Методические особенности изучения раздела «Основы алгоритмизации»

Содержание

Введение

В нашем мире современному человеку все чаще приходится сталкиваться с большими объемами информации. И от того насколько эффективно он с ней работает, будет зависеть его жизненный и профессиональный успех в условиях информатизации общества. А осознание и эффективное использование информации невозможно без навыков ее систематизации.

Информатика - в настоящее время одна из фундаментальных отраслей научного знания, формирующая системно-информационный подход к анализу окружающего мира, изучающая информационные процессы, методы и средства получения, преобразования, передачи, хранения и использования информации, стремительно развивающаяся и постоянно расширяющаяся область практической деятельности человека, связанная с использованием информационных технологий.

В связи с этим основной линией обучения в базовом курсе информатики является линия "Основы алгоритмизации".

В любой среде программирования реализуются основные алгоритмические конструкции, развивающие алгоритмический стиль мышления, важность которого отмечена Н.М. Амосовым, А.Н. Лонда, Н.Н. Моисеевым и другими учеными.

Алгоритмы используются в ходе описания какого-либо процесса (физического, химического, биологического, математического), в управлении, воспитании, во всей социальной сфере жизни человека. Именно это и доказывает необходимость их введения в обучение. Таким образом, алгоритм - это не программа-шаблон, а механизм, согласно которого функционирует, развивается любая самоорганизующаяся система. Некоторые алгоритмы человек осваивает самостоятельно, другие требуют обучения.

Актуальность: В течение всего периода преподавания информатики в школе (с 1985 года) актуальность темы «Основы алгоритмизации» претерпела

значительные изменения. В силу некоторых обстоятельств: наличия теоретической базы предмета и технического обеспечение кабинета информатики, значимость преподавания темы в период с 2005 года по 2010 год значительно снизилась. А в 2017 году 5-6 классы были отведены на внеурочную деятельность. Поэтому уменьшилось количество уроков, отводимых на изучение этой темы в 5-11 классах. Большая часть времени отводится на преподавание тем цикла «Информационные и коммуникационные технологии».

Наряду с этим нисколько не изменились требования к уровню усвоения знаний и умений этого раздела программы по информатике, так как он остается основой фундаментальных знаний по предмету. Актуальность исследования обуславливается необходимостью пропедевтического обучения основам алгоритмизация и программирования, которое нивелирует трудности при изучении содержательной линии «Основы алгоритмизации» в базовом школьном курсе информатики, возникающие из-за несоответствия между достаточно большим объемом содержания и относительно небольшим количеством часов, выделенным на изучение данной содержательной линии.

1. Место содержательной линии «Основы алгоритмизации» в ФГОС

Введение нового Федерального государственного образовательного стандарта общего образования привело к пересмотру содержания обучения во всех дисциплинах, в том числе и в рамках дисциплины «Информатика и ИКТ» в 5-9 классах.

В федеральном образовательном стандарте по информатике и ИКТ тема алгоритмизации присутствует в разделе «Обработка информации»: «Алгоритм, свойства алгоритмов. Способы записи алгоритмов; блок-схемы. Алгоритмические конструкции. Логические значения, операции, выражения. Разбиение задачи на подзадачи, вспомогательный алгоритм».

В примерной программе теме «Основы алгоритмизации» выделяется 19 ч. Содержание темы расписано более подробно, чем в стандарте: «Алгоритм. Свойства алгоритма. Способы записи алгоритмов; блок-схемы. Возможность автоматизации деятельности человека. Исполнители алгоритмов (назначение, среда, режим работы, система команд). Компьютер как формальный исполнитель алгоритмов (программ). Алгоритмические конструкции: следование, ветвление, повторение. Разбиение задачи на подзадачи, вспомогательный алгоритм. Алгоритмы работы с величинами: типы данных, ввод и вывод данных».

Приоритетными объектами изучения в курсе информатики основной школы остаются информационные процессы и информационные технологии. В теоретической части курса раскрывается содержание информационной технологии решения задачи. Рассматриваются обобщающие понятия – информационный процесс, информационная модель и информационные основы управления. Практическая часть курса направлена на освоение учащимися навыков использования средств информационных технологий, которые являются значимыми не только для формирования компьютерной грамотности, социализации школьников и последующей деятельности выпускников, но и для повышения эффективности освоения других учебных предметов. Изучение

информатики и информационных технологий в основной школе направлено на достижение следующих целей:

- освоение знаний, составляющих основу научных представлений об информации, информационных процессах, системах, технологиях и моделях;
- овладение умениями работать с различными видами информации с помощью компьютера и других средств информационных и коммуникационных технологий, организовывать собственную информационную деятельность и планировать ее результаты;
- развитие познавательных интересов, интеллектуальных и творческих способностей средствами ИКТ;
- воспитание ответственного отношения к информации с учетом правовых и этических аспектов ее распространения; избирательного отношения к полученной информации;
- выработка навыков применения средств ИКТ в повседневной жизни, при выполнении индивидуальных и коллективных проектов, в учебной деятельности, дальнейшем освоении профессий, востребованных на рынке труда.

Федеральный базисный учебный план для образовательных учреждений Российской Федерации отводит 105 часов для обязательного изучения информатики и информационно-коммуникационных технологий на ступени основного общего образования. В том числе в VIII классе – 35 учебных часов из расчета одного учебного часа в неделю и IX классе – 70 учебных часов из расчета двух учебных часов в неделю. В примерной программе предусмотрен резерв свободного учебного времени в объеме одиннадцати часов для реализации авторских подходов; использования разнообразных форм организации учебного процесса; внедрения современных методов обучения и педагогических технологий; учета региональных условий. В примерной программе по дисциплине «Информатика и ИКТ» на раздел «Алгоритмы и

исполнители» отводится 19 часов – это 20% от общего времени, выделяемого на дисциплину. Соотношение удельных весов различных разделов содержания по дисциплине «Информатика и ИКТ» представлено в Таблице1.

Таблица1

	1. Информационные процессы						2. Информационные технологии					
Общее число часов: 105 Резерв времени: 11 часов (10,5%) Число часов на раздел 1: 46 Число часов на раздел 2: 48	Информация и информационные процессы	Представление информации	Компьютер как универсальное устройство обработки информации	Алгоритмы и исполнители	Формализация и моделирование	Информационные процессы и технологии в обществе	Обработка текста	Обработка графики	Мультимедийные технологии	Обработка числовой информации	Хранение информации	Коммуникационные технологии
Учебные часы из примерной программы	4	6	4	20	8	4	14	4	8	6	4	12
Процент от общего числа часов	4,2	6	4,2	21	9	4,2	15	4,2	9	6	4,2	13

В ФГОС и обязательном минимуме по информатике содержание алгоритмической линии определяется через следующий перечень понятий: центральное теоретическое понятие – алгоритм, вводится как содержательное понятие, свойства алгоритмов, исполнители алгоритмов, система команд исполнителя; формальное исполнение алгоритмов; основные алгоритмические конструкции; вспомогательные алгоритмы.

Понятия управления и обратной связи вводятся в контексте работы с компьютером. Также данные понятия переносятся и в более широкий контекст социальных, технологических и биологических систем. Он поддержан построением программ управления движущимися объектами в виртуальных и реальных средах. Резерв времени отводится на разработку алгоритмов, решающих поставленные задачи с использованием математических функций для записи арифметических выражения, операторов ветвления и цикла. Данный раздел реализуется в изучении таких образовательных областей, как информатика и информационные технологии, математика и естествознание. В соответствии с ФГОС изучение информатики в основной школе должно обеспечить:

- формирование информационной и алгоритмической культуры; формирование представления о компьютере как универсальном устройстве обработки информации; развитие основных навыков и умений использования компьютерных устройств;
- формирование представления об основных изучаемых понятиях: информация, алгоритм, модель – и их свойствах;
- развитие алгоритмического мышления, необходимого для профессиональной деятельности в современном обществе; развитие умений составить и записать алгоритм для конкретного исполнителя; формирование знаний об алгоритмических конструкциях, логических значениях и операциях; знакомство с одним из языков программирования и основными алгоритмическими структурами – линейной, условной и циклической;
- формирование умений формализации и структурирования информации, умения выбирать способ представления данных в соответствии с поставленной задачей – таблицы, схемы, графики, диаграммы, с использованием соответствующих программных средств обработки данных;
- формирование навыков и умений безопасного и целесообразного поведения при работе с компьютерными программами и в Интернете, умения соблюдать нормы информационной этики и права. Исходя из требования ФГОС, необходимо решить одну из важнейших проблем образования – отбор средств обучения. Процесс вхождения школы в современное мировое образовательное пространство требует совершенствования компьютерно-информационной составляющей. Особый интерес представляют вопросы, связанные с компьютеризацией обучения, поскольку традиционные методы без использования технических средств давно исчерпали свои возможности. Актуальной в данном вопросе будет реализация обучения с использованием образовательной среды исполнителей с

обратной связью. Главная цель внедрения реальных исполнителей в образовательную деятельность школ – это создание благоприятных условий для разностороннего развития личности, которое включает в себя интеллектуальное развитие, 10 удовлетворение интересов, выявление склонностей, развитие способностей обучающихся, их самообразование и профессиональное самоопределение

1.2 Межпредметные связи изучения курса информатики и темы «Основы алгоритмизации»

Основная особенность МПИ - связь с другими, прежде всего методического цикла, предметами.

