

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ  
«АКАДЕМИЯ СОЦИАЛЬНОГО УПРАВЛЕНИЯ»

Факультет профессиональной переподготовки педагогических работников

Кафедра «Методика преподавания технологии, информатики и ИКТ»

Итоговая аттестационная работа

слушателя программы профессиональной переподготовки  
«Содержание и методика преподавания предмета «Информатика»  
(название программы)

Шептунова Алексея Сергеевича

Дифференцированное обучение информатике  
на примере темы «Алгоритмизация и  
программирование» в 9 классе

Научный руководитель  
Шутикова Маргарита Ивановна  
д.п.н., профессор

Москва 2018 г.

## ОГЛАВЛЕНИЕ

Введение.....	2
Глава 1. Психолого-педагогические основания дифференцированного обучения информатике .....	5
1.1. Дифференцированное обучение: подходы, классификация, достоинства и недостатки.....	5
1.2. Индивидуальные особенности учащихся как основа дифференциации .....	10
1.3. Виды и формы дифференцированного обучения информатике .....	12
Глава 2. Методика изучения раздела «Алгоритмизация и программирование с учетом дифференциации».....	17
2.1. Описание раздела «Алгоритмизация и программирование» в программах, реализующих ФГОС ООО .....	17
2.2. Методические особенности изучения раздела «Алгоритмизация и программирование» .....	22
2.3. Определение эффективности методики дифференцированного обучения информатике (на примере темы "Алгоритмизация и программирование" в 9 классе) .....	51
Заключение .....	56
Список источников и литературы .....	59

## **Введение**

С 2010 года в Концепции модернизации российского образования большое внимание уделяется реализации личностно-ориентированного обучения. Одним из значимых моментов достижения высокого качества общего образования является обеспечение дифференциации и индивидуализации образования.

Осуществление личностно-ориентированного обучения на основе дифференцированного подхода требует учета уровня интеллектуального развития обучающегося, его подготовки по данному предмету, его способностей, его задатков, индивидуальных особенностей обучающегося.

Необходимость такого учета очевидна, так как учащиеся в значительной степени отличаются друг от друга по этим параметрам.

В современной школе, где учитель занимается одновременно с большой группой учащихся, необходимость учета индивидуальных особенностей является основной проблемой индивидуализации обучения. Частично эту проблему удастся решать при помощи дифференциации обучения.

Дифференциация обучения применима для многих предметов, в том числе и для информатики. Большой потенциал информационных технологий, привнесенный в учебный процесс информатикой, широкие межпредметные связи информатики с другими предметами, прикладное назначение информатики позволяют наиболее полно реализовать дифференцированное обучение.

Реализация дифференциации предполагает учет таких особенностей учащихся, которые влияют на их учебную деятельность и от которых зависят результаты обучения. Это могут быть различные физические и психические качества и состояния личности: особенности познавательных процессов и памяти, черты характера, свойства нервной системы, силы воли, мотивации, способности, одаренность.

Для эффективной реализации дифференцированного обучения необходима качественная диагностика индивидуальных особенностей учащихся, которая позволяла бы учителю и психологу своевременно выявлять состояние каждого школьника. Использование результатов диагностики позволит спроектировать эффективную методическую систему дифференцированного обучения информатике на основе индивидуальных особенностей учащихся.

Актуальность темы основывается на следующих фактах:

- информационные технологии проникают во все сферы деятельности человека, что, в свою очередь, ведет к повышению спроса на специалистов в области программирования;

- практика дифференцированного подхода к обучению становится жизненно важной в эпоху гуманизации образования.

Объектом исследования является процесс дифференцированного обучения информатике в средней и старшей школе.

Предмет исследования: дифференцированное обучение школьников алгоритмизации и программированию.

Цель исследования: разработать систему уроков, использующих дифференцированный подход в изучении алгоритмизации и программирования.

Задачи исследования:

- провести теоретический анализ дидактической и психологической литературы по теме исследования;

- исследовать возможность разработки дифференцированного подхода в обучении алгоритмизации и программированию, опираясь на современные достижения педагогики, информатики, психологии;

- разработать систему уроков по теме «Алгоритмизация и программирование», использующих дифференцированный подход к обучению.

Методы исследования:

- теоретические: анализ научной литературы, сравнение, классификация;
- эмпирические: проектирование;
- диагностические: тестирование в процессе определения эффективности предложенной системы уроков.

Методологической основой исследования являются труды Ю.К. Бабанского, Н.К. Гончарова, И.С. Якиманской, Л.В. Виноградовой, В.А. Смирнова, С.В. Алексеева, М.П. Лапчика.

Практическая значимость состоит в применении разработанных уроков с дифференцированными заданиями в дальнейшей работе в девятом классе для развития индивидуально-личностных особенностей учащихся.

# **Глава 1. Психолого-педагогические основания дифференцированного обучения информатике**

## **1.1. Дифференцированное обучение: подходы, классификация, достоинства и недостатки**

В настоящее время в педагогической литературе не существует единого обобщенного понятия дифференциации обучения.

В трудах Ю.К. Бабанского [1], [2] и других дифференциация рассматривается как особая форма организации обучения с учетом типологических индивидуально-психологических особенностей обучающихся и особой взаимосвязи учителя и учеников.

В работах И. Унта [16] под дифференциацией обучения подразумевается учет особенностей обучающихся в той форме, когда они разбиваются на группы на основании каких-либо особенностей для отдельного обучения по разным учебным планам или программам.

Согласно И.С. Якиманской [18], дифференциация обучения, основанная на индивидуальном подходе к каждому ученику, должна обеспечивать реальные условия для саморазвития и самореализации личности в процессе овладения знаниями.

С социальной точки зрения целью дифференциации является целенаправленное взаимодействие на формирование творческого, интеллектуального, профессионального потенциала общества, вызываемого стремлением общества к наиболее полному и рациональному использованию возможностей каждого члена общества в его взаимоотношениях с окружением.

С психолого-педагогической точки зрения целью дифференциации является индивидуализация обучения, основанная на создании оптимальных условий для выявления задатков, развития интересов и способностей каждого школьника.

С дидактической точки зрения целью дифференциации является решение проблем школ путем создания новой методической системы дифференцированного обучения учащихся, основанной на принципиально новой мотивационной основе.

Школа призвана не только формировать основы знаний (развивать когнитивный компонент), но развивать творческое мышление, учить самостоятельно добывать знания, использовать полученные знания в учебных и жизненных ситуациях, то есть развивать познавательную активность, инициативность, самостоятельность.

Личностно-ориентированная модель обучения направлена на создание условий для максимального развития/раскрытия индивидуальных особенностей школьников. Основой личностно-ориентированной модели может быть: выбор программы образования соответствующего уровня; сочетание дифференциации и интеграции; создание системы деятельности учащихся, максимально развивающей их способности; создание благоприятных условий в социуме.

Для осуществления личностно-ориентированного обучения с позиций дифференциации содержания обучения необходимы:

- разные варианты программ, учебников, дидактических материалов, позволяющих на едином базовом содержании варьировать и индивидуализировать процесс обучения;
- новые формы проведения групповых и индивидуальных занятий в целях активизировать творческие способности учащихся;
- постоянное внимание к анализу и оценке способов учебной деятельности ученика, побуждающих его к осознанию результатов и самого процесса деятельности.

Дифференцированный подход к организации процесса обучения на уроках информатики позволяет активно развивать творческое и критическое мышление учащихся, находить подходы к мотивации обучения.

Дифференцированный подход можно применять на различных этапах урока.

На этапе введения нового материала можно осуществить сочетание двух подходов: дифференцированного и проблемного:

- учитель указывает лишь результат, формулирует саму проблему, ученики самостоятельно ведут поиск;
- учитель указывает на проблему, но не сообщает конечного результата, ученики сами формулируют проблему;
- учитель не указывает на проблему, а постепенно подводит учащихся к тому, что они самостоятельно формулируют проблему.

На этапе самостоятельной работы учащихся по изучению нового материала, применению изученной теории к решению задач - самостоятельная работа учащихся может быть разделена на несколько групп в зависимости от степени помощи со стороны учителя:

- учитель указывает тип задачи, правила, на которые опирается решение задачи;
- учитель дополняет задачу чертежом, схемой и т.п.;
- условие задачи записывается в виде таблицы, матрицы, графика;
- указывается алгоритм решения задачи;
- приводится аналогичная задача, решенная ранее;
- учитель объясняет ход выполнения подобной задачи;
- учителем предлагается решение вспомогательной задачи, наводящее на решение основной задачи;
- учитель наводит учащихся на поиск решения с помощью ассоциаций;
- учитель указывает на причинно-следственные связи, необходимые для решения задачи;
- учитель заранее указывает на результат решения или дает ответ;
- сложная задача разбивается на ряд элементарных;
- учитель задает наводящие вопросы;



- учитель указывает теоремы или формулы, на основе которых решается задача;

- учитель предупреждает о наиболее типичных ошибках или неправильных подходах;

- использование опорных конспектов;

- использование рабочих тетрадей с печатной основой.

На этапе работы с учебником одной группе учащихся можно предлагать прочитать материал и выделить основную мысль, другой группе - составить опорный план-конспект.

На этапе проверки готовности к уроку можно проводить письменный опрос по двум основным компонентам содержания: формулировка определений, правил и решение задач.

На этапе выдачи домашнего задания можно использовать дифференцированное домашнее задание: первая часть домашнего задания предназначается для всего класса, вторая часть представляет некоторую дополнительную трудность.

Для обеспечения разноуровневого обучения требуется определить индивидуально-личностные особенности учащихся (требуется психологическая диагностика), реализовывать принцип воспитывающего обучения, проводя диагностику ценностных ориентаций, разработать гибкое структурирование учебного материала, обеспечивающее учебно-познавательные способности учеников. Причем главный акцент должен быть направлен на дифференциацию учебного материала, а не на деление учащихся по способностям. Ученику должна быть предоставлена возможность выбора, самоопределения в изучении учебного материала.

Особенностью дифференцированного обучения, дифференциации по уровню умственного развития, по уровню навыков и умений является то, что в ней, наряду с положительными, имеются и отрицательные стороны.

К положительным можно отнести:

- исключается уравниловка детей;
- усвоение материала в слабых группах идет легче;
- более сильные учащиеся продвигаются быстрее в изучении материала;
- появляется возможность более эффективно работать со слабыми детьми;
- повышается уровень мотивации учащихся;
- повышается уровень самосознания учащихся.

