УДК 004

А. Г. Леонов, Ю. А. Первин

Система КуМир в непрерывном школьном курсе информатики

Из описанного здесь обоснования школьного курса информатики непосредственно следует его главное качество – непрерывность информатики в цикле школьного обучения. В рамках этого следствия приведены доводы для деления школьного курса на три составные части – пропедевтический курс, базовое и профильное обучение информатике. Непрерывный курс поддерживает идею единого подхода к содержанию и методике учебных языковых систем программного обеспечения. Рассказано о языковой системе КуМир, обеспечивающей непрерывный школьный курс информатики на всех его этапах.

Ключевые слова: школьная информатика, непрерывный курс, КуМир, пропедевтический курс, базовое обучение, предпрофильное обучение, программирование.

A. G. Leonov, Ju. A. Pervin

The System KuMir in a Non-Stop School Course of Informatics

The justification and topicality of the non-stop course of Informatics, its content and methodology are described in relation to develop and disseminate new versions of software known as a methodical system KuMir, which played an important role in informatization of the national school. We consider the updated features in all three stages of school education: a propaedeutic course, a main course and pre-professional education. The non-stop evolution of the software courses is marked.

Keywords: a non-stop course of Informatics, KuMir, a main course, a propaedeutic course, pre-profiled training, programming.

Теория и практика проектирования прикладных программных систем и, в частности, языковых высокоуровневых систем программирования многократно демонстрировали тот факт, что эффективность программного обеспечения в значительной степени определяется требованиями предметной области. Объективной реальностью поэтому является многообразие языков программирования с разными парадигмами и средствами реализации, различными трактовками основных используемых объектов. Проблемы информатизации образования как главного направления информатизации современного общества определяют весьма специфическую предметную область. Отмечая требования этой области, А. П. Ершов в статье [5], предваряющей представление реформы 1985 года, которая вводила курс «Основы информатики и вычислительной техники» в советскую общеобразовательную школу, писал:

«Отбор содержания и методов обучения новому предмету оказался очень нелегким делом, главным образом, потому, что значительная часть факторов, влияющих на конкретные решения, носила противоречивый характер. Укажем на основные противопоставления:

- стабильность и общепринятость научного багажа общего образования наряду с динамичным и становящимся характером информатики;
- стратегическая необходимость компьютерной грамотности и недостаточная подготовленность общественного сознания, в частности, в учительской среде;
- разные облики программирования: математическая деятельность и сумма приемов работы с ЭВМ, интровертивное и экстравертивное программирование, системное и прикладное программирование, трудная специфическая профессия и массовая человеческая практика;
- разнообразие языковой практики программирования и единство учебного процесса в школе;
- работа в вычислительном кабинете с неограниченным доступом к ЭВМ и традиционная форма школьного урока;
- необходимость единого нормативного начала и необходимость разнообразного и поискового эксперимента.

Надо сказать, что в одном эти разноречивые факторы действовали совместно: они властно диктовали, требовали жестко ограничить учебное пособие по объему материала и сделать его максимально доступным, прежде всего, для учителей – математиков и физиков, которым будет поручено преподавание основ информатики и вычислительной техники после сравнительно короткой курсовой подготовки».

Но еще задолго до реформы 1985 года, в 1979 году, академик А. П. Ершов вместе со своими коллегами по группе школьной информатики ВЦ Сибирского отделения АН СССР обосновал актуальность нового школьного предмета [10][1], дав конкретную формулировку целей информатизации школьного образования. Тот факт, что информатизация образования не является данью текущей или перспективной моде, был ясен всем. Однако среди многих высоких администраторов и даже известных ученых было распространено суждение, способное подменить собою цели информатизации образования и грозившее стать порочным стереотипом. Они говорили, что число компьютеров велико и они достаточно до́роги (это было верным утверждением даже в канун реформы 1985), их будет становиться все больше и больше (и это верно!), их производительность будет все больше увеличиваться (эволюция компьютеров подтверждала и этот факт), в связи с этим (и далее следовал лежащий на поверхности, но совершенно необоснованный «вывод») для эффективного использования такого стремительно растущего парка вычислительной техники необходимо обучать более медленно развивающемуся программированию все население планеты. А. П. Ершов пошел иным путем в построении обоснования школьной информатики: он заметил неверную расстановку акцентов в социальной оценке ценностей информатизации современного общества – более, чем эффективность использования дорогостоящего и распространенного оборудования, важно формирование нового, современного стиля мышления у всего подрастающего поколения, который (стиль) был бы адекватен требованиям информационного общества: еще до реформы 1985 года человеческое общество уже заслуживало название информационного (в одном из своих публичных выступлений А. П. Ершов говорил, что общество становится *информационным* с того момента, когда стоимость одного книжного типографского знака начинает превышать цену хранения одного печатного символа в памяти компьютера).

Не повторяя здесь аргументацию ершовского обоснования, проследим его основную линию, чтобы обнаружить в числе главных результатов столь актуальный для *исследования по методологии и технологии проектирования учебного программного обеспечения* вывод о требованиях к школьным языковым программно-методическим средствам учебной ориентации. Итак, стартовая позиция в логических рассуждениях А. П. Ершова – формулировка цели внедрения информатизации общества как формирования нового (замена существующего) стиля мышления молодого человека, вступающего в жизнь в условиях информационного общества. По определению Р. С. Немова [18]: «Мышление – это движение идей, раскрывающее суть вещей. Его итогом является не образ, а некоторая мысль, идея... Мышление – это особого рода теоретическая и практическая деятельность, предполагающая систему включенных в нее действий и операций ориентировочно-исследовательского, преобразовательного и познавательного характера». Системные связи между законами и сущностями изучаемых вещей и явлений выстраиваются в результате целенаправленной познавательной деятельности человека в некую интеллектуальную сеть способом, который выбирает сам человек. Этот способ ментальных построений и операций, производимых над идеями и приводящих к постижению или созданию понятий, законов и сущностей, называется *стилем мышления*.

Первой задачей, поставленной А. П. Ершовым перед группой своих коллег, с которыми он обосновывал школьный непрерывной курс информатики, стало построение модели выпускника эпохи информационного общества, при этом под моделью понималась совокупность знаний, умений и навыков, которыми должен владеть выходящий из школы современный молодой человек. Главной идеей такого моделирования стал выбор прототипа этой модели. А. П. Ершов предложил выбрать таким прототипом программиста как человека, который в существенно большей степени, чем любой другой специалист, всей своей повседневной как рутинной, так и творческой деятельностью формировал в себе умения и навыки, помогавшие ему наиболее эффективно использовать вычислительную технику. Интересно в этом отношении высказывание А. П. Ершова в его философской статье [8]: «Про-

[1] Позднее это, ставшее раритетом, препринтное издание Вычислительного центра СО АН СССР было в связи с десятилетием реформы 1985 года выпущено ретроспективным изданием в журнале «Информатика и образование». – №1. – 1995 (32).

граммист обязан обладать способностью первоклассного математика к абстракции и логическому мышлению в сочетании с эдисоновским талантом сооружать все, что угодно, из нуля и единицы, он должен соединять в себе аккуратность бухгалтера с проницательностью разведчика, фантазию автора детективных романов с трезвой практичностью бизнесмена, а кроме того, иметь вкус к коллективному труду, быть лояльным к организатору работ и т. д... Программист – солдат второй промышленной революции и как таковой должен обладать революционным мышлением и мужеством».

Обоснование школьной информатики начинается с рассмотрения перечня основных знаний, умений и навыков, которые необходимы программисту как прототипу модели, эффективно использующему современный универсальный дидактический инструмент – компьютер.