Как отмечает Н.В. Софронова, «преподавание информатики на современном уровне опирается на сведения из различных областей научного знания: биологии (биологические самоуправляемые системы, такие как человек, другой живой организм), истории и обществоведения (общественные социальные системы), русского языка (грамматика, синтаксис, семантика и пр.), логики (мышление, формальные операции, истина, ложь), математики (числа, переменные, функции, множества, знаки, действия), психологии (восприятие, мышление, коммуникации)».

Другой 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.

Еще одна особенность МПИ – связь предмета с использованием компьютера, который обладает несравненно большей “самостоятельностью”, чем любой другой прибор.

1.3 Анализ существующих учебно-методических комплексов по дисциплине «Информатика и ИКТ»

В настоящее время по дисциплине «Информатика и ИКТ» имеются значительные учебно-методические наработки для разных возрастных групп учащихся. Издано множество учебников, учебных пособий и методических рекомендаций для преподавателей. Одной из задач основного курса является обучение основам алгоритмизации и программирования, направленное на развитие алгоритмического, логического мышления учеников, а также на формирование операционного типа мышления. Методика обучения основам алгоритмизации и программирования представлена в рамках курсов известных авторских коллективов. Рассмотрим подробнее курсы этих авторов.

УМК «Информатика и ИКТ», автор Семакин И.Г. 7-9 классы, основная школа

Данный учебно-методический комплекс, обеспечивающий обучение курсу информатики, соответствует ФГОС и включает в себя:

- учебник «Информатика» для 7 класса. Авторы: Семакин И.Г., Залогова Л.А., Русаков С.В., Шестакова Л.В.;
- учебник «Информатика» для 8 класса. Авторы: Семакин И.Г., Залогова Л.А., Русаков С.В., Шестакова Л.В.;
- учебник «Информатика» для 9 класса. Авторы: Семакин И.Г., Залогова Л.А., Русаков С.В., Шестакова Л.В.;
- задачник-практикум под редакцией Семакина И.Г., Хеннера Е.К.;
- методическое пособие для учителя. Авторы: Семакин И.Г., Шеина Т.Ю.;
- комплект цифровых образовательных ресурсов, помещенный в Единую коллекцию ЦОР;
- комплект дидактических материалов для текущего контроля результатов обучения информатике в основной школе, под редакцией Семакина И.Г.

Данный курс опирается на базовые научные представления предметной области: информация, информационные процессы, информационные модели. Большое внимание в курсе уделено формированию алгоритмической культуры

учащихся, развитию алгоритмического мышления, входящим в перечень предметных результатов ФГОС. Этой теме посвящена большая часть содержания и учебного планирования в 9 классе.

Для практической работы используется исполнитель алгоритмов, разработанный авторами и входящий в комплект ЦОР – «Стрелочка». Для изучения основ программирования используется язык Паскаль. На тему «Управление и алгоритмы» в данной авторской программе отводится 12 часов. В курсе предполагается решение большого количества задач, позволяющих усвоить учащимся основы алгоритмизации и программирования на высоком уровне.

Задачник-практикум дает обширный материал для организации практической работы на уроках и домашней работы учащихся. Большое число разнообразных заданий предоставляет учителю возможность варьировать содержание курса по времени и уровню сложности. При изучении данного курса учащиеся смогут: выполнять трассировку заданных простых алгоритмов; строить блок-схемы несложных алгоритмов; использовать школьный алгоритмический язык для описания алгоритмов; работать с готовой программой на одном из языков программирования высокого уровня; составлять несложные программы решения вычислительных задач; осуществлять отладку и тестирование программы.

<u>УМК «Информатика и ИКТ», автор Угринович Н.Д. 8-9 классы, основная школа</u>

Состав УМК:

- учебник «Информатика и ИКТ. Базовый курс», 8 класс;
- учебник «Информатика и ИКТ. Базовый курс», 9 класс;
- практикум по информатике и информационным технологиям, 8-11 классы;
- методическое пособие для учителя «Информатика и ИКТ. Методическое пособие», 8-11 классы;
- комплект плакатов, «Информатика и ИКТ. Основная школа»;

- методические рекомендации по использованию плакатов «Информатика и ИКТ. Основная школа»;
- авторская мастерская Угриновича Н.Д.;
- ЭОР клавиатурный тренажер «Руки солиста». Особое место в данном комплексе по дисциплине «Информатика и ИКТ» для 9 класса занимает тема «Алгоритмизация и основы объектноориентированного программирования» [16]. В этой теме рассматриваются все основные алгоритмические структуры и их кодирование на трех языках программирования:
- алгоритмическом языке OpenOffice Basic, который входит в свободно распространяемое интегрированное офисное приложение OpenOffice Basic в операционных системах Windows и Linux;
- объектно-ориентированном языке Visual Basic 2005, который распространяется по лицензии корпорации Microsoft;
- объектно-ориентированном языке Gambas (аналог – Visual Basic в операционной системе Linux), который распространяется по лицензии компании AltLinux При изучении данного курса учащиеся смогут объяснить структуру основных алгоритмических конструкций и использовать их для построения 13 алгоритмов; определить основные типы данных и операторы; разработать и записать на языке программирования типовые алгоритмы; создавать проекты с использованием визуального объектно-ориентированного программирования. Объектно-ориентированный подход к решению задач позволяет сформировать у учащихся объектный стиль мышления и способствует подготовке учащихся к дальнейшему изучению программирования.

<u>УМК «Информатика и ИКТ», автор Макарова Н.В. 8-9 классы, основная школа</u>

Состав УМК:

- учебник «Информатика и ИКТ». 8-9 классы;
- Информатика и ИКТ. Практикум. 8-9 класс;

- Информатика и ИКТ. Задачник по моделированию. 9-11 класс;
- авторская мастерская Макаровой Н.В. В данном курсе основы программирования рассматриваются в среде Лого. Изучая тему «Программирование» в данной среде, учащиеся знакомятся:
- с программами для реализации типовых конструкций алгоритмов;
- с понятиями процедуры и модуля, процедуры с параметрами;
- с функциями;
- с инструментами логики при разработке программ. Методика разработки простейших программ в среде ЛогоМиры позволяет развить у обучающихся навыки решения задач с применением алгоритмического, системного и объектно-ориентированного подходов к решению задач; формирует алгоритмическое и логическое мышление; способствует развитию интереса школьников к обучению.

УМК «Информатика и ИКТ», автор Гейн А.Г. 7-9 классы, основная школа

Состав УМК:

- рабочие программы;
- учебник;
- рабочая тетрадь;
- задачник-практикум;
- тематические тесты;
- книга для учителя.

В основе курса лежит установка на формирование у учащихся системы базовых понятий информатики и представлений об информационных технологиях, а также выработка умений применять их для решения жизненных задач. Изучение основ алгоритмизации осуществляется с использованием учебного исполнителя Паркетчик, который является свободно распространяемым продуктом. Использование Паркетчика позволяет реализовать принцип наглядности, доступности, переход от простого к

сложному. Русский интерфейс и автоматизированный ввод команд позволяет сосредоточить внимание обучающихся на изучении алгоритмизации, не теряя времени на освоение сложного синтаксиса языка программирования.

- Изучение данного курса дает возможность учащимся:
- составлять и записывать алгоритмы для учебных исполнителей с использованием соответствующих алгоритмических конструкций;
- составлять протоколы исполнения алгоритмов;
- распознавать необходимость применения той или иной алгоритмической конструкции при решении задачи;
- использовать готовые вспомогательные алгоритмы при создании нового алгоритма.
- Результат анализа учебно-методических комплексов по дисциплине «Информатика и ИКТ» представлен в Таблице 2.

