Тема 10. Основы программирования

Цели изучения:
- формирование умений и навыков программирования;
- развитие системного мышления.

Ключевые слова: программа, программирование, среда программирования, транслятор, отладчик, язык программирования, команда, оператор, данные, типы данных, переменные, константы.

Методическое обеспечение: [1, 7]: главы 9–16, [3, 4].

Количество часов:

Класс	Базовый вариант		Расширенный вариант	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)
10	11	20	25	25
11	9	20	23	25
Итого:	20	40	48	50

В результате изучения темы учащиеся должны овладеть основными приёмами программирования. Конкретные знания, умения и получаемые учащимися возможности, специфические для данной области знаний, описаны отдельно для каждой главы темы «Основы программирования».

Место темы в непрерывном курсе информатики в 7–11 классах

Изучение программирования базируется на предметных знаниях, полученных в курсе информатики основной школы при изучении содержательной линии «Алгоритмы и элементы программирования» (ПООП ООО [10]):
- интерфейса выбранного языка программирования;
- типов и структур данных;
- основных конструкций языка программирования,

а также на умениях кодирования базовых алгоритмических конструкций на выбранном языке программирования и создания программ для решения типовых задач базового уровня с использованием основных алгоритмических конструкций.

В результате обучения должны быть достигнуты предметные результаты освоения образовательной программы в соответствии с требованиями ФГОС [9]: владение «умением понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня; знанием основных конструкций программирования; умением анализировать алгоритмы с использованием таблиц», «стандартными приёмами написания на алгоритмическом языке программы для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ», «компьютерными средствами представления и анализа данных».

Анализ списка требований ФГОС [9] к предметным результатам освоения информатики позволяет сделать вывод о возрастающей роли навыков алгоритмизации и программирования в образовании современного школьника, так как они составляют для базового уровня 42% от общего количества требований к результатам освоения образовательной программы. Возрастающая роль программирования в школьном курсе информатики объясняется тем, что деятельность по программированию создаёт условия для развития мышления ученика, что в свою очередь обеспечивает становление его культуры при работе с информацией. Таким образом, программирование должно в полной мере раскрыть не пользовательские, а образовательные возможности, которыми обладает компьютер как средство обучения.

Образовательные возможности компьютера должны обеспечить познание основ «научных методов познания окружающего мира» (ФГОС [9]), что возможно, в частности, при изучении программирования в процессе моделирования, так как интеграция обучения программированию с задачами моделирования позволяет организовать системную исследовательскую деятельность, основанную на научном подходе, и реализовать системно-деятельностный подход в обучении. Таким образом, требования ФГОС [9] определили новую парадигму информатики, которая на новом витке развития школьного образования в нашей стране восходит к мировоззренческим идеям первого периода школьной информатики, сформулированным А. П. Ершовым в работе «Программирование — вторая грамотность» как необходимость «актуализировать в виде программ информационную модель мира», «сблизить мир машин и мир живого, программы природы и программы, составленные человеком».

Изучение темы «Основы программирования» вносит вклад в достижение результатов освоения образовательной программы.

Ожидаемые результаты обучения:
- личностные: Л-4, Л-5, Л-6, Л-7, Л-9;
- метапредметные: М-1, М-3, М-4;
- предметные: О-2, О-3, И-3.1, И-3.2, И-3.3, И-4-1, И-5.1, И-6.

Педагогические технологии, используемые при изучении темы:
- диалогическая технология;
- технология организации самостоятельной деятельности;
- технология организации проектной деятельности.

Концептуальная основа методики обучения программированию

Методика обучения программированию, реализованная в учебнике [2, 7], основана на системно-деятельностном подходе, организуемом в процессе информационного моделирования средствами языка программирования. Моделирование, с одной стороны, представляет собой современный инструмент системного анализа и синтеза, с другой стороны, обеспечивает обучение в процессе деятельности, определённой учебно-познавательными мотивами. Таким образом, обучение программированию при решении задач моделирования обеспечит достижение предметных результатов освоения программирования и развитие системы универсальных учебных действий.

Для изучения в общеобразовательной школе на базовом уровне выбраны языки программирования BASIC и Pascal, которые являются алгоритмическими языками выского уровня, реализующими технологию структурного программирования. Технология структурного программирования, опирающаяся на структуризацию действий, с одной стороны, соответствует особенностям мыслительной деятельности человека, которая сформировалась в ходе эволюции, с другой стороны, даёт фундаментальные знания, на базе которых возможно понимание новых открытий в области технологий программирования.

Простота синтаксиса выбранных для изучения языков программирования делает их доступными для изучения и понимания, т. е. обеспечивает соответствие сложности языка

программирования учебным возможностям учащихся. Доступность изучения, восприятия и достаточные возможности этих языков позволяют включить учащихся в творческий процесс деятельности по моделированию в среде программирования на основе понимания и интереса. Таким образом, выбранные языки могут обеспечить решение задачи познавательной мотивации, и, как следствие, формирование универсальных учебных действий. Сходство лингвистических конструкций выбранных для изучения языков программирования позволяет рассматривать методы программирования в двух средах параллельно до момента, определённого сложностью решаемых задач и спецификой языковых конструкций, не изучаемых на базовом уровне.

Существует множество различных версий языков BASIC и Pascal. Для изучении основ программирования авторы выделяют универсальные возможности каждого языка программирования, не связанные с их конкретными версиями. Так как изучение будет строиться на универсальных возможностях выделенных в конкретных версиях языков, будем обозначать их как BASIC и Pascal.

Компонентами учебной деятельности при освоении материалов каждой главы являются:

1) мотивирование деятельности;
2) освоение инструментария программирования;
3) деятельность по моделированию.

Первый компонент «Мотивирование деятельности» предполагает выделение в окружающей действительности информационных процессов, характеризующихся общими признаками по типу действий, например процессов, содержащих действия, которые выполняются последовательно без повторений и выбора, или содержащих повторяющиеся действия, или предполагающих выбор действия в зависимости от условия. Выделенные информационные процессы должны предполагать возможность исследования и быть в области интересов учащихся. Так как исследование будет выполняться в среде программирования, это потребует освоения инструментария программирования.

Особое внимание следует уделить формированию внутреннего познавательного мотива, который обеспечит осознанное выполнение учебной деятельности. Если внутренний мотив есть, то учащиеся охотно включаются в познавательную

ную деятельность. Решение этой задачи во многом зависит от учителя, который должен суметь привлечь внимание, инициировать возникновение интереса и побудить учащихся к деятельности. Это потребует от учителя эффективного использования примеров, заданий, которые личностно значимы для учащихся, обращаются к их жизненному опыту, являются современными и актуальными.

В качестве основного методического подхода к мотивированию деятельности следует выбрать объяснительно-иллюстративный. Этот метод предполагает устный рассказ учителя, обсуждение с учащимися, работу с печатными материалами или передачу учащимся готовой информации с помощью наглядных средств обучения (презентации, видеоматериалы).

В результате компонент мотивирования деятельности должен обеспечить понимание необходимости освоения инструментария программирования для проведения компьютерного моделирования информационных процессов.

Второй компонент «Освоение инструментария программирования» обеспечивает формирование знаний, умений и навыков применения инструментария программирования. Компонент использует богатый опыт обучения, основанный на передаче готовой информации, т. е. основным педагогическим методом должен быть репродуктивный.

Выбранный метод обучения определяет единую структуру компонента освоения инструментария программирования для каждого изучаемого параграфа, которая обеспечивает достижение уровней усвоения «понимание», «узнавание», «воспроизведение» и «применение». Достижение каждого уровня обеспечивается решением конкретных задач обучения.

Первая задача «Формирование представления об инструментарии программирования». Язык программирования представляет собой знаковую систему, изучение которой предполагает выделение синтаксиса — правил построения сообщений в языковой системе и семантики — правил истолкования сообщений в языковой системе. Знакомство с синтаксисом и семантикой конструкций языка программирования обеспечивает достижение первого уровня усвоения учебного материала — понимания.

Для параллельного описания синтаксиса и семантики двух языков применены таблицы с синтаксисом, пояснениями и примерами.

Вторая задача «Узнавание инструментария программирования». Задача решается за счёт набора готовых примеров. Работа с примерами требует от учащихся увидеть изучаемую конструкцию языка программирования и правильно её интерпритировать.

Третья задача «Формирование умения применять инструментарий программирования в типовых ситуациях». Решение этой задачи обеспечивает достижение уровня «воспроизведение». Достижение этого уровня может быть реализовано на основе метода упражнений, сущность которого заключается в том, что учащиеся производят многократные действия, т. е. упражняются в применении изучаемого инструментария программирования.

Для многократного воспроизведения действий по применению инструментария программирования предлагается набор заданий, который подобран таким образом, чтобы реализовать следующие приёмы обучения:

1) объяснение учителем, как реализуются команды, закодированные в конструкциях языка программирования;
2) воспроизведение учащимися действий по применению знаний;
3) тренировочная деятельность учащихся, направленная на совершенствование приобретаемых практических умений и формирование навыков.

Многократное выполнение однотипных заданий вырабатывает у учащихся умение применять изучаемый инструментарий программирования. Набор заданий отражает последовательное усложнение действий. Первоначально выполняются задания, аналогичные предложенным учителем. Однако подражание полезно в том случае, когда из него вырастает самостоятельная деятельность, поэтому набор заданий содержит не только действия «по образцу», но и задания, требующие собственных размышлений, а именно:

- задания на поиск ошибок в синтаксисе изучаемого инструментария программирования;
- задания, содержащие неопределённость.