К отрицательным можно отнести:

- более явным становится социально-экономическое неравенство;
- деление детей по уровню развития не является гуманным;
- перевод в слабые группы плохо отражается на самооценке детей;
- понижается уровень самосознания (сильные группы начинают чувствовать свою исключительность);
- понижается уровень мотивации (в слабых группах);
- слабые учащиеся, отделенные от сильных, лишаются возможности получать от них помощь, соревноваться с ними;
- диагностика для разделения на сильных и слабых несовершенна.

Несмотря на все минусы и несовершенства, дифференцированная технология наиболее понятна и осуществима в массовой школе.

Особого внимания требуют учащиеся с высоким уровнем подготовки. Чаще всего при общеклассной работе они оказываются занятыми не в полную меру. Такие учащиеся нуждаются в заданиях повышенной трудности, нестандартных заданиях, заданиях творческого характера.

Учащиеся с высоким уровнем подготовки могут выполнять поручения, направленные на практическое применение знаний, более глубокое их осмысление. Учащемуся с высоким уровнем подготовки можно поручить:

- проверку заданий, выполненных другими учащимися;
- помощь другим учащимся при выполнении заданий;
- выполнение обязанностей консультанта при работе в группах, при

проведении практических работ.

Так же в особом внимании нуждаются слабые ученики. Основная задача – довести их до уровня средних учеников, обучить приемам рациональной умственной деятельности. Со временем помощь учителя должна уменьшаться, самостоятельность учащегося должна увеличиваться. С этой целью возможно использование карточек для индивидуальной работы, образцы выполненных заданий (образцы), алгоритмы действий и т.п.

## **1.2. Индивидуальные особенности учащихся как основа дифференциации**

Основной вопрос, который возникает при попытке применения дифференцированного обучения – как разделить учащихся на группы?

Необходима психолого-педагогическая диагностика, чтобы при помощи методов изучения личностных особенностей и педагогического наблюдения произвести деление учащихся на группы сменного состава.

Необходимо подобрать критерии дифференциации учащихся на гомогенные группы при внутренней дифференциации, разработать дидактическое обеспечение по усвоению содержания учебного материала. Для деления учащихся на гомогенные группы можно использовать следующую типологизацию:

- интерес учащихся к изучению предмета,
- интересы учащихся;
- возможности учащихся в процессе овладения знаниями;
- мотивы учения учащихся,
- причины, не позволяющие учащимся учиться лучше.

При изучении нового материала учащихся можно разделять на группы по качеству знаний: первая группа – сильные (высокая обучаемость, высокая обученность); вторая группа – средние (средняя обучаемость, средняя обученность); третья группа – слабые учащиеся (низкая обучаемость, низкая обученность).

обученность). Или по способу мышления: первая группа – учащиеся со стандартным мышлением, вторая группа – учащиеся с творческими способностями.

Учитель объясняет новый материал для всего класса. Затем первой группе учащихся (сильная группа) учитель дает творческое задание. Вторая и третья группа прослушивают повторное объяснение новой темы, затем учащиеся второй группы (средние ученики) получают задания с элементами творческого характера. Для учащихся третьей группы материал объясняется еще раз с использованием учебника, наглядного материала, учащиеся третьей группы получают практическое задание.

Разноуровневые группы подвижны по своему составу – если учащийся второй или третьей группы работает успешно, справляется с заданиями, он может перейти в первую группу.

Целью дифференцированного обучения является предоставление каждому учащемуся возможности реализации своих способностей на максимальном для себя уровне, но не ниже базового.

Для реализации дифференцированного обучения необходимы диагностические методики, обеспечивающие получение необходимой информации об ученике. Особенности психической организации учащегося, которые влияют на успешность освоения учебной программы, можно подразделить на:

- особенности когнитивной сферы: особенности интеллекта или то, как ученики получают, хранят, используют информацию (потенциальные возможности ученика в учебе);
- личностные факторы: особенности мотивации, самооценки, межличностных отношений (возможные проблемы личностного плана, которые мешают учащемуся).

В основу уровневой дифференциации таким образом должны быть положены особенности интеллекта учащегося. В основу определения уровня

интеллектуального развития ученика должны быть положены тесты, определяющие уровень мышления, памяти, внимания.

На успешность обучения также могут влиять межличностные отношения с одноклассниками, учителями или родителями. Для успешного усвоения образовательной программы важны представления подростков о себе, о своем месте среди сверстников (подросток воспринимает себя через отражение в межличностных отношениях со сверстниками), взаимоотношения с родителями и учителями (взрослыми).

Также для школьного обучения важна мотивация. Мотивы обучения могут быть различны: познавательный мотив (ученик заинтересован в получении новых знаний по предмету), мотив достижения успеха (не зависимо от того, нравится ученику предмет или нет, ученик заинтересован в высокой оценке), мотив общения (общение для подростков является ведущим видом деятельности), мотив школьной тревожности и мотив негативного отношения к школе (последние два блокируют учебную деятельность). Важно выяснить, какой из мотивов является ведущим.

### **1.3. Виды и формы дифференцированного обучения информатике**

По индивидуально-психологическим особенностям детей, составляющим основу формирования гомогенных групп, различают дифференциацию:

- по возрастному составу (школьные классы, возрастные параллели, разновозрастные группы);
- по полу;
- по области интересов (гуманитарные, физико-математические, технические, биолого-химические, информационные и т.д.);
- по уровню умственного развития;
- по личностно-психологическим типам (типу мышления, типу характера, темпераменту и т.п.);

- по уровню здоровья (физкультурные группы, часто болеющие, группы слабослышащих/слабовидящих и т.д.).

По организационному уровню однородных групп выделяют дифференциацию:

- по типу школ (спецшколы, гимназии, лицеи, колледжи и т.п.);
- внутришкольную (уровни, профили, углубленное изучение предметов, уклоны);
- по параллелям (группы и классы различных уровней);
- межклассную (факультативы);
- внутриклассную или внутреннюю (группы в составе класса).

В особую дифференциальную группу может быть выделена любая группа, обучение в которой отличается какими-либо условиями или компонентами образовательного процесса. По этим признакам выделяют следующие виды дифференцированных групп:

- по целям обучения (группы компенсирующего обучения, творческие, работа с одаренными детьми и т.п.);
- по содержанию обучения (профильные классы, классы с углубленным изучением предмета, спецклассы и т.д.);
- по методам и технологиям (группы развивающего обучения, коллективного способа обучения, работающие по авторским методикам, социоигровые, адаптирующего уровня, специальные и т.д.);
- по темпу (по времени) обучения (группы опережающего обучения, ускоренного и замедленного обучения).

В настоящее время наибольшее распространение получили два вида дифференциации по индивидуально-психологическим особенностям: по уровню умственного развития и по области интересов (профилю). В нашей школе дифференциация организуется при переходе из начальной ступени в среднюю (по уровню умственного развития) и при переходе из средней ступени в старшую (по области интересов).

Информатика (по сравнению с другими учебными предметами) предоставляет большие возможности для реализации дифференцированного обучения. Эти возможности обусловлены: потенциалом информационных технологий, принесенных в учебный процесс именно информатикой; широкими межпредметными связями информатики с другими учебными дисциплинами; значительной прикладной составляющей содержания обучения – средства информационных технологий и методы их использования в различных областях деятельности человека.

Информатика является одним из тех предметов, в которых дифференциация обучения реализуется наиболее естественным путем.

Существенные различия в начальных знаниях, в навыках и умениях работы на компьютере, разный уровень компьютерной грамотности учащихся подталкивает учителя к поиску новых форм организации урока. Одной из таких форм является уровневая дифференциация.

Подбор заданий осуществляется с учетом обязательных результатов обучения, межпредметных связей, уровней усвоения учащимися знаний (базового, среднего и высокого).

На основе полученных теоретических знаний учащиеся могут выбрать задания по нарастающему уровню сложности:

Базовый уровень. Задания этого уровня содержат в себе обязательный уровень обучения. Это репродуктивные упражнения с четким алгоритмом их выполнения.

Средний уровень. Задания требуют обобщения нового материала, заставляют делать выводы, применять свои знания в новых ситуациях. Задания данного уровня – это мини-проекты, рассчитанные на одно занятие. При этом обговариваются цель, план и средства выполнения данного задания.

Высокий уровень. Задания творческого характера и повышенной трудности, требующие сравнения, анализа, проведения исследовательской

деятельности, то есть это проектно-творческая деятельность, рассчитанная на выполнение в течение изучения одной темы. Для их выполнения необходимо:

1. Добывание дополнительной теоретической информации из научной и популярной литературы, электронных учебных курсов и индивидуальных консультаций преподавателя.

2. Большая самостоятельность выполнения заданий по сравнению с другими уровнями, так как ученики сами выполняют данный проект.

По такой нарастающей схеме каждый ученик начинает работу с базового уровня, постепенно поднимаясь до того уровня, который считает для себя необходимым. И от того, как в начале урока учитель поставит проблему, сможет ли заинтересовать учеников, зависит стремление детей выполнить больший объем заданий.

Для отбора учеников в группу определенного уровня можно использовать анкетирование. С помощью анкетирования учитель может выявить интерес учеников к предмету (информатике), уровень знаний по предмету. Применение данного вида анкетирования можно проводить в начале урока, при изучении новой темы. Нужно это для того, чтобы выяснить, в группу какого уровня следует отнести ученика. При этом учитель должен объяснить цель данного анкетирования: в зависимости от способностей и интереса к информатике, ученик имеет возможность решать на уроках посильные для него задания 1-го (базового), 2-го (среднего), или 3-го (высокого) уровня. В этом случае оценка будет зависеть от выбранного уровня, и для получения более высокого балла ученик может попытаться решить задания других уровней. Если ученик почувствует в себе силы решать задачи высокого уровня, или наоборот, не будет справляться с заданиями, он может перейти на другой уровень и решать посильные задания.

В современных условиях важно то, что каждый ученик может



добровольно выбрать для себя уровень усвоения и отчетности в результатах своего учебного труда. При этом школьник несет ответственность за выполнение обязательных требований, что позволяет ему иметь положительную оценку по информатике. В то же время учащийся получает право самостоятельно решать, достаточно ли ему базового уровня образовательных требований или он будет двигаться дальше. Это кардинально меняет традиционные подходы к организации обучения информатике: не следует решать за ученика, какой уровень усвоения соответствует его способностям, но следует создать в классе такие условия, при которых достижение обязательного уровня будет реальным, ученики, способные двигаться дальше, будут заинтересованы в этом продвижении.

