплина «Интерфейсы информационных систем» включена в учебный план для подготовки специалистов по информационным технологиям, так как ее целью является формирования объема знаний о взаимосвязи ключевых концепции и принципов программирования, хотя эта дисциплина не является темой или модулем из рекомендаций [3]. В основе метода преподавания лежит тезис о том, что в результате изучения курса в голове студента должна сформироваться некоторая целостная картина, а не пёстрая мозаика из разрозненных фактов.

Взаимодействие как главная тема программирования.

В учебных программах ведущих вузов, готовящих программистов, имеются курсы, целью которых является формирование общего взгляда на объем знаний по информационным технологиям в целом. Часто этот курс называется системным или теоретическим программированием. Например, в МГУ имеется курс «Фундаментальные аспекты системного программирования. Теория и практика разработки системного ПО и компонентов ядра ОС Linux». В этом курсе основой объединения ключевых понятий программирования является задача построения операционной системы (ОС). В БГУ на кафедре технологий программирования читается дисциплина «Системное программирование». Следует отметить, что дисциплина «Системное программирование» не везде используется для формирование общего взгляда, часто это спецкурс для подготовки системных администраторов – высших IT менеджеров. Во многих вузах основой курса «Системное программирование» является задача построения компилятора языка программирования. Эти курсы не дают студентам понимания, как работают программное обеспечение в целом, как прикладные программы используют операционную систему или как аппаратура компьютера, влияет на производительность и правильность программ приложений.

Для решения этой проблемы в университете Карнеги-Меллон разработан вводный курс по компьютерным системам, который называется «Введение в компьютерные системы» [5]. В этом курсе программирование рассматривается более широко, чем в традиционных курсах по организации процесса программирования или логическому проектированию, и охватывает темы архитектуры компьютеров, операционных систем, компиляторов и вычислительных сетей. Авторы курса считают его целью объяснить устойчивые концепции, лежащие в основе всех компьютерных систем, и показать конкретные способы, которыми эти идеи влияют на правильность, производительность, и полезность прикладных программ. Это цель близка направлению нашего курса, однако в университете Карнеги-Меллон он читается в начале обучения программированию, а наш – в конце.

В нашем курсе для БГУИР основой материала для преподавания является не задача, а понятие, именно понятие взаимодействия. Информационное взаимодействие – это воздействие объектов или агентов друг на друга, приводящее к изменению их состояния. Понятие взаимодействия является одним из центральных в кибернетике и информатике. Это понятие связано с понятием интерфейса. Слово интерфейс пришло в профессиональный язык

программистов из английского языка, где его определяют как: Interface is a shared boundary between two separate subsystems (сопряжение, линия раздела, перегородка двух подсистем) или как Interface is a point of interaction between subsystems (точка взаимодействия подсистем). Понятие взаимодействия естественно подходит для объединения как ключевых, так и самых современных идей программирования.

Разделы в совокупности знаний, на которых концентрируется этот курс – это: 1) архитектура и организация ЭВМ, 2) операционные системы, 3) распределенные вычисления. Упор в этом курсе сделан на предотвращение чрезмерного упрощения процесса программирования. В начале обучения эти темы преподаются поверхностно, чтобы сделать материал доступным для начинающих студентов. Основное внимание уделяется не разработке программы, а более простому процессу кодирования. Особенностью нашего курса является более углубленное изучение структуры программного обеспечения с целью демонстрации многогранности связей его частей (рисунок 1). Форма изучения решений, использованных в современных программных системах построена так, чтобы студенты не переоценивали свое владение навыками программирования. Уверенность в достаточности их умения в программировании скрывает необходимость в фундаментальных знаниях, отсутствие которых будут мешать им в поиске решений новых проблем, которые будут возникать в их профессиональной деятельности.

