БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Механико-математический факультет

Кафедра веб-технологий и компьютерного моделирования

СОГЛАСОВАНО
Председатель учебно-методической комиссии механико-математического факультета
В. Г. Кротов
17 мая 2012 года

СОГЛАСОВАНО
Декан механико- математического факультета
Д. Г. Медведев
17 мая 2012 года

Регистрационный № УД 8853

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

Методика преподавания информатики

Для специальности
1-31 03 01-02 Математика (научно-педагогическая деятельность)

Составитель: старший преподаватель Аленский Н.А.

Обсуждено и утверждено
Советом ММФ БГУ *15 мая 2012 года*,
протокол № 7

ОГЛАВЛЕНИЕ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Учебно-методический комплекс (УМК) преследует ***цель:*** оказать посильную помощь студентам в усвоении учебного и нормативного материала, сориентировать в подборе специальной литературы для подготовки к практическим занятиям, направить на развитие навыков самостоятельного решения задач.

Содержание тем УМК структурировано с учетом вопросов, рассматриваемых на лекционных занятиях. Для подготовки к практическим занятиям, промежуточным и итоговым формам контроля следует исходить из содержания Типовой учебной программы по дисциплине для высших учебных заведений по специальности 1-31 03 01-02 «Математика (научно-педагогическая деятельность)», утвержденной Министерством образования Республики Беларусь в 2011г., вопросов и заданий к семинарским занятиям, размещенным в настоящем УМК.

Центральной идеей образования по методике преподавания информатики является необходимость дать студентам тот набор знаний и умений, который будет полезен им при работе с обучающимися на уроках информатики. ***Основной целью*** дисциплины «Методика преподавания информатики» является подготовка студентов к предстоящему преподаванию информатики в учреждениях общего среднего образования. Содержание данной дисциплины отвечает на три основных вопроса методических дисциплин: ***зачем*** учить информатике, ***что*** изучать и ***как*** обучать информатике.

Дисциплина Методика преподавания информатики» посвящена изучению способов и методов обучения информатике в учреждениях общего среднего образования, где компьютерная грамотность становится составляющей профессионального уровня учителя. Будущие выпускники по направлению специальности 1-31 03 01-02 «Математика (научно-педагогическая деятельность)» будут работать учителями математики и информатики. Более того, они должны не только учить школьников, но и способствовать применению компьютерных технологий в учреждениях общего среднего образования. Поэтому рассматриваемая здесь дисциплина играет важную роль в подготовке преподавателей информатики.

В ***первом разделе*** дисциплины рассматриваются теоретические основы методики преподавания информатики, структура и содержание учебной программы, учебников и пособий. Всё многообразие тем школьного предмета «Информатика» можно сгруппировать в отдельные блоки. Примерно третья часть всего отведённого на информатику времени занимают основы алгоритмизации и программирования. Этот материал является самым важным с точки зрения развития умственных способностей школьников и самым сложным как для учителей с точки зрения преподавания, так и для усвоения школьниками. Поэтому это изучается все шесть лет, с шестого по одиннадцатый класс. Во ***втором разделе*** «Методики преподавания информатики» рассматриваются особенности и методика

изучения этого раздела. Наряду с методическими вопросами определенное внимание уделяется здесь более глубокому изучению программирования на примере языка С++, который в основной дисциплине «Методы программирования и информатика» на млаших курсах не изучался. Этим самым параллельно с методикой расширяются знания и умения студентов в области современных информационных технологий. Этот раздел изучается в форме лекций и выполнения индивидуальных заданий по разработке программ. В ***третьем разделе*** рассматривается методика преподавания остальных тем школьного предмета «Информатика».

Каждый обучающийся имеет возможности закрепить лекционный материал на практических занятиях, во время которых в форме докладов с презентациями студенты рассказывают, как вести занятия по определенным темам. При этом значительно возрастает роль самостоятельной работы студентов над дисциплиной, без чего успешное освоение дисциплины представляется маловероятным. Кроме этого для закрепления и расширения навыков программирования студент должен выполнить три индивидуальных задания по программированию. При этом студенту предоставляется возможность выбора системы программирвания и уровень сложности заданий.

Программа составлена с учетом межпредметных связей и программ по смежным дисциплинам.

В соответствии с образовательными стандартами по указанной специальности в результате изучения дисциплины выпускник должен

знать:

теоретические основы методики преподавания информатики;

основные понятия и принципы дидактики информатики;

структурные элементы урока и основные требования к ним;

виды планирования деятельности учителя;

методы обучения школьной информатике;

программы по информатике, структуру и содержание учебных пособий и учебников;

учебно-методическое, техническое и программное обеспечение предмета;

уметь:

разрабатывать и составлять план-конспект урока, факультатива, кружка;

составлять планирование работы учителя;

проводить анализ плана-конспекта урока;

проводить анализ проведения урока, факультатива, кружка;

использовать современное программное обеспечение, электронные учебные пособия и различные педагогические технологии в учебном процессе;

организовать работу школьного кабинета информатики;

применять различные методы контроля и оценки знаний учащихся;

осуществлять внеклассную и внешкольную работу.

Учебная дисциплина рассчитана на один семестр общим объёмом 146 часов, в том числе 68 часов аудиторных (из них: 34 часа лекций и 34 часа семинарских занятий) и 78 часов самостоятельной работы.

КОНСПЕКТЫ ЛЕКЦИЙ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

Конспекты лекций соответствуют типовой программе, содержанию учебного материала и учебно-методической карте. Сокращенный их вариант приведен здесь.

Введение.

Несмотря на то, что информатика изучается в учреждениях среднего образования более четверти века, до сих пор нет систематического учебного пособия по всей дисциплине “Методика преподавания информатики”, соответствующего действующей в настоящее время учебной программе по школьной информатике и современной концепции ее преподавания. На книжном рынке можно найти в основном литературу по частной методике, в которой рассматривается содержание того или иного раздела информатики и методика его преподавания. Одна из основных причин этого — обстановка неуверенности вокруг проблемы места этого предмета в учебном плане школы и его целей и содержания. Последние годы наблюдается относительная стабильность в этих вопросах. Поэтому стало возможным написание этого пособия, которое хотя бы частично восполнит дефицит такого рода литературы.

Лекции написаны на основе разработанной автором и утвержденной в министерстве образования РБ типовой учебной программы для учреждений высшего образования по направлению специальности 1-31 03 01-02 «Математика (научно-педагогическая деятельность)».

Раздел 1. Теоретические основы преподавания информатики

1. Понятие информатики.

Прежде чем говорить о преподавании учебного предмета “Информатика”, попытаемся определить область, именуемую этим словом. Появление и начальное становление информатики как науки относится ко второй половине прошлого века. Терминологические и понятийные трудности, связанные с сущностью самого понятия “информатика” (равно как и производных понятий) не преодолены до сих пор. Его толкование, которое приведено дальше, до настоящего времени нельзя считать установившимся и общепринятым, нет строгого определения этого термина.

Необходимо заметить, что в русском языке примерно с середины 60-х годов такой же термин был связан с узкой областью научно-технической информации и документалистики. Подобное определение связывало информатику с библиотековедением, библиографией, методами поиска информации в массивах документов. Эта дисциплина изучала структуру и общие свойства научной информации, а также закономерности ее создания преобразования, передачи и использования в различных сферах человеческой деятельности. Мы будем рассматривать информатику несколько в другом смысле.

Сначала в конце 40-х годов прошлого века появился и широко использовался термин ***“кибернетика”*** как общая наука об управлении и связи в системах различной природы – искусственных, биологических, социальных. Развиваясь одновременно с прогрессом электронных вычислительных машин, кибернетика со временем превращалась в общую науку о преобразовании информации. Вскоре вслед за появлением термина “кибернетика” в мировой науке стало использоваться англоязычное “Computer Science” (компьютерная наука). Этот термин и сейчас в США, Канаде и некоторых странах латиноамериканского континента достаточно широко распространен вместо “информатика”. Позже, на рубеже 60-х и 70-х гг. XX века, во французском языке было введено слово ***„информатика”*** (*informatique*), образованное, судя по всему, как производное от двух французских слов: ИНФОРмация (*informatione*) и автоМАТИКА

(*avtomatique*) или ИНФОРМАция (*informatione*) и автомаТИКА. Новый термин получил распространение в СССР и странах Западной Европы, а позже в России и странах СНГ, включая и Республику Беларусь. Этим словом вначале обозначали область автоматизированной обработки информации, при которой львиную долю операций над ней выполняют ЭВМ и другие технические средства, а окончательные решения по управлению принимает человек. Были и другие определения. Например, известный советский ученый в этой области, академик А. П. Ершов утверждал, что этот термин вводится в русский язык как название фундаментальной естественной науки, изучающей процессы передачи и обработки информации. При таком толковании информатика оказывается более связанной с общенаучными категориями и ее можно отнести к фундаментальным наукам, что отражает общенаучный характер понятия информации и процессов ее обработки. Многие ученые подчеркивали, что информатика —, это следствие развития ЭВМ.

Область интересов информатики — это структура и общие свойства информации, а также вопросы, связанные с процессами поиска, сбора, хранения, преобразования, передачи и использования информации в различных сферах человеческой деятельности. Обработка огромных объемов и потоков информации немыслима без автоматизации и систем связи. Поэтому ЭВМ и современные информационные и коммуникационные технологии являются и фундаментальным ядром, и материальной базой информатики.

В школьных учебниках приведено, например, следующее определение: „Информатика — наука, исследующая законы и методы переработки и накопления информации”. Далее можно прочитать следующее: „ ... Эта наука позволяет не только понять принципы работы и возможности использования ЭВМ, но и даёт представление о законах и методах представления информации при общении людей и в жизни общества”. Дать определение, отражающее специфику информатики, не просто. Представляется удачным рабочее определение этого термина с помощью трёх компонент:

1) *HARDWARE* – “твёрдые”, аппаратные средства (иногда говорят“железо”). Дословно переводится как аппаратура.
2) *SOFTWARE* – “мягкие”, программные средства, к которым можно и нужно отнести не только программы, но и обрабатываемые данные. Дословно переводится как программное обеспечение.
3) *BRAINWARE* – алгоритмическая, в буквальном переводе – мозговая составляющая (*brain* - мозг). Это не просто алгоритмы обработки данных, что снова вело бы к *SOFT*, а интеллектуальная составляющая, знание, которое воплощено и в программах, и в данных, и в аппаратуре.

Это определение информатики удивительно перекликается с наивно-фундаментальным представлением древних о существовании четырёх стихий – первоэлементов: 1) земля – *HARD*; 2) вода – *SOFT*; данные и программы не постоянны, текучи, изменчивы; 3) воздух – это та сфера, в которой происходят информационные процессы; 4) огонь –*BRAIN*.

Таким образом, анализ различных определений этого термина позволяет выделить следующие аспекты его содержания:

— совокупность средств автоматизированной информационной техники и технологии;
— особая отрасль экономики, включающая всю сферу автоматизированной обработки и использования информации;
— отрасль научного знания, изучающая процессы передачи и обработки информации и средства её автоматизированной обработки (техническое, математическое и программное обеспечение);
— теория научной информации и научно-информационной деятельности с акцентом на средства автоматизации.

2. История преподавания школьной информатики, её современное состояние.

2.1. Основные этапы преподавания информатики.

Необходимо обратить внимание на то, что информатика является одним из немногих общеобразовательных учебных предметов, родившихся в XX веке. Большинство учебных предметов для сферы общего образования начали изучать еще в XIX веке.

Приведем основные ***этапы*** становления школьной информатики в прошлом веке в бывшем Советском Союзе и в нашей Республике Беларусь.

В 50-е годы практиковалось изучение программирования в ряде школ г. Новосибирска. Главным идеологом этого нововведения был академик А.П.Ершов. В 60-е годы осуществлялась подготовка программистов в московских школах с математической специализацией, а позже, в 70-е годы, в Москве, Ленинграде, Новосибирске и некоторых других научных центрах готовили школьников по специальностям, связанным с ЭВМ

В конце 70-ых годов началось массовое производство микро ЭВМ, что расширило области применения и доступность ЭВМ. Началась разработка концепции школьной информатики, главным идеологом которой был академик А.П.Ершов. Были разработаны уникальная система программирования “Школьница” и язык Рапира. В 1982 году министерство просвещения СССР приняло решение о введении калькуляторов в учебный процесс школы. В 1984 году были разработаны основные направления реформы общеобразовательной и профессиональной школы. Одной и основных задач провозгласили обеспечение всеобщей компьютерной грамотности молодежи. Решено ввести в школе новый предмет, который сначала назывался “Основы информатики и вычислительной техники” (ОиивТ). Под руководством А.П.Ершова и В.М.Монахова была разработана ***первая экспериментальная программа***, а на ее основе в предельно сжатые сроки были подготовлены и изданы первые пробные учебники по информатике в двух частях под редакцией А.П.Ершова [1, 2]. С 1 сентября 1985 г. началось преподавание основ информатики и вычислительной техники в массовой школе и подготовка учителей информатики в пединститутах по новым учебным планам.

В 1986 году была опубликована ***вторая программа*** преподавания школьного курса информатики в машинном варианте и объявлен конкурс на создание учебника по ОИиВТ. В соответствии с конкурсной программой разработано первое поколение учебников по информатике авторов В. А. Каймина и др.[3], А. Г. Кушнеренко и др.[4], А. Г. Гейна и др.[5]. С 1986 г. начинает издаваться научно-методический всесоюзный журнал “ИнфО” (“Информатика и образование”). Во второй половине 80-х годов на его страницах была проведена оживленная полемика вокруг содержания предмета ОИиВТ.

Появились отечественные компьютеры, предназначенные для обучения школьников. В 1985 и 1986 гг.проводилась ускоренная подготовка школьных учителей математики и физики к преподаванию нового школьного предмета. На физико-математических факультетах педагогических институтов начала осуществляться подготовка студентов в области информатики (в основном в рамках различных спецкурсов). Но педвузовская подготовка по программированию носила исключительно образовательный характер, она не была ориентирована на преподавание этого предмета школьникам. В 1987 году в педагогических вузах был введен курс МПИиВТ, который был предназначен для подготовки будущих преподавателей по специальности Учитель мат,информатики и выч техники. Анализируя первую программу этого курса, мы убеждаемся, что в результате его изучения будущий учитель должен был понимать значение школьного предмета ОИиВТ в общем образовании, уметь объяснить принципы отбора содержания этого предмета, овладеть основными методическими и дидактическими формами и приемами предмета МПИ, понимать взаимосвязи ОИиВТ с другими школьными предметами, осознавать его определяющую роль в решении компьютеризации образования.

В первой половине 90-х годов продолжалась оживленная дискуссия по вопросу преподавания информатики в Республике Беларусь, были разработаны новые концепции информатизации образования[6, 7] и новое поколение программ предмета и учебников белорусских авторов, таких, как Ю.А.Быкадоров [8], В.М. Котов, О. И. Мельников, А.Т.Кузнецов, А.И.Павловский и др [8 – 11]. Были переделаны и переизданы первые учебники российских

авторов. Все более очевидной становилась нецелесообразность (и недостаточность) обучения информатике только на старшей ступени школы. К началу 90-х годов постепенно в Республике Беларусь, как и в других странах СНГ, начинает складываться новая структура обучения информатике в общем среднем образовании. В ***1994 г.*** в РБ основной курс информатики перенесли в базовую школу в 8 – 9 классы. Этот двухгодичный курс был обязательным для всех типов учебных заведений. В 10 -11 классах курс продолжался на углубленном и повышенном уровнях, в специальных классах, школах, лицеях и гимназиях.

В 1998 г. Советом министров РБ была утверждена программа «Информатизация системы образования», в 2000 г. коллегией Министерства образования принят документ «Основные направления развития информатизации общего среднего образования», в 2003 г. — «Электронная Беларусь». В 2002 г. при Министерстве образования.РБ создан координационный совет по проблемам информатизации образования, переименованный в 2005г. в «Совет по проблемам информатизации образования при Министерстве образования РБ». В 2007 г. принята программа «Комплексная информатизация системы образования РБ на 2007 – 2010годы». Самый совершенный педагог не сможет дать самому старательному и способному ученику запас знаний на всю жизнь. Теперь поставлена другая цель — научить школьников самостоятельно добывать информацию из различных источников, производить ее отбор, обработку, хранение и творческое использование, что невозможно без развитой информационной культуры учащихся. Основная ответственность за ее формирование ложится на плечи учителей информатики.

Начиная с 2004/05 учебного года, в рамках реализации реформы общеобразовательной школы обязательный курс информатики изучается не два, а четыре года в 7(6), 8(7), 9(8) и 10(9) классах *(в скобках указана старая нумерация классов согласно 11-летнему сроку обучения)*. Разработана сквозная четырёхлетняя программа этого предмета. Подготовлены и изданы четыре учебника заведующего кафедрой информатики Академии Последипломного Образования Пупцева А. Е. и соответствующие методические пособия для учителей. Начиная с 2006/2007 учебного кода, информатика изучается ещё на один год раньше, с 6-го класса.

2.2. Формирование содержания образования в области информатики.

Общие требования к содержанию образования и принципы его формирования указаны в соответствующих нормативных документах, изучаются в других дисциплинах и здесь не рассматриваются. В образовании можно выделить три компоненты: обучение, воспитание и развитие. Центральное положение среди них занимает образование. Обратим внимание на две группы основных факторов, традиционно находящихся в диалектическом противоречии.

1. *Научность и практичность.* Содержание учебного предмета информатики должно идти от науки, т.е. не противоречить её современному состоянию и быть методологически цельным. Изучение предмета информатики должно давать такой уровень фундаментальных познаний учащихся, который мог бы обеспечить их подготовку к будущей профессиональной деятельности.
2. *Доступность и общеобразовательность*. Включаемый в предмет материал должен быть посилен основной массе учащихся, отвечать уровню их умственного развития и имеющемуся запасу знаний, умений и навыков. Этот учебный курс должен содержать все наиболее общезначимые, общеобразовательные и общекультурные сведения из соответствующих разделов науки информатики.

Другими словами, школьный предмет информатики, с одной стороны, должен быть современным, отвечать усложняющимся требованиям науки и практики, а с другой — быть простым и доступным для изучения. Совмещение этих двух противоречивых требований было и остаётся наиболее сложной методической задачей. Поэтому не удивительно, что формирование содержания образования по информатике — сложный и противоречивый процесс, который охватывает период от начала 60-х годов прошлого века до настоящего времени. При этом надо заметить, что фактическое состояние теоретических разработок этой проблемы и экспериментальных достижений в этой области слабо коррелируется с фактическим состоянием модели содержания этого предмета, которая находится на вооружении практического учителя информатики.

В основу разработки **первой программы** школьного предмета ОИиВТ (1985г.) были положены три базовых понятия: информация, алгоритм, ЭВМ. Они и составили концептуальную основу первой версии содержания этого школьного предмета. Ставились следующие задачи нового предмета: овладение основными умениями алгоритмизации; систематизация и завершение алгоритмической линии курса алгебры; формирование представлений о возможности автоматизации выполнения алгоритма; усиления прикладной направленности алгоритмической линии, заключающееся в конкретной реализации алгоритмов решения задач с помощью ЭВМ; формирование представления об этапах решения задач на ЭВМ; ознакомление с основами современной вычислительной техники на примере рассмотрения общих принципов работы микрокомпьютера; ознакомление с основными сферами применения вычислительной техники, ее ролью в развитии общества.

Курс ОИиВТ преподавался в двух старших классах средней школы. В 9-м классе на изучение этого предмета отводилось 34 часа (1 час в неделю). В 10-м классе предлагались два варианта. Первый из них, полный курс объемом 68 часов, рекомендовался для школ, располагающих вычислительными машинами или имеющих возможность организовать занятия школьников на ВЦ других организаций. Краткий курс объемом 34 часа предназначался для школ, не имеющих такой возможности. Теоретическая часть для 10-го класса была одинаковой для обоих вариантов, а отличались объем и содержание практической части . Для школ, имеющих доступ к ЭВМ, дополнительные 34 часа рекомендовалось использовать для решения задач в машинном варианте и отработки навыков применения компьютеров и его программного обеспечения.

Кроме определения содержания для информатики, как ни для какого другого предмета, важным и непростым является вопрос построения оптимальной последовательности изучения, соответствующей логике науки и уровню развития учащихся. С учетом этого курс ОИиВТ в ***9-м классе*** складывался из следующих тем: введение (2часа); алгоритмы и алгоритмический язык (6 ч); алгоритмы работы с величинами (10 ч); построение алгоритмов для решения задач(16 ч.).

В ***10-м классе*** предлагались по программе следующие темы: принципы устройства и работы ЭВМ (12 ч.); знакомство с программированием (16 ч.); роль ЭВМ в современном обществе, перспективы развития вычислительной техники (2 ч.); экскурсия на вычислительный центр (4 ч.).

Анализ первой программы показывает, что был принят алгоритмический подход в изучении информатики. Больше всего часов (48 из 68) отводилось на построение алгоритмов и разработку на их основе программ для ЭВМ. Избыточность алгоритмизации и программирования подвергалась резкой критике. Но в оправдание разработчиков программы и соответствующих учебников надо учитывать реальное состояние отечественной практики на то время, возможности оснащения школ компьютерами, готовность учительских кадров и другие обстоятельства. Этим можно объяснить и то, что преподавание информатики на начальном этапе носило в основном безмашинный характер.

Основным средством описания алгоритмов являлся специально разработанный под руководством академика Ершова А. П**.** ***учебный алгоритмический русскоязычный язык,*** частично напоминающий *Pascal*. Теперь можно уверенно сказать, что он сыграл большую роль в истории преподавания информатики и полностью себя оправдал. Он обладал рядом привлекательных свойств, которые и объясняют, почему именно ему было отдано предпочтение перед распространенными в то время официальными языками программирования (например, Бейсик, Алгол, *Pascal*). Язык использовал русскую или национальную лексику, что не создавало дополнительных трудностей, не имеющих никакого отношения к сути предмета. В отличие, например, от языка Бейсик, школьный язык построен на современных идеях структурного программирования, что наилучшим образом соответствовало операционному мышлению человека. Язык не зависел от ЭВМ, в нем не было деталей, связанных с устройством ЭВМ. Это позволяло основное внимание сосредоточить на алгоритмической сути решаемых задач.

Вторая учебная программа, как и первая, (1986г.) была рассчитана на обучение этому предмету в двух старших классах в объеме 102 часа. Между этими первыми двумя вариантами программ существенных различий нет. Основная особенность ее в том, что эта программа представляла собой машинный вариант и была ориентирована на обучение информатике в

условиях активной работы школьников на ЭВМ в кабинете вычислительной техники. Поэтому больше времени отводилось на практическую работу. Важным элементом этой программы являлся впервые официально объявленный примерный перечень программного обеспечения в поддержку курса ОИиВТ. Он по существу повторял все разделы программы и состоял из компонентов, которые в условиях объявленного конкурса следовало рассматривать как приглашение к разработке. Позже большинство из них действительно были разработаны, причем в нескольких вариантах. Особое положение в перечне программных средств занимал интерпретатор учебного языка, первая версия которого уже фактически действовала к моменту объявления конкурса.

Концепция содержания, заложенная в программе машинного варианта, была практически реализована в нескольких подготовленных на ее основе учебных пособиях. Учебные книги по информатике авторов В. А. Каймина и др.[3], А. Г. Кушнеренко и др.[4], А. Г. Гейна и др.[5] получили широкое распространение в школах. Появление этих учебников вызвало поток жесткой критики, которая относилась не столько к самим книгам, сколько к программе, их породившей. Критиковались в основном гипертрофированная линия алгоритмизации и программирования и слабое продвижение в развитии фундаментальных основ школьной информатики.

Необходимость введения нового предмета объяснялась широким внедрением вычислительной техники и особенно персональных компьютеров во все области человеческой деятельности, в том числе в сферу быта и систему народного образования. Бурное развитие микроэлектроники и на её базе производства мини- и микро-ЭВМ, а позже персональных компьютеров, совершенствование их программного обеспечения и средств общения с ними способствовало компьютеризации школы и других средних учебных заведений и введению предмета ОИиВТ. Но в то же время освоение нового школьного предмета в первые годы во времена Советского Союза существенно тормозилось общим состоянием процессов информатизации общества, недостаточным развитием материальной базы и современных информационных технологий.

В настоящее время общие вопросы методики преподавания информатики в основном понятны и сформулированы, ответы на них в значительной степени найдены. Детально разработана и методика преподавания ряда конкретных тем. В то же время постоянная открытость информатики новому ставит более сложную и важную задачу – научить студента самостоятельно разрабатывать методику, научить методическому творчеству, передать ему опыт подобной творческой деятельности.

3. Дисциплина «Методика преподавания информатики» в системе педагогических знаний

3.1. Цели и задачи учебной дисциплины

Информатика в учреждениях общего среднего образования разных видов преподаётся более четверти века. Однако методика преподавания этого предмета остается актуальной и в настоящее время. Вопросы и задачи в новой педагогической науке продолжают возникать и требуют своего разрешения в виде теоретического обоснования и опытной проверки. Поэтому теория и методика обучения информатике в настоящее время продолжает развиваться.

МПИ формируется не на пустом месте. Исследование целей и содержания общего кибернетического образования, накопленный практический опыт преподавания учащимся элементов кибернетики, основ алгоритмизации и программирования, элементов логики, вычислительной и дискретной математики имеют, как видим из предыдущей темы, более чем полувековую историю. Будучи фундаментальным разделом педагогической науки, методика информатики опирается в своем развитии на философию, педагогику, психологию, информатику, а также обобщенный практический опыт средней школы.

Одновременно с введением информатики в школьное образование была проведена массовая переподготовка уже работающих учителей и выпускников педвузов, а в 1987 году разработаны

новые учебные планы для систематической подготовки учителей информатики. Вскоре появилось и первое учебное пособие по этой вузовской дисциплине [12] , которое позже несколько раз переиздавалось[13]. Методику преподавания информатики (МПИ) как новую дисциплину начали преподавать в пединститутах в 1987/88 учебном году. На тот момент существовало лишь фрагментарное видение предмета, отраженное в ряде статей, докладов, дискуссий. Поэтому преподавание курса проводилось паралельно с его разработкой. Курс МПИ вначале строился как методика решения задач на алгоритмизацию в безмашинном варианте.

Основной целью дисциплины «Методика преподавания информатики» является подготовка студентов к предстоящему преподаванию информатики в учреждениях общего среднего образования. Перед методикой преподавания информатики, как и перед всякой предметной школьной методикой, ставятся следующие три основные вопросы: 1) зачем учить информатике? 2) что изучать? 3) как обучать информатике? Поэтому з***адачи*** дисциплины следующие:

- обеспечение глубокого изучения студентами научных и психолого-педагогических основ структуры и содержания предмета информатики учреждений общего среднего образования, понимание методических идей, заложенных в них;
- формирование знаний о месте и значении школьной информатики в образовании школьника, о связи этого предмета с другими изучаемыми в школе предметами;
- освоение содержания учебников и программ по информатике;
- освоение классических и новых методов обучения и организационных форм занятий с учетом особенностей информатики;
- выработка умения организовать работу школьников в современном компьютерном классе;
- обучение студентов ориентированию в постоянно изменяющемся парке вычислительной техники и программном обеспечении и применению их в учебном процессе.

Согласно концепции информатизации РБ, МПИ не может и не должна сводиться к преподаванию только информатики. Она обеспечивает реализацию следующей группы целей этой концепции:

- изучение новых областей знаний, связанных с информатикой;
- изучение современных информационных технологий, приобретение навыков их использования;
- развитие и закрепление логико-алгоритмического стиля мышления, изучение современных методов программирования и;

В результате изучения курса будущий учитель должен подготовиться не только к преподаванию информатики, но и к полноценной работе в компьютеризованной школе.

3.2. Место дисциплины, связи с другими дисциплинами.