Таким образом, можно сказать, что в условиях уровневой дифференциации процесс обучения информатике является наиболее продуктивным. Уровневая дифференциация способствует закреплению знаний, умений и навыков учащихся, активизирует их работу на уроке, повышают работоспособность. Правильно подобранные задания помогают учащимся с разным уровнем знаний раскрыть свои возможности, повышают интерес к предмету.

Следовательно, уровневая дифференциация обучения на уроках информатики способствует повышению уровня знаний, умений и навыков, улучшению успеваемости учеников, учету особенностей и интересов учащихся, снижению нагрузки на них.

## **Глава 2. Методика изучения раздела «Алгоритмизация и программирование» с учетом дифференциации**

### **2.1. Описание раздела «Алгоритмизация и программирование» в программах, реализующих ФГОС ООО**

В Федеральных государственных образовательных стандартах [19] во всех видах образовательных результатов – личностных, метапредметных, предметных – прослеживаются результаты, связанные с обучением по разделу «Алгоритмизация и программирование»: алгоритмизация задач в любых областях деятельности, умение логически мыслить и анализировать имеющиеся данные.

Изучение раздела «Алгоритмизация и программирование» имеет следующие цели:

- сформировать у учащихся алгоритмическую культуру;
- сформировать у учащихся знания об основных понятиях: алгоритм и его свойства;
- сформировать у учащихся знания об алгоритмических конструкциях;
- познакомить учащихся с основными алгоритмическими структурами: линейной, ветвящейся, циклической;
- развить у учащихся алгоритмическое мышление, необходимое для дальнейшей профессиональной деятельности в современном обществе;
- развить у учащихся умения составлять и записывать алгоритм для конкретного исполнителя;
- познакомить учащихся с основами одного из языков программирования.

Согласно образовательной программе и учебному плану на изучение курса информатики в основной школе отводится 105 часов в неделю (с пятого по седьмой класс по одному часу в неделю). При таком варианте на изучение раздела «Алгоритмизация и программирование» отводится 30

часов.

В программе предлагается следующее содержание раздела «Алгоритмизация и программирование»:

1. Базовые понятия (исполнитель, алгоритм, алгоритмический язык, программа).

Дать учащимся следующие понятия:

Понятие исполнителя. Обстановка исполнителя. Возможные состояния исполнителя. Допустимые действия исполнителя, система команд, конечность набора команд. Команды-приказы и команды-запросы. Необходимость формального описания возможных состояний исполнителя и обстановки, в которой он находится, а также действий исполнителя. Примеры исполнителей. Построение моделей реальных процессов как процессов функционирования исполнителей. Понятие алгоритма как формального описания последовательности действий исполнителя при заданных начальных данных. Алгоритмический язык - формальный язык для записи алгоритмов.

Программа - запись алгоритма на алгоритмическом языке. Непосредственное и программное управление исполнителем. Различие: исполнитель выполняет команды; компьютер (человек) выполняет программу. Управление. Сигнал. Обратная связь. Неветвящиеся (линейные) программы.

Предполагается следующая аналитическая деятельность:

- уметь анализировать системы команд и отказов учебных исполнителей (например, Робот, Чертёжник, Черепаха, Удвоитель и др.), арифметических исполнителей; придумывать аналогичные учебные исполнители и задачи по управлению ими;

- уметь анализировать процессы, происходящие в различных системах, как процессы функционирования исполнителей, описывать обстановки этих исполнителей, команды-действия и команды-вопросы;

- уметь составить и записать алгоритм решения для несложных задач, которые решаются исполнителем, управляемым с помощью пульта;

- уметь анализировать работу алгоритмов в зависимости от исходных данных алгоритмов.

В теме базовых понятий учащимся предполагается проявить себя в следующей практической деятельности:

- решать задачи по управлению исполнителем для достижения требуемого результата, командуя учебным исполнителем с помощью пульта;

- строить цепочки команд, дающих нужный результат при конкретных исходных данных для Робота; для вычисления значения конкретного арифметического выражения (исполнителем арифметических действий);

- уметь записать (неформально) план управления учебным исполнителем при решении простейших задач, уметь записать (формально) план управления в какой-либо реальной системе программирования;

- исполнять алгоритм при заданных исходных данных;

- строить линейные программы на выбранном алгоритмическом языке по словесному описанию алгоритма, записывать и выполнять их в выбранной среде программирования.

## 2. Утверждения. Логические значения.

Дать учащимся следующие понятия:

Утверждения (условия). Истинность утверждений. Логические значения, логические операции и логические выражения. Команды исполнителей, проверяющие истинность условий (команды-запросы с ответом в форме «да» или «нет»). Необходимость комбинирования информации, полученной из ответов на несколько запросов. Три правила комбинирования логических значений: операции «и», «или» и «не». Правила записи логических выражений; приоритеты логических операций.

Предполагается следующая аналитическая деятельность:

- научиться анализировать логическую структуру фраз естественного языка.

В теме логических значений учащимся предполагается проявить себя в следующей практической деятельности:

- формально записывать условия нахождения исполнителя в заданном состоянии, например, Робот стоит в закрашенной клетке, из клетки, где стоит Робот, есть более одного выхода, рядом с Роботом нет ни одной стены;

- используя операции сравнения числовых значений, формально записывать на выбранном алгоритмическом языке условия принадлежности точки с заданными координатами простейшим фигурам на плоскости: начало координат; множество из двух точек; первый квадрант; замкнутый луч - биссектриса первого квадранта; полоса, параллельная одной из осей координат, и др.;

- вычислять истинное значение логической формулы, в том числе заданной на каком-нибудь языке программирования.

### 3. Основные конструкции алгоритмических языков.

Дать учащимся следующие понятия:

Алгоритмические конструкции, связанные с проверкой условий: ветвление (условный оператор) и повторение (операторы цикла в форме «пока» и «для каждого»), Понятие вспомогательного алгоритма. Понятие величины (переменной). Типы величин: целые, вещественные, символьные, строковые (литеральные), логические. Знакомство с табличными величинами (массивами).

Предполагается следующая аналитическая деятельность:

- уметь анализировать программы, написанные с применением перечисленных управляющих конструкций;

- уметь анализировать изменение значений величин путём пошагового выполнения программ.

В теме основных конструкций алгоритмических языков учащимся

предполагается проявить себя в следующей практической деятельности:

- создавать и выполнять программы управления исполнителями с применением перечисленных управляющих конструкций;
- вносить добавления и исправления в представленные учителем программы так, чтобы они решали поставленную задачу;
- создавать и выполнять несложные программы с использованием перечисленных типов величин;
- рисовать графики изменения значений числовых величин с помощью графического исполнителя.

#### 4. Решение задач на составление алгоритмов и программ.

На данную тему раздела отводится самое большое количество часов и предлагаются определённые типы задач, с которыми должны справляться учащиеся:

Составление алгоритмов и программ по управлению исполнителями: нахождение минимального и максимального числа из двух, трёх, четырёх данных чисел; нахождение всех корней заданного квадратного уравнения; заполнение числового массива в соответствии с формулой или путём ввода чисел; нахождение суммы элементов данной конечной числовой последовательности или массива; нахождение минимального (максимального) элемента массива.

Знакомство с алгоритмами решения этих задач.

Реализации этих алгоритмов в выбранной среде программирования.

Понятие об этапах разработки программ: составление требований к программе (спецификации), выбор алгоритма и его реализация в виде программы на выбранном алгоритмическом языке, отладка программы с помощью выбранной системы программирования, тестирование.

Простейшие приёмы диалоговой отладки программ (выбор точки останова, пошаговое выполнение, просмотр значений величин, отладочный вывод).

Знакомство с документированием программ.

Составление описания программы по образцу.

Планируется следующая аналитическая деятельность:

- определять зависимость времени работы программы (количества шагов выполнения) от размера исходных данных, например, длины массива.

Практическая деятельность заключается в следующем:

- решать задачи на составление алгоритмов и программ;
- разрабатывать и отлаживать программы в выбранной среде программирования;
- составлять документации программ по образцам.

## **2.2. Методические особенности изучения раздела «Алгоритмизация и программирование»**

Раздел «Алгоритмизация и программирование» является одним из самых сложных при изучении курса информатики.

В нашей школе изучение раздела «Алгоритмизация и программирование» начинается в девятом классе и продолжается практически целый год. Освоив работу с графическими исполнителями (Чертежник, Стрелочка), учащиеся довольно свободно начинают писать программы на языке программирования Паскаль. Паскаль удобен для изучения основ программирования, так как является чисто процедурным языком, позволяющим изучить структурное программирование. Особенности языка являются строгая типизация и наличие средств структурного программирования. По мнению создателя (Н.Вирта), язык должен способствовать дисциплинированию программирования, синтаксис языка Паскаль сведен к минимуму и интуитивно понятен даже при первом знакомстве с языком.

Изучая с учащимися тему «Программирование» невозможно обойтись только теоретическим изучением материала. Основной формой изучения

языка программирования является практика выполнения программ. В процессе самостоятельного составления программы, отладки учащийся получает необходимые навыки. Исполнение или неисполнение программы является эффективной формой контроля полученных знаний. Учащиеся приучаются к аккуратности, внимательности, умению доводить начатое до конца, у них развивается рациональное и логическое мышление.

Учащиеся должны получить возможность работать на компьютере, возможность «потрогать» систему программирования, осваивать приемы работы в ней. В самом начале, уже при освоении простых линейных алгоритмов, вместе с учащимися лучше проверять работу программы ручной трассировкой алгоритма (трассировочные таблицы). Это помогает учащимся лучше понять работу программы, как бы почувствовать процесс исполнения, увидеть свои ошибки, допущенные в алгоритме. Позже, когда учащимися будет приобретен некоторый навык в программировании, можно будет отказаться от ручной трассировки.

Обучение программированию желательно проводить на примерах типовых задач с постепенным усложнением структуры алгоритмов. Например, 1) линейные алгоритмы: вычисления по формулам, обмен ячеек памяти значениями переменных; 2) ветвящиеся алгоритмы: поиск наибольшего/наименьшего значения из нескольких предложенных, сортировка двух/трех значений, построения диалога с ветвлениями; 3) циклические алгоритмы: вычисление сумм/произведений числовых последовательностей, циклический ввод данных с последующей обработкой.

Задачи по теме «Программирование» можно разбить по сложности на несколько типов:

- исполнение программы;
- поиск ошибки в готовой программе;
- определить результат выполнения программы;
- построение математической модели, составление алгоритма, написание



программы, проверка программы.

