Оглавление

Введение.

Современные образовательные стандарты предполагают смещение целеполагания учебного процесса в сторону развития когнитивных способностей обучаемых, в частности, если рассматривать обучение информатике – то основной целью будет являться развитие алгоритмического стиля мышления.

Формирование алгоритмического мышления начинается в возрасте 5-6 лет и продолжается в школьном возрасте, при изучении школьного курса информатики и других естественнонаучных и точных дисциплин. В значительной степени формирование и развитие алгоритмического мышления происходит именно на уроках информатики и особенно при изучении темы «Алгоритмизация и программирование».

Современные задачи обработки информации, новые аппаратные и сетевые решения вычислительной техники делают необходимым изучение основных парадигм и технологий программирования, сложившихся в современной науке – не только императивной, но и декларативных – объектно-ориентированной, функциональной, логической, а также параллельного программирования. Возникшее с появлением глобальной сети Internet Web-программирование и языки, его реализующие, определяют формирование современной, сценарной технологии программирования

Каждая парадигма программирования предполагает формирование определенного стиля мышления – объектного, функционального, логического, параллельного, сценарного.

Поскольку Internet достаточно глубоко и прочно вошел в нашу повседневную жизнь, можно сказать, что сценарный стиль мышления на начальном, повседневном уровне сформирован у современных школьников, являющихся активными пользователями сервисов и услуг, реализованных в глобальной сети. Одной из основных целей обучения теме «Коммуникационные технологии и разработка Web-сайтов» с начала 2000-х годов включенной в школьный курс информатики, на наш взгляд является

дальнейшее развитие сценарного стиля мышления в частности и, как следствие, алгоритмического мышления обучающихся в целом.

Противоречия:

1.Между требованиями современного информационного общества к уровню развития алгоритмического мышления в целом и сценарного мышления как его составляющей части выпускников школы и недостатком исследований в области теории АМ

2.Между растущими потребностями современных школьников и содержанием ШКИ

Проблема:

Какие методические приемы необходимо использовать при обучении теме: «Коммуникационные технологии и разработка Web-сайтов», чтобы сформировать сценарное мышление у выпускников школы на достаточном уровне?

Цель исследования:

Исследовать процесс развития сценарного мышления при обучении теме «Коммуникационные технологии и разработка Web-сайтов» в ШКИ, определить условия его развития и способы диагностики

Объект исследования:

Процесс обучения теме «Коммуникационные технологии и разработка Web-сайтов» в ШКИ

Предмет исследования:

Развитие сценарного мышления в процессе обучения теме «Коммуникационные технологии и разработка Web-сайтов» в ШКИ

Гипотеза:

Достаточный уровень сценарного мышления учащихся 8 классов будет сформирован, если при обучении теме: «Коммуникационные технологии и разработка Web-сайтов»:

- сдвинуть целеполагание обучения теме с предметных результатов на личностные, нацелить его на развитие сценарного мышления;

- уточнить понятие сценарного мышления, построить его структурную модель, выделить уровни сформированности;
- предложить диагностику уровней сформированоости сценарного мышления;
- включить в ШКИ наряду с изучением HTML элементы языка сценарного программирования PHP;
- использовать в учебном процессе учебные элементы по всем составляющим темы.

Задачи:

1)Выявить особенности обучения теме «Коммуникационные технологии и разработка Web-сайтов» в ШКИ

2) Уточнить понятие «сценарный стиль мышления» как одной из составляющих алгоритмического мышления, построить его структурную модель

3) Проанализировать процесс формирования и развития сценарного стиля мышления, рассмотреть средства и методы его развития и определить способы его диагностики

4) Разработать учебные элементы по теме «Коммуникационные технологии и разработка Web-сайтов» (по HTML И PHP), провести апробацию разработанных материалов

Основная эмпирическая база:

Результаты были апробированы во время деятельности учителя в г. Красноярске МАОУ «Гимназия №5» на 8 классах.

Ключевые слова:

методика обучения информатике, сценарная парадигма программирования, теория мышления, алгоритмическое мышление, сценарное мышление, учебные элементы.

Глава 1. Теоретические основы процесса развития сценарного мышления при обучении теме «Коммуникационные технологии и разработка Web-сайтов» в ШКИ

§ 1.1. Уточнение понятия «сценарный стиль мышления»

Парадигмы программирования

В настоящее время сложилось несколько взглядов на классификацию парадигм программирования. Проводя их анализ, можно выделить два основных подхода: укрупненный и раздробленный.

В укрупненном подходе выделяют две основные парадигмы: императивную и декларативную (Рис.1); все остальное относятся к методам, приемам и технологиям программирования

Рисунок 1. Классификация парадигм программирования.

В раздробленном подходе каждый метод программирования, каждая технология программирования выделяются в отдельную парадигму:

- Императивная парадигма;
- Декларативная парадигма;
- Функциональное программирование;
- Логическое программирование;
- Объектно-ориентированное программирование;
- Визуальное программирование;
- Параллельное программирование;
- Сценарная парадигма программирования.

Подробно рассмотрим понятие парадигма программирования и кратко охарактеризуем каждую парадигму и технологию программирования.

Парадигма программирования — это совокупность идей и понятий, определяющих стиль написания программ. Парадигма, в первую очередь, определяется базовой программной единицей и самим принципом достижения модульности программы. В качестве этой единицы выступают машинный код (машинно-ориентированная парадигма), вычисляемое выражение (функциональное программирование), действие (императивное программирование), правило (логическое программирование), объект (объектно-ориентированное программирование) и т.д.

Парадигма программирования определяет, какие термины лежат в основе написания программы. В императивном программировании программа описывается как последовательность действий, в функциональном программировании представляется в виде выражения и множества определений функций, в логическом — это набор фактов (аксиом), правил (механизмов вывода нового знания), целей и подцелей (теорем, требующих доказательства или опровержения). В объектно-ориентированном программировании программа рассматривается как набор взаимодействующих объектов. Парадигма программирования не определяется однозначно языком программирования — многие современные языки программирования являются мультипарадигмальными, т. е. допускают использование различных парадигм. Например, Паскаль — это язык структурного, процедурного, модульного, объектно-ориентированного программирования, Фортран — язык процедурного, модульного, параллельного программирования.

Развитие языков программирования с точки зрения различных парадигм и технологий представлено на рисунке 2.

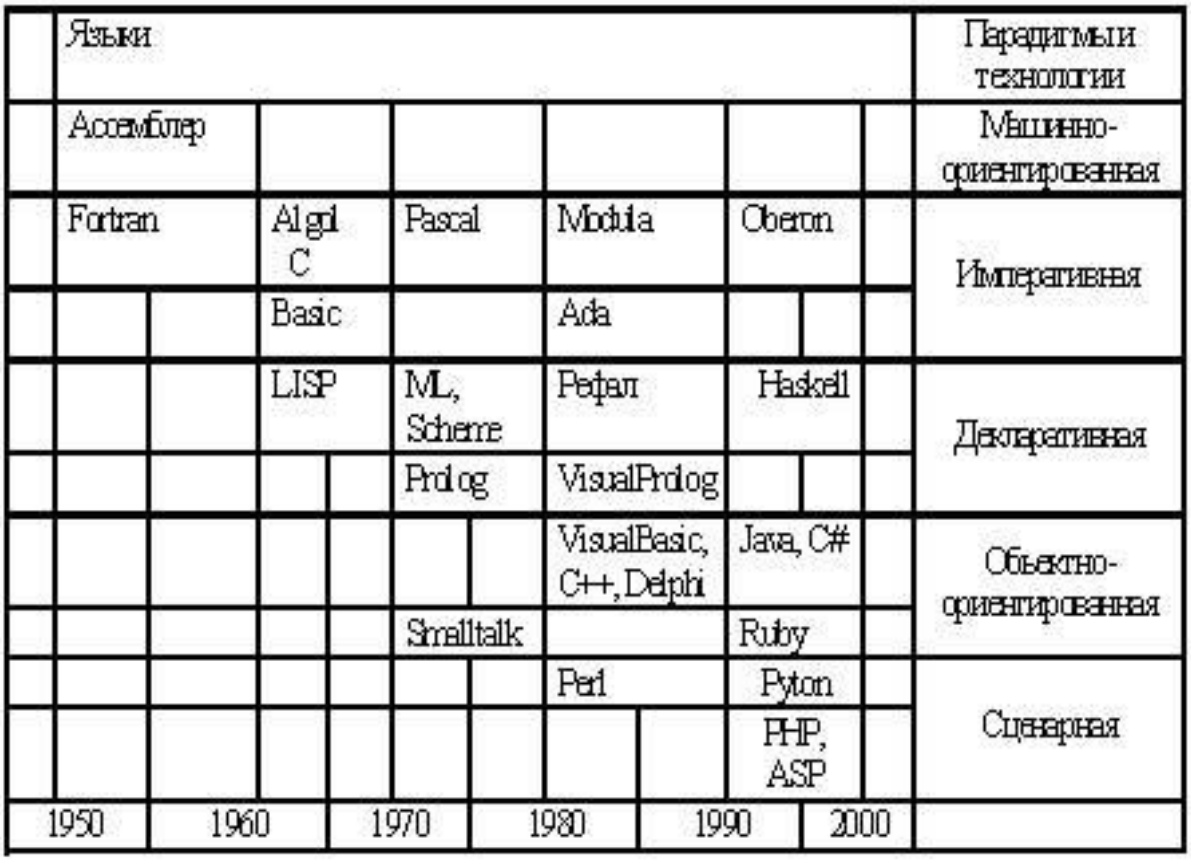


Рисунок 2 Развитие языков программирования

Идеи и основополагающие понятия основных парадигм

На начальном историческом этапе возникло *машинно-ориентированное программирование*, являющееся допарадигмальным.

На заре компьютерной эры электронные вычислительные машины программировались только на языках машинного уровня. Процессору посылались бинарные коды из системы команд данного процессора, как правило, вместе с данными для обработки. Обычно речь шла об операциях по перемещению данных из памяти в регистр или о простой арифметике над содержимым регистра.

На смену языкам машинного уровня очень быстро пришли языки ассемблера, в которых операции формулировались читаемым открытым текстом, но все еще были ориентированы на регистр. Шестнадцатеричные

машинные коды заменялись простыми, по возможности содержательными сокращениями, так называемыми ***мнемониками.***

Программа на языке ассемблера переводилась на машинный язык с помощью транслятора, часто тоже называемого Ассемблером.

Машинно-ориентированное программирование характеризуется аппаратным подходом к организации работы компьютера, нацеленным на доступ к любым возможностям оборудования. В центре внимания – конфигурация оборудования, состояние памяти, команды, передачи управления, очередность событий, исключения и неожиданности, время реакции устройств и успешность реагирования.

Машинно-ориентированные языки называют также языками низкого уровня. В дальнейшем для облегчения труда программистов были созданы языки программирования, которые строились на основе определенного алфавита и строгих правил построения предложений. Отличительной особенностью этих языков является их ориентация не на систему команд той или иной ЭВМ, а на систему операторов, характерных для записи определенного класса алгоритмов. Такие языки принято называть языками программирования высокого уровня. К ним, в первую очередь, относят: Fortran, PL, Algol, C, Basic. С возникновением языков высокого уровня формируется первая парадигма программирования – императивная.

Императивная парадигма. Императивное программирование — один из наиболее естественных для человека подходов к написанию программ. При написании подобной программы необходимо найти такую цепочку команд, которая приведет к вычислению одной или нескольких искомых величин. Эти цепочки команд — директивы — совершенно однозначно и четко предписывают выполнение каждого шага алгоритма (отсюда синоним императивной парадигмы — директивная).

Наиболее популярными императивными языками являются язык Pascal, созданный в целях обучения программированию, и язык C, на котором в основном работают профессиональные программисты.

Декларативная парадигма. Главное различие между императивным и декларативным программированием заключается в том, что декларативная программа заявляет (декларирует), что должно быть достигнуто в качестве цели, а императивная предписывает, как ее достичь.

Декларативные программы не предписывают выполнять определенную последовательность действий, в них лишь дается разрешение совершать их. Исполнитель должен сам найти способ достижения поставленной перед ним составителем программы (программистом) цели, причем зачастую это можно сделать различными способами.