В результате репродуктивный метод обучения, который является ведущим при освоении инструментария программирования, дополняется элементами активного обучения.

Важным структурным компонентом процесса обучения является контроль знаний и умений. Для организации самоконтроля материалы раздела содержат вопросы и задания для

контроля и задания-упражнения, которые группируются по степени их сложности следующим образом:

- воспроизведение изученных определений, синтаксиса и семантики инструментария программирования;
- самостоятельное определение дополнительных возможностей изучаемого инструментария программирования.

Четвёртая задача «Формирование навыков применения инструментария программирования». Навык применения инструментария программирования опирается на уровень усвоения, при котором учащиеся способны самостоятельно воспроизвести усвоенную информацию и применить её в нетиповых ситуациях, т. е. на уровне «применение». Поэтому важным элементом процесса обучения становится самостоятельная деятельность учащихся, которая реализуется набором заданий для освоения инструментария программирования. Задания для освоения инструментария программирования группируются по мере их усложнения следующим образом:

- чтение фрагмента готовой программы, определение результата;
- поиск, исправление ошибки в фрагменте готовой программы;
- составление программы для типовых ситуаций;
- составление программы для нетиповых ситуаций.

Задания составлены так, чтобы обеспечить обучение при различных уровнях трудности — от типовых действий до составления программ для нетиповых ситуаций. Тренировочная деятельность с использованием заданий, основанных на нетиповых ситуациях, содержит трудность, связанную с явлением переноса знаний. Усвоив конкретный инструментарий программирования, который показал учитель, учащиеся способны применить его. Когда инструментарий программирования необходимо применить в нетиповых ситуациях, учащимся приходится переносить усвоенные знания и умения на другие условия, что представляет собой трудность. Такое регулируемое преодоление трудностей обеспечивает осмысление изучаемого приёма программирования и формирование навыка программирования.

Для того чтобы инициировать деятельность учащихся по исследованию дополнительных возможностей инструментария программирования, каждая глава включает структурный элемент, называемый «Совет программиста». Такой струк-

турный элемент оформлен в виде обращения к учащемуся и может содержать:

- рекомендации самостоятельно с помощью справочной системы разобраться с дополнительными возможностями инструментария программирования;
- выделение особенностей выбранного для изучения языка программирования (Пример из параграфа 10.1: «Обратите внимание на то, что в языке программирования Pascal разделение данных на символьный и строковый типы является принципиальным. В языке программирования BASIC данные символьного типа являются частным случаем данных строкового типа»);
- советы по улучшению программного кода;
- предложение проявить любознательность (Пример из параграфа 12.2: «Поинтересуйтесь понятием “пространственная сложность” программы (алгоритма). Постарайтесь понять, какой вид сложность программы — временная или пространственная — наиболее актуальны для современных компьютеров»).

Самостоятельное изучение дополнительных возможностей инструментария программирования придаёт тренировочным упражнениям развивающий характер.

Третий компонент «Деятельность по моделированию» основан на системно-деятельностном подходе. Учащиеся, используя освоенный инструментарий и умение разрабатывать информационные модели, решают задачи по моделированию в среде программирования. Задачи по моделированию, которые ставятся перед учащимися, должны иметь сюжетное описание и основание для исследования.

Наиболее точно этим требованиям отвечают ситуационные задачи. Такая задача представляет собой интеграцию некоторой ситуации и цели исследования. Ситуационные задачи имеют единую структуру:

1) название;
2) описание ситуации;
3) наличие личностно-значимого вопроса (в явном или неявном виде).

В материалах учебника используются задачи из школьного курса различных предметов, задачи по моделированию в прикладных программных средах, задачи из старинных учебников, которые отвечают требованиям ситуационных задач, а

также современные ситуации из различных областей, содержащие основание для исследования.

Постановка задачи всегда предлагается в неформальном описании. В отличие от привычных для учащихся задач по школьным предметам, в которых точно формулируются исходные данные и вопрос, ситуационные задачи должны быть сформулированы в общем виде.

По характеру постановки цели все ситуационные задачи для моделирования средствами языка программирования можно разделить на группы: задачи, в которых цель исследования сформулирована в явном виде; и задачи, допускающие множественность целей, формулировка которых выполняется учащимися самостоятельно или в диалоге с учителем.

По степени подробности описания ситуационные задачи для моделирования средствами языка программирования можно разделить на группы: задачи, в которых параметры исследуемых объектов, условия, отношения описываются точно, при этом учащиеся должны отобрать те параметры, которые существенны при построении информационной модели; и задачи, в которых параметры исследуемых объектов не оговариваются, при этом учащиеся должны уточнить необходимые для информационной модели параметры, их значения, диапазон их изменения в процессе формализации.

Ситуационные задачи по моделированию включают демонстрационные примеры и набор заданий по моделированию, предназначенных для самостоятельной работы.

Этапы моделирования позволяют структурировать деятельность по моделированию в среде программирования:

Этап I. Постановка задачи.

Этап II. Разработка модели.

Этап III. Компьютерный эксперимент.

Этап IV. Анализ результатов моделирования.

Этап I. Постановка задачи

На этом этапе учащимся предлагается ситуационная задача в описательной форме. Определение цели моделирования является самым важным действием на этапе постановки задачи, так как от этого будет зависеть выбранный прототип моделирования и его параметры, существенные при построении информационной модели. Отталкиваясь от описания задачи, учащиеся выделяют цель исследования, прототип моделирования, его свойства, определяют параметры, которые являются исходными данными и результатами, соотношения между

исходными данными и результатами, строят образ действий по достижению цели моделирования. Таким образом, в виде постановки вопросов и поиска ответов выполняется формализация задачи.

Методически формализацию проводят в виде поиска ответов на вопросы, уточняющие общее описание задачи. От учителя требуется организовать диалог на основе наводящих вопросов. Возможной схемой обсуждения может быть следующая последовательность вопросов.

1. Что моделируется?
2. Каковы параметры моделируемого объекта?
3. Какие параметры являются исходными, какие — итоговыми?
4. Какие значения имеют исходные параметры?
5. Каков диапазон изменений параметров?
6. Как описать отношения и связи между параметрами?
7. Каковы правила получения результата?

Рассмотрение задачи моделирования на уроке должно отвечать необходимости применения определённого инструментария программирования, поэтому последовательностью и содержанием вопросов необходимо управлять таким образом, чтобы выделенная цель моделирования раскрывала возможности изученного инструментария программирования. Например, учитель знает, что задача «Поймай бабочку» (глава 12 «Ветвящиеся алгоритмы и модели») иллюстрирует возможность исследования ветвящихся процессов, которые описываются условным оператором. Поэтому в процессе диалога с учащимися он подводит их к требуемой цели исследования: «Смоделировать ситуацию случайного появления бабочки и сачка, накрывающего область примерного нахождения бабочки. Вычислить процент попадания бабочек в сачок». Сформулированная цель определяет прототип моделирования: система «бабочка и сачок, характеризующиеся взаимным расположением». Цель и прототип моделирования позволяют выполнить формализацию задачи.

Этап постановки задачи развивает умения и навыки осознания учебной задачи, осмысливания учебного материала, постановки целей, выделения существенного для решения, анализа и синтеза, абстрагирования и конкретизации, обобщения.

Это наиболее сложный этап, так как требуется из контекста задачи выделить цель исследования и осуществить

переход от описательной постановки задачи к её формализованному виду. Так как формализация выполняется в форме диалога учителя и учащихся, то от обеих сторон требуется умение с достаточной полнотой и точностью выражать свои мысли, умение слушать и слышать друг друга, умение ставить вопросы, вести диалог, критично относиться к своему мнению, что способствует развитию речевой коммуникации учащихся. В этом процессе огромная роль принадлежит учителю, который активизирует и направляет познавательную деятельность учащихся, организует диалог так, чтобы обеспечить переход к действиям по разработке информационной модели исследуемого объекта.

Этап II. Разработка модели

Информационная модель обязательно представляется в табличной форме (табл. 10.1), где отражаются параметры прототипа реального объекта и соответствующие им параметры для среды программирования с указанием типа данных и диапазона значений.

Таблица 10.1

Описание информационной модели объекта в среде программирования

Объект моделирования	Параметры				
	реального объекта		для среды программирования		
	Название	Значение	Имя/ значение	Переменная/ константа	Тип

В зависимости от цели моделирования информационная модель может быть создана дополнительно в геометрической, математической и других формах.

Информационная модель является основанием для построения компьютерной модели в среде программирования. Разработка компьютерной модели обязательно предполагает моделирование последовательности действий в виде алгоритма и программы. При разработке компьютерной модели в среде программирования исполнителем действий по созданию модели является компьютер. Разработчик модели должен продумать точную последовательность действий исполнителя-компьютера, в результате которых будет создана модель в

среде программирования. При этом собственно описание последовательности действий также является моделью. Учащиеся, освоившие курс информатики основной школы, знают, что модель действий исполнителя может быть представлена в следующих знаковых формах: алгоритм, программа. Они знакомы с различными формами представления алгоритма, среди которых наиболее распространёнными в школьном курсе информатики являются графическая и словесная. Выбор формы представления алгоритма определяется наглядностью, простотой и удобством чтения.

Компьютерная модель представляет собой модель последовательности действий, описанную на языке программирования. Среда программирования предоставляет инструментарий описания алгоритма для компьютера.