Рисунок 1. Зависимости понятия «мультиагентная система»

Материал курса можно считать ядром структуры информатики, так как он расположен в области перекрытия всех четырёх ее частей. Предполагается, что слушатели обладают опытом программирования, т.е. овладели принципом программирования. Стоит задача не учить программированию; а учить правильному подходу к разработке программ. Студенты, которые только начали заниматься профессиональным программированием, должны понять, что существует множество приемов для решения типичных задач.

Состояние набора понятий, связанных с программированием, фиксируется в глоссариях терминов (например [7]). Глоссарий [7] содержит около 12000 понятий. Очевидно, учебный курс не может использовать такое количество понятий. В преподавании курсов, формирующих математический подход к решению проблем, стремятся обойтись небольшим количеством понятий (обычно порядка 30). Глоссарий терминов, связанных с компьютерами, для пенсионеров [8] содержит 37 понятий. Для профессионального обучения в обсуждаемом курсе

выбраны 150 понятий. Довольно, часто отбор терминов в глоссарии осуществляется статистическими методами, путем анализа литературы. В глоссарии обсуждаемого курса отобраны понятия ядра информационных технологий, встречающиеся во всех четырех основных дисциплинах. Порядок обсуждения выбранных понятий формировался так, чтобы создать представление об их внутренней взаимосвязи.

Эксперименты с методом наставничества в информатике.

Можно предположить, что снижение посещаемости занятий происходит из-за стандартной практики преподавания – «мудреца на сцене», обусловленной теоретической, абстрактной природой преподаваемого материала, которая делает лекцию скучной. Предполагается, что решить проблему можно сделав лекции привлекательными, менее сухими и пассивными, чтобы студенты захотели приходить на них. Другая форма лекции, являющаяся более интерактивным способом передачи знаний и обучения, например метод наставничества, возможно, увеличит посещаемость.

Обучение методом наставничества (Peer Instruction) [9] широко использовалось в преподавании различных дисциплин, таких как философия, психология, геология и биология, и показало многообещающие результаты в обучении студентов. На Западе этот подход рассматривается как перспективная область теоретических и практических разработок, способная изменить подходы к обучению, особенно в высшем профессиональном образовании и в корпоративном секторе. В этом случае речь идёт об «управляемой» коммуникации посредством использования методик, требующих активной самостоятельной работы и взаимодействия учащихся. Подход был предложен Эриком Мазуром и его коллегами при изучении физики. Методика предполагает вовлечение студентов в активную деятельность по изучению материала дисциплины с последующим объяснением их своим сверстникам [9].

На занятии методом наставничества преподаватель включает в лекцию последовательность вопросов; задаваемых слушателям. Занятия проводятся по четко определенному протоколу преподавания. Каждый вопрос ориентирован на освещении очередного понятия, и от каждого слушателя требуется ответ в форме сигнала определенного формата. Для сигнализации о варианте ответа часто используются специальные интерактивные устройства – кликеры. Одной из важнейших предпосылок успеха метода является задание на чтение перед занятием; то есть, студенты должны заранее знакомиться с темой, затрагиваемой на занятии. Таким образом, предполагается, что класс обладает базовыми знаниями по теме, достаточными для понимания вопросов во время лекции и обсуждения ответов с другими студентами.

Тем не менее, несмотря на область знания и способ подачи материала лекции, и студенты не упускают возможность пропускать их. Сам по себе метод наставничества не увеличивает посещаемость занятий на старших курсах [2]. Важно отметить, что на младших курсах особых проблем с посещаемостью нет.

Применение методики наставничества на старших курсах наталкивается на трудности, тоже связанные со снижением посещаемости занятий. Как уже упоминалось ранее, ключевым условием успеха этой методики является необходимость самостоятельной знакомства обучаемого с темой, затрагиваемой на занятии, до начала занятия. Невозможно предположить, что студент, который пропускает занятия (не имеет мотивации к изучению материала), будет осваивать тему не в учебное время.

Очевидно, проблема не в процессе преподавания (лекторе, способе подачи материала или самом предмете), а в объекте обучения – студентах старших курсов. У этих студентов наблюдается снижение мотивации к обучению.