- Умение планировать структуру собственных действий: программист обязан предвидеть в деталях еще не реализованное действие описываемой им программы, используя для этих целей команды – элементарные стандартизованные языковые средства.
- Умение строить информационные модели: программист очень естественно убеждается, что хотя компьютер и способен распознавать типы и структуры используемых в программе сведениях, тем не менее возможность предварительно передать компьютеру дополнительную информацию о данных существенно повышает эффективность человеко-машинной системы.
- Умение организовать поиск информации: актуальность этого умения близка программисту, который ощущает, что операции вычислений и обработки информации вообще выполняются многократно быстрее, чем поиск используемых операндов, и тем более если размещаются они в непредсказуемых областях запоминающих устройств с различным быстродействием выборки. Следовательно, для суммарного быстродействия создаваемой программы необходимо ясно представлять где, как и какую информацию следует хранить и искать, и предусмотреть эти действия в программе.
- Дисциплина и структурированность общения: необходимые строгость и формализм общения человека с компьютером обязывают программиста внимательно учитывать уровень программных средств интерфейса. Действительно, если машина практически лишена программного обеспечения, то общение с нею возможно лишь на очень трудоемком, с точки зрения человека, языке двоичного кодирования информационных символов; в компьютерах, оснащенных трансляторами с языков программирования высокого уровня, можно записывать задания машине, например, в виде простых математических формул, наконец, если программист использует машину в многоязыковой среде, включающей разнообразные информационно-технологические инструменты, ему достаточно уметь скомпоновать программу из готовых блоков, не зная подчас языки реализации таких блоков.
- Навыки своевременного обращения к компьютеру необходимо использовать всякий раз, когда возникает потребность в автоматизированных процессах обработки информации. Следовательно, речь должна идти не только о широкой эрудиции программиста, но еще чаще о компетентности применения простых, ранее не знакомых и не изучавшихся приемов работы с информацией.

Рис. 1. Умения и навыки эффективного использования вычислительной техники

Важность перечисленных на рис. 1 знаний, умений и навыков понятна, однако вывод об их значимости в общеобразовательной школе делать нельзя раньше, чем будет показана связь этих категорий с умениями и навыками существенно более важными и широкими, общечеловеческими, общекультурными и социальными.

Действительно, очень востребованное сегодня умение планирования своей деятельности актуально в каждой профессии: руководителя промышленного производства, спортивного тренера, агронома, военного стратега. И при планировании посевных площадей, и при проведении спортивной тренировки, и при сооружении моста навыки планирования нужны в той же мере, в какой они необходимы программисту при создании компьютерных программ. Не в меньшей, и даже, пожалуй, большей степени навыки планирования востребованы учительством: трудно представить себе учителя, ведущего импровизированный урок. Он планирует и занятие, и неделю, и учебную четверть, и учебный год.

Умение моделирования нужно каждой творческой личности: архитектор сделает сначала домики проектируемого квартала из фанеры и лишь потом – в натуре, из кирпича; авиаконструктор сначала строит модель самолета, математик пишет уравнение изучаемого процесса. Информационное моделирование, которое делает программист, описывая структуры данных в программируемой задаче, – это частный случай моделирования. Для учителя модель ученика – формально записываемые требования к знаниям и умениям учащихся.

Независимо от профессии каждый творчески работающий человек испытывает острую необходимость в методах организации и хранения информации. При этом переводчик с одного языка на другой активно использует метод бинарного поиска (так быстрее всего искать слова в словаре), историк практически всегда предпочтет хронологическую последовательность представления информации, а библиотекарь организует свои каталоги в алфавитном порядке. Любой коллекционер – от заурядного филателиста до высококвалифицированного музейного работника – назовет вам признаки упорядочения его коллекций, превращающие набор объектов в содержательную культурную ценность. Для учителя поиск информации имеет принципиальное значение: сейчас уже ясно, что ни 12-летним образованием, никаким другим экстенсивным способом обучения невозможно решить проблему передачи школьнику за 10 лет той суммы знаний, которую человечество накапливало тысячелетиями. И именно информационный взрыв последних двух десятков лет показал стратегический путь обучения – научить ребенка искать информацию, необходимую ему для решения поставленной задачи в тот момент, когда в ней возникает необходимость. Другими словами, надо научить школьника учиться.

Проблемы коммуникации в информационном обществе существенно повышают свою важность потому, что постоянно растут требования не только к объему используемой информации, но и к скорости ее передачи по каналам связи. Такие заметные количества и скорости передачи информации приводят, во всяком случае, в организации учебно-воспитательного процесса на всех уровнях образования и во всех его формах к инновационным, качественным его изменениям. Поэтому на рис. 2 отмечен в числе принципиальных переход от дисциплины общения и структурирования сообщений к умениям и навыкам общения в широком, общекультурном смысле слова.

Последняя (правая) из стрелок рис. 2 появилась только вместе со становлением информационного общества. К этому моменту уже была решена проблема использования механических, электрических и электронных устройств как помощников человека, способствующих механизации и автоматизации его физического труда. Информационное общество поставило новые задачи, в связи с этим достижения в сфере поисковых систем, искусственного интеллекта, анализа естественных языков и др. делают актуальным инструментарий автоматизации различных сфер интеллектуальной деятельности человека, ранее не автоматизировавшихся.

Объединяемая совокупность знаний, умений и навыков, представленная на рис. 2, и называется операционным стилем мышления человека информационного общества.

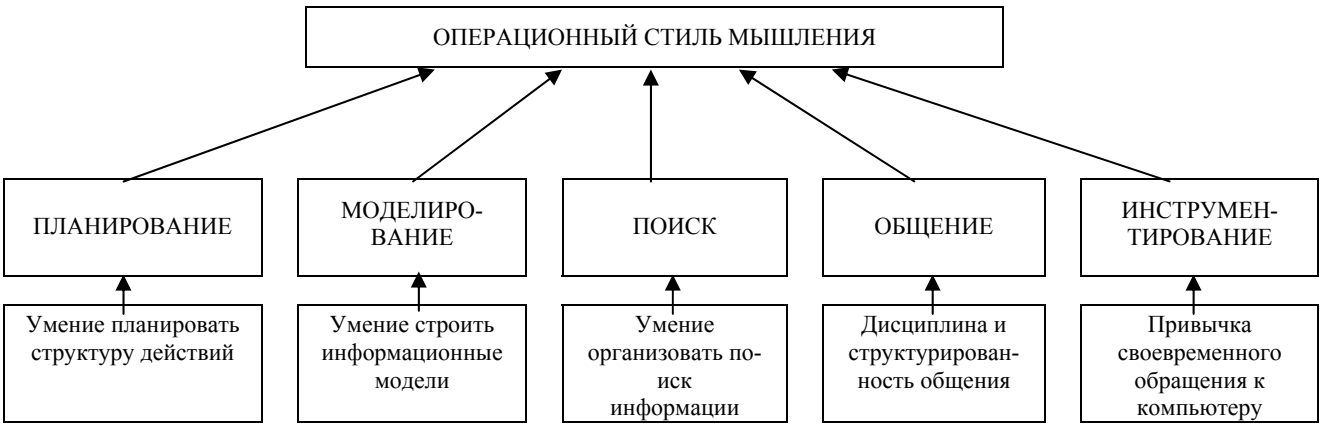

Рис. 2. Операционный стиль мышления

Стиль мышления, обоснованный в качестве образовательной цели школьной информатики и описанный в работах А. П. Ершова, был назван «программистским», и только несколько лет спустя сам автор предложил использовать более точный термин «операционный стиль мышления», в котором подчеркивались как алгоритмическая направленность этого способа мышления, позволяющая и сего-

дня многим употреблять термин «алгоритмический стиль мышления» в тех случаях, когда речь идет о важнейшем педагогическом назначении школьной информатики, так и его операциональная сущность, благодаря которой представления о прикладных аспектах школьной информатики часто совмещают с понятием информационных технологий. Вместе с тем ясно, что операционный стиль мышления представляет собой дидактическую категорию, более широкую, чем алгоритмика и информационные технологии, рассматриваемые по отдельности.

Схема, представленная на рис. 3, которая определяет операционный стиль мышления, будет продолжена ниже, чтобы показать не только актуальность, но и реализуемость главной поставленной задачи – формирование стиля мышления нового, молодого поколения общества. Однако уже сейчас могут быть сделаны выводы из актуальности проблемы, которая является, по существу, социальным заказом общества системе образования. Поскольку такие выводы по определению касаются общества в целом, а не отдельных личностей, проблема формирования стиля мышления должна стать задачей некоторого социального института, единого для *всех* членов общества. Таким социальным институтом не может стать ни профессиональный колледж, ни детский сад. А вот миновать в своем становлении государственно обязательное общее, школьное, среднее образование человек не может. Значит, задачи, представленные на рис. 2, должны решаться именно в общеобразовательной школе.