В среде программирования правильность разработанной модели напрямую зависит от личных навыков программиста. Выявление и исправление явных ошибок программы осуществляется в ходе отладки. Владение технологией трассировки позволяет продумать, как выполняются действия и оценить результат каждого действия. Учащимся предлагается освоение новых понятий, связанных с технологией программирования: трассировка, трассировочная таблица.

Разработка программы потребует от учащихся применить на практике умение работать с инструментарием программирования, просчитать действия программы, возможно, освоить новые инструментарии и технологические приёмы работы в среде программирования. Результатом этапа разработки модели должна стать программа, в которой нет явных ошибок.

Этап III. Компьютерный эксперимент

Компьютерный эксперимент включает тестирование компьютерной модели-программы и собственно эксперимент. Модель, в которой нет явных ошибок, ещё не гарантирует достижение цели моделирования. Доказательством соответствия модели реальному объекту и цели моделирования является тестирование, т. е. процесс проверки правильности построенной модели, соответствия модели реальному объекту и цели моделирования. Учащиеся должны продумать набор исходных данных, для которых результат известен или предварительно определён другими способами. Если в результате тестирования выявляется несоответствие модели реальному объекту, то следует скорректировать модель на любом из предыдущих этапов моделирования. Если выявляется со-

ответствие модели реальному объекту, то можно переходить к компьютерному эксперименту, удовлетворяющему цели моделирования, т. е. к проведению исследования.

Эксперимент в общем случае — это опыт, который проводится с объектом или моделью и представляет собой воздействие на изучаемый объект посредством специальных инструментов для определения того, как реагирует экспериментальный образец на воздействия. Компьютерный эксперимент в среде программирования заключается в воздействии на модель, варьировании значений её параметров инструментами программной среды, поэтому по способам действия компьютерный эксперимент приближается к естественному эксперименту.

Этап компьютерного эксперимента опирается на план эксперимента. Первым в плане эксперимента является обязательное проведение тестирования. Выбор тестовых примеров для каждой задачи обсуждается и выбирается отдельно. Это может быть визуальный контроль для образно-знаковых моделей или сравнение результатов расчётов, полученных разными способами, для математических моделей. Далее в плане предусматривается серия экспериментов, удовлетворяющих цели моделирования и расширяющих возможности модели по исследованию действительности.

Проводя компьютерный эксперимент, учащиеся осваивают общие подходы к экспериментальному исследованию: составление плана эксперимента, включающего обязательное тестирование, проведение эксперимента в соответствии с целью исследования и дополнительные эксперименты. Этап компьютерного эксперимента развивает у учащихся умение планировать деятельность и оценивать полученные результаты. Каждый эксперимент должен сопровождаться осмыслением результатов, которые станут основой для анализа результатов моделирования.

Этап IV. Анализ результатов моделирования

Конечной целью исследования является принятие решения, которое вырабатывается на основе анализа результатов моделирования. Этот этап определяет дальнейшую последовательность действий: либо исследование заканчивается принятием решения, либо продолжается.

Если результаты эксперимента не соответствуют целям поставленной задачи, то это означает, что допущены ошибки на предыдущих этапах моделирования. Ошибки необходимо

выявить и скорректировать модель на любом из предыдущих этапов, т. е. выполняется возврат к одному из этапов моделирования. Процесс повторяется до тех пор, пока результаты эксперимента не будут отвечать целям моделирования.

На этом этапе моделирования у учащихся формируются навыки оценки и осмысления результата, а также смысловая личностная установка, заключающаяся в понимании роли программирования как инструмента моделирования.

В материалах учебника [2, 7] деятельность по моделированию в среде программирования организована состоящей из двух стадий, которые характеризуются различным уровнем освоения инструментария программирования и сложностью ситуационных задач.

Первая стадия. Деятельность по моделированию, ориентированная на применение изучаемого инструментария программирования:

- демонстрационные примеры;
- набор заданий для самостоятельной деятельности.

Вторая стадия. Проектная деятельность по моделированию:

- демонстрационные примеры;
- набор тем проектов для самостоятельной деятельности.

Первая стадия деятельности по моделированию предполагает разработку модели информационных процессов, которые могут быть описаны главным образом изучаемым инструментарием программирования. Например, если исследуются циклические процессы с известным числом повторений, то для их описания предлагается оператор цикла с параметром. После освоения инструментария программирования предлагаются задачи по моделированию: получение суммы чисел от 1 до 100 (легенда о Гауссе), решение старинной задачи «За первый гвоздь — полушка...» — вычисление нарастающей стоимости гвоздей в подковах и т. д., т. е. задачи, в которых процессы могут быть описаны с помощью цикла с параметром. Если исследуются процессы, основанные на выборе действия в зависимости от условия, то для их описания предлагается условный оператор. После освоения инструментария программирования предлагаются задачи: иллюстрация попадания бабочки в сачок, вычисление числа ϖ и т. д., т. е. задачи, в которых процессы могут быть описаны с помощью условного оператора.

Деятельность по моделированию с использованием изученного инструментария программирования должна, с одной стороны, поддержать формирование навыка программирования, с другой стороны, ответить на вопрос о том, для чего изучался данный инструментарий. Такая деятельность даёт понимание роли моделирования как метода познания мира, а программирования — как инструмента моделирования и обеспечивает решение задачи «сформированности мировоззрения, соответствующего современному уровню развития науки» (ФГОС [10]) у учащихся.

Организация первой стадии деятельности по моделированию включает демонстрационный пример с решением и набор заданий для самостоятельной работы.

Демонстрационные примеры предполагают обсуждение с учителем или самостоятельное рассмотрение учащимися. Такие задачи дают наглядное представление об этапах моделирования в среде программирования и позволяют на практике применить выработанный подход к моделированию. Этап компьютерного эксперимента, описывающего увлекательную, иногда забавную ситуацию, решает задачу мотивации и позволяет организовать активную учебно-познавательную деятельность по анализу параметров модели и модификации модели-программы, учитывая различные образовательные потребности учащихся. Например, моделирование обработки символьных данных иллюстрируется задачей «Испуганный НЛО», целью которой является создание ASCII-анимации. При рассмотрении задачи приводится программный код, моделирующий движение объекта по экрану. Объект изображается символами. Когда объект достигает границ области, символами рисуется удар о стену, а направление движения изменяется. Учащимся предлагается протестировать программу и модифицировать её, добавив эффект увеличения размеров НЛО при ударе и сопровождающий его звуковой сигнал. Часть учащихся будет способна только протестировать программу, другая часть — самостоятельно познакомиться с дополнительными возможностями среды программирования. Примеры учебных задач с решением и набор заданий по моделированию для самостоятельной работы можно использовать в различной комбинации. Варианты комбинирования определяются уровнем изучения информатики, уровнем подготовки учащихся, вида их мотивации к учебной деятельности и возможностями учебного времени.

Вторая стадия деятельности по моделированию включает демонстрационный пример проекта и набор тем проектов для самостоятельного выполнения. Задачи этой стадии деятельности по моделированию позволяют обобщить практический опыт программирования в процессе учебной деятельности учащихся, создавая условия для реализации метода проектов.

Задачи этой стадии являются более сложными и не только потребуют от учащихся возможности использовать изученный инструментарий программирования и типовые алгоритмы обработки данных, но и предполагают самостоятельную деятельность по освоению новых инструментариев программирования, алгоритмов обработки данных, технологий, поэтому они ориентируются на учащихся, чей интерес к программированию выходит за рамки школьного курса на базовом уровне. Выполнение проектных задач второй стадии деятельности по моделированию определяется: конкретными возможностями учебной организации, уровнем усвоения материала учащимися, уровнем изучения информатики в школе, уровнем мотивации учащихся. Если условия работы в конкретном классе по конкретному учебному плану не позволяют организовать работу над проектными задачами в урочное время, то такие задачи могут быть предложены для учащихся, заинтересованных в углублённом уровне изучения программирования в ходе внеурочной деятельности.

Предложенная в учебнике методика предлагает изучать программирование как инструмент познания мира посредством моделирования. Компоненты деятельности включают: мотивацию познавательной деятельности, освоение знаний в области инструментария программирования, приобретение умений и навыков использования алгоритмов обработки информации и инструментария программирования, выполнение компьютерного моделирования.

Общая структура деятельности обучения программированию в процессе моделирования представлена на рис. 10.1.

Рекомендации по раскрытию содержания темы

Изучение элементов программирования на основе структуризации способов управления вычислительным процессом позволяет выделить три типа действий, описываемых алгоритмическими конструкциями и конструкциями языка программирования: линейные, циклические, ветвящиеся. Первоначально осваиваются способы линейного управления, затем

Рис. 10.1. Структура целенаправленной учебной деятельности обучения программированию в процессе моделирования

циклические действия с известным количеством повторений, ветвления. Далее рассматривается возможность структуризации задачи с использованием освоенных способов управления вычислительным процессом. Выбранная последовательность связана с тем, что в развитии учащегося сначала проходит стадия овладения последовательными операциями, многократное повторение операций будет представлять собой этап их усложнения. Овладение логическими операциями как умением понимать логику рассуждений является наиболее сложным, поэтому изучение ветвлений следует после освоения описания линейных и циклических алгоритмов с известным количеством повторений.

Глава 9 «Линейные алгоритмы и модели в графике» включает описание команд и операторов графики языков программирования BASIC и Pascal.

Изучение базируется на предметных знаниях, полученных в курсе информатики основной школы: понятиях линейного алгоритма, информационной модели и этапов моделирования.