Дисциплина «Методика преподавания информатики» посвящена изучению способов и методов обучения информатике в учреждениях общего среднего образования, где компьютерная грамотность становится составляющей профессионального уровня учителя. Будущие выпускники по направлению специальности 1-31 03 01-02 «Математика (научно-педагогическая деятельность)» будут работать учителями математики и информатики. Более того, они должны не только учить школьников, но и способствовать применению компьютерных технологий в учреждениях общего среднего образования. Поэтому рассматриваемая здесь дисциплина играет важную роль в подготовке преподавателей информатики.

Основная особенность курса МПИ – связь с другими, прежде всего методического цикла, предметами. С методикой преподавания математики она связана, можно сказать, генетически. Понятие алгоритма пришло из математики. Возможна такая аналогия: множество/элемент = алгоритм/команда. С другой стороны, многие доказательства в математике имеют алгоритмическую структуру, и существует задача научить выявлять эту алгоритмическую составляющую в доказательствах. Время от времени возникала и продолжает возникать тенденция объединить информатику, например, с математикой. В лучшем варианте, у сильного учителя, могут раскрыться межпредметные связи. Но сама информатика может проиграть в этом случае в

силу математизации задач и связанного с этим неизбежного роста их трудности. Это противоречит тенденции разгрузки начала изучения информатики как раз от математики (решение квадратных уравнений). При этом могут быть утрачены и связи информатики с другими дисциплинами, и предметы будут по-прежнему изучаться обособленно. Не связано с математикой и умение пользоваться готовыми ПС.

Сходство МПИ с методиками преподавания физики, химии проявляется в склонности информатики к опыту, эксперименту. Действительно, запуск программы на компьютере — это своеобразный эксперимент. С точки зрения физики компьютер — это прибор, поведение которого можно исследовать. У методики физики можно позаимствовать методы выполнения подобных опытов. Область пересечения интересов МПИ с психологией – это прежде всего проблемы общения. Особая форма общения – компьютерная игра. Методика информатики уже предполагает и использует органическое включение игровых элементов в учебную деятельность с компьютером. Можно, например, рассматривать отладку программы как игру с компьютером – “кто умнее”. Своеобразной является связь с методикой изучения иностранного языка. Перевод, например, алгоритма на конкретный формальный язык (С, *Pascal*, алгоритмический язык и др.) – это вполне языковая, речевая деятельность, иногда и непростая проблема.

Методика преподавания информатики связана с дисциплиной «Методы программирования и информатика», в которой изучался подробно язык *Pascal*. Чтобы расширить знания студентов по современным технологиям программирования, методику преподавания самого большого раздела информатики «Основы алгоритмизации и программирования» предлагается изучать на примере родственного объектно-ориентированного языка С++ с использованием современной визуальной системы *Builder*. Дисциплина «Методика преподавания информатики» связана также с дисциплинами «Методика преподавания математики», «Педагогика», «Психология». Кроме этого, имеется связь с учебной вычислительной практикой.

Будучи первой дисциплиной методического цикла, связанного с ЭВМ, МПИ играет корректирующую роль в компьютеризации образования в целом. Проблемы информатизации образования лежат не только и не столько в области техники. Намного важнее наличие хороших программных средств. Но даже полное обеспечение системы образования мощными современными компьютерами с дружественным программным обеспечением ещё не означает автоматически, что проблемы информатизации образования решены. Большую роль играет степень психологической готовности и потребность педагогов и управленцев в пересмотре средств и способов своей деятельности в связи с использованием ЭВМ.

3.3. Требования к усвоению дисциплины

Требования к уровню усвоения содержания дисциплины «Методика преподавания информатики» установлены образовательным стандартом высшего образования по направлению специальности **1-31 03 01-02** «Математика (научно-педагогическая деятельность)». В нем определены общенаучные знания и умения, которыми должен овладеть студент. В результате изучения дисциплины обучаемый должен:

знать:

теоретические основы методики преподавания информатики;

основные понятия и принципы дидактики информатики;

структурные элементы урока и основные требования к ним;
виды планирования деятельности учителя;
методы обучения школьной информатике;
программы по информатике, структуру и содержание учебных пособий и учебников;

учебно-методическое, техническое и программное обеспечение предмета;

уметь:

разрабатывать и составлять план-конспект урока, факультатива, кружка;
планировать работу учителя;
проводить анализ плана-конспекта урока;

проводить анализ проведения урока, факультатива, кружка;

использовать современное учебно-методическое, техническое и программное обеспечение, электронные учебные пособия и различные педагогические технологии в учебном процессе;

организовать работу школьного кабинета информатики;

применять различные методы контроля и оценки знаний учащихся;

осуществлять внеклассную и внешкольную работу.

3.4. Объём дисциплины

Учебная дисциплина «Методика преподавания информатики» предназначена для студентов направления специальности **1-31 03 01-02** «Математика (научно-педагогическая деятельность)». Она рассчитана на один семестр общим объёмом 146 часов, в том числе 68 часов аудиторных (из них: 34 часа лекций и 34 часа семинарских занятий) и 78 часов самостоятельной работы.

3.5. Средства диагностики и критерии оценок

Для контроля качества образования используются следующие средства диагностики:

индивидуальные типовые задания;
тесты по отдельным разделам и дисциплине в целом;
письменные контрольные работы;
устный опрос во время занятий;
коллоквиумы;
составление рефератов;
выступление студентов на семинарах по разработанным ими темам;

письменный или устный экзамен.

Текущий контроль знаний выполняется по каждому разделу предложенной здесь программы.

В качестве итогового контроля по завершению изучения первого раздела рекомендуется электронное тестирование. Кроме этого, или вместо тестирования по усмотрению кафедры первый раздел можно проверить с помощью классического устного опроса.

По второму разделу необходимо выполнить индивидуальные задания, темы которых соответствуют разделу «Основы алгоритмизации и программирования» школьной информатики, например, 1)массивы, 2)строки, 3)графика. По каждому из них необходимо отладить программу, используя дополнительно изученную современную систему программирования, отличную от той, которая изучалась на младших курсах в дисциплине «Методы программирования и информатика». При оценке каждого из заданий учитывается уровень сложности, своевременность его сдачи, выполнение требований к заданиям, эффективность алгоритма, полноту тестирования.

Второй, а также третий разделы проверяются в форме разработки, презентации и обсуждения на практических занятиях методики изучения конкретной темы школьной информатики. Студент должен предложить методы изучения и формы занятий, показать, как используется персональный компьютер и другие технические средства, и в качестве примера разработать план конспект одного урока. По некоторым темам этого раздела предлагается выполнить индивидуальные задания (*Excel*, СУБД, графический и текстовый редактор, Интернет, электронная почта). ***Итоговый результат*** работы студента: учебно-методический комплекс (УМК) по определенной теме курса, который включает:

одну или несколько лекций;

подробный план проведения лабораторных (практических) занятий;

дидактический материал: упражнения, задачи, задания с решениями и комментариями; или без них;

контрольные вопросы;

тесты с ответами и комментариями для самостоятельной работы или без них.

По результатам текущего контроля выставляется общая текущая отметка.

Итоговый контроль осуществляется в форме зачета и экзамена. Для зачета необходимо получить положительную отметку по каждому разделу. Экзамен предлагается проводить в два этапа: письменная контрольная работа с элементами тестирования и выполнение заданий на компьютере по изучаемым в школе темам. В результате получается экзаменационная отметка. Оценка знаний студента на экзамене производится по 10-балльной шкале.

3.6. Другие особенности дисциплины.

Компьютер – достаточно дидактически мощное средство, которое заставляет пересматривать и содержание, и формы, и методы обучения, всю систему в конкретных дисциплинах. Вот некоторые примеры влияния компьютера на курс математики:

1) Традиционное повышенное внимание школьной математики к логарифмической функции связано с упрощением вычислений вручную при переходе от умножения чисел к сложению их логарифмов. При использовании компьютеров исчезает основание для приоритета логарифмов.

2) Исследование графика функции может быть начато с его быстрого построения на экране и резко упрощается. Если функция в точке не существует, компьютер выдаёт сообщение об ошибке типа деление на нуль.

3) С помощью компьютера легче понять смысл итерационных методов для решения тех или других задач (вычисление квадратного корня, решение уравнений и их систем и др.).

Оптимальным решением является интеграция информатики с другими дисциплинами, но не в учебных часах, а на уровне решаемых задач, системы знаний. Можно ожидать взаимного обогащения понятий, разнообразия связей и как следствие прочности и полезности знаний в целом.

В идеале возможности применения ЭВМ должны практически раскрывать учителя-предметники. Пока этот процесс идет крайне медленно, курс школьной информатики и кабинет ВТ остаются в определенной изоляции. Не будем останавливаться на очевидных причинах этого. В оправдание этого можно сказать, что сфера образования вообще инерционна в большей степени, чем общество. Потери от игнорирования компьютеров при образовании школьников слишком велики.

Сильные внутри предметные связи информатики порождают нетривиальную задачу поиска оптимального порядка изучения материала в соответствии с принципами последовательности и доступности изложения. При буквальном понимании последовательности предполагается, что учебный материал выстраивается в логическую цепочку или может быть представлен в виде дерева, где нет логических кругов, и повторение идет лишь как закрепление материала. В информатике это, увы, или к счастью, невозможно. Сильные внутри предметные связи между различными темами не позволяют “выпрямить материал”, как это имеет место, например, при изучении истории, литературы и некоторых других предметов.

В информатике нельзя некоторые темы (например, оператор цикла) изучить сразу полностью, за один или несколько подряд идущих уроков. Особенно это касается раздела “Основы алгоритмизации и программирования”. Поэтому еще академиком А.П.Ершовым была предложена реализация принципа последовательности в форме ***цикличности***. Это означает, что сначала изучается некоторый элемент в простейшем, наиболее распространенном на практике виде. Затем через относительно немалый промежуток времени (возможно, год и больше) это понятие повторяется, обогащаясь новыми возможностями, дополнительными особенностями. Если для других дисциплин это желательный путь, то для информатики многие разделы или темы по-другому изучить просто невозможно.

Особенностью МПИ является ***динамический, изменяющийся характер самой информатики*** и как науки, и как учебного предмета, ее нестабильность, постоянное развитие и совершенствование как технических, так и особенно программных средств. В этих условиях вынужденным и плодотворным решением является максимальная опора на результаты общей дидактики, на конкретные методики близких дисциплин – математики и физики. Необходимо руководствоваться принципом единого классического образования школьников в области информатики, не зависящего, или в малой степени зависящего от типов компьютеров и ПО. Надо

больше внимания уделять наиболее общим, фундаментальным знаниям. Следует по возможности минимизировать или полностью избегать машинно-зависимых знаний и умений, которые могут оказаться бесполезными, а, возможно, и вредными, при работе на другом типе компьютера, с другой операционной системой или другой версией языка.

Основная особенность МПИ – связь предмета с ***использованием компьютера***, который обладает несравненно большей “самостоятельностью”, чем любой другой прибор, например, в физике. Общение с компьютером требует развития особых черт мышления и поведения, адекватных методов обучения и воспитания. Достаточно сказать, что цена ошибки в обычном тексте и в программе для компьютера различна. Компьютер является таким средством обучения, мощность, эффективность и дидактические возможности которого потенциально превосходят то, что доступно иным средствам обучения. Это средство не может не вести к обновлению методик преподавания и других дисциплин.

При изучении МПИ на лабораторных занятиях студент “играет” три роли и соответственно выполняет три вида деятельности:

1) роль учащегося – для лучшего понимания материала “изнутри” с позиции ученика и освоения тех учебников, которые появились в школе после поступления студента в вуз;

2) роль учителя – разработка инструкций, заданий, вопросов для контроля и других материалов для учащегося, управление с рабочего места для преподавателя работой класса;

3) роль методиста-предметника – разработка методических материалов, прежде всего, для себя как учителя, а фактически и для других учителей.

Формы и методы обучения зависят от изучаемых разделов, на которые разделен учебный материал.

Первый раздел является в основном теоретическим и изучается в форме лекций, бесед, «круглых» столов. Структуру и содержание учебных пособий и учебников предлагается изучать кроме лекций в форме подготовки студентами и обсуждения на семинарских занятиях докладов и рефератов.

Всё многообразие тем школьного предмета «Информатика» можно сгруппировать в отдельные блоки. Примерно третья часть всего отведённого на информатику времени занимают основы алгоритмизации и программирования, которые изучаются все шесть лет. Поэтому во ***втором разделе*** программы рассматриваются особенности и методика изучения этого раздела, самого сложного для школьников и учителей и важного с точки зрения развития мышления. Наряду с методическими вопросами определенное внимание уделяется здесь более глубокому изучению программирования. Этим самым параллельно с методикой расширяются знания и умения студентов в области современных информационных технологий. Этот раздел изучается в форме лекций и выполнения индивидуальных заданий по разработке программ.

В ***третьем разделе*** рассматривается методика преподавания остальных тем. Этот материал предлагается изучать в форме лекций, а также разработки и обсуждения плана-конспекта проведения нескольких уроков по конкретной теме.

При проведении занятий наряду с традиционными методами обучения следует использовать и инновационные: лекцию-визуализацию, работу с электронным дидактическим комплексом, видеотренинг, дистанционные занятия, работу с тренажером, компьютерное тестирование. Необходимо также разумно сочетать групповую и индивидуальную работу.

Содержание и формы ***контролируемой самостоятельной работы*** студентов (СРС) разрабатываются деканатом и кафедрами вуза в соответствии с Положением о СРС, разработанным в учреждении высшего образования. СРС осуществляется в виде аудиторных и внеаудиторных форм. На основании бюджета времени в соответствии с образовательным стандартом, учебным планом учебной дисциплины устанавливаются виды, объем и содержание заданий по СРС. При проведении самостоятельной работы используется электронный вариант лекций, лабораторных занятий и другие материалы.

Для оценки качества самостоятельной работы осуществляется контроль за ее выполнением. Формы контроля устанавливаются учреждением высшего образования: собеседование, защита индивидуальных заданий, коллоквиум, контрольная работа, тестирование, проверка рефератов,

зачет (итоговый или по отдельной теме), устный и (или) письменный экзамен.

При изучении второго и третьего разделов с целью активизации самостоятельной работы студентов рекомендуется использовать метод проектов, что позволяет реализовать индивидуальный подход к обучению. В ходе работы над проектом студенты активно работают с различными источниками и системами обработки информации. В результате каждый студент по определенной теме создает аудио-видео-материалы в виде презентаций, конспектов проведения уроков и других форм. Такая организация учебного процесса способствует развитию как информационной, так и профессионально-методической компетентности будущего учителя информатики.

4. ЦЕЛИ ПРЕПОДАВАНИЯ ИНФОРМАТИКИ

4.1. Значение предмета в учреждениях общего среднего образования

Рассмотрим, что означает **школьная информатика**. Как известно, одной из областей человеческой деятельности, испытывающей активное влияние информатики, является образование. Ветвь информатики, обслуживающая проблемы учреждений общего среднего образования, получила название школьной информатики. Она занимается исследованием и разработкой программного, технического, учебно-методического и организационного обеспечения применения ЭВМ в школьном учебном процессе.

Программное (или математическое) обеспечение поддерживает информационную, управляющую и обучающую системы средней школы. Оно включает в себя программистские средства для проектирования и сопровождения таких систем, а также средства общения с ними, ориентированные на школьников, учителей и работников аппарата управления. В области *технического обеспечения* школьная информатика имеет своей целью экономически обосновать выбор технических средств для сопровождения учебно-воспитательного процесса школы, определить параметры оборудования школьных компьютерных классов. Учебно-методическое обеспечение школьной информатики состоит в разработке учебных программ, методических пособий и учебников по школьному предмету информатики и другим предметам, при преподавании которых планируется использование средств информатики. К проблемам *организационного обеспечения* можно отнести следующие: организационно-технические мероприятия по обеспечению и последующему сопровождению технической азы школьной информатики; организация разработки, тиражирования и доставки педагогических программных средств; подготовка и переподготовка кадров школьных учителей, способных нести в массовую школу информатику как новую научную дисциплину, как инструмент совершенствования преподавания других школьных предметов, как стиль мышления.

Школьный учебный предмет информатики не может включать всего того многообразия сведений, которые составляют содержание быстро развивающейся науки информатики. В то же время это предмет, выполняя общеобразовательные функции, должен отражать в себя наиболее важные, фундаментальные понятия и сведения, раскрывающие существо науки, вооружать учащихся знаниями, умениями и навыками, необходимыми для изучения основ других наук и будущей жизни и деятельности в современном информационном обществе. На процесс формирования школьного предмета информатики сказывается чрезвычайно малая временная дистанция между возникновением информатики как самостоятельной отрасли науки и включением в практику общеобразовательной школы соответствующего ей нового учебного предмета. По этой причине определение его содержания является непростой задачей. Более того, долгое время не являлось абсолютно бесспорным, как в целом изучать информатику в общеобразовательной школе — в отдельном предмете или целесообразнее рассредоточить учебный материал по информатике среди ряда учебных дисциплин. Вопрос заключался в следующем: чего должно быть больше — того, что должно составить отдельный предмет, или того, что может или должно быть неразрывно связано с содержанием и методикой изучения остальных школьных предметов. Как известно, на сегодняшний день остановились на следующей дидактической формуле: всякий базовый компонент общего образования, каким является и

информатика, включается в содержание образования двояко — в виде отдельного учебного предмета и в виде “вкраплений” в другие учебные предметы.

Школьная информатика должна выполнять роль межпредметной, интегрирующей дисциплины. Если ее рассматривать просто как локальный предмет, не связанный с другими, если ее изолировать в кабинете ВТ, то такой путь является тупиковым, сковывающим творчество и учителей, и школьников.

4.2. Уровни работы с компьютером

Чтобы конкретизировать цели обучения, рассмотрим возможные уровни работы любого человека, не обязательно преподавателя, с компьютером.

Пассивный пользователь, не работая непосредственно с ЭВМ, пользуется компьютерными “благами”: получает в банкоматах зарплату, покупает билет на поезд, осуществляет поиск информации в Интернете, пользуется электронной почтой и т.д. Минимальные сведения о компьютерах получает из средств массовой информации. Профессиональные навыки работы на ЭВМ отсутствуют.

Активный (параметрический) пользователь работает с готовыми программами, подставляя в них свои параметры: добавляет базы данных, осуществляет её корректировку и поиск нужной информации и т.п. Он ориентируется в типах данных, алгоритмов, умеет писать, рисовать с помощью ЭВМ. Такой пользователь способен длительное время обходиться без помощи программиста. С позиции школьной информатики это наиболее массовая фигура в настоящем и ближайшем будущем.

Программирующий пользователь способен внести небольшие изменения в готовую программу. Например, он может вставить свою формулу в готовую универсальную программу построения графика функции. Он может программировать небольшие задачи в простых средах (формулы в электронных таблицах типа *Excel*), способен грамотно поставить задачу программисту, выбрать необходимое готовое программное обеспечение для решения своих задач.

Парапрограммист (“настройщик”) работает на языках сверхвысокого уровня: в СУБД, электронных таблицах и др. Основное его умение: с одной стороны формализовать прикладные задачи пользователя и довести их до состояния работающей программы, и наоборот, видеть конкретные применения готовых программ. Он в состоянии настроить готовые программные продукты общего назначения на конкретные нужды пользователя. Такой специалист перебрасывает мостик между реальностью и компьютером, что требует системно-комбинаторного, прикладного образа мышления. От него в конечном счёте зависит, дойдет ли разработка программиста до практического применения, сможет ли пользователь с ней работать. Такой “настройщик” пока необходим, так как рост дружественности программ происходит в первую очередь по отношению к программистам.

Программист разрабатывает средства для парапрограммистов и для пользователей. То есть предполагается, что он разрабатывает, прежде всего, общее программное обеспечение. Но нельзя согласиться с автором книги [14], который утверждает, что программист прикладные задачи обычно не решает. А кто тогда их программирует, если нет готовых программ? Программист использует достаточно мощные и сложные для массового освоения системы и языки программирования: Delphi, Paskal, Builder, C++, Visual C++, Java и др. Его понятийный аппарат достаточно формализован.

Наконец, *системный программист* обеспечивает эффективность работы на всех предыдущих уровнях.

С позиций школьной информатики и сегодняшних реалий видно, что наиболее массовая фигура в настоящем и ближайшем будущем — параметрический пользователь. Но эффективность его деятельности резко возрастает, если он подтягивается до уровня программирующего пользователя, так как уменьшается его зависимость от программистов. Подготовка всех учащихся до этого уровня — посильная для школы и социально необходимая задача.

4.3. Цели преподавания школьной информатики как единство образования, развития и воспитания.

Определение целей преподавания конкретного предмета — наиболее трудная часть практической методики. Цели обучения в общей дидактике рассматриваются в единстве образования, развития и воспитания. Информатика подчеркивает и практический аспект — подготовку молодого человека к полноценной жизни в компьютеризованном обществе. Общие цели обучения информатике определяются с учетом ее особенностей как науки, ее роли и места в системе наук, в жизни современного общества. Рассмотрим, как общие цели, характерные для школы в целом, могут быть отнесены к образованию школьников в области информатики.

Образовательная и развивающая цель обучения информатике в школе — дать каждому школьнику фундаментальные начальные знания науки информатики и на этой основе раскрыть учащимся значение информационных процессов, информационных технологий и вычислительной техники в формировании научной картины мира и в развитии современного общества. Изучение этого предмета призвано также вооружить учащихся базовыми умениями и навыками, которые необходимы для прочного и сознательного усвоения не только информатики, а и основ других наук, изучаемых в школе. Все это должно также существенно влиять на общее умственное развитие учащихся, развитие их мышления и творческих способностей.

Общее образование при изучении информатики заключается в формировании компьютерного мировоззрения, понимании принципов работы, возможностей и ограничений ЭВМ и достигается лишь на основе системных знаний, выводящих за рамки прагматичного, потребительского подхода к ЭВМ. Его признаки следующие:

1) понимание общих принципов работы ЭВМ (а не частностей!);
2) представление о сути деятельности программиста и других специалистов в области информационных технологий;
3) знания о типах информации и способах ее обработки;
4) представление об алгоритмах, их типах, способах их записи, в том числе о программах;
5) знания о возможностях и ограничениях автоматизации мышления.

Цели развития в основном сводятся к формированию двух взаимодополняющих стилей мышления.

Логико-алгоритмическое мышление проявляется в умении строить логические утверждения о свойствах данных и запросы к поисковым системам; мыслить индуктивно и дедуктивно при анализе своих затруднений; формализовать свои намерения вплоть до записи на алгоритмическом языке или языке программирования.

Признаками *системно - комбинаторного мышления* являются: видение предметов и явлений в целостности, взаимосвязях; умение строить несколько взаимодополняющих точек зрения на одну и ту же проблему; умение комбинировать понятийные и орудийные средства из различных дисциплин при построении моделей. Например, с точки зрения алгебры функция есть соответствие, с точки зрения геометрии – кривая, а с точки зрения информатики – алгоритм вычисления результатов по заданным аргументам. Особенностью системно-комбинаторного мышления является также и то, что работа с компьютером — это своеобразная языковая деятельность. Как пользователи, так и особенно программисты излагают свои мысли и намерения в формализованном виде, оперируя правилами специальных языков. Заметим, что системно–комбинаторный стиль мышления, к сожалению, в основном чужд практике средней и высшей школы. Предметы расчленены на высшие и низшие, теоретические и практические. Не всегда видна связь между ними. Поэтому и студенту, и школьнику сквозь обилие частностей не видно интегральных, целостных знаний.

При использовании компьютера, особенно при разработке программ, у школьника развивается умение и склонность к наблюдению за собственным мышлением. В компьютерной деятельности, как ни в одной другой, можно проследить за следующей цепочкой: «что я хотел (алгоритм) – что сделал (текст программы) – что получилось и почему (результат выполнения программы)». Компьютер овеществляет мышление, делает наглядными наши намерения.

Практическая цель школьной информатики — внести вклад в трудовую и технологическую подготовку учащихся, т. е. вооружить их теми знаниями, умениями и навыками, которые могли бы обеспечить подготовку к трудовой деятельности после окончания школы. Это означает, что школьный курс информатики должен быть практически ориентированным — обучать школьника

работе на компьютере и использованию их программных средств. В профориентации предмет информатики должен давать учащимся сведения о профессиях, непосредственно связанных с ЭВМ, а также различными приложениями изучаемых в школе предметов, опирающихся на использование ЭВМ. Практические цели предусматривают также и бытовой аспект — готовить молодых людей к грамотному использованию компьютерной техники в повседневной жизни, в быту. Таким образом, прикладной компонент образования проявляется в умении извлекать практическую пользу из общения с ЭВМ (запросы к поисковым системам, Интернет, небольшие расчеты и т.п.), что определяет уровень и качество жизни.

Воспитательная цель рассматриваемого предмета обеспечивается, прежде всего, тем мощным воздействием на ученика, которое оказывает осознание возможностей и роли вычислительной техники в развитии общества и цивилизации в целом. При изучении информатики формируется культура умственного труда и такие важные качества, как умение планировать свою работу и рационально ее выполнять, настойчивость и целеустремленность, творческая активность и самостоятельность, ответственность и трудолюбие, четкость и лаконичность мышления, изложения и написания и другие полезные характеристики личности. Сами по себе эти и некоторые другие черты личности не сформируются, требуется с самого начала кропотливая работа учителя в этом направлении.

В результате реализации целей воспитания формируются следующие черты и качества личности:

- объективное отношение к результатам компьютерных вычислений, т. е. критичность и самокритичность в оценке своих способностей;
- бережное отношение к технике и к информации, как к своей так и чужой; этическое, нравственное неприятие компьютерного вандализма и вирусотворчества;
- личная ответственность за результаты своей работы на компьютере, за возможные свои ошибки, а также за решения, принимаемые на основе компьютерных данных;
- потребность и умение работать в коллективе при решении больших задач, забота о пользователе продуктов своего труда.

Названные выше основные цели обучения информатике взаимосвязаны между собой, ни одна из них не может быть достигнута изолированно друг от друга.

4.4. Компьютерная грамотность, образованность, культура

В соответствии с первой программой предмета, школьные учителя должны были сформировать всеобщую компьютерную грамотность старшеклассников. Академик А. П. Ершов вводит понятие **Компьютерная грамотность** как владение навыками решения задач с помощью ЭВМ, умение планировать действия и предвидеть их последствия, понимание основных идей информатики, представление о роли информационных технологий в жизни общества. При этом он подчеркивал, что основы компьютерной грамотности учащихся 80-х годов перерастут в информационную культуру общества первых десятилетий 21 века.

Впервые на нормативном уровне понятие **Информационная культура учащегося** было введено в 1986 году во второй программе курса ОИиВТ и, по сути, являлось расширением понятия Компьютерная грамотность. А.П. Ершов считал, что информац. Культуру развивает, главным образом, курс школьной информатики. Этот курс должен формировать:

навыки грамотной постановки задач и формализованного их описания для решения с помощью ЭВМ;

знания основных алгоритмических структур и умение применять их для разработки алгоритмов решения задач;

элементарные навыки составления программ для ЭВМ на одном из языков программирования высокого уровня;

понимание устройства и функционирования ЭВМ;

навыки квалифицированного использования основных типов информационных систем и пакетов прикладных программ общего назначения.

В качестве исходной характеристики конкретных целей обучения информатике в учреждениях среднего образования уже в первой программе предмета ОИиВТ была объявлена ***компьютерная грамотность*** (КГ) учащихся. Это понятие формировалось вместе с новым

предметом. Если сравнивать с обычной грамотностью в привычном понимании, то под компьютерной грамотностью можно понимать умение считать, читать, писать, рисовать, искать информацию с помощью ЭВМ. Такое определение было дано в одном из первых учебников по школьной информатике. Признаком высокой, сформировавшейся грамотности является самостоятельность и эффективность работы с применением ЭВМ. Это первая характеристика качества обучения школьника информатике.

