Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«Амурский государственный университет»

Кафедра информационных и управляющих систем

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ДИСЦИПЛИНЫ

«Мировые информационные ресурсы»

Основной образовательной программы по специальности

230201.65 – Информационные системы и технологии

Благовещенск 2012

УМКД разработан канд. техн. наук, доцентом Т.А. Галаган

Рассмотрен и рекомендован на заседании кафедры

Протокол заседания кафедры от «__» _____201_ г. № _____

Зав. кафедрой _____ / А.В. Бушманов /
(подпись) (И.О. Фамилия)

УТВЕРЖДЕН:

Протокол заседания УМСС _____
(указывается название специальности (направления подготовки)

от «____» _____ 201__ г. № _____

Председатель УМСС _____ /_____ /
(подпись) (И.О. Фамилия)

РАБОЧАЯ ПРОГРАММА

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

Информационные процессы накопления знаний и превращение их в информационный ресурс общества становятся важнейшим фактором социально-экономического развития страны и национальной экономики. Многочисленные международные, региональные и национальные информационные системы обладают громадными потенциальными возможностями для поиска идей, поэтому важно понимание методов доступа к этим информационным массивам на основе современных информационно-вычислительных и коммуникационных технических средств.

Цель курса – дать студентам комплекс знаний по организации мировых информационных ресурсов; о состоянии рынка электронной информации Российской Федерации и условиям предоставления и технологии доступа к информационным ресурсам; по основным принципам построения компьютерных сетей, их стандартизации.

Задачи дисциплины – изучение истории развития мирового рынка информационных услуг, его современного состояния.

По завершению курса «Мировые информационные ресурсы и сети» студент должен:

иметь представление о национальных и мировых информационных ресурсах;

знать принципы работы и построения компьютерных сетей, сетевые стандарты, требования предъявляемые к сетям на современном предприятии;

владеть умениями и навыками эффективного использования мировых ресурсов, основами программирования на JavaScript.

Материал дисциплины тесно связан с материалом дисциплин «Информатика», «Операционные системы», «Алгоритмические языки и программирование».

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП ВПО

Программа курса «Мировые информационные ресурсы и сети» составлена в соответствие с требованиями государственного образовательного стандарта блок дисциплине специализаций ДС. 08.

3. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

Общая трудоемкость дисциплины составляет 108 часов.

№ п/п	Раздел дисциплины	Семестр	Неделя семестра	Виды учебной работы, включая самостоятельную работу студентов и трудоемкость в часах			Формы текущего контроля успеваемости (по неделям семестра) Форма промежуточной аттестации (по семестрам)
				лекц.	лаб.	сам.	
1	Рынок информационных услуг.	5	1 – 3	6		6	Тестирование №1
2	Современный мировой рынок информационных услуг	5	4 – 7	8		8	Тестирование №1
3	Информационный рынок	5	8 - 11	8		6	Тестирование №2

	Российской Федерации.						
4	Вычислительные сети.	5	12 - 14	6		8	Тестирование №3
5	Стандарты сетей	5	16 - 18	8		8	Тестирование №3
6	Создание web-приложений	5	1 - 18		18	18	Отчет по лаб. раб.
	ИТОГО			36	18	54	

4. СОДЕРЖАНИЕ РАЗДЕЛОВ И ТЕМ ДИСЦИПЛИНЫ

4.1 Лекции

4.1.1. История развития рынка информационных услуг. Биржевая, статистическая и коммерческая информация. Информационный потенциал. Вопросы эффективности поиска в сети Интернет.

4.1.2. Современный мировой рынок информационных услуг. Этапы развития мирового рынка информационных услуг. Профессиональные базы данных. Информационные ресурсы сети Интернет. Эффективность поиска информации. Мирровый рынок знаний.

4.1.3. Информационный рынок Российской Федерации. Правовые вопросы информационной работы РФ. Государственные информационные ресурсы. Авторское право. Библиотечная сеть РФ. Патентная информация. Статистическая информация. Научно-техническая информация.

4.1.4. Вычислительные сети. Классификация сетей. Требования, предъявляемые к современным сетям. Функциональные устройства сети. Коммуникационные устройства. Топологии подключения устройств. Структуризация сети.

4.1.5. Стандарты сетей. Понятие «открытая система». Модель взаимодействия открытых систем (OSI). Организации по стандартизации. Спецификации сети Ethernet. Метод доступа, взаимодействие устройств в сети Ethernet.

4.2 Лабораторные работы

Лабораторные работы посвящены изучению основ программирования JavaScript, который широко используются в разработке web-приложений

4.2.1. Переменные, операции, функции ввода-вывода, операторы ветвления.