При организации занятий по разделу «Алгоритмизация и программирование» мною используются различные методы и средства обучения: объяснение/демонстрация нового материала (рассказ, объяснение, лекция, беседа, работа с учениками, наблюдение, презентация); инструктаж (печатная инструкция, устная постановка задачи); практические методы (лабораторные работы, практические работы); метод проектов; проблемное обучение.

По организационной форме предпочтительным является комбинированный урок, который включает в себя объяснение нового материала и первичное закрепление полученных знаний на практике. С одной стороны, достигается наибольший педагогический эффект, с другой – время работы за компьютером минимизируется до регламентированного. В первой части урока учащимся даются новые теоретические знания, во второй части выполняется практическая работа с компьютером.

Обучение информатике и программированию в частности объективно характеризуется большой дифференциацией знаний, способностей, интересов, которые объясняются современным уровнем информатизации общества. Различный уровень сложности упражнений позволяет осуществлять дифференцированный подход к обучению.

Контроль над усвоением полученных знаний, умений и навыков осуществляется на основе выполняемых практических работ, контрольных работ, защиты проекта.

Проект темы "Алгоритмизация и программирование" в 9 классе состоит из 11 уроков.

Урок 1. Понятие алгоритма. Свойства алгоритмов.

Урок 2. Алгоритмы работы с величинами. Компьютер как исполнитель алгоритмов.

Урок 3. Линейные вычислительные алгоритмы. Присваивания, свойства

присваивания.

Урок 4. Знакомство с языком Паскаль. Возникновение и назначение языка Паскаль.

Урок 5. Самостоятельная работа по линейным алгоритмам (Паскаль)

Урок 6. Алгоритмы с ветвящейся структурой (Паскаль): оператор ветвления, программирование полного и неполного ветвления, программирование вложенных ветвлений, логические операции, сложные логические операции.

Урок 7. Самостоятельная работа по ветвящимся алгоритмам (Паскаль)

Урок 8. Программирование циклов (Паскаль): цикл со счетчиком, цикл с предусловием, цикл с постусловием.

Урок 9. Самостоятельная работа по циклическим алгоритмам (Паскаль)

Урок 10. Массивы (Паскаль): что такое массив, описание и ввод значений в массив, обработка массива.

Урок 11. Самостоятельная работа по массивам (Паскаль)

Рассмотрим несколько уроков из разработанной системы.

### **Урок 1. Понятие алгоритма. Свойства алгоритмов.**

#### **Цели урока:**

- познакомить учащихся с понятием «алгоритм»
- способствовать развитию алгоритмического мышления

**Тип урока:** изучение нового материала.

#### **Оборудование урока:**

- компьютер учителя;
- мультимедийный проектор;
- презентации к уроку

#### **План урока:**

1. Организационный момент.
2. Актуализация опорного материала.
3. Сообщение темы и постановка цели урока.

4. Изучение нового материала.
5. Закрепление нового материала, решение задач.
6. Подведение итогов.
7. Постановка домашнего задания.

### **Ход урока.**

#### **1. Организационный момент.**

Проверка присутствующих.

#### **2. Актуализация опорного материала.**

Наводящие вопросы: как понимают слово алгоритм, где могли встречать слово «алгоритм».

#### **3. Сообщение темы и постановка цели урока.**

Сегодня мы начинаем изучение нового раздела «Алгоритмизация. Основы программирования» Тема нашего урока «Понятие алгоритма. Свойства алгоритма» Значит сегодня мы будем говорить о том, что такое «алгоритм», откуда пошло это слово, что оно означает. А также поговорим о том, какими свойствами должен обладать алгоритм.

#### **4. Изучение нового материала.**

Происхождение термина «алгоритм» связано с математикой. В IX веке в Багдаде жил ученый по имени Мухаммед бен Муса ал-Хорезми (Мухаммед, сын Мусы из Хорезма). Он был математиком, астрономом, географом. В одном из своих трудов он описал десятичную систему счисления и впервые сформулировал правила выполнения арифметических действий над целыми числами и обыкновенными дробями. Оригинал книги был утерян, но остался ее перевод на латинский язык, по которому Западная Европа познакомилась с десятичной системой счисления и правилами выполнения арифметических действий.

Аль-Хорезми очень хотел, чтобы сформулированные им правила арифметических действий были понятными для всех. Однако в девятом веке,

когда еще не были придуманы знаки арифметических операций, это было сделать довольно трудно. Однако, ученому удалось выработать четкий стиль строгого словесного описания, который не давал отклониться от предписанного или пропустить какие-либо действия.

Правила эти в латинском переводе начинались со слов «Алгоризми сказал...», что означало «алгоритм гласит...». Со временем было забыто, что Алгоризми – это имя автора правил, и сами правила стали называть алгоритмами.

В настоящее время понятие алгоритма уточнено. Алгоритм – точное и понятное предписание исполнителю совершить последовательность действий, направленных на решение поставленной задачи.

Каждый алгоритм обладает определенными свойствами.

**Дискретность.** (прерывность, раздельность) — алгоритм должен представлять процесс решения задачи как последовательное выполнение простых (или ранее определенных) шагов. Каждое действие, предусмотренное алгоритмом, исполняется только после того, как закончилось исполнение предыдущего.

**Определенность.** Каждое правило алгоритма должно быть четким, однозначным. Благодаря этому свойству выполнение алгоритма носит механический характер и не требует никаких дополнительных указаний или сведений о решаемой задаче.

**Понятность.** Исполнитель алгоритма должен понимать команды, которые он получает.

**Результативность (конечность).** Алгоритм должен приводить к решению задачи за конечное число шагов.

**Массовость.** Алгоритм решения задачи разрабатывается в общем виде, то есть он должен быть применим для некоторого класса задач, различающихся только исходными данными. При этом исходные данные

могут выбираться из некоторой области, которая называется областью применимости алгоритма.

## 5. Закрепление нового материала, решение задач.

Для закрепления понятия «алгоритм» проводится командная игра. Учащиеся разбиваются на две команды. Каждая команда получает задания для выполнения.

Команда 1	Команда 2																				
Ячейки для внесения результатов работы алгоритма	Ячейки для внесения результатов работы алгоритма																				
<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr></table>	1	2	3	4	5						<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td></td><td></td><td></td><td></td><td></td></tr></table>	1	2	3	4	5					
1	2	3	4	5																	
1	2	3	4	5																	
Алгоритм 1 1. Напиши слово ГЕЙМЕР 2. Убери все буквы Е 3. Убери первые две буквы 4. Оставшиеся буквы поменяй местами 5. Результат внеси в ячейку 3	Алгоритм 1 1. Напиши слово РАСШИРЕНИЕ 2. Убери последние две буквы 3. Убери первые пять букв 4. Среднюю букву перенеси в конец 5. Результат внеси в ячейку 3																				
Алгоритм 2 1. Напиши слово ПЛАТА 2. Убери первые три буквы 3. Оставшиеся буквы поменяй местами 4. Результат внеси в ячейку 4	Алгоритм 2 1. Напиши слово НИК 2. Убери первую букву 3. Оставшиеся буквы поменяй местами 4. Результат внеси в ячейку 1																				
Алгоритм 3 1. Напиши слово ЗНАКИ 2. Убери первые три буквы	Алгоритм 3 1. Напиши слово ХАКЕР 2. Убери первую букву																				

3. Последнюю букву переставь вперед 4. Среднюю букву переставь в конец 5. Результат внеси в ячейку 5	3. Убери последние две буквы 4. Оставшиеся буквы поменяй местами 5. Результат внеси в ячейку 5
Алгоритм 4 1. Напиши слово СОФТ 2. Убери первую букву 3. Убери последнюю букву 4. Оставшиеся буквы поменяй местами 5. Результат внеси в ячейку 2	Алгоритм 4 1. Напиши слово ПАЛИТРА 2. Убери все буквы А 3. Убери первые две буквы 4. Убери последнюю букву 5. Оставшиеся буквы поменяй местами 6. Результат внеси в ячейку 4
Алгоритм 5 1. Напиши слово АНТИВИРУС 2. Убери буквы ВИРУС 3. Убери первую букву 4. Убери среднюю букву 5. Оставшиеся буквы поменяй местами 6. Результат внеси в ячейку 1	Алгоритм 5 1. Напиши слово БРАУЗЕР 2. Убери все буквы Р 3. Убери средние три буквы 4. Результат внеси в ячейку 2
<b>ИНФОРМАТИКА</b>	<b>КИБЕРНЕТИКА</b>

Подведем итоги. Что такое алгоритм? Какими свойствами он обладает?

**Урок 2. Алгоритмы работы с величинами. Компьютер как исполнитель алгоритмов.**

**Цели урока:**

- сформировать у учащихся понятий «данные», «величина»;

- познакомить учащихся с представлением структуры и принципа хранения данных в памяти компьютера;

- познакомить учащихся с системой команд исполнителя (компьютера);

- разобрать принципы работы команд присваивания, ввода и вывода.

**Тип урока:** изучение нового материала.

**Оборудование урока:**

- компьютер учителя;

- мультимедийный проектор;

- презентации к уроку

- карточки с заданиями

**План урока:**

1. Организационный момент.
2. Повторение пройденного материала.
3. Сообщение темы и постановка цели урока.
4. Изучение нового материала.
5. Закрепление нового материала, решение задач.
6. Подведение итогов.
7. Постановка домашнего задания.

**Ход урока.**

**1. Организационный момент.**

Проверка присутствующих. Раздача карточек с заданиями.

**2. Повторение пройденного материала.**

Фронтальный опрос по теме «Алгоритмизация»: Алгоритм, свойства алгоритмов, базовые структуры алгоритмов.

**3. Сообщение темы и постановка цели урока.**

Сегодня мы продолжим изучение раздела «Алгоритмизация. Основы программирования» Тема нашего урока «Алгоритмы работы с величинами. Компьютер как исполнитель алгоритмов». Значит сегодня мы будем говорить

об алгоритмах, в качестве исполнителя рассматривая компьютер, оснащенный системой программирования на определенном языке. Мы должны разобрать понятие «данные» и «величина», систему команд исполнителя (компьютера).

#### 4. Изучение нового материала.

Вам уже известно, что всякий алгоритм составляется для конкретного исполнителя. Теперь в качестве исполнителя мы будем рассматривать компьютер, оснащенный системой программирования на определенном языке. Компьютер-исполнитель работает с определенными данными по определенной программе. Информация, обрабатываемая программой, называется данными.

