1. Описать историю создания языка HTML. Перечислить принципы гипертекстовой разметки.

**История:** Британец Тимоти Джон Бернерс-Ли в Женевском ЦЕРНе изобрел язык гипертекстовой разметки, он же HyperText Markup Language, он же HTML, предназначенный для разметки и оформления документов World Wide Web.

**Принципы:** Гипертекстовая разметка это – набор инструкций, которые предписывают что и в каком виде в данном месте следует отобразить

Тэги можно разделить на группы:

Определяющие структуру документа, Оформление блоков гипертекста, Гипертекстовые ссылки и закладки, Формы для организации диалога, Вызов программ

Текст тега заключается в «< >».

Значение атрибута следует за знаком «=».

1. Описать основные принципы синтаксиса языка гипертекстовой разметки HTML. Привести структуру HTML-документа.

**Синтаксис**: <тег атрибут1="значение" атрибут2="значение">

Парный тег <p>… </p>

**Структура:** Язык HTML следует правилам, которые содержатся в файле объявления типа документа DTD – Document Type Definition

DTD представляет собой XML документ определяющим какие теги, атрибуты и их значения действительны для конкретного типа HTML.

Doctype – отвечает за корректное отображение веб-страницы браузера.

<!DOCTYPE HTML>  
<HTML>   
<HEAD>  
<META>  
<TITLE> </TITLE>  
<LINK>  
<SCRIPT> </SCRIPT>  
<BODY>  
</BODY>  
</HTML>

1. Описать физическое и логическое форматирование текста. Объяснить абсолютные и относительные ссылки.

**Физическое :**

<b> - жирный

<i> - курсив

<u> - подчеркнутый

**Логическое:**

<strong> - выделение значимых фрагментов текста

<em> - курсив

<acronym> - аббревиатура

<cite> и <q> - цитата, при использовании q кавычки автоматические.

<code> - отображение программного кода

<> - если в тексте встречаются определения

**Абсолютные и Относительные ссылки:**

Абсолютные ссылки применяются для указания документа на другом сетевом ресурсе, хотя допустимо делать абсолютные ссылки и внутри текущего сайта.

Относительной ссылкой называется ссылка на файлы, находящиеся на одном сервере. При создании относительных ссылок надо понимать какое значение для атрибута href следует указывать, поскольку оно зависит от исходного расположения файлов.

Основное отличие этих ссылок в URL-адресе

1. Изложить особенности вставки графики, звука и видео на Web-страницу. Описать создание списков: нумерованного, ненумерованного, списка определений.

**Графика:**

Для вставки графики используется тег **<img>**. Атрибут src определяет имя файла вместе с его расширением.

**Звук:**

Для добавления фонового звука на web-страницу применяется элемент bgsound , адрес аудиофайла прописывается при помощи, уже знакомого, атрибута src. Аудиофайл имеет ограниченное время звучания, поэтому количество повторений задается атрибутом loop. < bgsound src="audio/file.wav" loop="5" >

**Вставка видео. Тег <video>**

Добавляет, воспроизводит и управляет настройками видеоролика на веб-странице. Путь к файлу задается через атрибут src или вложенный тег <source>.

**Списки:**

**Нумерованный список** - создаётся с помощью парного тега <ol></ol>. 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.

<ol>  
    <li> Элемент №1 нумерованного списка;</li>  
    <li> Элемент №2 нумерованного списка;</li>  
    <li> Элемент №3 нумерованного списка;</li>  
</ol>

**Ненумерованный (Маркированный список)** представляет собой неупорядоченный список. Создаётся с помощью парного тега <ul></ul>. В качестве маркера элемента списка выступает метка, например, закрашенный кружок.

<ul>  
    <li>Элемент №1маркированного списка;</li>  
    <li>Элемент №2маркированного списка;</li>  
    <li>Элемент №3маркированного списка.</li>  
</ul>

Каждый элемент списка создаётся с помощью парного тега <li></li>. Вы также можете выставить для элементов <li> атрибут type: disc (закрашенные кружочки), square (квадратики) или circle (незакрашенные кружочки или окружности).