Количество часов:

Классы	Базовый вариант		Расширенный вариант	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)
10	1	2	2	2
11	0	0	0	2
Итого:	1	2	2	4

В результате изучения параграфа учащиеся

должны знать:

- особенности линейных процессов;
- команды и операторы среды программирования, обеспечивающие графический режим и создание графических примитивов;
- этапы моделирования в среде программирования;

должны уметь:

- определять результат готового фрагмента программы;
- составлять линейные программы, используя операторы графики;

получат возможность:

- познакомиться с использованием в программах команд и операторов, управляющих графическими возможностями среды программирования;
- моделировать линейные графические процессы в среде программирования.

Изучение программирования следует начинать с рассмотрения линейных алгоритмов в графике, так как, с одной стороны, графика играет особую роль в восприятии человека, с другой стороны, создание графических моделей является процессом, доступным по уровню сложности учащимся и позволяющим вовлечь школьников в деятельность по программированию. Учащиеся охотно вовлекаются в деятельность по моделированию графических объектов, что обеспечивает формирование осознанного познавательного мотива, который определяет последующую учебную деятельность.

В начале изучения главы следует предложить иллюстрированный рассказ о роли зрительных образов для человека, практическом применении и возможностях реализации компьютерной графики, выделив графические возможности среды программирования. Дальнейшее изучение команд и операторов графики следует организовать в форме практической самостоятельной работы учащихся с учебником [2, 7]. Учащиеся должны:

1) прочитать описание команды, оператора (параграфы 9.2–9.4);
2) набрать в среде программирования готовый демонстрационный пример;
3) провести эксперимент на основе примера с целью исследования каждого параметра команды, оператора в соответствии с рекомендациями в учебнике.

Следует обратить внимание учащихся на «Совет программиста», содержащий алгоритм построения изображения в среде программирования (параграф 9.4).

Для закрепления теоретических знаний в области базовых понятий компьютерной графики учащимся необходимо подготовить устные ответы на вопросы и задания 9.1–9.9. Для закрепления практических навыков необходимо выполнить задания для освоения инструментария программирования. Набор заданий:

- обеспечивает тренировочную деятельность;
- допускает вариативность использования заданий с учётом начального уровня подготовки учащихся и результатов их самостоятельной практической работы.

Включение в осознанную деятельность возможно только в том случае, если учащийся понимает, для чего он выполняет эту деятельность, и видит возможность реализации познавательных целей с использованием предложенного инструментария, поэтому обязательным является решение задачи моделирования графического объекта «Схематическое изображение лица» (параграф 9.5).

Из курса информатики в основной школе учащимся известны этапы построения компьютерной модели, которые следует конкретизировать применительно для среды программирования:

– на этапе разработки модели необходимо создание информационной модели в табличной форме, включающее описание объекта моделирования с помощью параметров (названия, значения), которым необходимо поставить в соответствие параметры объекта моделирования в среде программирования (имя, значение);
– на этапе разработки модели возможно создание модели в форме, удобной для решения задачи (математической, графической, словесной и т. д.);
– на этапе разработки модели в среде программирования обязательным является создание модели последовательности действий: алгоритма, программы;
– на этапе компьютерного эксперимента обязательным являются отладка и тестирование модели, которые должны предшествовать проведению исследования.

Задача моделирования в среде программирования «Схематическое изображение лица» имеет низкий уровень сложности, для того чтобы сделать акцент на:

- реализации моделирования в среде программирования;
- технологических приёмах отладки программы.

Повысить эффективность обучения возможно *за счёт самостоятельной деятельности* учащихся по освоению инструментария программирования (как базовый набор рекомендованы задания 9.10–9.12) и по моделированию (как базовый набор рекомендованы задания по моделированию 9.1 (а, б)), для которой можно рекомендовать 2 часа.

Глава 10. «Линейные вычислительные алгоритмы и модели» содержит базовые сведения о языках программирования: концепцию данных и типов данных, правила записи арифметических выражений, синтаксис и семантику оператора присваивания, ввода, вывода данных.

Перечисленные понятия языка программирования изучены в курсе информатики основной школы, поэтому в курсе средней школы они должны быть рассмотрены в форме повторения и применения при компьютерном моделировании.

Количество часов:

Класс	Базовый вариант		Расширенный вариант	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)
10	2	2	4	2
11	0	0	0	2
Итого:	2	2	4	4

В результате изучения темы параграфа учащиеся

должны знать:

- структуру программы;
- концепцию типов данных;
- правила записи арифметических выражений;
- синтаксис и семантику оператора присваивания, ввода и вывода;

должны уметь:

- определять результат готового фрагмента программы;
- выявлять и исправлять ошибки в готовом фрагменте программы;
- составлять линейные программы, используя оператор присваивания, ввод и вывод;
- моделировать линейные вычислительные процессы в среде программирования;

получат возможность:

- расширить свои знания о возможностях операторов присваивания, вводе и выводе данных;
- создавать модели обработки данных в среде программирования на основе линейного алгоритма;
- интерпретировать результаты, получаемые в ходе моделирования реальных процессов;

- анализировать в процессе тестирования разработанные в среде программирования информационные модели на предмет соответствия реальному объекту.

На первом уроке необходимо организовать повторение базовых понятий. Опираясь на материалы учебника (параграфы 10.1, 10.2), используя объяснительно-иллюстративный метод, следует повторить понятие данных как существенное для изучаемой области знаний.

Данные следует определить как инструмент, с помощью которого можно описать информацию об объекте исследования так, чтобы её можно было обрабатывать в среде программирования.

Первоначально вводится понятие простых данных как структур, которые состоят из одного элемента и имеют одно значение. Понятие структурированных данных как структур, которые состоят из фиксированного количества элементов, будет введено далее (глава 15 учебника). Такое представление о данных проще современного представления о данных в программировании, но оно используется на практике и доступно по восприятию учащимся базового уровня изучения информатики.

Работа с простыми данными предполагает выделение среди них классов констант и переменных, которые в свою очередь принадлежат одному из типов данных. Тип данных определяет множество значений, которые могут принимать данные, набор операций, которые можно выполнять с ними, а также объём памяти для их хранения. Выбранные для изучения языки программирования позволяют работать с большим количеством разных типов данных. Для достижения результатов освоения, сформулированных ФГОС [9], в соответствии с принципами научности и доступности достаточно выделить в простых данных следующие типы: числовые и символьные, а числовые данные классифицировать как целые и вещественные.

Повторение правил записи арифметических выражений, оператора присваивания, ввода и вывода данных может быть организовано в форме практической работы, предложенной учителем. Практическая работа должна обеспечить возможность обсуждения и применения операторов. Таким примером может быть составление программы вычисления $y = 5 * x^2 + \frac{3}{4} * x$, где значение переменной x задаётся с клавиатуры, результат вычисления y выводится на экран монитора.

Самостоятельная деятельность учащихся должна обеспечить закрепление теоретических знаний и практических навыков, поэтому включает в себя:

1) работу с учебником:
   - параграфы 10.1, 10.2;
   - рубрика «Совет программиста» параграфов 10.1, 10.2;
2) работу над вопросами и заданиями:
   - базовый набор — вопросы и задания 10.1–10.16;
   - набор вопросов и заданий 10.17–10.21 позволяет вовлечь учащихся в исследовательскую деятельность, углубить базовые знания;
3) Задания для освоения инструментария программирования:
   - базовый набор — задания для освоения инструментария программирования 10.1–10.18;
   - набор заданий для освоения инструментария программирования 10.19–10.29 ориентирован на учащихся, мотивированных на изучение программирования выше базового уровня.

Самостоятельная деятельность учащихся должна быть организована как на уроке, так и в форме домашней работы, поэтому целесообразно комбинировать задания так, чтобы оптимально обеспечить самостоятельную деятельность.

На втором уроке следует рассмотреть демонстрационный пример моделирования, основанный на обработке числовых данных (параграф 10.5). Моделирование выполняется на основе системных принципов, поэтому предполагает выделение объекта или системы моделирования, описания параметров модели.

Описание информационной модели объекта применительно к среде программирования требует привязки к используемым правилам представления данных (см. табл. 10.1). Параметры реального объекта характеризуются названием и значением, названию параметра соответствует имя, значение параметра требует указания класса (переменная/константа) и типа данных.

Для иллюстрации моделирования в среде программирования предлагается рассмотреть задачу, имеющую следующее описание: «В популярной телевизионной передаче «Квартирный вопрос» планируют ремонт в детской комнате. Организаторы должны решить, сколько потребуется масляной краски на воплощение идеи дизайнера по окраске крышки прямоугольного стола». Описание задачи представлено в неформализованном виде.

В диалоге с учащимися необходимо определить цель моделирования — определить количество расходуемой краски. В процессе формализации задачи выявляется объект моделирования — система «крышка стола и краска». Объекты, образующие систему, связаны пространственными отношениями, так как краска покрывает поверхность стола. В процессе формализации задачи необходимо выбрать параметры, которые являются существенными при моделировании реального объекта, уточнить значения этих параметров, возможный диапазон изменения параметров, а также математическое описание отношения между объектами системы. Результатом разработки информационной модели системы «крышка стола и краска» для среды программирования является обязательное описание модели в табличной форме (табл. 10.2), дополнительно может быть выполнена модель в форме, учитывающей особенности моделируемого объекта или системы. В данном случае это геометрическая и математические модели.