Компьютер работает с информацией, хранящейся в его памяти. Отдельный информационный объект – число, символ, строка – называется величиной.

Всякая обрабатываемая программой величина, занимает свое место в памяти ЭВМ (ячейке). Значение величины – это информация, хранимая в ячейке памяти.

#### Основные типы величин (Рисунок 1):

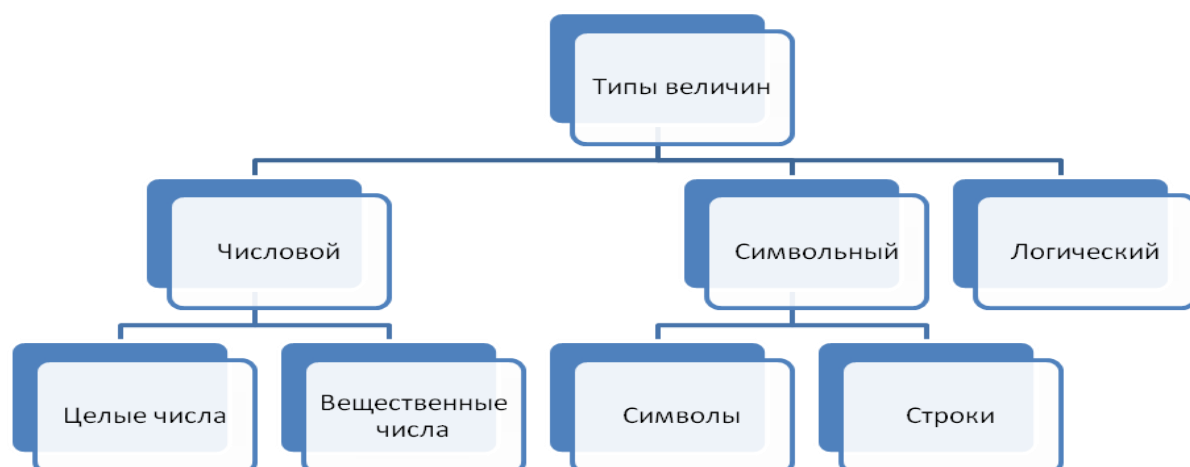


Рисунок 1. Основные типы величин



Числовые величины в программировании делятся на переменные и константы (постоянные).

**Константы** записываются в алгоритмах своими десятичными значениями, например, 23, 3.5, 34. Значение константы хранится в выделенной под нее ячейке памяти и остается неизменным в течение работы программы.

**Переменные** в программировании, как и в математике, обозначаются символическими именами. Эти имена называют **идентификаторами** (от глагола "идентифицировать", что значит "обозначать", "символизировать"). Идентификатор может быть одной буквой, множеством букв, сочетанием букв и цифр и т. д. Примеры идентификаторов: A, X, B3, prim, r25 и т. п.

### **Система команд**

Вам уже известно, что всякий алгоритм строится исходя из системы команд исполнителя, для которого он предназначен. Независимо от того, на каком языке программирования написана программа, алгоритм работы с величинами составляется из следующих команд:

- присваивание;
- ввод;
- вывод;
- цикл;
- ветвление;
- обращение к вспомогательному алгоритму.

### **Команда присваивания.**

Команда присваивания – одна из основных команд в алгоритмах работы с величинами. Записывается она следующим образом:

<переменная> := <выражение>

Пример: A := B+C

Знак  $:=$  читается как «присвоить». Компьютер сначала вычисляет выражение, затем полученный результат присваивает переменной, стоящей слева от знака  $:=$

Значение переменных до выполнения команды

A	B	C
–	5	2

Значение переменных после выполнения команды

A	B	C
7	5	2

### **Команда ввода.**

Значения переменных, являющихся исходными данными решаемой задачи, как правило, задаются вводом.

Команда ввода в описаниях алгоритмов будет выглядеть так:

ввод <список переменных>.

Например:

ввод A, B, C

На современных компьютерах ввод чаще всего выполняется в режиме диалога с пользователем. По команде ввода компьютер прерывает выполнение программы и ждет действий пользователя. Пользователь должен набрать на клавиатуре вводимые значения переменных и нажать клавишу <ВВОД>. Введенные значения присвоятся соответствующим переменным из списка ввода, и выполнение программы продолжится.

Вот схема выполнения приведенной выше команды.

1. Память до выполнения команды:

A	B	C
–	–	–

2. Процессор компьютера получил команду ввод A, B, C, прервал свою работу и ждет действий пользователя.

3. Пользователь набирает на клавиатуре:

1     3     5

и нажимает клавишу <ВВОД> (<Enter>).

4. Память после выполнения команды:

A	B	C
1	3	5

5. Процессор переходит к выполнению следующей команды программы.

При выполнении пункта 3 вводимые числа должны быть отделены друг от друга какими-нибудь разделителями. Обычно это пробелы.

Из сказанного выше можно сделать вывод:

Переменные величины получают конкретные значения в результате выполнения команды присваивания или команды ввода.

Если переменной величине не присвоено никакого значения (или не введено), то она является неопределенной. Иначе говоря, ничего нельзя сказать, какое значение имеет эта переменная.

### **Команда вывода**

Результаты решения задачи сообщаются компьютером пользователю путем выполнения команды вывода.

**Команда вывода** в алгоритмах будет записываться так:

вывод <список вывода>

Например:

вывод X1, X2

По этой команде значения переменных X1 и X2 будут вынесены на устройство вывода (чаще всего это экран).

О других командах, применяемых в вычислительных алгоритмах, вы узнаете позже.

## 5. Закрепление нового материала, решение задач.

Задания первой группы (задания для всех, выполняются устно)

Ответьте на следующие вопросы:

- что такое величина?
- что такое данные?
- какие существуют типы величин в программировании?
- как записывается команда присваивания? Команда ввода? Команда вывода?

Задания первой группы (задания для всех, выполняются письменно)

В схематическом виде (внесение значения в таблицу) отразите изменения значений в ячейках, соответствующих переменным А, В и С, в ходе последовательного выполнения команд присваивания.

№1

Команда	Значение А	Значение В
A:=1		
B:=2		
A:=A+B		
B:= 2xA		

№2

Команда	Значение А	Значение В	Значение С
A:=1			
B:=2			
C:=A			
A:=B			

№3

Команда	Значение А	Значение В
A:=1		
B:=2		

$A := A + B$		
$B := A - B$		
$A := A - B$		

Задания второй группы (усложненные задания, выполняются письменно учащимися, выполнившими предыдущее задание.)

№1. Вместо многоточия впишите в алгоритм несколько команд присваивания, в результате чего должен получиться алгоритм возведения в 4-ю степень введенного числа (дополнительные переменные, кроме A, не использовать):

ввод A . . . вывод A

№2. Даны две переменные - A и B. Например, они равны  $A=4$ ;  $B=7$ . Необходимо поменять местами эти числа (то есть  $A=7$ ,  $B=4$ ). Но не используя третью переменную.

**Урок 3. Линейные вычислительные алгоритмы. Присваивания, свойства присваивания.**

**Цели урока:**

- сформировать представление о линейных вычислительных алгоритмах;
- совершенствовать умения и навыки в использовании присваивания;
- развивать алгоритмическое мышление.

**Тип урока:** изучение нового материала.

**Оборудование урока:**

- компьютер учителя;
- мультимедийный проектор;
- презентации к уроку
- карточки с заданиями

**План урока:**

1. Организационный момент.
2. Повторение пройденного материала.
3. Сообщение темы и постановка цели урока.
4. Изучение нового материала.
5. Закрепление нового материала, решение задач.
6. Подведение итогов.
7. Постановка домашнего задания.

### **Ход урока.**

#### **1. Организационный момент.**

Проверка присутствующих. Раздача карточек с заданиями.

#### **2. Повторение пройденного материала.**

Фронтальный опрос: Алгоритм, величина, данные, переменные, константы, типы величин, команда присваивания.

#### **3. Сообщение темы и постановка цели урока.**

Сегодня мы продолжим изучение раздела «Алгоритмизация. Основы программирования». Тема нашего урока «Линейные вычислительные алгоритмы. Присваивания, свойства присваивания». Сегодня мы будем говорить о линейных алгоритмах, более подробно поговорим о команде присваивания.

#### **4. Изучение нового материала.**

Так как присваивание является одной из важнейших операций в программировании, поговорим о нем более подробно.

Переменная величина получает значение в результате присваивания. Присваивание производится компьютером при выполнении одной из двух команд: команды присваивания или команды ввода.

Рассмотрим выполнение четырех команд присваивания, в которых участвуют две переменные А и В. Результаты будем заносить в трассировочную таблицу (так называется таблица, где против каждой

команды указываются значения переменных, которые устанавливаются после ее выполнения), то есть будем выполнять трассировку алгоритма.

Команда	A	B
A:= 1	1	-
B:= 2 x A	1	2
A:= B	2	2
B:= A + B	2	4

Этот пример иллюстрирует три основных свойства присваивания:

- пока переменной не присвоено значение, она остается неопределенной
- значение, присвоенное переменной, сохраняется вплоть до выполнения следующего присваивания переменной нового значения
- новое значение, присвоенное переменной, заменяет ее предыдущее значение, старое значение переменной нигде не сохраняется.

Попробуем решить одну задачу. С подобным алгоритмом нам придется часто встречаться при программировании.

Даны две переменные A и B. Требуется произвести между ними обмен значениями. Если изначально A=6, B=4, то после обмена значениями должно получиться A=4, B=6.

Для совершения подобного обмена нам понадобится третья переменная C (запасной кармашек). Последовательность действий будет следующей:

- переложить значение A в C
- переложить значение B в A
- переложить значение C в B

Мы достигли цели – переменные обменялись значениями.

Составим трассировочную таблицу для проверки:

Команда	A	B	C
ввод A, B	6	4	-
C:=A	6	4	6
A:=B	4	4	6
B:=C	4	6	6
вывод A, B	<b>4</b>	<b>6</b>	6

Рассмотрим пример составления линейного алгоритма для решения следующей математической задачи: даны две простые дроби; получить дробь, являющуюся результатом их деления.

В школьном учебнике математики правила деления обыкновенных дробей описаны так:

1. Числитель первой дроби умножить на знаменатель второй.
2. Знаменатель первой дроби умножить на числитель второй.
3. Записать дробь, числителем которой является результат выполнения пункта 1, а знаменателем - результат выполнения пункта 2.