1. Объяснить создание ссылок на документы и файлы. Описать способ возвращения ссылки на переменную в языке PHP.

**Создание ссылок на документы и файлы:**

**Ссылка на почту в HTML будет выглядеть следующим образом:**

<a href="mailto:andrew@poleshuk.ru">Моя почта</a>

**Проигрывание звукового файла .wav осуществляется с помощью элемента:**

<a href=”xxx.wav”> текст </a>, щелчок по тесту приведет к проигрыванию файла.

Загрузка документов word, excel, access осуществляется с помощью элемента

<a href=”xxx.doc”> текст </a>

**Ссылка на графическую картинку**

<a target = “blank” href = “xxx.gif”> <img src=“xxx.gif”> </a>

**Ссылка на flash-анимацию :**

<a target = “blank” href = “xxx.swf”> анимация </a>

**Возвращение ссылки:**

Чтобы получить из функции ссылку, нужно при объявлении перед ее именем написать знак амперсанд (&) и каждый раз при вызове функции перед ее именем тоже писать амперсанд (&). Обычно функция возвращает ссылку на какую-либо глобальную переменную ссылку на статическую переменную или ссылку на один из аргументов, если он был также передан по ссылке.

1. Описать понятие и назначение изображений-карт (карт-ссылок). Привести пример размещения изображений-карт на веб-страницах.

**Понятие и назначение:**

Карты-изображения позволяют привязывать ссылки к разным областям одного изображения.

Реализуется в 2х вариантах:

1.Серверный – браузер посылает запрос для получения выбранной информации и ждет ответа.

2.Клиентский – карта располагается в том же HTML-док. что и ссылка на изображение.

Для указания того, что изображение является картой, используется атрибут usemap тега **<img>**.

**Пример размещения:**

Значение атрибута name данного тега должно соответствовать имени в usemap. Для задания активой области, являющейся ссылкой на HTML-документ, используется тег <area> и <map>.

Атрибуты тега AREA

Shape - Определяет форму активной области. Форма может быть в виде окружности (circle), прямоугольника (rect), полигона (poly).

Alt - Добавляет альтернативный текст для каждой области.

Coords - Задает координаты активной области. Координаты отсчитываются в пикселах от левого верхнего угла изображения.

Href - Определяет адрес ссылки для области. Правила записи такие же, как и для тега <a>.

noHref – определяет область как не активную.

1. Описать создание таблиц в языке HTML и работу с ними. Привести синтаксис составления таблицы средствами языка HTML.

Таблица состоит из строк и столбцов ячеек, которые могут содержать текст и рисунки. Для добавления таблицы на веб-страницу используется тег **<table>**. Элемент <table> может содержать различные атрибуты, например:

**Align** – устанавливает расположение таблицы по отношению к левой и правой границам документа

**Width** – ширина таблицы в пикселях

**Border** – устанавливает ширину рамки вокруг таблицы в пикселях (умолчание 0).

**Background** – фоном всей таблицы будет картинка.

Любая таблица состоит из строк и ячеек, которые задаются соответственно с помощью тегов **<tr>** и **<td>**.

1. Описать создание форм в языке HTML. Перечислить теги создания форм и элементов управления, используемых в формах.

**Формы:**

Определяются формы тэгом <FORM>. В одном документе может быть определено несколько форм для заполнения, но они не должны быть вложенными одна в другую.

Формат контейнера <FORM> следующий:

<FORM action="url" method="POST">...</FORM>

Тэг <FORM> имеет следующие атрибуты:

**Name-** имя формы

**action="..."-** обязательный параметр. Его значением является URL - адрес программы, которая будет обрабатывать содержимое формы

**method="..."-** определяет метод, используемый для посылки содержания заполненной формы на сервер.

**GET** - этот метод используется браузером по умолчанию и осуществляет присоединение содержимого заполненной формы к URL;

**POST** — при использовании этого метода содержимое заполненной формы пересылается не как часть URL, а как содержимое тела запроса.