Кривая обучения.

О понятиях «Кривая обучения» (learning curve) и «Крутая кривая обучения» сейчас часто говорят и пишут в области компьютерных наук. Кривая обучения является графическим представлением того, как результат обучения зависит от его длительности; или зависимостью

производительности работника от опыта работы (чем больше кто-то или что-то выполняет операцию, тем лучше она выполняется). Более широко распространено понимание кривой обучения как математического описание изменения производительности работников при выполнении повторяющихся задач [10].

Кривую обучения можно рассматривать как представление зависимости умственных затрат на обучения чему-либо от времени. Умственные затраты могут быть измерены в числе изученных понятий. Именно такой подход используется в рекомендациях [3]. Число понятий, которыми должен овладеть выпускник, должно постоянно возрастать. Выражение «крутая кривая обучения» определяет кривую обучения как скорость, с которой приобретается навык, поэтому скорость обучения характеризует как способности обучаемого, так и трудность предмета. Число понятий можно считать оценкой трудности предмета, так как это число растет в процессе обучения, крутизна кривой обучения должна уменьшаться. Для демонстрации процесса изучения языка [11] предложена двухмодальная кривая обучения – bipolar learning graph (рисунок 2).

У студентов старших курсов велика уверенность, что обучение специальности уже закончилось – первый пик на кривой обучения на рис. 2. Это уверенность поддерживается тем, что большинство из них уже работают в организациях отрасли производства программного обеспечения. К моменту окончания обучения, по специальности или в смежных областях работают почти 100% обучающихся. Кроме того, они преодолели крутизну первоначального обучения и не сомневаются в том, что специальность программирования легкодоступна (реклама курсов по программированию уверяет, что почти любого грамотного человека можно сделать программистом за пару месяцев). Серьезные знания фундаментальных наук (математики, логики, физики) не требуются на должностях, которые им предложили их работодатели. Они убеждены в том, что обучение происходит быстро, а знания состоят из отдельных фактов. Большинство студентов, особенно слабых, сильно недооценивают время, необходимое для освоения профессии, и считают, что одной хорошей книги достаточно. Они пытаются запоминать отдельные факты, а не считают важным понимание взаимосвязи понятий.

Сомнение в достаточности знаний (спад уверенности после пика на рисунке 2) в дальнейшем по мере профессионального роста появится. Продвижение в области профессиональной деятельности связано с усложнением работы. Чем выше должность (и зарплата) тем сложнее и дороже работа, которую выполняет работник. Работник повышается в должности до тех пор, пока не займёт место, на котором он окажется не в состоянии справиться со своими обязанностями, то есть окажется некомпетентным (принцип Питера [12]). В области информационных технологий повторяемость одинаковых рабочих заданий невысока. Обнаружение собственной некомпетентности происходит не в конце профессиональной карьеры как в традиционных отраслях, а в ее начале. Не приобретая новых знаний, сотрудник рискует «застрять» в должности до тех пор, пока не покинет организацию (то есть не уволится, не умрёт или не выйдет на пенсию). Проблема в том, что мотивация к обучению возникает после окончания обучения в университете.

Курс «Интерфейсы информационных систем» эффективен на стадии обучения, соответствующей промежуточному минимуму на рисунке 2, когда появляется мотивация к повышению уровня образованности вследствие ощущения недостаточности знаний. В процессе обучения в университете у студента осознание недостаточности знаний само по себе не может возникнуть. Следовательно, задачей преподавателя на этом этапе является демонстрация того, что имеющиеся у обучаемых представления об основных понятиях программирования слишком поверхностны.