Наиболее существенными классами декларативных языков являются функциональные (functional) языки, например Lisp (Лисп), и логические (logic) языки, например, Prolog (Пролог).

Функциональное программирование. Функциональная программа состоит из совокупности определений функций, которые, в свою очередь, представляют собой вызовы других функций и предложений, управляющих последовательностью вызовов. При этом функции часто либо прямо, либо опосредованно вызывают сами себя (рекурсия). Каждая функция возвращает некоторое значение в вызвавшую его функцию, вычисление которой после этого продолжается. Этот процесс повторяется до тех пор, пока начавшая процесс вычислений функция не вернет конечный результат пользователю.

«Чистое» функциональное программирование не содержит оператора присваивания, в нем вычисление любой функции не приводит ни к каким побочным эффектам, отличным от собственно вычисления ее значения. Отсутствие оператора присваивания делает переменные, используемые в функциональных языках программирования, очень похожими на переменные в математике — получив однажды свои значения, они больше никогда их не меняют.

Логическое программирование. Еще одной реализацией декларативного стиля является логическое программирование, основанное на логике предикатов. В логическом программировании основное внимание уделяется

описанию структуры прикладной задачи, а не выработке предписаний компьютеру, что ему следует делать. Программа рассматривается как набор логических фактов и правил вывода, а выполнение программы заключается в вычислении истинности (попытке доказательства) некоторого утверждения. Prolog (от фр. PROgrammation LOGique) — наиболее известный язык логического программирования. Его (наряду с функциональным языком Lisp) часто называют языком искусственного интеллекта С его помощью решаются задачи создания экспертных систем и систем обработки естественных языков.

Объектно-ориентированное программирование. Объектно-ориентированное программирование (ООП) по сути является императивным программированием, дополненным принципом инкапсуляции данных и методов в объект (принцип модульности) и наследованием (принципом повторного использования разработанного функционала).

В основе ООП лежат три основных понятия:

- инкапсуляция (сокрытие данных в классе или методе);
- наследование;
- полиморфизм.

Объектно-ориентированное программирование — очень мощный метод при разработке больших программ. При использовании ООП резко возрастает КПД работы программиста и снижается количество ошибок. Для большинства языков были созданы большие библиотеки готовых объектов. Практически все современные языки программирования, независимо от принадлежности к тому или иному стилю (императивному или декларативному), поддерживают концепцию ООП. Среди них Delphi, C++, Java, Ruby и Haskell. Существуют и версии объектно-ориентированного Пролога (например, Visual Prolog)

Визуальное программирование. Визуальное программирование основано на ООП и возникло вслед за возникновением и распространением графического интерфейса операционных систем. Основополагающая идея визуального программирования заключается в перетаскивании объектов

мышкой из библиотеки (хранилища объектов) в нужное место программы. А система сама должна написать нужный для этого код. Так работают Delphi, C++ Builder, Visual C, Visual Basic и др.

Параллельное программирование. В параллельном программировании программа порождает совокупность параллельно протекающих процессов обработки информации, полностью независимых или связанных между собой статическими или динамическими пространственно-временными или причинно- следственными отношениями.

Наиболее известные технологии, позволяющие разрабатывать параллельные программы : OpenMP, стандарт которой был разработан для языков Fortran, C и C++ ; система параллельного программи- рования PVM (Parallel Virtual Machine), позволяющая объединить набор разных компьютеров, связанных сетью, в общую вычислительную систему, называемую параллельной виртуальной машиной.[19]

Сценарное программирование. Возникшее с появлением глобальной сети Интернет, Web- программирование и языки, его реализующие, определяют формирование современной, сценарной, технологии программирования.

Сценарные языки. Сценарные языки, или языки скриптов (scripting languages), за последние годы сделали огромный шаг вперед. Еще лет десять назад им отводилась роль вспомогательных средств, сейчас же скепсис по отношению к ним сменился интересом и признанием.

Сценарные языки имеют достаточно длительную историю развития. Концепция сценарного программирования явилась как естественное развитие языка LISP. К первым сценарным языкам относят встроенные средства управления командной оболочки операционной системой. Командный файл на языке операционной системы, представляет собой управляющий сценарий, который выполняет заданную последовательность действий. Можно сказать, что сценарий «склеивает» различные части операционной системы и осуществляет их взаимодействие.

В настоящее время популярность сценарных языков связана с развитием Internet-технологий. Скриптовые языки используются для создания динамических, интерактивных web-станиц, содержание которых модифицируется в зависимости от действий пользователя и состояния остальных страниц и данных.

Отличительной особенностью скриптовых языков является формирование программы на некотором внешнем языке как результата выполнения сценария. Сценарный язык в малой степени опирается на создание конечного продукта с нуля и в большей степени – на использование тех мощностей, которыми обладает операционная система, графическая среда, прикладная сервисная машина и прочие подобные компоненты, взаимодействие которых осуществляется с помощью сценариев.

Сценарная парадигма предполагает разбиение задачи на отдельные части, каждая из которых решается специализированными программными средствами, сценарий выступает в роли «диспетчера», ответственного за организацию их взаимодействия.

Сценарные языки для web-разработки в основном созданы в 90-е годы XX века и включают в себя элементы различных парадигм программирования от императивной до объектно-ориентированной. Среди наиболее мощных и популярных скриптовых систем можно отметить следующие: Perl, Pyton, PHP, ASP. Синтаксис и семантика различных сценарных языков весьма похожи. Это обусловлено значительным влиянием языков C и C++ на сообщество программистов. Поддержка рекурсии в сценарных языках реализована аналогично императивной и объектно-ориентированной парадигмам.[28]

Сценарная парадигма для своего существования должна обладать базовыми принципами и превосходить их, т. к. это новый уровень программирования.

Принцип эквивалентности.

Любая задача, которую можно решить с помощью сценарной парадигмы программирования в иерархии из рисунка, должна быть решаемой с помощью остальных парадигм (приложение 3).

Весь код должен преобразовываться в машинный, чтобы для каждой программы в сценарной парадигме с высоким уровнем абстракции существовала эквивалентная программа на машинном коде.

Поскольку любую программу, составимую в машинном коде, также можно составить на ассемблере или процедурном языке, то можно утверждать эквивалентность всех программ в иерархии.

Парадигмы в приведенной иерархии имеют слабую эквивалентность, то есть хотя они все позволяют вычислить одну и ту же функцию, делать это они могут по-разному.

Принцип превосходства.

Так как сценарная парадигма является новой в иерархии, она должна превосходить предшественниц и предоставлять новые механизмы абстрагирования и программирования, которые удовлетворяют больше потребностей, чем предыдущие. Проиллюстрируем этот принцип примерами:

- управляющие структуры процедурного программирования превосходят регистры, команды адресации, флаги и всевозможные переходы, присущие программированию на ассемблере;
- процедуры и модули в процедурном программировании превосходят блоки кода в ассемблере;
- классы в объектно-ориентированном программировании превзошли структуры данных и процедуры, характерные для процедурного программирования; наследование и полиморфизм в ООП превзошли наборы процедур и механизмы вызова, свойственные процедурному программированию;
- аспекты АОП превзошли разбросанный и запутанный код ООП; точки соединения (join point) и срезы (pointcut) в АОП абстрагируют

статическое и динамическое связывание, а также другие особенности ООП.

- Сценарная парадигма программирования превосходит предшественниц, она предназначено для иных задач, поэтому фундаментально отличается от них. Эта парадигма проектировались с расчетом на построение структур данных и алгоритмов, начиная с самых примитивных компьютерных компонентов, таких как слова памяти. Языки сценариев создавались для "склеивания" мощных готовых компонентов в предположении, что большинство из них уже существует и надо лишь связать их между собой

Сценарная парадигма программирования должна занимать место в иерархии, т. к. ее уровень превосходствует над уровнями существующих парадигм программирования.

Принцип связанных абстракций.

Очередной механизм абстрагирования или программирования, предоставляемый сценарной парадигмой, должен быть абстракцией одного или более механизмов непосредственной предшественницы этой парадигмы, т.е. абстракция данных.

Абстракция данных - это придание объекту характеристик, которые чётко определяют его концептуальные границы, отличая от всех других объектов. Основная идея состоит в том, чтобы отделить способ использования составных объектов данных от деталей их реализации в виде более простых объектов, подобно тому, как функциональная абстракция разделяет способ использования функции и деталей её реализации в терминах более примитивных функций, таким образом, данные обрабатываются функцией высокого уровня с помощью вызова функций низкого уровня.[3]

Принцип ограниченной видимости.

Сценарная парадигма программирования создается ради выполнения определенных потребностей, но круг этих потребностей ограничен

свойственным авторам парадигмы пониманием текущих и будущих требований моделирования и написания кода. Предусмотреть можно далеко не все, поэтому слабости и недостатки различных методов обычно открываются лишь в повседневном использовании. Данный принцип иллюстрируют следующие примеры:

- структурное программирование, лишенное переходов, было изобретено, чтобы удовлетворить потребность в интеллектуальном и семантическом контроле над управляющими структурами (ассемблер не давал возможности ощутить преимущества программирования без Goto, инкапсуляции и процедурных абстракций);
- ограничения процедурного программирования были связаны с невозможностью признать классы, наследование и полиморфизм в качестве более удачных механизмов абстрагирования;
- объектно-ориентированному программированию были свойственны ограничения в связи с неспособностью осознать преимущества инкапсуляции сквозной функциональности.

Сценарному программированию свойственно написание программного кода при помощи скриптов, что облегчает работу программисту при создании сайтов.

Данный принцип указывает на то, что для разработчиков языков нужна способность прогнозировать будущие требования программной инженерии.

Принцип порога полезности.

У скриптовой парадигмы есть свой порог полезности, определенный типом задач, которые она позволяет решить. Этот порог достигается, когда задачи, для которых не подходят абстракции конкретной парадигмы, становятся серьезным препятствием к ее использованию.

Этот принцип отвечает на два важных вопроса: как узнать, когда нужна новая, более мощная парадигма программирования, и каковы задачи, трудности и зоны ответственности, для которых конкретная парадигма не подходит? Следующие примеры иллюстрируют этот принцип:

- порог полезности для программирования на ассемблере был достигнут, когда на первый план вышли потребности переносимости, понятности и сопровождаемости кода;
- порог полезности процедурного программирования был достигнут, когда стала первоочередной потребность в уменьшении семантического пробела между задачами реального мира и программными моделями наряду с потребностью в инкапсуляции данных, а также в более развитых абстракциях для управления данными и алгоритмами;
- порог полезности ООП был достигнут, когда стала жизненно важной потребность в инкапсуляции и управлении запутанным и разбросанным кодом.
- порог полезности сценарной парадигмы программирования заключается в том, что типизация позволяет выявлять ошибки в процессе компиляции, так что в проверках во время исполнения уже нет нужды. Однако платой за эффективность являются жесткие ограничения, которые приводят к усложнению кода и снижению гибкости программ.

Этот принцип не означает, что язык уже нельзя использовать после достижения им порога полезности: ряд языков (например, Кобол) прошли свои пороги, но по-прежнему применяются. Однако данный принцип указывает на маловероятность того, что новые программные проекты будут разрабатываться на каком-либо языке, уже прошедшем свой порог полезности.

Принцип продолжения срока жизни.

В связи с непрерывным ростом производительности компьютеров, появление новых языков сценариев, растущая значимость графических интерфейсов пользователя и компонентных архитектур и, наконец, экспансия Internet чрезвычайно расширили сферу применения языков сценариев. Эти тенденции останутся в силе в течение следующего десятилетия, и соответственно будет расти число приложений, написанных полностью на

языках сценариев, в то время как языки программирования систем станут использоваться в основном для создания компонентов. Тем самым сценарная парадигма превосходит свой срок жизни, по сравнению с другими парадигмами.

Принцип неизбежной сложности.

В сценарной парадигме программирования приводят к сложности жесткие ограничения, которые приводят к усложнению кода и снижению гибкости программ [17, 24].

Структура алгоритмического мышления

Изучение языка программирования относящегося к другой парадигме, вызывает определенный ряд сложностей. Так как при переходе к программированию методами, которые относятся к другой парадигме, необходимо изменить не только подход к решению поставленной задачи, но и перестроить мыслительную деятельность относительно новой парадигмы. Каждая парадигма программирования предполагает формирование определенного стиля мышления – объектного, функционального, логического, параллельного, сценарного.