**enctype="..."-** задает тип кодирования содержимого заполненной формы

**Для создания полей для ввода данных внутри формы применяют теги:**

**<INPUT>** - используется для задания простого элемента ввода

**<SELECT>** - предназначен для создания списков в форме, при этом внутри разрешена только последовательность тегов <OPTION>, за каждым из которых следует некоторое количество простого текста.

**<TEXTAREA>** - может быть использован для расположения многострокового поля ввода с необязательным содержимым в форме

1. Охарактеризовать понятие и назначении скриптового языка CSS. Описать преимущества языка CSS.

**Понятие и назначение CSS:**

[CSS](https://developer.mozilla.org/ru/docs/Glossary/CSS) - это язык стилей с помощью которого мы придаём стиль отображения нашего HTML контента, например придаём цвет фону (background) и шрифту, придаём контенту многоколоночный вид.

**Преимущества CSS:**

Имеется несколько  вариантов дизайнов страницы для просмотра на разных устройствах.

Уменьшается время загрузки страниц web-сайта за счет переноса правил представления данных в отдельный CSS-файл.

Простота изменения дизайна. Один CSS управляет отображением множества HTML-страниц.

CSS предоставляет дополнительные возможности форматирования

Повышение совместимости с разными платформами за счет использования web-стандартов.

1. Раскрыть способы связывания таблиц стилей с html-документом. Описать селекторы в языке CSS.

**Способы связывания документа и таблицы стилей:**

***Связывание*** – позволяет использовать одну таблицу стилей для форматирования многих страниц html.

***Внедрение*** – позволяет задавать все правила таблицы стилей непосредственно в самом документе.

***Импортирование*** – позволяет встраивать в документ таблицу стилей, расположенную на сервере.

***Встраивание*** в теги документа – позволяет изменять форматирование в конкретных элементах страницы.

**Селекторы:**

Селектор - это часть CSS-правила, которая сообщает браузеру, к какому элементу (или элементам) веб-страницы будет применён стиль.

**Селектор class** – в одной таблице стилей можно создавать 2 и больше класса одинаковых элементов с помощью селектора **class.** К этим классам элементов можно применить различные стили. Имя класса указывается через точку, например, p.red, p.blue. вызов в самой программе <p class=”red”>.

**Селектор id** - задает уникальное имя элемента, которое используется для ссылок в элементе. Параметр id можно применять к любому элементу документа. Правило таблиц стилей регламентирует использование уникального идентификатора имени элемента в качестве селектора, начиная его с символа «#».

1. Изложить понятие о динамическом HTML. Охарактеризовать объектную модель документа.

Объектная модель документа (DOM) – связывает в единое целое html язык сценария и css, предоставляя разработчикам веб-документов инструмент, называя динамически html. DOM является интерфейсом прикладного программирования для документов html. Она определяет логическую структуру документа и способ доступа и манипулирования составляющими документ элементами. В объективной модели документа любой документ представляется в виде логической древовидной структуры.

Основное назначение реализации DOM – предоставить возможность доступа и манипулирование элементами документа с помощью объектов, встроенных в некую иерархическую структуру, а также обеспечить взаимодействие между объектами, поэтому любая реализация модели

1. Охарактеризовать добавление сценария, написанного на языке JavaScript на HTML-страницу. Описать и прокомментировать структуру кода языка JavaScript.

**Добавление сценария JavaScript на HTML-страницу.**

Добавить Java Script в документ html с помощью эл-та script двумя способами:

\* Определить встроенный сценарий, который располагается непосредственно между парой тегов <script> </script>.

\* Подключить внешний файл Java Script

Обычно программу Java Script можно записать в любом месте html-станицы.