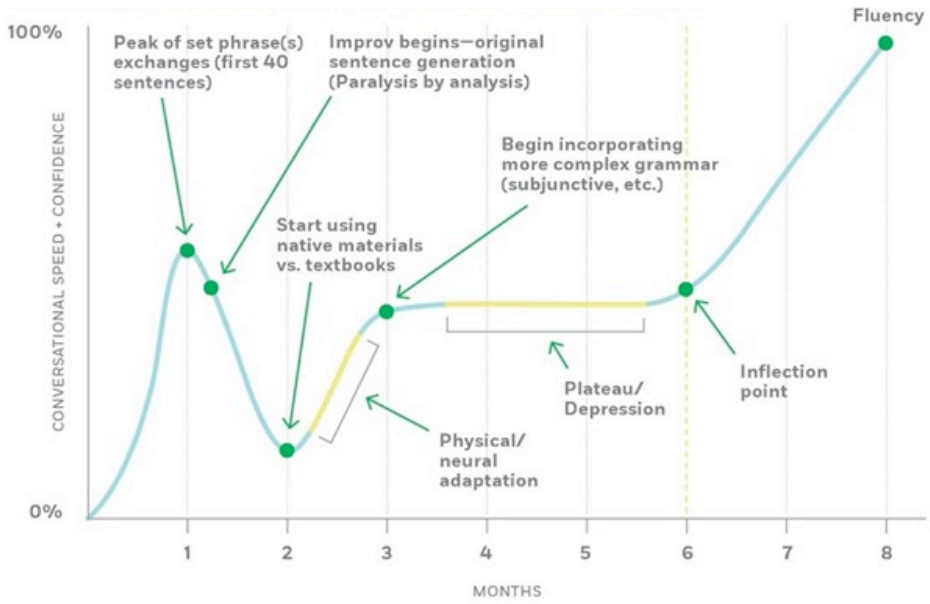

Рисунок 2. Зависимость уверенности в овладении темой от времени обучения из [10]

Повышение мотивации к углубленному изучению программирования.

Многочисленные исследования, обзор которых приведен в [13], продемонстрировали эффективность образовательной технологии обучения «перевернутый класс» – Flipped classroom – (например, методики наставничества) на курсах информатики. В попытке применить активную обучающую методику на лекциях по дисциплине были разработаны вопросы, которые предлагаются на лекциях. Набор состоит из 50 вопросов с несколькими вариантами ответов, на которые студенты отвечают, как индивидуально, так и в группах. Наш опыт преподавания показывает, что активная обучающая методика не повлияла на посещаемость. Отзывы студентов о курсе лекций и до ее применения были в основном положительными.

Неопределенный результат применения методики объясняется несколькими факторами. Во-первых, в классическом виде метод наставничества эффективен для первоначального знакомства с понятиями. Достаточная подготовка перед занятиями является проблемой для студентов старших курсов в применении этой методики, так как обучаемые уже знакомы со всеми понятиями, упомянутыми в глоссарии курса. В результате студенты полагают ненужным готовиться к лекциям, считая, что их знаний достаточно. Во-вторых, метод наставничества ориентирован на учебу друг у друга без вмешательства преподавателя. Роль преподавателя сводится к организации в аудитории дискуссии на тему задаваемых вопросов. В теории методом наставничества учащиеся получают знания посредством наблюдения, изучения, обучения других учащихся и на основе собственного опыта. Проблемой студентов старших курсов является то, что опыт недостаточен, и отсутствует осознание этой недостаточности.

Эффективность вопросов, задаваемых на лекции, сильно зависит от их качества. Вопросы для лекции отличается по педагогической цели от вопросов для экзамена или домашних заданий [14]. Они должны разрабатываться, чтобы направлять внимание учащихся, стимулировать когнитивные процессы и содействовать формулированию и сопоставлению идей. Вопросы, предлагаемые на лекциях, должны формироваться на основе глоссария курса, направлены на выявление заблуждений и формирование правильных представлений. Очевидно, что все термины глоссария (150 понятий) не могут быть использованы в вопросах. На каждой из 24 лекциях времени хватит на обсуждение взаимосвязи не более двух, трех понятий. Разработанные 50 вопросов имеют педагогическую цель демонстрации того, что имеющиеся у обучаемых представления об основных понятиях программирования слишком поверхностны. Они применяются на лекции для того, чтобы снизить уверенность обучаемых в достаточности

их знаний и таким образом повысить мотивацию для восприятия материала лекции. Таким образом, в остальном тактика преподавания остается традиционной. Вопросы к аудитории на лекции имеют цель организации дискуссии, в которой у обучаемых должно появиться сомнение в достаточности их знаний по обсуждаемой проблеме.