И еще два вывода можно сделать уже сейчас, не углубляясь в реализацию схемы, изображенной на рис. 3. Действительно, с одной стороны, коль скоро речь идет о формировании мышления молодого человека, такая задача должна ставиться достаточно рано – *в начальной школе*: запоздание с формированием стиля мышления (а следовательно, с формированием личности) может затем привести к ломке уже устоявшегося стиля мышления, что и сложнее, и, по оценкам психологов, опаснее. С другой стороны, с необходимостью понимать и использовать инструментальные средства современных универсальных или специализированных компьютеров не следует торопиться потому, что курс должен быть построен так, чтобы стать нужным не только сегодняшним людям, но и тем, для кого будут современными машины более поздних поколений.

Отсюда вывод: социальным заказом общества является необходимость некоторого *непрерывного* курса, занимающего широкий диапазон школьного обучения, начиная с порога школы. Очевидно, что такой протяженный по времени курс длиной в несколько учебных лет должен быть построен по принципу дидактической спирали [22], по которой можно двигаться вверх, поднимаясь каждый год на новый понятийный уровень.

Коль скоро выясняется, что поставленная задача должна быть решена в общеобразовательной средней школы, неизбежно возникает вопрос о том, какой из школьных дисциплин следовало бы ее поручить. Преподаватель каждой дисциплины – от гуманитарных до естественных (биологии, физики, химии и, конечно, математики) – мог бы привести аргументы в пользу своей науки. Но очевидно, что только *информатика* – и единственная из них – обладает полным концептуальным набором (множеством элементов, понятий, операций, отношений), с помощью которых формируются навыки и умения нижнего уровня – навыки эффективного использования компьютерной техники, а следовательно, и общекультурные умения и навыки, формирующие операционный стиль мышления (рис. 3).

И в самом деле, для того, чтобы формировать навыки планирования, весьма полезными оказываются *управляющие структуры* – серии, ветвления, циклы, – из которых, как доказано в теоретическом программировании, может быть построено описание любого процесса, в том числе мыслительного.

Многообразные *структуры данных*, определяемые в информатике – различные типы чисел, символы, строки и тексты, массивы, записи, файлы разных форматов, – предлагают конструктивный материл для проектирования информационных моделей и в целом для моделирования в разных сферах интеллектуальной творческой деятельности.

Поисковые механизмы информатики – от простого перебора до сложных способов поиска в системах управления базами данных и в сложнейших современных сетевых поисковиках – обеспечивают функционирование многочисленных прикладных программных систем.

Общение (как человеческое, так и машинное) становится эффективным только тогда, когда оно опирается на доведенные до автоматизма навыки использования процедур, функций, макросов.

Наконец, *инструментирование* (автоматизация и информатизация) всевозможных интеллектуальных видов деятельности становится возможным с помощью компьютерных программ и их систем.

Схема, представленная на рис. 3, логически замыкает цепочку ментальных объектов от азов информатики до сформированного стиля мышления. Теперь можно продолжить выводы, прямо вытекающие из обсуждаемого здесь обоснования школьного курса информатики. Напомним, что первыми в списке таких выводов стоят тезисы о непрерывности курса и его стартовой позиции.

Рис. 3. Место информатики в формировании операционного стиля мышления

В советской педагогической школе было принято при анализе учебного процесса оперировать понятием методической системы обучения, введенной А. М. Пышкало. Она представляла собой «структуру, компонентами которой являются цели обучения, содержание обучения, методы обучения, формы и средства обучения» [25]. Такую пятикомпонентную педагогическую систему часто используют и нынешние исследователи учебного процесса. Нам важно оценить, в какой мере выводы из обоснования школьного курса информатики повлияли на компоненты методической системы обучения. Цели обучения, как было показано выше, сливаются с очерченными проблемами формирования операционного стиля мышления.

Прослеживая все остальные компоненты курса информатики, приходится постоянно сталкиваться с социальными стереотипами, возникающими на пути информатики как школьного курса и тормозящими его развитие. Одним из первых таких стереотипов стало установление возрастного порога информатики в школе. Действительно, первые государственные учебники по общеобразовательному школьному курсу информатики в реформе 1985 года (см. [20], [34]) были адресованы учащимся двух выпускных классов и учителям информатики. Аргументация была простой, если не сказать тривиальной: мол, курс трудный, требующий высокого уровня общих математических и физических знаний и, значит, доступный не ранее предпоследнего класса. Отношение к информатике как курсу, доступному только старшим школьникам, долго господствовало в общественном мнении, подстегиваемом не только недостаточно компетентными СМИ, но и администраторами системы образования. Пришлось приложить немало усилий и потратить много времени, чтобы сломать этот стереотип, несмотря на отмеченные выше толкования целей информатизации образования, вытекающие из них выводы и постоянно расширяющийся педагогический эксперимент энтузиастов информатики и ее методы, используемые в учебном процессе начальной школы. Достаточно сказать, что еще в 2004 году при создании Большого Московского семинара по методике раннего обучения информатике пришлось декларировать в числе первоочередных его основных задач конструктивную борьбу со стереотипом возрастного порога информатики в школе [20].

Содержание обучения в информатизированной школе в том виде, в каком оно виделось авторам концепции непрерывного курса информатики [10], представлено в табл. 1. В содержании обучения

выделяются четыре части (модуля) с не очень четкими возрастными границами. Первый модуль относится к пропедевтическому курсу, начинающемуся в первом классе, и включает в себя фундаментальные навыки, знания, умения, понятия и представления, необходимые для формирования операционного стиля мышления. Второй модуль – информатизированные, инновационные межпредметные связи, насыщающие информатикой другие школьные дисциплины. Это наиболее подходящее место, чтобы показать, что компьютеры они встретят, не только выйдя из школы в жизнь, а уже сейчас, в окружающей их учебной деятельности по каждому школьному предмету.

Начинающееся вторым модулем базовое обучение информатике завершается в третьем модуле, где школьник, обобщая собственный опыт, накопленный учебной деятельностью в двух предыдущих модулях, умея строить абстракции из частных проявлений процессов или явлений, уже способен увидеть роль информатики в системе научных знаний.

Наконец, последний, четвертый модуль, заканчивает курс. Он непосредственно примыкает к модулю 3 (и завершает его) и полностью поглощает профильное обучение. Этот модуль изучается только теми учениками, которые выбрали для себя физико-математические или информационно-технологические профили обучения. Здесь и только здесь, в год завершения школы выпускникам следует рассказывать о современных компьютерах и современных информационно-программных системах и технологиях: если выпускникам будут называть современным компьютер, на котором они упражнялись в 6-м классе, то такой урок можно приравнять, в лучшем случае, к экскурсии в политехнический музей вместо практики в производственном, действительно современном учреждении.

Таблица 1

Схема содержания обучения информатике в школе

Совокупность фундаментальных навыков, знаний, умений, понятий и представлений, необходимых для формирования операционного стиля мышления	Совокупность прикладных навыков, необходимых для применения идей и методов информатики в других отраслях человеческой деятельности	Система основных положений информатики как науки в соответствии с ее местом в современной системе научных знаний	Комплекс знаний, необходимых для общей ориентации в возможностях современной и перспективной техники и прикладных систем информатики
1–5 классы	3–8 классы	9–10 классы	11 класс

В этой несколько идеализированной схеме пока нет абсолютного совпадения с нынешними учебными планами по информатике и информатизированным предметным дисциплинам, однако год от года (еще точнее, от одного поколения образовательных стандартов к другому) различие заметно уменьшается. Вместе с тем данная схема носит слишком общий характер, чтобы отразить в содержании образования принципиальный тезис о *месте программирования в курсе*. У этого тезиса отношения со школьным курсом информатики еще более сложные, чем у тезиса возрастного порога информатики.