**Сценарий скрипт, внедренный в web-страницу имеет следующий вид:**

<Script>

Java Script-код

</Script>

1. Описать лексическую структуру языка JavaScript. Объяснить, что представляют собой клиентские сценарии языка JavaScript.

**Лексическая структура языка JavaScript:**

**Идентификатор** — это уникальное имя переменной, константы, пользовательской функции, объекта, массива, ключевых и зарезервированных слов, меток и т.д.

**Литерал** — это значение переменной заданное программистом, оно может быть числом, строкой, логическим значением (true/false), регулярным выражением (для поиска по шаблону) и т.д. Значение созданное программой, литералом не является.

**Операция** — это специальные знаки которые манипулируют операндами, например: плюс +, минус -, больше <, меньше > и т.д.

**Выражение** — это комбинация операндов и операций, которая может быть вычислена интерпретатором для получения значения.

**Сценарии JS:**

Сценарии JavaScript способны обрабатывать данные, введенные пользователями в полях форм, а также события, возникающие в процессе манипуляций пользователя с мышью, копировать в окно браузера другие страницы HTML или изменять содержимое уже загруженных страниц.

1. Изложить иерархию классов языка JavaScript. Охарактеризовать объект Array языка JavaScript.

**Иерархия классов JavaScript. Основные функции работы с окнами.**

Методы объекта window в объектно-ориентированном языке программирования JavaScript позволяют разработчику вести диалог с пользователем приложения, уведомлять его о важных событиях, а также управлять окнами браузера. Основные методы JavaScript для работы с окнами:

alert() – диалоговое окно содержащее информацию для пользователя;

confirm() – диалоговое окно в котором пользователь подтверждает, либо отвергает действие требуемое от него;

prompt() – диалоговое окно, с текстовым полем. Необходимо для получения от пользователя необходимой текстовой информации;

open() – открытие нового окна браузера

close() – закрытие окна браузера;

**Объект Array** - Массив является объектом и имеет ряд свойств и методов. Единственным свойством массива является length, которое показывает количество элементов, содержащихся в массиве.

1. Описать объект Date языка JavaScript. Объяснить циклы в языке JavaScript.

**Объект Date** предоставляет набор методов для работы с датой и временем.

Методы Date начинаются с префикса set и устанавливают определенные значения.

setFullYear – устанавливает год, в качестве необязательных параметров можно указать месяц и день;

setHours – устанавливает часы, необязательными параметрами являются минуты, секунды и миллисекунды;

setMinutes– устанавливает минуты, необязательными параметрами являются секунды и миллисекунды;

setMonth– устанаваливает месяц, необязательный параметр – день от начала месяца;

setSeconds– устанавливает секунды, необязательный параметр – миллисекунды;

setTime – устанавливает время в миллисекундах, прошедших с 1 января 1970 года.

**Циклы:**

Цикл - простой способ сделать какое-то действие несколько раз.

Операторы:

[for](https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide/Loops_and_iteration#%D1%86%D0%B8%D0%BA%D0%BB_for)-повторяет действия, пока не произойдёт какое-либо специальное событие завершения цикла

[do...while](https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide/Loops_and_iteration#%D1%86%D0%B8%D0%BA%D0%BB_do...while) - повторяется пока заданное условие истинно.

[while](https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide/Loops_and_iteration#%D1%86%D0%B8%D0%BA%D0%BB_while) - выполняет выражения пока условие истинно.

[break](https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide/Loops_and_iteration#break) - прерывает цикл

[continue](https://developer.mozilla.org/ru/docs/Web/JavaScript/Guide/Loops_and_iteration#continue)- используется, чтобы шагнуть на шаг вперёд в циклах while, do-while, for

1. Представить и прокомментировать синтаксис пользовательских функций. Продемонстрировать и прокомментировать общий вид синтаксиса while и do .. while в языке JavaScript.