Тактика преподавания с использованием вопросов, снижающих уверенность в достаточности знаний, конечно, не может увеличить посещаемость занятий. Однако она помогает удержать тех, кто однажды решил их посетить. При ее применении после нескольких первых лекций число слушателей стабилизируется до конца курса.

Оценка достижения целей курса.

В начале этого курса обучаемые уже знакомы с понятиями, которые будут изучаться. Но этот курс не повторение пройденного. Так как целостность системы знаний в обсуждаемом курсе является важной характеристикой успешности усвоения курса, то нужна специальная методика ее контроля, состоящая проверке «интеграции знаний». В качестве наиболее просто проверяемого признака целостности знаний используется степень взаимосвязи наиболее важных терминов изученной дисциплины. Предполагается, что чем выше измеряемый показатель целостности системы понятий курса, тем лучше последний усвоен. Проведённая описываемым методом экспериментальная проверка знаний студентов подтверждает обоснованность выдвинутых предположений [15]. В начале курса студенты выполняют тест, в котором задачей является указание взаимосвязей терминов из глоссария курса. Результат теста – это граф, вершинами которого являются понятия глоссария, а ребрами – связи между понятиями. Цель курса состоит в том, чтобы этот граф был односвязным. В конце курса студенты выполняют этот тест повторно. Курс усвоен успешно, если в графе, построенном по результатам повторного теста повышается связность. Этот метод контроля усвояемости курса использовался при чтении курса последние три года. Односвязный граф никогда не возникал в тесте, проводимом в начале курса. В повторном тесте мера связности повышалась у подавляющего большинства студентов.

Заключение.

В дидактике под методом обучения понимают способы достижения цели обучения, способы совместной деятельности преподавателя и студентов. Для студентов старших курсов исследования продемонстрировали эффективность образовательной технологии обучения методом «интеграцией знаний» на курсах информатики. Под «интеграцией знаний» в статье понимается способность представлять всю совокупность связей между понятиями информатики. В данной статье подчёркивается важная роль способности программиста объединять идеи (ability to make links among ideas) при конструировании программы, и что «интеграция знаний» ведёт к глубокому пониманию, поскольку акцентирует связи между научными идеями (the coherence across science ideas).

В работе используется метод, позволяющий получить количественную оценку систематичности концептуального усвоения учебного курса. Метод строится на процедуре учёта связей между понятиями, которые указал студент в ходе проверочного тестирования. В графе результатов тестирования компоненты связности графа группируют связанные по мнению студента между собой понятия в несколько непересекающихся групп. В идеале все термины должны образовывать единую группу, но на практике так не получалось, причём некоторые компоненты связности состоят всего из 2-3 терминов (такие группы уместно считать отдельными фактами, которые студент не связывает с общей картиной). Результаты экспериментов также свидетельствуют, что студенты очень плохо различают типы связей между понятиями: они часто путают классические часть/целое и общее/частное (класс/подкласс), не говоря уже об остальных видах связей.

Список литературы

[1] Черемисинов Д.И. Учебный курс «Интерфейсы информационных систем» в обучении специалистов по информационным и коммуникационным технологиям // Информационные технологии и системы 2016 (ИТС-2016): материалы междунар. науч. конф. (БГУ-ИР, Минск, Республика Беларусь, 26 октября 2016) = Information Technologies

and Systems 2016 (ITS-2016): Proceeding of the Inter-national Conference (BSUIR, Minsk, Republic of Belarus, 26th October 2016) / редкол. : Л. Ю. Шилин [и др.]. - Минск : БГУИР, 2016. – С. 272-273.