Структурная схема алгоритмического мышления представлена на Рисунке 3.

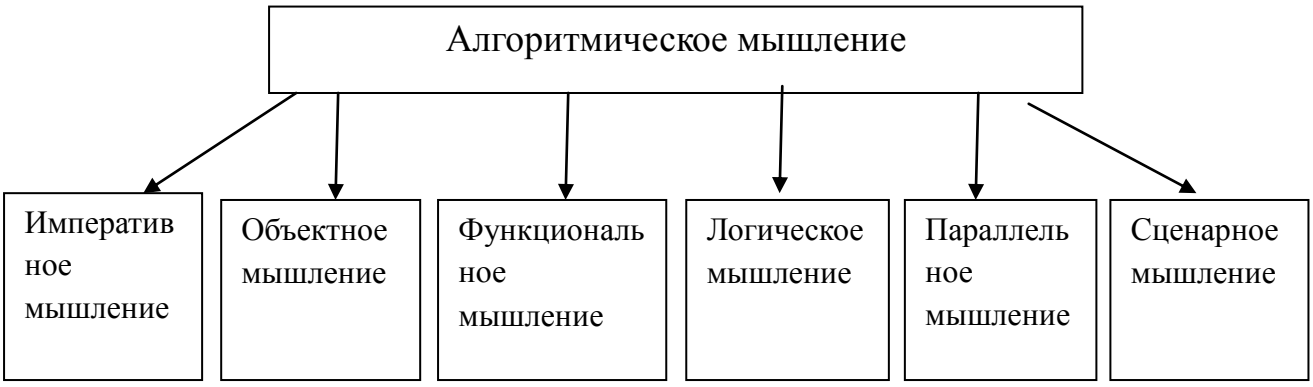


Рисунок 3. Структура алгоритмического мышления

Сценарное мышление как составляющая алгоритмического мышления возникло с появлением, развитием и широким использование в повседневной и профессиональной деятельности Internet-технологий.

Сценарное мышление, развивающееся в процессе изучения web-программирования, предполагает способность осознать, что программа формируется на некотором внешнем языке как результат выполнения сценария. Программируя на сценарном языке, следует понимать, что

- Сценарный язык в малой степени опирается на создание конечного продукта с нуля и в большей степени – на использование тех мощностей, которыми обладает операционная система, графическая среда, прикладная сервисная машина и прочие подобные компоненты, взаимодействие которых осуществляется с помощью сценариев.
- Сценарная парадигма предполагает разбиение задачи на отдельные части, каждая из которых решается специализированными программными средствами, сценарий выступает в роли «диспетчера», ответственного за организацию их взаимодействия
- Работая в рамках сценарной парадигмы, занимаясь Web-программированием, программист, в отличие от программиста эпохи «до Интернета» обязан предусматривать поведение в сети всех тех пользователей, которые будут обращаться к его сайтам. Программист же «эпохи до-Интернета» сам задавал поведение пользователей его программ

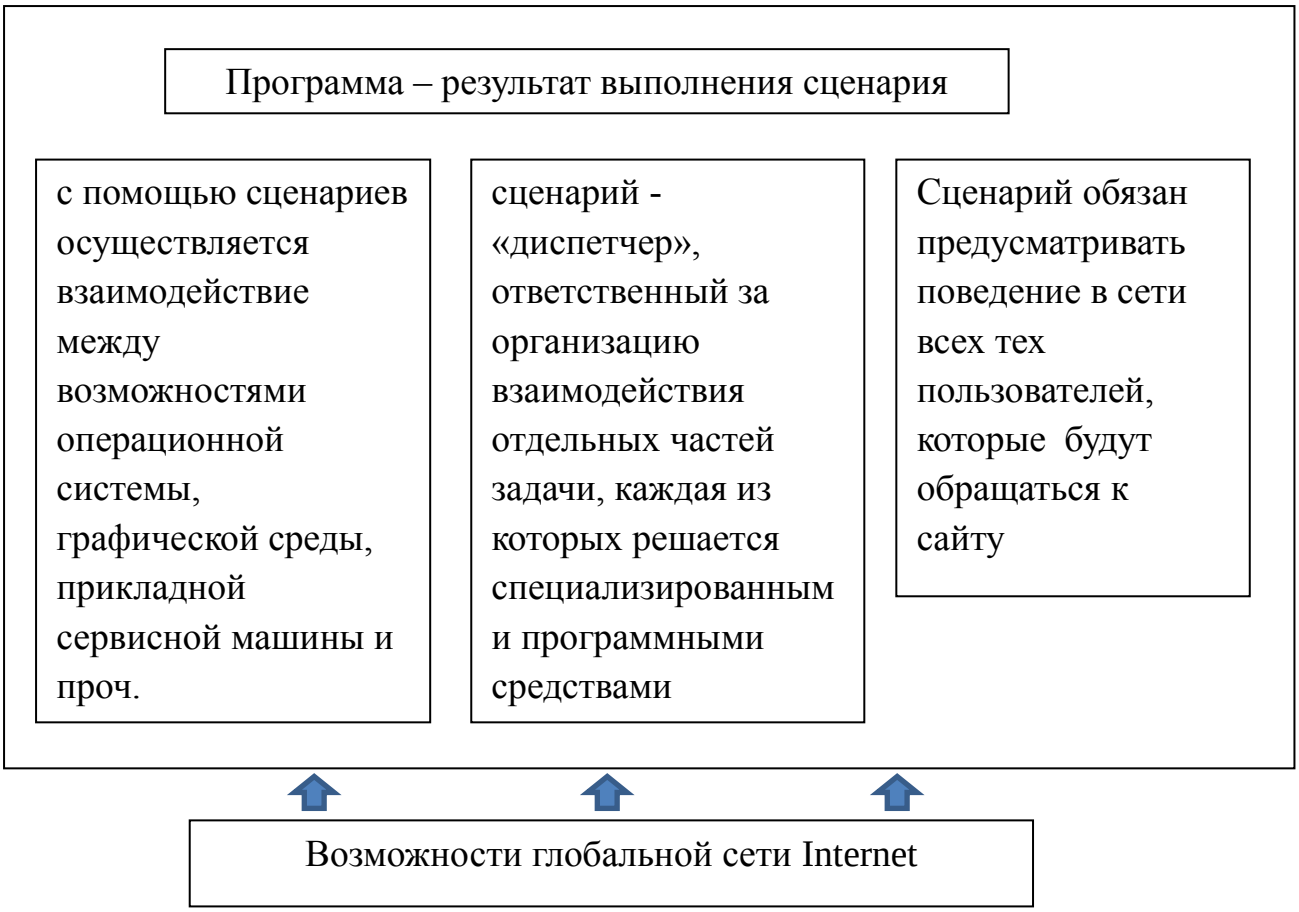


Рисунок 4. Структурная модель сценарного мышления

Уровнем развития сценарного мышления определяется поведение субъекта в сети.

В этой связи можно выделить следующие уровни развития сценарного мышления:

Первоначальный уровень - способность ориентироваться в сети, использовать сервисы и услуги, предоставляемые интернетом в повседневной деятельности

Необходимый для современного человека уровень – способность СВОБОДНО ориентироваться в сети ГРАМОТНО использовать Internet-технологии в профессиональной и учебной деятельности.

Продвинутый уровень – иметь представление о способах, методах и средствах создания web-сайтов, их конфигурации, владение навыками создания несложных, статичных Web-страничек с помощью, допустим языка

разметки гипертекстов или несложных средств создания web-страниц (FrontPage, DreamWeaver и т.п.)

Профессиональный уровень – понимание принципов сценарной парадигмы программирования, владение навыками web-программирования, создания полноценных, динамических, интерактивных web-сайтов, при помощи сценарных, скриптовых языков программирования и современных информационных систем CMS (Joomla, Drupal, WordPress и др.)

Первые два уровня определяются умением пользоваться интернет-технологиями, знать ЧТО, какие возможности предоставляются в глобальной сети. Третий уровень определяется уже осознанием того КАК работают интернет-технологии, четвертый – способностью САМОМУ разрабатывать web-сайты.

На наш взгляд, первоначальный уровень в современном мире формируется уже у дошкольников и младших школьников. Так как они уже в младшем возрасте знакомятся с коммуникационными технологиями (телефон, персональный компьютер) и уже пользуются и ориентируются в сети интернет.

Цель школьного курс информатики – как раз сформировать сценарное мышление у выпускников средней школы уже на необходимом, и, более того, на продвинутом уровне, и, возможно, в плане профориентации, определении выбора будущей профессиональной деятельности начать его формирование на профессиональном уровне, в рамках профильного курса информатики, системах дополнительного образования. Профессиональный уровень окончательно формируется в системе профессионального образования.

В основном процесс развития сценарного мышления в школьном курсе информатики происходит во время изучения темы «Коммуникационные технологии и разработка Web-сайтов.

§ 1.2. Особенности обучения теме «Коммуникационные технологии и разработка Web-сайтов» в ШКИ

Современный период развития общества по праву называют этапом информатизации. Отличительной особенностью этого периода является тот факт, что преобладающим видом деятельности в сфере общественного производства, повышающим его эффективность и наукоѐмкость, становится сбор, обработка, хранение, передача и использование информации, которые осуществляются на базе современных информационных технологий.

Тема «Коммуникационные технологии и разработка Web-сайтов» изучается в 7-11 классах, в зависимости от программы. Например: данная тема изучается в 8 классе, как продолжение темы «Коммуникационные технологии» изучаемой в 7 классе по учебнику Н.Д. Угриновича «Информатика», а по учебнику для базового уровня И.Г. Семакина, Е.К. Хеннера, Т.Ю. Шеиной «Информатика» только в 11 классе.

Данная тема изучается не углубленно и построена на простых аспектах. Работа проходит в простой системе создания Web - страниц, а именно в текстовом редакторе «Блокнот», т.к. он умеет не добавлять в текст свои специальные символы, что удобно при кодировании на языке гипертекстовой разметки — HTML.

По учебнику Н. Д. Угриновича «Информатика. 8 класс» изучается в 4 четверти и отводится 6 часов.

При изучении данной теме в основном используется проектный метод обучения, где учащиеся в группах либо самостоятельно создают свои проекты. Выполнение проектов завершается защитой перед классом результатов и рефлексией.

Основной тип занятий – практикум. Все задания осуществляются на персональном компьютере. Каждое занятие начинается с характеристики образовательного продукта, который предстоит создать ученикам. Учитель проводит ознакомление с данным продуктом – гиперссылки, мультимедиа и т.п.

Для учащихся данная тема вызывает интерес, так как они живут в сфере коммуникационных технологий. Они каждый день проводят по несколько часов в сети Internet, поэтому у них есть представления, как выглядит сайт. Структура на HTML языке осваивается довольно легко, так как она ассоциируется с предметными вещами, например <head> - «голова».

HTML – это язык гипертекстовой разметки размещенных в интернете документов. Он является одним из «базовых» языков web-программирования, т.к. HTML является одной из главных составляющих в движке сайтов и в их структуре. [12]

Характеристики HTML:

- Разработан специально для Web;
- Открытый стандарт;
- В HTML включен гипертекст;
- В HTML включена поддержка мультимедиа.

Краткий перечень возможностей языка HTML:

1. Структурирование информации на странице;
2. Форматирование текста;
3. Создание таблиц, ссылок, гиперссылок;
4. Вставка мультимедийных документов.

На моей практике, учащееся начали задавать вопросы такие, как: «мы будем создавать поля, где можно будет вводить логин и пароль при входе на наш сайт, как в социальной сети Вконтакте или Facebook?», «мы сможем отследить, сколько человек посетило наш сайт?». Исходя из данных вопрос, возникла идея включить в данный курс элементы сценарного языка PHP, чтобы наглядно показать ученикам, как это работает.

Тема «Коммуникационные технологии и разработка web-сайтов» способствует развитию сценарного мышления у учащихся, так как она дает возможность не только быть хорошим пользователем сети Internet, но дает представление о способах, методах и средствах создания web-сайтов с помощью языка HTML. Но при включении сценарного языка PHP развитие

сценарного мышления будет выше, так как учащиеся получат первоначальные навыками web-программирования, создания полноценных, динамических, интерактивных web-сайтов.