Первый включенный в утвержденную государственную учебную программу курс «Основы информатики и вычислительной техники 9–10» в 1986 году вынужденно был безмашинным. К счастью, такая форма курса со всей очевидностью не могла стать стереотипной. Но курс оказался, по существу, *курсом программирования*. Он включал в себя даже три языка программирования: один – интересный, но сравнительно мало распространенный процедурный язык Рапира, разработанный группой школьной информатики в ВЦ СО АН СССР, другой – широко известный, но методически порочный Бейсик, а третий – алгоритмический, но не имевший транслятора и системы программирования. Зато этот последний язык, называвшийся в то время Е-языком (по первой букве фамилии его автора – А. П. Ершова), или даже более откровенно – Ершолом, дал энергичный толчок проектированию и реализации мощного учебного языка, который сегодня активно эксплуатируется в педагогической вузовской и школьной практике. Это КуМир, создававшийся программистами механико-математического факультета Московского государственного университета им. М. В. Ломоносова [34].

Однако этому школьному курсу информатики сравнительно не долго удалось сохранять статус ориентированного на обучение программированию. Программирование как содержание курса быстро вытеснялось из дисциплины с названием «Информатика», и к 2004 году, когда в государственном образовательном стандарте появился курс «Информатика и информационно-коммуникационные технологии» (ИКТ), от программирования не осталось практически ничего (за исключением школ, сохранивших тяготение к КуМиру). Отсутствие элементов программирования в курсе информатики стало еще одним стереотипом школьной информатики.

История языковой линии в школьной информатике весьма поучительна. Первой оригинальной учебной системой программных средств, ориентированных на пропедевтический курс информатики, стала языковая система Лого С. Пайперта (1969), в которой, впрочем, главным действующим лицом был программный исполнитель – Черепашка [19]. В Советском Союзе пропедевтический курс составлялся совокупностью исполнителей [9]. Существенно, что в этой системе впервые прозвучала идея отбора инвариантных свойств системы исполнителей, конструктивно подводящих к последовательному освоению адаптивного учебного языка (Робик), а затем и учебно-ориентированного языка (Рапира). Интересно с этой точки зрения сравнить системы программирования Робик+Рапира с родившимся почти одновременно КуМиром [9]: в первой из этих двух систем начальный этап освоения – «исполнительский» – оказался более продолжительным, чем во второй. Это совершенно естественно, если вспомнить, что система (Робик)+(Рапира), получившая позднее официальное название Школьница, исходно замышлялась в качестве компонента пропедевтического курса, тогда как при освоении КуМира «исполнительский» этап заметно короче именно по той причине, что его экспериментальное внедрение проводилось, в первую очередь, в базовой части курса информатики, где контингент обучаемых составлялся в том числе и студентами младших курсов вуза. Продуманная инкапсуляция (автономность), а также разумное «взвешивание» пропедевтической и базовой частей курса, предопределили жизнеспособность КуМира, занимающего и ныне достойное место в поддержке курса школьной информатики [15].

Другое, не менее весомое обоснование этой жизнеспособности учебно-ориентированной языковой системы состоит в том, что развитие непрерывного курса информатики раскрыло недостоверность стереотипного утверждения, что современный курс имеет право на исключение программирования из системы школьного обучения. По существу, это обоснование располагается не столько в обсуждении содержания обучения, сколько на границе между содержанием и *методикой* курса: каждый из исполнителей служит не только для локальной дидактической задачи формирования того или иного отдельного умения или навыка, но и подсказывает формальный план решения задачи – алгоритм, который реализуется командами исполнителя. Это прямой путь к языку программирования, который нельзя не использовать. В данном отношении показательна одна из последних педагогических разработок [32], где ученик первого класса работает с программами обработки символьной информации. Термин «программа» здесь используется явно.

Современная тенденция преподавания информатики (особенно в вузовских курсах) предполагает не только развитие алгоритмического стиля мышления [5], но и изучение одного из современных языков, поддерживающих принципы объектно-ориентированного программирования (ООП). К ним можно отнести такие языки программирования, как C++ с богатой историей развития и использования и более молодые C# и Python [17], или хорошо известный Java [33].

Можно сказать, что современные средства ООП содержатся практически во всех современных языках. Даже последние поколения языка Pascal, который был придуман Н.°Виртом исключительно в учебных целях, содержат в себе элементы ООП [35]. Основу этого языка составляла тройка понятий, логически выводимая из объектов и действий над объектами:

Действия $\rightarrow$ *Команды (Циклы)* $\rightarrow$ *Вспомогательные алгоритмы*

Объекты $\rightarrow$ *Величины (Таблицы)*

Однако сами объекты в Pascal было трудно разделять по применению. Не существует полноценной записи для объектов, структурирующих их по предназначению и способам действий с ними. Вероятнее всего, это объясняется общим состоянием информатики в 70-х годах прошлого столетия, когда структурированы были вспомогательные алгоритмы (процедуры), однако аналогичные средства для работы с объектами еще не использовались.

Хотя уже в конце 60-х годов в Норвежском Вычислительном центре (Осло) его сотрудниками: Кристианом Нюгордом и Оле-Йоханом Далем был придуман вполне законченный объектно-

ориентированный язык Simula-67. Этот язык опередил свое время и не был воспринят программистским сообществом [28], хотя именно в нем впервые было введено фундаментальное понятие класса (объекта) и связывание его с допустимыми над ним действиями:

```
Class Rectangle (Width, Height);
        Real Width, Height; ! Class with two parameters
        Begin Real Area, Perimeter; ! Атрибуты.
        Procedure Update; ! Методы.
        Begin  Area := Width * Height;
                Perimeter := 2*(Width + Height)
        End of Update;
        Boolean Procedure IsSquare;
        IsSquare := Width=Height;
        Update; ! Жизнь объекта начинается здесь.
End of Rectangle;
```

Рис. 4. Классы в Simula-67

Язык Simula можно, без сомнения, считать инновационным шагом в развитии языков программирования, однако революционным прорывом явились идеи Алана Кея (Исследовательский центр в Пало-Альто Xerox PARC) [27], воплощенные в первом настоящем объектно-ориентированном языке Smalltalk-72, а затем и в Smalltalk-80.

К этим идеям относятся, во-первых, простота для понимания и красивая идея существования в языке только объектов и сообщений как способов взаимодействия объектов: строки, целые числа, логические значения, определения классов и т. п. – все является объектами. Выполнение программы состоит в пересылке сообщений между объектами. Любое сообщение может быть послано любому объекту, при этом объект-получатель определяет, является ли это сообщение правильным, и что надо предпринять.

```
a := 6. b := 7.
hypotenuseSquared := a squared + b squared. "By
   Pythagoras"
hypotenuse := hypotenuseSquared sqrt.
Transcript show: hypotenuse; flush.
```

Рис. 5. Пример на Smalltalk-80

Во-вторых, Smalltalk, по сути, является не языком программирования, а системой программирования, так как именно в подобной системе можно полноценно реализовать идеи ООП. Забегая вперед, можно добавить, что точно так организованы язык и система программирования КуМир.

Термин ООП подчеркивает первичность объектов над действиями и алгоритмами. Хотя на практике важны и алгоритмы, и объекты: и те, и другие надо уметь структурировать [11] и иметь средство для работы с ними. В различных языках программирования такие понятия имеют различное именование: в языке Ада они названы «пакетом» (package) [26], в языке Modula-2 – «модулем» [3], в C++, C#, Java и Simula – «объектом» и «экземпляром класса» [31], Pascal вводит конструкцию Unit [4], школьный алгоритмический язык КуМир – «исполнитель» [13]. В учебнике А. Г. Кушниренко и др. [13] соответствующее понятие называют еще «информационной моделью исполнителя».

При этом само понятие *исполнитель* более широко по своему значению. Под исполнителем понимается не только конструкция языка программирования КуМир, но и человек, автомат или прочее устройство или группа устройств, связанных общими свойствами и имеющих навсегда фиксирован-

ную систему команд. К важным свойствам исполнителя относится его «незнание» об управляющей им системе, что в ООП именуется *абстрагированием*.

Возвращаясь к системе КуМир, важно отметить, что исполнитель является не только внутренней структурой алгоритмического языка, но и одновременно, подобно знакомым всем исполнителям из реальной жизни, может существовать отдельно от системы программирования. Так, для человека простейшим исполнителем (с минимальной системой команд-предписаний) можно считать электрическую лампочку освещения комнаты, с которой ежедневно приходится сталкиваться каждому. Заходя в темную комнату, человек «включает свет», а покидая ее, «выключает», используя кнопку-выключатель. В таком случае принято говорить, что исполнитель «лампочка» имеет *кнопочный режим управления*.