Таблица 2

Авторы УМК	Структура УМК	Цель обучения	Среда для изучения алгоритмизации	Особенности УМК
Макарова Н.В.	Учебник 8-9 класс; практикум 8-9 класс; пособия для учителя	Развитие системного мышления и формирование элементов информационной культуры	ЛОГО	Моделирование различных объектов средствами ИТ
Семакин И.Г.	Учебники 8-9 класс; задачник 7-11 класс; пособие для учителя	Обеспечить освоение стандарта ГРИС	PASCAL	После каждой главы система основных понятий; два уровня изложения материала

Угринович Н.Д.	Учебники 8-9 класс; пособие для учителя с диском	Обеспечить освоение стандарта	VBAISIC (QBASIC)	BASIC; диск с тестовыми и другими материалами
Гейн А.Г.	Программа 7-9 класс; учебники 7,8,9 классы; рабочие тетради; Метод. реком. для учителя; задачник-практикум	Применять ИТ для решения жизненных задач	Исполнитель ПАРКЕТЧИК	Направлен на формирование УУД

На основе проанализированных учебных комплексов можно сделать вывод, что основным критерием выбора курса для изучения алгоритмизации и программирования является не только содержание и методические приемы изучения, а также наглядность и простота изучения. Правильно организованное обучение должно способствовать развитию алгоритмического и логического мышления в естественной для этого обстановке. Такая обстановка должна предоставлять опыт работы с различными моделями; знакомить с общими принципами и методами программирования; позволять учащимся адаптировать приобретенные навыки для решения жизненных задач.

Для развития алгоритмического мышления обучающегося, а также для реализации его творческих способностей, необходимо создать ему соответствующие условия и предоставить возможность участвовать в групповой деятельности. Рассмотренные курсы ограничиваются виртуальными средами для исполнителя алгоритмов, что не позволяет в полной степени реализовать принцип наглядности. Также не все УМК предполагают решение большого количества задач, что может привести к усвоению основ алгоритмизации и программирования на недостаточно высоком уровне. Начинать изучение темы «Алгоритмизация и программирование» целесообразно со знакомства с исполнителями с обратной связью. Такое

обучение обеспечит подготовку к последующему изучению объектно-ориентированных языков и языков программирования высокого уровня в наглядной форме.

Рассмотрение базовых алгоритмических конструкций, в применении с программируемым устройством, позволит сформировать навыки их использования при решении более сложных практических задач. В сравнении с исполнителями виртуальными, реальные исполнители послужат фундаментов для формирования умения проанализировать задачу и формализовать ее условие применительно к условиям реальной обстановки.

2 . Методические особенности изучения раздела «Основы алгоритмизации»

2.1. Анализ примерной программы для основной средней школы.

В примерной программе по информатике предлагается изучение темы «Основы алгоритмизации». Здесь ученикам даются такие понятия как: алгоритм, исполнители (состояния, возможные обстановки и система команд исполнителя), команды-приказы и команды-запросы, отказ исполнителя. Также объясняется необходимость формального описания исполнителя. Понятие алгоритм объясняется как план управления исполнителем (исполнителями). Изучаются понятия алгоритмический язык и программа (запись алгоритма на алгоритмическом языке). Компьютер — автоматическое устройство, способное управлять по заранее составленной программе исполнителями, выполняющими команды. Исходя из примерной программы, изучаются такие понятия как «управление», «сигнал», «обратная связь». И предлагается привести примеры: компьютер и управляемый им исполнитель; компьютер, получающий сигналы от цифровых датчиков входе наблюдений и экспериментов, и управляющий реальными (в том числе движущимися) устройствами.

Базовые учебные элементы:

1.Линейные программы. Их ограниченность: невозможность предусмотреть зависимость последовательности выполняемых действий от исходных данных.

2.Конструкции ветвления (условный оператор) и повторения (операторы цикла в форме «пока» и «для каждого»).

3.Имя алгоритма и тело алгоритма. Использование в теле алгоритма имен других алгоритмов. Вспомогательные алгоритмы.

4.Величина (переменная): имя и значение. Типы величин: целые, вещественные, символьные, строковые, логические. Знакомство с табличными величинами (массивами). Представление о структурах данных.

5.Примеры задач управления исполнителями, в том числе -обработки числовых и строковых данных; реализация алгоритмов решения в выбранной среде программирования. Сортировка и поиск: постановка задач.

6.Примеры коротких программ, выполняющих много шагов по обработке небольшого объёма данных; примеры коротких программ, выполняющих обработку большого объёма данных.

7.Сложность вычисления: количество выполненных операций, размер используемой памяти; их зависимость от размера исходных данных.

8.Понятие об этапах разработки (алгоритма) программ и приемах отладки программ. После изучения данного раздела ученики **должны научиться:**

1.Использовать термины «исполнитель», «алгоритм», «программа», а также понимать разницу между употреблением этих терминов в обыденной речи и в информатике;

2.Составлять неветвящиеся (линейные) алгоритмы управления исполнителями;

3.Использовать логические значения, операции и выражения с ними; Выполнять без использования компьютера («вручную») алгоритмы анализа числовых данных и управления исполнителями, описанные на алгоритмическом языке с использованием конструкций ветвления и повторения, вспомогательных алгоритмов, простых и табличных величин;

4.Для более узкого класса задач - создавать и выполнять на компьютере программы для их решения. Выпускник получит возможность:

1.познакомиться с использованием в программах строковых величин и с операциями со строковыми величинами;

2.создавать программы для решения задач, возникающих в процессе учебы и вне её;

3.познакомиться с задачами обработки данных и алгоритмами их решения;

4.познакомиться с понятием «управление», с примерами того, как компьютер управляет различными системами (летательные и космические аппараты, станки, оросительные системы, движущиеся модели и др.)

В результате освоения курса информатики в основной школе учащиеся получат представление о методах представления и алгоритмах обработки данных, дискретизации, о программной реализации алгоритмов.

У выпускников будут сформированы: основы алгоритмической культуры; умение составлять несложные программы; обучающиеся познакомятся с одним из языков программирования и основными алгоритмическими структурами - линейной, условной и циклической; получат опыт написания и отладки программ в выбранной среде программирования.

2.2. Методика преподавания темы «Алгоритмы» в курсе информатики

2.2.1 Методика введения понятия алгоритм

При изучении основ алгоритмизации важно соблюдать методическую последовательность введения теоретических понятий и создать плавный переход между ними. Учащиеся должны легко ориентироваться в предыдущей теме прежде чем переходить к новой. Таким образом, рекомендуется следующая последовательность изучения материала:

1 Понятие алгоритм

2 Свойства алгоритма

3 Способы представления алгоритма

4 Структуры алгоритмов

5 Методика разработки алгоритмов

6 Структурное программирование

Изучение алгоритмизации делится на два этапа, это и есть само изучение алгоритмизации, а затем программирования. Во многих учебных программах останавливаются только на изучении алгоритмизации, так как небольшое количество учителей информатики имеют должный уровень подготовки для преподавания программирования на каком-либо конкретном языке программирования.

Изучение алгоритмизации помогает развить у учащихся алгоритмическое мышление, что само по себе является базой для освоения программирования. Поэтому изучение алгоритмизации является важной частью курса информатики и при преподавании этой части курса учитель должен быть особенно внимателен и осторожен.

В образовательном стандарте базового курса по информатики и ИКТ основное содержание по линии алгоритмизации определяется через следующие понятия:

- алгоритм, свойства алгоритма, способы записи алгоритмов;

- исполнители алгоритмов (назначение, среда, режим работы, система команд);

- компьютер как формальный исполнитель алгоритмов;

- основные алгоритмические конструкции (следование, ветвление, повторение);

- разбиение задачи на подзадачи, вспомогательный алгоритм;

- алгоритмы работы с величинами (тип данных, ввод и вывод данных).