Таблица 10.2

Описание модели для среды программирования

Объект моделирования		Параметры				
		реального объекта		для среды программирования		
		Название	Значение	Имя	Переменная/ константа	Тип
Система «крышка стола и краска»	Крышка стола	Длина стороны стола (м)	1,2 м	a	Переменная	Вещественное число
		Длина стороны стола (м)	0,5 м	b	Переменная	Вещественное число
		Площадь крышки стола ($м^2$)	Результат расчёта	s	Переменная	Вещественное число
	Краска	Расход краски на единицу площади ($кг/м^2$)	0,2 или 0,3	r	Переменная	Вещественное число
		расход краски на поверхность крышки стола (кг)	Результат расчёта	v	Переменная	Вещественное число

Моделирование выполняется в среде программирования и предполагает точное описание последовательности действий, позволяющих построить модель. Разработка модели последовательности действий в виде алгоритма и программы, т. е. алгоритмических моделей, является обязательной на этапе разработки информационной модели.

В примере, выбранном для иллюстрации форм описания информационных моделей, алгоритм представлен в виде блок-схемы (рис. 10.2), программа описана на выбранных для изучения языках программирования (табл. 10.3).

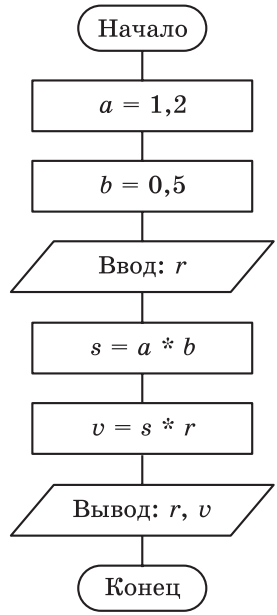

Рис. 10.2. Пример моделирования последовательности действий: алгоритм

Во всех случаях, когда это целесообразно, учащимся предлагаются словесные или словесно-формульные способы представления алгоритмов. Так подчёркивается возможность выбора формы описания алгоритма.

Таблица 10.3

Пример моделирования последовательности действий: программа

BASIC	Pascal

PRINT "Окраска крышки стола"
a = 1.2
b = .5
INPUT "Расход краски на единицу площади"; r
s = a * b
v = s * r
PRINT v; " кг краски необходимо для стола площадью "; s; " м кв.";
	var
	  a, b, r, s, v: real;
	begin
	  writeln ('Окраска крышки стола');
	  a:=1.2;
	  b:=.5;
	  write ('Расход краски на единицу площади');
	  readln (r);
	  s:=a*b;
	  v:=s*r;
	  writeln (v, ' кг краски необходимо для стола площадью ', s, 'м кв.');
	end.

Компьютерный эксперимент включает обязательную отладку и тестирование компьютерной модели. Учащиеся должны выполнить последовательность действий, предложенную в учебнике (параграф 10.5) и описывающую алгоритм отладки и тестирования.

Компьютерные эксперименты следует оформить в табличной форме, и на основе полученных данных сделать выводы о зависимости расхода краски от параметров стола, от вида краски.

Обратите внимание на то, что демонстрационные примеры моделирования предложены в подробном описании в соответствии со схемой моделирования. Такое описание допускает вариативность в использовании материалов в зависимости от подготовленности учащихся (самостоятельное составление алгоритма или программы, чтение готового алгоритма или программы), что обеспечивает индивидуальный подход в обучении.

Самостоятельная деятельность учащихся обеспечена набором заданий по моделированию в конце параграфа 10.5. Самостоятельная деятельность учащихся по моделированию линейных алгоритмов может быть построена вариативно с учётом подготовленности учащихся и их мотивации для изучения программирования, для её выполнения можно рекомендовать 2 часа.

Глава 11 «Циклические алгоритмы и модели с известным числом повторений» содержит описание циклического алгоритма, классификацию циклов, синтаксис оператора цикла с параметром и алгоритмы вычисления суммы числовой последовательности.

Изучение базируется на предметных знаниях, полученных в курсе информатики основной школы: понятиях цикла, циклического алгоритма.

Количество часов:

Класс	Базовый вариант		Расширенный вариант	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)
10	2	4	4	2
11	2	2	4	2
Итого:	4	6	8	4

В результате изучения темы параграфа учащиеся

должны знать:

- особенности циклических процессов с известным числом повторений;
- синтаксис и семантику оператора цикла с параметром;
- алгоритм вычисления суммы (произведения) элементов числовой последовательности;
- понятие трассировки;

должны уметь:

- определять результат фрагмента программы;
- анализировать работу фрагмента программы по трассировочной таблице;
- выявлять и исправлять ошибки в готовом фрагменте программы;
- составлять циклические программы, используя операторы цикла с параметром;
- моделировать циклические процессы в среде программирования;

получат возможность:

- оценивать числовые параметры моделируемых объектов и процессов;
- моделировать циклические вычислительные процессы;

- моделировать циклические вычислительные процессы с использованием рекуррентных соотношений;
- интерпретировать результаты, получаемые в ходе моделирования реальных процессов;
- анализировать в процессе тестирования разработанные в среде программирования информационные модели на предмет соответствия реальному объекту.

Первый урок по данной теме *в 10 классе* следует посвятить понятию цикла и оператору цикла с параметром.

Изучение темы следует начать с определения понятия цикла. В диалоге с учащимися на основе примеров из окружающего мира (можно обратить внимание учащихся на особенности движения небесных тел и влияние роста космического мусора на движение планет; смену времён года, как и на опасность её нарушить в результате деятельности человека; жизненные циклы растений, живых организмов и возможность ими управлять; смену цивилизаций, поколений людей, экономических кризисов, производственных циклов и необходимость их предвидеть; этапы развития техники, науки и т. д.) и знаний из курса информатики основной школы следует ввести понятие цикла и обязательные элементы для организации цикла, классификацию циклов.

Следует выделить цикл с параметром как частный случай конечного цикла, рассмотреть способ его представления в форме блок-схемы, обсудить синтаксис оператора цикла с параметром (параграф 11.1).

Работу с примерами использования цикла с параметром (таблица 11.2 параграфа 11.1) можно построить следующим образом: предложить учащимся внести свои изменения в представленные фрагменты и предсказать, как изменится результат работы фрагментов.

В классе на уроке:

1) следует разобрать в классе пример с построением трассировочной таблицы (задание 11.1 из параграфа), рассмотреть задания 11.1–11.4 из набора для освоения инструментария программирования;
2) следует рассмотреть задание 11.2 из параграфа, обратив внимание учащихся на основные типовые ошибки, возникающие при работе с оператором цикла с параметром;
3) можно предложить учащимся разбиться на пары, выбрать какое-либо из заданий 11.7–11.14 на освоение

инструментария программирования и выполнить его в виде практической работы на компьютере, а затем представить получившиеся программы одноклассникам, рассказать о проблемах, с которыми пришлось столкнуться, и способами их решения.

Для самостоятельной деятельности дома следует предложить учащимся на выбор выполнить какое-либо из заданий на освоение инструментария программирования циклов с параметром (базовый набор образуют задания 11.5–11.11). Навык трассировки программ поможет ученикам в дальнейшем самостоятельно искать ошибки в программах. Стоит также использовать для домашнего задания вопросы и задания 11.1–11.10, обратив внимание учащихся, что ответы на задания 11.8–11.10 могут быть получены или проверены экспериментально с помощью среды программирования.

Второй урок данной темы *в 10 классе* посвящён алгоритмам вычисления суммы числовой последовательности. При наличии в классе учащихся разного уровня подготовленности и мотивации можно предложить им выбор способа знакомства с темой: с учителем или самостоятельно.

- Учитель с первой группой учащихся может разобрать на доске модель «Легенда о Гауссе» (параграф 11.3).
- Ученикам, способным на выполнение самостоятельной деятельности, следует предложить в это время самостоятельно по учебнику выполнить практическую работу на компьютере «За первый гвоздь — полушка...», в которой используется рекуррентная формула получения очередного слагаемого (параграф 11.4).

Самостоятельная деятельность учащихся обеспечена заданиями по моделированию 11.1, 11.2 (параграф 11.4), для выполнения которых можно рекомендовать 4 часа.

При возвращении к изучению программирования циклов с параметром *в 11 классе* необходимо организовать самостоятельную работу учащихся по повторению основных знаний и навыков программирования циклов с известным числом повторений с помощью материалов учебника (параграфы 11.1, 11.2), выполнение теста-тренажёра. Для актуализации знаний можно использовать вопросы и задания 11.1–11.10, задания для освоения инструментария программирования циклических вычислений 11.18–11.22 (обратите внимание на то, что задания 11.20 и 11.21 требуют умения выделять и обрабатывать цифры числа). Такая деятельность должна предшествовать уроку.

Во время урока изучение циклических алгоритмов с известным числом повторений в 11 классе следует построить на выполнении заданий по моделированию 11.3 и 11.4 (параграф 11.4) или проекта на обобщение знаний «Альпинист-экстремал» (параграф 11.5). Выполнение задания из набора заданий по моделированию циклических вычислений позволит обеспечить уровень усвоения «применение знаний в типовых ситуациях» и включить учащихся в деятельность по моделированию.

Проект на обобщение знаний «Альпинист-экстремал» (параграф 11.5) служит хорошей иллюстрацией использования вложенных циклов, поэтому при дефиците времени можно предложить учащимся ознакомиться с готовым кодом проекта, формируя навык чтения готовой программы, и провести эксперименты в компьютерной модели. Если учебное время позволяет, то целесообразно выполнить проект, соблюдая все этапы моделирования.