Существует много более сложных исполнителей, имеющих кнопочное управление: видеоплеер, телефон, автомобиль и, наконец, компьютер. Легко заметить, что ежедневное использование исполнителей и кнопочное управление ими не создают непреодолимых методических трудностей введения педагогом нового для учащихся понятия. Несомненно, понятия класса в С++ потребуют больших усилий как от обучаемого, так и от учителя [2].

Вполне закономерным является изучение в школьном курсе информатики по А. Г. Кушниренко [12] исполнителя *Робот*. И, как следствие, закономерным является желание учащихся управлять роботом не только в кнопочном режиме, но и программируя его, то есть составляя план будущих действий робота:

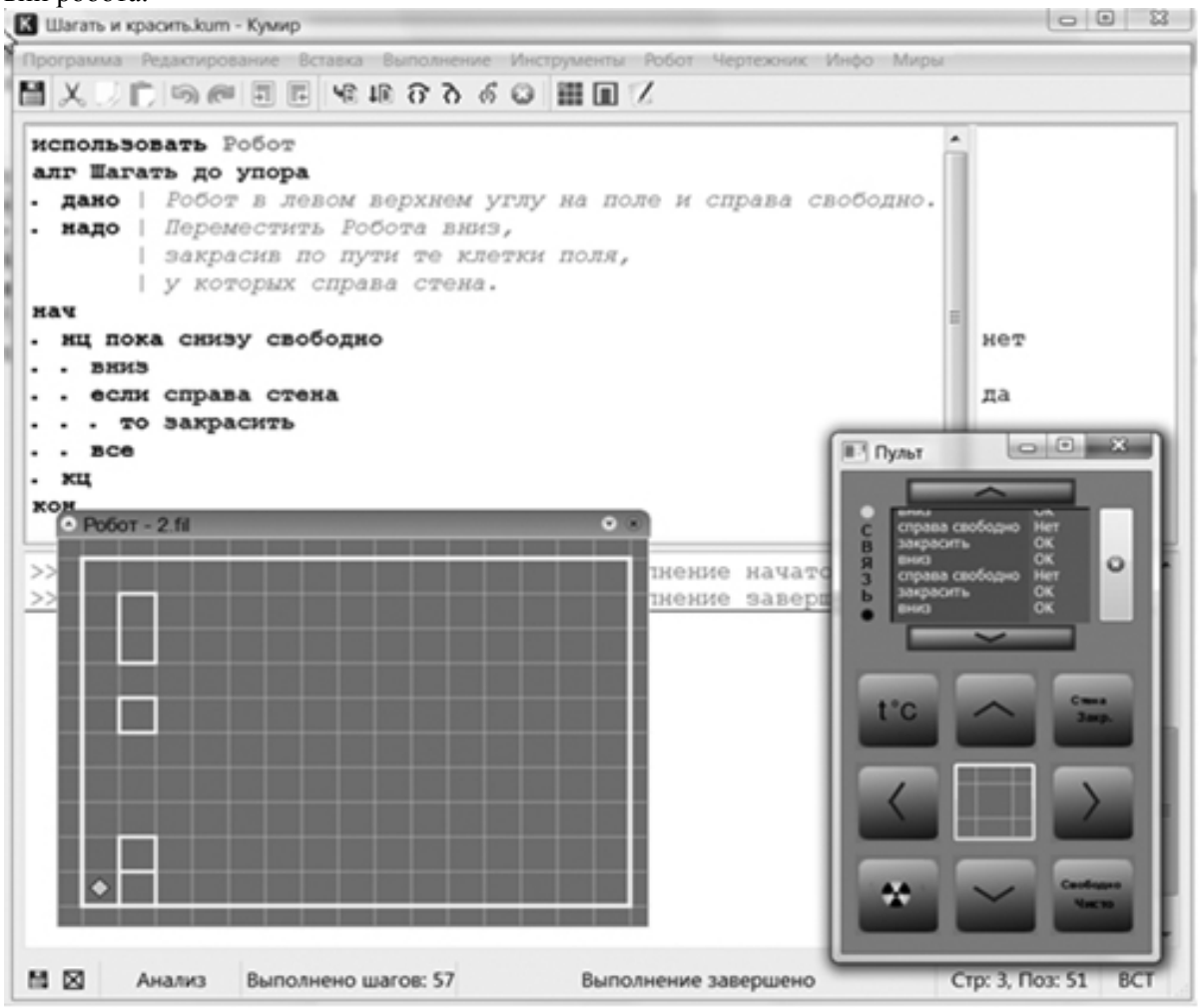

Рис. 6. Пример программы на КуМире (Робот выполнил алгоритм. Справа – протокол работы)

Попутно можно заметить, что, изучая исполнителей системы КуМир, учащийся не видит внутренней (скрытой) сущности и деталей реализации того или иного исполнителя, хотя полностью известен

функционал изучаемого объекта. Это не означает отсутствие обратной связи у исполнителей в КуМире, но относится к еще одному «киту» ООП, именуемому *инкапсуляцией.*

Таким образом, в языке КуМир вводятся четыре фундаментальных понятия информатики:

Действия $\rightarrow$ *Команды (Циклы)* $\rightarrow$ *Вспомогательные алгоритмы*

Объекты $\rightarrow$ *Величины (Таблицы)* $\rightarrow$ *Исполнители*

Первые два понятия отражают методы записи действий и объектов (в частности, большого количества действий и большого количества объектов). Вторые отражают фундаментальные приемы структуризации, которые человечество выработало за последние годы. Эти понятия просты и доступны школьникам, могут быть поняты и освоены в процессе решения задач, и все вместе образуют фундамент, на котором можно развивать и внутренние способности человека к алгоритмическому мышлению, и понимание реалий окружающего мира. Освоив основные понятия современной информационной культуры, можно развиваться в разных направлениях: от изучения способов конструирования структур данных и новых языков программирования до решения более сложных прикладных задач.

Система КуМир позволяет не только использовать готовых исполнителей и обучаться программированию, составляя алгоритмы управления ими, но и создавать новых внутренних исполнителей в программе, имеющих свои локальные и глобальные величины и предоставляющих доступ к алгоритмам исполнителя другим исполнителям.

К числу «трех китов» ООП относят еще такие свойства, как *наследование* и *полиморфизм*. Еще в предыдущей версии системы КуМир, широко распространенной в 90-х годах прошлого столетия, была доступна возможность переопределения операций, что было продемонстрировано примером исполнителя «Комплексные числа». Вполне закономерно, что и современная многоплатформенная версия системы имеет аналогичные возможности. Однако в производственных языках программирования доступны и столь экзотические методы ООП, как множественное наследование. Смысл таких приемов, особенно при обучении, вызывает вполне закономерное сомнение в его надобности. И здесь неуместны аналоги с рекурсией как методом в программировании и замене его итерацией.

Подводя промежуточный итог в обсуждении содержания школьного курса информатики, можно утверждать, что КуМир может занимать вполне заслуженное место в профильном обучении в начальном курсе ООП. При этом использование КуМира позволяет существенно упростить процесс обучения, сократить число затрачиваемых на темы часов и, что наиболее важно, показать учащимся закономерность возникновения ООП, и продемонстрировать его положительные, фундаментально философские качества – инкапсуляцию, полиморфизм и наследование.