Изучение алгоритмизации начинается с введения понятия алгоритма. Понятие алгоритма относится к исходным математическим понятиям, поэтому не может быть определено через другие, более простые понятия. Из-за этого определение алгоритма в школьных учебниках по информатике отличается большим разнообразием. Вот некоторые из них:

В учебнике И.Г. Семакина и др. алгоритм определяется как *последовательность команд, управляющих работой какого-либо объекта*, и далее дается более строгое определение – *понятное и точное предписание исполнителю выполнить конечную последовательность команд, приводящую от исходных данных к искомому результату.*

В учебнике А.Г. Кушниренко алгоритм определяется как *программа, записанная на специальном школьном алгоритмическом языке.*

В учебнике Н.Д. Угриновича алгоритм вводится как *чёткое описание последовательности действий.*

В обычной жизни дети не встречаются с этими понятиями, но они находят применение алгоритмов в различной деятельности человека, о чем важно сообщить детям на первом же уроке и подтвердить это примерами. Вводя понятие алгоритма, учителю следует акцентировать внимание учащихся

на том, что алгоритм всегда составляется с ориентацией на *исполнителя алгоритма.*

Так как одной из особенностей курса «Алгоритмизация и программирование» является его практическая направленность, то понятие исполнителя алгоритма следует, вводит на основе практических примеров из жизни учащихся. Основным исполнителем на начальном моменте изучения темы должен быть человек. Ученики сами должны выступить в роли исполнителей не сложных алгоритмов, например рисование окружности при помощи циркуля. В зависимости от класса, в котором изучается данная тема, задачи для исполнителя могут быть и сложнее: найти корень квадратного уравнения, построить вписанную в треугольник окружность, и т. д.

Основной характеристикой исполнителя, с точки зрения управления, является *система команд исполнителя* (СКИ). Это конечное множество команд, которые понимает исполнитель, т.е. умеет их выполнять. Для знакомства с СКИ можно дать ученикам такой алгоритм, который они заведомо не смогут выполнить. После этого должно следовать закрепление данного понятия на основе задач определения СКИ у различных исполнителей.

СКИ определяет первое свойство алгоритма – понятность, то есть алгоритм может включать в себя *только команды, входящие в СКИ.* Он не должен быть рассчитаны на принятие исполнителем самостоятельных решений, не предусмотренных составителем алгоритма.

После свойства понятности следует свойство *точность. Опять* же можно привести несколько примеров алгоритмов, которые выполняются не точно. Вот один из них: кулинарный рецепт можно рассматривать как алгоритм для исполнителя-повара по приготовлению блюда. Но если одним из пунктов в нем будет написано: «Положить несколько ложек сахара», то это пример неточной команды. Сколько ложек? Каких ложек? Каждый повар может это понимать по-своему, и результаты будут разными. Пример точной команды: «Положить 2 столовые ложки сахара».

Еще одно свойство, которое отражено в определении алгоритма – *конечность.* Оно формулируется так: исполнение алгоритма и, следовательно, получение искомого результата должно завершиться за конечное число шагов. Здесь под шагом подразумевается выполнение отдельной команды. В данном случае это свойство отражает ситуации, когда алгоритм «зацикливается» и не дает результата. Такой алгоритм бесполезен, учащиеся должны научится отличать эти алгоритмы.

Еще одно свойство алгоритма дискретность. «Дискретность состоит в том, что команды алгоритма выполняются последовательно, с точной фиксацией моментов окончания выполнения одной команды и начала выполнения следующей». Требование последовательного выполнения команд заложено в определении алгоритма, но, на мой взгляд, на данном свойстве нужно заострить внимание. Не каждый ребенок сможет выделить его из определения алгоритма.

«Свойство массовости выражается в том, что алгоритм единым образом применяется к любой конкретной формулировке задачи, для решения которой он разработан». От свойства массовость легко перейти к такому понятию как исходные данные. По сути, это свойство можно назвать универсальностью алгоритма по отношению к исходным данным решаемой задачи. Данное свойство не является необходимым свойством алгоритма, а скорее определяет качество алгоритма: универсальный алгоритм лучше неуниверсального (алгоритм решения частной задачи – тоже алгоритм!). Следует указать учащимся на то, что исполнителю всегда необходимо иметь *исходные данные* с которыми он будет работать (деньги, продукты, детали, таблицы чисел и т.п.). Например, исполнителю, решающему математическую задачу нужна исходная числовая информация, которая обычно задаётся в условии. Если вам нужно найти номер телефона нужного человека, то исходными данными будут фамилия человека, его инициалы, телефонная книга, а иногда ещё и домашний адрес, так как Ивановых или Петровых с одинаковыми инициалами может оказаться в телефонной книге несколько.

Если все данные свойства выполняются, то исполнитель выполняет алгоритм формально. Это означает, что при выполнении алгоритма исполнитель строго следует командам и не какого творчества с его стороны быть не может. Отсюда следует вывод о возможности создания автоматических исполнителей. Таким автоматическим исполнителем по обработке информации является компьютер. Дети сами могут назвать таких автоматических исполнителей: роботы, станки с автоматическим управлением, автоматическая стиральная машина и так далее.

После того как все свойства алгоритма разобраны следует их закрепить при помощи задач. Для этого полезно рассмотреть с учениками несколько заданий следующего содержания:

1) выполнить роль исполнителя: дан алгоритм, формально исполнить его;

2) определить исполнителя и систему команд для данного вида работы;

3) в рамках данной системы команд построить алгоритм;

4) определить необходимый набор исходных данных для решения задачи.

В качестве примера задачи первого типа можно использовать *алгоритм игры Баше*, рассматриваемый в учебниках. Правила игры определены так: в игре используются 7, 11, 15, 19 предметов. За один ход можно брать 1, 2 или 3 предмета. Проигрывает тот игрок, который берет последний предмет. Предлагается алгоритм выигрыша для первого игрока. После того как ученики поиграли в эту игру по тем правилам, что описаны в учебнике, можно предложить им несколько заданий аналитического характера на тему игры Баше. Задания могут быть предложены в качестве домашней работы.

Теперь рассмотрим пример задания второго типа.

Задача: Описать систему команд исполнителя «Геометр», который мог бы выполнять геометрические построения с помощью циркуля и линейки.

Решение. Ученикам знаком класс задач, которые в геометрии называются задачами на построение с помощью линейки, циркуля и карандаша. *Полной системой команд* для исполнителя «Геометр» является следующий список:

1. Провести отрезок прямой между двумя данными точками.

2. Установить раствор циркуля, равный длине данного отрезка.

3. Установить ножку циркуля в данную точку.

4. Провести окружность.

5. Выделить общие точки двух линий (пересечения или касания).

Необходимо обратить внимание учеников на элементарность каждой команды. Делить их на более простые шаги не имеет смысла.

При построении СКИ ученики должны решать две проблемы: проблема элементарности команд и проблема полноты системы команд. *Система команд исполнителя называется полной, если она содержит весь минимально-необходимый набор команд, позволяющий построить любой алгоритм в том классе задач, на который ориентирован исполнитель.*

Отрешения предыдущей задачи можно перейти к задачам третьего типа. Оставив исполнитель и СКИ прежними ученикам можно дать такую задачу: «Записать для исполнителя Геометр алгоритм построения окружности, для которой задан её диаметр отрезком АВ».

Данный переход способствует лучшему восприятию задачи, так как ученики уже знакомы с исполнителем и его СКИ.

Решение: 1.установить ножку циркуля в т. А;

2.установить раствор циркуля, равный АВ;

3.провести окружность установить ножку циркуля в т. В;

4.провести окружность;

5.выделить точки пересечения окружностей: т. Сит. D;

6.провести отрезок CD;

7.выделить точку пересечения АВ и CD: т. О;

8.установить ножку циркуля в т. О;

9.установить раствор циркуля, равный ОВ;

10.провести окружность.

С учениками необходимо проанализировать данную задачу на соответствие свойствам алгоритма. Учеников следует подвести к такому выводу: «данный алгоритм удовлетворяет всем основным свойствам:

понятности, точности, конечности; благодаря чему может исполняться формально».

Задания четвертого типа относятся к проблеме постановки задач на построение алгоритмов. Для выполнения требуемой работы – решения данной задачи – необходим не только алгоритм, но и полный набор исходных данных. Это могут быть какие-то материальные объекты (например, детали для сборки устройства; продукты для приготовления блюда и пр.) или информация (числовые данные для расчетов). Вот пример задачи на определение полного набора данных.

Задача: Определить полный набор данных для вычисления времени падения кирпича с крыши дома.

Ответ: высота дома, ускорение свободного падения.