В настоящее время можно выделить следующие группы компонентов КГ школьников:

1. Понятие об алгоритме, его свойствах, способах описания. Представление о программе, как средстве и методе описания алгоритма для ЭВМ, знание основ программирования на одном из языков. С одной стороны, подготовка программистов не является целью общеобразовательной школы, но учащиеся должны понимать основные принципы алгоритмизации и программирования и приобрести начальные навыки составления и отладки самостоятельных программ. Это основывается на алгоритмической культуре, которая может быть сформирована на простых и наглядных средствах. Сначала это были так называемые бытовые алгоритмы, а позже использовались чертежник, робот, черепашка и др.
2. Принцип действия и устройство ЭВМ и ее основных элементов. Здесь можно выделить две основные составляющие: структура персонального компьютера и функции его основных устройств: физические основы и принципы действия основных элементов компьютера.
3. Практические навыки работы на ЭВМ, то есть умение обращаться (или, на языке информатиков — общаться) с ЭВМ: умение запустить и правильно выключить компьютер: умение запустить готовые программы; овладение клавиатурой, печатью и другими устройствами. Сюда же позже были отнесены и навыки работы с простейшими сервисными программами: текстовый и графический редакторы, электронные таблицы, системы управления базами данных, игровые, обучающие программы, а в последние годы электронная почта и Интернет и др.
4. Области применение и роль компьютеров в производстве и других отраслях деятельности человека. Формирование этой составляющей компьютерной грамотности не является задачей исключительно курса информатики. Эти вопросы целесообразно по возможности раскрывать учащимся в процессе практического использования компьютера для решения различных задач в других учебных предметах. Школьный компьютер можно использовать для вычислительных работ по математике, физике, химии, анализе данных учебного эксперимента при проведении лабораторных работ по этим предметам. В гуманитарных предметах персональный компьютер можно использовать для разработки информационной системы по географии, истории и другим гуманитарным школьным предметам.

Анализ перечисленных компонентов показывает, что понятие КГ явилось результатом расширения понятия алгоритмической культуры (АК) учащихся путем добавления компонентов, связанных с устройством и использованием ЭВМ. Зародившись на начальном этапе преподавания информатики, понятие КГ и по настоящее время активно и успешно работает в методике этого предмета. Сокращенно изложенную выше структуру КГ можно обозначить совокупностью четырех ключевых слов: программирование, устройство, общение, применение. Усиленное акцентирование внимания на том или другом из них может приводить к изменению цели преподавания предмета. Если, например, преимущество отдать компоненту *общение*, то курс информатики станет пользовательским. Если доминировать начнет *программирование*, то цели курса сведутся к подготовке программистов. Эти два направления в смысле целей и содержания информатики долго вели между собой борьбу. В последние годы найден разумный компромисс между этими двумя крайностями.

Вторая характеристика качества образования школьника по информатике, а, значит, и одна из ее целей — *информационная культура (ИК).* Всего лишь один год, прошедший со времени публикации первой программы по ОИиВТ (1985 г.), показал, что цели преподавания информатики в школе не могут ограничиваться только рамками КГ и что в ближайшей перспективе потребуется развитие и расширение самих целей. Наряду с понятием КГ в новой программе (1986 г.) впервые на нормативном уровне появляется новое понятие: «информационная культура» учащихся. Оно было образовано путем добавления к КГ новых компонент, относящихся в основном к моделированию. В курсе информатики необходимо было сформировать навыки формализованного описания поставленных задач, элементарные знания о методах математического моделирования поставленных задач и умение строить простую математическую

модель. Кроме этого, по модели надо было уметь составить алгоритм и реализовать его в виде программы для ЭВМ. В ИК были также расширены некоторые прежние компоненты КГ. Например, компонент КГ, отождествляемый в прежней редакции с развитием упрощенных навыков общения с компьютером, в новой системе целей связывается уже с навыками квалифицированного использования основных типов современных информационных систем и понимания основных принципов, лежащих в основе их функционирования.

Схематически эволюцию целей образования школьников в области информатики схематически можно обозначить следующим образом: ***АК —>КГ —> ИК —> ?*** Знак вопроса означает, что введенное вместе со второй программой понятие ИК не могло (да и не предполагало) замыкаться на перечне объявленных в то время компонентов. Это противоречило бы динамическому характеру целей образования. Будет развиваться и уточняться как состав, так и наполнение компонентов ИК учащихся, что является отражением требования к общему школьному образованию соответствовать современному состоянию развития науки и практики. Например, с развитием каналов связи и компьютерных коммуникаций возникла потребность включения в содержание понятия ИК представлений о коммуникационных технологиях.

В учебном пособии [14] Бочкин А.И. под ИК понимает, прежде всего, этику использования компьютера в контексте общечеловеческих ценностей. Этот термин рассматривается в стиле религиозных заповедей, веками регулирующих нормальное человеческое поведение.

- „Не убий” чужую информацию случайно или умышленно. Отсюда неприятие вирусотворчества и небрежности.
- „Не укради” чужую программу или данные. Минимальное требование культуры здесь — не извлекать для себя прибыль из чужого труда и бесплатно полученное бесплатно же и передавать, причем обязательно с согласия автора. Копирование без разрешения автора и тем более взлом защиты от копирования есть присвоение чужой собственности.
- „Не сотвори себе кумира”, что означает отказ от компьютерного фанатизма, от излишнего зацикливания на программах, от попыток сужения реального мира до компьютерной среды. Особенно это актуально в связи с широким развитием сети Internet.
- „Не лжесвидетельствуй”, т.е. не обсуждай те программы и данные, которые недостаточно хорошо знаешь.
- «Возлюби ближнего как самого себя»: помогай товарищам и коллегам, не занимай лишнего места в общей памяти, удаляя ненужные устаревшие файлы и их копии.

В ИК есть и эстетическое содержание – умение видеть, ценить и создавать красивое: оригинальный, понятный, эффективный алгоритм, красивое оформление сценария работы программы и её результатов (окна, кнопки и другие элементы экрана, использование графики, цвета). Частью информационной культуры является также самодисциплина и аккуратность: наличие комментариев, структурная запись текстов алгоритмов, стройная система в обозначениях, продуманная организация хранения информации во внешней памяти и др.

В отличие от российских авторов учебного пособия [13] в книге [14] Бочкин А.И. выделяет еще одну характеристику качества обучения информатике. Он считает, что сложилась традиция трактовать КГ излишне широко, расширяя смысл вплоть до системной грамотности. Видимо, нужна иная, более точная категория — *компьютерная образованность.* Ее основные признаки вытекают из обычных представлений об образованном человеке, взятых в контексте информатики. К ним можно отнести:

1) регулярное чтение литературы по информатике;
2) широкий кругозор в компьютерной области, понимание возможностей и ограничений ЭВМ;
3) ориентирование в многообразии программных средств (ПС): знать их назначение, качественные характеристики, умение выбрать оптимальные ПС для конкретной работы, наличие и ведение собственной библиотеки ПС.

Школа не может обеспечить полностью такую образованность, но заложить основы и сформировать потребность в ней — должна!

Таким образом, можно сделать следующие ***выводы***:

1) цели практического образования служат формированию КГ;
2) цели общего образования и умственного развития связаны с компьютерной образованностью;
3) цели воспитания служат формированию информационной культуры.

Заметим, что между уровнем освоения работы на ЭВМ, грамотностью, образованностью и культурой не всегда существует однозначная зависимость. Например, пользователь, не умеющий программировать, может быть весьма эрудированным, и наоборот, фанатичный программист может ничего не знать кроме, скажем, своего языка или используемой системы. С этой точки зрения создатель вирусов грамотен и, возможно, образован, но культуры, разумеется, у него нет. Если рост уровня компетентности в информатике не сопровождается повышением уровня культуры, последствия могут быть весьма негативными. Это особенно актуально в связи с развитием и широким использованием компьютерных сетей, включая и Интернет. Поэтому все три указанные цели компьютерного обучения требуют определенных усилий педагогов и не достигаются автоматически при простом усвоении материала.

Проведенный анализ системы целей позволяет предложить ответы на некоторые дискуссионные вопросы преподавания информатики:

1) Нужно ли всем школьникам изучать программирование? — Да, но лишь в той мере, в какой это обеспечивает подготовку программирующих пользователей, формирует логико-алгоритмическое мышление. Чрезмерное увлечение этим чревато потерей времени, нужного для повышения уровня компьютерной образованности. Признаком профессионализма для программиста является избегание программирования в случаях, когда можно использовать готовые программы. С другой стороны, квалифицированный пользователь не должен бояться программирования небольших, нестандартных задач.
2) Какой язык изучать или использовать? — Неважно какой. Главное, чтобы с его помощью можно было изучить основные типы алгоритмов, основы современных методов и технологий программирования. Не обязательно использовать простую систему программирования. Можно, например, выбрать профессиональный сложный язык C++, но некоторые его достаточно непростые возможности не изучать.
3) Нужно ли будущему пользователю знать устройство ЭВМ? — Да, в той мере, в какой это способствует компьютерной образованности и успеху в практическом использовании компьютера.
4) Сложным, носящим межпредметный характер, является следующий вопрос: нужно ли в информатике заниматься моделированием? – С одной стороны, не так важно, где моделировать, на уроке информатики или в других предметах (биологии, физике). Именно через моделирование формируется системно-комбинаторное мышление, умение решать реальные задачи из разных областей, формируется научная картина мира. В тоже время ввиду сложности этой темы, вокруг которой было очень много споров и дискуссий, начиная с 2009 года она просто выброшена из основного курса школьной информатики.

5. ПРИНЦИПЫ ДИДАКТИКИ И ПРЕПОДАВАНИЕ ИНФОРМАТИКИ

При отборе содержания, средств и методов преподавания курса на помощь учителя информатики должны прийти общие принципы дидактики.

5.1. Принцип научности

Затронем важный с точки зрения дидактики вопрос об объекте и предмете науки информатики. Объект — это область действительности, на которую направлена деятельность исследователя, а предмет — это посредствующее звено между субъектом и объектом исследования. Другими словами, представители разных наук видят один и тот же объект по-разному, в свете разных задач, в разных системах понятий, выделяют в объекте разные стороны, связи и отношения. Например, обучение по-разному рассматривают методист, психолог,

кибернетик и другие специалисты.

Применительно к информатике ее предмет образуется на основе широких областей своих приложений, а объект — на основе общих закономерностей, свойственных любым информационным процессам в природе и обществе. Предмет информатики определяется многообразием ее приложений, которые охватывают различные виды деятельности: производство, управление, науку, образование, медицину, торговлю и другие. Информатика изучает то общее, что свойственно всем многочисленным разновидностям конкретных информационных процессов, которые и являются объектом информатики. Принцип научности требует, чтобы в содержании образования нашли отражение новейшие достижения соответствующей области знаний с адаптацией на познавательные возможности учащихся. Эта задача упрощается тем, что пока нет деления на высшую информатику и низшую, как это имеет место, например, в математике. Любое понятие из “большой ” информатики находит свои аналогии в школьной информатике.

Фундаментальными являются понятия “информация”, “алгоритм”, “исполнитель”. Исполнитель выполняет несколько функций:

- это дидактическое средство для придания процессу исполнения алгоритмов наглядности (Робот, Чертежник и др.);
- это понятие, позволяющее с единых позиций трактовать многие вопросы: Робот — исполнитель над графикой; Редактор — исполнитель над текстами; Операционная Система — исполнитель для файлов; Принтер – исполнитель для листа бумаги и т.д.

Компьютерную модель всякого исполнителя можно понимать в терминах объект но – ориентированного программирования как модуль или объект. Научность обучения подразумевает также современность методов обучения, что применительно к информатике означает, прежде всего, моделирование в широком смысле, а также исследовательскую деятельность учащегося.

5.2. Сознательность усвоения и деятельности

В традиционном смысле сознательность — это полное понимание учащимся содержания и средств своей деятельности. Но компьютер, будучи сложным устройством, заранее вынуждает ограничивать эту сознательность целями обучения. Можно ли за ограниченное время полно, детально рассказать обо всех процессах, происходящих в компьютере, например, при нажатии клавиши “Enter”. Все это можно и не знать. Конструктор ЭВМ, программист и пользователь имеют различные точки зрения на такое событие. Надо сформировать у учащегося взаимодополняющих точек зрения на подобные ситуации, что и дает многостороннее знание.Здесь решающее значение имеет уровень знаний учителя и, самое главное, умение отобрать, ограничить материал. Таким невольным вынужденным отбором содержания можно объяснить нередкий парадоксальный успех среднего, не очень эрудированного студента на педпрактике.

5.3. Доступность и наглядность содержания

Принцип доступности реализуется через выделение уровней обучения и работы за компьютером. Например, самый низкий уровень: простое использование готового программного обеспечения. Это доступно всем учащимся. В разделе “алгоритмизация и программирование ” предлагаемые задачи можно разделить на 3 уровня, чтобы школьник мог выбрать доступную для себя задачу (на 5 баллов, среднего уровня на 7 баллов, сложную задачу на 9 баллов). На 10 баллов надо предложить нестандартную оригинальную задачу.

Наглядность достигается с помощью:

- блок – схем алгоритмов;
- структурной (с отступами) записи текстов программ (алгоритмов);
- использования цвета;
- демонстрации готовой программы, процесса ее выполнения.

5.4. Активность и самостоятельность

Активность учащегося при изучении информатики имеет следующую особенность. Если

при изучении других дисциплин педагог работает в прямом контакте с обучаемыми, видит их реакцию, сам реагирует, то здесь возможна работа ученика один на один с компьютером. В информатике активность учащегося является не только целью, но и необходимым условием успешности обучения.

Формы проявления активности различны, например, самоконтроль; контроль над работой товарища, оказание реальной ему помощи; модификация готовых и разработка собственных алгоритмов и программ. Активность следует из интереса к предмету, но учителю важно четко сформулировать, что является контролируемым результатом обучения, то есть, что нужно «сдать». Активизировать работу, особенно в начале обучения, можно, практикуя работу учащихся по двое за одним компьютером, даже если их достаточное количество в классе. В таком случае уменьшается неуверенность, возникает диалог, происходит взаимное обучение. Но при этом возникает вопрос: вместе дать возможность работать двум сильным ученикам, хорошо успевающий должен быть в паре со слабым, или оба должны быть с невысокими способностями?

Самостоятельность учащегося, как цель и условие успешного изучения информатики, следует за активностью. Возможные этапы нарастания самостоятельности: от полного управления учителем, через дозированную помощь к самоуправлению познавательной деятельности с помощью компьютера. Полностью самостоятельность реализуется при переходе к творческой деятельности, при выполнении индивидуальных заданий. Когда ученик обращается за помощью к учителю, к товарищу — это, понятно, проявление не самостоятельности, а активности.

Значение самостоятельности: она ведет к большей результативности обучения, учит находить выходы из затруднительных, иногда, казалось бы, тупиковых ситуаций. Особенно это касается отладки программ, поиска ошибок. Самостоятельность учит также искать нужную литературу, пользоваться ею, а также учит использовать компьютерные средства помощи (Help).

5.5. Прочность и системность знаний

Прочность знаний тесно связана с их системностью, основанной на поиске, построении и учете внутри- и межпредметных связей и ассоциаций. Обычная, например, для зоологии структура изучаемого предмета в виде дерева обязательна: в информатике должна быть дополнена «паутиной» связей между листьями-понятиями, их взаимным обогащением в их комбинациях. Содержание информатики и как науки, и как учебного предмета в виде одного дерева представить невозможно. Скорее всего, это лес с переплетенными кронами, растущий из таких фундаментальных для этой дисциплины понятий, как «информация», «алгоритм», «исполнитель» и т.д.

5.6. Индивидуализация и коллективность обучения

Индивидуализация и коллективность обучения дополняют друг друга, особенно в информатике. В этом отношении компьютер — дидактически двойственный инструмент. Тиражируя обучающие или готовые программы, он способствует организации единообразной, фронтальной групповой деятельности, но способ работы учащегося с программой — все же “один на один”, со своим индивидуальным темпом, своими путями преодоления трудностей.

Индивидуализация возможна:

- через выполнение индивидуальных, а не общих, одинаковых для всех, заданий, классифицированных по уровню сложности;
- через гибкую настройку обучающей программы (например, на тип мышления обучаемого);
- через освобождение времени педагога для индивидуальной работы при автоматизации рутинной части педагогического труда.

При работе учащихся вдвоем за компьютером, что может оказаться весьма полезным, могут сложиться устойчивые отношения типа “работник—указчик“. Поэтому время от времени учащихся надо менять местами и ролями.

5.7. Эффективность учебной деятельности

Эффективность предполагает оптимизацию усилий педагога и ученика для обеспечения наибольшего их КПД, отношение результат/усилие. Это требует, прежде всего, отсутствия

постороннего содержания в их деятельности. Например, блок-схемы, наглядные и удобные для малых задач, могут превратить информатику в черчение при более сложных алгоритмах. Паскаль и другие учебные языки, безусловно, эффективны для изучения алгоритмизации или программирования, но мало приспособлены для вычисления выражений типа 693/13. здесь лучше использовать калькулятор.

При дефиците машинного времени, а также для сохранения зрения и здоровья эффективность работы за дисплеем должна обеспечиваться предварительной подготовкой учащегося, изучением инструкций. Эффективным является непосредственное редактирование текстов программ (особенно *Cut, Copy, Paste*), отладка программ (пошаговое выполнение, точки остановки, окно *Watch* и др.)

5.8. Связь теории и практики

Путь от теории, от приобретения знаний до их применения, то есть до практики, в информатике очень короткий, короче даже чем, например, на уроках труда. Учащийся может решить задачу, полезную учителю, классу или школе. Например, он может создать небольшую базу данных «успеваемость», или по школьному оборудованию, для библиотеки, и т.п. Понятия теории и практики в информатике обнаруживают полную параллельность с общенаучными категориями. Как и в «большой» науке, теория есть средство прогноза, предсказания или объяснения свойств, поведения компьютерного мира, а практика — средство проверки теории и, с другой стороны, источник гипотез для неё.

Теоретическим методом выглядит доказательство алгоритма без ЭВМ. Но такой способ приемлем для небольших по объему несложных алгоритмов. Эффективнее, конечно, тестирование программы на практике с использованием компьютера. Но в тоже время минимальная система тестов, которая охватывала бы максимум возможных ситуаций (ветвей программы) строится даже для конкретной задачи теоретически.

Другой аспект — это связь теории и практики при изучении непосредственно информатики. Как и в других науках, теория объясняет или предсказывает результат опыта (запуск алгоритмов на компьютере), а практика (работа на компьютере) служит средством проверки теории и источником гипотез, например, о поведении программы. Эти два вида деятельности тесно переплетаются на уровне мышления учащегося.

Комбинированным методом работы является отладка программ. Здесь тесно переплетаются практическое проявление ошибки при компиляции и (или) выполнении программ, теоретический анализ ее причин, их поиск с привлечением определенных знаний из соответствующей области и практическая проверка на компьютере внесённых исправлений.

6. МЕТОДЫ ОБУЧЕНИЯ ИНФОРМАТИКЕ

6.1 Особенность методов и их реализация на практике

Метод – это способ деятельности, направленный на достижения определенной цели. В общей дидактике, как известно, следует различать следующие понятия:

- учение – учебная деятельность учащегося, например, отладка программ самостоятельно;
- преподавание – деятельность учителя, например, разработка инструкций, текстов, заданий для индивидуальной работы и т.д.;
- обучение – их совместная деятельность, например, защита учеником индивидуальных заданий.

Особенность информатики в этом смысле заключается в следующем. Компьютер как посредник между учителем и учеником увеличивает объем относительно независимых видов деятельности учащегося и учителя и сокращает объем их совместной работы. Это связано с тем, что целью курса является в идеале независимость обучаемого при работе с ЭВМ от педагога, а затем и от программиста. Но это не означает выведение педагога от учебной деятельности. Его опыт и знания никогда не заменят никакие обучающие программы.

Как отмечалось раньше, учащийся, работающий за компьютером, более самостоятельно, имеет локальные собственные цели.

Метод преподавания в общем смысле можно понимать как метод управления познавательной деятельностью учащегося. *Метод обучения* можно понимать как метод познания обучаемым действительности в специально созданной учебной ситуации.

Известно множество оснований для классификации методов обучения:

- по содержанию обучения;
- по способу восприятия информации: словесные, практические и др.;
- по способу получения знаний (теория и(или) практика);
- по способу реализации обратной связи (контроль учителя, самоконтроль, контроль с помощью компьютера).

6.2 Традиционные и словесно-фронтальные методы

Рассказ. Характерный признак рассказа – яркое, занимательное, эмоциональное повествование. Оно выполняется, как правило, в следующей последовательности: вступление, изложение, заключение. Знания при этом приобретаются теоретически, логика рассуждений здесь нестрогая: аналогии, примеры. Обратная связь при этом по степени внимания. Можно попросить учащегося воспроизвести услышанное.

Основное назначение: передача конкретных сведений. Подходящие для такого метода темы: области применения ЭВМ, история развития вычислительной техники и компьютерных технологий, вирусы. Это можно поручить и «сильным» ученикам, так как рассказ имеет смысл использовать для изучения несложных тем, которые носят информационный характер.

Лекция. Основные черты этого метода: объемность материала, сложность, логичность, взаимосвязь отдельных вопросов. Поэтому критерий выбора тем для лекции: трудность в изучении; отсутствие хорошей, понятной и доступной для школьника литературы и др. Поэтому подходящими темами могут быть, например, такие: устройство ЭВМ, системы счисления и представления информации в памяти ЭВМ, вспомогательные алгоритмы. Примерами неудачных тем являются, например, следующие: клавиатура, подготовка и редактирование текстов в текстовом редакторе и др. Дидактическая функция аналогична, как и для рассказов: передача знаний, сведений. Логика рассуждений — в основном дедуктивная, строгая, но могут использоваться и такие умственные операции, как аналогия и сравнение. Обратная связь во время лекции обычно слабая и чаще всего отсроченная, так как в виду сложности материала надо дать время на обдумывание.

Классический вид лекции как диктовка, пересказ основных положений, фактов для информатики мало эффективен. Необходимо, чтобы лекция носила проблемный характер, по возможности использовала раздаточный материал (например, перечень и вид стандартных процедур и функций по определенной теме), компьютерные средства демонстрации.

Беседа. Особенность этого метода – системы управляющих вопросов, ведущих обучаемого к заранее намеченной преподавателем цели. Как правило, обсуждаются наиболее важные вопросы до и после практики. Для беседы можно выбирать также темы, по которым учащиеся уже что-нибудь знают: начальные сведения по ПЭВМ, понятия алгоритма. В тоже время учащиеся получают новые знания, систематизируют то, что было им известно. Поэтому дидактическая функция беседы — приобретения и упорядочивание знаний. Метод носит в основном теоретический характер, но может опираться на опередившую практику.

Инструктаж. Черты этого метода – краткость, повелительная форма. Логика почти не привлекается, развернутых рассуждений нет. Примеры тем: техника безопасности в компьютерном кабинете, начальные сведения о работе с клавиатурой, порядок работы со стандартным или другим программным обеспечением. Дидактическая функция — усвоение сведений и некоторых стандартных способов действия. Инструктаж может сопровождаться показом образца действия. Основная цель - подготовка к практике. Обратная связь выполняется, прежде всего, через практику, а также через устный контроль.

6.3. Мыслительные операции и работа на ЭВМ

Рассмотрим, как реализуются умственные логические операции в связи с ЭВМ. Если учитель

учтет их, ему будет проще реализовать конкретную комбинацию методов обучения.

Анализ и синтез в информатике имеют следующую специфику. Например, целью анализа может быть выяснение причин ошибки. При этом процесс выполнения программы расчленятся на шаги. Если поиск ошибки выполняется в диалоге с ЭВМ, путем трассировки программы, то это «грубый» анализ – разложение. Если ошибка ищется за столом, то это мысленное выделение частей.

Примером сложного синтеза может являться создание учащимся модели компьютерной среды, (например, электронные таблицы) на основе понимания отдельных команд и наблюдений за реакцией среды на эти команды.

Сравнение имеет место, когда сопоставляются, например, близкие элементы языка программирования (операторы *Если* и *Выбор*, на *Pascale* операторы *while* и *repeat*, функции и процедуры, статические и динамические массивы и т.п.). Эту же операцию можно использовать при изучении устройства ЭВМ (сравнение оперативной и внешней памяти), какой-нибудь системы (перемещения и копирование текста в *Word* или ячеек в *Excel*) и т.п.

Сравнение – достаточно мощный дидактический прием. С его помощью легче вводить новые понятия. С помощью этого метода сначала указываем на сходство нового элемента с уже изученным, а затем на различие. Используя этот прием, можно более объективно проверить знания учащихся. Одно дело, рассказать в отдельности, например, про массив, и отдельно про файл с типом на языке *Pascal*. Учащийся должен более глубоко знать материал, если необходимо сравнить эти два типа данных: найти общие характеристики и различия, особенности по отношению друг к другу.

Классификация как мыслительная операция имеет место при освоении учащимся достаточно большого объема материала. Особенно актуально это в настоящее время, когда необходимо осваивать большое количество стандартных средств (классов и их методов, самостоятельных функций, компонент и свойств в системах типа *Delphi* и т.п.). К сожалению, в книгах часто такого рода элементы описываются в алфавитном порядке. Такая классификация вполне уместна не при изучении, а при использовании такого пособия в качестве справочного при условии, что материал уже освоен. При изучении, особенно на начальном этапе, удобнее, если такие элементы будут классифицированы по назначению, по важности, по частоте использования на практике, или по другим критериям (например, строковые функции в языке C++ по типам возвращаемых значений).

Обобщение, индукция и дедукция. Ярким классическим приемом обобщения в информатике является переход от представления об отдельных повторениях похожих действий к команде цикла.

Развернутой формой обобщения является индукция. Имеется в виду не математическая, а неполная индукция. Пример. Учащийся впервые сел за компьютер и в текстовом редакторе нажал клавишу → «стрелка вправо». Если спросить, что произойдет, если нажать эту клавишу, ответ будет однозначным и даже недоуменным. Но не принят во внимание случай, когда курсор уже находится у правого края. В этом примере ход рассуждений учащегося, скорее всего, неосознанный, представляет собой неполную индукцию, причем очень неосторожную.

Несмотря на, казалось бы, неправильный результат в этом примере, такая схема часто используется и ведет к верным умозаключениям, подтверждаются практикой по следующим причинам.

Комбинаторно неисчерпаемое множество возможных состояний компьютера устроено очень регулярно: однотипны ячейки памяти, схемы, регистры, структуры алгоритмов (так называемые базовые управляющие структуры: следование, ветвление и повторение).

Современные программные средства, особенно работающие в среде *Windows*, всегда придерживаются определенных стандартов (меню, кнопки и другие элементы приложения, панель инструментов, строка состояний и т.д.). При разработке своих программ также надо соблюдать общепринятые требования.

Индукция проявляется и как обобщение команд: тело цикла получается из серий частных

повторяющихся действий при конкретных значениях параметра.

Индуктивным является и умозаключение о правильности программы на основании конечного числа тестов. Несмотря на не строгость этого вывода, так программирует весь мир. Популярно в информатике вводить новые понятия, отталкиваясь от примеров. Приведем несколько имен (идентификаторов), например, в языке *Pascal*: *MyMin, R1, r2, Cout_of_Max*. И наоборот, запишем имена, которые не являются идентификаторами: Число, 2max, R+1,.. . Анализ этих примеров позволяет сформулировать общеизвестные правила построения идентификаторов. Следует индуктивное обобщение: считать сделанные утверждения истинными не для данных примеров, а вообще для всех имен.

Для успешности такого индуктивного подхода система примеров должна обладать рядом свойств:

- представительностью: примеры представляют наиболее важные случаи;
- минимальностью: примеры не должны повторять одинаковые правила;
- ортогональности: примеры должны быть независимы;
- распространенностью.

В идеале система примеров должна минимально представлять все частные случаи.

Но не рекомендуется оглашать индуктивное заключение типа «Ученик имеет плохие способности», если он не справился с некоторым заданием. И, наоборот, даже единственный, частный успех является основанием для высказывания гипотезы о том, что он вообще способный.

Роль дедукции в школьной информатике скромнее, чем индукции, в силу опережающей практики возможности тут же проверять верность умозаключения опытом, прямо на компьютере.