Продолжая разговор о школьном курсе информатики и рассматривая его методику, отметим, что КуМир в немалой степени способствует ломке старых, но необычайно устойчивых стереотипов, восходящих к Бейсику с его простой, компактной, а потому дешевой и, значит, очень распространенной системой программирования. Такая простота связана, прежде всего, с *меточной* структурой программы: каждая команда-строка снабжена меткой, порожденной отсутствием процедур в языке. В конце 60-х годов XX века Э. Дейкстра, К. Хоор и А. П. Ершов доказали избыточность безусловных передач управления в языке программирования, использующем вспомогательные алгоритмы (процедуры и функции), что позволило избавиться от меток и, как следствие, от блок-схем. Достоинством блок-схем часто называли их «наглядность». Однако блок-схемы оказывались эффективными лишь при программировании простейших задач, не превосходящих по сложности нахождение максимума из трех чисел или, в лучшем случае, вычисление общего наибольшего делителя. В чуть более сложных задачах, когда блок-схема не умещалась на одной странице, возникала путаница при переносах ее на другие страницы. Формирование блок-схемы – это лишний, искусственно создаваемый источник ошибок на этапе, где поиск ошибок приводил к дополнительным трудностям, не поддающимся автоматизации. Со времени Э. Дейкстры и К. Хоора грамотные программисты не используют блок-схемы, записывая программы любой сложности с помощью ступенчатых левых отступов, по которым легко восстановить структуру написанной программы. Такой методический прием оказался единым как для высокоуровневых языков программирования, так и для языков разметки гипертекстовых сайтовых страниц (HTML). Возвратом в пещерный век следует считать блок-схемы в написанных в XXI столетии программах. Методическое наследие недоучившихся на скоростных курсах учителей теперь приходится искоренять с самых первых шагов программирования [22]. В этом отношении КуМир оказался одним из наиболее последовательных и методичных учебно-ориентированных языков про-

граммирования, строя листинг на экране компьютера так, что все левые отступы в управляющих структурах воссоздаются автоматически. В КуМире попросту негде, при всем желании, нарисовать методически порочную блок-схему.

Требования к программно-техническому обеспечению учебного процесса затронули также *формы* и *средства* из пятикомпонентного представления методической системы обучения. Действительно, все формы урока – контрольные работы и домашние задания, фронтальные опросы и индивидуальные траектории обучения – сохраняются и в информационно насыщенных школах [29]. Вместе с тем появляются и педагогические инновации: включение компьютеризованных фрагментов в урок по информатизируемым школьным предметам. Современные методические пособия рекомендуют строить последовательность этапов урока так, чтобы компьютеризованный фрагмент урока становился его кульминационным моментом.

В стереотипной дискуссии первых пореформенных лет педагоги энергично спорили на тему, кто должен вести урок по предмету в школьном кабинете информатики – учитель информатики или учитель-предметник? Еще более острой становилась дискуссия, когда обсуждалась роль учителя начальной школы. И хотя сегодня сторонники профессиональной «чистоты» учителя-предметника еще существуют, к счастью, в учительской среде все более обоснованным становится утверждение о том, что процесс информатизации школы как социального института общества можно будет считать результативным лишь тогда, когда он станет целью, содержанием, методикой, формой и средствами каждого учителя-предметника. В частности, такой же вывод следует формулировать и по отношению к начальному образованию. Ответственность за организацию и проведение уроков информатики в любой форме в начальной школе обязана взять на себя «хозяйка класса», его классный руководитель. Учитель начальной школы имеет самый высокий авторитет в глазах малышей. Следовательно, наравне с математикой, русским языком и естествознанием учитель начальной школы должен преподавать предмет, который, как следует из обоснования школьного курса информатики, является стержневой предметной дисциплиной, организующей общешкольный учебный процесс. Несмотря на весьма высокий уровень сложности проблемы, состоящей в повышении квалификации педагогов начальной школы, ее особую актуальность, ответственность и ресурсоемкость, откладывать ее или подменять подключением учителей информатики к урокам начальной школы нельзя.

Новациями дидактики современного школьного образования стали не только формы урока, но и формы организации учебного процесса, востребованные информационным обществом. Ни И. Г. Песталоцци, ни К. Д. Ушинский, ни даже С. Френе в их время не могли не только использовать, но даже представить себе педагогические возможности дистанционного обучения, которое ныне становится реальной возможностью и актуальной потребностью учебного процесса не только в вузе, но и в школе, в том числе начальной [21]. Точно так же Джон Дьи не мог представить, какую огромную популярность и педагогическую эффективность получит в XXI веке предложенный им метод проектов, существенным образом реорганизовавший педагогику дополнительного и общего современного образования, благодаря инновационным средствам информатизации, в первую очередь, сетевым.

Заметную роль в системе образования играет олимпиадное движение. Разнообразные формы региональных и всероссийских соревнований – олимпиад, конкурсов, турниров, очных и дистанционных – анимированы средствами информационно-программных ресурсов КуМира. КуМир является программным инструментом Всероссийской системы обработки документов ЕГЭ.

Требования к *средствам технического и программного обеспечения* непрерывного курса информатики как следствия из его приведенного выше обоснования опираются на возрастные особенности модулей курса: младшему школьнику, учебная деятельность которого органично переплетается с игровой, нужны быстродействующие компьютеры, способные выводить на экраны высокого разрешения качественные анимированные изображения; старшие школьники не должны испытывать ограничений в объемах памяти, необходимых для многообразных прикладных систем. Этими требованиями в определенной степени могут быть объяснены ограничения курса, в которых проходило становление реформы 1985: в то время страна не могла дать школам такое количество машин, тем более попросту еще не существовали персональные компьютеры, удовлетворяющие требованиям компьютеризируемой начальной школы. В [23] приведен рассказ об одной из встреч А. П. Ершова со школьниками – участниками Всесоюзной летней школы юных программистов в новосибирском Академгородке на берегу Обского моря. В 1986 году в дополнение к серии «Агатов» и УКНЦ ему удалось поставить класс из 6 новеньких французских персональных компьютеров, богато оснащенных специальной пе-

риферией – световыми перьями, графическими магнитными картами, емкими дискетами, цветной графикой на экранах хорошего разрешения, встроенными динамиками… Но как же были удивлены старшие сибирские школьники и многочисленные гости, не впервые приезжавшие в Академгородок на летние школы и потому без смущения называвшие себя «ветеранами», будучи учениками 9-х или 10-х классов, когда узнали, что решением А. П. Ершова домик с лучшими компьютерами отдается в распоряжение новичков – четырех сибиряков и двух ленинградцев – от 1-го до 3-го класса, составивших кафедру Лого. «Ветераны» в первый же вечер пришли к Ершову за справедливым решением. И состоялся разговор по душам. Он сказал ребятам такие, надолго запомнившиеся слова: «Вы, понимающие, как работает компьютер, – говорил А. П. Ершов, – и умеющие найти в программах любую свою ошибку, немного потеряете, поработав на машинах, на которых вы сумеете написать сложные программы, несмотря на ограниченные возможности ваших машин. А вот малышам, которые еще не умеют средствами программистского ремесла уйти от технических ограничений машин, надо дать компьютеры с высоким быстродействием процессора, с самым хорошим разрешением экранов, с самыми удобными устройствами ввода и вывода. Это нужно для того, чтобы с первых своих шагов в общении с компьютерами они полюбили информатику. Никакого парадокса нет в том, что новичкам, самым юным программистам, которые пока еще не в состоянии сами сделать сложные программы и настройки машин, нужны лучшие из нашего технического арсенала компьютеры».

Таким образом, проблемы обоснования непрерывного школьного курса информатики и прямо, и косвенно связаны с судьбами учебного программного обеспечения. Однако нельзя не сказать, что в целевых установках динамического курса информатики – формировании операционного стиля мышления – тоже происходила и происходит обоснованная эволюция. Двум ярким проявлениям этой эволюции – *компетентностному* подходу в обучении информатике и *универсальным учебным действиям* (УУД) – посвящена заключительная часть статьи.

В последние годы в дидактику прочно вошло понятие компетентности как критерия эффективности педагогической деятельности. Компетентностный подход, в отличие от превалировавшего ранее знаниевого подхода, ориентирует педагогику не столько на накопление знаний учащимися в ходе учебного процесса, сколько на умение использовать знания, умение внедрять их в процессе своей деятельности. Обновление ориентационных дидактических категорий кажется стремительным. Одновременно со становлением школьной информатики изменились взгляды на динамику формирования содержания школьных предметов. Если ранее их стабильное содержание обеспечивалось многолетним апробированием феноменов социальной жизни прежде, чем они пополняли школьную программу, то ныне такое содержание определяется стремительно меняющимися требованиями общества.