4.2.2. Операторы цикла, функции JavaScript.

4.3.3. Встроенные объекты JavaScript.

4.3.4. Основы создание сценариев. Работа с фреймами и окнами

4.3.5. Объектная модель браузера и документа.

4.3.6. Динамическое изменение элементов HTML-документа средствами JavaScript.

5. САМОСТОЯТЕЛЬНАЯ РАБОТА

№ п/п	№ раздела (темы) дисциплины	Форма (вид) самостоятельной работы	Трудоемкость в часах
1	Рынок информационных услуг.	Изучение учебной литературы Подготовка к тестированию	6
2	Современный мировой рынок информационных услуг	Изучение учебной литературы Подготовка к тестированию	8
3	Информационный рынок Российской Федерации.	Изучение учебной литературы Подготовка к тестированию	6
4	Вычислительные сети.	Изучение учебной литературы. Подготовка к тестированию	8
5	Стандарты сетей	Изучение учебной литературы Подготовка к тестированию	8

6	Создание web- приложений	Изучение учебной литературы Подготовка отчетов по лабораторным работам	18
	Итого		54

6. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

К образовательным технологиям, используемым в преподавании данной дисциплины, относятся лекции и лабораторные работы.

В изложении лекционного материала наряду с традиционной лекцией используются такие неимитационные методы обучения, как:

проблемная лекция, начинающаяся с постановки проблемы, которую необходимо решить в ходе изложения материала,

лекция-визуализация, учащая студента преобразовывать устную и письменную информацию к визуальной форме в виде схем, рисунков, чертежей,

лекция с заранее запланированными ошибками, которые студенты должны обнаружить самостоятельно в конце лекции.

На лекциях используются информационные технологии – презентации. Лабораторные работы проводятся в компьютерных классах и предназначены для решения прикладных задач с использованием современных инструментальных средств.

При проведении лабораторных работ используются неигровые имитационные методы обучения:

контекстное обучение, направленное на решение профессиональных задач,

работа в команде – совместная деятельность студентов в группе, направленная на решение общей задачи с разделением ответственности и полномочий.

7. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ИТОГАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ И УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

7.1. Вопросы к зачету

1. Понятие «информация»
2. Понятие «информационные продукты и услуги»
3. Классификация информационных продуктов и услуг
4. Рынок информационных продуктов и услуг
5. Этапы развития мирового рынка информационных услуг
6. Профессиональные базы данных
7. Информационные ресурсы Internet
8. Рынок финансовой информации
9. Ресурсы коммерческой информации
10. Эффективность поиска информации в Internet и профессиональных базах
11. Правовые основы информационной работы в РФ
12. Государственные информационные ресурсы
13. Библиотечная сеть РФ
14. Информационные ресурсы архивного фонда
15. Статистическая информация РФ
16. Научно-техническая информация РФ
17. Патентная информация
18. Авторское право
19. Государственные системы правовой информации

20. Биржевая и финансовая информация РФ
21. Классификация сетей
22. Требования, предъявляемые к современным компьютерным сетям
23. Функциональные устройства сети
24. Коммуникационные устройства
25. Топологии подключения устройств
26. Стандарты передающей среды
27. Понятие «открытая система»
28. Модель взаимодействия открытых систем (OSI)
29. Спецификации сети Ethernet
30. Метод доступа, взаимодействие устройств в сети Ethernet
31. Сеть Token Ring: метод доступа, взаимодействие устройств формат пакета
32. Сеть Token Ring
33. Сеть FDDI

7.2. Примеры вопросов из тестовых заданий

1. Сеть библиотек Министерства культуры и массовых коммуникаций РФ является наиболее обширной и разветвленной и включает ___ федеральных библиотек.

2. Задачи по охране изобретений, полезных моделей, промышленных образцов, товарных знаков, знаков обслуживания, наименований мест происхождения товаров в соответствии с законами РФ осуществляет:

а) Федеральная служба по интеллектуальной собственности, патентам и товарным знакам (Роспатент)

б) Федеральное государственное учреждение Федеральный институт промышленной собственности (ФГУ ФИПС);

в) Российский государственный институт интеллектуальной собственности (РГИИС);

г) Палата по патентным спорам

3. Собственниками государственных информационных ресурсов являются _____.

4. Производители документов доставляют в Российскую книжную палату:

____ обязательных бесплатных экземпляров книг, брошюр, альбомов, продолжающихся изданий, журналов, географических карт и атласов на других языках народов Российской Федерации и на иностранных языках:

____обязательных бесплатных экземпляров авторефератов диссертаций;

____обязательных экземпляров стандартов.

5. Выберите список наименований полностью относящихся к объектам авторского права:

а) производные произведения, сборники и другие составные произведения, представляющие собой по подбору и расположению материалов результат творческою труда;

б) произведения литературы, искусства и народного творчества, являющиеся результатом творческой деятельности;

в) произведения науки, литературы и искусства, являющиеся результатом творческой деятельности, а также государственные символы и знаки;

г) произведения науки и официальные документы.