Поэтому современный учитель должен не только обладать навыками работы с языком HTML, но и с другими скриптовыми языками, ведь предметная подготовка учителя информатики не должна ограничиваться школьной программой. Скриптовые языки не изучают в школьном курсе информатики, т.к. для этого нужно изучать еще дополнительный языки программирования. Но для учеников, заинтересованных информатикой, будет интересно изучение нового и современного, которое пригодится в им будущем. Поэтому учитель может предложить, например, элективный курс по созданию Web-сайтов на другом языке (PHP, JavaScript).

Так как было описано выше, что в основном уроки проходят в режиме практических занятий, где учащиеся самостоятельно работают над своими проектами, а значит, каждый ученик работает в своем темпе. Поэтому, на мой взгляд, эффективным средством организации самостоятельной работы являются учебные элементы, в которых будут предложены описание каждого образовательного предмета и на примерах рассмотрены использование каждого, на которые ученик сможет опереться при создании собственного проекта.

§ 1.3. Использование учебных элементов программированного обучения в курсе информатики.

В современном образовании существует множество средств программированного обучения. С развитием информационных технологий учитель может использовать в качестве средств интерактивные носители информации такие, как презентации, инструкции для работы с приложениями, различные справки и голограммы. Кроме интерактивных средств может использовать устную речь, бумажные носители информации (карточки, рисунки, схемы). Все вышеперечисленные средства могут

использоваться в программированном обучении, если они созданы в соответствии с его принципами:

- принцип деления материала на небольшие, тесно связанные между собой части, порции, шаги;
- принцип активизации деятельности учащихся, изучающих учебный материал;
- принцип немедленной оценки каждого ответа учащегося;
- принцип индивидуализации темпа и содержания обучения;
- принцип эмпирической верификации (проверки) программированных текстов.

Можно выделить следующую классификацию средств программированного обучения:

По уровню активности учащихся:

a) пассивные - когда учащиеся только наблюдают и воспринимают информацию, практические задания отсутствуют (различные рисунки, схемы, устная речь учителя, справки приложений, разъясняющие правильную последовательность действий и т.д.);
b) активные - когда учащиеся воспринимают знания и практически закрепляют их, выполняя задания, предложенные средствами (карточки с заданиями, учебные элементы, устные задания учителя и т.д.).

По источнику знаний:

1) словесные (устные инструкции и объяснения учителя);
2) наглядные (рисунки, схемы, презентации);
3) практические (карточки с заданиями, учебные элементы).

На основе чувственной модальности:

1) визуальные (рисунки, схемы);
2) аудиальные (устные инструкции и объяснения учителя);
3) аудиовизуальные (презентации, мультимедийные, электронные учебные элементы).

По используемой форме обучения:

1) фронтальные (устные инструкции и объяснения учителя, рисунки, схемы, презентации);
2) групповые (карточки с заданиями, презентации);
3) индивидуальные (презентации, карточки с заданиями, учебные элементы);
4) коллективные (карточки с заданиями, презентации).

На основе данной классификации учебный элемент является объектом программированного обучения, так как он относится к данной классификации так, как он является активным, практическим, визуальным или аудиовизуальным, индивидуальным средством программированного обучения. [29]

Также в нем проявляются все принципы программированного обучения:

1) Принцип деления материала на небольшие, тесно связанные между собой части (порции, шаги) - учебный элемент представляет собой структурированный текстовый и иллюстративный материал, включающий в себя теоретическую базу, инструкции для формирования и развития определенных навыков учащихся, а также тренировочные задания и упражнения;
2) Принцип активизации деятельности учащихся, изучающих программированный текст - учебный элемент содержит задания и подробные инструкции, для их выполнения, учащиеся выполняют задания, используя персональные компьютеры, что позволяет развивать и совершенствовать практические навыки учащихся - таким образом реализуется деятельностный подход;
3) Принцип немедленной оценки каждого ответа учащегося - учебный элемент содержит задание, выполнение которых может быть проверено учителем в процессе урока, в любой момент работы учащихся над учебным элементом;

4) Принцип индивидуализации темпа и содержания учения - каждый учащийся в классе получает учебный элемент и работает с ним в своем собственном темпе, наличие у всех учащихся одинаковых элементов позволяет учащимся не ждать пока остальные товарищи завершат работу над заданием для перехода к следующему заданию;
5) Принцип эмпирической верификации (проверки) программированных текстов - все задания, включенные в учебный элемент, соответствуют возрастным и психологическим особенностям учащихся и рассчитаны на выполнение в течение одного урока без необходимости использования дополнительных источников информации.

Преимущества учебного элемента, над другими средствами программированного обучения:

1) в учебном элементе прописаны все образовательные цели данного урока. Таким образом, учащиеся точно знают цели урока, предполагаемые виды деятельности, возможно, предполагаемые результаты урока, т.е. то, чего от них потребует учитель.
2) учебный элемент также содержит теоретические сведения, что позволяет учащимся не тратить время урока на поиск информации, необходимой для выполнения заданий учебного элемента.
3) учебный элемент содержит полную подробную инструкцию для успешного выполнения заданий, отпадает необходимость самостоятельного поиска решения.
4) учебный элемент может служить средством контроля учащихся, задания располагаются для их последовательного исполнения, таким образом, учитель в любой момент урока может проверить успешность выполнения задания.
5) благодаря тому, что данное средство предназначено для индивидуальной работы, каждый учащийся работает по своему индивидуальному плану, со своей скоростью, не тратя время на ожидание остальных учащихся.

6) учебный элемент является визуальным или аудиовизуальным средством обучения, что позволяет повысить уровень восприятия и усваивания информации.

Также преимуществом использования учебных элементов является возможность эффективной реализации принципов:

- индивидуализации работы обучаемых;
- гибкой организации учебного процесса;
- постоянной обратной связи в ходе обучения;
- интенсификации учебной деятельности.

С учебными элементами обучаемый работает в темпе, соответствующем его индивидуально-психологическим особенностям. Скорость работы одних обучаемых позволяет им за время занятия освоить один учебный элемент, другие успевают проработать несколько и даже выполнить дополнительные задания. Применение учебных элементов поддерживает одну из главных компонент модульного обучения - репетиторство, когда более сильные учащиеся выступают в роли ассистентов преподавателя, консультируя и помогая более слабым. Организация взаимной помощи, совместной работы обучаемых. [18]

Нужно отметить, что индивидуальные контакты между обучаемыми часто проходят успешнее, чем контакты между преподавателем и учащимися. Причина заключается в том, что у обучаемых сходная мотивация учения, близкие интересы, похожие цели. При такой организации обучения у преподавателя оказываются востребованными его организаторский потенциал, высокая компетентность, позволяющая отвечать на сложные вопросы, возникающие у обучаемых при работе с учебными элементами.

Использование учебных элементов позволяет преподавателю сделать учебную деятельность более насыщенной, обеспечить ее интенсификацию, обеспечить глубокую организацию обучения, а также при помощи учебных элементов преподаватель контролирует ход обучения.

Выводы по 1 главе .

В результате изучения и анализа методической, психологической и педагогической литературы можно сделать следующие выводы и результаты.

1. Сценарная парадигма превосходит все существующие парадигмы программирования по своей функциональности, и она очень востребована в современном обществе;
2. На основе выделенной новой парадигмы программирования, можно выделить сценарный тип мышления, было уточнено понятие сценарное мышления, а так же его выделены уровни;
3. Были выведены особенности изучения темы: «Коммуникационные технологии и разработка web-сайтов». Главная особенность заключается в том, что уроки в основном проходят в виде практических занятий;
4. Исходя из особенностей изучения темы, пришла к выводу, что уместно будет использовать на уроках информатики учебные элементы;
5. Использование учебных элементов позволяет преподавателю сделать учебную деятельность более насыщенной, обеспечить ее интенсификацию, обеспечить глубокую организацию обучения, а также при помощи учебных элементов преподаватель контролирует ход обучения.

Глава 2. Практические аспекты реализации условий развития сценарного мышления учащихся 8 классов на уроках информатики

§ 2.1. Разработка учебных элементов по теме «Коммуникационные технологии и разработка Web-сайтов».

Разработанные учебные элементы предназначены для проведения уроков по теме «Коммуникационные технологии и разработка web – сайтов» на уроках информатики 8 классов по учебнику Н. Д. Угриновича. При разработке учебных элементов учитывалось то, чтобы они были максимально понятными для учащихся, последовательность выполнения заданий. Каждое занятие начинается с примеров и описания возможности языка, который нужно изучить на данном уроке, и который необходимо реализовать учащимся в своем сайте, и представлены задания, которые учащиеся должны выполнить на данном занятии, а так же домашнее задание. Данные учебные элементы прикрепляются в электронном журнале, в конце каждого элемента прописано домашнее задание.

Так как на данную тему «Коммуникационные технологии и разработка web – сайтов» в 8 классе отведено 6 часов, поэтому было разработано 5 учебных элементов, а 6 занятие посвящено защите разработанных проектов.

Цель учебных элементов состояла в том, чтобы развить первоначальные навыки программирования и заложить фундамент для дальнейшего изучения в 9 классе программирование на языке Pascal, а так же алгоритмическому языку с русской лексикой и встроенными исполнителями Робот и Чертёжник. И конечно же, развить сценарное мышление, так оно предполагает успешное достижение цели по решению задачи.

Первые три занятия при составлении учебных элементов пользовалась учебником Н. Д. Угринович «Информатика» для 8 класса.

Первое занятие проходит в виде лекции, где учитель знакомит с гипертекстовым языком HTML и основными продуктами, которые должны присутствовать при выполнении проектов. Учащиеся пробуют создать свою

первую страницу, форматируют текст (задание цвета, шрифта, расположение текста) с помощью парного тега <*font*>…</*font*> и задают цвет фона страницы с помощью тега <*bgcolor*>. Цвета задают с помощью таблицы RGB, либо прописывают название цвета на английском языке. В учебном элементе подробно описано, как создать страницу, основная структура программы, в каком формате сохраняется данный файл, а так же как правильно и с помощью какой программы открывается.

Так же на первом занятии, договариваются о критериях оценивания сайта, что должно присутствовать в сайте, чтобы получить оценку «отлично», «хорошо», «удовлетворительно». По данным критериям учащиеся будут предполагать по ходу выполнения заданий, на какую оценку они могут рассчитывать.

При создании сайта должны быть проговорены заранее учащимся критерии оценивания, опираясь на которые они будут следовать при создании своего проекта. Были выделены следующие критерии оценки сайта:

1. Дизайн.

а) Сайт должен быть привлекательным и аккуратным (не обязательно его делать максимально красивым, главное чтобы он не был отвратительным и не отталкивал людей);

б) Шрифт (не стандартный Times New Roman, но при этом должен быть читабелен);

в) Единый стиль (оформление каждой страницы должно быть выдержано в одном стиле, т.е. одинаковый шрифт, расположение текста и т.д.).

2. Юзабилити (т.е. ваш сайт должен быть максимально удобным и простой в использовании).

3. Наполнение сайта.

а) Текст (подходящий к сайту, должен быть оригинальным и приятным для чтения, читабелен, отсутствие грамматических и синтаксических ошибок);

б) Мультимедийные документы (подходящие к тематике сайта);

в) Содержание сайта не должно быть представлено сплошным потоком текста, он должен быть разбавлен (но не перегружен!) тематическими мультимидийными документами. Данный критерий относится и к мультимедийным документам.

4. Обязательные требования, что должно присутствовать в вашем сайте.

а) Вставка мультимедийных файлов (изображение, видео – и аудио – файлы);

б) Списки (маркированный, нумерованный или список определений);

в) Гиперссылки (сайт должен содержать не менее трех страниц);

в) Интерактивные формы.

г) Счетчик посещений сайта.

Перевод данных критериев на оценку представлен в таблице 1.

Таблица 1. Критерии оценивания сайта.

Оценка	Критерии оценивания
«отлично»	Выполнены все требования и полное соответствие сайта данным критериям.
«хорошо»	Допущены ошибки, незначительное расхождение с критериями, либо не выполнен один из пунктов.
«удовлетворительно»	Допущены значительные ошибки, несоответствие критериям, либо не выполнены два пункта.