Дедуктивным является, например, поиск ошибки в программе, если она ищется пошаговым исполнением. Но и в этом случае заключительный этап понимания, «схватывания» ошибки происходит очень быстро, практически неосознанно. Есть все основания говорить в этом случае о так называемой «свернутой» дедукции, идущей от теста, текста программы, дидактического и ожидаемого результата. Заметим, что рекомендуется вести протокол ошибок, особенно сложных, над которыми долго думали, на которые потратили много времени. Это полезно делать как ученику, так и учителю.

Аналогия и перенос. Аналогия – это осторожная индукция. Схема рассуждений следующая. Пусть при обстоятельствах *B* имеет место факт *F*. Обстоятельства *B1* в каком-то смысле подобны обстоятельствам *B*. Значит, можно ожидать, что и при них будет иметь место факт F.

Конкретизируем это на таком примере. Пусть в некоторой системе сочетание клавиш (например, *Ctrl+Del*) при редактировании удаляет строку. Пусть есть другая система той же фирмы. Тогда и в ней удаление строки должно выполняться аналогично. Тем самым умение переносится в близкую обстановку более обоснованно, чем при неполной индукции.

Аналогия и перенос имеют место и при составлении и использовании описания команд (или операторов) в общем виде.

Абстракция и конкретизация. Эти операции связаны прежде всего с компьютерным ***моделированием***. Исходная задача всегда ставится конкретно, в терминах области знания, где она возникла, и её перевод на язык информатики – самостоятельная и трудная проблема. Суть дела – в переводе с конкретного языка на абстрактный. Затем полученные результаты должны быть представлены пользователю или для себя на языке постановки задачи, т.е. выполнен обратный перевод.

Переводом задачи с формального языка на понятный, содержательный является написание ***комментариев***, что является, в общем, не простой задачей. Это искусство своего рода, творчество, хотя и не очень сложное. Главное здесь, найти «золотую середину», чтобы было не много пояснений, но в то же время достаточно для понимания. Особенно это актуально для учебных программ и алгоритмов.

7. ОРГАНИЗАЦИОННЫЕ ФОРМЫ ОБУЧЕНИЯ ИНФОРМАТИКЕ

7.1. Общие сведения

Организационные формы обучения (ОФО) являются внешним, видимым проявлением структуры информационных связей между педагогом и учащимися.

В общей дидактике конкретные ОФО различаются по числу участников совместной деятельности (фронтальная, бригадная или звеньевая, индивидуальная, парная работа) и по роли участников учебного процесса, то есть в зависимости от того, кто управляет, учитель или учащиеся. С древних времен начался монотонный отход от индивидуального обучения. Индивидуальное обучение стало расточительным по отношению к усилиям учителя. Традиционный учебник представлял собой первый шаг к синтезу фронтального и индивидуального обучения.

В информатике новым формообразующим элементом этой структуры является компьютер. С одной стороны, появление компьютера возрождает индивидуальные формы обучения, доступные ранее лишь для детей из высших классов. Благодаря обучающей программе или электронному учебнику ученик может воспринимать заложенные в них знания и опыт в своем индивидуальном темпе. Более того, все чаще предлагается не жесткий единообразный алгоритм обучения, а спектр вариантов обучения. Что и в каком порядке познавать, определяет учащийся в зависимости от способностей, уровня знания и других личных качеств.

С другой стороны, за счет тиражирования информации сохраняется и преимущество фронтальных форм: возможность учиться у лучших учителей. Методика преподавания информатики указывает на две новые ситуации: работа ученика под управлением обучающей программы либо “обучение” компьютера учеником. И то и другое представляет собой самоуправление познавательной деятельностью при немедленном отражении её результатов на экране.

Организационные формы обучения определяются целями, содержанием и методами преподавания, которые тесно взаимосвязаны. На организационную форму влияет содержание обучения. Так как метод обучения определяется содержанием, то он принимает свою форму. Применительно к информатике демонстрация учителем образца деятельности за компьютером (метод обучения) с целью передачи опыта такой деятельности (содержание обучения) наибольшему числу учащихся приобретает адекватную, фронтальную форму занятий.

В информатике в связи с возможностью использования компьютера в пределах одного урока могут комбинироваться многие формы. Особенно это касается сдвоенных уроков, которые более эффективны, так как сокращаются потери времени на подготовку вычислительной техники и вхождение класса в работу.

7.2.Организационные формы в зависимости от числа участников.

Как и в других предметах, ***фронтальные формы*** применяются при усвоении всеми учащимися одного и того же содержания или вида деятельности. Примеры: Лекция о системах счисления, семинар с выступлением учащегося о применении ЭВМ в медицине, демонстрация нового типа алгоритма или программы (отлаженной первым из решивших задачу) на доске или, лучше, на проекторе.

Достаточно сложной является фронтальная форма работы с использованием всеми учениками компьютеров. Особенно это касается сложных и не очень интересных тем. При этом надо сделать все возможное, чтобы учащиеся не перешли на игру, в *Internet* и т. д. Учитель должен решительно прервать начавшуюся индивидуальную самодеятельность и восстановить единое состояние компьютерной среды на всех ПЭВМ. Надо уметь вовремя перейти к парной или индивидуальной работе.

Обучение в составе бригады (звена) полезна с той точки зрения, что она при умелой организации может отражать реальное разделение труда в коллективе программистов, работающих над одной большой задачей. Типичным примером учебной задачи для использования бригадной формы может являться рисование в графическом режиме большой “картины”

состоящей из несколько простых элементов (дом, деревья, солнышко т. д.). Сначала эти части рисуются каждым членом бригады отдельно в виде, например, подпрограмм, а затем собираются на один компьютер.

Основные преимущества бригадной формы: интенсивное взаимное обучение, помощь друг другу а, значит, современная ликвидация пробелов в знаниях. Чтобы уменьшить возможные недостатки такой формы, надо правильно ответить на следующие вопросы. Какой должен быть состав бригады по способностям, по отношению к работе Все должны быть одинакового уровня (например, все “сильные” в одной бригаде, а все “слабые” в другой) ? При этом должны быть и задачи разной сложности. Какое должно быть оптимальное количество членов бригады? Кто должен ею руководить? Постоянным ли должен быть состав бригады? Предлагается подумать над этими вопросами.

Парная работа на ЭВМ является разновидностью бригадной, и сформировалась, казалось бы, из-за нехватки компьютеров. Но было замечено, что и при достаточном их количестве такая форма может быть полезной в начале обучения при освоении или закреплении новой сложной темы.

Учащийся, работающий один за компьютером, может по разным причинам (стеснителен, не у кого спросить, самолюбив и т.д.) не обратиться за помощью. Когда за одним компьютером работают двое, то вероятность незнания одного и того же обоими сразу уменьшается. Смелее обратятся к учителю. С другой стороны, в целом класс будет реже отвлекать его и будет возможность учителю больше внимания уделять слабым ученикам. При парной работе могут возникать такие же вопросы, как и при бригадной форме. Оба должны быть одного уровня или слабый ученик должен заниматься с сильным? Постоянными ли должны быть пары? Как оценивать результат парной работы? Как правильно распределить роли двух учащихся, работающих за одним компьютером? На эти и другие подобные вопросы предлагается ответить самостоятельно.

Заметим, что втроем работать за компьютером менее эффективно и нецелесообразно. Это может быть оправдано вот в каком случае. Когда большой класс или его половина (больше 10 человек) или школьники «тяжелые» в смысле успеваемости, то имеет смысл уменьшить количество одновременно работающих компьютеров, которые используются при объяснении нового материала. Тогда учителю легче руководить синхронной, под диктовку, работой, например, на 5-ти, а не на 12-ти компьютерах, даже если их достаточное количество. Надо учитывать, что такая форма занятия очень сложная. Легче сначала объяснить без компьютера, если, конечно, это целесообразно и в принципе возможно, а затем воспроизвести на компьютере.

Тогда, а также в других случаях, возможна работа **один на один с компьютером**. Отличие этой формы от классической самостоятельной в том, что в виде программы присутствует знание, обладающее собственной активностью. Надо учитывать, что поведение программы зависит от действий учащегося. В этом случае возобновляется фронтальное обучение, но с индивидуальными темпом и способами усвоения.

Один и без компьютера. Для осмысления того, что происходило за компьютером, бывает полезно от него удалиться во время урока, особенно при появлении трудных моментов или неожиданных действий ПЭВМ.

Полезно также и отстранение от компьютера тех учащихся, которые не подготовились к занятию: не изучили инструкцию по выполнению работы в диалоговом режиме; не разработали блок-схему алгоритма или не написали программу, предлагавшуюся в качестве домашнего задания, в бумажном варианте. Но, с другой стороны, такая форма не обязательно должна быть как наказание. Иногда надо объяснить, что это делается для его же блага (например, чтобы не потерял свой файл, чтобы надежно его сохранил).

7.3. Организационные формы обучения в зависимости от того, кто управляет

Управление со стороны учителя очевидно при фронтальных формах. Как говорилось ранее, сложнее управлять индивидуальной деятельностью по понятным причинам: ситуация за каждым компьютером практически уникальна. Можно предложить следующие варианты выхода

из этого: привлечь для помощи сильных учеников; «автоформализовать собственный педагогический опыт» (А.Г. Ершов) в виде обучающих программ; использовать педагогические знания других учителей, не обязательно в виде обучающих программ.

Работа возможна **под руководством товарища**. Иногда помощь товарища оказывается эффективнее, доступнее и понятнее, чем помощь учителя. Одна из причин этого в том, что обучаемый не боится спросить у товарища такое, что, по его мнению, спрашивать у учителя было бы стыдно. Другая причина в том, что некоторые учащиеся боятся спрашивать у учителя, так как, по их мнению, существует опасность получить плохую оценку. Если возникнут общие для класса затруднения, то есть смысл выйти на несколько минут из класса. За это время школьники друг у друга могут выяснить все возникшие вопросы намного быстрее, чем по инструкции или с помощью учителя.

7.4. Основные организационные формы.

Лекция имеет два смысла: это и форма, и метод. Лекция всегда фронтальная. Из всего многообразия вопросов, связанных с лекцией, обратим внимание на следующее. Она может поддерживаться компьютером как средством наглядности и демонстрации и, если позволяет материальная база, проводиться в дисплейном классе. Упражнение при этом выполняет учитель. При наличии подготовленных на компьютере конспектов у учащихся усиливается самоуправление познавательной деятельностью. Полезно использовать конспект, в котором слева отпечатан кратко основной материал, а справа оставлено место для комментариев учащегося, куда записываются пояснения, результаты компьютерных экспериментов и т.д. Во время компьютерной лекции не следует увлекаться демонстрацией готовых программ. Дело в том, что при изучении алгоритмизации и программирования часто бывает полезным сам процесс разработки алгоритма и программы, важно, как учитель думает, размышляет при этом.

Предлагается подумать над следующими вопросами. Какое должно быть соотношение теории и примеров в лекции? В каком порядке рассматривать те или другие вопросы? Какой оптимальный объём материала, рассматриваемый на лекции?

Семинар является переходной формой от фронтальной к индивидуальной работе и поэтому сохраняет свое значение и в изучении информатики. Прежде чем использовать компьютер, необходимо вырабатывать ряд навыков и умений, так как, например, разрабатывать алгоритм или осваивать новую систему прямо за экраном тяжело и это могут делать не все. Работать без предварительного изучения инструкции расточительно по отношению к машинному времени. Кроме этого, нужна адекватная форма работы для коллективного осмысления того, что сделано на компьютере, анализ того, что и почему получилось, и анализ типичных ошибок, основных затруднений и путей их преодоления. Это можно делать на семинаре в без машинном варианте.

Характерный признак ***рассказа*** – яркое, занимательное, эмоциональное повествование. Оно выполняется, как правило, в следующей последовательности: вступление, изложение, заключение. Знания при этом приобретаются теоретически, логика рассуждений здесь нестрогая: аналогии, примеры. Обратная связь при этом по степени внимания. Можно попросить учащегося воспроизвести услышанное. Основное назначение: передача конкретных сведений. Подходящие для такого метода темы: области применения ЭВМ, история развития вычислительной техники и компьютерных технологий, вирусы. Это можно поручить и «сильным» ученикам, так как рассказ имеет смысл использовать для изучения несложных тем, которые носят информационный характер.

Особенность ***беседы*** – системы управляющих вопросов, ведущих обучаемого к заранее намеченной преподавателем цели. Как правило, обсуждаются наиболее важные вопросы до и после практики. Для беседы можно выбирать также темы, по которым учащиеся уже что-нибудь знают: начальные сведения по ПЭВМ, понятия алгоритма. В тоже время учащиеся получают новые знания, систематизируют то, что было им известно. Поэтому дидактическая функция беседы — приобретения и упорядочивание знаний. Метод носит в основном теоретический характер, но может опираться на опередившую практику.

Лабораторная работа является основной формой в компьютерном классе. Все учащиеся одновременно работают на своих местах с программными средствами, переданными им учителем. Дидактическое назначение этих средств может быть различным. Например, с помощью обучающей программы можно освоить новый материал; а с помощью программы тренажера — материал, объясненный учителем. Во время лабораторных работ можно проверить усвоение полученных знаний или операционных навыков, например, с помощью контролирующей программы. Такие занятия или их фрагменты можно также использовать для решения общих, наиболее типичных и сложных примеров по изучаемой теме. В одних случаях действия школьников во время выполнения лабораторной работы могут быть синхронными, то есть все ученики (или их абсолютное большинство) выполняют одинаковые или похожие задания примерно с одинаковой скоростью. Если в классе есть очень способные ученики, или, наоборот, очень слабые, или пропустившие предыдущие занятия, то таким можно предложить другие задания. Поэтому не исключаются ситуации, когда школьники занимаются в различном темпе или с различными программными средствами.

Желательно, чтобы перед началом работы учитель проверил, готов ли класс выполнять запланированную работу по соответствующей теме. Тем, кто не готов, не стоит садиться за компьютер. Поэтому кроме компьютеров, в классе должны быть и столы для обычных занятий. Но учащийся не всегда должен воспринимать такое отстранение от компьютера как наказание, хотя это тоже может иметь место. Причиной неготовности может быть и излишняя самоуверенность, надежда разобраться за компьютером, по ходу дела. Требования к предварительной подготовке к лабораторным занятиям повышаются, если школа арендует машинное время в другом учебном заведении или в организации. При этом время может быть выделено «залпом», в течение 2-3 недель. Роль учителя во время фронтальной лабораторной работы — наблюдение за работой учащихся и при необходимости оказание им оперативной помощи. При этом можно использовать локальную компьютерную сеть. Ситуация несколько меняется при использовании встроенной помощи Help, которая рассчитана на обучение (но не с нуля!) в ходе работы. Учитель должен научить пользоваться такой помощью, чтобы быстро и, главное, разумно находить нужную информацию.

Более высокой формой работы по сравнению с фронтальными лабораторными работами является ***индивидуальный практикум*** (или учебно-исследовательская практика). Учащиеся получают индивидуальные задания для самостоятельной работы. Если в средних классах (6 – 8) их можно выполнить в течении одного урока, то в 9 – 11 классах на их выполнение может понадобиться в зависимости от темы и сложности задания два и более урока. Не исключается выполнение части работы дома. Такое задание, как правило, выдается для отработки знаний и умений после изучения целого раздела или большой по объему темы. Учащиеся сами или учитель решают, когда воспользоваться компьютером (в том числе и для поиска в сети), а когда поработать с книгой или обдумать некоторые вопросы без компьютера.

При этом учитель должен следить за тем, чтобы учитывались гигиенические требования к организации работы в КВТ, чтобы время непрерывной работы за компьютером не превышало рекомендуемых норм. Во время практикума учитель наблюдает за ходом выполнения заданий, оказывая школьникам оперативную помощь. При необходимости приглашает всех к обсуждению общих вопросов, анализу наиболее характерных часто повторяющихся ошибок.

Отметим некоторые характерные ***черты индивидуального практикума***.

Задания должны быть разнотипными по сложности. В связи с этим возникает ряд непростых вопросов. Кто определяет уровень сложности, сам школьник или учитель? Как найти «золотую середину», чтобы задание было не слишком простое (иначе мало пользы) и, в то же время, посильным (в противном случае ученик не получит удовлетворения от работы)? Как оценивать работу, если учащийся задание сложного уровня не выполнил, а с более простым смог бы справиться? Предлагается подумать над ответами на эти вопросы.

Индивидуальный практикум предполагает большую, по сравнению с лабораторными занятиями, самостоятельность. Связано это, прежде всего, с разнотипностью заданий по содержанию внутри одной темы. Трудно или даже иногда невозможно определить, какую задачу решать в качестве образца. Особенно это имеет место при изучении основ алгоритмизации и

программирования. Например, в теме «Циклические алгоритмы» это может быть и вычисление конечных и бесконечных сумм, произведений и факториалов, оригинальный вывод, задачи целочисленной арифметики и другие алгоритмы. Ещё более разнообразны задачи, например, по теме „Одномерные массивы”.

Из предыдущей особенности следует необходимость обращаться к учебникам, к справочному материалу при выполнении индивидуальных заданий. Поэтому такая форма занятий, как ни одна другая, учит самостоятельности, работе с литературой и другими источниками информации, включая и компьютерные (справки, Интернет и др.). Учителю необходимо иметь в виду и быть готовым к тому, что при выполнении индивидуальных заданий со стороны учеников, особенно сильных, могут поступать разнообразные, самые неожиданные, нестандартные вопросы.

Иногда полезно выполнять индивидуальные задания вдвоем. Но в этом вопросе не должно быть двух крайностей: поощрение коллективной парной работы и, наоборот, полный запрет совместной работы. В связи с этим возникает ряд вопросов. Например, предлагать одно задание на двоих или каждому свое? Если давать два задания, то они должны быть одного типа, одного уровня сложности или разные? Оба школьника должны быть одинаковы по способностям или разные? Как вы отнесетесь к тому, если один из двух учеников будет играть, возможно, пассивную роль? Предлагается также подумать над ответами на эти и другие подобного рода вопросы.

7.5. Вспомогательные организационные формы обучения

Экскурсия.

Основные её цели:

1) показать «живую» информатику, а точнее её какой-нибудь раздел в конкретной сфере;
2) скорректировать у учащихся «книжные» теоретические представления о настоящей информатике;
3) сформировать интерес к предмету (разделу), если экскурсия проводится до его изучения;
4) обобщение знаний, их систематизация, показ связи курса с реальностью, если экскурсия проводится на завершающем этапе;
5) профориентация на профессии, связанные с использованием ЭВМ.

Экскурсия не обязательно должна быть на крупное предприятие, в большую организацию. Может быть вполне достаточно, чтобы бухгалтер или секретарь, работник отдела кадров, или библиотекарь показали и прокомментировали конкретную систему в действии, на реальных данных, в «прямом эфире».

Экскурсия должна быть тщательно подготовлена:

— учитель должен договориться кто, когда и в каком объеме, в каком режиме будет показывать;
— не мешало бы предварительно учителю самому ознакомиться, хотя бы кратко, с тем, что будет показано;
— заготовить перечень вопросов, на которые учащиеся должны ответить во время экскурсии, по её окончании, или во время уроков. Например: какой объем информации хранится? Тип ЭВМ, ее характеристики? Точность расчетов? и др.;
— подготовить разъяснения по другим вопросам, на которые не обязательно должны отвечать школьники.

Итог экскурсии – ее коллективное обсуждение, анализ увиденного. Школьники должны выразить свое отношение к проведенному мероприятию.

Факультативные курсы.

Цели факультативных курсов:

— углубление знаний в конкретном разделе информатики (язык программирования, устройство ЭВМ, использование с демонстрацией и изучением конкретных систем и др.);
— показать связи информатики с другими дисциплинами;
— профориентация, в том числе и на профессию педагога;

Особое внимание надо обратить на межпредметный факультатив: информатика и математика, информатика и физика, даже информатика и литература или история и др. Такие курсы полезны по следующей причине. Образование поделено на непересекающиеся дисциплины (темы). Поэтому что учащемуся трудно привести знания, полученные при изучении разных предметов, в какую-то единую систему. Такие факультативные занятия помогают увидеть связи между предметами.

Полезен факультативный курс, связанный с управление школой. С администрацией можно согласовать и разработать подсистемы: «Оборудование», «Библиотека», «Документация», «Расписание» и т. п. Такая работа должна заинтересовать учащихся. Кроме этого, она будет иметь определенный воспитательный эффект.

Не обязательно ставить целью на факультативных курсах изучить углубленно конкретный язык программирования, что обычно практикуется. Отталкиваться надо от конкретных задач. Работая с библиотекаршей, зная ее задачи, можно лучше понять, например, необходимость внешней памяти, то есть файлов и средств работы с ними.

Характерные особенности факультативных занятий:

—большая самостоятельность;
—самоуправление познавательной деятельностью;
—меньшее число обучаемых.

Кружок.

Это наиболее гибкая, глубоко индивидуальная форма работы с разнообразным содержанием. Могут участвовать ученики разных возрастов. Обычно в кружке занимаются те, кто проявил повышенный интерес к предмету. По-видимому, его могут посещать и не очень способные ученики. Главное, чтобы кружок школьник посещал по собственному желанию и его участие (или не участие) не влияло на оценку по предмету.

Для старшего возраста на кружке можно решать объемные задачи, связанные целостным содержанием: разработка определенных баз данных для школы, сервисных программ для учителя информатики или обучающих (контролирующих) программ по другим предметам.

При работе с младшими школьниками следует избегать уклона в стандартный курс информатики, который они еще не изучали. Позже будет не интересно. Возможное содержание кружка: освоение готовых ПС, способствующих закреплению знаний из других дисциплин, изучаемых параллельно, т. е. обучающих программ. Для поддержания интереса можно рекомендовать простейшие графические пакеты.

8. ОРГАНИЗАЦИЯ ОБУЧЕНИЯ ИНФОРМАТИКЕ В ШКОЛЕ.

8.1. Урок.

Основной формой организации учебно-воспитательной работы с учащимися по всем предметам в учреждениях среднего образования является *урок*. Школьный урок образует основу классно-урочной системы обучения, характерными признаками которой являются:

- постоянный состав учебных групп учащихся;
- строгое определение содержания обучения в каждом классе на каждом уроке;
- определенное расписание учебных занятий;
- сочетание индивидуальной и коллективной форм работы учащихся;
- ведущая роль учителя;
- систематическая проверка и оценка знаний учащихся.

Как показывает пока незначительный опыт, который накопила наша школа после введения курса информатики, преподавание основ этого предмета наследует все дидактическое богатство отечественной школы. С другой стороны, в условиях внедрения в учебный процесс КВТ и новых информационных и коммуникационных технологий (ИКТ) весь известный опыт должен быть подвергнут критическому анализу, чтобы все прогрессивное стало достоянием практики в

преподавании этого молодого предмета. Применение КВТ и ИКТ может существенно изменить характер школьного урока, что делает еще более актуальным поиск новых и совершенствование старых форм и методов обучения.

8.2. Типы уроков.

Классификация типов уроков (или их фрагментов) можно проводить, используя различные критерии. Главный признак урока — его дидактическая цель, показывающая, к чему должен стремиться учитель. Исходя из этого признака, в дидактике выделяются следующие типы уроков:

1) урок сообщения новой информации (урок-объяснение);
2) урок развития и закрепления умений и навыков (тренировочный урок);
3) урок проверки знаний, умений и навыков.

В большинстве случаев учитель имеет дело не с одной целью, а с несколькими или даже со всеми сразу. Поэтому на практике широко используются комбинированные уроки. Они могут иметь разнообразную структуру и обладают рядом достоинств: обеспечивается смена видов деятельности, создаются условия для быстрого применения только что полученных новых знаний, имеет место обратная связь, накопление оценок, индивидуальный подход в обучении.

Важнейшая особенность постановки курса информатики на базе КВТ — это регулярная работа школьников на ЭВМ. Поэтому учебные фрагменты на уроках информатики можно классифицировать по объему и характеру использования ЭВМ. Так, уже первая программа машинного варианта курса ОИиВТ [15] предусматривала три основных вида организационного использования КВТ на уроках: демонстрация, фронтальная лабораторная работа, практикум.

В свое время классическая педагогика осознала недостаточность словесного образования, указав на необходимость приобретения информации не из книг и лекций, а через наблюдение, познание самих предметов и явлений. Поэтому полезным методом преподавания применительно к информатике является наблюдение учащегося за работой товарища или учителя, а также наглядные методы: ***иллюстрация и демонстрация***.

Их основная дидактическая цель — наглядно сообщить новую информацию. Используя демонстрационный экран, учитель показывает различные учебные элементы содержания темы: новые объекты языка или изучаемой системы, фрагменты программы, рисунки, схемы, тексты и т. п. При этом учитель сам работает за пультом компьютера, а учащиеся наблюдают за его действиями, чтобы затем повторить это же самостоятельно на своих компьютерах. Второй синхронный вариант предполагает, что школьники на экранах своих компьютеров повторяют те же действия одновременно, параллельно с учителем. Учитель может пересылать специальные демонстрационные материалы на ученические компьютеры, а учащиеся работают с ними самостоятельно. Не только учитель, а и учащиеся могут демонстрировать свои результаты.

Полезно иллюстрировать, прежде всего, то, что «не видно»: например, модель всей памяти ЭВМ или одной ячейки с записью и перезаписью в неё значений. Учитель может демонстрировать работу за компьютером в принципе и молча («делай, как я»). Любой современный компилятор демонстрирует место, и даже смысл ошибки. Особенность компьютерной демонстрации – динамичность и управляемость наглядными образами. Есть возможность вмешаться в процесс демонстрации для индивидуализации темпа или повторений. Возрастание роли и дидактических возможностей компьютерных демонстраций объясняется развитием и совершенствованием общих графических возможностей современных компьютеров и специальных технологий, предназначенных для этих целей. Например, к демонстрации можно также отнести современные средства презентации (например, PowerPoint).

8.3. Роль учителя.

В связи с распространением технологий компьютерного обучения, которые берут на себя все больше и больше педагогических функций, становится актуальным вопрос о возможных изменениях роли и обязанностей учителя. Большинство участников всевозможных дискуссий приходят по этому вопросу к следующему мнению: компьютер, вооруженный хорошими педагогическими программами, должен быть надежным и дружественным помощником учителя и ученика. Он берет на себя наиболее рутинные функции. Учителю остаются наиболее творческие

задачи обучения, воспитания и развития. Ведение дискуссий, поощрение тех или других действий ученика, поддержание дисциплины, личностное общение с учеником и его родителями и другие подобного рода функции ни одна самая лучшая программа, ни один самый мощный компьютер лучше учителя еще долго, а, может, и всегда, не сделает.

8.4. Особенности уроков по информатике.

Остановимся на некоторых дидактических особенностях уроков по информатике, связанных со специфическим характером учебного материала и использованием компьютера. Некоторые из них были подмечены еще до введения предмета ОИиВТ в школах. Обучение школьников в условиях постоянного доступа к ЭВМ обычно проходит при повышенном эмоциональном состоянии учащихся. Объясняется это частично тем, что школьник очень скоро обнаруживает состояние власти над “умной” машиной. Это придает ему уверенности, у школьника возникает естественное стремление поделиться своими знаниями с теми, кто ими не обладает. Поэтому наиболее успешно занимающиеся учащиеся могут помогать учителю.

При обучении школьников компьютерным технологиям полезным является деятельностный подход к обучению, и, в частности, так называемый *метод проектов*. Под разработкой учебного проекта понимается определенным образом организованная целенаправленная деятельность. Таким проектом может быть, например, компьютерный курс изучения определенной темы, логическая игра, смоделированный на компьютере макет лабораторного оборудования, разработка программы на языке программирования и многое другое. В простейшем случае, на начальном этапе изучения информатики в качестве сюжетов для изучения компьютерной графики можно предложить задачи проектирования рисунков животных, строений, симметричных узоров и т.п.

При использовании метода проектов необходимо учитывать ряд условий. Учащимся следует предоставить широкий набор проектов для реального выбора, которые могут быть как индивидуальными, так и коллективными. Школьника необходимо снабдить инструкцией по работе над проектом. При этом можно учитывать, что одни лучше усваивают материал, читая текст, другие — слушая объяснение, третьи — непосредственно пробуя, ошибаясь и находя решения в процессе практической работы.