6. Какая из организаций является лидером на мировом рынке в области информа-

ции об интеллектуальной собственности?

а) Dun & Bradstreet; б) LEXIS-NEXIS; в) Questel-Orbit; г) Dialog.

7. Выберите семантические показатели оценки качества поиска в массивах документов:

а) полнота выдачи и информационный шум;

б) полнота выдачи и достоверность информации;

в) пертинентность и релевантность.

8. Какую долю в ресурсах пространства WWW составляют оконечные web-страницы (к которым нельзя прийти из ядра)?

а) 27% б) 22% в) 7%

9. На какие секторы делятся мировые информационные ресурсы?

а) деловой, научно-технической и специальной; потребительской информации;

б) биржевой, потребительской; научно-технической и специальной информации;

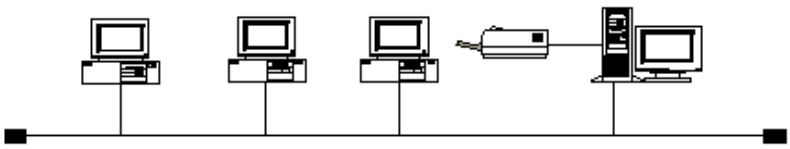
в) деловой, статистической, финансовой; потребительской информации;

10. Как называется характеристика сети, обеспечивающая сохранность информации и защиту от искажений и несанкционированного доступа?

а) интегрируемость; б) прозрачность;

в) надежность; г) масштабируемость.

11. По какой топологии изображено подключение устройств на рисунке?



а) звезда; б) шина; в) ячеистая топология; г) кольцо.

12. Специалистами какой организации разработали основные стандарты сетей Ethernet и Token Ring?

а) IEEE б) ANSI в) CCITT г) ECMA

8. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

3.1 ОСНОВНАЯ ЛИТЕРАТУРА

1 Мировые информационные ресурсы. Интернет. Практикум. Рек. УМО / Под редакцией П.В. Акинина. – М.: КноРус, - 2008. – 256 с.

2 Олифер, В.Г. Олифер. Компьютерные сети. Принципы, технологии, протоколы. /В.Г. Олифер, Н.А. Олифер. – СПб: Питер, – 2009, 2010. – 994с.

3.2. ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА

1 Дунаев, В. JavaScript. Самоучитель. – СПб.: Питер. – 2005. – 394с.

2 Галаган, Т.А. Программирование на языке JavaScript. Практикум. – Благове-

щенск: изд-во АмГУ, 2008. – 88с.

3 Хорошилов, А.В. Мировые информационные ресурсы. Учебное пособие. Рек. УМО. /А.В. Хорощилов, С.Н. Селетков. – СПб.: Питер, 2004. – 176с.

4 Хорошилов, А.В. Управление информационными ресурсами. Учебное пособие. Доп. МО. /А.В. Хорощилов, С.Н. Селетков, Л.В. Днепровская. – М.: Финансы и статистика, 2006. – 276с.

3.3. ИНТЕРНЕТ-РЕСУРСЫ

	Наименование ресурса	Характеристика
1	http://www.intuit.ru	ИНТУИТ - сайт, который предоставляет возможность дистанционного обучения по нескольким образовательным программам, касающимся, в основном, информационных технологий. Содержит несколько сотен открытых образовательных курсов.
2	http://ru.wikipedia.org	Википендия – свободная общедоступная мультиязычная универсальная интернет-энциклопедия. Поиск по статьям, написанным на русском языке. Избранные статьи, интересные факты, текущий день в истории, ссылки на тематические порталы и родственные проекты.

3.4. ПЕРИОДИЧЕСКИЕ ИЗДАНИЯ

Журналы “Информационное общество”; “Проблемы информатизации”.

9. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

Мультимедийная лекционная аудитория (331)

В качестве программного обеспечения используются свободно распространяемые инструментальные средства.