На втором занятии учащиеся знакомятся с оформлением самой страницы с помощью вставки мультимедийных документов и с основными атрибутами для их форматирования. Для каждого документа, прописан пример как он прописывается на странице. Например, чтобы вставить изображение используется тег <img>, важно, чтобы формат изображения соответствовал с форматом в программе и правильно прописан путь данного изображения. Что касается вставки видео - и аудио – документов, они

прописываются аналогично как при вставке изображения, только используются теги <video> и <audio> соответственно. Так же на втором занятии учащиеся знакомятся со вставкой на фон изображения с помощью тега *<background>* .

На третьем занятии, у учащихся уже готова их главная страница сайта и они уже преобразуют ее в полноценный сайт с помощью списков и гиперссылок. Они знакомятся с маркированным, нумерованным списками, которые задаются парными тегами *
… и …
* соответственно. Далее самостоятельно выбирают, какой из предложенных списков им использовать в своем сайте. На полученный список они накладывают гиперссылки для перехода на другую страницу, которую они оформляют по аналогии стартовой странице.

На четвертом и пятом занятии уже включается сценарный язык PHP, так как знакомятся и создают интерактивные формы и счетчик посещений. Сценарный язык дается просто для ознакомления, так как время не позволяет его изучить полноценно, и он не предусмотрен в программе. Изучаются только те возможности языка, которые необходимы для создания интерактивных форм и счетчика посещений и учащиеся с ним знакомятся на интуитивном уровне. Но для углубленного изучения предлагается элективный курс по изучению сценарных языков для заинтересованных учащихся.

При составлении данных элементов пользовалась учебником Шкрыль А. «PHP – это просто. Программирование для web – сайта». Так как в этом учебнике подробно описаны интерактивные формы и счетчик посещений.

На четвертом занятии, учащиеся знакомятся с формами, которые задаются с помощью парного тега *<form>…</form>*. К элементам форм относятся однострочное текстовое поле, текстовая область, переключатели, флажки, списки, скрытые поля форм, поля ввода паролей и кнопки. В примерах используются две Web-страницы. Первая страница содержит форму для ввода данных пользователя и имеет расширение .html. Вторая страница

обрабатывает введенную информацию средствами языка PHP и имеет расширение .php. Так же на данном занятии учащиеся знакомятся с портативным web – сервером *Denwer* для разработки web – сайтов, который позволяет отладить сайт без необходимости непосредственного выхода в Internet. В учебном элементе, подробно описано, как запускать данный локальный сервер.

На пятом занятии, учащиеся создают счетчик посещений и отлаживают, так же свой сайт с помощью локального сервера *Denwer.*

На шестом занятии предусматривается защита выполненных проектов учащихся и рефлексия. Выступающий представляет свой проект, показывает и объясняет, какие именно он использовал возможности языка. Остальные учащиеся слушают выступающего ученика, по необходимости задают вопросы, оценивают, указывая на ошибки, если они присутствуют. Сравнивают с критериями оценивания, которые были приняты на первом занятии.

Такая постановка шестого занятия будет эффективно, так учащиеся смогут наглядно увидеть свои ошибки, адекватно оценить свою деятельность, и если были ошибки, они уже будут знать, как их разрешить в следующий раз. А так же при постановке такого занятия, у учащихся формируются личностные качества такие как: умение четко формулировать свои мысли, выдвигать собственную версию ответа, умение защищать и отстаивать свое мнение перед другими.

На мой взгляд, такая постановка занятий при изучении темы «Коммуникационные технологии и разработка web – сайтов» способствует эффективному усвоению данной темы и развитию сценарного мышления учащихся.

Разработанные учебные элементы представлены в Приложении 1.

§ 2.2. Диагностика уровня сформированности сценарного стиля мышления

На основе описанной в 1 главе структурной модели сценарного мышления и выделенных уровней его развития были разработаны следующие критерии для определения этих уровней (табл. 2).

Таблица 2. Критерии определения сформированности уровней сценарного мышления.

Уровень	Характеристика уровня	Критерии (задания на проверку)
Первоначальный уровень	способность ориентироваться в сети, использовать сервисы и услуги, предоставляемые интернетом в повседневной деятельности	1.Если вам необходимо найти срочно какую – либо информацию, куда вы чаще всего обращаетесь? 2. Допустим, Вы захотели найти биографию Леонардо Да Винчи. Что в этом случае нужно набрать в поисковой системе? А) Я хочу найти биографию Леонардо Да Винчи Б) Мне нужна биография Леонардо Да Винчи В) Леонардо Да Винчи биография
Необходимый уровень	для современного человека уровень – способность СВОБОДНО ориентироваться в сети ГРАМОТНО использовать Internet-технологии в профессиональной и учебной деятельности	1. Какие вы знаете приложения, относящиеся к Интернет-телефонии? 2. Какие сайты и Интернет-ресурсы Вы используете?

Продолжение таблицы 2.

Продвинутый уровень	иметь представление о способах, методах и средствах создания web-сайтов, их конфигурации, владение навыками создания несложных, ститчных Web-страничек с помощью, допустим языка разметки гипертекстов или несложных средств создания web-страниц	1.Доступ к файлу **book.txt**, находящемуся на сервере **bibl.ru**, осуществляется по протоколу **http**. Фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет. А) :// Б) book. В) bibl Г) txt Д) .ru Е) http Ж)/ 2.Напишите основную структуру Web-страницы на HTML.

Опираясь на предложенные критерии, была разработана диагностика уровня сформированности сценарного мышления.

Чтобы выбрать более походящее средство диагностики для проведения исследования, были проанализированы следующие средства диагностики:

- Беседа.
- Анкетирование.
- Тестирование.

Беседа - метод изучения личности, ее мотивов, чувств, мышления, намерений, понимания стоящих задач. Беседа дополняет наблюдение и эксперимент. В ходе беседы человек объясняет другим людям особенности своего поведения и психического состояния в той или иной ситуации. [31]

Данный метод я исключила из рассматриваемых, так как он времязатратен и не даст точно отследить изменение развития сценарного мышления до изучения темы «Коммуникационные технологии и разработка web-сайтов» и после ее изучения.

Основной выбор стоял между анкетированием и тестированием.

Проанализировав литературу, я пришла к решению, что анкетирование более подходящее в данном исследовании, нежели тестирование.

Анкетирование в отличие от тестирования, предполагает, что ответы на вопросы испытуемый фиксирует самостоятельно, а в тестировании он выбирает из предложенных ответов. При анкетном опросе испытуемый может высказать свое мнение, желание, что при тестировании не возможно. А также один из немаловажных плюсов анкетирования, является анонимность, что позволяет, испытуемому более открыто отвечать на задаваемые вопросы. Один из немаловажных недостатков тестирования, является элемент случайности, что так же не даст объективно оценить уровень развитии сценарного мышления, так как он может дать на сложный вопрос правильный ответ, и мы не узнаем, знал ли он ответ на данный вопрос или нет.[28]

В моем исследовании, я проверяю не знание по изученной теме, где было бы уместно провести тестирование, а уровень развития сценарного мышления. На основе проведенного анализа, я пришла к выводу, что анкетирование более уместно, чем тестирование в данном исследовании.

Анкета включает в себя, как и вопросы, предполагающие письменный ответ учащихся, так и вопросы с предложенными ответами. При составлении были включены вопросы, определяющие все уровни сценарного мышления, кроме профессионального уровня, так как профессиональный уровень формируется только при работе в сфере программирования, либо начинает развиваться при обучении в высшем заведении по специальности программист.

На каждый уровень было предложено по 5 вопросов, поэтому анкета состояла из 15 вопросов.

Вопросы, относящиеся к персональному уровню, составлялись на основе того, чтобы проверить способность ориентироваться в сети, использовать сервисы и услуги, предоставляемые интернетом в повседневной деятельности.

К необходимому уровню составлялись на основе того, чтобы проверить способность СВОБОДНО ориентироваться в сети и ГРАМОТНО использовать Internet-технологии в профессиональной и учебной деятельности.

Вопросы на продвинутый уровень составлялись на основе того, чтобы проверить имеет ли испытуемый представление о способах, методах и средствах создания web-сайтов, их конфигурации, владение навыками создания несложных, статичных Web-страничек с помощью, допустим языка разметки гипертекстов или несложных средств создания web-страниц.

Анкетирование проходит анонимно, указывется только пол. Так как при подсчете результатов можно будет отследить развитие сценарного мышления по отдельности у юношей и девушек. Мышление по половому признаку отличается друг от друга виденьем на объект, так как у юношей преобладает направленность на технические объекты, у девушек – на знаковые образы. Девушки менее предрасположены к конкретной и абстрактной деятельности, нежели юноши. Но на что не влияет половой признак так, это на техническую одаренность, т.е. на механическую понятливость, так как его формирование зависит от индивидуальных способностей человека.

Анкета составлена из 15 вопросов и исходя по подсчету времени и учитывая сложность некоторых заданий, было отведено 15 минут. (Приложение 2)

После проведения анкетирования мои предположения о том, что сценарное мышление различается по половому признаку, подтвердились. Все полученные результаты описаны в апробации.

§ 2.3. Результаты апробации

Для проверки влияния изучения темы «Коммуникационные технологии и разработка web-сайтов» на формирование сценарного мышления с 2017-2018 год на базе МАОУ Гимназия №5 был проведено исследование.

Цель исследования: исследовать процесс развития сценарного мышления при обучении теме «Коммуникационные технологии и разработка Web-сайтов» учащихся 8 класса с математическим уклоном.

Перед изучением темы проводилось входное анкетирование с целью проверки уровня сценарного мышления до изучения темы «Коммуникационные технологии и разработка Web-сайтов».

Анкетирование проводилось анонимно, указывался только пол. В анкетировании участвовало 29 человек.

Результаты диагностики представлены в табличном (табл.1) и графическом (рис.5) виде.

Таблица 3. Результаты диагностики

Пол	Первоначальный уровень	Необходимый уровень	Продвинутый уровень
Юноши	9%	18%	63%
Девушки	30%	50%	20%

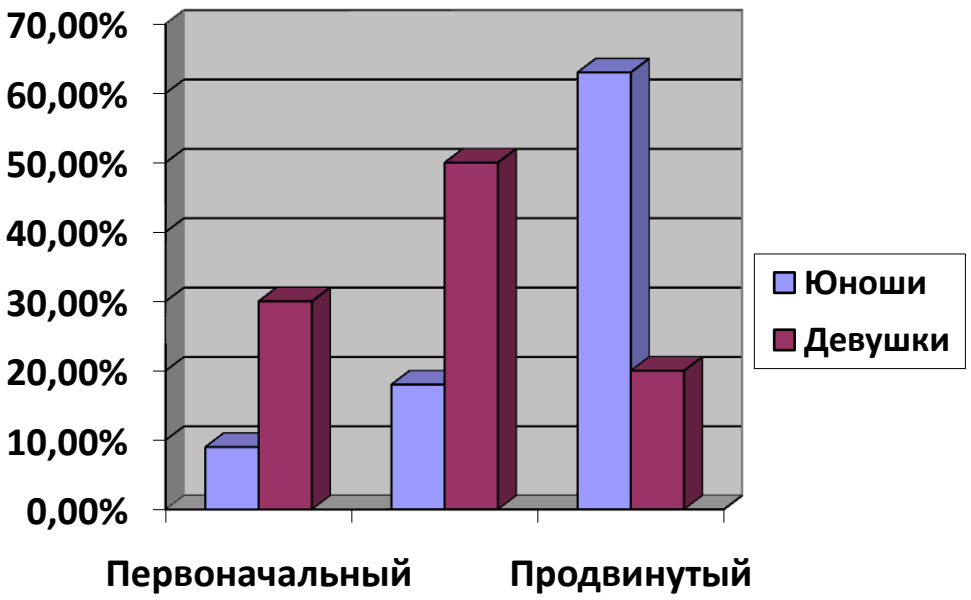


Рисунок 5. Результаты диагностики

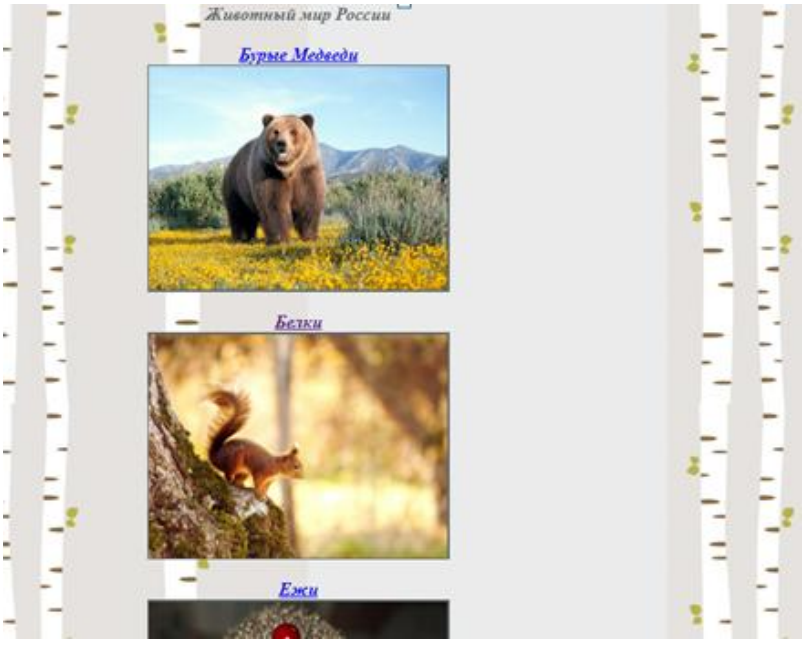
Как показали результаты порог выполненных заданий у мальчиков выше, чем у девочек. И это говорит о том, что уровень сценарного мышления у большинства юношей 8 класса сформирован на продвинутом уровне, это говорит о достаточном багаже знаний и благодаря этим знаниям они умеют составлять последовательности. У девушек в данном классе преобладает необходимый уровень, это говорит о том, что девушки чуть слабее. Каждый показатель зависит от индивидуальных способностей каждого учащегося.