2.2.2 Обучение методам построения алгоритмов

Главной целью раздела алгоритмизации является овладение учащимися структурной методикой построения алгоритмов. Традиционно применяемым дидактическим средством в этом разделе являются учебные исполнители алгоритмов. Главным достоинством учебных исполнителей является: ясность для ученика решаемых задач, наглядность процесса работы в ходе выполнения программы. Как известно, дидактический принцип наглядности является одним из важнейших в процессе любого обучения.

Для того чтобы ученикам было легко работать с учебными исполнителями, они должны удовлетворяет следующим условиям:

• это должен быть исполнитель, работающий «в обстановке»;

• этот исполнитель должен имитировать процесс управления некоторым реальным объектом (черепахой, роботом и др.);

• в системе команд исполнителя должны быть все структурные команды управления (ветвления, циклы);

• исполнитель позволяет использовать вспомогательные алгоритмы (процедуры).

Изучая работу любого исполнителя алгоритмов, учителю следует привести его характеристики, совокупность которых называется *архитектурой исполнителя.* К ним относятся:

- среда, в которой работает исполнитель;
- режим работы исполнителя;
- система команд исполнителя;
- данные, с которыми работает исполнитель.

Обучение программированию лучше организовать в ходе решения задач, подобранных в специально выстроенной последовательности, которая определяется следующими дидактическими принципами:

От простого к сложному – т.е. постепенное усложнение решаемых задач.

-Новизна – каждая задача должна вносить новый элемент знаний – новую команду, новый приём программирования.

-Наследование – решение каждой следующей задачи требует использования знаний, полученных при решении предыдущих.

Для написания алгоритмов в учебных исполнителей используется алгоритмический язык и блок-схемы. С ними можно познакомить на одном уроке, а затем продолжать изучение алгоритмизации и блок-схем совместно с построением алгоритмов на учебных исполнителях. Это поможет изучить основные алгоритмические структуры с теоретической и практической стороны.

Основное достоинство блок-схем – наглядность представления структуры алгоритма. Это достигается изображением блок-схем стандартным способом – сверху вниз.

Алгоритмический язык есть текстовая форма описания алгоритма, которая близка к языку программирования, но как таковым ещё не является, и поэтому не имеет строгого синтаксиса. Для структурирования текста алгоритма

в алгоритмическом языке используются строчные отступы. При этом соблюдается правило: все конструкции одного уровня вложенности записываются на одном вертикальном уровне (отступе), а вложенные конструкции смещаются относительно внешней вправо. Это правило улучшает наглядность структуры алгоритма. Поэтому учителю желательно потратить определённое учебное время на формирование навыка правильной записи алгоритма.

После ознакомления с архитектурой исполнителя и способами записи алгоритмов следует приступить к решению задач, соответствующих приведенным выше дидактическим принципам. Только практическая работа на учебных исполнителях помогает освоить построение алгоритмов.

На практических занятиях используются следующие типы задач:

• составление простых линейных алгоритмов;

• составление и использование вспомогательных алгоритмов;

• составление циклических алгоритмов;

• использование ветвлений в алгоритмах;

•использование метода последовательной детализации при составлении сложных алгоритмов.

При разборе этой задачи необходимо обратить внимание учеников на два обстоятельства. Первое: управление учебным исполнителем для достижения поставленной цели будет происходить без обратной связи. В данном случае алгоритм управления будет иметь линейную структуру.

Второе: алгоритм зависит не только от сформулированной цели (искомого результата), но и от исходного состояния исполнителя. Если бы исходное состояние было другим, то был бы другим и алгоритм, несмотря на то, что в результате получается один и тот же рисунок. Для алгоритмов работы «в обстановке» начальное состояние исполнителя является исходным данным задачи. Состояние учебного исполнителя определяется местом его расположения на поле и ориентацией. Результатом же выполнения алгоритма

становится не только рисунок (главная цель), но и конечное состояние исполнителя.

Следующие задачи должны помочь составить и использовать вспомогательные алгоритмы. Обычно рассматривается такая задача: составить алгоритм рисования числа «1919».

Решая данную задачу можно поступить следующим образом: предложить ученикам написать алгоритм прежними средствами. Такое задание, очевидно, не вызовет энтузиазма учеников, поскольку принцип им уже понятен, а писать длинный линейный алгоритм довольно скучно. В этой ситуации вполне возможно самостоятельное «открытие» учениками идеи вспомогательного алгоритма. Обратив внимание на то, что в рисунке дважды присутствуют цифры «1» и «9», ученики могут прийти к идее отдельного описания алгоритмов рисования этих цифр, а затем использования их для получения четырехзначного числа «1919». После обсуждения этой идеи необходимо ввести понятие вспомогательного алгоритма и объясняет, как производится его описание и использование.

Умение использовать вспомогательные алгоритмы необходимо вырабатывать у учеников как можно раньше, уже на примерах линейных алгоритмов. Важнейший прием алгоритмизации и программирования – декомпозиция задачи, т.е. выделение в исходной задаче некоторых более простых подзадач. Алгоритмы решения таких подзадач называются *вспомогательными алгоритмами,* а реализующие их программы – *подпрограммами* (процедурами). Таким образом, решение исходной задачи разбивается на несколько алгоритмов: основной алгоритм и вспомогательные алгоритмы. Как правило, в основном алгоритме происходит многократное обращение к вспомогательному алгоритму.

Далее следует изучение циклов. Для составления циклических алгоритмов, следует сначала теоретически подготовить учащихся. Необходимо подробно разобрать циклические алгоритмы при помощи блок схем и алгоритмического языка. И только потом переходить на практику, иначе дети

могут не усвоить циклы, и действовать по примерам, не думая о содержании задачи.

Примером задачи на циклы может служить задача на составление алгоритм рисования горизонтальной линии, проведенной от края до края поля. Эта задача вносит в данную тему следующие новые элементы: управление с обратной связью; структурная команда цикла. Обратная связь между объектом управления и управляющей системой заключается в том, что перед выполнением каждого шага проверяется условие «впереди не край?». Если оно истинно, т.е. ответ положительный, то делается шаг, в противном случае выполнение цикла прекращается.

Команда цикла является *структурной командой* в отличие от *простых команд* «шаг», «поворот», «прыжок». Структурная команда включает в себя несколько действий: проверка условия, выполнение тела цикла, которое, в свою очередь, может состоять из нескольких команд.

И наконец, изучение основных алгоритмических структур заканчивается ветвлением. Тут можно предложить такую задачу: нарисовать орнамент, состоящий из квадратов, расположенных по краю поля. На примере этой задачи еще раз демонстрируется методика последовательной детализации. Причем, в отличие от предыдущих программ, здесь используется два шага детализации, поскольку в процедуре РЯД содержится обращение к процедуре следующего уровня – КВАДРАТ.

2.2.3 Методика обучения алгоритмизации на учебных исполнителях, работающих «в обстановке»

Обучение методам построения алгоритмов - один из наиболее отработанных разделов школьной информатики. Традиционно применяемым дидактическим средством в этом разделе являются учебные исполнители алгоритмов, которые удовлетворяют следующим условиям:

- это должен быть исполнитель, работающий «в обстановке»;

- этот исполнитель должен имитировать процесс управления некоторым реальным объектом (Черепахой, Роботом и др.);
- в системе команд исполнителя должны быть все структурные команды управления (ветвления, циклы);
- исполнитель позволяет использовать вспомогательные алгоритмы (процедуры).

Последние два пункта означают, что на данном исполнителе можно обучать структурной методике алгоритмизации. Всякое педагогическое средство должно соответствовать поставленной учебной цели. Главной целью раздела алгоритмизации является овладение учащимися структурной методикой построения алгоритмов.

Обучение алгоритмизации (программированию) для исполнителя нужно строить на последовательности решаемых задач. Эта последовательность должна определяться следующими принципами:

- от простого к сложному — постепенное усложнение задач;
- новизна — каждая задача вносит какой-то новый элемент знаний (новая команда, новый прием программирования);
- наследование — следующая задача требует использования знаний, полученных при решении предыдущих задач.

Традиционно в школьной информатике используются два способа описания алгоритмов: блок- схемы и учебный алгоритмический язык. В базовом курсе информатики необходимо использовать обе эти формы. Основное достоинство блок-схем - наглядность алгоритмической структуры. Однако это качество проявляется лишь в том случае, если изображение блок-схемы происходит стандартным способом. Основным следствие освоения учениками структурной методики должно стать умение при построении алгоритмов «мыслить структурами». Структурно изображенные блок-схемы на рис. 1(а,б) помогают такому видению алгоритма.