10. РЕЙТИНГОВАЯ ОЦЕНКА ЗНАНИЙ СТУДЕНТОВ ПО ДИСЦИПЛИНЕ

12.1. Балльная структура оценки за семестр

Семестровый модуль дисциплины						
Учебные модули		Виды контроля	Сроки выполнения (недели)	Макс. кол-во баллов	Посещение занятий, активно	Макс. кол-во баллов за уч. модуль
1	Рынок информационных услуг.	Тест №1	1 – 3	2	1,5	3,5
2	Современный мировой рынок информационных услуг	Тест №1	4 – 7	4	2	6
3	Информационный рынок Российской Федерации.	Тест №1	8 - 11	4	2	6
4	Вычислительные сети.	Тест №2	12 - 14	5	1,5	6,5
5	Стандарты сетей	Тест №2	15 - 18	5	2	7
6	Создание web-	Отчет по лаб. раб.	1 - 18	22	9	31

	приложений					
	Зачет					40
	Итого					100

План-конспект лекций

Лекция 1 История развития рынка информационных услуг

Рынок информационных услуг - совокупность экономических, правовых и информационных отношений по торговле между поставщиками и потребителями. Он характеризуется определенной номенклатурой услуг, условиями и механизмами их предоставления и ценами.

Товаром на рынке информационных услуг является *информация.* Информация, зафиксированная на материальных носителях и хранящаяся в *информационных системах* (библиотеках, архивах, фондах, банках данных, других информационных системах), образует *информационные ресурсы*.

Рынок информационных услуг имеет многолетнюю историю. Качественные изменения произошли в середине 1960-х годов с появлением вычислительной техники и ее использованием в области обработки и передачи информации. В этот период основными источниками информации являлись государственные информационные службы, учебные заведения, различные общественные организации, занимающиеся сбором информации в определенной области, и библиотеки.

Пользователям предоставлялась библиография, реферативная и аналитическая информация. Обслуживание производилось на коммерческой основе.

Первые автоматизированные информационные системы (АИС) получили название информационно-поисковые системы (ИПС). В связи с крайне ограниченными возможностями первых ЭВМ в ИПС хранились вторичные документы (поисковые образцы документов). Первичные документы по-прежнему хранились в библиотеках и архивах.

Такие системы работали в двух режимах:

избирательного распространения информации;

ретроспективного поиска.

Следующим шагом, который стал возможным в связи с развитием вычислительной техники, явилось возникновение фактографических ИС. Эти системы содержали формализованную информацию в виде значений свойств различных объектов (лиц, организаций, событий и т.д.).

Виды деятельности, связанные с формированием информационных ресурсов, поддержанием их в актуальном состоянии, созданием средств связи, обработки и копирования информации, объединяют в понятие *информационной индустрии***.** Под средствами обработки наряду с вычислительной техникой понимается и программное обеспечение.

Способность решать задачи информационного обслуживания на уровне максимальных возможностей, определяемых достигнутым на данный момент состоянием развития вычислительной техники и связи, называют *информационным потенциалом.*

Структуры, которые работают на информационном рынке, предлагают потребителям следующие виды услуг:

непосредственный доступ к базам данных – режим on-line;

пакетный доступ к базам данных – режим off-line;

базы данных на дискетах и компакт-дисках;

консультации, оказываемые специалистами в области информационных ресурсов;

обучение доступу к мировым информационным ресурсам.

Поставщики информации на рынке информационных услуг – коммерческие структуры, государственные и общественные организации, частные лица. Обычно они именуются информационными корпорациями, информационными агентствами, информационными службами, информационными центрами.

Становление рынка электронной информации сопровождалось специализацией организаций, занимающихся информационным обслуживанием. Сформировалось три груп-

пы информационных служб:

центры-генераторы (производители информации) – специализируются на добыче информации, формировании баз данных и поддержании их в актуальном состоянии;

центры распределения (поставщики информации, так называемые вендоры) – занимаются информационным обслуживанием пользователей на основе баз данных, поставляемых им на коммерческой основе центрами-генераторами;

информационные агентства – помимо функции сбора информации, формирования и ведения баз данных осуществляют и функции обслуживания пользователей.

Лекция 2 Биржевая, статистическая и коммерческая информация

Современный информационный рынок можно разделить на четыре области: электронная информация, электронные сделки, системы сетевых коммуникаций, программное обеспечение.

В свою очередь рынок электронной информации состоит из 4 секторов: деловая информация, юридическая информация, информация для специалистов, массовая или потребительская информация.

Деловой сектор (в рамках электронной информации):

биржевая и финансовая информация, генераторами которой являются банки, биржи и брокерские конторы. Эта информация о рынке ценных бумаг, котировки валют, рынке товаров, капиталов, услуг, а также инвестициях и ценах.

экономическая и статистическая информация, числовая информация.

коммерческая информация - информация о коммерческих предложениях, о куплепродаже по определенным товарным группам.

новости в области экономики и бизнеса.

Юридический сектор включает системы доступа к электронным сборникам указов и т.п.