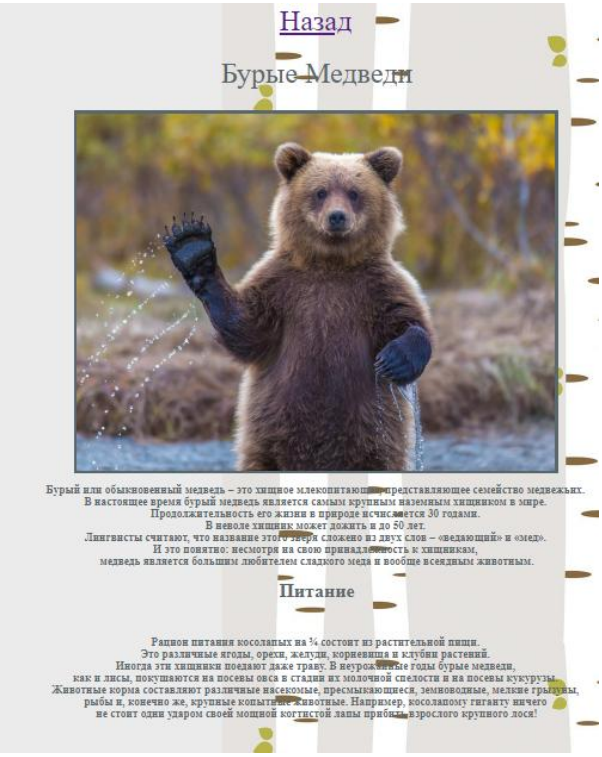
При изучении темы были использованы разработанные учебные элементы, изучения языка разметки гипертекстов html было дополнено изучением сценарного языка PHP.

На заключительном занятии была проведена защита проектов, где они представляли свои работы перед классом. По проделанной работе у многих учащихся были созданы проекты на высоком уровне, в которых были соблюдены все требования. Некоторые учащиеся не смогли достичь желаемого результата, на мой взгляд, это обусловлено тем, что они серьезно не подошли к данному процессу.

Приведу несколько примеров созданных сайтов учащимися.

Пример 1. Учащийся 8 класса Вячеслав Н. создал сайт про животный мир России, где на главной странице перечислил всех животных, которые он посчитал нужным, а с помощью гиперссылок сделал переход на их описание.





Пример 2. Учащийся 8 класса Влад И. создал сайт города Красноярск, где он описал историю данного города.

История города

Город Красноярск был основан в 1628 году Андреем Дубенским как военный острог.

Статус города Красноярск получил в 1690 году, когда Сибирь была окончательно присоединена к России.

После изучения темы «Коммуникационные технологии и разработка web-сайтов» было проведено повторное анкетирование. Результаты повторной диагностики представлены в таблице 4 и на рисунке 6:

Таблица 4. Результаты повторной диагностики

Пол	Первоначальный уровень	Необходимый уровень	Продвинутый уровень
Юноши	3%	9%	88%
Девушки	10%	30%	60%

.

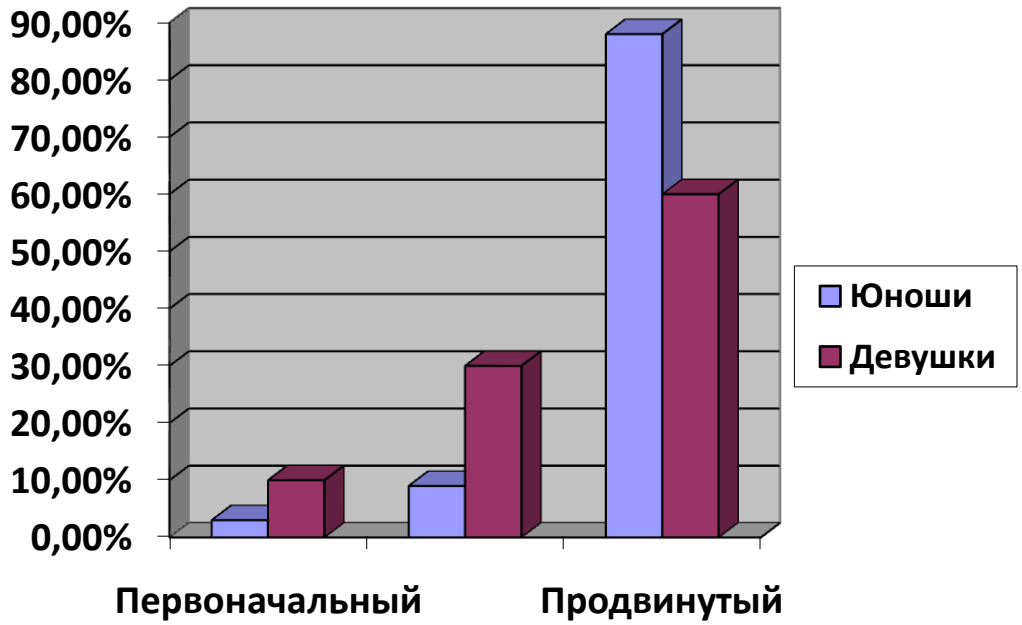


Рисунок 6. Результаты повторной диагностики

Для наглядности объединим полученные результаты до и после изучения данной темы в одну таблицу (Табл.5).

Таблица 5.

	До изучения темы			После изучения темы		
Пол	Первоначальный уровень	Необходимый уровень	Продвинутый уровень	Первоначальный уровень	Необходимый уровень	Продвинутый уровень
Юноши	9%	18%	63%	3%	9%	88%
Девушки	30%	50%	20%	10%	30%	60%

Из полученных данных наглядно видно, что уровень сценарного мышления повысился как у юношей, так и у девушек. Это говорит о том, что у учащихся сформировалось представление о способах, методах и средствах создания web-сайтов, их конфигурации, владение навыками создания

несложных, статичных Web-страничек с помощью, допустим языка разметки гипертекстов или несложных средств создания web-страниц.

Кроме того, качество созданных школьникам во время выполнения проектов их собственных сайтов, их соответствие выдвинутым критериям оценивания отражают уровень развития сценарного мышления авторов.

Выводы по 2 главе.

На основе изученной литературы в Главе 1, были созданы учебные элементы, для проведения практических занятий по теме «Коммуникационные технологии и разработка web-сайтов», а так же был включен скриптовый язык программирования PHP для развития сценарного мышления у учащихся 8 класса.

Была проведена диагностика в ходе, которой были получены результаты о том, что действительно тема «Коммуникационные технологии и разработка web-сайтов» влияет на уровень сформированности сценарного мышления у учащихся.

Заключение.

В заключении приведем полученные результаты и выводы, полученные в результате исследования.

1. Уровнем развития сценарного мышления определяется поведение субъекта в сети, на основе данного вывода была построена информационная модель сценарного мышления;
2. Выделены уровни его развития, сформированы критерии для определения этих уровней;
3. Проведена диагностика уровня сформированности сценарного мышления;
4. В ШКИ информатики включены элементы PHP;
5. Разработаны учебные элементы по теме «Коммуникационные технологии и разработка Web-сайтов», на которые получено положительное экспертное заключение учителя информатики МАОУ Гимназии №5 Мярясова А. П. (Приложение 4)
6. Результаты проведенного исследования показали что, использование учебных элементов способствует успешному усвоению учебного материала, а включение элементов PHP оказывает влияние на формирование сценарного стиля мышления.

Таким образом, можно сделать вывод, что тема «Коммуникационные технологии и разработка Web-сайтов» в ШКИ, способствует формированию сценарного мышления у учащихся, цель исследования достигнута, выдвинутая гипотеза подтверждена.

Результаты исследования внедрены в учебный процесс МАОУ Гимназия №5, акт о внедрении представлен в приложении 5.

По результатам исследования имеется публикация в материалах IV Международного научно-образовательного форума. «Человек, семья и общество: история и перспективы развития» (КГПУ им. В.П. Астафьева, Красноярск, 2015) и одна статья подготовлена к печати.

Дальнейшие перспективы исследования видятся в:

- дальнейшее уточнение понятия «сценарное мышление»;
- совершенствование диагностики уровня сформированности;
- разработке и проведении факультативного курса «Создание web-сайтов на языке JavaScript», в котором будет более глубокого изучаться язык сценарного программирования.

Библиографический список.

1. Андреева Е.В. Математические основы информатики. Элективный курс: Учебное пособие /Е.В.Андреева, Л.Л.Босова, И.Н.Фалина – М.:БИНОМ. Лаборатория знаний, 2005 – 328 с.:ил.
2. Бесценная В.В. Компетентностный подход в реализации содержания элективных курсов по информатике // Сибирский торгово-экономический журнал № 3, 2006.- С. 78-80. – 0,3 п.л.
3. Большая советская энциклопедия. [Электронный ресурс] – http://dic.academic.ru/ (20.03.2018)
4. Боршуляк М.П. Flash 5. 10-11 классы. Практикум. Информатика и ИКТ. [Текст] - изд. «Дрофа».-2002.- С. 140.
5. Википедия. [Электронный ресурс] - https://ru.wikipedia.org/wiki/ (22.05.2018)
6. Газейкина А.И. Стили мышления и обучение программиро- ванию студентов педагогического вуза. [Электронный ресурс] – http://ito.edu.ru/2006/Moscow/I.html/ (22.05.2018)
7. Жужжалов В.Е. Интеграция парадигм программирования в курсе информатики. Москва. Информатика и образование. №10. 2004. С. 32–36
8. Залогова Л.А. Практикум по компьютерной графике. – М.: Лаборатория Базовых Знаний, 2001.
9. Заславская, О.Ю. Информатика и информационно-коммуникационные технологии. Справочные материалы: Учеб. пособие для учащ. средних шк. и абитуриентов вузов / О.Ю. Заславская, И.В. Левченко. – М.: АПКиППРО, 2005. – 80 с.
10. История JavaScript. [Электронный ресурс] – http://chinapads.ru/c/s/javascript_-_istoriya (15.03.2018)
11. Киселёва О. В. Сценарная парадигма программирования // IV Международный научно - образовательный форум // «Человек, семья и

общество: история и перспективы развития» // КГПУ им. В. П. Астафьева, 2015.