**Синтаксис:**

Функции — это блоки кода, которые имеют своё имя и их можно вызывать по этому имени.

// создание функции

function имя\_функции() {

блок кода;

}

// вызов функции

имя\_фнкции();

**Операторы do и do..while**

**Цикл while** имеет вид:

while (условие) {

// код, тело цикла

}

Пока условие верно – выполняется код из тела цикла.

**Цикл do…while**

Проверку условия можно поставить под телом цикла, используя специальный синтаксис do..while:

do {

// тело цикла

} while (условие);

1. Охарактеризовать оператор выбора SWITCH в языке JavaScript. Охарактеризовать методы библиотеки JQuery которые позволяют манипулировать вставляемыми элементами (after(content), befor(content), insertAfter(content), insertBefore(content)).

**Оператор выбора switch**

switch (B)

{

case L1: S1;

case L2: S2;

... ... ...

case Ln: Sn;

default:S

}

**Методы JQuery**

**after(content)** - добавляет content ПОСЛЕ всех выбранных элементов

**befor(content)** - добавляет content ДО всех выбранных элементов

**insertAfter(content)** - вставляет все выбранные элементы ПОСЛЕ content-а.

**insertBefore(content)** - вставляет все выбранные элементы ПЕРЕД content-ом .

1. Изложить назначение и возможности библиотеки JQuery. Описать подключение библиотеки JQuery.

**Библиотека jQuery** помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими.

**Подключение библиотеки JQuery**

<script src="js/jquery-2.2.3.min.js"></script>

1. Расписать синтаксис JQuery-команд. Описать и прокомментировать форматы функции $() в языке PHP

.

**Стандартный синтаксис jQuery команд:**

$(селектор).метод();

* Знак **$** сообщает, что символы идущие после него являются jQuery кодом;
* Селектор позволяет выбрать элемент на странице;
* Метод задает действие, которое необходимо совершить над выбранным элементом.

**Форматы функций:**

$(“div”) – выбор всех элементов div.

$(“div, p”) – выбор всех элементов div и элементов p.

$(“.class”) – выбор всех элементов класса class.

$(“#id”) – выбор элемента с идентификатором id.

$(\*) – выбор вообще всех элементов.

1. Охарактеризовать селекторы библиотеки JQuery. Описать использование фильтров odd, eq(index), gt(index), lt(index) в библиотеке JQuery.

**Селекторы:**

Селектор позволяет выбрать элемент на странице.

$(“div”) – выбор всех элементов div.

$(“div, p”) – выбор всех элементов div и элементов p.

$(“.class”) – выбор всех элементов класса class.

$(“#id”) – выбор элемента с идентификатором id.

$(\*) – выбор вообще всех элементов.

**Фильтры:**

**odd** - выбирает нечетные элементы, начиная с нуля.

**eq(index)** - выбирает элемент по его индексу (начиная с нуля).

**gt(index)** - выбирает элементы с индексом больше указанного.

**lt(index)** - выбирает элементы с индексом меньше указанного.

1. Перечислить возможные селекторы форм. Изложить основные методы создания анимационных эффектов и синтаксис их использования в библиотеке JQuery.

**Селектор class** – в одной таблице стилей можно создавать 2 и больше класса одинаковых элементов с помощью селектора **class.** К этим классам элементов можно применить различные стили.

**Селектор id** не влияет на отображение браузером элемента html, но он задает уникальное имя элемента, которое используется для ссылок в элементе. Параметр id можно применять к любому элементу документа. Правило таблиц стилей регламентирует использование уникального идентификатора имени элемента в качестве селектора, начиная его с символа «#».