Для ребенка важна практическая значимость полученного результата и оценка со стороны учителя, сверстников, родителей. Поэтому проект должен предполагать для исполнителя законченность и целостность проделанной им работы, желательно по возможности в игровой или имитационной форме. Важно также, чтобы завершенный проект был презентован. Необходимо создать условия, при которых школьники имеют возможность обсуждать друг с другом свои успехи и неудачи.

8.5. Средства обучения информатике. Кабинет вычислительной техники.

Введение в учебный план средней школы нового предмета ОИиВТ потребовало разрешения проблемы взаимодействия учащихся с ЭВМ. Это затрагивало в конечном итоге и интересы преподавания других школьных дисциплин, постановки всего школьного дела. Рассматривались различные пути решения этой задачи, например, вычислительные центры коллективного пользования. Основным из них явился следующий: оборудование в школах кабинетов, оснащенных комплексами учебной вычислительной техники на базе ПЭВМ, включенных в локальные и глобальные сети.

Кроме компьютеров, к техническим средствам обучения информатике можно отнести также:

учебное, демонстрационное оборудование, сопрягаемое с ПЭВМ (например, учебные роботы, управляемые компьютером; модели для демонстрации принципов работы ПЭВМ, ее частей, устройств);

средства телекоммуникаций, обеспечивающие доступность информации для обучаемых, вовлеченность их в учебное взаимодействие, богатое возможностями и разнообразием видов использования ресурсов Всемирной информационной сети.

Используемое в кабинете информатики программное обеспечение (ПО) должно включать:

системное ПО (операционная система, операционные оболочки, сетевое ПО, антивирусные

средства и т. п.);

программные системы базовых информационных технологий (текстовый и графический редакторы, системы управления базами данных, электронные таблицы, графические системы, системы подготовки компьютерных презентаций и др.);

инструментальное ПО — средства разработки программ;
ПО учебного назначения;
ПО издательской деятельности для нужд школы.

Так как программные средства быстро сменяются и устаревают, то их полный обзор сделать практически невозможно или, правильнее, бесполезно.

Помимо компьютерного оборудования, кабинет информатики рекомендуется оснащать:

набором учебных программ для изучения информатики и отдельных разделов некоторых других предметов;

комплектом учебно-методической, научно-популярной и справочной литературы;

журналом входного и периодического инструктажей учащихся по технике безопасности;

журналом использования КУВТ на каждом рабочем месте;

журналом сведений об отказах ПЭВМ и их ремонте;

аптечкой первой помощи и средствами пожаротушения;

инвентарной книгой учета имеющегося в кабинете учебного оборудования, планами его дооборудования.

При оборудовании и использовании компьютерного класса большое значение имеет строгое соблюдение санитарных норм и правил, предназначенных для предотвращения неблагоприятного воздействия на человека вредных факторов, связанных с использованием видеодисплейных терминалов. Этот вопрос еще более актуален, так как речь идет о здоровье детей.

Согласно гигиеническим требованиям, для учителей длительность работы в дисплейных классах и кабинетах информатики устанавливается не более 4 часов, а для инженеров — 6 часов в день. Для снижения нагрузки в течение рабочего дня должны устраиваться регламентированные перерывы в работе. Разрешаемое время непрерывной работы учащихся за дисплеями зависит от их возраста и не должно превышать 20 мин. для учащихся 6 – 7 классов, 25 мин. — для 8 – 9 кл. и 30 мин. — для 10 – 11 кл. После установленной выше длительности непрерывной работы должен проводиться комплекс специальных упражнений для глаз, а после каждого урока — физические упражнения для профилактики общего утомления. Число уроков для учащихся 10 – 11 классов с использованием компьютеров должно быть не более двух в неделю, а для остальных классов — не более одного урока. Внеклассная работа с использованием ПЭВМ должна проводиться не чаще двух раз в неделю общей продолжительностью не более 60 мин. для учащихся 2 – 5 классов и 90 мин. — для 6 класса и старше.

Фактор санитарно-гигиенических требований к организации учебного процесса в КВТ накладывает жесткие ограничения на структуру каждого урока по информатике, что должно учитываться при их планировании. В частности, это касается прежде всего учета продолжительности не важно какой работы за включенными компьютерами.

Для обеспечения организации работы кабинета информатики приказом директора школы назначается заведующий КВТ из числа учителей информатики. Он является организатором работы учителей и учащихся по применению средств вычислительной техники, информационных технологий в преподавании информатики и других учебных предметов. Кроме этого, заведующий принимает меры по дооборудованию и пополнению кабинета учебно-наглядными пособиями, вычислительной техникой и другими техническими средствами, несет ответственность за работоспособность и сохранность всего имеющегося в кабинете оборудования, своевременность и тщательность его профилактического технического обслуживания, за поддержание в КВТ санитарно-гигиенических требований и требований техники безопасности.

Вводный и периодический инструктаж по ней должен проводить, как правило, учитель

информатики, который проводит занятия в соответствующем классе. На вводном инструктаже, который проводится в виде небольшой лекции, а лучше беседы, учитель знакомит учеников с распорядком в кабинете, правилами техники безопасности и гигиены труда, с потенциально опасными моментами, которые могут возникнуть в процессе работы, и с соответствующими мерами предосторожности. Краткий периодический инструктаж непосредственно перед работой за компьютером дополняет вводный и знакомит с правилами выполнения конкретной работы, с возможными опасными ситуациями и правилами поведения при их возникновении. Все сведения по проведению инструктажа заносятся в специальный журнал, в котором должны быть записаны фамилии, кто и кого инструктировал и их подписи, а также краткое содержание инструктажа.

Помощь в работе заведующему кабинетом оказывает лаборант (или техник), который находится в непосредственном подчинении заведующего кабинетом. Он отвечает за правильное хранение и использование учебного оборудования, участвует в его приобретении и ведет его учетность и инвентаризацию. По плану преподавателя и под его руководством лаборант готовит оборудование к работе, обеспечивает постоянную готовность противопожарных средств и средств первой помощи. Он должен регистрировать отказы техники во время занятий и проводить по возможности мелкий ремонт вышедшего из строя оборудования. Согласно санитарным требованиям при кабинете информатики должна быть лаборантская комната определенной площади, соединенной с учебным помещением.

РАЗДЕЛ 2. МЕТОДИКА ПРЕПОДАВАНИЯ ОСНОВ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ.

9. Методические особенности раздела “ОАиП”

Раздел “Основы алгоритмизации и программирования ” (ОАиП) имеет явно выраженную практическую направленность, что способствует усилению связи обучения с жизнью. В этом разделе, как ни в одном другом, большое внимание должно уделяться решению задач и выполнению упражнений. При решении некоторых задач строится математическая модель и вычислительный алгоритм, требующие обоснований, выходящих за рамки школьной программы. Чтобы больше внимания уделить вопросам алгоритмизации, можно не давать обоснований используемых методов и фактов. Желательно, чтобы таких задач было как можно меньше.

Происходит расширение традиционного, известного из других школьных предметов понятия величина. Вводятся и используются не только числовые, но и литерные величины. Кроме этого, ученики знакомятся с новыми способами организации данных в форме одномерных и двумерных таблиц (массивов). При изучении программирования школьника надо приучить абсолютно точно по определенным фиксированным правилам записывать алгоритм. Поэтому с самого начала необходимо привить школьникам такие качества, как четкость, ясность, аккуратность в записях, предельную внимательность и сосредоточенность, особенно при работе на ЭВМ.

Необходимо иметь в виду следующую методическую сложность. Алгоритмический стиль мышления, который мы должны сформировать, отличается от математического, хотя и основывается на нем. Решить задачу и составить алгоритм (программу), с помощью которой ЭВМ должна решить ту же задачу, это не всегда одно и то же, то есть требует различных способностей, различных сторон мышления. Например, даже ученики с хорошими математическими способностями не сразу понимают динамический смысл записи алгоритма. Если любое записанное действие при решении математической задачи выполняется всегда, если оно записано, то в программе, например, может выполняться только одна из двух ее ветвей. Некоторая последовательность действий может повторяться, динамически может меняться значение некоторой величины и т. п. Эти трудности возрастают в условиях безмашинного изучения алгоритмов. Как преодолеть эту сложность? Опыт изучения других дисциплин почти не помогает, так как формулы в математике, физике, химии имеют другой, статический смысл. Для лучшего понимания учениками работы алгоритма можно рекомендовать следующие приемы: исполнение алгоритма вручную; использование блок-схем ; выполнение программы по шагам, т. е. трассировка программы, с помощью современных средств программирования (команды *Trace Into, Step Over* и др. в современных интегрированных средах программирования).

10. Общие методические принципы обучения ОА и П

Рассматриваются принципы, которыми можно руководствоваться при изучении ОА и П как в машинном , так и в безмашинном вариантах.

10.1. Принцип многоуровневости

Этот принцип связан с тем, что в алгоритмизации и программировании, как ни в одной другой области деятельности, все темы, элементы языка тесно переплетены между собой. Поэтому некоторую трудность представляет определение порядка изучения понятий и конструкций языка, приемов и методов программирования. Большинство учебников построены таким образом, что сначала рассматриваются элементы выбранного языка программирования (константы и переменные, все их типы, все операции над ними, правила построения выражений и т. д.), а потом последовательно изучаются операторы, методы и технологии программирования. При этом, как правило, сразу в одном пункте приводятся все возможности изучаемого элемента языка, например, оператора, хотя некоторые из них редко используются или достаточно сложные. Аналогично при изучении какой-нибудь другой системы (не обязательно системы программирования) приводится полное описание некоторого элемента, хотя на начальном этапе достаточно знать ее простейшую форму, лишь некоторые ее возможности.

Такую общепринятую методику можно усовершенствовать следующим образом. Параллельное изучение ЯП и средств разработки программы делим на этапы (уровни). Первый из них (основы, начало, быстрое введение) включает лишь некоторые наиболее простые и часто используемые в дальнейшем элементы (типы данных (вещественные и целочисленные), операции над ними (арифметические), операторы, стандартные функции и процедуры, и т.п.), обеспечивающие возможность быстрой разработки простейших программ. После приобретения первоначальных навыков программирования школьник осваивает второй, основной уровень. На этом этапе изучаются новые элементы языка и ОС а также рассматриваются новые возможности тех же конструкций языка и ОС, основы которых были изучены на первом этапе. Первые два уровня должны быть усвоены всеми учениками класса.

Третий, заключительный уровень состоит в изучении дополнительных возможностей системы и языка, которые могут встретиться реже, при разработке более сложных программ. Изучаемые на этом этапе возможности ОС, ЯП позволяют повысить эффективность использования ЭВМ и оптимизировать программы. Такие дополнительные возможности есть смысл изучать со всеми учениками только в специализированных классах, а в обычной школе лишь на дополнительных необязательных занятиях с учениками, проявляющими повышенный интерес и способности к программированию.

10..2. Принцип предварительной мотивации

Рекомендуется в качестве основного принципа изучения ОА и П использовать принцип предварительной мотивации элементов выбранного ЯП. Это означает, что любую тему лучше изучать “от частного к общему”, а не наоборот. Элементы языка рекомендуется вводить по следующей схеме:

1) наглядная простая задача, которая требует для своего решения введения нового элемента, т. е. показывается необходимость изучаемого элемента;
2) пример использования нового элемента при решении этой задачи;
3) общий вид конструкции, ее формальное описание;
4) примеры на закрепление;
5) выполнение индивидуальных заданий на ЭВМ и (или) упражнений без использования компьютеров.

При этом пункты 1, 2, 3 лучше рассмотреть в форме лекции, 4 и 5 на практических занятиях. При этом теорию желательно максимально сокращена, основные методы, приемы программирования должны изучаться на простых наглядных примерах. Следуя этому принципу, не обязательно всегда соблюдать одинаковый порядок форм занятий: лекция – семинарские занятия – лабораторные работы. При изучении некоторых тем, например, ввод-вывод, эффективнее сначала решить задачу с использованием ЭВМ и (или) выполнить упражнения без ЭВМ, а потом на лекции обобщить, подвести итоги, объяснить новые дополнительные

возможности изучаемого элемента и обратить внимание на наиболее сложные вопросы темы.

10.3. Принципы сравнения и повторения

При изучении многих тем следует руководствоваться принципом сравнения, который предполагает анализ различных алгоритмов и (или) программ решения одной и той же задачи, выбор из них наилучшего. Этот же принцип можно использовать также для того, чтобы показать необходимость той или иной конструкции языка, быстрее и лучше понять ее. Для этого можно записать несколько вариантов некоторого фрагмента программы решения одной и той же задачи, используя разные элементы. Можно привести, например, следующие такого рода элементы:

представление положительных и отрицательных целых чисел в памяти ЭВМ;

логические операции „и” и „или” (не важно, в каком языке);

операторы цикла *while* и *for* в языках *Pascal* и *C++*;

операторы цикла *while* и *Repeat* в языке *Pascal* или *while* и *do …while* в *C++*;

операторы ветвления *if* (если) и *Case* (выбор) в языке *Pascal* или *if* и *switch* в *C++*;

вспомогательные алгоритмы и вспомогательные алгоритмы для вычисления функций в “школьном” алгоритмическом языке;

процедуры и процедуры-функции в языке *Pascal*;

функции типа *void* и функции с одним результатом в языке C++;

атрибуты доступа *private* и *public* в объектно-ориентированном программировании.

Один из вариантов, как правило эффективнее и “красивее”, хотя он может быть труднее для изучения.

Благодаря такому методическому приему реализуется на практике принцип повторения. Один из сравниваемых элементов изучается обычно раньше (например, оператор *if*). Повторение при изучении ОА и П в большей степени, чем в математике, играет важную роль. Это связано с тем, что при изучении нового элемента языка, при написании программ по новой теме надо знать практически почти все ранее изученные темы. Например, выражения, операторы *if*, ввода-вывода будут использоваться при написании программ по любой новой теме.

10.4. Принцип индивидуальных заданий

При обучении программированию следует исходить из того, что развить алгоритмическое мышление школьника, научить его программировать и решать задачи с применением ЭВМ можно только при условии, если каждый из них будет регулярно составлять программы, выполнит самостоятельно несколько индивидуальных заданий. Каждое такое задание закрепляет и проверяет знания по одной или нескольким наиболее важным темам и предполагает выполнение следующих этапов:

изучение постановки задачи;
разработку алгоритма ее решения;
запись алгоритма в виде программы;
подготовку программы на компьютере в соответствующей системе программирования;

отладку и тестирование программы;
анализ результатов и подготовку отчета.

В отличие от традиционной методики, когда всему классу предлагается общая задача, при такой организации работы школьника исключается формальное переписывание друг у друга. При этом можно учесть способности школьников, варьируя уровнем сложности заданий. . Важно также что при выполнении индивидуальных заданий повышается активность на занятиях и ответственность за его своевременное выполнение.

При этом эффективность обучения возрастает, если использовать следующие методические приемы:

– выполнение одного задания вдвоем (наиболее способный ученик со слабым) ;

– доклад наиболее интересных, оригинально, нестандартно выполненных заданий на семинарском занятии и его обсуждение;

– обмен готовыми программами друг с другом с объяснением;

– конкурс на лучшую программу решения одной и той же задачи;

– бригадный метод работы (одно задание на 2-4 человека). Это позволяет ученикам обмениваться идеями друг с другом, искать вместе ошибки, учит работать в коллективе.

Обсуждая этот принцип, возникает вопрос, как распределять варианты заданий. Всем давать задания примерно одинаковой сложности внутри одной темы или учитывать способности учеников? Одинаковым ли должно быть количество заданий или задания по некоторым темам рекомендовать только наиболее „сильным" ученикам? Неправильно поступают те учителя, которые всему классу в качестве домашнего задания предлагают одни и те же или одинаковые по сложности задачи. Более способным ученикам мало пользы от простых заданий, а слабые ученики вынуждены обращаться за помощью к друзьям, родителям, репетиторам и т. д.

10.5. Принцип параллельности

Важным принципом изучения ОА и П в машинном варианте, кроме рассмотренных, должен быть принцип параллельности изучения выбранного языка программирования (*Pascal, C++* и др.) и интегрированной среды разработки программ. Параллельно с этим можно изучать и операционную систему (например, *Windows*). Это позволит уже с первых уроков часть занятий проводить в дисплейных классах.

Необходимо обратить внимание на то, что не должно быть двух крайностей в изучении начал программирования при наличии в школе компьютеров. Первая из них заключается в том, что сначала в течении некоторого времени, например, одной четверти, изучаются основы ЯП в безмашинном варианте традиционными, методами с мелом у доски и лишь потом осваиваются методы разработки программ и школьников допускают к компьютерам. При таком подходе теряется интерес к программированию и многие его элементы (например, ввод-вывод), динамический смысл выполнения операторов без общения с ЭВМ остаются непонятными длительное время.

Нельзя согласиться и с теми учителями информатики, которые все уроки проводят в дисплейных классах и (или) на несколько минут отключают компьютеры для объяснения нового материала. При такой методике меньше внимания уделяется развитию алгоритмического мышления и наиболее принципиальным общим вопросам и методам программирования, что является самым важным при изучении любого ЯП.

11. Формы занятий

С учетом отмеченных принципов рекомендуется организовать изучение ОА и П в машинном варианте с использованием следующих форм: 1) лекция без ЭВМ; 2) семинарское занятие без ЭВМ; 3) лабораторные занятия в дисплейных классах с использованием ЭВМ; 4) индивидуальные занятия и консультации.

Основным принципом организации занятий в школе по ОА и П в условиях систематического использования ЭВМ должно являться взаимосогласованное изложение теории и практики с опережающим изложением практических вопросов во время работы на машине. При этом приоритет должен быть отдан решению задач, поэтому основными формами занятий должны быть семинарские и лабораторные, во время которых решаются задачи и выполняются упражнения. Такие занятия рекомендуется проводить с ½ или 1/3 частью класса (10-15 человек), чтобы каждый ученик имел возможность готовить и отлаживать на ЭВМ программы.

Лекцию можно проводить со всем классом или с двумя параллельными классами (до 40 человек) в течении не более 45 минут. Двухчасовая лекция в школе малоэффективна, несмотря на то, что такой формой занятий иногда руководители учреждений образования восхищаются в средствах массовой информации. Опыт показывает, что в этом случае после первого часа ученики

с трудом воспринимают материал. Сэкономленное за счет объединения двух классов время можно использовать для индивидуальной работы. Школьникам, проявляющим повышенный интерес к программированию, можно рекомендовать предварительно изучить материал лекции по литературе. Тогда на лекции, практическом занятии он может получить ответ на возникшие во время самостоятельного изучения вопросы.

Распределение часов между разными формами занятий зависит от изучаемой темы, количества учеников на лекции и практических занятиях, уровня их подготовленности. Например, если тема имеет теоретическую направленность, то соотношение может быть таким: 1л: ½ сем: ½ ЭВМ. При таком варианте семинарское и лабораторное занятия проводятся за 1 урок в дисплейном классе. Если тема имеет практический характер, то можно рекомендовать следующее соотношение: ½ л: ½ сем: 1 ЭВМ.

Эффективными являются также сдвоенные уроки информатики. Тогда за 2 часа в дисплейном классе можно провести все формы занятий. Два часа подряд без компьютера можно с пользой работать, мастерски используя различные методические приёмы и умело варьируя формами занятий (повторение в форме беседы предыдущей темы, лекция по теоретической части нового материала, выполнение небольших упражнений учителем или учениками, беседа по закреплению изученного только что материала). Но при любой организации занятий из медицинских соображений школьник не должен работать за дисплеем более 45 минут подряд в течение дня.

Можно предложить несколько разнообразных форм проведения лабораторных занятий с использованием компьютеров. Они выбираются в зависимости от изучаемой темы и поставленной цели занятия, уровня подготовленности учащихся, наличия компьютеров, количества учеников и других факторов.

1) Сначала, не включая компьютеры, учитель дает возможность записать основные команды, порядок работы, а затем на этом же занятии каждый ученик, используя конспект, выполняет на ЭВМ то, о чем было рассказано.
2) Если учеников не очень много (не более 10 человек), объясняемый материал можно сразу показывать на одном (или двух) дисплеях, а затем каждый ученик закрепляет показанные элементы на своем компьютере.
3) Этот вариант обучения предполагает синхронное выполнение учениками действий на ЭВМ по командам, которые учитель записывает на доске. Недостатком этого способа является то, что выполнять команды ученики будут неравномерно, и некоторые из них должны будут ждать следующего указания учителя. Следующую команду учитель может подавать лишь после того, когда предыдущий пункт выполнен всеми учениками.

Эти три методических приема целесообразно использовать на начальном этапе обучения работе на компьютерах или при демонстрации новых возможностей ЭВМ. При этом не следует включать много компьютеров. Чем больше включено компьютеров при объяснении нового материала, тем труднее объяснять его. На первых занятиях, а также когда осваивается новая тема, за одним компьютером можно работать нескольким школьникам.

При закреплении показанных навыков, при выполнении индивидуальных заданий основной является индивидуальная форма работы учителя с учениками, при которой по мере возникновения у них вопросов учитель оказывает необходимую помощь. Теперь можно работать с большим количеством компьютеров (до 15), равным количеству учеников.

12. Выбор задач по ОАиП

12.1. Общие сведения.

При выборе задач по ОА и П необходимо руководствоваться следующими ***принципами***. Важно не только и не столько изучить правила конкретного языка программирования и обучить школьника записывать с его помощью алгоритм. Прежде всего, и это самое главное, задачи должны способствовать развитию алгоритмического стиля мышления, помочь овладеть общими методами и приемами программирования. С помощью задач школьник должен освоить основные типы алгоритмов (ветвления, циклы с известным и неизвестным количеством повторений),

научиться использовать и обрабатывать данные различных типов (целые, вещественные, логические, символьные, строки, массивы, структуры).

Не существенно, из какой области знаний взяты задачи. Гораздо важнее, можно ли с их помощью научиться разработке качественных программ и освоить фундаментальные понятия информатики и, в частности, алгоритмизации. Но с другой стороны, с целью поддержания меж предметных связей, демонстрации возможностей применения ЭВМ, для разнообразия примеров желательно решать с помощью ЭВМ не только математические задачи. В противном случае школьник, проявляющий способности к программированию, но у которого определённые проблемы с математикой, может потерять интерес к информатике. Задачи должны быть интересными, наглядными, понятными школьнику, разнообразными не только по тематике, но и по методическим приемам, используемым при их решении. Задачи не должны требовать много время на объяснение их содержательной постановки. Перед выбором каждой задачи должны быть поставлены определенные цели, и ее решение должно способствовать их достижению.

Все задачи по программированию можно разделить на ***два больших класса: задачи-программы и упражнения.*** Первый из них включает задачи на составление алгоритма и полной программы на выбранном языке и их анализ, а в машинном варианте, кроме этого, и её отладку. В упражнениях требуется записать один или несколько вариантов элемента языка (записать выражение, оператор) или части программы и (или) проанализировать их. При анализе программы (ее части) надо ответить на ряд вопросов, например, есть ли ошибки, всегда ли ошибка будет проявляться, как повлияет на результат какое-нибудь изменение в программе и т.п. Первый класс задач является более важным, и его необходимо чаше практиковать, особенно для закрепления одной или нескольких тем и при выполнении индивидуальных заданий. Но на начальном этапе изучения темы и при объяснении наиболее сложных принципиальных вопросов нельзя игнорировать упражнения, которые играют подготовительную, вспомогательную роль. Упражнения особенно полезны при работе с отстающими школьниками. Частным случаем упражнений являются тесты.

Одним из этапов обучения ОА и П, важным условием умения составлять качественные программы является понимание готовых программ, умение их “читать”. поэтому полезно выполнять упражнения на восстановление постановки задачи, для которой записана программа или ее часть. В случае затруднения можно рекомендовать исполнить ее для одного или нескольких тестов или составить блок-схему.

Необходимо обратить внимание на такие упражнения, в которых надо сравнить несколько вариантов программы или алгоритма. При этом необходимо выбрать правильный вариант из предложенных. Полезны упражнения, в которых требуется исследовать на эффективность программу или её часть, выбрать наилучший из правильных вариантов или записать, если можно, более эффективный вариант.

12.2. Упражнения на составление и анализ блок-схем

При изучении основных типов алгоритмов, наиболее важных и сложных операторов и приемов программирования полезно использовать задачи и упражнения на составление и анализ блок-схем. Кроме простоты и наглядности, важной отличительной их особенностью и достоинством является то, что они не зависят от языка программирования. Научившись записывать алгоритмы в виде блок-схем, достаточно освоить синтаксические правила конкретного языка, чтобы можно было разрабатывать эффективные программы. Блок-схемы способствуют быстрому развитию алгоритмического мышления. Их можно использовать как в машинном, так и в без машинном вариантах изучения ОА и П. К сожалению, в последнее время в учебниках по программированию этому удаляется недостаточно внимания.

Блок-схемы можно изучать как параллельно с языком программирования, так и последовательно. В первом случае сначала в течение 1-2 уроков учащимся дают простейшие элементы блок-схем, правила их разработки и требования к ним. При изучении методов программирования и наиболее сложных операторов языка составляется при необходимости блок-схема алгоритма, а потом с ее помощью разрабатывается программа. При таком способе, хотя и требуется больше времени, школьники лучше понимают динамический смысл выполнения

программ. Последовательное изучение блок-схем и языка программирования заключается в том; что на начальном этапе изучения ОА несколько уроков (5-7) учащихся разрабатывают и анализируют блок-схемы различных типов алгоритмов (линейных, разветвляющихся, циклических с известным и неизвестным количеством повторений, итерационных, алгоритмы с вложенными циклами). Затем эти же типы алгоритмов изучаются на уровне программ.

При этом не обязательно решать только математические задачи. Эффективным является составление и анализ блок-схем так называемых “бытовых” алгоритмов. Вот примеры таких алгоритмов, возникающих в повседневной жизни:

переход улицы при различных условиях (есть (нет) светофора);

проход в метро пассажира, пользующегося (не пользующегося) льготами;
алгоритм, как приготовить и съесть завтрак;
различные игровые алгоритмы и т.п.
На уровне блок-схем полезно использовать также задачи по управлению движением Робота на клетчатом поле.

Научившись разрабатывать блок-схемы основных типов алгоритмов, можно изучать основные конструкции языка программирования, основываясь на полученных навыках и умениях. Это выполняется качественнее и быстрее, чем без предварительного изучения блок-схем. При таком подходе удачно реализуется принцип повторения. Кроме этого, экономится машинное время, так как разработка блок-схем не требует использования компьютеров. На уровне блок-схем легко показать основные принципы и понятия структурного программирования: пошаговая детализация алгоритма и базовые алгоритмические (управляющие) структуры с одним входом и одним выходом.

В наброске программы не используются никакие геометрические фигуры. Некоторые принципиальные операторы (цикла, if), но не обязательно все, записываются по синтаксическим правилам конкретного языка. А вместо группы некоторых операторов или других элементов записываем на русском языке, что они делают. Например, рассмотрим следующую задачу. В каждой строке матрицы *A[10,15]* все положительные числа разделить на первое по порядку отрицательное число. Если отрицательных чисел в строке нет, оставить ее без изменения. Набросок программы для этой задачи выглядит следующим образом:

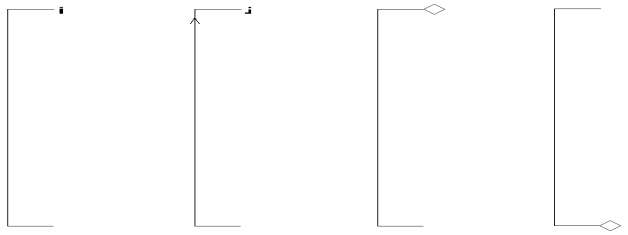
```
Ввод матрицы
For i: = l to 10 DO
        Begin

поиск первого отрицательного числа в i-ой строке
if есть отрицательное число
then преобразование i-ой строки
Else writeln ( 'отриц. числа в',  i,  '-ой строке нет');
    End
       Вывод матрицы.
```

Некоторые из действий, например, поиск первого отрицательного числа в *i*-ой строке, можно детализировать, используя в зависимости от сложности этой части программы обычную блок-схему, схему циклов, набросок этого фрагмента программы, или операторы языка.