[2] Keet, M. C. Experiment with Peer Instruction in Computer Science to Enhance Class Attendance // Huillet, E., Eds. Proceedings 23rd Annual Meeting of the Southern African Association for Research in Mathematics, Science, and Technology Education (SAARMSTE'15) – Maputo, Mozambique, 2018. – P 319-331.

[3] IEEE/AIS/ACM Joint Task Force on Computing Curricula. Computing Curricula 2005. The Overview Report covering undergraduate degree programs in Computer Engineering, Computer Science, Information Systems, Information Technology, Software Engineering. 2005 (http: //www.acm.org /education /curricula.html).

[4] Рекомендации по преподаванию информатики в университетах / Ред. В. Л. Павлов, А. А. Терехов. – СПб.: Изд-во СПбГУ, 2002. –367 с.

[5] Кознов Д. В. Методика обучения программной инженерии на основе карт памяти // Системное программирование – Т. 3, № 1, 2008. – С. 121-140.

[6] Bryant R. E., O'Hallaron D. R. Computer Systems: A Programmer's Perspective – Prentice Hall, 2002. – 978 p.

[7] Glossary of Computer Related Terms [Electronic resource] / Hohn M. W. 2011. – Mode of access: http://www.math.utah.edu/ wisnia/glossary.html – Date of access: 1.09.2016.

[8] Glossary of Computer and Internet Terms for Older Adults [Electronic resource] / National Institute on Aging 2016. – Mode of access: http://www.cheyney.edu/informationtechnology/documents/glossarycomputerterms.pdf – Date of access: 1.09.2016.

[9] Crouch C. H., and Mazur E. Peer instruction: Ten years of experience and results // American Journal of Physics – v. 69, 2001. – pp. 970-977.

[10] Anzanello M. J., Fogliatto F. S. Learning curve models and applications: literature review and research directions // International Journal of Industrial Ergonomics, 41, 2011. – pp. 573-583.

[11] Ferriss T. Meta Meta-Learning. Frequency: Cramming six months culinary school into 48 hours. // The 4-Hour Chef: The Simple Path to Cooking Like a Pro, Learning Anything, and Living the Good Life – New Harvest; 2012. – pp. 86-87.

[12] Питер Л. Принцип Питера, или, Почему дела идут вкривь и вкось. – АСТ, 2002. – 285 с.

[13] Szabo C., Falkner N., Knutas A., Dorodchi M. Understanding the Effects of Lecturer Intervention on Computer Science Student Behaviour. // ITiCSE-WGR'17, July 3–5, 2017, Bologna, Italy – ACM, New York, NY, USA, 2017. – pp. 105-124.

[14] Beatty I. D., Leonard W. J., Gerace W. J., Dufresne, R. J. Designing effective questions for classroom response system teaching // American Journal of Physics – v. 74(1), 2006. – pp. 31-39.

[15] Еремин Е.А. Количественная оценка целостности системы базовых понятий как мера усвоения учебного материала // Образовательные технологии, 2014, N 4, с.78-98.

ABOUT THE METHODOLOGY OF TEACHING PROGRAMMING AT THE LAST YEARS OF HIGHER EDUCATION

D.I. Cheremisinov
Leading researcher, United Institute of Informatics Problems of the National Academy of Sciences of Republic of Belarus,PhD of Technical Sciences, Associate Professor

United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Republic of Belarus
E-mail: cher@newman.bas-net.by

Abstract. The discipline "Interfaces of information systems" in the curriculum for training specialists in information technology is aimed at creating a body of knowledge about the relationship between key concepts and principles of programming. The course reflects modern concepts and methodological approaches to teaching computer science to senior students of universities. The course material can be considered the core of the structure of computer science concepts, since it is located in the area of overlap of all four of its parts. One of the important tasks of teaching the course is to demonstrate that the ideas formed by students at the initial stages of learning about the basic concepts of programming are too superficial.

Keywords: Computer science education, Information, Interaction, Interface, Leaning curve, Peer instruction method.