Сектор информации для специалистов состоит из следующих частей:

профессиональная информация, дифференцированная по областям науки и техники,

доступ к первоисточникам (библиографическая и реферативная информация).

Массовая и потребительская информация: информация служб новостей и агентств, пресса и др.

Рынок электронных сделок включает системы банковских и межбанковских операций, системы электронных торгов, системы резервирования товаров и услуг. В рамках этого рынка имеет значение электронный обмен данными, который обеспечивает возможность безбумажного документооборота. При этом велика роль службы безопасности, предотвращающей несанкционированный доступ к этой информации.

Рынок программного обеспечения - все виды программной продукции и их обслуживание.

Системы сетевых коммуникаций - электронная почта, телеконференции, электронные сетевые доски объявлений и др., системы ТВС.

К наиболее предоставляемым услугам распространения относят:

телекоммуникационные услуги (обмен сообщениями в режиме электронной почты как между пользователями одной сети, так и между разными сетями; обмен сообщениями в телеконференциях и телесеминарах; организация электронных бюллетеней, электронных новостей; организация общения в режиме запрос-ответ; передача больших массивов информации в файлах, тиражирование информации и передача по определенному списку адресов; выдача копий сообщений по запросу абонента),

информационные услуги (поиск информации по запросам в справочных системах),

консультационные услуги (консультации по программному сетевому обеспечению, консультации по технологии использования общественных ресурсов в сети и обучение

навыкам работы с компьютером и техническими средствами),

технические услуги (установка и обслуживание программного обеспечения и тестирование техники и программ),

рекламные услуги.

Лекция 3 Вопросы эффективности поиска в сети Интернет

Классификация источников информации в Интернете может проводиться по разным основаниям.

По способам представления информации выделяют:

web-страницы — наиболее распространенный и используемый из информационных ресурсов. Этот ресурс представляет собой страницы гипертекста, содержащие наряду с текстовой - графическую, звуковую, видеоинформацию;

файловые серверы, представляющие собой реализацию в Интернете традиционного способа представления информации;

телеконференции. Они разбиваются на группы (рубрики) по тематике. Участвующие в телеконференциях могут написать свое сообщение или послать комментарии на чужое сообщение;

базы данных, содержащие, кроме текстовой, и другие виды информации.

Информационные ресурсы также могут быть разделены *по языковому признаку* (бесспорным лидером являются англоязычные сайты).

В сети имеет место классификация и *по территориальному признаку*.

Одними из наиболее важных показателей эффективности информационных систем, содержащих текстовую информацию, являются семантические показатели. Семантические показатели основаны на оценке релевантности между документами и запросами.

Релевантность — объективно существующее смысловое соответствие между содержанием документа и запроса. Объективность оценок релевантности обеспечивается тем, что они устанавливаются экспертным путем, а не автором запроса.

Семантическими показателями являются *полнота выдачи* (потери информации) и *точность выдачи* (информационный шум).

Введем следующие обозначения:

а - множество релевантных и выданных системой документов;

б - множество нерелевантных, но выданных системой документов;

в - множество релевантных, но не выданных системой документов.

$$\text{Полнота выдачи (ПВ)} = \frac{a}{a + \hat{a}} \times 100\%$$

$$\text{Точность выдачи (ТВ)} = \frac{a}{a + \acute{a}} \times 100\%$$

$$\text{Потери информации (ПИ)} = \frac{\hat{a}}{a + \hat{a}} \times 100\%$$

$$\text{Информационный шум (ИШ)} = \frac{\acute{a}}{a + \acute{a}} \times 100\%$$

Другой группой показателей оценки эффективности документальных информационных систем являются прагматические показатели. Эти показатели можно определить только в процессе эксплуатации информационной системы.

Прагматические показатели определяют абоненты системы на базе оценок пертинентности выданных документов. *Пертинентность* — это субъективно оцениваемое соответствие содержания документов или текстов информационным интересам потребителя. Пертинентность может оценить только автор запроса, работающий с информационной системой. Оценки пертинентности, как правило, отличаются от результатов, полученных на основе оценок релевантности.

Лекция 4 Этапы развития мирового рынка информационных услуг.

Рынок информационных услуг имеет многолетнюю историю. Качественные изменения он претерпел в середине 1960-х гг. с появлением вычислительной техники и ее использованием в области обработки и передачи информации. В этот период основными источниками информации являлись государственные информационные службы, учебные заведения, различные общественные организации, занимающиеся сбором информации в определенной области, библиотеки. Пользователям предоставлялась библиография, реферативная и аналитическая информация.

Первые автоматизированные информационные системы получили название «Информационно-поисковые системы» (ИПС). При внедрении ИПС была автоматизирована лишь часть функций информационного обслуживания.