Кроме обычных блок-схем в виде фигур, соединенных линиями, при разработке алгоритмов, представляющих вложенные и (или) последовательные циклы, полезно составить схему циклов и (или) набросок программы. Например, при изучении языка Паскаль для изображения различных типов цикла можно использовать следующие наглядные представления:

1)For i…to…Do 2) For j…downto…Do 3) While…Do 4) Repeat



1) 2) 3) 4)

Например, для последовательности чисел по итерационной формуле Герона найти $y = \sqrt{a}$. Признак конца ввода – число 0 . Тогда схема циклов может иметь, например, вид, представленный на рис.1. Для удобства объяснений циклы можно нумеровать.

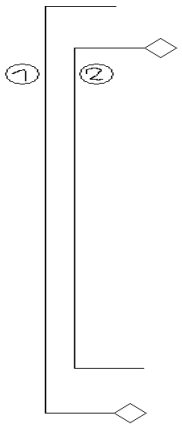


Рис.1.

При использовании блок-схем алгоритмов традиционным основным типом задач являются задачи: “составить блок-схему (схему циклов, набросок программы) для решения следующей задачи...”. Кроме этого, на разных этапах изучения ОА и П можно рекомендовать следующие типы задач и упражнений:

– по готовой блок-схеме составить программу (часть программы, записать оператор);

- по заданной программе или ее фрагменту составить блок-схему (схему циклов);

- по заданной схеме циклов написать одну или несколько из возможных постановок задач на определенную тему. Такие упражнения можно использовать на заключительном этапе изучения ОА и П, когда учащиеся получат определенный опыт и будут иметь некоторый набор задач разного типа и их алгоритмов.

Кроме этого, при изучении и использовании блок-схем можно рекомендовать те же типы упражнений и методические приемы, которые перечислены в других параграфах этой главы.

12.3. Тесты

Тесты можно практиковать не только во время проверки знаний (контрольные работы, зачётные занятия, экзамен и т.п.), но и во время обычных занятий, как правило, без использования компьютера. Основная их особенность в том, что проверку правильности ответов на них можно формализовать и поэтому поручить компьютеру. Тесты могут быть нескольких типов.

Основным является тест с вариантами ответов, из которых надо выбрать один или несколько правильных. Вот несколько примеров теста с вариантами ответа.

Дан код:

```
void fun (int x, int *y)
```

```
 {  *y=x*10;   }                    //1
void main()
   {     int a=5,
        *b=new int (2);             //2
         ????????;
         cout<< a<< “   “<< (*b);   //3
         getch() ;    }
```

Как вызвать функцию на месте ????????, чтобы вывести 5 50?

Варианты ответов: 1) ошибка в строке //1. 2) ошибка в строке //2. 3) ошибка в строке //3.

4) *fun6(a,&b);* 5) *fun6(a, b);* 6) *fun6(a, *b);*

Из предложенных вариантов правильных может быть несколько.

Но надо иметь в виду, что иногда наличие вариантов ответа может оказать определённую помощь. Поэтому более сложным является тест, в котором надо записать ответ.

12.4. Комплекс взаимосвязанных задач

Особое внимание следует обратить на решение комплекса взаимосвязанных задач. При этом задачи могут быть как однотипными в алгоритмическом отношении и одинаковыми по трудности, так и с последовательным возрастанием их сложности.

Пусть дана матрица A(25,10), где a[i,j] – оценка i-го ученика по j-му предмету за одну четверть. Примером комплекса однотипных задач являются следующие:

1) Для каждого ученика найти количество пятерок.
2) По каждому предмету найти количество пятерок.
3) Во всей матрице найти количество пятерок.

В алгоритме первой задачи внутренний цикл по номеру строки, а внешний – по номеру столбца, а во второй задаче наоборот. В алгоритме третьей задачи несущественно, какой цикл внутренний, какой внешний. Но в отличие от первых двух задач, в которых оператор K5:=0, где K5 – количество пятерок, записывается между операторами цикла, в третьей задаче этот оператор должен быть перед внешним циклом.

Примером последовательности задач с усложнением являются следующие задачи:

1) Найти наибольшее из двух чисел.
2) Найти наибольшее из трех чисел.
3) Найти наибольшее число в одномерном массиве.
4) Найти наибольшее и наименьшее числа в одномерном массиве и их номера.
5) Найти второе наибольшее число и количество его повторений.
6) Рассортировать одномерный массив, используя метод выбора (т.е. с помощью нахождения наибольшего элемента).
7) Найти наибольший элемент среди наименьших элементов строк матрицы A[10,20].
8)Найти k-й наибольший элемент в одномерном массиве.

В такой последовательности задач решение следующей задачи основывается на предыдущих.

Комплекс взаимосвязанных задач можно использовать также при сравнении различных типов алгоритмов, а, значит, и различных операторов. Например, для демонстрации отличия циклов с известным и неизвестным количеством повторений можно предложить следующие три задачи. Найти сумму отрицательных чисел массива, если

а) массив не рассортирован;
б) массив рассортирован по возрастанию;
в) массив рассортирован по убыванию.

В первом из предложенных вариантов условия необходимо проанализировать все элементы массива. В остальных задачах количество повторений цикла зависит от элементов массива. Как только найдём неотрицательное число, можно прекращать цикл.

Для того, чтобы лучше понять отличие двух базовых управляющих структур “цикл пока” (оператор *while*) и “цикл до” (оператор *repeat* языка *pascal*), можно решить две следующее задачи. Задан массив и число x. Найти сумму чисел, предшествующих числу x, 1) не включая это число, 2) включая его. Для каждой из них можно рассмотреть два случая: 1) число x есть в массиве, 2) числа x может не быть в массиве.

Это были примеры последовательности задач, которые используются при изучении различных тем (оператор *if*, одномерные табличные величины и оператор цикла, сложные циклы, двумерные табличные величины). Такой же комплекс задач можно решать и при изучении одной темы. Например, в теме “Строки” можно рекомендовать следующую последовательность задач с возрастающей сложностью:

1) В слове найти количество буквы “а”.
2) В слове найти, какие буквы встречаются чаше других.
3) В тексте, в котором слова разделены одним пробелом, найти слова, в которых чаше всего встречается заданная буква.
4) В тексте найти слова, в которых чаще всего встречается заданная буква, если между словами один символ-разделитель (; , ! , ? и т.п.) без пробелов.
5) В тексте найти слова, в которых чаще всего встречается заданная буква, если справа и слева от символа-разделителя слов может быть любое количество пробелов.

Использование комплекса взаимосвязанных задач позволяет учесть уровень подготовленности обучаемых. В одном классе можно остановиться, например, на третьем этапе, а в другом (или с наиболее сильными учениками) можно “дойти” и до последней задачи. Начинать также можно с любой задачи. При такой методике можно предоставить ученикам возможность самостоятельно выбирать задачи соответствующего уровня. Это эффективнее случайного набора задач, разделенных по сложности на несколько уровней. Если ученик не может справиться с задачей выбранной сложности, он может, не переключаясь на постановку принципиально другой задачи, “опуститься на этаж ниже”. И, наоборот, если задача для ученика окажется простой, он имеет возможность усложнить ее,

По каждой наиболее важной теме можно подготовить несколько таких последовательностей задач. И тогда этот принцип можно использовать при подготовке индивидуальных и домашних заданий, при проведении проверочных работ. Тогда ученик в состоянии сам оценить уровень своей подготовки. Оценка в таком случае должна зависеть от того, с задачами какого уровня справился ученик.

12.5. Методические приемы, используемые при решении задач

Для более эффективного обучения ОА и П необходимо стремиться разнообразить методические приемы, используемые при решении задач. Рассмотрим некоторые из них.

Для того чтобы показать важность места ключевых слов, можно использовать следующий методический прием. Записываем два варианта алгоритма, отличающихся местоположением ключевых слов, один из которых правильный. Например:

```
алг приготовление отварной картошки
нач если картошка не очищена
то очистить картошку
        сварить картошку

все
съесть картошку
кон
```

Во втором варианте ключевое слово “все” записывается строкой выше. При анализе необходимо найти правильный вариант, объяснить, всегда ли будет неправильно работать плохой вариант.

Место ключевых слов “*нц*” – “*кц*” в циклических алгоритмах можно показать с помощью следующего примера:

```
алг завтрак1
```

нач пока в тарелке не меньше ложки каши
нц съесть ложку каши
выпить стакан чая
кц
кон

Аналогично можно поднять ключевое слово кц на строку выше.

Для сравнения с командой “если” можно привести следующий пример:

алг завтрак2
нач если в тарелке не меньше ложки каши
то съесть ложку каши
все
выпить стакан чая
кон

Исполнение и анализ следующих трех вариантов фрагмента программы помогает понять назначение ключевых слов *Begin – End* и их роль при совместном использовании операторов *For* и *If* языка *Паскаль.*

1 вариант	2 вариант	3 вариант
-------- *For i:=l to 10 Do* *If A[i]>0 then* *Begin k:=k+l;* *write (‘ k= ‘ , k)* *End*	-------- *For i:=l to 10 Do* *If A[i]>0 then* *k:=k+l;* *write (‘ k= ‘ , k)*	-------- *For i:=l to 10 Do* *Begin If A[i]>0 then* *k:=k+l;* *write (‘ k= ‘ , k)* *End*

Варианты отличаются наличием и расстановкой рассматриваемых ключевых слов. В каждом из вариантов спрашивается, что будет выведено.

Для того чтобы объяснить новый элемент языка, один и тот же пример можно решить разными способами. Например, рассмотрим следующее упражнение из темы “Логические выражения”: переменной *y* присвоить значение *x- 2* , если *3 < x < 5* и 0 в остальных случаях. Можно проанализировать следующие варианты, правильно решающие данную задачу:

1) *If x< = 3 then y: = 0 else If x < 5 then y: = x-2 else y:=0;*
2) *y:=0; If x<5 then if x>3 then y:=x-2;*
3) *If (x>3) and (x < 5) then y:=x-2 else y:=0;*
4) *If (x <=3) or (x >=5) then y:=0 else y:=x-2;*

При этом некоторые ив вариантов можно предложить школьникам записать самостоятельно.

При решении наиболее сложных задач эффективно использование следующего методического приема. Записываем один или два неправильных варианта решения одной и той же задачи, которые часто приводят школьники. Например, преобразовать массив *A[10]* следующим образом: на место первого элемента поместить 10-й, на место второго – 9-й и т. д., то есть “перевернуть массив”.

1-й вариант (неправильный).	2-й вариант (неправильный).
-------- *For i:=l to 10 Do* *A[i]:=A[11-i]*	-------- *For i:=1 to 10 Do* *begin d :=a[i];* *a[i]:=a[11-i];* *a[11-i]:=d* *end*

Проанализировав их выполнение на конкретном числовом массиве, школьники должны

исправить ошибку во втором варианте

For i:=1 to 5 Do

т. е. записать правильный вариант.

Полезно использовать упражнения на выбор наилучшего оператора или фрагмента программы для решения одной и той же задачи. Например, записан следующий оператор *if* языка *C++*:

*if (x<0) y=x*x*x*x +x*x +1; else y=x*x +x +1;*

Записать решение этого упражнения с помощью сокращенной формы *if*. Некоторые из учеников могут предложить следующий вариант:

y:=x*x+x+1; if (x<0) y=x*x*x*x +x*x +1;

Но можно записать более эффективный и “красивый” вариант:

```
if (x<0) x=x*x;
y=x*x+x+1;
```

Его могут записать лишь наиболее способные ученики.

Если с задачей большинство учеников не может справиться, полезным является следующий методический прием. Необходимо дописать пропущенный оператор или его элемент в указанном месте. Например, пусть программа выполняет циклический сдвиг элементов массива на одну позицию вправо. При записи оператора цикла значения начального и конечного параметров должны дописать школьники.

```
const n=10; int a[n]={1, -2, 33, -40, 5, 6, 7, 80, 99, 100};
r=a[n-1];
for (int i=?; i<? ; i++)
a[n-1-i]=a[n-2-i];
a[0]=r;
```

Можно предложить дописать индексы элементов массива в операторе присваивания:

a[?]=a[?];

При этом заголовок оператора цикла должен быть записан полностью.

При изучении синтаксических правил языка, для иллюстрации стиля программирования и внешнего представления данных можно применять в качестве наглядного материала заранее подготовленные распечатки текстов программ, их результатов, копий экрана. Это существенно экономит время, затрачиваемое на изучение и таких, например, вопросов, как ввод-вывод информации, использование комментариев и других. Это целесообразно использовать при рассмотрении тем, в которых существенна визуальная, наглядная сторона вопроса. Особенно большое значение это имеет в условиях безмашинного обучения или с ограниченным доступом на ЭВМ.

Так как программа лучше воспринимается иногда в целом, когда она вся “перед глазами”, то можно практиковать следующее. Вместо того чтобы писать программу на доске, можно школьникам раздать заранее подготовленные распечатки программ или продемонстрировать готовую программу на компьютере и объяснить ее. Но в то же время этим нельзя злоупотреблять. Дело в том, что при изучении некоторых тем (например, сложные циклы, сортировка массивов) важен сам процесс разработки алгоритма и написания программы, а не ее готовый текст. Важно бывает продемонстрировать учащимся, как учитель думает, рассуждает, как он доходит до того или другого алгоритма, чтобы и ученики вместе с ним размышляли и участвовали в разработке программы или алгоритма.

Почти все рассмотренные выше методические приёмы не зависят от изучаемого языка программирования.

13. Методика преподавания PascalABC

Графические возможности PascalABC изучаются в теме ОАиП после основных простейших понятий программирования (среда PascalABC, переменные и типы данных, ввод, вывод, присваивание, работа с целым типом и составление линейных программ). Изучать графические возможности PascalABC, как и в целом программирование, авторы учебников рекомендуют на конкретных примерах. Как в шестом, так и в седьмом классах, примеры графических программ используют только константы в качестве координат, хотя в 7-м классе понятие переменной уже было. Раньше при изучении переменных целого и вещественного типов изучались стандартные функции ***abs, sqr, trunc, round, frac***. При объяснении графических программ понятие процедура и функция не используются.

Необходимо обратить внимание на то, что графические команды (вместо термина “процедуры”) можно разделить на несколько видов:

- команды, задающие параметры рисунков: для пера: ***SetPenColor, SetPenWidth, SetPenStyle(psSolid, psDash, psDot, psDashDot, psDashDotDot, psClear), SetPenMode****;* для кисти: ***SetBrushColor, SetBrushStyle (bsSolid, bsClear, bsHatch, bsGradient)****;*
- команды, рисующие геометрические фигуры: ***SetPixel*** *(x, y, color: integer)****, MoveTo, LineTo, Line, Circle, Ellipse, Rectangle*** (рисует границу прямоугольника), ***FillRect*** (закрашивает прямоугольник цветом текущей кисти), ***FloodFill***(заливает замкнутую область одного цвета указанным цветом);
- команды, задающие параметры шрифта: ***SetFontSize, SetFontColor, SetFontStyle***;
- команда для вывода строки: ***TextOut*** *(x, y, color: integer);*
- команды для работы с графическим окном: ***SetWindowSize***, ***SetWindowLeft*** (отступ графического окна от левого края экрана в пикселях), ***SetWindowTop*** (отступ от верхнего края), ***SetWindowPos*** (отступ от верхнего левого края), ***SetWiydowTitle*** (заголовок), ***ClearWindow*** или ***ClearWindow (color)*** — очищает окно белым или указанным цветом, ***SaveWindow (fname: string)*** — сохраняет содержимое окна в файл с указанным именем, ***LoadWindow (fname: string)*** — загружает содержимое окна из файла с указанным именем,

Необходимо заметить, что команды для задания значения цветов, стилей пера, кисти и других параметров, а также команды рисования такие же, как в Delphi или в Builder. Существенное отличие в том, что в названных системах соответствующие процедуры и функции являются не самостоятельными, как в PascalABC, а членами класса, то есть методами.

При изучении этой темы, а также других, не обязательно по основам алгоритмизации и программирования, в которых большое количество элементов, надо уметь рассортировать их. Предлагается выделить то, что является обязательным, наиболее простым и понятным, с чего следовало бы начинать изучение темы, Например, задание размера графического окна более важно, чем команды, устанавливающие его положение на экране.

Вводятся и объясняются такие понятия, как *пиксель*, *разрешающая способность (разрешение)* экрана, *координаты пикселя*. Необходимо уделить особое внимание компьютерным координатам, так как они отличаются от геометрических. Для этого можно предложить школьникам такие упражнения: показать точку с указанными координатами (например, x=5, y=60; x=60, y=5) и наоборот, определить координаты указанной точки. На компьютере полезно синхронно с преподавателем подготовить и выполнить следующую программу рисования точки с заданными координатами.

При объяснении особое внимание надо обратить на команду *Uses GraphABC* и провести компьютерный эксперимент, задавая разные разрешающие способности (320 x 200, 800 x 600, 1024 x 768, и др.), меняя координаты точки и ее цвет. Можно нарисовать несколько точек, не используя команду цикла.

Для более одаренных детей можно эту же программу усложнить так. Для координат объявить две переменные (x, y), ввести их, и в левой половине окна точку нарисовать одним цветом (например, красным), а справа — другим. Окно можно разбить на четыре прямоугольника.

Следуя программе и учебнику, дальше надо научить рисовать линии разным цветом и

толщины, треугольник и другие многоугольники из линий, включая и прямоугольник, и окружность. При этом вначале с помощью ***Line*** и (или) ***LineTo*** рисуем только их границы, а затем показываем, как закрасить эти фигуры с помощью ***FloodFill.*** Надо учесть, что объективно это очень сложная команда, тем более для школьников. Параллельно с рисованием графические элементы подписываем. Для закрепления этих навыков в учебнике составлена программа для рисования домика, состоящего из треугольника, прямоугольника и круга.

Варианты проведения урока (части урока):

- Сначала записываем программу на доске, без компьютеров, а школьники в тетради. Затем надо дать возможность отладить эту или частично видоизмененную программу. Этот прием используется, когда рассматривается новая достаточно сложная тема (новый вопрос темы), требующая объяснений. Положительно то, что можно вернуться к элементу, который не поняли при объяснении, и индивидуально выяснить у учителя непонятные вопросы.
- Можно сэкономить время и сделать больше программ, если предложить ученикам сразу синхронно с записью на доске готовить программу на компьютере. Это непростой вид занятия, так как школьникам трудно будет сосредоточиться, чтобы слушать объяснения, комментарии учителя и одновременно работать за компьютером.
- Для закрепления материала предлагается выполнить одно общее или два (три) варианта задания одного или разных уровней сложности самостоятельно в классе. При такой форме проведения урока или его части можно обсудить алгоритм или другие общие вопросы, часто встречающиеся ошибки. Возникает *вопрос*, снижать ли оценку, если школьник при выполнении задания обращается за помощью к учителю.
- Выполнение индивидуальных заданий с выбором уровня сложности. При этом можно начинать выполнять задание в классе, а заканчивать его дома, если есть компьютер. *Вопросы:* указывать уровень сложности или дать школьнику выбрать его самостоятельно? Что делать, если школьник не справился с более сложным заданием. А как быть с теми, у кого нет дома компьютера?

Раздел 3. Методика преподавания основных тем

Одной из наиболее заметных тенденций в развитии школьной информатики является увеличение места информационных технологий в ее содержании. На начальном этапе преподавания этому уделялось меньше внимания, так на большинстве доступных школам компьютеров отсутствовало соответствующее программное обеспечение. Из разных подходов к изложению этих вопросов в последние годы все больше применяется подход, при котором изучаются наиболее популярные, широко используемые на практике конкретные программные продукты для определенных типов ЭВМ и операционных систем: графические и текстовые редакторы, средства презентации, СУБД, электронные таблицы, сетевые информационные технологии. По каждой из этих систем необходимо изучить следующие группы вопросов:

- области применения;
- теоретические основы, которые включают в себя вопросы, связанные с различными типами данных и их структурированием, методы решения информационных задач с помощью технологических средств данного типа;
- технологическое содержание или прикладные программные средства содержат среду, различные режимы работы, систему команд и данные;
- аппаратные средства.

14. Методика преподавания текстовых и графических редакторов

Эти вопросы, как правило, изучаются на начальном этапе изучения информатики: текстовый редактор в 6-м и 8-м классах, а работу с графической информацией и соответствующие редакторы — в 6, 8 и 9 классах. Выбор текстового и простейшего графического редакторов,

используемых в учебных целях, учебной программой по школьной информатике, как правило, не регламентируется и зависит от технического и программного обеспечения школьного компьютерного класса. В программе перечисляются только вопросы, которые необходимо изучить, а право выбора остается за учителем. Для учебных целей не обязательно использовать профессиональный, достаточно громоздкий *Word*, в котором многие элементы его интерфейса оказываются невостребованными и создают определенный информационный шум. Все приемы работы можно изучить на примере более простого текстового редактора *WordPad,* интерфейс которого близок к интерфейсу. Освоив программные вопросы на примере *WordPad*, в конце данной темы можно познакомить учащихся с некоторыми возможностями *Word,* которых не было в более простом редакторе (например, проверка орфографии, работа с таблицами и др.).

Необходимо обратить внимание на то, что компьютер с работающим текстовым редактором является специализированным исполнителем для работы с текстовой информацией, который работает под управлением человека. Все его действия (например, ввод символа, копирование фрагмента текста, сохранение текста и т. п.) можно рассматривать как команды управления исполнителем. Отсюда следует, что изучение текстового редактора, как и любого другого прикладного средства информационных технологий, можно проводить по традиционной методической схеме, свойственной изучению всякого исполнителя: данные, среда редактора, режимы работы, система команд.

Данные. Текстовый редактор работает с символьной информацией, в которой можно выделить следующие виды структур: символы, слова, строки символов, фрагменты текста (блоки), файлы. Имеются команды для работы с каждой из этих совокупностей данных.

Среда ТР. Независимо от выбранного конкретного ТР необходимо, прежде всего, выделить и обратить внимание школьников на те общие элементы среды, которые характерны для любых ТР. Это рабочее поле, строка состояния, строка меню и др.

Режим работы. Под этим понимается определенное состояние исполнителя (в данном случае редактора), в котором возможно выполнение определенного вида работы. Это ввод и редактирование текста, его форматирование, орфографическая проверка, обращение за подсказкой, обмен с ВЗУ, печать и др.

Система команд ТР. Ее можно условно разделить на следующие группы: команды интерфейса, или работа с меню; команды изменения состояния ТР: нижний/верхний регистры, режим вставки / замены и т.д.; команды перемещения по тексту, редактирования, форматирования и др.

Практическую работу по изучению ТР необходимо начинать с того, что ученики должны освоить приемы основного стандарта редактирования, к которым относятся: перемещение по тексту, удаление, вставка, замена символа; разрыв и слияние строк.

Существует множество прикладных программ, предназначенных для работы с **графикой**. Как и при изучении работы с текстом, нигде в учебной программе по информатике не говорится о конкретном программном средстве. Поэтому возникает естественный вопрос, какой графический редактор выбрать для изучения. Этот термин применяется по отношению к прикладным графическим программам, не имеющим какой-либо специализированной ориентации и используемым для произвольного рисования или редактирования сканированных изображений. В соответствии с двумя принципами представления графической информации — растровым и векторным — графические редакторы делятся на такие же два типа. Профессиональные графические редакторы, такие как CorelDraw, Photoshop — дорогие программные продукты, и не всем школам доступны. С точки зрения учебных целей, стоящих перед базовым курсом информатики, вполне достаточно редактора *PaintBrush* или *Paint*, который стал результатом развития первого. Такой растровый редактор позволяет ученикам наглядно продемонстрировать дискретную структуру рисунка, дает возможность воздействовать на каждый отдельный элемент при увеличении масштаба в режиме прорисовки.

Особенность изучения компьютерной графики в том, что это такая область информационных технологий, которую ученикам хочется реально увидеть, а не слушать разговоры о ней. Поэтому большое внимание необходимо уделять демонстрации на компьютере

разнообразных красочных рисунков, схем, чертежей, диаграмм, образцов анимационной и трехмерной графики и других графических продуктов. Учителю надо не забыть сказать и о том, что любимые многими из них компьютерные игры в большинстве имеют графический интерфейс, причем достаточно сложный. Чтобы поддержать интерес к математике, школьникам надо сказать и о том, что для получения на компьютере трехмерного реалистического изображения часто надо выполнить сложные математические расчеты. К теоретическому содержанию этой темы относятся вопросы представления изображения в памяти компьютера.

Изучение графического редактора, как и любого другого, можно проводить по той же методической схеме, что и текстового редактора.

Данные. Итоговой информацией является созданный рисунок, который с позиций растровой графики представляет собой совокупность разноцветных пикселей. Значит, данными для графического редактора являются цвета. Кроме технологии работы с ними необходимо объяснить, как они представляются в памяти ЭВМ.

Среда ГР. Пользовательский интерфейс большинства ГР организуется единообразно. Поэтому необходимо, прежде всего, выделить и обратить внимание школьников на те общие элементы среды, которые характерны для любых ГР. Кроме панели инструментов, строки состояния, рабочего поля и меню, есть такие, которые присущи только графическим редакторам. Это панель палитры для выбора цвета, калибровочная шкала для установки ширины кисти, резинки и других рабочих инструментов.

Режим работы. Он определяет возможные действия пользователя, а также команды, которые он может отдавать редактору в данном режиме. Можно выделить следующие режимы: работа с рисунком (рисование), выбор и настройка инструмента, выбор рабочих цветов, работа с внешними устройствами. Для последнего характерно не только работа с файлами, а и сканирование рисунков, их печать.

Система команд ТР. Ее можно условно разделить на следующие группы: команды интерфейса, или работа с меню; команды изменения состояния ТР: нижний/верхний регистры, режим вставки / замены и т.д.; команды перемещения по тексту, редактирования, форматирования и др.

Методика изучения простейших приемов работы: ввод, сохранение и загрузка, редактирование и форматирование текста. Особенности изучения технологии обработки текстовых документов: создание и форматирование списков, таблиц, колонок; вставка декоративного текста, рисунков, формул; подготовка к печати. Методика обучения школьников работе в графическом редакторе Paint и в векторном графическом редакторе.

15. Методика преподавания информации

15.1. Методика изучения понятия информации, ее видов и свойств.

Блок вопросов, связанных с информацией, охватывает содержание всего курса, поскольку понятие информации является в нем центральным. Это должно быть очевидным хотя бы потому, что с этим термином связано название предмета. Каждое из понятий, связанных с информацией, рассматривается в информатике в двух аспектах: назовем их условно бескомпьютерным и компьютерным. Под первым из них понимается рассмотрение информации без привязки к компьютеру, с общих позиций, по отношению к человеку, обществу, природе. Это такие вопросы, как определение и измерение информации, информационные модели и процессы, а также процессы управления в природе и обществе. Под компьютерным аспектом понимается изучение информационной стороны функционирования самого компьютера, изучение компьютерных технологий хранения, передачи и обработки информации, а также методов программирования.

Однако если проанализировать в историческом плане сначала советские, а позже российские и белорусские учебники и учебные пособия по школьной информатике, то можно сделать вывод, что далеко не в каждом из них информации уделяется главное внимание. Этому есть две причины. В первых учебниках [1 – 5] центральными понятиями и объектами изучения являются алгоритм и компьютер, а информация упоминается лишь вскользь и в основном

определяется на интуитивном уровне. Вторая причина — в объективной сложности понятия информация. Оно относится к числу фундаментальных в науке, носит философский характер и является предметом постоянных научных дискуссий. Возникает необходимость обсуждения на уроках сложной проблемы определения информации, но в средних учебных заведениях это можно делать только языком, доступным для детей.

Кратко проанализируем варианты определения информации в различных учебниках. В учебнике А. П. Ершова и др. [1, 2] такого определения нет вообще. Почти все его содержание посвящено разбору вопросов, что такое ЭВМ и что такое алгоритм. В учебнике второго поколения А. Г. Кушниренко и др.[16] утверждается, что этот термин в информатике является первичным, неопределяемым, подобно тому, как отсутствует строгое определение прямой и точки в планиметрии.