**Анимация:**

**duration** — продолжительность выполнения анимации

Может быть задана в миллисекундах (200 и 600 миллисекунд). Может быть задана строковым значением 'fast' или 'slow'

**callback** — функция, заданная в качестве обработчика завершения анимации (появления или скрытия).

Синтаксис:

**.show([duration],[easing],[callback]).hide([duration],[easing],[callback])**

**easing** — изменение скорости анимации (будет ли она замедляется к концу выполнения или наоборот ускорится).

1. Охарактеризовать библиотеку JQuery. Раскрыть назначение фильтров атрибутов библиотеки JQuery.

**Библиотека JQuery** Данная библиотека позволяет изменять содержимое HTML-документов путем манипулирования объектами модели, создаваемой браузерами в процессе обработки HTML-кода.

**Назначение фильтров атрибутов:**

Атрибуты – это дополнительные свойства HTML-тегов, несущие в себе некоторую функциональность.

**[attribute]** - Фильтр по заданным атрибутам вне зависимости от их значения

**[attribute=value]** - Фильтр по заданным атрибутам, имеющим конкретное значение

**[attribute!=value]** - Фильтр по заданным атрибутам, не имеющим конкретное значение

[**attribute^=value]** - Фильтр по заданным атрибутам, значение которых начинается с определенной строки

**[attribute$=value]** - Фильтр по заданным атрибутам, значение которых заканчивается определенной строкой

**[attribute\*=value]** - Фильтр по заданным атрибутам, значение которых содержит в себе определенную строку

1. Изложить сущность технологии XML. Охарактеризовать назначение технологии XML.

XML – расширяемый язык разметки, представляет собой набор символов или последовательностей вставляемых в текст, для передачи информации о его выводе или строении.

Разметка разделяется на:

**Стилистическая** – отвечает за внешний вид документа

**Структурную** – задает структуру документа

**Семантическая** – позволяет описать логику представления данных

XML был разработан для упрощения машинного разбора документа.

Язык XML имеет строго определенный синтаксис. Он предает XML-документу предсказуемую форму и облегчает написание программой обработки. Язык XML был создан для хранения, транспортировки и обмена данными, с его помощью можно вызывать обмен данными между различными системами.

1. Изложить структуру XML-документа. Описать пролог, тело документа (элементы, атрибуты, комментарии, инструкции обработки, секция CDATA).

**Структура XML-документа:**

<?xml version="1.0" encoding="windows-1251"?>

<book category="WEB">

<title lang="en">Learning XML</title>

<author>Erik T. Ray</author>

<year>2003</year>

<price></price>

</book>

Пролог начинается с объявления (версия и кодировка) и т.д.

Комментарии – это содержимое непредназначенное для синтаксического анализа. Пример: <!--тест -->.

Инструкция обработки – это конструкция, применяемая в XML-документе для передачи определенных указаний обработчику XML-документа.

**CDATA**

Когда в XML-Документ требуется включить текст, имеющий элементы разметки, обработку которых следует избежать, необходимо использовать секцию CDATA.

Имеют следующий формат:

• <![CDATA[текст]]>

1. Охарактеризовать пространство имен XML-документа. Описать структуру XML-документа.

**Пространство имен:**

Пространства имен предотвращают конфликты имен элементов и атрибутов, а также позволяют обрабатывать и проверять элементы и атрибуты с одним и тем же именем

Избежать конфликта имен можно при помощи префикса имени элемента:

• Xmlns:префикс=”URI”

URI – унифицированный индификатор ресурса.

**Структура XML-документа:**

<?xml version="1.0" encoding="windows-1251"?>

<book category="WEB">

<title lang="en">Learning XML</title>

<author>Erik T. Ray</author>

<year>2003</year>

<price></price>

</book>

1. Сформулировать технологии обработки данных в формате XML. Описать схему DTD, недостатки технологии XML DTD перед XML Schema.

Данные, представленные в формате XML, обычно проходят процедуру

валидации, т.е. проверки грамматики документа на соответствие

определенным схемам.

**Схема DTD**

Схема DTD представляет шаблон разметки документа, в котором указываются наличие, порядок следования и расположение элементов и их атрибутов в документе XML. Т.е. схемы DTD задают структуру и синтаксис XML документов, а не их содержание.