Следующим шагом, который стал возможным в связи с развитием вычислительной техники, ростом объемов памяти и быстродействия, явилось возникновение фактографических информационных систем. Эти системы содержали уже формализованную информацию в виде значений свойств различных объектов (лиц, организаций, событий и т. д.) в виде баз данных.

Большое влияние на развитие сферы информационных услуг оказало создание национальных и мировых сетей передачи данных. Это обстоятельство, наряду с ростом возможностей вычислительной техники, породило возникновение принципиально новых информационных технологий, что в свою очередь обеспечило коренное улучшение качества информационного обслуживания. Одно из главных достижений в этой сфере – появление возможности диалогового доступа пользователей к удаленным базам данных (режим on-line). Новейшие информационные технологии и расширение номенклатуры и качества информационных услуг повысили производительность труда в информационной сфере. Переход к применению компьютеров при подготовке печатных изданий позволил резко сократить затраты на формирование баз данных с использованием этих источников.

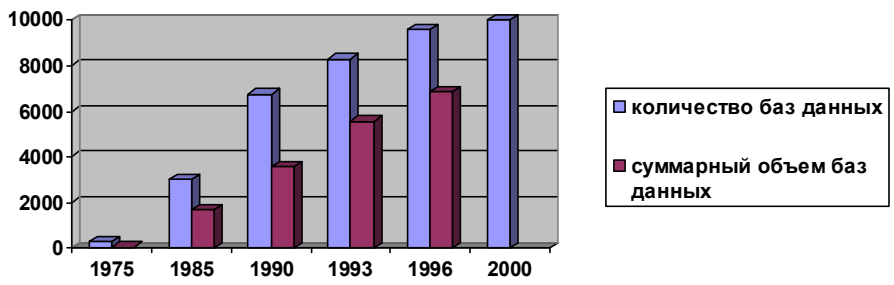
Необходимо отметить, что первопричиной скачка в развитии сферы информационного обслуживания явился резкий рост потребностей пользователей, связанный с ростом производства и развитием науки и техники. Данное обстоятельство определило выгодность вложения денежных средств в эту отрасль и создание свыше 7500 крупных информационных корпораций.

Ведущие мировые информационные корпорации:

Информационная корпорация «Дан энд Брэдстрит» (Dun & Bradstreet)
Информационная корпорация «Диалог» и компания Data-Star
Информационная корпорация Questel-Orbit
Информационное агентство LEXIS-NEXIS

Лекция 5 Профессиональные базы данных

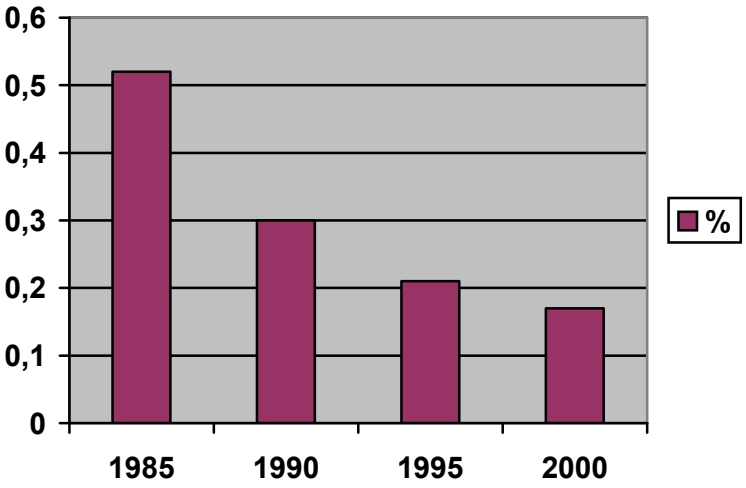
Одной из форм предоставления информационных услуг потребителю является обеспечение его доступа **к профессиональным базам данных** научно-технической, деловой, правовой и другой информации. На рис. показаны тенденции роста количества баз данных и их объема.



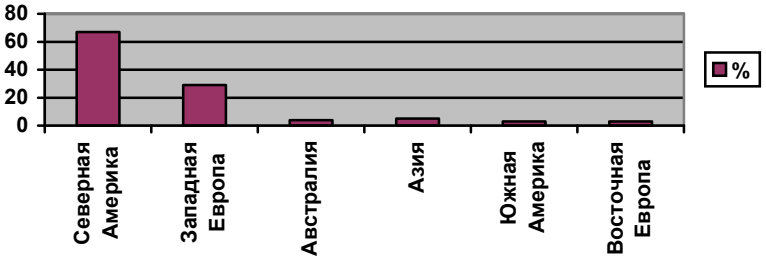
Содержание баз данных отражает потребности пользователей. Если перво-

начально потребители запрашивали главным образом научно-техническую информацию, то в настоящее время их основной интерес направлен на информацию, касающуюся бизнеса, торговли и промышленности. Большой интерес также проявляется к патентной и правовой информации.