Введенный в информатике операционный стиль мышления близок новому дидактическому понятию компетентностного подхода. Новый термин появился после того (и вследствие того), что рожденный в информатике операционный стиль мышления показал свою состоятельность в качестве критерия эффективности педагогической деятельности и был перенесен в частные методики других школьных дисциплин и, тем самым, стал общедидактической категорией.

Компетентностный подход – особая форма познания и осуществления образовательной деятельности, управления ею в условиях конкретных отраслевых границ и с позиций компетенций, определяемых обществом [30]. Операционный стиль мышления чрезвычайно близок к *информатической компетентности*, которую сегодня принято называть *ИКТ-компетентностью*. Компетентностный подход смог появиться на свет только после того, как были осознаны его корни в интегративной сущности информатики. В соответствии с этим подходом система общего образования должна быть нацелена на формирование *ключевых компетентностей*.

Ключевая компетентность:

- обладает *интегративной* природой, то есть вбирает в себя ряд однородных или близкородственных умений и знаний, относящихся к широким сферам культуры и деятельности (информационной, правовой и пр.);
- *многофункциональна*, то есть овладение ею позволяет решать различные проблемы в повседневной жизни;
- *надпредметна* и *междисциплинарна*, то есть применима в различных педагогических ситуациях;

- требует значительного интеллектуального развития; *многомерна*, то есть включает различные умственные процессы и интеллектуальные умения.

Компетенция – это, условно, то, к чему следует стремиться в том или ином виде деятельности, а *компетентность* – то, что достигнуто в этом направлении. Компетенция – круг вопросов, отражающих сущностные позиции требуемого качества, нормативное содержание признаков какого-либо опыта человеческой деятельности в его целостном представлении.

Многообразие педагогических практик, растущий потенциал программно-методического инструментария и актуальность раннего обучения информатике привели к необходимости очередного этапа стандартизации образования. Процессы глобализации, информатизации, ускорения внедрения новых научных открытий, быстрого обновления знаний и профессий выдвигают требования повышенной профессиональной мобильности и непрерывного образования. В стандартах второго поколения предложен новый уровень обобщений дидактических категорий. Системно-деятельностный подход позволил выделить основные результаты обучения и воспитания на новом уровне обобщений – формируемых школой *учебных универсальных действий* (УУД).

Расширением основных требований общества к образовательной системе сегодня принято считать:

- формирование культурной идентичности учащихся как граждан России;
- сохранение единства образовательного пространства, преемственности ступеней образовательной системы;
- обеспечение равенства и доступности образования при различных стартовых возможностях;
- достижение социальной консолидации и согласия в условиях роста социального, этнического, религиозного и культурного разнообразия нашего общества на основе формирования культурной идентичности и общности всех граждан и народов России;
- формирование универсальных учебных действий, порождающих образ мира и определяющих способность личности к обучению, познанию, сотрудничеству, освоению и преобразованию окружающего мира.

Новые социальные запросы определяют цели образования как общекультурное, личностное и познавательное развитие учащихся, обеспечивающее такую ключевую компетенцию образования как научить учиться (ср. с умением поиска актуальной информации в операционном стиле мышления). Важнейшая задача современной системы образования – формирование совокупности универсальных учебных действий, способствующих формированию компетенции научить учиться, а не только освоению учащимися конкретных предметных знаний и навыков в рамках отдельных дисциплин.

Формирование УУД как цель образовательного процесса определяет его содержание и организацию. Оно происходит в контексте усвоения разных предметных дисциплин. УУД определяют эффективность образовательного процесса – усвоение знаний и умений; формирование образа мира и основных видов компетенций учащегося, в том числе социальной и личностной компетентности. Овладение учащимися универсальными учебными действиями создает возможности самостоятельного успешного усвоения новых знаний, умений и компетентностей, включая организацию усвоения, то есть *умения учиться.*

В набор основных видов УУД включены не только задачи обучения, но и, в первую очередь, задачи воспитания и развития школьников. Ниже приведена построенная специалистами по педагогической психологии [1] классификация УУД. Учителю, привыкшему к использованию прикладных учебных программ и коллекций информационных ресурсов, не трудно увидеть те из них, которые уже поддержаны существующими программными средствами, могут непосредственно применяться в практике формирования УУД. Другая часть УУД менее формализована и может использоваться в качестве технических заданий на проектирование адекватного программного обеспечения. Такое проектирование несомненно должно выполняться совместно психологами, педагогами и программистами-разработчиками, которые владеют языковыми системами программирования, подчиняющимися указанным выше требованиям к учебному программному обеспечению.

- *Личностные* (самоопределение, смыслообразование и действие нравственно-этического оценивания).

Личностные универсальные учебные действия обеспечивают ценностно-смысловую ориентацию учащихся (умение соотносить поступки и события с принятыми этическими принципами, знание мо-

ральных норм и умение выделить нравственный аспект поведения) и ориентацию в социальных ролях и межличностных отношениях.

- действие смыслообразования, то есть установление учащимися связи между целью учебной деятельности и ее мотивом. Ученик должен задаваться вопросом о том, «какое значение, смысл имеет для меня учение», и уметь находить ответ на него;
- действие нравственно-этического оценивания усваиваемого содержания, исходя из социальных и личностных ценностей, обеспечивающее личностный моральный выбор.

• *Регулятивные* (целеобразование, планирование, контроль, коррекция, оценка, прогнозирование):

- *прогнозирование* – предвосхищение результата и уровня усвоения, его временных характеристик;
- *контроль* в форме сличения способа действия и его результата с заданным эталоном с целью обнаружения отклонений и отличий от эталона;
- *коррекция* – внесение необходимых дополнений и корректив в план и способ действия в случае расхождения эталона, реального действия и его продукта;
- *оценка* – выделение и осознание учащимся того, что уже усвоено и что еще подлежит усвоению, осознание качества и уровня усвоения;
- волевая *саморегуляция* как способность к мобилизации сил и энергии; способность к волевому усилию – к выбору в ситуации мотивационного конфликта и преодолению препятствий.

• *Познавательные* (общеучебные, логические и знаково-символические):

- самостоятельное *выделение* и формулирование познавательной *цели*;
- *поиск и выделение* необходимой информации; применение методов информационного поиска, в том числе с помощью компьютерных средств;
- структурирование знаний;
- *выбор* наиболее эффективных *способов решения* задач в зависимости от конкретных условий;
- *рефлексия* способов и условий действия, *контроль и оценка* процесса и результатов деятельности;
- *смысловое чтение* как осмысление цели чтения и выбор вида чтения в зависимости от цели; извлечение необходимой информации из прослушанных текстов различных жанров; определение основной и второстепенной информации;
- *умение* адекватно, осознанно и произвольно *строить речевое высказывание* в устной и письменной речи, передавая содержание текста в соответствии с целью и соблюдая нормы построения текста (соответствие теме, жанру, стилю речи и др.);
- *постановка и формулирование проблемы*, самостоятельное создание алгоритмов деятельности при решении проблем творческого и поискового характера;
- *действие со знаково-символическими средствами* (замещение, кодирование, декодирование, моделирование).

• *Логические*:

- выбор оснований, критериев для сравнения, оценки и классификации объектов;
- синтез как составление целого из частей, в том числе самостоятельно достраивая, восполняя недостающие компоненты;
- подведение под понятия, распознавание объектов;
- установление причинно-следственных связей, построение логической цепи рассуждений, доказательство;
- выявление родо-видовых и ситуативно существенных признаков;
- выдвижение гипотез и их доказательство.

• *Знаково-символические* УУД обеспечивают конкретные способы преобразования учебного материала, представляют действия моделирования, выполняющие функции отображения учебного материала; выделения существенного; отрыва от конкретных ситуативных значений; формирования обобщенных знаний.

• *Коммуникативные*:

- постановка вопросов – инициативное сотрудничество в поиске и сборе информации;

o планирование учебного сотрудничества с учителем и сверстниками – определение цели, функций участников, способов взаимодействия;

o разрешение конфликтов – выявление, идентификация проблемы, поиск и оценка альтернативных способов разрешения конфликта, принятие решения и его реализация;

o управление поведением партнера – контроль, коррекция, оценка действий партнера;

o умение с достаточной полнотой и точностью выражать свои мысли в соответствии с задачами и условиями коммуникации; владение монологической и диалогической формами речи в соответствии с грамматическими и синтаксическими нормами родного языка.