В алгебраической форме это выглядит следующим образом:

$$\frac{A}{B} : \frac{C}{D} = \frac{A D}{B C}$$

Теперь построим алгоритм деления дробей для компьютера. В этом алгоритме сохраним те же обозначения для переменных, которые использованы в записанной выше формуле. Исходными данными являются целочисленные переменные  $a, b, c, d$ . Результатом - также целые величины  $m$  и  $n$ .

### 5. Закрепление нового материала, решение задач.

Задания первой группы (задания для всех, выполняются устно)

Ответьте на следующие вопросы:

- что такое алгоритм? Виды алгоритма?
- что такое линейный алгоритм?



- что такое трассировка?
- что такое неопределенная переменная?
- что происходит с предыдущим значением переменной после присваивания ей нового значения?

Задания первой группы (задания для всех, выполняются письменно)

№1. Составьте алгоритм сложения двух простых дробей (без сокращения дроби).

№2. Составьте алгоритм нахождения периметра прямоугольного треугольника по двум известным катетам.

Задания второй группы (усложненные задания, выполняются письменно учащимися, выполнившими предыдущее задание.)

№1. Из одного населенного пункта одновременно выехали два автомобиля с разной скоростью (скорость первого автомобиля больше скорости второго). Составьте алгоритм для нахождения расстояния между ними через 1 час, через 2 часа, через 6 часов.

#### **Урок 4. Знакомство с языком Паскаль. Возникновение и назначение языка Паскаль.**

##### **Цели урока:**

- сформировать у учащихся первичные знания о возникновении и назначении языка Паскаль, структуре программы, синтаксисе программы, правилах записи арифметических выражений
- способствовать развитию у учащихся умений самостоятельно применять на практике знания, умения и навыки для решения поставленных задач при работе с языком программирования Паскаль

**Тип урока:** изучение нового материала.

##### **Оборудование урока:**

- компьютер учителя;
- мультимедийный проектор;

- презентации к уроку
- карточки с заданиями

### **План урока:**

1. Организационный момент.
2. Повторение пройденного материала.
3. Сообщение темы и постановка цели урока.
4. Изучение нового материала.
5. Закрепление нового материала, решение задач.
6. Подведение итогов.
7. Постановка домашнего задания.

### **Ход урока.**

#### **1. Организационный момент.**

Проверка присутствующих. Раздача карточек с заданиями.

#### **2. Повторение пройденного материала.**

Фронтальный опрос: Алгоритм, присваивание, ввод, вывод, типы переменных.

#### **3. Сообщение темы и постановка цели урока.**

Сегодня мы продолжим изучение раздела «Алгоритмизация. Основы программирования». Тема нашего урока «Знакомство с языком Паскаль. Возникновение и назначение языка Паскаль». Сегодня мы будем говорить о языке программирования Паскаль, структуре программы, программном синтаксисе, правилах записи арифметических выражений.

#### **4. Изучение нового материала.**

После того, как построен алгоритм решения задачи, составляется программа на определенном языке программирования.

Среди современных языков программирования одним из самых популярных является язык Паскаль. Этот язык разработан в 1971 году и назван в честь Блеза Паскаля - французского ученого, изобретателя механической вычислительной машины. Автор языка Паскаль - швейцарский

учёный, специалист в области информатики, один из известнейших теоретиков в области разработки языков программирования Никлаус Вирт.

**Паскаль** - это универсальный язык программирования, позволяющий решать самые разнообразные задачи обработки информации.

Команду алгоритма, записанную на языке программирования, принято называть **оператором**.

### **Структура программы на Паскале**

**Program** Имя программы; *(не обязателен)*

**Uses** Подключаемые библиотеки (модули); *(не обязателен)*

**Label** Список меток основной программы; *(не обязателен)*

**Const** Введение констант; *(не обязателен)*

**Type** Описание новых типов; *(не обязателен)*

**Var** Описание переменных и их типов;

Определение процедур;

Определение функций;

**Begin**

Тело основной программы;

**End.**

Раздел описания переменных начинается со слова Var (variables - переменные), за которым идет список имен переменных через запятую. Тип указывается после двоеточия. Например, раздел описания переменных может быть таким:

**var** a, b : integer; c, d : real;

Идентификаторы переменных состояются из латинских букв и цифр; первым символом обязательно должна быть буква.

Основные типы переменных в языке программирования Паскаль (рисунок 1):

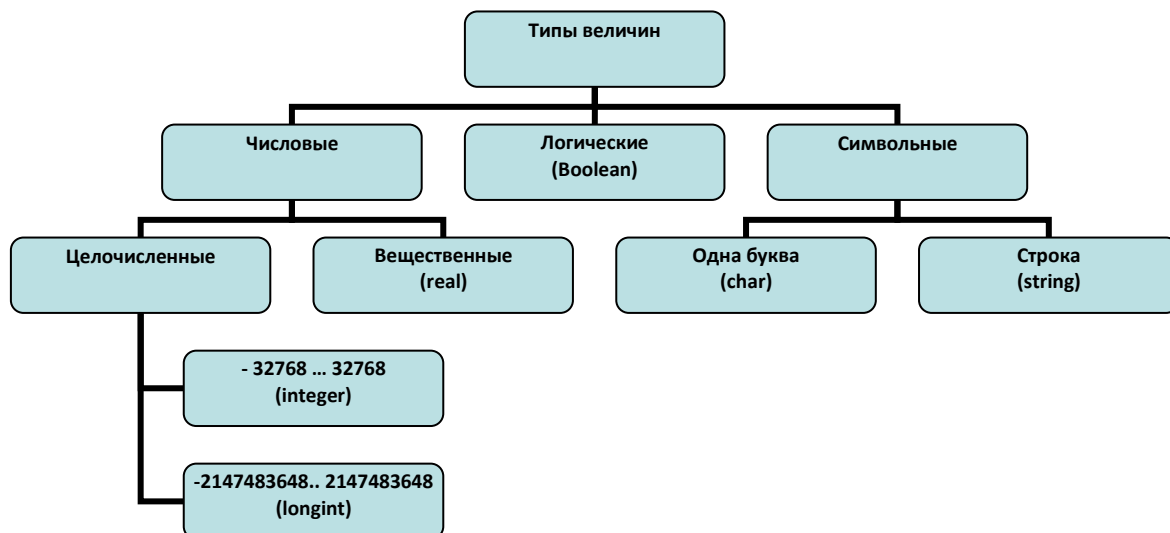


Рисунок 1. Основные типы переменных в языке программирования  
Паскаль

Раздел операторов - основная часть программы. Начало и конец раздела операторов программы отмечаются служебными словами **begin** (начало) и **end** (конец). В самом конце программы ставится точка:

**begin**

< операторы >

**end.**

### Операторы ввода, вывода, присваивания

Ввод исходных данных с клавиатуры происходит по оператору **read** (read - читать) или **readln** (read line - читать строку):

read(<список переменных>);

или

readln(<список переменных>);

При выполнении команды ввода компьютер ожидает действий пользователя. Пользователь набирает на клавиатуре значения переменных в том порядке, в каком они указаны в списке, отделяя их друг от друга пробелами. Одновременно с набором данных на клавиатуре они появляются на экране. В конце нажимается клавиша <ВВОД> (<Enter>). Разница в

выполнении операторов `readln` и `read` состоит в том, что после выполнения ввода по оператору `readln` экранный курсор перемещается в начало новой строки, а по оператору `read` этого не происходит.

Вывод результатов происходит по оператору **write** (`write` - писать) или **writeln** (`write line` - писать в строку):

```
write(<список вывода>);
```

или

```
writeln(<список вывода>);
```

Результаты выводятся на экран компьютера в порядке их перечисления в списке. Элементами списка вывода могут быть константы, переменные, выражения.

Разница в выполнении операторов `writeln` и `write` состоит в том, что после выполнения вывода по оператору `writeln` экранный курсор перемещается в начало новой строки, а по оператору `write` этого не происходит.

Арифметический оператор присваивания на Паскале имеет следующий формат:

**<числовая переменная> := <арифметическое выражение>**

Арифметическое выражение может содержать числовые константы и переменные, знаки арифметических операций, круглые скобки. Кроме того, в арифметических выражениях могут присутствовать функции.

Знаки основных арифметических операций записываются так:

+ сложение,

- вычитание,

\* умножение,

/ деление.

Кроме них используется оператор `div` – оператор целочисленного деления, оператор `mod` – вычисление остатка от целочисленного деления.

```
a:= 21;
```

b:=8;

s:=21 mod 8;

Переменная s будет равна 5.

a:= 21;

b:=8;

s:=21 div 8;

Переменная s будет равна 2.

### **Правила записи арифметических выражений**

Запись арифметических выражений на Паскале похожа на обычную математическую запись. В отличие от математики, где часто пропускается знак умножения (например, пишут  $2A$ ), в Паскале этот знак пишется обязательно:  $2*A$ . Например, математическое выражение

$$A^2 + B^2 - 12C$$

на Паскале записывается так:

$$A*A + B*B - 12*C$$

Это же выражение можно записать иначе:

$$SQR(A) + SQR(B) - 12*C$$

Здесь использована функция возведения в квадрат - SQR. Аргументы функций всегда пишутся в круглых скобках.

Последовательность выполнения операций определяется по их **приоритетам** (старшинству). К старшим операциям относятся умножение (\*) и деление (/). Операции сложения и вычитания - младшие. В первую очередь выполняются старшие операции. Несколько операций одинакового старшинства, записанные подряд, выполняются в порядке их записи слева направо. Приведенное выше арифметическое выражение будет вычисляться в следующем порядке (порядок вычислений указан цифрами сверху):

1      4      2      5      3  
**A\*A    +    B\*B    -    12 \* C**

Круглые скобки в арифметических выражениях влияют на порядок выполнения операций. Как и в математике, в первую очередь выполняются операции в скобках. Если имеются несколько пар вложенных скобок, то сначала выполняются операции в самых внутренних скобках. Например,

6          1          3          2          4          5  
**A   +   (( C- D)   /   (2 + K)   - J ) \* B**

### **Пунктуация Паскаля**

Необходимо строгое соблюдение правописания (синтаксиса) программы. В частности, в Паскале однозначно определено назначение знаков пунктуации.

**Точка с запятой** (;) ставится в конце заголовка программы, в конце раздела описания переменных, является разделителем операторов. Перед словом end точку с запятой можно не ставить.

**Запятая** (,) является разделителем элементов во всевозможных списках: списке переменных в разделе описания, списке вводимых и выводимых величин.