Достаточно много внимания уделено этому понятию в учебнике А. Г. Гейна и др. [5]. Сначала в первой главе читаем следующее: «Информация — постоянный спутник человека. Это те сведения, которые помогают ориентироваться нам в окружающем мире». Далее это понятие лишь используется, а в предпоследней главе авторы снова возвращаются к его определению. «В интуитивном, житейском смысле под информацией понимают сведения, знания и т. п., которые кого-либо интересуют. И чем интереснее сообщаемые сведения, тем больше информации (с житейской точки зрения) в них содержится». Дальше дается другое определение информации с технической точки зрения. «Когда речь идет об автоматической передаче информации, ее хранении и переработке, информация — это произвольная последовательность символов, т. е. любое слово; каждый новый символ увеличивает количество информации». На универсальность претендует определение информации В. А. Каймина и др. [3]. «Информация в наиболее общем определении — это отражение предметного мира с помощью знаков и символов».

Для истории литературы по школьной информатике значительным событием стал выход в России книги «Информатика. Энциклопедический словарь для начинающих»[17]. В ней впервые в современном на то время духе отражено все разнообразие предметной области информатики, ее научное содержание, сделана попытка объединить «человеческую» и «техническую» позиции по отношению к информации. Дано следующее определение: «Информация — это содержание сообщения, сигнала, памяти, а также сведения, содержащиеся в сообщении, сигнале или памяти». Недостаток в том, что просматривается тавтология: в чем разница между содержанием и содержащимися сведениями? Вряд ли это можно понять самому учителю, а тем более объяснить ученику.

Из всего сказанного можно сделать следующий ***вывод***. Если в центр содержания предмета «информатика» ставить информацию, а не алгоритм, компьютер, то обойти вопрос об определении информации нельзя. Бесспорно и то, что дать единое, универсальное определение информации нельзя. И в науке, и на практике известны различные подходы к информации, и в рамках каждого из них дается определение этого понятия. Ученики должны знать, что в зависимости от контекста, в котором используется термин «информация», он может нести разный смысл.

При раскрытии понятия «информация» с точки зрения субъективного (бытового, человеческого) подхода, следует отталкиваться от интуитивных представлений об информации, имеющихся у детей. Не следует сразу требовать от них определения этого термина. Целесообразно начать с беседы в форме диалога, чтобы подвести их к этому определению с помощью простых понятных вопросов.

— Откуда вы получаете информацию?

— Из книг, радио, телепередач, Интернета, от друзей.

Дальше попросите учеников привести примеры какой-нибудь информации, которую они получили в этот день. Например, кто-нибудь скажет:

— Утром по радио я слышал прогноз погоды.

Ухватившись за такой ответ, учитель подводит учеников к выводу:

— Значит, вначале ты не знал, какая будет погода, а после прослушивания радио стал

знать. Следовательно, получив информацию, ты получил новые знания.

Таким образом, учитель вместе с учениками приходит к определению: информация для человека — это знания, которые он получает от различных источников. На простых, понятных, известных детям примерах следует закрепить это определение. Аналогично можно придти к выводу, что информация — это содержимое нашей памяти, которая является средством хранения знаний. Но информацию можно хранить не только в собственной памяти, в голове, но и в записях на бумаге, на магнитных носителях и пр. Такую информацию можно назвать внешней. Чтобы ей можно было воспользоваться, он должен сначала ее найти, прочитать, т. е обратить ее во внутреннюю форму, а затем производить какие-то действия. Этим самым учитель подводит школьников к пониманию того, что у компьютера, подобно человеку, есть внутренняя и внешняя память. Использование дидактического приема аналогии между информационной функцией человека и компьютера позволит ученикам лучше понять суть устройства и работы ЭВМ.

15.2. Методика изучения измерения информация.

Проблема измерения информации напрямую связана с проблемой ее определения, поскольку сначала надо уяснить, ***что*** собираемся измерять, а потом — ***как*** это делать, какие единицы использовать. Поэтому просматривается следующая цепочка понятий: информация, сообщение, информативность сообщения, единица измерения информации, информационный объем сообщения. Под сообщением понимается информационный поток, который в процессе передачи информации поступает к принимающему его субъекту. Сообщение — это и речь, которую мы слышим; и то, что видим зрительно (фильм, сигнал светофора и т.п.); и текст, который читаем и т. д. Вопрос об информативности сообщения следует обсуждать на примерах, предлагаемых учителем и учениками. Нельзя отождествлять понятия «информация» и «информативность сообщения». Следующий пример показывает различие этих понятий. Вопрос: «Будет ли информативным текст вузовского учебника по высшей математике для первоклассника, если он попытается его прочитать? Иначе говоря, может ли первоклассник с помощью такой литературы пополнить собственные знания?» Очевидно, что ответ отрицательный, хотя, с другой стороны, такой учебник содержит определенную информацию, но для первоклассника она непонятна.

При объяснении этой непростой темы можно предложить ученикам поиграть в своеобразную викторину. Учитель предлагает детям вопросы, а ученики отвечают на них письменно или ставят знак вопроса, если ответ не знают. После этого учитель дает правильные ответы, а ученики отмечают, какие из них оказались информативными, а какие — нет. Для последних нужно указать причину отсутствия информации: не новое, непонятное. Например, можно задать следующие вопросы: Какой город является столицей Франции? Что изучает коллоидная химия? Какую высоту и вес имеет Эйфелева башня? Большинство учеников ответят лишь на первый вопрос. Учитель должен дать ответы на остальные два вопроса. Информативным является лишь последнее сообщение. Первое из них не является новым, а второе — непонятное для шестиклассника. Если сообщение неинформативно для человека, то количество информации в нем, с точки зрения этого человека, равно нулю, а в информативном сообщении количество информации больше нуля.

Характерным для ряда учебников является следующий прием. Обсуждая вопрос об измерении информации, тут же переходят к описанию компьютерного представления информации в форме двоичного кода (бита). Во многих учебниках понятие бита объясняется в основном одинаково. Информация кодируется с помощью последовательности сигналов всего двух видов: намагничено или не намагничено, включено или выключено, высокое или низкое напряжение и т. д. Затем утверждается, что количество информации равно количеству двоичных цифр (битов) в таком коде.

Бит — основная единица измерения информации. Следует обратить внимание учеников на то, что в любой метрической системе существуют основные (эталонные) и производные от них единицы (метр — сантиметр, километр; грамм — килограмм, миллиграмм). Так подходим к определению байта, килобайта, мегабайта и гигабайта. Обязательно надо заострить внимание, что килобайт больше за байт не в 1000 раз, как килограмм больше грамма, а в 1024 раза.

15.3. Методика изучения информационных процессов.

Понятие ***«информационные процессы»***, так же как и понятие «информация», является базовым в курсе информатики. Сначала на простых понятных примерах надо показать, что это такое. Это получение информации из средств СМИ, процесс обучения, принятие управляющих решений, разработка какого-нибудь проекта, документооборот на предприятии, сдача экзамена и др. — вот примеры информационных процессов, с которыми нам приходится постоянно иметь дело. Существуют три основных их типа: хранение, передача и обработка информации, Начинать рекомендуется с рассмотрения этих процессов без привязки к компьютеру, т. е. применительно к человеку. Затем, при изучении архитектуры ЭВМ, компьютерных информационных технологий речь пойдет о реализации тех же информационных процессов с помощью компьютера.

С ***процессом хранения*** информации связаны следующие понятия: *носитель информации (память), внутренняя и внешняя память, хранилище информации.* Их следует изучать, проводя аналогию с человеческой памятью. Биологическую память (мозг) можно рассматривать *внутренней*, так как ее носитель — мозг — находится внутри нас. Ее можно назвать также *оперативной* или быстрой, так как заученные знания воспроизводятся человеком мгновенно. Все прочие виды носителей информации можно назвать *внешними* (имеется в виду по отношению к человеку). Необходимо показать их примеры в историческом плане: камень, дерево, папирус, бумага и, наконец, магнитные, оптические и другие современные виды носителей информации.

Приведя примеры *хранилищ информации* (библиотека, справочники, картотеки и др.) и рассказав об их основных информационных единицах (анкета, книга, отчет и т.п.), можно дать следующее определение этого термина. Это определенным образом организованная информация на внешних носителях, предназначенная для длительного хранения и постоянного использования. Надо объяснить, что понимается под организацией хранилища (наличие определенной структуры, т. е. упорядоченность, классификация и т.п.) и для чего она нужна (для удобства добавления, удаления и поиска информации). Организованные определенным образом хранилища данных на устройствах внешней памяти компьютера принято называть *базами данных*. Подробнее это изучается в соответствующей теме.

Рассматривая человеческую память и то, что хранится, например, в библиотеке, надо привести сравнение внутренней и внешней памяти (объем, время доступа, надежность и др.).

Изучая ***процесс обработки*** информации, необходимо рассмотреть следующие вопросы: общая схема процесса обработки, постановка задачи обработки, исполнитель обработки, алгоритм обработки, типовые задачи обработки информации.

В процессе обработки информации решается следующая задача: дан некоторый набор исходных данных — *исходная* информация; требуется получить некоторые результаты — *итоговую* информацию. Тот объект или субъект, который осуществляет обработку, можно назвать *исполнителем* обработки. Поэтому общую схему обработки информации можно представить в следующем простом и понятном для школьников виде:

Исходная информация —> Исполнитель обработки —> Итоговая информация.

Описание последовательности действий, которую нужно выполнить, чтобы достичь нужного результата, в информатике принять называть *алгоритмом обработки*. Подробно понятие алгоритма и вопросы алгоритмизации изучаются в соответствующем разделе. Здесь же обращается внимание на то обстоятельство, что одно из фундаментальных понятий школьной информатики, понятие алгоритма исходит от другого базового понятия — понятия информации и, в частности, понятия информационных процессов.

В форме беседы с учениками необходимо рассмотреть примеры ситуаций, связанных с обработкой информации, и на этих примерах показать выше перечисленные понятия. Все примеры обработки информации можно разделить на **три типа**. Первый тип заключается в ***получении*** новой информации, нового содержания знаний. К ним относится, например, решение математических задач; действия следователя, который по набору улик находит преступника; разгадывание ученым историком тайн древних рукописей и т. п. Ко второму типу обработки информации относится ***обработка***, в результате которой меняется форма, а не содержание информации. К этому типу можно отнести перевод текста с одного языка на другой, кодирование

информации, ее структурирование, связанное с внесением определенного порядка, определенной организации в хранилище информации (сортировка, группировка по некоторым признакам классификации, табличное или графическое представление и т. п.). Можно выделить третий тип обработки — ***поиск информации***. Задача поиска формулируется так: в хранилище информации (телефонный справочник, словарь, расписание поездов и т.п.) найти нужную информацию, удовлетворяющую определенным условиям поиска (номер телефона конкретного товарища, английский перевод конкретного слова, время отправления поезда в определенный пункт назначения). Эти и другие примеры поиска, как и примеры других хранилищ информации должны привести сами школьники.

Ключевыми понятиями в описании ***процесса передачи информации*** являются источник и приемник информации, информационный канал. Необходимо это изобразить схематически. На естественных примерах разговора, чтения можно обратить внимание на то, что в этом процессе информация представляется и передается в форме последовательности сигналов, символов, знаков, показать, что является каналом связи (акустические волны в атмосфере, световые электромагнитные волны). Но надо не забывать, что такие примеры доступны старшеклассникам после изучения соответствующих тем в физике. Для младшего возраста надо придумать еще более простые примеры. Если в процессе передачи информации используются технические средства связи, то их называют каналами передачи информации (информационными каналами). К ним относятся знакомые всем телефон, радио, телевидение.

В этом вопросе можно рассмотреть также такие понятия, как кодирование и декодирование информации, шум и приемы защиты от него, скорость передачи информации и пропускная способность канала. При обсуждении последних двух вопросов можно привлечь аналогию с процессом перекачки воды по водопроводным трубам. Можно также предложить творческое задание: определить собственную скорость восприятия информации при чтении вслух и про себя.

15.4. Методика изучения представления информации.

Тема ***«системы счисления»*** является смежной с математикой, имеет прямое отношение к математической теории чисел. Но в школьном предмете математики она, как правило, не изучается. Необходимость включения ее в информатику связана с тем фактом, что числа и символы в памяти компьютера представляются в двоичной системе счисления, а для внешнего представления содержимого памяти, адресов памяти используют шестнадцатеричную систему.

В первом учебнике [1, 2]рассматриваемое понятие не упоминается совсем. Говорится лишь о том, что вся информация представляется в двоичном виде. То же самое можно сказать и про учебник [4]. Среди учебников второго поколения наибольшее внимание системам счисления уделено в книге [5]. В отдельном параграфе дано следующее определение: «Система счисления — способ записи чисел с помощью заданного набора специальных знаков (цифр)». Такое определение затрагивает только алфавит, синтаксис и семантику языка чисел. Более полное определение дано в [18]: «Система счисления —способ изображения и соответствующие ему правила действий над числами». Под правилами действий понимаются способы выполнения арифметических вычислений в рамках данной системы счисления. Их можно назвать прагматикой языка чисел. В качестве дополнительной литературы, при изучении этой темы на факультативах и других дополнительных занятиях можно рекомендовать учебное пособие [19], посвященное этой теме и раскрывающей ее наиболее полно.

В этой теме необходимо рассмотреть следующие вопросы:

- позиционные и непозиционные системы счисления;
- основные понятия позиционных систем: основание, алфавит;
- развернутая форма представления чисел в позиционных системах;
- перевод чисел из одной системы в другую;
- особенности двоичной арифметики;
- связь между двоичной и шестнадцатеричной системами.

С методической точки зрения эффективно подвести учеником к самостоятельному, хотя и маленькому, открытию. Желательно, чтобы ученики сами сформулировали различие между

позиционным и непозиционным принципом записи чисел. Для этого пишем на доске два числа, например, *XXX* (римское число тридцать) и *333* (арабское число триста тридцать три). В форме диалога выясняем, что означает каждая цифра в каждом из примеров и как получается значение каждого из чисел. Из этого диалога следуют отличия предложенных двух принципов записи чисел. Закрепив это на других примерах, можно ввести следующее определение: «Система счисления — это определенный способ представления чисел и соответствующие ему правила действий над числами». Предложите выполнить, например, умножение двух трехзначных чисел в одной и другой системах счисления. Выполняя это задание, ученики должны понять, что в непозиционной римской системе счисления, в отличие от позиционной арабской нет простых и понятных правил для выполнения вычислений с многозначными числами.

Теперь также в форме беседы нужно дать понять ученикам, что позиционных систем существует множество, и отличаются они друг от друга *алфавитом* — множеством используемых цифр. Размер алфавита или количество используемых цифр — *основание системы счисления*. Далее желательно попросить учеников дать значения этих терминов для десятичной с.с., вместе разобраться, почему она так называется, привести примеры других позиционных с.с.. Особое внимание необходимо уделить двоичной, восьмеричной и шестнадцатеричной с.с.

Прежде чем выполнять вычисления в различных с.с., рекомендуется сначала научить учеников записывать натуральный ряд чисел в различных позиционных с.с.. При этом желательно проводить сравнение с десятичной с.с. Показываем как с помощью индексом записать число в различных системах счисления. Но здесь нужно сделать следующее важное замечание: ни в коем случае нельзя называть недесятичные числа так же, как десятичные. Например, 36_8 — это не тридцать шесть, а три — шесть восьмеричное; 101_2 — это не сто один, а один — ноль — один двоичное

Методика изучения видов и свойств информации, ее измерения и представления в памяти, носителей информации, вирусов, информационных моделей, систем, технологий и процессов.

ЛИТЕРАТУРА

1. Основы информатики и вычислительной техники: Пробное учебное пособие для сред. учеб. заведений: В 2 ч. / Под ред. А. П. Ершова и В.М. Монахова.Ч.1. — М.: Просвещение, 1985.
2. Основы информатики и вычислительной техники: Пробное учебное пособие для сред. учеб. заведений: В 2 ч. / Под ред. А. П. Ершова и В.М. Монахова Ч. 2. — М.: Просвещение, 1986.
3. Каймин В. А. и др. Основы информатики и вычислительной техники. — М.: Просвещение, 1989.
4. Кушниренко А.Г., Лебедев Г.В., Сворень Р.А. Основы информатики и вычислительной техники. — М.: Просвещение, 1990, 1993, 1996.
5. Основы информатики и вычислительной техники./ А.Г.Гейн, В.Г.Житомирский, Е.В.Линецкий и др. — М.: Просвещение, 1991, 1993.
6. Ковалев М. М., Курбацкий А. Н., Листопад Н.И.. Концепция информатизации образования. – Мн.: М-во образования РБ, 1991.
7. Кузнецов А. А. О разработке стандарта школьного образования по информатике. ИнфО. – 1994. – № 1.
8. Быкадорау Ю. А., Кузняцоу А. Т., Насеннікава Л. М. Праграма базавага курса “Інфарматыка ”(8 – 9 класы) // Настаўніцкая газ. – 1994. – 20 ліп.
9. Быкадоров Ю. А., Кузнецов А. Т., Павловский А.И.. Информатика. – Мн.: Нар. асвета, 1994, 1995.
10. Информатика./ А. Г. Гейн, Е. В. Линецкий, М. А. Сапир, М.Ф. Шолохович. – М.: Просвещение, 1994.
11. Каймин В., Завальский Ю. Экспериментальная программа по курсу “Основы ИВТ” // ИнфО. – 1991– № 6.
12. Лапчик М. П. Методика преподавания информатики: Учеб. пособие для физ.-мат. факультетов пед. вузов. – Свердловск, 1987.

13. М. П. Лапчик и др. Методика преподавания информатики: Учеб. пособие для студ. пед. вузов. – М.: Издательский центр «Академия», 2005.
14. Бочкин А. И. Методика преподавания информатики: Учеб. пособие. – Мн.: Выш. шк., 1998.
15. Программа курса ОИиВТ. // Микропроцессорные средства и системы. – 1986. – № 2 (см. также: Математика в школе. – 1986.– №3).
16. Кушниренко А.Г., Лебедев Г.В., Сворень Р.А. Основы информатики и вычислительной техники: Учеб. для 10 – 11 кл.сред.шк. — М.: Просвещение, 1996.
17. Информатика. Энциклопедический словарь для начинающих. — М.: Педагогика-Пресс, 1994.
18. Словарь школьной информатики / Сост. А.П.Ершов // Математический энциклопедический словарь. — М.: Сов. Энцикл., 1988.
19. Андреева Е, Фомина И. Системы счисления и компьютерная арифметика. — Лаборатория Базовых Знаний, 1999.

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

№ № пп/п	Наименование разделов, тем	Количество часов			
		Аудиторные			Самост. работа
		Лекции	Практич., семинар.	КСР.	
	Введение	**2**		-	
	Раздел 1. Теоретические основы методики преподавания информатики	**12**	**6**	**2**	**28**
1.1.	Тема 1. Дисциплина «Методика преподавания информатики» в системе педагогических знаний	1		-	
1.2.	Тема 2. Цели преподавания информатики	1	-	-	
1.3.	Тема 3. Принципы дидактики и преподавание информатики	1	-	-	
1.4.	Тема 4. Методы обучения информатике	1		-	
1.5	Тема 5. Организационные формы обучения информатике	1		-	
1.6.	Тема 6. Технические и программные средства обучения информатике	1	1	-	
1.7.	Тема 7. Урок информатики	3	1	-	12
1.8.	Тема 8. Оценка знаний по информатике	1	1	-	6
1.9.	Тема 9. Содержательная характеристика учебных программ по информатике, учебников и учебных пособий.	2	3	2-	10

2	**Раздел 2. Методика преподавания основ алгоритмизации и программирования**	**18**	**14**	**4**	**40**
2.1.	Тема 1. Методические особенности преподавания основ алгоритмизации и программирования.	2		-	
2.2.	Тема 2. Методика преподавания алгоритмов.	2		-	
2.3.	Тема 3. Общие вопросы методики преподавания языка программирования.	2	0	-	
2.4.	Тема 4. Методика преподавания начал программирования.	2	2	-	2
2.5.	Тема 5. Методика преподавания ветвления и циклов.	2	2		2
2.6.	Тема 6. Методика преподавания массивов	1	1	1	4
2.7.	Тема 7. Методика преподавания символов и строк	1	1	1	6
2.8.	Тема 8. Методика преподавания методов программирования на кружках, факультативах, в лицеях, и т. п.	4	6	2	14
2.9.	Тема 9. Методика преподавания основ веб-конструирования.	2	2	-	12
3	**Раздел 3. Методика преподавания основных тем**	**2**	**6**	**2**	**10**
3.1.	Тема 1. Методика преподавания вопросов, связанных с информацией.	1		-	
3.2.	Тема 2. Методика преподавания аппаратного и системного программного обеспечения.	1		-	
3.3.	Тема 3. Методика преподавания текстовых и графических редакторов.	-	1	1	2
3.4.	Тема 4. Методика преподавания компьютерных презентаций.	-	1		2
3.5.	Тема 5. Методика преподавания электронных таблиц и систем управления базами данных.	-	2	1	6
3.6.	Тема 6. Методика преподавания коммуникационных технологий.	-	1	-	
3.7.	Тема 7. Методика преподавания сети Интернет.	-	1	-	
ИТОГО	**146**	**34**	**26**	**8**	**78**

Учебно-методическая карта

Номер раздела, темы, занятия	Название раздела, темы, занятия; перечень изучаемых вопросов	Количество аудиторных часов				
		лекции	практические (семинарские) занятия	лабораторные занятия	контроемая лиру	М
	Введение. Понятие информатики. История преподавания школьной информатики, её современное состояние. Краткое содержание предмета.	2	0			на Метод)) програ инфор 11клас
1	**Раздел 1.Теоретические основы преподавания информатики**	**12**	**6**		**2**	
1.1.	***Тема 1. Дисциплина «Методика преподавания информатики» в системе педагогических знаний.*** Цели и задачи учебной дисциплины. Место дисциплины, связи с другими дисциплинами. Требования к усвоению дисциплины. Объём дисциплины. Средства диагностики и критерии оценок. Другие особенности дисциплины.	1	0			на Метод)) Типова методи
1.2.	***Тема 2. Цели преподавания информатики.*** Значение предмета в учреждениях общего среднего образования. Уровни работы с компьютером. Цели преподавания школьной информатики как единство образования, развития и воспитания. Компьютерная грамотность, образованность и информационная культура.	1	0			на Метод)) Типова методи
1.3.	***Тема 3. Принципы дидактики и преподавание информатики.*** Принцип научности. Сознательность усвоения и деятельности. Доступность и наглядность содержания. Активность и самостоятельность. Индивидуализация и коллективность обучения. Эффективность учебной деятельности. Связь теории и практики.	1	0			на Метод
1.4	***Тема 4. Методы обучения информатике.*** Особенность методов и их реализация на практике. Традиционные и словесно-фронтальные методы.	1	0			на Метод

	Мыслительные операции и работа на ЭВМ. Индукция, дедукция и аналогия. Анализ и синтез.					
1.5.	***Тема 5. Организационные формы обучения информатике.*** Организационные формы в зависимости от числа участников и от того, кто управляет обучением. Основные организационные формы: лекция, рассказ, беседа, семинар, лабораторная работа, индивидуальный практикум. Организация внеклассной работы. Вспомогательные организационные формы: экскурсия, факультативное занятие, кружок, олимпиада.	1	0			на Метод
1.6.	***Тема 6. Технические и программные средства обучения информатике.*** Компьютерные классы, программное обеспечение, использование современных технических средств обучения, дидактических и справочных материалов. Преимущества и недостатки программированного обучения.. Специфика работы учителя в условиях локальной компьютерной сети, наличия электронной почты и Интернета.	1	1			на Метод
1.7.	***Тема 7. Урок информатики.*** Методические аспекты повышения его эффективности. Подготовка учителя к уроку. Дифференцированный и индивидуальный подходы при обучении информатике. Принципы подбора индивидуальных и групповых заданий. Особенности занятий без компьютеров и с их использованием. Управление и контроль за работой обучаемых.	3	1			на Метод
1.8.	***Тема 8. Оценка знаний по информатике.*** Методы контроля знаний, их особенности. Текущий контроль, проведение и проверка самостоятельных и контрольных работ. Итоговые зачётные занятия. Электронное и бумажное (безмашинное) тестирование: типы тестов, их подготовка, проведение тестирования, методы и критерии оценки результатов тестирования, преимущества и недостатки тестирования. Экзамен по информатике.	1	1			тести
1.9-	***Тема 9. Содержательная характеристика учебных программ по информатике, учебников и учебных пособий.*** Исторический анализ содержания предмета «Информатика» для каждого класса. Современное состояние, основные направления модернизации образования в области информатики. Анализ учебников и учебных пособий по информатике для учреждений общего среднего образования: исторический взгляд, действующие учебники и пособия, перспективы.	2	3		2	учебны инфор Учебна предме (6 – 11 Типова методи
2	**Раздел 2. Методика преподавания основ алгоритмизации и программирования**	**18**	**14**		**2**	
2.1.	***Тема 1. Методические особенности преподавания основ алгоритмизации и программирования.*** Связь раздела с математикой и другими школьными предметами. Реализация методических принципов при изучении основ алгоритмизации и программирования. Принцип многоуровневости, предварительной мотивации, параллельности. Сравнение и повторение, цикличность. Принцип индивидуальных заданий.	2				на Метод
2.2.	***Тема 2. Методика преподавания алгоритмов.*** Как ввести и объяснить основные понятия алгоритмизации:	2				на

	алгоритм и его свойства, исполнитель алгоритма и его команды, программа. Особенности изучения типов алгоритмов: линейные, ветвления, циклы. Использование бытовых алгоритмов. Блок-схемы и другие способы описания алгоритмов.					Метод
2.3.	***Тема 3. Общие вопросы методики преподавания языка программирования.*** Формы занятий. Сравнительный анализ систем программирования *Pascal, PascalABC, Delphi, C++, Builder*. Типы задач. Требования к задачам, принципы их выбора. Вспомогательные упражнения, тренировочные тесты для самоподготовки, общие и индивидуальные задания на программирование контрольные тесты. Методические приёмы, используемые при решении задач. Комплекс взаимосвязанных задач.	2				Сем заня
2.4.	***Тема 4. Методика преподавания начал программирования*** Линейные программы с использованием исполнителя Чертежник языка *PascalABC,* методика его изучения и использования. Как ввести и объяснить основные понятия программирования.	2	2			Сем заня
2.5.	***Тема 5. Методика преподавания ветвления и циклов.*** Как ввести и объяснить понятия ветвления и цикла. Методика преподавания простых и составных условий, логических операций, условного оператора, операторов цикла с параметром и предусловием. Сравнительный анализ ветвления и цикла в разных языках программирования.	2	2			Сем заня
2.6.	***Тема 6. Методика преподавания массивов.*** Как ввести и объяснить следующие понятия: массив, его объявление, индекс, размер массива, ввод и вывод. Методика решения различных типов задач для работы с одномерными массивами. Как ввести понятие двумерных массивов (таблиц).	1	1		1	Сем заня
2.7	***Тема 7. Методика преподавания символов и строк.*** Как ввести и объяснить понятия символ и строка. Методика использования стандартных функций и процедур для работы с символами и строками. Методика решения различных типов задач для работы с символами и строками**.**	1	1		1	Сем заня
2.8	***Тема 8. Методика преподавания методов программирования на кружках, факультативах, в лицеях, специализированных классах и школах.*** Как ввести и объяснить следующие понятия: процедура, функция, модуль, класс, поля и методы класса, объект. Методика преподавания процедур и функций, основ современного объектно-ориентированного программирования. Методика преподавания современной визуальной системы программирования *Builder*: окна, работа с компонентами, событийное программирование, меню.	4	6		2	Сем заня
2.9	***Тема 9. Методика преподавания основ веб-конструирования.*** Методика преподавания различных методов разработки веб-сайтов для различных предметных областей: использование офисных приложений, использование языка разметки HTML, веб-конструирование в редакторе *FrontPage,* подготовка изображений для Интернета.	2	2			Сем заня

3	**Раздел 3. Методика преподавания основных тем**	**2**	**6**		**2**	
3.1.	***Тема 1. Методика преподавания вопросов, связанных с информацией.*** Методика изучения видов и свойств информации, ее измерения и представления в памяти, носителей информации, вирусов, информационных моделей, систем, технологий и процессов.	1				учебны инфор Учебна предме (6 – 11
3.2	***Тема 2. Методика преподавания аппаратного и системного программного обеспечения.*** Методика изучения устройства компьютера и приемов работы на нем. Методика изучения операционной системы *Windows* и работы с файлами.	1				учебны инфор Учебна предме (6 – 11
3.3	***Тема 3. Методика преподавания текстовых и графических редакторов.*** Методика изучения простейших приемов работы: ввод, сохранение и загрузка, редактирование и форматирование текста. Особенности изучения тем: создание и форматирование списков, таблиц, колонок; вставка декоративного текста, рисунков, формул; подготовка к печати. Методика обучения школьников работе в графическом редакторе *Paint* и в векторном графическом редакторе.		1		1	учебны инфор Учебна предме (6 – 11
3.4	***Тема 4. Методика преподавания компьютерных презентаций.*** Методика обучения школьников созданию презентаций с использованием декоративного текста, изображений, фигур, эффектов. Особенности создания и использования презентаций на уроках по другим предметам.		1			учебны инфор Учебна предме (6 – 11
3.5	***Тема 5. Методика преподавания электронных таблиц и систем управления базами данных.*** Методика обучения школьников простейшим приемам работы с электронной таблицей Особенности обучения выполнению расчетов с помощью формул и функций, разработке диаграмм и сортировке. Методика изучения основных видов работы с системами управления базами данных: создание таблицы базы данных, связывание таблиц, создание и заполнение формы, поиск и сортировка данных, создание отчетов.		2		1	учебны инфор Учебна предме (6 – 11
3.6	***Тема 6. Методика преподавания коммуникационных технологий.*** Как ввести и объяснить основные понятия. Методика обучения работе в локальной компьютерной сети. Методика обучения работе с электронной почтой.		1			учебны инфор Учебна предме (6 – 11

3.7	***Тема 7. Методика преподавания сети Интернет*** Как ввести и объяснить основные понятия, их история. Методика обучения работе в сети Интернет. работа с электронными письмами (сообщениями). Методика обучения использованию образовательных и других информационных ресурсов Интернет при изучении школьных предметов.		1			учебны инфор Учебна предме (6 – 11
	И Т О Г О	**34**	**26**		**8**	

СЕМИНАРСКИЕ ЗАНЯТИЯ

Материалы для проведения семинарских занятий соответствуют содержанию учебного материала и учебно-методической карте и размещены в электронной библиотеке и пособиях:

Аленский Н. А.. Визуальное объектно-ориентированное программирование в примерах: пособие для студентов мех.-мат. фак. — Мн.: БГУ, 2009.—112с.