Надпредметный характер УУД и их прямое включение в задачи воспитания и развития детей обосновывают их место в школьной системе образования [1]. Одна из последних новаций начальной школы – Программа развития УУД для предшкольного и начального общего образования.

Основной вывод, касающийся судеб программно-методической системы КуМир, неоднократно упоминаемой в этом тексте и сравниваемой с другими претендентами на роль школьной системы программного обеспечения: КуМир оказался не только начальным импульсом для подобного рода разработок, но и платформой для дальнейшей перспективной информатизации российской системы образования во всем диапазоне непрерывного курса – от пропедевтического, начального до высшего профессионального образования и послевузовского повышения квалификации во всех его формах – общего, дистанционного и дополнительного.

Библиографический список

1. Асмолов, А. Г. Стандарты второго поколения. Формирование универсальных учебных действий в основной школе: от действия к мысли. Система знаний [Текст] / А. Г. Асмолов [и др.]. – М. : Просвещение, 2010. – 160 с.

2. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++ = Object-Oriented Analysis and Design with Applications [Текст] / Г. Буч. – М. : «Бином», 1998.

3. Вирт, Н. Программирование на языке Модула-2 [Текст] / Н. Вирт. – М. : Мир, 1987.

4. Вирт, Н., Йенсен, К. Паскаль. Руководство для пользователя и описание языка [Текст] / Н. Вирт, К. Йенсен. – М. : Финансы и статистика, 1982. – 151 с.

5. Ершов, А. П. Алгоритмический язык в школьном курсе основ информатики и вычислительной техники [Текст] / А. П. Ершов // Микропроцессорные средства и системы. – № 2. – 1985. – С. 48–51.

6. Ершов, А. П. Изучение основ информатики и вычислительной техники [Текст] : методическое пособие / А. П. Ершов [и др.]. – Ч. 1 (девятый класс – 192 с.), Ч. 2 (десятый класс – 176 с.). – М. : Просвещение, 1986.

7. Ершов, А. П. Основы информатики и вычислительной техники [Текст] : учебник для 9–10 классов / А. П. Ершов [и др.]. – М. : Просвещение, 1986. – 288 с.

8. Ершов, А. П. О человеческом и эстетическом факторах в программировании [Текст] / А. П. Ершов // Кибернетика. – № 5. – 1972.

9. Звенигородский, Г. А. Первые уроки программирования [Текст] / Г. А. Звенигородский. – М. : Наука, Библиотечка «Квант», 1985. – 208 с.

10. Звенигородский, Г. А. Школьная информатика (концепции, состояние, перспективы) [Текст] / Г. А. Звенигородский, Ю. А. Первин, А. П. Ершов // ВЦ СО АН СССР. – № 152. – 1979.

11. Лебедев Г. В., Кушниренко, А. Г. 12 лекций о том, для чего нужен школьный курс информатики и как его преподавать [Текст] / Г. В. Лебедев, А. Г. Кушниренко. – «Бином». Лаборатория знаний, 2001.

12. Лебедев, Г. В. Основы информатики и вычислительной техники [Текст] / Г. В. Лебедев, Р. А. Сворень, А. Г. Кушниренко. – М. : Просвещение, 1991. – 224 с.

13. Леонов, А. Г. Информационная культура. Кодирование информации. Информационные модели. 9–10 классы [Текст] / А. Г. Леонов [и др.]. – М. : Дрофа, 1996.

14. Леонов, А. Г. КуМир для младших школьников [Текст] / А. Г. Леонов // Труды Большого Московского семинара по методике раннего обучения информатике. – Т. 2. – М., 2011. – С. 69–73.

15. Леонов, А. Г., Кушниренко, А. Г. А. Кушниренко, А. Леонов. Методика преподавания основ алгоритмизации на базе системы «КуМир». Лекция 1–8 [Текст] / А. Г. Леонов, А. Г. Кушниренко. – Москва, 2010.

16. Леонов, А. Г., Кушниренко, А. Г. КуМир вернулся! [Текст] / А. Г. Леонов, А. Г. Кушниренко // Информатика. 1 сентября. – 2009. – № 6.

17. Лутц, М. Программирование на Python [Текст] / М. Лутц. – 4-е изд. – СПб. : Символ-плюс, 2011.

18. Немов, Р. С. Психология [Текст] : учебное для студентов высших пед. учеб. заведений / Р. С. Немов // Общие основы психологии : в 3 т. – Том 1. – М. : Гуманит. изд. центр ВЛАДОС, 2003. – 688 с.

19. Пейперт, С. Переворот в сознании: Дети, компьютеры и плодотворные идеи [Текст] / С. Пейперт. – М. : Педагогика, 1989.

20. Первин, Ю. А. Большой Московский семинар по методике раннего обучения информатике как продолжение традиций педагогической школы А. П. Ершова [Текст] / Ю. А. Первин // VII Международная конференция памяти академика А. П. Ершова секция «Информатика и образование»: сб. – 15–19 июня 2009. – Новосибирск, Академгородок.

21. Первин, Ю. А. Дистанционное обучение в методике школьной информатики [Текст] / Ю. А. Первин // Международная конференция ИТО-2001. Том IV «Информационные технологии в открытом образовании. Информационные технологии в управляющих системах». – М., 2001. – С. 3.

22. Первин, Ю. А. Методика раннего обучения информатике [Текст] / Ю. А. Первин. – М. : Бином. Лаборатория базовых знаний, 2008. – 288 с.

23. Первин, Ю. А. Раннее обучение информатике – государственная или региональная политика? [Текст] / Ю. А. Первин // Ярославский педагогический вестник. Естественные науки. – № 2. – 2011. – С. 169–174.

24. Первин, Ю. А. Школьная информатика – концепции, состояние, перспективы. Преамбула к ретроспективной публикации [Текст] / Ю. А. Первин // Информатика и образование. – № 1. – 1995.

25. Пышкало, А. М. Методическая система обучения геометрии в начальной школе : авторский доклад по монографии «Методика обучения элементам геометрии в начальных классах» [Текст] / А. М. Пышкало. – М. : Академия пед. наук СССР, 1975. – 60 с.

26. Себеста Р. У. Величайший проект в истории: язык Ada [Текст] / Р. У. Себеста // Основные концепции языков программирования. – М. : Вильямс, 2001. – 672 с.

27. Себеста Р. У. Объектно-ориентированное программирование : язык Smalltalk [Текст] / Р. У. Себеста // Основные концепции языков программирования = Concepts of Programming Languages. – М. : Вильямс, 2001. – 672 с.

28. Страуструп, Б. Дизайн и эволюция языка C++ [Текст] / Б. Страуструп. – СПб., 2006. – 448 с.

29. Уваров, А. Ю. Информатизация школы : вчера, сегодня, завтра [Текст] / А. Ю. Уваров. – М. : «Бином», Лаборатория базовых знаний, 2011. – 482 с.

30. Хуторской, А. В. Современная дидактика [Текст] : учебник для вузов / А. В. Хуторской. – СПб. : БЧВ-Питер, 2001. – 544 с.

31. Шилдт, Г. C# 4.0 : полное руководство = C# 4.0 The Complete Reference [Текст] / Г. Шилдт. – М. : Вильямс, 2010. – 1056 с.

32. Шумилина, Н. Д., Дуванов, А. А. Азбука Роботландии [Текст] / Н. Д. Шумилина, А. А. Дуванов // Вестник Ярославского отделения РАЕН. – № 1. – 2012. – С. 39–46.

33. Эккель, Б. Философия Java = Thinking in Java [Текст] / Б. Эккель. – 3-е изд. – СПб. : Питер, 2003. – 976 с.

34. Эпиктетов, М. Г., Леонов, А. Г. Практикум Е-87 [Текст] / М. Г. Эпиктетов, А. Г. Леонов // ВИНИТИ 4279-В88. – 30.05.88.

35. David Niguidula, Andries Van Dam, David Higuidula, Brookshire Conner Object Oriented Programming in Pascal: A Graphical Approach — Addison-Wesley, 2002.