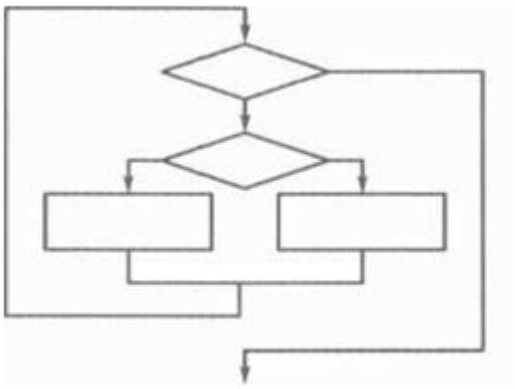

рис.1, а.

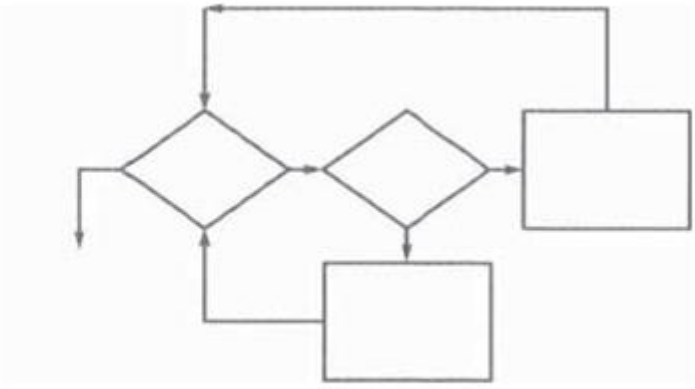

рис.1, б

Алгоритмический язык — это текстовая форма описания алгоритма. Она ближе к языкам программирования, чем блок-схемы. Однако, это еще не язык программирования. Поэтому строгого синтаксиса в алгоритмическом языке нет. Для структурирования текста алгоритма на АЯ используются строчные отступы. При этом соблюдается следующий принцип: все конструкции одного уровня вложенности записываются на одном вертикальном уровне, вложенные конструкции сметаются относительно внешней вправо. Соблюдение этих правил улучшает наглядность структуры алгоритма, однако не дает такой степени наглядности, как блок- схемы.

В настоящее время школьный курс информатики включает в себя программирование в системе КуМир Рис.2, которая разработана исходя из потребностей российского образования:

1. Свободно распространяемая
2. Многоплатформенная
3. Нулевые требования к ресурсам
4. Поддерживается российской академией наук
5. В числе разработчиков – авторы многих учебников и учебных пособий по информатике.

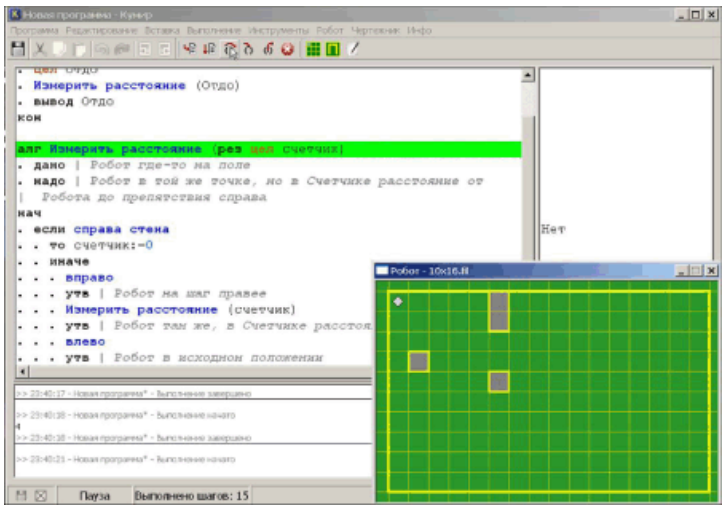

Рис.2 Система программирования КуМир

Система КуМир нацелена на проведение эффективного практикума по основам алгоритмизации.

1. Простой язык, интегрированная среда.
2. Постоянная полная диагностика синтаксиса в процессе редактирования программы.
3. Возможность автоматической проверки при самостоятельной работе.
4. Имеет несколько исполнителей.

2.2.4 Методические проблемы изучения алгоритмов работы с величинами

Есть две стороны в обучении алгоритмизации: обучение структурной методике построения алгоритмов, обучение методам работы с величинами.

Знакомясь с программным управлением исполнителями, работающими «в обстановке», ученики осваивали методику структурного программирования. При этом понятие величины могло быть не затронуто вовсе. Однако с величинами ученики уже могли встречаться в других темах базового курса: в частности, при изучении баз данных, электронных таблиц. Теперь требуется объединить навыки структурной алгоритмизации и навыки работы с величинами.

Обсуждение методических вопросов изучения темы «Алгоритмы работы с величинами» будем проводить в программистском аспекте. Составление любой

программы для ЭВМ начинается с построения алгоритма. Как известно, всякий алгоритм (программа) составляется для конкретного исполнителя, в рамках его системы команд. Рис.3 Программист составляет программу на том языке, на который ориентирована СП. Иногда в литературе по программированию такой комплекс называют «виртуальной ЭВМ» (компьютер).

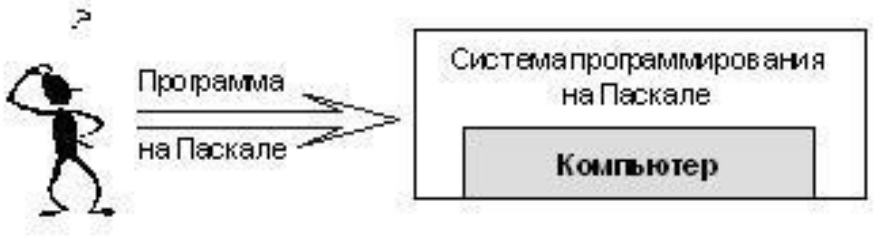

Рис.3 Взаимодействие программиста с компьютером

При изучении элементов программирования в базовом курсе необходимо продолжать ту же структурную линию, которая была заложена в алгоритмическом разделе. Поэтому при выборе языка программирования следует отдавать предпочтение языкам структурного программирования.

Процесс программирования подразделяется на три этапа:

- составление алгоритма решения задачи;
- составление программы на языке программирования;
- отладка и тестирование программы.

Для описания алгоритмов работы с величинами следует, как и раньше, использовать блок-схемы и учебный алгоритмический язык. Описание алгоритмов должно быть ориентировано на исполнителя со структурным входным языком независимо от того, какой язык программирования будет использоваться наследующем этапе.

Компьютер работает с информацией. Информация, обрабатываемая компьютерной программой, называется данными. Величина — это отдельный информационный объект, отдельная единица данных. Команды в компьютерной программе определяют действия, выполняемые над величинами.

Важнейшим понятием, которое должны усвоить ученики является следующее: всякая величина занимает свое определенное место в памяти компьютера - ячейку памяти. В результате в сознании учеников должен закрепиться образ ячейки памяти, сохраняющей величину. Термин «ячейка памяти» рекомендуется употреблять и в дальнейшем для обозначения места хранения величины. С понятием величины необходимо связать ее основные свойства: значение, имя, тип.

Действия над величинами, определяемые алгоритмом (программой), основываются на следующей иерархии понятий: операция - выражение - команда или оператор - система команд (рис. 4).

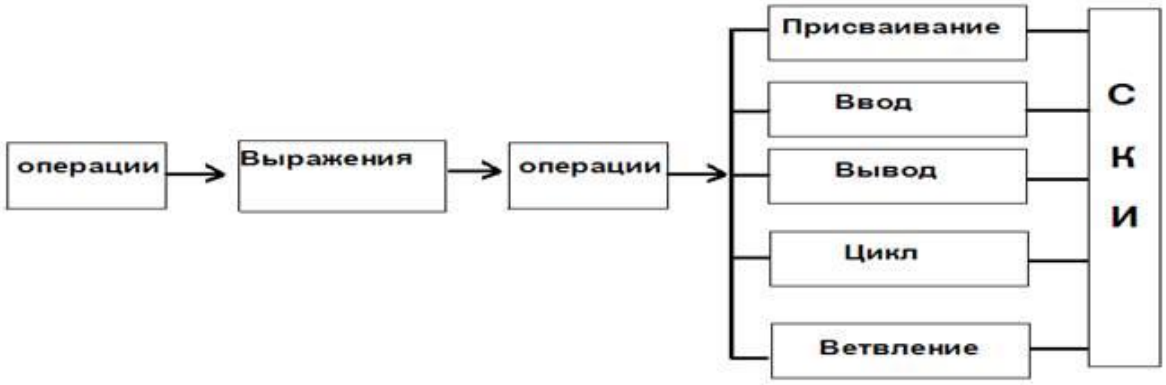

Рис.4 Действия над величинами

В большинстве случаев непонимание некоторыми учениками программирования происходит от непонимания смысла присваивания. Поэтому учителям рекомендуется обратить особое внимание на этот вопрос. В последние несколько лет школьный алгоритмический язык включается как один из предлагаемых в текстах задач ОГЭ по информатике.