Аленский Н.А. Методы программирования: лекции, примеры, тесты: пособие для студентов мех.-мат.фак. В 2 ч. Ч. 1. — Минск: БГУ, 2012. — 88с.

Аленский Н.А. Методы программирования: лекции, примеры, тесты: пособие для студентов мех.-мат.фак. В 2 ч. Ч.2. — Минск: БГУ, 2012. — 76с.

ПРОГРАММА СЕМИНАРСКИХ ЗАНЯТИЙ

Тема 1. Технические и программные средства обучения информатике (1 час).

Тема 2. Урок информатики (1час).

Тема 3. Оценка знаний по информатике (1час).

Тема 4. Содержательная характеристика учебных программ по информатике, учебников и учебных пособий (3часа).

Тема 5. Общие вопросы методики преподавания языка программирования (2 часа).

Тема 6. Методика преподавания начал программирования (4 часа).

Тема 7. Методика преподавания ветвления и циклов (4 часа).

Тема 8. Методика преподавания массивов (2 часа).

Тема 9. Методика преподавания символов и строк (2 часа).

*Тема10.*Методика преподавания основ визуального объектно-ориентированного программирования на кружках, факультативах, в лицеях, специализированных классах и школах (6часов).

Тема 11. Методика преподавания основ веб-конструирования (2час).

Тема 12. Методика преподавания текстовых и графических редакторов (1 час).

Тема 13. Методика преподавания компьютерных презентаций (1 час).

Тема 14. Методика преподавания электронных таблиц и систем управления базами данных (2 час).

*Тема 15.*Методика преподавания коммуникационных технологий (1 час).

Тема 16. Методика преподавания сети Интернет (1 час).

КОНТРОЛЬ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

ТЕСТИРОВАНИЕ В СОП EUNIVERSITY

Типы тестов по общей методике (раздел 1)

1. Термин “Информатика” возник

a) в начале 20-го века;

b) одновременно с появлением больших ЭВМ;

c) в начале 60-х годов 20-го века;

d) одновременно с появлением персональных ЭВМ;

e) одновременно с появлением объектно-ориентированного программирования.

2. Когда началось массовое преподавание в школах основ информатики и вычислительной техники в бывшем Советском Союзе?

3. Когда в Республике Беларусь дисциплину “Информатика” перенесли в базовую школу в 8 – 9 классы?

4. С какого года информатика изучается, начиная с 6-го класса? Введите четыре цифры года.

5. В каком году в педагогических институтах начали преподавать наш предмет, методику преподавания информатики? Введите четыре цифры года.

6. Выберите словесно-фронтальные методы обучения информатике из предложенных.

7. Расположите следующие уровни работы с компьютером по возрастанию уровня подготовки, указав без пробелов номера в нужном порядке: 1) программирующий пользователь, 2) пассивный пользователь 3) системный программист 4) программист.

8. Установите соответствие, например:

6) понимание общих принципов работы ЭВМ (а не частностей!);

7) умение записывать алгоритм, включая написание и отладку программы;

8) объективное отношение к результатам компьютерных вычислений, т. е. критичность и самокритичность в оценке своих способностей.

A. Признак логико-алгоритмического мышления.

B. Признак общего образования при изучении информатики.

C. Признак логико-алгоритмического мышления.

Другие тесты получены на основе экзаменационных вопросов (см дальше).

2. Типы тестов по методике преподавания основ алгоритмизации и программирования (раздел 2)

1. Определить результат после выполнения оператора *if* для каждого из четырёх вариантов исходных данных:

1) *int a=..., b=..., r=...;* 2) ... 3) ... 4) ...

Варианты оператора *if:*: 1) *if (a>=1) r=a+b; else if (b>0) r=a-b; else if (b<-5) r=a*b; else r=a;*

*2) if (a>=1) r=a+b; else if (b>0) r=a-b; else if (b<-5) r=a*b;*

```
3) if (a=b)    if (b)  r=a+b;   else   r=a-b;
4) if (a>=1)  {     if (b>0)  r=a+b;    }  else r=a-b;
5)  if (a>=1)   if (b>0)  r=a+b;       else r=a-b;
6)  if (a+b)   if (a-b)    {a*=b;      b*=a++;     }
else { a*=(--b);            b*=(a--);          }
else { a--;     b=a;     ++b;      }
7) if (a+b)   if (a-b)    {a*=b;  b*=a++;   }
 else { a*=(--b);    b*=(a--);   }    else  a--;     b=a;     ++b;
8): if (a=b)   {  if (b)  r=a+b; }  else   r=a-b;
9)if (a==b)    if (b)  r=a+b;        else   r=a-10;
10)if (a==b)    {  if (b)  r=a+b;     }        else   r=a-10;
11)if (a+b)   if (a-b)  {a*=b;  b*=a++; } else { a*=(--b);   b*=(a--);  } else { a--;   b=a; ++b;  }
12)if (a+b)   if (a-b)  {a*=b;  b*=a++;  }    else { a*=(--b);  b*=(a--);  }    else  a--;  b=a; ++b;
```

2. Определить результат (*true или false*) двух тестов: 1) *int x=..., y=...;* 2) *int x=..., y=...;* для каждого из нескольких (например, пяти) логических выражений (см. [Аленский Н. А. Основы программирования на языке С++], стр.21 <=9 баллов и стр. 117 – 119 на 10 баллов).

В вариантах по-разному расставлены скобки и меняются логические операции и операции сравнения. Может использоваться и операция ! (отрицание).

3. Запишите номера логических выражений, для которых получим *true*, если *int x=..., y=...;*

Логические выражения см. [Аленский Н. А. Основы программирования на языке С++], стр.21 <=9 баллов и стр. 117 – 119 на 10 баллов.

4 Запишите в указанном порядке значения *x, y, r,* которые получатся после выполнения следующих операторов: например*, 1) float x=..., y=..., ...;* *r= !(x>1 || y<=0) ? r=++x: r=y--;*

или в цикле: *2) float x,y;* *cin>>x>>y;* *while (x!=1000 && y!= 1000)*

*{ cout<<(y<x*x || y>3? x+y : x*y)<<" "; cin>>x>>y; }*

Записать результат, если последовательно ввести 0 2 2 0 2 2 1 -5 1000 2. Используется тернарная операция и операции ++ и --.

На 10 баллов вложенная тернарная операция и более сложные логические выражения [Аленский Н. А. Основы программирования на языке С++], стр.21 <=9 баллов и стр. 117 – 119 на 10 баллов.

5 Что будет выведено, если последовательно введём … (указана конкретная числовая последовательность)

```
int  m;   do {   cin>>m;   if (m== -1) break;
                 switch (m)
                { case 1:    case 2:    case 3:   cout<<endl<<2<<"  "; break;
                  case 4:    case 5:              cout<<endl<<3<<"  "; break;
                  case 6:          case 7:        case 8:      cout<<endl<<4<<"     "; break;
                 case 9:    case 10:             cout<<endl<<5<<"  "; break;
                  default:                        cout<<endl<<"Error"<<"  ";
                }              } while (1);
```

В других вариантах где-то нет *break.*

Оператор *switch* может быть с ошибкой . Найти ошибки. Например:

В каких строках (//1 -- //6) есть ошибки компиляции?

```
#define ONE 1
  int  m; const n=4; float const k=1;   cin>>m;
switch (m+k)                                          //1
  {  case ONE:                                        //2
    case m:   cout<<endl<<2<<"  ";    break;    //3
    case ONE+2:  cout<<3<<" “;                  //4
    case n:   cout<<4<<"  ";                    //5
    case k+n:  cout<<"Five"<<"  ";  break;  }   //6
```

В каких строках (//1 -- //7) есть ошибки компиляции?

```
char   C, B='B';   C=getch();
       switch (C)                          //1
         {  case 'A':  cout<<"One"<<"  ";          //2
           case 'B':  switch (B)                 //3
```

```
{ case 'A':  cout<<endl<<" BA ";    break;   //4
  case B :  cout<<endl<< " BB ";    break;   //5
 case 'C' : cout<< endl<<"  BC  "; break;    //6
 }
case 'C': cout<<"  C  "; } //7
```

Оператор *switch* может быть в цикле.

6. Определить результат. Если программа будет выполняться, записать, что будет выведено. В противном случае указать, в каких строках (//1 — //5) будет ошибка.

```
const n=10;  int K=0;   int a[n]={1, 2, -3, -4, -5 , 6, 7, 8, 90, -100 };      //1
for (int i=0; i<n; i++)     if   (a[i]<0)      K++;     cout<<K<<"  ";    //2
int i=0;  K=0;                                                                 //3
for ( ; i<n ;  )                                                               //4
   {      if   (a[i]<0)       K++;     cout<<K<<"  ";      i++;  }          //5
```

В вариантах по-разному записан цикл *for* (см. лекцию 4), разная расстановка {}., параметр цикла может меняться внутри цикла, например, так

```
if   (a[--i]<0)       K++;       cout<<K<<"  ";
```

Вместо *for* может быть оператор *while* или *do… while.*

7.Записать, что будет выведено на экран после выполнения последовательности операторов:

```
1)long int r=…,  a[5]={…},  i=0;
 do {    r*=a[i++];       cout<<r<<"  "<<i<<"  " ;
   } while (i<5);  cout<<endl<<i;
 2)  long int r=…,  a=…,  b=…;    while (r<a)    r*=++b;       cout<<r<<"  "<<b;
```

Внутри цикла может быть *break* или *continue*.

По-разному расставлены скобки {} и используются операции ++(--) справа или слева.

8 Записать номера правильных утверждений:

1) В операторе *while* значения некоторых переменных, записанных в выражении в круглых скобках, должны быть определены до входа в цикл.

2) В операторе *do … while* значения всех переменных, записанных в выражении в круглых скобках после ключевого слова *while*, должны быть определены до входа в цикл.

3) В операторе *while* значения некоторых переменных, записанных в выражении в круглых скобках после ключевого слова *while*, должны меняться в теле цикла так, чтобы на каком-нибудь этапе выражение стало ложным.

4) Тело оператора *while* выполняется всегда, как минимум, один раз.

9. Определить результат: *unsigned short N; N=…(выражение с битовыми операциями) cout<<N<<endl;*

10. С помощью битовых операций обработать к-й справа бит (включить, выключить, заменить на противоположный, вывести на экран и т.п.). Предлагается выбрать правильные ответы из предложенных вариантов.

11. Пусть *float x, y;* Нарисовать область плоскости, в которой следующее логическое выражение истинно. В выражении используется сравнение булевских величин, например a) !(x<0)==(y<0). b) (x<0) < (y<0).

12. Что будет выведено? *int x=4, y=2;*

```
cout<<(!(x&&y))<<(x&y)<<(x|y)<<(x||y) <<(x^y)<<(y<x<1)<<(!(x>y)>(y>0))<<endl;
```

13. Пусть *bool b1, b2, b3, b4; int x; b4=b1 || !b2 && b3;*

Какие из следующих операторов *if* и присваивания дают тот же результат, что и записанный выше оператор присваивания? Для ответов предлагается несколько вариантов.

14 Пусть в ячейке, объявленной *float a;* хранится указанная последовательность 0 и 1. Что это за число в 10-й системе счисления?

15. Как вещественное число (*указано конкретное число*) будет представлено в памяти компьютера в четырёх байтах? Ответ записать в шестнадцатеричном виде.

16. Как вывести k-ю справа шестнадцатеричную цифру целого положительного числа на экран в десятичной системе счисления? Выберите правильные ответы из предложенных вариантов: …

17 a) Как целое число … будет представлено в памяти компьютера в четырёх байтах? b) Как вещественное число … будет представлено в памяти компьютера в четырёх байтах? *Задано одно и то же число.* Ответы записать в шестнадцатеричном виде.

18. Определить результат выполнения программы, если последовательно введём следующие числа: a) 0 0 0 1 2 0 3 2 b) 0 0 0 1 1 1 0 2?

```
unsigned short num, r=0;
  for (int j=1; j<=8; j++)  {  r<<=2;      cin>>num;      r |= num;     }
printf(" \ n % x   % d",r,  r);
```

19. Пусть *float x, y; (y>x)>(x>1);* Какие из следующих выражений равносильны заданному? 1) y>x && x>1, 2) y<=x || x<=1, 3) y<=x && x>1, и т. д. (5 – 6 вариантов)

20. Что будет выведено? *unsigned short int n; n=~200|(-1); cout<<n<<'\n';*

```
bool b1,b2,b3;      b1=11<0x11;      b2=5;
cout<< (b1||b2)<<"  "<< (b2 & (b1=b2))<< (b2 & (b1==b2))<<endl;
```

21. Записать с помощью логических операций, не используя оператор **if**:

bool b1, b2, b3, r; if (b1) r=true; else if (b2) r=b3; else r=false;

Ответы выбрать из предложенных вариантов.

3. Типы тестов по методике преподавания основных тем (раздел 3).

1. В каком классе вводится и объясняется термин, понятие "… "? Например:

системный блок, системная (материнская) плата, процессор и его разрядность, оперативная память (ОЗУ), постоянное запоминающее устройство (ПЗУ), винчестер, адаптер, шина, сканер, веб-камера;

пиксел, разрешающая способность (разрешение) зкрана;

презентация, компьютерная презентация, слайд;

интерфейс, операционная система, рабочий стол, панель задач, ярлык, панель управления, панель инструментов, файл, расширение файлов, путь к файлу, папка, файловый менеджер, проводник;

идентификатор, переменная, константа, массив, одномерный массив, двумерный массив, матрица, индекс одномерного (двумерного) массива, размер одномерного(двумерного) массива;

указатель, адрес и операции над ним.

2. Что означает термин "…" (см. 1)?

3. В каком классе по программе должны начинать изучать "…"? Например:

вставку картинок из коллекции клипов в презентацию, вставку рисунков из файла в презентацию, использование декоративного (фигурного) текста в презентации, использование фигур (линии, звёзды, ленты, выноски и т.п.) в презентации, использование элементов мультимедиа в презентации, использование эффектов в презентации;

создание папок, поиск папок и файлов, копирование папок и файлов, перемещение папок и файлов, удаление папок и файлов;

тип *Real, Integer,* перечислимый тип, записной тип, символьный тип, строки и работа с ними; задание одномерного массива в виде констант при объявлении;

операторы присваивания, *read, readln, Write, Writeln* безформатного вывода, *Write, Writeln* с форматным выводом, *if*(полная и сокращённая форма), *case, for, while, repeat,* вложенные циклы*;*

операции логические, арифметические обычные +,-,*, /, ^, целочисленного деления *div, mod*, операции сравнения, битовые операции;

стандартные ***функции*** (*abs, sqr, sqrt, sin, cos* и др.), ***процедуры*** и функции пользователя, классы, объекты и другие элементы объектно-ориентированного программирования;

ввод одномерного (двумерного) массива с экрана; вывод одномерного (двумерного) массива на экран;

алгоритмы сортировки одномерного массива; нахождения суммы чисел одномерного (двумерного массива (таблицы, матрицы)), удовлетворяющих некоторому простейшему условию (например, положительных); нахождения суммы всех чисел одномерного (двумерного) массива; нахождение в одномерном (двумерном) массиве количества элементов с заданным условием; нахождения наибольшего (наименьшего) числа одномерног (двумерного) массива; преобразования одномерного (двумерного) массива (например, положительные числа увеличить в 2 раза, а отрицательные уменьшить в 10 раз); построения вертикальной диаграммы по заданному одномерному массиву; поиск в одномерном (двумерном) массиве (есть ли число с заданным условием); задание одномерного (двумерного) массива случайным образом.

Могут быть и другие алгоритмы, понятия, вопросы.

4. В каком классе изучается конкретная тема и (или) каков её примерный объём в часах? Смотри программу по информатике для учреждений среднего образования для школьников, которая есть в папке “Программы”.

Тесты могут быть открытыми или с вариантами ответов.

При ответах на тесты необходимо иметь в виду, что некоторые вопросы, термины, понятия, элементы языка *PascalABC* и других систем в учреждениях среднего образования согласно действующей сквозной шестилетней программе могут не изучаться в основном курсе или не существуют вообще.

ВОПРОСЫ К ЭКЗАМЕНУ

Раздел1. Теоретические основы методики преподавания информатики.

1. Что такое информатика?
2. История преподавания информатики в СССР и РБ.
3. Цели и задачи учебной дисциплины «Методика преподавания информатики».
4. Каковы особенности информатики и методики её преподавания по сравнению с другими дисциплинами? Связь информатики и курса её преподавания с другими дисциплинами.
5. Назовите уровни работы с компьютером и дайте им краткую характеристику.
6. Цели преподавания информатики в учреждениях среднего образования.
7. Назовите характерные особенности компьютерной грамотности, образованности, информационной культуры.
8. Что такое принцип научности в преподавании информатики? Объясните фундаментальные понятия информатики: “информация”, “алгоритм”, “исполнитель”.
9. Какие особенности сознательности усвоения и деятельности в информатике?
10. Как достигается доступность и наглядность в информатике?
11. Особенности активности и самостоятельности при изучении информатики.
12. Как реализуется принцип коллективности и индивидуализации?
13. Связь теории и практики, проблема эффективности учебной деятельности.
14. Обзор методов обучения информатике, их классификация.
15. Характерные особенности словесно-фронтальных методов в информатике.
16. Как реализуются анализ, синтез, сравнение и классификация? Приведите примеры обобщения, индукции и дедукции. Особенности аналогии и переноса, абстракции и конкретности в информатике.
17. Как влияет использование компьютера на организационные формы обучения?
18. Какие формы обучения и их особенности при изучении информатики?
19. Какие особенности основных организационных форм обучения в информатике?
20. Какие особенности вспомогательных организационных форм обучения?

21. Требования к компьютерным классам. Специфика работы учителя в условиях локальной компьютерной сети, наличия электронной почты и Интернета.
22. Преимущества и недостатки программированного обучения.
23. Подготовка учителя к уроку, варианты занятий без компьютеров и с их использованием. Методические аспекты повышения эффективности урока.
24. Дифференцированный и индивидуальный подходы при обучении информатике. Принципы подбора индивидуальных и групповых заданий.
25. Управление и контроль за работой обучаемых.
26. Методы контроля знаний, их особенности.
27. Электронное и бумажное (безмашинное) тестирование, преимущества и недостатки тестирования.
28. Особенности построения сквозной программы по информатике для VI — XI классов.
29. — 35. Какие основные разделы информатики и в каком объёме должны изучаться по действующей в настоящее время программе в VI классе? в VII классе? в VIII классе? в IX классе? в X классе? в XI классе? Любой раздел опишите подробно.

36 — 41. Что и в каком объёме должно изучаться по действующей в настоящее время программе в разделе ОАиП в VI классе? в VII классе? в VIII классе? в IX классе? в X классе? в XI классе? Приведите примеры задач и упражнений.

Раздел 2. Методика преподавания основ алгоритмизации и программирования

42. Какие методические особенности преподавания раздела “Основы алгоритмизации и программирования”?
43. Какие методические принципы используются при изучении ОА и П?:
44. Методика преподавания алгоритмов.
45. Какие формы занятий и в каких пропорциях целесообразно применять при изучении ОА и П?
46. Предложите несколько форм занятий при изучении ОАиП с использованием компьютеров.
47. Каким требованиям, на ваш взгляд, должны удовлетворять задачи при изучении ОАиП?
48. Ваше отношение к вспомогательным упражнениям в программировании.
49. Значение блок-схем алгоритмов и программ, их виды и типы упражнений. Когда их использовать при изучении ОАиП? Ваше отношение к блок-схемам.
50. Приведите примеры комплекса взаимосвязанных задач. Их значение, преимущества и недостатки. Когда целесообразно использовать этот методический приём?
51. Какие приведенные в электронных лекциях методические приемы, используемые при решении задач в разделе ОАиП, вам понравились, какие нет и почему? Предложите другие методические приёмы.
52. Методика преподавания начал программирования
53. Методика преподавания ветвления и циклов
54. Методика преподавания массивов
55. Методика преподавания символов и строк
56. Методика преподавания методов программирования на кружках, факультативах, в лицеях, специализированных классах и школах
57. Методика преподавания основ веб-конструирования.

Раздел 3. Методика преподавания основных тем

58. Методика преподавания вопросов, связанных с информацией.
59. Методика преподавания аппаратного и системного программного обеспечения
60. Методика преподавания текстовых и графических редакторов.
61. Методика преподавания компьютерных презентаций.
62. Методика преподавания электронных таблиц и систем управления базами данных.
63. Методика преподавания коммуникационных технологий.
64. Методика преподавания сети Интернет.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

Основная

1. Бочкин А.И. Методика преподавания информатики. – Мн.: Выш. шк., 1998.
2. Лапчик М.П.. Методика преподавания информатики. Учеб. пос. М. — 2001.
3. Семакин И.Г. Преподавание базового курса информатики в средней школе.- М. ЛБЗ, 2004.
4. Софронова Н.В. Теория и методика обучения информатике. М.: Высш. школа, 2004.
5. Угринович Н.Д. Преподавание курса «Информатика и ИКТ» в основной и старшей школе. - М. Бином. 2007.
6. Трайнев В.А., Трайнев И.В. Информационные коммуникационные педагогические технологии.– М, 2005.
7. Аленский Н.А. Методические рекомендации по спецкурсу «Информатика в средней школе». – Мн.: БГУ, 1992.

Учебные пособия для учреждений общего среднего образования.

1. Пупцев А.Е., Макарова Н.П., Лапо А.И. Информатика 6. — Мн.: Нар. асвета, 2008.
2. Заборовский Г.А., Козинский А.А., Пупцев А.Е., Якунина Н.В. Информатика 7. — Мн.: Нар. асвета, 2009.
3. Михайлова Е.Л., Вербовиков Д.А., Коледа Н.Р., Якунина Н.В. Информатика 8. — Мн.: Нар. асвета, 2010.
4. Заборовский Г.А., Лапо А.И., Пупцев А.Е. Информатика 9. — Мн.: Нар. асвета, 2009.
5. Гращенко П.Л., Лапо А.И., Пупцев А.Е. Информатика 10 — Мн.: Нар. асвета, 2006.
6. Гращенко П.Л., Лапо А.И., Огейко А.Г. Информатика 10 — Мн.: Нар. асвета, 2007.
7. Заборовский Г.А., Пупцев А.Е. Информатика 11. — Мн.: Нар. асвета, 2010.

Дополнительная

Дополнительные учебные пособия для учреждений общего среднего образования.

1. Котов В.М. и др. Методы алгоритмизации. Учебное пособие для 8 – 9. — Мн.: Нар. асвета, 2000.
2. Котов В.М., Мельников О.И. Информатика. Методы алгоритмизации. Учебное пособие для 10 – 11 классов. — Мн.: Нар. асвета, 2000.

3. Пупцев А.Е. Информатика, 7 класс. Опорные конспекты. Поурочные тематические задания. — Мн.: Новое знание. 2004.

4. Пупцев А.Е. Информатика, 7 класс. Контрольные и самостоятельные работы — Мн.: Новое знание. 2004.

5. Пупцев А.Е. Информатика, 8 класс. Опорные конспекты. Поурочные тематические задания. — Мн.: Новое знание. 2005.

6. Пупцев А.Е. Информатика, 8 класс. Контрольные и самостоятельные работы. — Мн.: Новое знание. 2005.

Литература по языку C++ и системе C++Builder (для организации самостоятельной работы студентов).

1. Романчик В.С., Люлькин А.Е. C++. Лабораторные работы по курсу «Методы программирования»: учеб.-метод. пособие. — Мн.: БГУ, 2006.

2. Романчик В.С., Люлькин А.Е. Программирование в C++ Builder: учеб. пособие. — Мн.: БГУ, 2008.

3. Аленский Н.А. Методы программирования: лекции, примеры, тесты: пособие для студентов мех.-мат.фак. В 2 ч. Ч. 1. — Минск: БГУ, 2012. — 88с.

4. Аленский Н.А. Методы программирования: лекции, примеры, тесты: пособие для студентов мех.-мат.фак. В 2 ч. Ч.2. — Минск: БГУ, 2012. — 76с.

5. Аленский Н. А.. Основы программирования на языке C++: учеб. пособие. — Мн.: АПО, 2005. — 148с.

6. Аленский Н. А.. Практическое руководство по языку C++: учеб. пособие. — Мн.: АПО, 2007. — 278с.

7. Аленский Н. А.. Визуальное объектно-ориентированное программирование в примерах: пособие для студентов мех.-мат. фак. — Мн.: БГУ, 2009.—112с.

Сборники задач по программированию:

1. Аленский Н. А. Сборник задач по программированию на языке C++. Учеб-метод. пособие. – Мн.: БГУ, 2005. – 48с.

2. Зборнік задач па курсу метады праграміравання і інфарматыка. У 2 ч. Ч. I. Асноўныя прыемы праграміравання. Вучэбна-метадычны дапаможнік. / Г. А. Расолька, Е. В. Крэмень, Ю. А. Крэмень. – Мінск: БДУ, 2013.

Примечание. Более полный список дополнительной литературы по программированию можно найти в приведенных выше методических разработках и учебных пособиях преподавателей кафедры Веб-технологий и компьютерного моделирования ММФ БГУ.

ТИПОВАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Типовая программа курса хранится в электронной библиотеке.