Выбор заданий для деятельности во время урока определяется уровнем подготовки и мотивирования учащихся.

Самостоятельная деятельность учащихся обеспечивается набором тем проектов, предложенных в параграфе 11.5, выбор и выполнение которых зависит от уровня подготовленности и мотивации учащихся при изучении основ программирования (рекомендовано 2 часа).

Глава 12 «Ветвящиеся алгоритмы и модели» включает понятие ветвления, описание особенностей полного и неполного ветвления, синтаксис условного оператора, правила записи условий, а также задачи по моделированию ветвящихся процессов.

Изучение базируется на предметных знаниях, полученных в курсе информатики основной школы: понятиях ветвления, ветвящегося алгоритма, логического выражения.

Количество часов:

Класс	Базовый вариант		Расширенный вариант	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендация)
10	2	4	4	2
11	2	2	4	2
Итого:	4	6	8	4

В результате изучения темы параграфа учащиеся

должны знать:

- особенности ветвящихся процессов;
- синтаксис и семантику условного оператора;
- понятие условия;
- правила записи условий;

должны уметь:

- определять результат фрагмента программы;
- анализировать работу фрагмента программы по трассировочной таблице;
- выявлять и исправлять ошибки в готовом фрагменте программы;
- составлять программы, используя условный оператор;
- моделировать ветвящиеся процессы в среде программирования;

получат возможность:

- использовать оператор выбора для реализации ветвлений;
- успешно решать типовые экзаменационные задачи с использованием условного оператора, составных условий и вложенных циклов;
- оценивать числовые параметры моделируемых объектов и процессов;
- интерпретировать результаты, получаемые в ходе моделирования реальных процессов;
- анализировать в процессе тестирования разработанные в среде программирования информационные модели на предмет соответствия реальному объекту.

При изучении программирования ветвящихся процессов следует обратить внимание на особенности реализации полного и неполного ветвлений, особенности синтаксиса условного оператора (параграф 12.1). При работе с примерами использовании условного оператора (таблица 12.2 параграфа 12.1) можно предложить учащимся составить соответствующие примерам блок-схемы. Особенное внимание следует уделить вложенным ветвлениям и неопределённости, возникающей, когда присутствуют как полное, так и неполное ветвление (пример 4 из таблицы 12.2 параграфа 12.1).

Работу с таблицей 12.3 (параграф 12.1) можно организовать следующим образом:

1) выделить пары учащихся;
2) каждый учащийся пары выполняет задания в тетради;
3) учащиеся в паре обмениваются тетрадями и выполняют проверку.

Обратите внимание на то, что задания 5, 6 из таблицы 12.3 представляют наиболее распространённые ошибки при использовании условного оператора.

Далее рассматривается понятие условия как логического выражения. Следует обратить внимание учащихся на то, что логическое выражение может быть простым или составным (параграф 12.2).

При знакомстве (или повторении) правил получения составных условий следует:

- обратить внимание на то, что логическая операция OR (ИЛИ) не абсолютно совпадает с действием союза «или» в обычной речи;
- необходимо остановиться на особенностях приоритета операций отношения относительно приоритета логических операций в выбранном языке программирования.

Рекомендуется задание 12.1 для освоения инструментария программирования ветвлений разобрать в классе. В нём предлагаются разные варианты реализации типовой задачи выбора одного из трёх возможных вариантов. Учащиеся должны проверить, нет ли среди приведённых вариантов ошибочных. Для этого они могут использовать тестирование программы или представление алгоритма в виде блок-схемы с последующим её анализом.

Учебник [2, 7] содержит большое количество заданий для освоения инструментария программирования ветвлений, которые можно использовать как в классе, так и для самостоятельной деятельности учащихся: задания для освоения инструментария программирования 12.1–12.22 обеспечивают базовый уровень усвоения, задания 12.23–12.27 можно рекомендовать учащимся для подготовки к государственной итоговой аттестации по информатике.

Одной из форм дополнительных заданий может быть предложение учащимся составить свои задания, например аналогичные заданиям для освоения инструментария программирования 12.20–12.22, и организовать взаимопроверку или соревнования между группами.

Работу с моделью «Поймай бабочку» (параграф 12.3) можно строить по-разному в зависимости от подготовленности учащихся, их мотивации и количества учебного времени:

- либо выполнить последовательно все этапы построения, анализа модели, компьютерного эксперимента;
- либо организовать работу с готовой программой (провести предложенные в учебнике эксперименты, предложить иные способы записи условия, составить вопросы о том, как изменится работа модели при внесении определённых изменений в программу).

Самостоятельная деятельность учащихся обеспечена заданиями по моделированию ветвящихся процессов (параграф 12.3), для выполнения которых можно рекомендовать 4 часа.

При возвращении к изучению программирования ветвлений *в 11 классе* необходимо организовать самостоятельную работу учащихся по повторению основных знаний и навыков с помощью материалов учебника (параграфы 12.1, 12.2), выполнение теста-тренажёра.

Освоение программирования ветвящихся процессов в 11 классе следует построить на выполнении моделирования (задания по моделированию ветвящихся процессов из параграфа 12.3) и проекта на обобщение знаний «Поиск числа π» (параграф 12.4).

Предлагая учащимся проект на обобщение знаний «Поиск числа π», следует учитывать, что экспериментальное значение числа π, полученное изложенным в модели способом, будет существенно отличаться от правильного. В следующей главе будет представлена другая модель, которая позволит найти число π более точно. Модель из параграфа 12.4 призвана показать, что не всегда гипотеза исследователя подтверждается экспериментально. Учащиеся должны получить опыт осмысления отрицательного результат эксперимента.

Самостоятельная деятельность учащихся обеспечивается набором тем проектов, предложенных в параграфе 12.4, выбор и выполнение которых зависит от уровня подготовленности и мотивации учащихся при изучении основ программирования (рекомендовано 2 часа).

Глава 13 «Циклические алгоритмы и модели с неизвестным числом повторений» содержит описание циклов с предусловием и постусловием, алгоритмы обработки данных на основе циклов с неизвестным числом повторений.

Изучение базируется на предметных знаниях, полученных в курсе информатики основной школы: понятиях цикла с предусловием и постусловием.

Количество часов:

Класс	Базовый вариант		Расширенный вариант	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)
10	0	0	2	2
11	2	4	2	2
Итого:	2	4	4	4

В результате изучения темы параграфа учащиеся

должны знать:

- особенности циклических процессов с неизвестным числом повторений;
- синтаксис и семантику операторов цикла с предусловием и цикла с постусловием;

должны уметь:

- определять результат фрагмента программы;
- анализировать работу фрагмента программы по трассировочной таблице;
- выявлять и исправлять ошибки в готовом фрагменте программы;
- составлять программы, используя операторы циклов с предусловием и постусловием;
- моделировать циклические процессы с неизвестным числом повторений в среде программирования;

получат возможность:

- успешно решать типовые задачи, связанные с определением результата выполнения цикла с неизвестным числом повторений;
- успешно решать типовые задачи, связанные с чтением фрагмента программы и поиском ошибки в программе с использованием цикла с неизвестным числом повторений;
- оценивать числовые параметры моделируемых объектов и процессов;
- интерпретировать результаты, получаемые в ходе моделирования реальных процессов;
- анализировать разработанные в среде программирования информационные модели на предмет соответствия реальному объекту в процессе тестирования.

Первый урок может быть организован по-разному в зависимости от того, насколько успешно учащиеся освоили в основной школе способы построения циклов с предусловием и постусловием. Если учащиеся успешно освоили тему, то можно предложить им составить сравнительную таблицу для циклов с постусловием и с предусловием, самостоятельно продумав критерии сравнения. Если учащиеся слабо подготовлены, то критерии для сравнения могут быть заданы учителем и сравнение следует выполнять в процессе обсуждения.

После сравнения циклов с постусловием и с предусловием следует предложить учащимся знакомство с синтаксисом операторов языка программирования, реализующих изучаемые циклы (табл. 13.1, 13.3 из учебника) и примеры записи операторов (табл. 13.2, 13.4 из учебника).

При работе с примерами оператора цикла с постусловием (таблица 13.4 из учебника) следует обратить внимание на последний пример. Он иллюстрирует способ организации ввода данных, обладающих необходимыми свойствами, и может быть использован учащимися в дальнейшем при составлении программ.

Примеры решения одной задачи с использованием различных циклических алгоритмов (табл. 13.5 из учебника) могу быть выполнены либо в качестве практической работы на компьютере, либо как чтение и анализ программы, в зависимости от конкретных условий. Примеры дают возможность обсудить с учащимися особенности разных циклов и их взаимозаменяемость.

Вопросы и задания, тест-тренажёр могут быть рассмотрены в классе для оценки достижения цели урока или использоваться для самостоятельной деятельности учащихся. Важно обратить внимание учащихся на задание 13.11, предлагающее найти ошибки в записи циклов с условием.

Большой набор заданий для освоения инструментария программирования может быть использован вариативно с учётом реальных возможностей учащихся и их потенциала.

Работу с моделью «От дома до школы» (параграф 13.3) следует организовать *на втором уроке* при изучении темы. Урок можно строить по-разному в зависимости от подготовленности учащихся, их мотивации и количества учебного времени: либо пройти последовательно все этапы построения и анализа модели, либо организовать работу с готовой программой: провести предложенные в учебнике эксперименты, использовать

для реализации модели другой цикл, составить вопросы о том, как изменится работа модели при внесении определённых изменений в программу.