По своей форме информационные ресурсы, содержащиеся в базах данных, могут быть разделены на числовые (фактографическая информация) и текстовые. Объемы текстовых баз данных значительно превышают объемы числовых баз, и этот разрыв все более увеличивается с течением времени.



Распределение информационных мировых ресурсов по регионам мира в 2000 г:



Ориентироваться в гигантском объеме мировых информационных ресурсов позволяет популярное во всем мире справочное издание Gale Directory of Databases, выпускаемое дважды в год в виде двух томов фирмой Gale Research, Inc.

Лекция 6 Оценка эффективности использования мировых ресурсов

Мировые ресурсы представляют собой масштабную систему, развитие которой может протекать в различных направлениях. Поэтому оценка эффективности использования мировых ресурсов производится по тем же правилам, что и оценка других систем.

Эффективность системы - это в общем случае совокупность свойств, характеризующих качество функционирования системы, оцениваемое как соответствие требуемого и достигаемого результата.

Для оценки эффективности системы разрабатывают совокупности критериев оценки. В зависимости от типа системы и внешних воздействий предлагают детерминированные, вероятностные, качественные критерии; вводят понятия технической (с точки зрения технических характеристик), экономической, социально-экономической эффективности.

Основные этапы оценивания эффективности можно выделить следующим образом:

Этап 1. Определение цели оценивания.

Можно выделить два типа целей: качественная - цель, достижение которой выражается в номинальной шкале или в шкале порядка; количественная - цель, достижение которой выражается в количественных шкалах. Определение цели должно осуществляться с позиции, в которой рассматриваемая система является элементом (подсистемой), т.е. с позиций надсистемы.

Этап 2. Измерение свойств систем, признанных существенными для целей оценивания.

Для этого выбираются соответствующие шкалы измерений свойств и всем иссле-

дуемым свойствам систем присваивается определенное значение на этих шкалах.

Этап 3. Обоснование предпочтений — критериев качества и критериев эффективно-сти функционирования систем на основе измеренных на выбранных шкалах свойств.

Этап 4. Собственно оценивание.

Все исследуемые системы, рассматриваемые как альтернативы, сравниваются по сформулированным критериям и, в зависимости от целей оценивания, ранжируются, выбираются, оптимизируются и т. д.

Существенные свойства в соответствии с представлением системы как семантической модели можно условно классифицировать не только по уровню сложности, но и по принадлежности к системообразующим (общесистемным), структурным или функциональным группам.

В общем случае оценка функциональных свойств системы проводится как оценка двух аспектов:

а) исхода (результатов) функционирования; б) «алгоритма», обеспечивающего получение результатов.

Качество исхода и «алгоритм», обеспечивающий получение результатов, оцениваются по показателям качества. Показатели качества вводятся с учетом конкретных особенностей системы и условий ее функционирования.

К основным укрупненным показателям качества операции относят результативность, ресурсоемкость, оперативность.

В совокупности результативность, ресурсоемкость и оперативность порождают комплексное свойство - эффективность процесса - степень его приспособленности к достижению цели. Это свойство, присущее только операциям, проявляется при функционировании системы и зависит как от свойств самой системы, так и от внешней среды.

Выбор критерия эффективности - центральный, самый ответственный момент исследования системы. Процесс выбора критерия эффективности, как и процесс определения цели, является в значительной мере субъективным, творческим, требующим в каждом отдельном случае индивидуального подхода. Наибольшей сложностью отличается выбор критерия эффективности решений в операциях, реализуемых иерархическими системами.

Лекция 7 Мировой рынок знаний

Рынок знаний не имеет четких границ. Его можно определить как рынок, где происходит продажа и передача знаний.

Рынок знаний своеобразен. Так, значительная часть знаний на нем распространяется безвозмездно или за символическую плату, например школьное и в большей части среднее и высшее профессиональное образование. Бесплатность или низкая плата за эти знания для их потребителей объясняется тем, что процесс передачи этих знаний финансируется из государственного бюджета или некоммерческими организациями.

Структурно рынок знаний представлен большим количеством секторов, из которых выделяются по своим размерам такие сектора, как наука (НИОКР, НИР), образование (образовательные услуги), средства массовой информации, хранение информации (архивы, библиотеки, информационные сети) и др.