Недостатки XML DTD перед XML Schema

1. Отличный от XML синтаксис языка.

2. Нет проверки типов данных. В XML DTD существует лишь один тип данных – строка.

3. Нельзя поставить в соответствие одному XML-документу больше одного DTD описания.

1. Описать назначение веб-сервера Apache, его возможности. Охарактеризовать поддержку языка PHP.

**Apache - это** полнофункциональный, расширяемый веб-сервер, полностью поддерживающий протокол HTTP и распространяющийся с открытым исходным кодом. Сервер может работать практически на всех распространенных платформах.

На основе сервера Apache можно создавать не только простые любительские сайты, но и ресурсы, требующие серьезной криптографической защиты передаваемых данных.

1. Описать возможности языка PHP. Охарактеризовать внедрение PHP-сценариев в HTML-документ.

PHP используется для создания скриптов, работающих на стороне сервера. Выделяют три основные области применения PHP.

\* Первая область – это создание приложений (скриптов), которые исполняются на стороне сервера.

\* Вторая область – это создание скриптов, выполняющихся в командной строке. То есть с помощью PHP можно создавать такие скрипты, которые будут исполняться, вне зависимости от web-сервера и браузера, на конкретной машине.

\* И последняя область – это создание GUI -приложений (графических интерфейсов), выполняющихся на стороне клиента.

С помощью PHP можно создавать изображения, PDF -файлы, флэш-ролики, в него включена поддержка большого числа современных баз данных, встроены функции для работы с текстовыми данными любых форматов, включая XML, и функции для работы с файловой системой.

**PHP-сценарии:**

Существует несколько способов внедрения кода PHP в HTML документы:

\* С помощью открывающего тега <?php и закрывающего тега ?>.

Такого вида специальные теги позволяют переключаться между режимами HTML и PHP. В этом случае модуль PHP передает веб-серверу теги HTML без изменений и начнет обработку только команд, помещенных между дескрипторами PHP:

\* С помощью коротких тегов <? и ?>. Данная возможность доступна только при специальной настройке.

\* С помощью тэгов <script language="php"> и </script>

\* Путем использования echo тэгов в стиле ASP: <% и %>. Такая возможность доступна при соответствующей конфигурационной настройке.

1. Описать особенности синтаксиса языка PHP. Перечислить типы данных в PHP.

**Особенности синтаксиса:**

1. Код всегда заключен в теги
2. В конце инструкций всегда ставится ;
3. Инструкции можно записать в одну строку с любым количеством пробелов
4. Инструкция может разрываться на несколько строк и будет работать
5. Есть два вида комментариев: многострочный и одностр
6. Переменная в PHP обозначается знаком доллара, за которым следует ее имя. $a

**Типы данных:**

1 Тип boolean (булев или логический тип)

2 Тип integer (целые)

3 Тип float (числа с плавающей точкой)

4 Тип string (строки)

5 Тип array (массив)

6 Тип object (объекты)

7 Тип resource – содержит ссылку на внешний ресурс

8 Тип Null – переменная не имеет значения

1. Написать и прокомментировать синтаксис создания стандартных функций. Охарактеризовать функции .slideDown([duration],[easing],[callback]) .slideUp([duration],[easing],[callback]) библиотеки JQuery.

**Стандартные функции:**

 Стандартные функции составляют "функциональный фундамент" PHP, поскольку являются частью самого языка PHP.

**Функции .slideDown и .slideUp:**

*.slideDown([duration],[easing],[callback])*

*.slideUp([duration],[easing],[callback])*

С помощью этих функций можно показывать и скрывать выбранные элементы на странице, за счет плавного разворачивания и сворачивания.

1. Объяснить стандартные функции языка PHP для управления временем. Написать и прокомментировать синтаксис функций языка PHP для управления временем.