2.3 Методические рекомендации учителю информатики при обучении темы «Основы алгоритмизации»

Тема «Основы алгоритмизации» изучается на всех ступенях средней школы, но на разном уровне. В начальной школе происходит знакомство на интуитивном уровне с понятиями алгоритма, алгоритмических конструкций, основ алгебры логики. В качестве учебных задач рассматривают бытовые, игровые, сказочные алгоритмы.

В средних классах школы в рамках данной темы происходит уточнение понятия алгоритма, основы алгебры логики излагаются на более формальном уровне. При решении учебных задач учащиеся знакомятся с разными способами записи алгоритмов, изучают свойства алгоритма, рассматривают некоторые алгоритмы (алгоритм Евклида, *сортировка* данных и т.д.).

В старших классах, и особенно в классах физико-математического, информационно-технологического профилей, изучение этой темы строится в соответствии со Стандартом. Успешность учащихся в освоении этой темы во многом зависит от приобретенных ими общеучебных навыков в предыдущие годы обучения. Без сомнения, навыки, составляющие основу алгоритмического мышления, должны формироваться, начиная с младших классов.

Требования к знаниям и умениям учащихся по линии алгоритмизации
Учащиеся должны знать:

- что такое алгоритм;
- какова роль алгоритма в системах управления;
- в чем состоят основные свойства алгоритма;
- способы записи алгоритмов: блок-схемы, учебный алгоритмический язык;
- основные алгоритмические конструкции: следование, ветвление, цикл; структуры алгоритмов;
- назначение вспомогательных алгоритмов; технологии построения сложных алгоритмов:

- метод последовательной детализации и сборочный (библиотечный) метод;
- основные свойства величин в алгоритмах обработки информации: что такое имя, тип, значение величины; смысл присваивания;
- назначение языков программирования;
- разницу между языками программирования высокого уровня и машинно-ориентированными языками;
- правила представления данных на одном из языков программирования высокого уровня (например, на Паскале);
- правила записи основных операторов: ввода, вывода, присваивания, цикла, ветвления; правила записи программы;
- понятие: трансляция;
- назначение систем программирования;
- содержание этапов разработки программы: алгоритмизация – кодирование – отладка – тестирование.

Учащиеся должны уметь:

- пользоваться языком блок-схем, понимать описания алгоритмов на учебном алгоритмическом языке;
- выполнять трассировку алгоритма для известного исполнителя;
- составлять несложные линейные, ветвящиеся и циклические алгоритмы управления одним из учебных исполнителей;
- выделять подзадачи; определять и использовать вспомогательные алгоритмы;
- составлять несложные программы решения вычислительных задач с целыми числами;
- программировать простой диалог;
- работать в среде одной из систем программирования (например, Турбо Паскаль);
- осуществлять отладку и тестирование программы.

Тем не менее сама тема имеет очень важное, в том числе культурологическое, значение, и в профильной школе должна быть рассмотрена обязательно, а в основной – по крайней мере должен быть сформулирован сам факт существования алгоритмически неразрешимых задач. Ведь сам по себе этот факт является весьма неочевидным, о чем говорят и многовековые попытки решить те или иные проблемы (задача доказать их возможную алгоритмическую неразрешимость при этом изначально даже не ставилась). Ученики также воспринимают данный факт с недоверием, причем даже после доказательства неразрешимости. Ведь их практический опыт подсказывает, что очень часто они сами могут эту проблему для конкретной программы решить, в том числе и доказать, что определенная программа на конкретных входных данных зациклится. Здесь очень важно провести грань между решением задачи для частного случая и построением универсального алгоритма. Действительно, проблему останова для конкретной программы или даже класса программ решить можно. Так, для программы, состоящей только из линейных конструкций, легко показать, что она всегда закончит свою работу. Приведенное же доказательство говорит о невозможности построения общего алгоритма, пригодного для любых программ и входных данных.

2.4 Проблемы учащихся, возникающие при изучении раздела «Основы алгоритмизации»

При изучении данной темы можно столкнуться со следующими сложностями:

- непонимание учащимися понятия «алгоритм»;
- неправильное приведение примеров алгоритма (не выполняются все его свойства, команды не входят в систему команд исполнителя);
- неправильное представление алгоритма в виде блок-схемы;
- неправильное использование простейших алгоритмических конструкций;

С первой из предложенного списка трудностей сталкиваются уже на начальном этапе изучения данной темы. В связи с этим раскрытие этого

вопроса дается с использованием примеров, основанных на жизненном опыте учащихся. А также огромное значение имеет тот факт, что знакомство с алгоритмом происходит через раскрытие его неотъемлемых атрибутов или свойств, которые и позволяют некий текст (при условии представления его в словесной форме) воспринимать как алгоритм. Примеры, приводимые учителем во время урока должны отличаться разнообразием, касаться различных сфер человеческой деятельности. Это могут быть зарядка, выполняемая каждым человеком по утрам, переход через улицу, разведение костра, пришивание пуговицы, приготовление блюда по поваренной книги и другие.

Следующим этапом является закрепление понимания учащимися понятия алгоритм, и здесь возникает следующая проблема. Несмотря на то, что учащиеся знают определение и основные свойства они не могут правильно сформулировать примеры, которые являлись бы алгоритмами. Чаще всего забываются какие-нибудь важные атрибуты, им не уделяется достаточного внимания. Например, распространенной является ошибка, когда учащиеся забывают что необходимо, чтобы все действия приводили к какому-нибудь результату, а не выполнялись просто так.

Для закрепления основных понятий и для преодоления вышеназванных трудностей, связанных с определением алгоритма, полезно рассмотреть с учениками несколько заданий следующего содержания:

- выполнить роль исполнителя: дан алгоритм, формально исполнить его;
- определить исполнителя и систему команд для данного вида работы;
- в рамках данной системы команд построить алгоритм;
- определить необходимый набор исходных данных для решения задачи.

Следующая проблема связана с неправильным представлением алгоритма в виде блок-схемы. Чтобы избежать серьезных проблем с изучением этого подраздела, нужно выработать у учащихся практические навыки по разработке блок-схемы, для этого необходимо соблюдать единообразие представления основных алгоритмических конструкций, а также осуществлять многократное повторение упражнений на использование данных схем.

Большую сложность у учащихся вызывает изучение базовых алгоритмических конструкций. Подробно необходимо остановить на каждой из них: линейность, ветвление и цикл.

Линейные алгоритмы воспринимаются легче всего, но необходимо подвести учащихся к выводу о невозможности их использования для большого круга задач.

При разборе конкретного алгоритма ветвления на схеме следует отметить разными цветами два возможных способа выполнения команды, точку входа и выхода из команды. Обязательно вслух проговаривается алгоритм с использованием ключевых слов «если», «то», «иначе». Это позволяет, во-первых, лучше усвоить данную структуру, а во-вторых, осуществить пропедевтику записи алгоритма с помощью псевдокода. Обращается внимание на то, что слева всегда записывается действие, которое будет выполнено в случае соблюдения условия, т. е. путь «да», а справа — действие, выполняемое при несоблюдении условия, т. е. путь «нет». Необходимо обратить внимание учащихся на то, что команда ветвления заканчивает свою работу, как только выполнится одна из двух предложенных команд.

Следует обсуждать с учащимися необходимость использования команды ветвления. Для этого можно задать следующие вопросы: почему алгоритм решения задачи не может иметь линейную структуру? Какое условие надо проверять при выполнении алгоритма? Какие действия выполняются при соблюдении условия, а какие — при его несоблюдении? Какая форма команды ветвления применена? В каких задачах используется данная структура алгоритма? и т. п.