Строгий синтаксис в языке программирования необходим потому, что компьютер является формальным исполнителем программы. Если, допустим, разделителем в списке переменных должна быть запятая, то любой другой знак будет восприниматься как ошибка. Если точка с запятой является разделителем операторов, то в качестве оператора компьютер воспринимает всю часть текста программы от одной точки с запятой до другой. Если программист забыл поставить ";" между какими-то двумя операторами, то компьютер будет принимать их за один с неизбежной ошибкой.

В программу на Паскале можно вставлять комментарии. Комментарий - это пояснение к программе, которое записывается в фигурных скобках. В

комментариях можно использовать русские буквы. На исполнение программы комментариев никак не влияет.

Заметим, что в Паскале нет различия между строчными и прописными буквами. Например, для Паскаля тождественны следующие варианты записи: begin, Begin, BEGIN, BeGiN. Использование строчных или прописных букв - дело вкуса программиста.

### **5. Закрепление нового материала, решение задач.**

Задания первой группы (задания для всех, выполняются устно)

Ответьте на следующие вопросы:

- Когда появился язык Паскаль и кто его автор?
- Как записывается заголовок программы на Паскале?
- Как записывается раздел описания переменных?
- С какими типами числовых величин работает Паскаль?
- Как записываются операторы ввода и вывода в Паскале?
- Что такое оператор присваивания?
- Как записываются арифметические выражения?
- По каким правилам определяется порядок выполнения операций в арифметическом выражении?

Учащиеся разделяются на группы.

Задания первой группы (задания для всех, выполняются письменно)

№1. Какая задача решается по следующей программе?

```
Var a,b,c:integer;
```

```
begin
```

```
  readln(a);
```

```
  readln(b);
```

```
  c := a + b;
```

```
  writeln(c);
```

```
end.
```



№2. Какой результат будет получен, если в качестве исходных значений А и В ввести соответственно 10 и 5? 15 и 6? Выполнить трассировку (в виде таблицы)

№3. Какая задача решается по следующей программе?

```
var A, B, C: integer;  
begin  
  readln(A,B);  
  C:=(A+B)*(B-A);  
  writeln(C)  
end.
```

Какой результат будет получен, если в качестве исходных значений А и В ввести соответственно 7 и 8? Выполнить трассировку (в виде таблицы)

Задания второй группы (усложненные задания, выполняются письменно учащимися, выполнившими предыдущие задания.)

№1. Какая задача решается по следующей программе?

```
const pi=3.14;  
var r:real;  
begin  
  read(r);  
  r:=pi*r*r;  
  writeln(r);  
end.
```

№2. Составить программу для решения следующей задачи: Даны два круга с общим центром и радиусами R1 и R2 ( $R1 > R2$ ). Найти площади этих

кругов  $S_1$  и  $S_2$ , а также площадь  $S_3$  кольца, внешний радиус которого равен  $R_1$ , а внутренний радиус равен  $R_2$

## **Урок 5. Самостоятельная работа по линейным алгоритмам (Паскаль)**

### **Цели урока:**

- закрепление полученных знаний
- продолжить формировать у учащихся навыков работы с языком программирования Паскаль
- проверка усвоенных знаний

**Тип урока:** проверка и оценка знаний

### **Оборудование урока:**

- карточки с заданиями

### **План урока:**

1. Организационный момент
2. Выполнение работы (по группам)

### **Ход урока**

#### **1. Организационный момент**

Проверка присутствующих. Распределение учащихся по группам (4 группы по три-четыре человека)

#### **2. Выполнение работы.**

Задание 1 (выполняется всеми группами, составляется трассировочная таблица)

Группа 1	Группа 2	Группа 3	Группа 4
Определите значение перемен-	Определите значение перемен-	Определите значение перемен-	Определите значение перемен-

ной b после выполнения алгоритма:	ной b после выполнения алгоритма:	ной b после выполнения алгоритма:	ной b после выполнения алгоритма:
<b>a := 2</b>	<b>a := 5</b>	<b>a := 4</b>	<b>a := 6</b>
<b>b := 4</b>	<b>b := 4</b>	<b>b := 4</b>	<b>b := 4</b>
<b>a := 2*a + 3*b</b>	<b>a := 2*a + 3*b</b>	<b>a := 2*a + 3*b</b>	<b>a := 2*a + 3*b</b>
<b>b := a/2*b</b>	<b>b := a/2*b</b>	<b>b := a/2*b</b>	<b>b := a/2*b</b>

Задание 2 (выполняется всеми группами)

Группа 1	Группа 2	Группа 3	Группа 4
Составить программу для решения задачи:	Составить программу для решения задачи:	Составить программу для решения задачи:	Составить программу для решения задачи:
Дана сторона квадрата, найти его периметр.	Дана сторона квадрата, найти его площадь.	Даны стороны прямоугольника, найти его периметр.	Даны стороны прямоугольника, найти его площадь.

Задание 3 (выполняется всеми группами)

Группа 1	Группа 2	Группа 3	Группа 4
Составить программу для решения задачи:	Составить программу для решения задачи:	Составить программу для решения задачи:	Составить программу для решения задачи:
Даны два ненулевых числа.	Даны два ненулевых числа.	Даны два ненулевых числа.	Даны два ненулевых числа.

Найти сумму их квадратов.	Найти разность их квадратов.	Найти произведение их квадратов.	Найти частное их квадратов.
---------------------------	------------------------------	----------------------------------	-----------------------------

Задание 4 (выполняется по возможности)

Группа 1	Группа 2	Группа 3	Группа 4
Составить программу для решения задачи:  Даны два ненулевых числа. Найти сумму их квадратов.	Составить программу для решения задачи:  Даны два ненулевых числа. Найти разность их квадратов.	Составить программу для решения задачи:  Даны два ненулевых числа. Найти произведение их квадратов.	Составить программу для решения задачи:  Даны два ненулевых числа. Найти частное их квадратов.

Задание 5 (повышенной сложности, выполняется по возможности)

Группа 1	Группа 2	Группа 3	Группа 4
Даны координаты двух противоположных вершин прямоугольника: $(x_1, y_1)$ , $(x_2, y_2)$ . Стороны прямоугольника параллельны осям координат. Найти периметр и площадь данного прямоугольника.			

### 2.3. Определение эффективности методики дифференцированного обучения информатике (на примере темы "Алгоритмизация и программирование" в 9 классе)

Педагогический эксперимент – это специальная организация педагогической деятельности учителей и учащихся с целью проверки и обоснования заранее разработанных теоретических предположений или

гипотез.

Цели эксперимента:

- изучить рассматриваемую проблему на практике изучения раздела «Алгоритмизация и программирование»;
- оценить уровень обученности школьников при использовании разработанной методики;
- скорректировать разработанную методику обучения разделу «Алгоритмизация и программирование».

Достижение целей предполагалось при решении следующих задач:

- выбор методики проведения эксперимента;
- проведение эксперимента в соответствии с заданной методикой;
- проведение качественной обработки результатов эксперимента;
- качественная интерпретация результатов эксперимента.

Использовались следующие методы:

- анализ программ, учебных и методических пособий по преподаванию школьной информатики;
- анализ передового педагогического опыта преподавания информатики;
- организация методического сопровождения темы;
- наблюдение за деятельностью учащихся в процессе изучения раздела «Алгоритмизация и программирование», анализ результатов этой деятельности;
- тестирование, анкетирование, беседы с учащимися;
- статистическая обработка результатов эксперимента и их анализ.

Гипотеза исследования: при применении разработанной методики повышается результативность обучения. В эксперименте сравнивались результаты обученности в двух разных классах.

На этапе проведения констатирующего эксперимента ставилась задача определения уровня знаний учащихся в области пользовательских навыков, интересов учащихся в области информатики. Были получены результаты,

позволяющие сделать следующие выводы:

- большинство учащихся имеют хорошие пользовательские навыки работы с компьютером;
- большая часть учащихся имеет представление о программировании в целом, некоторые (меньшая часть) имеют не очень четкие представления о программировании;
- многие учащиеся проявляют повышенный интерес работе в системах программирования, что практически снимает проблему мотивации изучения темы.

На формирующем этапе проводился эксперимент, в ходе которого определялась эффективность разработанной методики изучения раздела «Алгоритмизация и программирование» и объективной оценке знаний учащихся. Для контроля полученных знаний проводились проверочные работы, выполнялись тестовые задания, проводились устные опросы (беседы).

Обученность включает имеющийся/усвоенный к настоящему моменту запас знаний, сложившиеся способы и приемы их получения (умение учиться). В педагогике выделяются пять уровней обученности:

- различение – характеризуется тем, что ученик может отличить один объект от другого по наиболее существенным признакам;
- запоминание – характеризуется тем, что ученик может пересказать содержание текста, правила, положения теоретического утверждения;
- понимание – характеризуется тем, что ученик может устанавливать причинно-следственные связи явлений, событий, фактов, свободно выводить причину и следствие;
- уровень репродуктивных умений – характеризуется тем, что ученик владеет закрепленными способами применения знаний на практике;
- перенос – характеризуется способностью применить/использовать полученные знания, умения в нестандартных ситуациях.

Работа с обучающимися по проверке уровня обученности строилась по-разному: выполнение практических работ, контрольные срезы, тестирование.

Для расчета уровня обученности использовалась формула академика В.П.Симонова:

Уровень обученности = (количество пятерок + количество четверок\*0,64 + количество троек\*0,36 + количество двоек\*0,08)/общее количество учащихся

При помощи компьютерной программы MS Excel был произведен расчет среднего уровня обученности в экспериментальной и контрольной группах.

Результаты проведенного эксперимента (см. диаграмму 1) показали, что в экспериментальной группе по сравнению с контрольной группой выше уровень знания синтаксиса языка, уровень сформированности умения описать алгоритм создаваемой программы, выше уровень знания принципов разработки программ.