Самостоятельная деятельность. С учётом ограниченного количества учебных часов проект на обобщение знаний «Новый способ вычисления числа π» (параграф 13.4) может быть предложен учащимся для самостоятельной деятельности. Выполнение проектов по темам из параграфа 13.4 также может обеспечить самостоятельную деятельность учащихся, для которой рекомендовано 4 часа.

В главе 14 «Алгоритмы и модели обработки символьных данных» приводятся стандартные средства обработки символьных/строковых данных, рассматриваются алгоритмы обработки символьных/строковых данных.

Изучение базируется на предметных знаниях, полученных в курсе информатики основной школы: кодировании текстовой информации, циклических алгоритмах.

Количество часов:

Класс	Базовый вариант		Расширенный вариант	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)
10	0	0	2	2
11	2	4	2	2
Итого:	2	4	4	4

В результате изучения темы параграфа учащиеся
должны знать:
- особенности символьных и строковых данных;
- синтаксис и семантику операторов обработки символьных и строковых данных;

должны уметь:
- определять результат готового фрагмента программы;
- выявлять и исправлять ошибки в готовом фрагменте программы;
- составлять программы, используя операторы обработки символьных и строковых данных;
- моделировать процессы обработки строковых данных в среде программирования;

получат возможность:

- познакомиться с алгоритмами обработки строковых данных, имеющими практическую ценность;
- интерпретировать результаты, получаемые в ходе моделирования реальных процессов;
- анализировать разработанные в среде программирования информационные модели на предмет соответствия реальному объекту в процессе тестирования.

Изучение темы стоит начать с актуализации представления учащихся о кодировании текстовой информации. Затем нужно познакомить учащихся с имеющимися в выбранном для изучения языке программирования средствами обработки символьных/строковых данных. В данной теме можно обратить внимание учащихся на различия в подходах к обработке символьных/строковых данных в двух рассматриваемых языках. Далее необходимо выполнить в классе задания 14.10–14.12 для освоения инструментария программирования.

Самостоятельная деятельность учащихся должна включать:

- подготовку устных ответов на вопросы и задания;
- выполнение заданий для освоения инструментария программирования, среди которых следует выделить задание 14.3, которое предполагает самостоятельный эксперимент учащихся.

Отметим, что набор заданий для освоения инструментария программирования таков, что позволяет обеспечить индивидуализацию обучения за счёт вариативности использования заданий.

Далее стоит выбрать для выполнения либо модели из параграфа 14.3, либо проекта из параграфа 14.4. Задача на моделирование «Испуганный НЛО» (параграф 14.3) не имеет непосредственного отношения к обработке строковой информации и хорошо подходит для учащихся, нуждающихся в дополнительной, может быть игровой мотивации к изучению программирования. Реализация данной программы может ориентировать учащихся на создание собственных проектов-игр или проектов-анимаций и тем самым вовлечь школьников осознанное изучение программирования.

Проект на обобщение знаний «Сколько шагов от ученика до гения» (параграф 14.4) знакомит учащихся с практически

значимым алгоритмом, ставит несколько интересных вопросов и открывает дорогу в решению олимпиадных и исследовательских задач.

Самостоятельная деятельность. Также в параграфе 14.3 предложены интересные задания по моделированию обработки символьных данных, в параграфе 14.4 — темы проектов, которые помогут учителю организовать самостоятельную деятельность учащихся, для которой рекомендовано 4 часа.

Глава 15 «Алгоритмы обработки структурированных типов данных» содержит определение понятия массива, описание особенностей одномерного и двумерного массивов, типовых алгоритмов обработки массивов. Также в параграфе содержится материал, посвящённый работе с файловыми типами данных.

Изучение базируется на предметных знаниях, полученных в курсе информатики основной школы: понятии массива и его описании, алгоритма заполнения и вывода массива, а также на знаниях, полученных при изучении циклических алгоритмов с известным числом повторений.

Количество часов:

Класс	Базовый уровень		Расширенный уровень	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)
10	3	4	3	3
11	0	4	3	3
Итого:	3	8	6	6

В результате изучения темы параграфа учащиеся
должны знать:
- понятие структурированных типов данных;
- описание одномерных массивов;
- описание двумерных массивов;
- алгоритм заполнения, вывода данных в массиве;
- алгоритм поиска элементов массива, удовлетворяющих условию;
- алгоритм упорядочения данных в массиве;
- описание файлового типа данных;

должны уметь:
- определять результат фрагмента программы;

- анализировать работу фрагмента программы по трассировочной таблице;
- выявлять и исправлять ошибки в готовом фрагменте программы вставки (удаления) элементов массива, перестановки элементов массива, проверки соответствия элементов массива условию;
- составлять программы поиска элементов массива, упорядочения массива;
- составлять программы, используя действия с файлами;
- моделировать процессы обработки одномерных, двумерных массивов в среде программирования;

получат возможность:

- успешно решать типовые экзаменационные задачи, связанные с обработкой структурированных данных;
- оценивать числовые параметры моделируемых объектов и процессов;
- интерпретировать результаты, получаемые в ходе моделирования реальных процессов;
- анализировать в процессе тестирования разработанные в среде программирования информационные модели на предмет соответствия реальному объекту.

Методику преподавания *первого урока* главы «Алгоритмы обработки структурированных типов данных» целесообразно выбирать в зависимости от того, насколько хорошо учащиеся освоили в курсе информатики основной школы понятие массива, способы описания массивов, алгоритмы заполнения и вывода элементов массива.

Если перечисленные базовые понятия усвоены в основной школе недостаточно успешно, то первый урок следует начать с обсуждения, которое содержит естественные для учащихся аналогии с понятием массива: примеры больших объёмов данных и практических потребностей в их обработке. В результате обсуждения учащиеся должны подойти к содержанию понятия массива как необходимости объединения данных в структуру. Примеры практических задач по обработке структурированных данных должны подвести учащихся к необходимости решения задач поиска элементов массива, удовлетворяющего определённому условию, поиска наибольшего или наименьшего элемента массива, к необходимости упорядочения (сортировки) данных.

На уроке следует:

1) рассмотреть способ описания массива (табл. 15.1 из учебника), примеры описания массивов (табл. 15.2 из учебника), примеры обращения к элементу массива (табл. 15.3 из учебника);
2) рассмотреть примеры заполнения и вывода массива (табл. 15.4 из учебника);
3) выполнить задание на поиск ошибок при описании и заполнении массива (задание 15.1 для освоения инструментария программирования);
4) рассмотреть алгоритм поиска максимального/минимального элемента массива (параграф 15.3).

Самостоятельная деятельность учащихся по результатам урока должна включать: подготовку устных ответов на вопросы и задания, выполнение заданий для освоения инструментария программирования обработки одномерных массивов 15.2–15.6. Для учащихся, выделяющихся высоким уровнем мотивации в обучении, следует предложить самостоятельно рассмотреть алгоритм сортировки (параграф 15.3) и разработать программу, реализующую алгоритм сортировки.

Если базовые понятия усвоены в основной школе успешно, то возможно предложить учащимся самостоятельную деятельность по учебнику, предшествующую уроку, по следующему плану:

1) понятие и описание массива, примеры описаний, примеры обращений к элементам массива (параграф 15.1);
2) особенности и примеры описаний одномерных массивов (параграф 15.2);
3) алгоритм поиска максимального/минимального значения одномерного массива (параграф 15.3);
4) вопросы и задания 15.1–15.14;
5) задание для освоения инструментария программирования обработки одномерных массивов 15.1;
6) составление трассировочных таблиц для программ заполнения массива значениями, вычисления суммы или среднего арифметического элементов массива (задания для освоения инструментария программирования 15.2–15.6).

Успешное освоение учебного материала в основной школе и самостоятельная деятельность учащихся позволяют на уроке разобрать выполненные задания и посвятить урок обсуждению алгоритма сортировки (параграф 15.3):

1) разобрать алгоритм на примерах (пример должен содержать небольшое количество элементов и предполагать наибольшее количество перестановок;
2) обсудить возможные критерии оценки оптимальности алгоритма сортировки;
3) составить программу, реализующую алгоритм сортировки.

Самостоятельная деятельность учащихся по результатам урока должна включать выполнение заданий для освоения инструментария программирования обработки одномерных массивов 15.7–15.9. Учащимся, заинтересованным изучением программирования, следует предложить найти/придумать другие алгоритмы сортировки.

С учётом ограниченного количества учебных часов на изучение алгоритмов обработки структурированных данных, нужно сразу готовить учащихся к тому, что индексов для определения положения элемента в массиве может быть несколько.

Второй урок изучения данной главы следует посвятить разработке модели «Средняя температура по больнице» (параграф 15.4). Разработка модели опирается на типовые алгоритмы обработки одномерных массивов, поэтому следует разобрать:

- реализацию алгоритма поиска среднего арифметического значения элементов массива;
- реализацию алгоритма поиска максимума (минимума) среди элементов массива;
- проблемы, возникающие при поиске максимума (минимума): совпадающие значения (в том числе случай совпадения всех значений в массиве), способы инициализации максимума (минимума).

Отдельно стоит коснуться вопроса об использовании генератора случайных чисел при моделировании, поговорить о том, когда это может быть оправданно, и подвести учащихся к необходимости сохранения больших объёмов данных после (до) времени работы программы, т. е. подготовиться к знакомству с файловыми типами данных.