В ряде учебников первой изучаемой конструкцией после команды следования является цикл, поскольку это дает возможность сократить запись алгоритма. Как правило, это конструкция «повторить n раз». Такой подход приводит к трудностям в освоении циклов как структуры организации действий, качественно отличающейся от линейной. Во-первых, другие разновидности цикла с предусловием и с постусловием (цикл «пока», цикл с

параметром, цикл «до») воспринимаются как изолированные друг от друга и главный признак — повторяемость действий — не выступает в качестве системообразующего. Во-вторых, без внимания остаются опорные умения, которые необходимы при разработке циклов: правильное выделение условия продолжения или окончания цикла, правильное выделение тела цикла. Проверка условия в цикле «повторить п раз» практически не видна, и циклический алгоритм часто продолжает восприниматься учащимися как линейный, только иначе оформленный, что порождает неверный стереотип у учащихся в восприятии циклов вообще. Поэтому методически более целесообразным является изучение вначале команды ветвления, в которой используется условие, а уже затем команды повторения.

Изучение команды повторения следует начинать с введения цикла с постусловием, поскольку в этом случае учащемуся дается возможность вначале продумать команды, входящие в цикл, и только после этого сформулировать условие (вопрос) повторения этих команд. Если же сразу вводить цикл с предусловием, то учащимся придется выполнять оба эти действия одновременно, что снизит эффективность проведения занятий. В то же время цикл с постусловием рассматривается в качестве подготовки восприятия учащимися цикла с предусловием, обеспечивает перенос знаний на другой вид команды повторения, дает возможность работать по аналогии. Следует обратить внимание учащихся на то, что данные виды цикла отличаются по месту проверки условия, по условию возврата к повторению выполнения тела цикла. Если в команде повторения с постусловием тело цикла выполняется хотя бы один раз, то в команде повторения с предусловием оно может ни разу не выполняться.

Для решения последней трудности, связанной с отсутствие понимания принципа перевода алгоритма на формальный язык можно предложить следующее решение: необходимо начать изучение языка программирования с использования базовых алгоритмических конструкций, а также необходимо

использовать параллельно различные формы представления алгоритма, что позволит более формально подойти к конкретному алгоритму.

2.5. Виды и формы контроля знаний на уроках информатики

Чтобы своевременно выявить проблемы освоения материала, нужен постоянный контроль знаний и умений учащихся по теме, который направлен на повышение эффективности учебного процесса.

Проверочно-оценочная деятельность учителя – неотъемлемая часть всей педагогической работы, важный фактор улучшения качества обучения. Часто для контроля знаний ограничиваются устным опросом школьников, в процессе которого лишь пересказывается текст учебника. Для более качественной проверки нужно применять различные виды и формы контроля знаний такие как:

1. Диктант. Эта форма письменной проверки знаний дает возможности подготовить учащихся к усвоению нового материала, обобщению и систематизации пройденного, хорошей отработки навыков и умений при выполнении элементарных операций. Диктант представляет собой перечень вопросов, которые могут: диктоваться преподавателем через определенный интервал времени, демонстрироваться через калейдоскоп поочередно; быть записанными на магнитофон; быть представленными в виде таблиц с набором ответов.
2. Самостоятельная работа. Система самостоятельных работ должна обеспечивать усвоение необходимых знаний и навыков и их проверку; отражать все основные понятия, предусмотренные программой; формировать приемы учебной работы; подводить учащихся к самостоятельному нахождению приемов; обеспечивать повторяемость одних и тех же вопросов в различных ситуациях. (Приложение 2)

3. Тест. Тест представляет собой системы небольших по объему заданий, охватывающих в совокупности большой круг вопросов отдельных глав учебника информатики и курса в целом. Тесты представлены тремя видами в двух вариантах:

- первый вид тестов (предполагает заполнение пропусков < многоточий> таким образом, чтобы получилось истинное высказывание. Учащиеся ограничиваются тем, что вместо многоточий они указывают одно – два слова, которые считают необходимо недостающими);
- второй вид тестов (учащиеся должны установить, истинно или ложно каждое из предложенных высказываний. Учащиеся должны не просто дать ответ <да> или <нет>, а проявить умение рассуждать, делать соответствующие выводы, распознавать верно сформулированное математическое предложение от неверного);
- третий вид тестов (предлагает на выбор несколько ответов, среди которых есть верный и неверный и ответ, предполагающий отказ от выполнения задания. Количество ответов ограничено тремя наиболее значимыми, так как набор ответов должен быть легко обозримом для учащихся).

Большую популярность получили онлайн-тесты, которые не требуют дополнительной проверки и мгновенно оценивают работу. Например, ISPRING, LearningApps и др. (Приложение 3)

4. Контрольная работа. Письменную проверку знаний и умений учащихся необходимо проводить на различных этапах усвоения изученного, что даст возможность несколько раз получить информацию об усвоении одного и того же материала. С этой целью целесообразно проводить различного рода контрольные работы, которые можно разделить на два вида:

1.проверочные контрольные работы – предназначены для проверки усвоения отдельного фрагмента курса в период изучения темы;

2.итоговые контрольные работы – являются завершающим моментом повторения в конце года. (Приложение 4) Необходимым компонентом этих работ служат задания на повторение основных теоретических вопросов.

Контрольная работа является составной частью процесса обучения и несет на себе образовательную, воспитательную и развивающую функции.

5. Применение теории на практике. Уровень изучения материала по алгоритмизации и программированию можно выявить при проведении практических работ, где наглядно видно как ребенок может применять полученные знания и владеть ими. Как правило, учащиеся с хорошей теоретической подготовкой легко ориентируются в поставленной задаче и успешно ее выполняют, что нельзя сказать об учащихся с низким уровнем знаний.

2.6 Проблемы учителя, возникающие при изучении раздела «Основы алгоритмизации»

При изучении алгоритмизации, а тем более программирования школьники традиционно испытывают затруднения, так как материал требует наличия хорошо развитого абстрактного, логического мышления и мало привязан к реальным событиям жизни. Программирование, во первых, является объективно сложным предметом, во вторых учащиеся слабо мотивированы на его изучение.

По мнению многих ученых и специалистов в области образования вопросы, связанные с алгоритмизацией и программированием являются

фундаментальными и обязательно должны изучаться на вводных курсах информатики вне зависимости от дальнейшего профиля обучения.

В настоящее время уменьшение количества часов на изучение раздела алгоритмизации и программирования в старшей школе объективно связано с бурным развитием информационных технологий. Чрезмерное увлечение готовым прикладным программным обеспечением вытеснило изучение этих вопросов не только из некоторых профильных курсов, но даже из ряда учебников базового курса. Несмотря на перенасыщенность школ компьютерной техникой, на всеобщую доступность компьютеров и сети Интернет, нет положительных сдвигов в уровне общей подготовки учащихся.

Стоит отметить, что изучение информатики в школьном курсе в рамках федерального государственного образовательного стандарта согласно новым образовательным стандартам основного общего образования второго поколения начинается с седьмого класса. Однако, по мнению педагогов, интерес учеников к изучению информатики в целом и основ программирования в частности возникает гораздо раньше. Решением названной проблемы может служить школьный компонент, за счет которого становится возможным изучение информатики с пятого класса по учебному плану в качестве уроков или внеклассных мероприятий. Перед учителем возникает задача поиска методов и средств, которые позволят сделать процесс обучения информатике увлекательным и познавательным с тем, чтобы заинтересовать школьников.

Существует еще ряд проблем, таких как:

- недостаточное количественные и качественные характеристики программного обеспечения, которое предназначается для постоянного поддержания обучения учащихся соответственной возрастной категории;
- соблюдение санитарногигиенических норм и т.д.
- резкая смена ведущей деятельности с игровой на учебную (в особенности для младших школьников). На данном этапе необходимо организовать плавный переход от преимущественно игровой деятельности к учебной,

используя по возможности игровые дидактические компьютерные технологии.

- отсутствие современного взгляда на информационную деятельность, как на вид творческой деятельности, которая требует кроме развитого логического и системного мышления способность мыслить находчиво и продуктивно, ориентирует учителя информатики на развитие фантазии и творческого воображения учащихся .

На уроках информатики формируется системное восприятие мира, понимание единых информационных связей различных природных и социальных явлений, развивается системное мышление, уровень которого, во многом определяется способностью оперативно обрабатывать информацию и принимать на ее основе обоснованные решения, что требует от школьников дополнительных возможностей, а от педагогов , применение все новых методов и средств обучения.