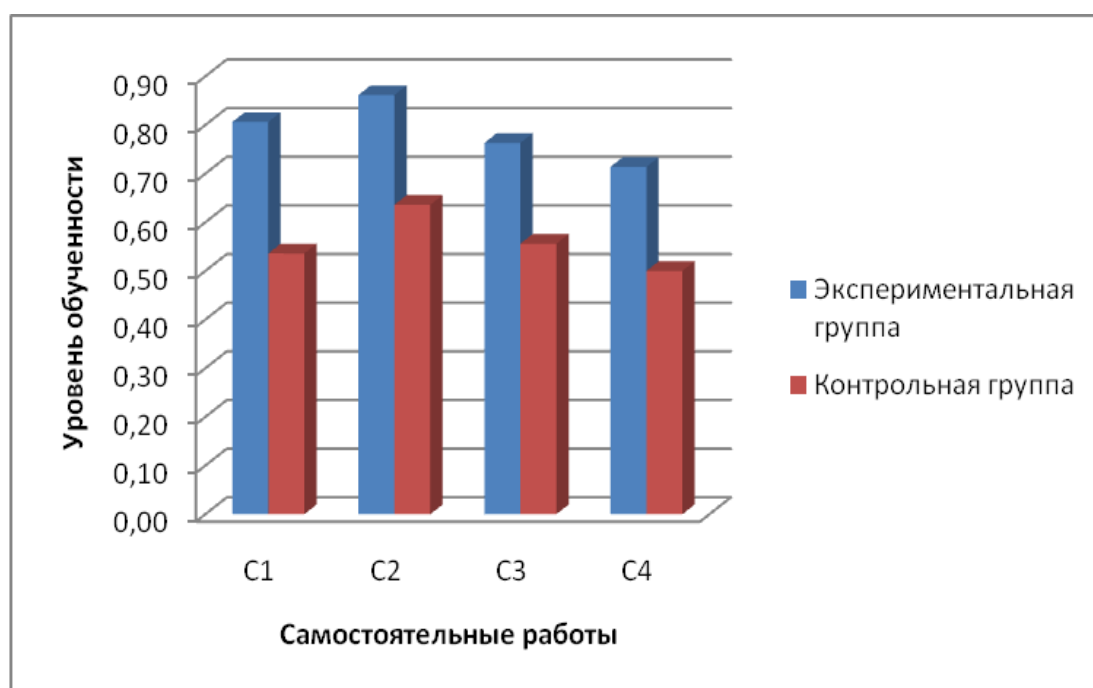


Диаграмма 1. Соотношение уровня обученности в экспериментальной и контрольной группах.

В результате проведенного эксперимента было показано, что применение разработанной методики изучения основ алгоритмизации и программирования способствует повышению уровня знаний и повышает интерес к изучению программирования.



## **Заключение**

По результатам анализа проведения в 2016 году Государственной итоговой аттестации по информатике и ИКТ на территории Московской области в форме ОГЭ выявлена проблема заметного снижения успешности выполнения заданий, связанных с разделом «Алгоритмизация и программирование» (задания 6, 8 -10, 14, 16). Крайне низка результативность выполнения заданий 20.1 и 20.2. Каждый третий участник экзамена не приступил к выполнению задания или неправильно написал программу для формального исполнителя или программу по обработке числовой последовательности на алгоритмическом языке. Основная причина подобного результата, по мнению В.И.Филиппова [17], заключается в том, что изучению данных разделов уделяется недостаточное внимание в курсе информатики основной школы.

Изучение основ алгоритмизации и программирования в школьном курсе информатики должно приводить к усвоению учащимися фундаментальных понятий современной информатики и к получению практических навыков работы с программными средствами. Обучение основам программирования должно проводиться на качественном уровне и должно быть связано с дальнейшей профессиональной деятельностью.

Одним из возможных путей решения вышеуказанной проблемы может стать реализация дифференцированного многоуровневого подхода к организации учебной деятельности обучающихся на основе учета индивидуальных особенностей учащихся и специально построенной системы заданий. Однако мой опыт преподавания информатики в основной школе показывает, что для достижения за оптимальное время установленных общеобразовательным стандартом требований к уровню подготовки школьника в области алгоритмизации и программирования необходимо в полном объеме задействовать весь методический арсенал.

В процессе выполнения работы была достигнута цель – разработана система уроков с использованием дифференцированного подхода в изучении алгоритмизации и программирования, в состав которых входят задания разного уровня сложности, позволяющие учащимся с различными личными данными и возможностями понять и лучше усвоить поставленные перед ними задачи. Особая роль при разработке уроков отводилась подбору учебных задач, учитывалась необходимость делать акцент на их занимательность, ведь среди основных задач любого учителя – поддержание интереса обучающихся к изучаемому предмету, и одним из способов достижения этого является решение занимательных по содержанию и форме задач. Педагогически оправданная занимательность имеет целью привлечь внимание к заданиям, активизировать мыслительную деятельность учащихся, побудить их к работе с литературой.

В ходе подготовки к аттестационной работе, во время стажировки, при разработке системы уроков с использованием дифференцированного многоуровневого подхода в изучении алгоритмизации и программирования я придерживался двух основных принципов – принципа развития и принципа индивидуализации. Принцип развития обучения программированию реализовывался мною в постоянном расширении и обновлении системы задач, решаемых с помощью программирования, и средств их достижения, под которыми я понимаю дальнейшее изучение алгоритмических конструкций языка программирования, приемов и методов решения задач.

Накопив большой опыт, я сделал для себя вывод: мышление всегда предметно, т.е. осуществляется на определенном конкретном материале. На основе анализа большого количества задачников по алгоритмизации и программированию, учебной литературы, книг, обучающих конкретным алгоритмическим языкам, а также различных структур данных и алгоритмов их обработки мною были выделены типы задач и предложены стандартные

алгоритмы их решения, разработаны методы решения обобщенных и аналогичных задач на основе типовых.

Практика показала, что обычно состав класса и даже подгруппы очень неоднороден по уровню знаний учащихся. Те ученики, которые имеют компьютеры дома или занимаются в кружках по информатике или робототехнике, намного быстрее усваивают учебный материал, иногда их знания по программированию намного больше учебной программы. Таким учащимся либо надо давать задания индивидуально, либо готовить более сложные задания на основе задач, решаемых в классе. С другой стороны, тем ученикам, которые испытывают трудности в изучении программирования, желательно давать задания, минимально отличающиеся от решенных в классе.

Опираясь на анализ имеющегося опыта, результаты выполненного исследования, можно сказать, что использование системы задач, включающие задания по работе с готовыми алгоритмами и программами, различные способы представления алгоритмов, различные виды алгоритмов, дает положительные диагностируемые результаты.

Данное обстоятельство может служить хорошей основой для дальнейшего формирования ИКТ-компетенции обучающихся, значимость которой неоспорима в процессе жизнедеятельности в условиях современного информационного общества.

### **Список источников и литературы**

1. Бабанский Ю.К. Избранные педагогические труды / Ю.К. Бабанский. - М.: Педагогика, 1989. - 560 с.
2. Бабанский Ю.К. Оптимизация процесса обучения (Общедидактический аспект) / Ю.К. Бабанский - М.: Педагогика, 1977.
3. Бородин М.Н. Информатика. УМК для основной школы: 5 – 6, 7 – 9 классы (ФГОС). Методическое пособие для учителя. Бинوم, 2013.
4. Воронцова Л.А. Из опыта обучения алгоритмизации и программированию в основной школе. // Информатика в школе. – 2012. - № 9. – С. 44 – 48.
5. Кирсанов А.А. Индивидуализация учебной деятельности как педагогическая проблема / А.А. Кирсанов-Казань, 1982.
6. Кузнецов А. А., Захарова Т. Б. Принципы дифференциации содержания обучения информатике [Текст] // Информатика и образование. – 2007. - №7. – С. 9-11.
7. Лапчик М. П. Теория и методика преподавания информатики: учебник / [М.П.Лапчик, И.Г.Семакин, Е.К.Хеннер, М.И.Рагулина и др.] под ред. М.П.Лапчика - М.: Издательский цент «Академия», 2008. – 592 с.
8. Левченко И.В., Заславская О.Ю. Учебно-методический материал по теме «Основы алгоритмизации и программирования. // Информатика в школе. – 2012. – С. 35 - 43
9. Лукин С. Н. Мой взгляд на информатику. [Текст] // Педагогическая информатика. - 2011. - №3. – С. 12-17.
10. Новичков В.С., Панфилова В.С., Пылькин А.Н. Алгоритмизация и программирование на Турбо Паскале: учеб. пособие. М.: Горячая линия – Телеком, 2005.
11. Осломовская И.М. Как организовать дифференцированное обучение. [Текст] – М.: 2012 – 160 с.

12. Программирование на языке Паскаль: задачник/ под ред. О.Ф.Усковой. СПб.: Питер, 2005.
13. Сборник психологических тестов. Часть I: Пособие / Сост. Е.Е.Миронова – Мн.: Женский институт ЭНВИЛА, 2006. – 155 с.
14. Сборник психологических тестов. Часть II: Пособие / Сост. Е.Е.Миронова – Мн.: Женский институт ЭНВИЛА, 2006. – 146 с.
15. Селевко Г.Г. Энциклопедия образовательных технологий. Москва. Народное образование. 2005. - 556 с.
16. Унт И.Э. Индивидуализация и дифференциация обучения / И.Э. Унт-М.: Педагогика, 1990.- 192 с.
17. Филиппов В.И. Анализ результатов государственной итоговой аттестации по информатике и ИКТ на территории Московской области в 2016 году // Анализ результатов государственной итоговой аттестации по основным образовательным программам на территории Московской области в 2016 году. М.: АСОУ. 2016
18. Якиманская И.С. Дифференцированное обучение: «внешние» и «внутренние» формы // Директор школы. 1995. № 3. С. 39-45.

#### **Сайты:**

19. Федеральные государственные образовательные стандарты общего образования. Режим доступа: <https://минобрнауки.рф/документы/543> (22.12.2018)
20. Гузаева М.Ю. Особенности обучения младших школьников программированию. Режим доступа: <http://pedsovet.su/publ/44-1-0-4056> (22.12.2018)
21. Кушнер Ю.З. Методология и методы педагогического исследования (учебно-методическое пособие). Режим доступа: [http://pedlib.ru/Books/1/0473/index.shtml?from\\_page=28](http://pedlib.ru/Books/1/0473/index.shtml?from_page=28) (07.12.2018)

22. Актуальный характер экспериментально-педагогической деятельности. Режим доступа: <https://www.booksite.ru/fulltext/kos/hmar/ped/1.htm> (22.12.2018)

23. Опыт 10 стран по обучению школьников программированию. Режим доступа: <http://www.sfx.tula.ru/news/infoblog/8105> (22.12.2018)

24. Современное программирование на языке Паскаль. Режим доступа: <http://pascalabc.net/> (22.12.2018)

25. Дифференцированное обучение: «внешние» и «внутренние» формы. Режим доступа: <http://ecsocman.hse.ru/direktor/msg/171966.html> (22.12.2018)