На третьем уроке изучения данной главы необходимо сочетать знакомство с текстовыми файлами и работу с двумерными массивами. Поэтому последовательность изучения материала на уроке может быть следующей:

1) обсуждение описания двумерного массива (параграф 15.5);
2) рассмотрение примеров описания двумерного массива (параграф 15.5);

3) выполнение заданий на поиск грубых ошибок в описании двумерного массива (параграф 15.5);
4) обратить внимание учащихся на неудобство ввода большого количества данных, особенно если требуется многократный ввод при отладке или тестировании, т. е. на необходимость однократного ввода данных для программы и сохранения их в файле;
5) рассмотрение примеров создания файла, заполненного данными, и применения этого файла в программе (параграф 15.7).

Для учащихся с низким уровнем подготовки и низкой мотивацией к обучению можно ограничиться только разбором примера работы с текстовым файлом, в качестве самостоятельной работы предложить ответить на вопросы и задания (параграф 15.7) и выполнить задания для освоения инструментария программирования (параграф 15.7).

Для учащихся с высоким уровнем подготовки и/или мотивации стоит на основе этого примера работы с текстовым файлом начать работу по реализации алгоритмов работы с двумерными массивами и выбрать для самостоятельной деятельности задания из рубрики «Задания для освоения инструментария программирования. Обработка двумерных массивов». Следует заранее спланировать набор заданий, которые могут быть успешно решены в классе, с учётом фактического уровня знаний и подготовленности учащихся, и заданий для домашней работы.

Самостоятельная деятельность. Проект на обобщение знаний «Доска Гальтона» (параграф 15.6) служит хорошей иллюстрацией использования массива для накопления информации об эксперименте (в данном случае — количестве шариков, упавших в каждый из жёлобов), поэтому целесообразно организовать ознакомление заинтересованных учащихся с данным материалом. Для этого возможно предложить самостоятельную деятельность учащихся на основе данного примера и тем проектов из параграфа 15.7 (рекомендовано 4 часа).

Глава 16 «Структурное программирование» предлагает описание принципов структурного программирования. В параграфе вводятся понятия локальных и глобальных переменных, подпрограмм (процедур и функций), формальных и фактических параметров, обсуждается использование возвращаемых и невозвращаемых параметров, вводится понятие рекурсивного алгоритма, рассматриваются проблемы и достоинства рекурсии.

Изучение базируется на предметных знаниях, полученных учащимися при изучении элементов программирования на основании всего предыдущего материала учебника [2, 7].

Количество часов:

Класс	Базовый вариант		Расширенный вариант	
	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)	Урочная деятельность, ч	Самостоятельная деятельность, ч (рекомендации)
10	1	4	2	2
11	1	4	2	2
Итого:	2	8	4	4

В результате изучения темы параграфа учащиеся

должны знать:

- понятие подпрограммы;
- понятие глобальных и локальных переменных;
- принципы структурного программирования;
- синтаксис и семантику операторов, позволяющих реализовать подпрограммы в виде процедур, функций;

должны уметь:

- определять результат фрагмента программы;
- анализировать работу фрагмента программы по трассировочной таблице;
- выявлять и исправлять ошибки в готовом фрагменте программы;
- составлять программы с помощью подпрограмм;
- моделировать информационные процессы, используя принципы структурного программирования;

получат возможность:

- реализовывать сложные алгоритмы, используя принципы структурного программирования;
- познакомиться с понятием рекурсии и примером её использования для построения элементов фрактального изображениями, а также с проблемами, возникающими при создании рекурсивных алгоритмов;
- оценивать числовые параметры моделируемых объектов и процессов;
- интерпретировать результаты, получаемые в ходе моделирования реальных процессов;
- анализировать в процессе тестирования разработанные в среде программирования информационные модели на предмет соответствия реальному объекту.

К моменту изучения данной главы у учащихся должен быть накоплен значительный опыт «хаотичного» программирования, поэтому обсуждение путей получения упорядоченного, легко читаемого и исправляемого кода должно найти отклик аудитории. Знакомство с технологией структурного программирования следует начать с обращения к опыту учащихся, обсуждения ситуаций, когда возникает необходимость использовать свои или чужие программы. Можно предложить учащимся заранее изучить историю вопроса и подготовить небольшое сообщение, только нужно помнить об очень маленьком запасе времени и не увлекаться обсуждением.

Основное время *урока в 10 классе* следует уделить рассмотрению алгоритма вычисления НОД трёх чисел (параграф 16.3). Выбор этого примера основан на готовности учащихся применить известные им знания. Алгоритм вычисления НОД двух чисел знаком учащимся из курса математики, блок-схема данного алгоритма рассмотрена при изучении главы 8. Применение известного алгоритма для расширения круга решаемых задач, а именно для вычисления НОД трёх чисел, наглядно иллюстрирует основные принципы структурного программирования (параграф 16.2), области применения локальных и глобальных переменных (параграф 16.3).

В условиях ограниченного времени на изучение темы синтаксис описания подпрограмм можно изучать на примерах. Для этого учащимся необходимо предложить найти особенности в записи процедур и функций в вопросах и заданиях 16.19–16.23. На основе анализа заданий нужно сформулировать вывод об основной цели использования подпрограмм с уточнением различий между процедурами и функциями. Следует отдельно обратить внимание на передачу массивов в качестве параметров (табл. 16.1 из учебника).

Для закрепления теоретических знаний учащимся необходимо подготовить устные ответы на вопросы и задания. Для закрепления практических навыков необбходимо выполнить задания для освоения инструментария программирования:

- базовым набором заданий являются:
  - задания для освоения технологии программирования 16.1–16.7;
  - задание 16.9 (поиск НОК на основании известного НОД) в форме блок-схемы алгоритма;
- для заинтересованных учащихся:
  - предлагаются задания для освоения инструментария программирования 16.1–16.12;

– задания следует выполнять на языке программирования.

Вариативность использования вопросов и заданий позволяет обеспечить индивидуальный подход к учащимся с различным уровнем мотивации изучения элементов программирования.

Самостоятельная деятельность учащихся обеспечена заданиями для освоения инструментария программирования (параграф 16.4). Самостоятельная деятельность учащихся может быть построена с учётом подготовленности учащихся и их мотивации для изучения программирования, для её выполнения можно рекомендовать 4 часа.

Освоение структурного программирования *в 11 классе* следует построить на основе проекта на обобщение знаний «То берёзка, то рябина…» (параграф 16.5), который позволяет повторить и реализовать на практике принципы структурного программирования, повторить понятия локальных и глобальных данных, подпрограмм (процедур и функций), формальных и фактических параметров, ввести понятие рекурсии, обратить внимание на вопрос выделения памяти при использовании подпрограмм. Этот проект обладает огромным потенциалом:

- может быть интересен слабо мотивированным учащимся, так как результат нагляден и позволяет легко выполнять компьютерный эксперимент;
- предоставляет материал для знакомства с понятием рекурсии, обсуждения проблем и достоинств рекурсии и проведения ряда экспериментов;
- содержит материал для обсуждения дополнительных вопросов, например фрактальной графики, и для разработки учащимися собственных проектов и/или исследовательских работ, в том числе в рамках внеурочной деятельности.

С целью экономии времени урока можно предложить учащимся разделить между собой элементы пейзажа: кто-то работает с ёлочкой, кто-то — со снежинкой и т. п. Затем выдать домашнее задание: объединение различных подпрограмм, выполнение экспериментов и представление индивидуального результата моделирования. Раздельная работа над отдельными модулями программы с последующим их объединением отлично вписывается в контекст обсуждения принципов структурного программирования, достоинства такой работы и проблемы, возникающие в её ходе, следует обсудить с учащимися.

Для мотивированных учащихся, чей интерес выходит за рамки базового уровня изучения информатики, можно предложить следующие эксперименты:

- наблюдение за временем выполнения программы при изменении уровня вложенности рекурсии;
- наблюдение за выделением памяти.

Хорошим поисковым заданием может стать выяснение с помощью справочной системы встроенных возможностей языка (библиотек) для измерения потраченного времени, свободной памяти. Можно предложить групповую работу, когда один участник выясняет способ измерения времени, другой — свободной памяти, кто-то проводит эксперименты с измерением времени, кто-то оформляет в электронных таблицах результаты эксперимента (с построением диаграммы зависимости). В программе, разобранной в проекте, есть три объекта, использующие различные уровни вложенности рекурсии — снежинка (три поколения потомков), ёлочка (четыре поколения потомков), дерево (семь поколений потомков). Также каждый объект состоит из различного количества рекурсивных элементов: снежинка из шести, ёлочка из трех уровней по пять элементов, дерево — это единственный рекурсивный элемент. Получается, что для проведения эксперимента необходимо выделить из проекта один объект.

Результатом эксперимента может быть таблица наблюдений (табл. 10.4).

Таблица 10.4

Время выполнения программы и объём выделяемой памяти

Объект	**Снежинка**		
Количество разрешённых поколений потомков	**Время выполнения программы**	**Доступная память в начале выполнения**	**Доступная память в момент вызова последнего потомка**
…	…	…	…

Самостоятельная деятельность. Предложенные темы проектов 16.1 «Город», 16.2 «Плакат», 16.3 «Фракталы» могут быть предложены учащимся для самостоятельной деятельности. Работа над ними позволит компенсировать недостаточность выделенного количества учебных часов для изучения темы и обеспечит эффективность обучения учащихся с высоким уровнем мотива изучения программирования. Для выполнения самостоятельной проектной деятельности рекомендовано 4 часа.