12. Книга «Язык гипертекстовой разметки HTML». [Электронный ресурс] – http://www.mini-soft.ru/document/yazyk-gipertekstovoy-razmetki-html
13. Кузнецов А.А. Непрерывный курс информатики (концепция, система модулей, типовая программа)// Информатика и образование. – 2005. - №1
14. Курпатов А. В. «Что такое мышление?» [Электронный ресурс] – https://www.vshm.science/blog/avkurpato (13.03.2018)
15. Лекция в введения в программирование на языке Python [Электронный ресурс] – http://www.plam.ru (20.03.2018)
16. Описание программы Dreamweaver. [Электронный ресурс] – http://chem-otkrit.ru/ (10.04.2018)
17. Остеравт Джон. Сценарии: высокоуровневое программирование для XXI века. [Электронный ресурс] – http://www.osp.ru/(10.03.2018)
18. Поцелуева Э. Б. Учебные элементы — как способ реализации информационной образовательной технологии модульного обучения в курсе информатики. [Электронный ресурс] – http://festival.1september.ru/articles/314192/ (12.03.2018)
19. Программирование : В 2 т. Т. 1 : учебник для студ. учреждений высш. проф. образования / Э.А.Нигматулина, Н.И.Пак, М.А.Сокольская, Т.А.Степанова ; под ред. Н.И.Пака. — М.: Издательский центр «Академия», 2013. — 000 с. — (Сер. Бакалавриат)
20. Русаков М. Ю. Как создать сайт. [Электронный ресурс] – yakushin@tspu.tula.ru(7.04.2018)
21. Роббинс Д. Н. «HTML5, CSS3 и JavaScript. Исчерпывающее руководство». 4-ое издание (2014)
22. Словарь [Электронный ресурс] - http://psihotesti.ru/gloss/tag/ (13.06.2018)

23. Сокольская М. А. Методическая система обучения основам параллельного программирования будущих учителей информатики / М. А. Сокольская, Диссертация на соискание ученой степени кандидата педагогических наук: 13.00.02, Красноярск, 2012.
24. Симмондс Девон. Эволюция парадигмы программирования. [Электронный ресурс] – http://www.osp.ru/ (18.03.2018)
25. Степанова Т. А. Теория алгоритмического мышления (учебное пособие). – Красноярск: КГПУ им. В. П. Астафьева, 2014.
26. Степанова Т. А. Сущность алгоритмического мышления с позицией информационного подхода // Инновации в непрерывном образовании – 2012.
27. Тест Беннета «Механическая понятливость» [Электронный ресурс] – http://nazva.net/logic_test5/ (15.03.2018)
28. Угринович Н. Д. Информатика: учебник для 8 класса/2-е изд. – М.: БИНОМ. Лаборатория знаний, 2014
29. Угринович Н.Д., Цветкова М.С., Самылкина Н.Н. Информатика. 7-9 классы. Программа для основной школы. ФГОС, 2015
30. Шкрыль А. PHP — это просто. Программируем для Web-сайта. — СПб.: БХВ-Петербург, 2006. — 368 с.: ил.
31. Якушин А. В. Рекурсивные вычисления в различных парадигмах программирования./ Якушин А. В., Ванькина Г.В. – Тула: ТГПУ им. Л. Н. Тольстого [Электронный ресурс] – http://new.math.msu.su/conference/nikolsky-100/Articles/Vanikina.htm/ (23.04.2018)

Приложения

Приложение 1.

Учебные элементы.

Занятие №1. Первая страница. Форматирование текста.

Задание 1.

1. Запустите текстовый редактор Блокнот.
2. Напишите свою первую программу.

```
<html>
 <head>
  <title>Моя первая страница </title>
 </head>
 <body>
  <h1>Заголовок</h1>
  <!-- Комментарий -->
  <p>Первый абзац.</p>
  <p>Второй абзац.</p>
 </body>
</html>
```

1. Сохраните файл под именем index.html (обязательно укажите тип файла HTML при сохранении) в личной папке.
2. Для просмотра Web-страницы используйте любую программу браузера (Internet Explorer, Opera, Mozilla Firefox или другую). Для этого, не покидая программу Блокнот (сверните окно на панель задач), откройте личную папку и двойным кликом по файлу index.html откройте окно браузера.

Задание 2.

1. Откройте текст Web-страницы в *Блокноте* (1 щелчок правой клавишей мыши по файлу index.html, в контекстном меню выбрать команду

Открыть с помощью... и выбрать программу *Блокнот*). При необходимости открыть файл в браузере – двойной клик по значку файла левой клавишей мыши.

2. Внесите изменения в файл:
   - Задайте цвет фона с помощью атрибута *bgcolor* .
   - Оформите «Заголовок» (задайте цвет, выравнивание по центру)
   - Отформатируйте текст с помощью тэга *...* (измените цвет, размер и стиль текста).
3. Сохраните текст с внесенными изменениями в файле index.html (меню Файл | Сохранить). Если у вас уже отображается Web-страница, то вам достаточно переключиться на панели задач на программу браузера и обновить эту страницу.

Не забывайте каждый раз сохранять текст Web-страницы при ее корректировке в программе *Блокнот* и обновлять страницу при ее просмотре в программе браузера.

Дома: Придумайте тему своего сайта, и оформите начало своей стартовой страницы.

Занятие №2. Вставка изображений и видео.

Тэг <IMG> позволяет вставить изображение на Web-страницу. Оно появится в том месте документа, где находится этот тег. Тэг <IMG> является одиночным. Необходимо помнить, что графические файлы должны находиться в той же папке, что и файл HTML, описывающий страницу. Графика в Web, как правило, распространяется в трех форматах: GIF, JPG, PNG.

Для выполнения следующего задания поместите изображение в свою рабочую папку.

Следует помнить, что для браузера важно, в каком регистре вы задаете описание имени и типа файла. Выработайте для себя определенное правило и строго следуйте ему. Если вы размещаете файл графического изображения во вложенной папке, то при описании изображения необходимо указывать путь доступа к файлу изображения, отображая вложенность папок.

Задание 1. Добавление изображения.

1. Внесите изменения в файл index.html.
   <IMG SRC="Название вашего файла">
   Например: <IMG SRC="picture.png">
2. Просмотрите изменения вашей Web-страницы в браузере.

Тэг <IMG> имеет немало атрибутов, описанных в таблице 2. Эти атрибуты можно задавать дополнительно и располагаться они могут в любом месте тега после кода IMG.

Атрибуты изображения

Атрибут	Формат	Описание
ALT	<IMG SRC="CLOCK.JPG" ALT="Картинка">	Задает текст, заменяющий изображение в том случае, если браузер не воспринимает изображение
BORDER	<IMG SRC="CLOCK.JPG" BORDER="3">	Задает толщину рамки вокруг изображения. Измеряется в пикселях
ALIGN	<IMG SRC="CLOCK.JPG" ALIGN="TOP"	Задает выравнивание изображения относительно текста: • относительно текста выровнена верхняя часть изображения – **"TOP",** • относительно текста выровнена нижняя часть изображения – **"BOTTOM",** • относительно текста выровнена средняя часть изображения – **"MIDDLE".**
HEIGHT	<IMG SRC="CLOCK.JPG" HEIGHT="111">	Задает вертикальный размер изображения внутри окна браузера
WIDTH	<IMG SRC="CLOCK.JPG" WIDTH="210">	Задает горизонтальный размер изображения внутри окна браузера
VSPACE	<IMG SRC="CLOCK.JPG" VSPACE="8">	Задает добавление верхнего и нижнего пустых полей
HSPACE	<IMG SRC="CLOCK.JPG" HSPACE="8">	Задает добавление левого и правого пустых полей

Задание 2. Использование атрибутов изображения

1. Опробуйте использование таких атрибутов графики, как ALT, BORDER, ALIGN, HEIGHT, WIDTH, VSPACE, HSPACE.

Всегда обращайте внимание на размер графического файла (в байтах), так как это влияет на время загрузки Web-страницы.

2. Просмотрите изменения вашей Web-страницы в браузере

Задание 3. Установка фонового изображения на Web-странице

Фоновое изображение – это графический файл с небольшим рисунком, который многократно повторяется, заполняя все окно браузера независимо от его размеров. Графика, используемая в качестве фоновой, задается в тэге <BODY>.

1. Внесите изменения в файл index.html, предварительно подготовив и сохранив в рабочей папке графический файл фонового рисунка (FON.PNG).
2. Добавьте с помощью атрибута **BACKGROUND**
   <BODY BACKGROUND="Изображение">
   Например: <BODY BACKGROUND="FON.PNG">
3. Поэкспериментируйте с фоновым рисунком Web-страницы и выберите оптимальный вариант с вашей точки зрения.

Вставка видео- и аудио- файлов аналогично вставке изображения. Используются тэги **<video>** и **<audio>** соответственно.

Дома: оформите стартовую страницу вашего web-сайта.

Занятие №3. Гиперссылки и списки.

Задание 1. Вставка гиперссылки.

Гиперссылки состоят из двух частей: **адреса** и **указателя ссылки.** Создаются с помощью тэга <*A*>...</*A*> и его атрибута <*HREF*>, указывающего, в каком файле храниться загружаемая web – страница.

1. Создайте в своей папке новый текстовый документ *Блокнот*, напишите простую программу, которая выводит несколько слов, и сохраните как name.html файл.
2. Перейдите к редактированию своей стартовой страницы index.html. Пропишите гиперссылку для перехода на созданный name.html файл.
   <A HREF=" **Адрес**"> Указатель ссылки</A>

Если загружаемая Web-страница размещена на локальном диске в той же папке, то в качестве адреса указывается просто имя файла:

<A HREF=" name.html "> Указатель ссылки</A>

Если в качестве адреса указывается Интернет – страница, то п то в качестве адреса указывается Интернет-адрес:

Например:

<A HREF=" http://www/server.ru/Web-сайт /name.html "> Указатель ссылки</A>

3. Просмотрите изменения вашей Web-страницы в браузере.

Замечание:

Гиперссылки могут содержать адреса не только Web-страниц, но и адреса файлов других типов, активизация таких гиперссылок будет приводить:

- К просмотру изображения:

  <A HREF=” picture.jpg ”> Изображение</A>
- К запуску проигрывателя (воспроизведение звукового файла):

  <A HREF=” sound.wav ”> Звук</A>
- К сохранению файла (загрузка файла):

  <A HREF=” Архив.zip ”> Скачать файл</A>

Задание 2. Создание списка.

Маркированный список представляет собой неупорядоченный список. Создаётся с помощью парного тэга . В качестве маркера элемента списка выступает метка, например, закрашенный кружок.

Каждый элемент списка создаётся с помощью парного тэга .

С помощью атрибута TYPE тэга <UL> можно задать вид маркера списка: “disc” (диск), “square” (квадрат), “circle” (окружность).

Пример:

Программа	Результат
  - Microsoft - Google - Apple - IBM 	• Microsoft • Google • Apple • IBM

Нумерованный список создаётся с помощью парного тега . Каждый пункт списка также создаётся с помощью элемента <li>. Браузер нумерует элементы по порядку автоматически и если удалить один или несколько элементов такого списка, то остальные номера будут автоматически пересчитаны.

Для тега <li> доступен атрибут value, который позволяет изменить номер по умолчанию для выбранного элемента списка. Например, если для

первого пункта списка задать <li value="10">, то остальная нумерация будет пересчитана относительно нового значения.

Для тега <ol> доступны следующие атрибуты:

Атрибут	**Описание, принимаемое значение**
reversed	Атрибут reversed задает отображение списка в обратном порядке (например, 9, 8, 7…).
Start	Атрибут start задает начальное значение, от которого пойдет отсчет нумерации, например, конструкция <ol start="10"> первому пункту присвоит порядковый номер «10». Также можно одновременно задавать тип нумерации, например, <ol type="I" start="10">.
Type	Атрибут type задает вид маркера для использования в списке (в виде букв или цифр). Принимаемые значения: 1— значение по умолчанию, десятичная нумерация. A— нумерация списка в алфавитном порядке, заглавные буквы (A, B, C, D). a— нумерация списка в алфавитном порядке, строчные буквы (a, b, c, d). I— нумерация римскими заглавными цифрами (I, II, III, IV). i— нумерация римскими строчными цифрами (i, ii, iii, iv).

Пример:

Программа	Результат
  - Microsoft - Google - Apple - IBM 	1. Microsoft 2. Google 3. Apple 4. IBM

Списки определений создаются с помощью тега <dl></dl>. Для добавления термина применяется тег <dt></dt>, а для вставки определения — тег <dd></dd>.