**Функция time, формат timestamp.**

Функция time возвращает разницу в секундах между 1-го января 1970 года и текущим моментом времени. Такое представление даты называется форматом timestamp.

Чтобы получить timestamp за любую дату следует использовать функцию mktime:

Функция **mktime** работает аналогично функции time, но, в отличие от нее, принимает параметры: **mktime**(час, минута, секунда, месяц, день, год).

Функция **strtotime** – это аналог функции mktime (тоже возвращает timestamp), только в отличие от нее принимает дату в **более** свободном формате. К примеру, можно передать ей строку **'2025-12-31'** и функция сама разберет, где тут год, где месяц, а где день, и вернет эту дату в формате timestamp.

**Синстаксис функции mktime:**

<?php  
// Устанавливаем используемый по умолчанию часовой пояс.  
date\_default\_timezone\_set('UTC');  
  
// Выводит: July 1, 2000 is on a Saturday  
echo "July 1, 2000 is on a " . date("l", mktime(0, 0, 0, 7, 1, 2000));  
  
// Выводит что-то вроде: 2006-04-05T01:02:03+00:00  
echo date('c', mktime(1, 2, 3, 4, 5, 2006));  
?>

1. Описать создание функций, определяемых пользователем в языке PHP. Написать и прокомментировать синтаксис функций, определяемых пользователем.

**Пользовательские функции:**

 C помощью стандартных функций невозможно сделать все, что нам требуется. На помощь приходит такой механизм PHP, как **функции пользователя**. С их помощью мы можем создавать свои функции, принцип работы которых аналогичен стандартным функциям PHP.

**Синтаксис:**

function Имя\_функции (параметр1, параметр2,... параметрN){

Блок\_действий

return "значение, возвращаемое функцией";

}

1. Перечислить категории аргументов функций в языке PHP. Описать использование глобальных переменных в языке PHP.

**Категории аргументов функции:**

1.по назначению(по умолчанию)

2.по ссылке

Когда аргумент передается в функцию **по значению**, изменение значения аргумента внутри функции не влияет на его значение вне функции. Чтобы позволить функции изменять ее аргументы, их нужно передавать **по ссылке**.

**Глобальные переменные:**

Чтобы использовать внутри функции переменные, заданные вне ее, эти переменные нужно объявить как глобальные. Для этого в теле функции следует перечислить их имена после ключевого слова **global**:

global $var1, $var2;

1. Охарактеризовать назначение аргументов функций в языке PHP. Прокомментировать использование статических переменных в языке PHP.

**Назначение аргументов:**

С помощью аргументов в функцию передается различная информация.

Каждый аргумент представляет собой переменную или константу. С помощью аргументов данные в функцию можно передавать тремя различными способами:

-передача аргументов по значению (используется по умолчанию),

-по ссылке,

-задание значения аргументов по умолчанию.

**Статические переменные:**

Чтобы использовать переменные только внутри функции, при этом сохраняя их значения и после выхода из функции, нужно объявить эти переменные как **статические**. Статические переменные видны только внутри функции и не теряют своего значения, если выполнение программы выходит за пределы функции. Объявление **static** $var1, $var2;

Статической переменной может быть присвоено любое значение, но не ссылка.

1. Охарактеризовать возвращаемые значения функций языка PHP. Написать и прокомментировать синтаксис возвращаемого значения языка PHP.

Любая **функция может возвращать** как результат своей работы какое-нибудь значение. Это делается с помощью утверждения **return**. Возвращаемое значение может быть любого типа, включая списки и объекты.

**Синтаксис:**

составим функцию, которая возвращает возраст человека. Если человек не умер, то возраст считается относительно текущего года.

<?php

function Age($birth, $is\_dead){

if ($is\_dead) return $is\_dead-$birth;

else return date("Y")-$birth;

}

echo Age(1971, false); // для 2009 года выведет 38

echo Age(1971, 2001); // выведет 30

?>