Пример:

Программа	Результат.
<dl> <dt>Режиссер:</dt> <dd>Петр Точилин</dd> <dt>В ролях:</dt> <dd>Андрей Гайдулян</dd> <dd>Алексей Гаврилов</dd> <dd>Виталий Гогунский</dd> <dd>Мария Кожевникова</dd> </dl>	Режиссер: Петр Точилин В ролях: Андрей Гайдулян Алексей Гаврилов Виталий Гогунский Мария Кожевникова

Задание: Создайте списки, которые вам необходимы для оформления вашего сайта.

Дома: Оформите гиперссылки для вашего сайта и необходимые Web-страницы для перехода.

Занятие №4. Интерактивные формы на Web-страницах.

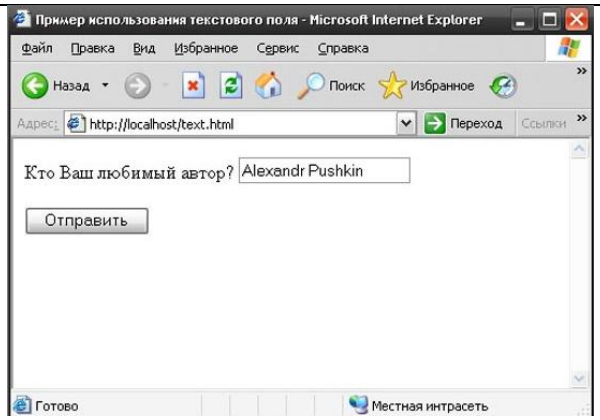
Для создания формы используется парный тэг *<form> ...</form>*. Форма прописывается в том месте, где мы бы хотели ее видеть.

К элементам форм относятся однострочное текстовое поле, текстовая область, переключатели, флажки, списки, скрытые поля форм, поля ввода паролей и кнопки. В примерах используются две Web-страницы. Первая страница содержит форму для ввода данных пользователя и имеет расширение .html. Вторая страница обрабатывает введенную информацию средствами языка PHP и имеет расширение .php.

Текстовое поле.

Создается с помощью тэга *<input>* со значением атрибута *type=”text”*. Атрибут *name* является обязательным и служит для идентификации полученной информации. Значением атрибута *size* является число, задающее длину поля ввода в символах.

Пример:

Код для ввода информации	Результат
<HTML> <HEAD> <TITLE>Пример использования текстового поля</TITLE> </HEAD> <BODY> <FORM method= "GET" action="text.php"> Кто Ваш любимый автор? <INPUT name="Author" type="text"> <INPUT type="submit" value= "Отправить"> </FORM> </BODY> </HTML>	

Для того чтобы программа вывела значения введенные пользователем, необходимо составить *программу - обработчик.* Вы создаете новый текстовый документ, пишите программу и сохраняете форматом *php.*

Чтобы программа работала, нужно запустить через *Denwer*. Для этого заходите:

Компьютер ->Локальный диск (Z:) ->home -> localhost -> www

и помещаете в папку www свою папку с сайтом. Далее заходите в браузер и в адресной строке прописываете путь к вашему файлу.

Например: *127.0.0.1/denver.ru/text.html*

Где,

*127.0.0.1/*Название папки где хранится ваш сайт/ ваш *html* файл

Пример Кода программы – обработчика:

```
<HTML>
<HEAD>
<TITLE>Пример использования текстового поля</TITLE>
</HEAD>
<BODY>
<B>Ваш любимый автор:<B>
<?php
echo $_GET['Author'];
?>
</BODY>
</HTML>
```

Реализация данного кода представлена на Рис. 1.

Рис.1 Реализация кода.

Как видно на рисунке, на экран выводится информация о любимом авторе, то есть “Alexandr Pushkin”. Программа обработки введенных данных text.php, состоит из одной строки PHP-кода: “$_GET ['Author']”, которая отображает содержимое переменной “$Author”. Данная переменная не генерируется в PHP-коде, а автоматически создается, как часть массива “$_GET”, при передаче данных на сервер. При этом “GET” обозначает метод протокола HTTP, с помощью которого передаются данные.

Замечание: имена переменных в языке PHP чувствительны к регистру символов. Так переменные “Author” и “author” являются двумя разными переменными. Если, в представленном выше примере вместо переменной “$Author” написать “$author”, то содержимое данной переменной на сервер передаваться не будет и на экране не выведется.

Флажки.

Флажки являются еще одним элементом управления формами. Они применимы в ситуациях, когда пользователю необходимо ответить на вопрос, требующий строгого однозначного ответа “да” или “нет”, путем установки или снятия “галочки”. Флажки создаются с помощью тега <INPUT>, в котором атрибуту “type” присваивается значение “checkbox”.

Синтаксис:

<INPUT name="checkbox" type="checkbox" checked>

где type - тип поля,

name - имя поля как элемента формы,

checked - атрибут, который используется для того, чтобы отметить флажок, как выделенный.

В отличие от текстового поля используется метод *post* . Использование этого метода необходимо для того, чтобы скрыть информацию из формы при передаче на сервер.

Пример использования.

Код для ввода информации	Код программы – обработчика	Результат
<HTML> <HEAD> <TITLE>Пример использования нескольких флажков</TITLE> </HEAD> <BODY> <P>Выберите язык, на котором Вы программируете: <FORM method="POST" action="checkboxes.php"> <INPUT name="Choice1" type="checkbox" value="Delphi"> Вы программируете на Delphi? <INPUT name="Choice2" type="checkbox" value="C++/C#"> Вы программируете на C++/C#? <INPUT name="Choice3" type="checkbox" value="Assembler"> Вы программируете на Assembler? <INPUT type="submit" value="Отправить"> </FORM> </BODY> </HTML>	**<HTML> <HEAD> <TITLE> Пример использования нескольких флажков </TITLE> </HEAD> <BODY> <P>Вы выбрали ответ:</P> <?php echo "$_POST[Choice1] "; echo "$_POST[Choice2] "; echo "$_POST[Choice3] "; ?> </BODY> </HTML>**	Результат кода для ввода информации Выберите язык, на котором Вы программируете: ☐ Вы программируете на Delphi? ☑ Вы программируете на C++/C#? ☐ Вы программируете на Assembler? Отправить Результат работы данного примера **Вы выбрали ответ:** C++/C#

Задание: Создайте одну из предложенных форм для своего сайта и отработайте сайт с помощью *Denwer.*

Дома: Оформите свой сайт с формами.

Занятие №5. Создание счетчика посещения Web – сайта.
(Работа с файлами)

Для того чтобы создать счетчик посещений, необходимо для начала в вашей папке, где храниться сайт, создать следующие файлы: *counter.txt, counter.php*

- counter.txt – это файл для хранения данных счётчика
- counter.php - это сам файл скрипта.

Текстовый файл не трогаем — в них будет хозяйничать наш скрипт.

Чтобы на Вашей страничке размещался счетчик, который показывал бы количество посещений данной страницы. Для этого разработаем отдельную программу с именем counter.php

Пропишем следующий код:

```
<?php
//открываем файл
$f=fopen("counter.txt", "a+t") or die ("no open file");
//Блокируем файл, чтобы никто к нему не мог обратиться
//пока мы с ним не закончим работу
flock($f, 2);
// читаем в переменную $s значение счетчика
$s=fgets($f);
//увеличиваем значение $s на 1
$s=$s+1;
//удаляем все содержимое файла counter.txt
ftruncate($f, 0);
//записываем новое число
fputs($f, $s);
 //снимаем блокировку
flock($f, 3);
//закрываем файл, теперь с ним могут работать другие
fclose($f);
//выводим показатели счетчика на экран
```

```
echo $s;
?>
```

Описание.

Прежде чем работать с файлом, его нужно открыть, с помощью функции fopen(), она имеет следующий синтаксис:

fopen(имя файла, режим работы);

Мы получаем данные из файла с помощью функции fgets(), она имеет следующий синтаксис:

fgets (файловая переменная);

Теперь необходимо записать новое значение счетчика. Но логично перед этим удалить предыдущее. Как раз для этого используется функция ftruncate(), ее синтаксис выглядит следующим образом:

ftruncate(файловый дескриптор, размер);

Теперь ничто не мешает осуществить запись нового значения счетчика в файл counter.txt, что и происходит с помощью функции fputs(). Ее полный синтаксис выглядит следующим образом:

fputs(файловый дескриптор, данные);

Способы работы с файлом	Описание
R	Файл открывается для чтения, указатель текущей позиции помещается в начало файла. Если файла не существует, то возникнет ошибка
r+	Файл открывается для чтения и записи, указатель текущей позиции помещается в начало файла. Если файл не существует, то возникнет ошибка
W	Создается пустой файл и открывается только для записи. Если файл с таким именем существует, то он перезаписывается
w+	Создается пустой файл и открывается для записи и для чтения. Если файл с таким именем существует, то он перезаписывается
A	Файл открывается для записи, указатель текущей позиции помещается

	в конец файла. Если файла не существует, то он создается
a+	Файл открывается для записи и чтения, указатель текущей позиции помещается в конец файла. Если файла не существует, то он создается

Основная работа сделана, осталось только подключить счетчик к вашей HTML – страничке (главная страница вашего сайта).

Код:

```
<?php
//Выводим поясняющую запись
echo "Данную страницу посетило уже: ";
//в середине поясняющей надписи выводим значение счетчика
require_once("counter.php");
//оканчание поясняющей надписи
echo "человек";
?>
```

Задание: Разработать счетчик посещения для своего сайта, по примеру.

Дома: оформление своего сайта.

Приложение 2.

Анкета.

1. Если вам необходимо найти срочно какую – либо информацию, куда вы чаще всего обращаетесь?
2. Как вы считаете, где хранится наибольшее количество информации?
3. Через что вы чаще всего пользуетесь услугами, такими как: заказ еды с доставкой на дом, вызов такси и т.д.?
4. Допустим, Вы захотели найти биографию Леонардо Да Винчи. Что в этом случае нужно набрать в поисковой системе?

   А) Я хочу найти биографию Леонардо Да Винчи

   Б) Мне нужна биография Леонардо Да Винчи

   В) Леонардо Да Винчи биография
5. Для чего вы пользуетесь Интернетом?

   А) для общения с друзьями

   Б) Для поиска информации

   В) Я не пользуюсь Интернетом
6. Какие поисковые системы вы знаете?
7. Какие вы знаете приложения, относящиеся к Интернет-телефонии?
8. Какие браузеры вы знаете?
9. Какие сайты и Интернет-ресурсы Вы используете?
10. Пользуетесь ли вы Интернет - почтой? Если да, то какой?
11. Доступ к файлу **book.txt**, находящемуся на сервере **bibl.ru**, осуществляется по протоколу **http**. Фрагменты адреса файла закодированы

буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла в сети Интернет.

А) ://

Б) book.

В) bibl

Г) txt

Д) .ru

Е) http

Ж)/

12. Как называется символьное имя, служащее для идентификации сайтов в сети Интернет?
13. Напишите основную структуру Web-страницы на HTML.
14. Все ли файлы формата HTML можно запустить через обычный браузер?
15. Если в строке поиска ввести запрос Kolya@emeil.ru, то результат будет следующим:

А) Откроется соответствующий почтовый адрес

Б) Откроется ссылка на аккаунт пользователя

В) Откроются все страницы, на которых имеется ссылка на конкретный адрес

Ответы:

№ вопроса	Ответ	Балл	№ вопроса	Ответ	Балл
1	Интернет	1 балл	9	Да	1 балл
2	В интернете	1 балл	10	ЕАВДЖБГ	1 балл
3	Через интернет	1 балл	11	Домен	1 балл
4	В	1 балл	12	Да	1 балл
5	А или Б	1 балл	13	<html> <head> <title>...<title> </head> <body> ... </body> </html	2 балла
6	Google, yandex, mail...	3 и более – 2 балла; Менее 3 – 1 балл	14	Да	1 балл
7	Viber, WatsApp, Skype...	1 балл	15	В	1 балл
8	www.vk.com , www.ok.ru , www.game.com ...	3 и более – 2 балла; Менее 3 – 1 балл			

Подсчет результатов:

Уровень	Балл
Первоначальный уровень	5-7 баллов
Необходимый уровень	8-15 баллов
Продвинутый уровень	16-18 баллов

Приложение 3.

Иерархия парадигм