Наконец, своеобразная черта рынка знаний — это большое количество посредников на этом рынке, а главное, что целый ряд секторов (образования, хранения информации и др.) выступает преимущественно как посредник между производителями знаний (например, НИОКР) и потребителями (домашними хозяйствами, фирмами, государственными учреждениями.

Технологии выступают на мировом рынке в виде некоего комплекса научно-технических знаний о приемах и методах производства, а также формах его организации и управления. Основными участниками мирового рынка технологий оказываются государственные структуры, научно-исследовательские и образовательные учреждения, частные

компании и физические лица - специалисты и эксперты в той или иной области. Мировой рынок технологий развивается достаточно динамично, даже невзирая на экономические потрясения. Например, еще в середине девяностых годов XX века сектор сведений и технологий давал до 75% внутреннего валового продукта США и вместе со сферой услуг обеспечивал до 80% рабочих мест в стране.

Позиции лидеров на мировом рынке технологий, безусловно, принадлежат таким странам, как США, Япония, Германия, Франция и Великобритания. Они расходуют на научные разработки едва ли не вполовину больше средств, чем все остальные страны мира, вместе взятые.

В основе информационно-технологической сферы международных экономических отношений лежит так называемый международный рынок информации и технологий - МРИТ. Объектами сделок на нем становятся результаты интеллектуальной деятельности человека в овеществленной и неовеществленной формах. Под овеществленной формой понимается готовое производственное оборудование, а под неовеществленной - различные виды информации, производственный опыт и знания.

С конца XX века ядром мирового рынка технологий стал такой товар, как информационные технологии (IT). Это понятие служит для обозначения широкого класса дисциплин и областей деятельности, относящихся к технологиям управления и обработки данных, а также создания данных, в том числе, с применением вычислительной техники. Современные информационные технологии характеризуются такими признаками, как компьютерная обработка информации по заданным алгоритмам, хранение больших объёмов информации на машинных носителя и передача информации на значительные расстояния в ограниченное время.

Лекция 8 Информационный рынок Российской Федерации. Правовые вопросы информационной работы РФ

1991-1993 годы знаменуются рождением информационного рынка в России. Свидетельством тому является:

Во-первых, конец монополии государственной телефонной сети, появление ряда совместных с западными фирмами-производителями средств связи предприятий, которые ввели в эксплуатацию международные выделенные цифровые сети и стали предлагать альтернативные Министерству связи услуги по передаче речи, обеспечения факсимильной связи и электронного обмена данными.

Изменился и характер деятельности самого министерства связи. Если в доперестроечные времена это была просто большая компания операторов, руководителем которой был член Правительства, и которая отвечала на жалобы, организовывала очереди, и т.п., то теперь Министерство оставило за собой только задачи регулирования развития общества (лицензирование, сертификацию), переведя все подведомственные структуры на хозрасчетные отношения. За три последних года Минсвязи выдало 353 лицензии на эксплуатацию средств связи, из них 151 - на средства электросвязи. Среднее время от заявки до выдачи лицензии равно приблизительно 2 месяцам.

Вторым свидетельством рождения информационного рынка является появление множества (более десятка) сетевых структур, предлагающих своим пользователям сходный набор услуг. Между этими информационными предприятиями возникла реальная конкуренция.

Развитие информационного рынка в России имеет некоторые отличия от того процесса развития, который происходит на западе. В России появляющиеся сетевые образования инвестируются исключительно крупным бизнесом, поэтому они коммерциализованны от рождения. Имеются также мировые тенденции развития сетей компьютерной связи, которые отражаются на характере развития отечественных сетевых структур. В частности, в современном мире важным условием конкурентной способности предприятия,

оказывающего телекоммуникационные услуги, является представление пользователям возможности связи с компьютером, находящимся в любой точке планеты. Поэтому имеется общая тенденция объединения в той или иной форме различных сетевых структур.

Структура информационного рынка:

1. электронная информация;
2. электронные сделки;
3. системы сетевых коммуникаций;
4. программное обеспечение.

Основным законом, определяющим правовые основы информационной работы в России, является Федеральный закон от 27 июля 2006 г. N 149-ФЗ "Об информации, информационных технологиях и о защите информации" (с изменениями и дополнениями).

В соответствии с этим законом информационные ресурсы делятся на государственные и негосударственные. Собственниками государственных информационных ресурсов являются Российская Федерация и субъекты РФ. Эти ресурсы создаются, приобретаются, накапливаются за счет федерального бюджета, бюджетов субъектов РФ, а также иными установленными законом способами.

Физические и юридические лица являются собственниками тех документов (массивов документов), которые созданы за счет их средств, приобретены ими на законных основаниях, получены в порядке дарения или наследования. Эти ресурсы являются негосударственными.

Граждане, органы государственной власти, органы местного самоуправления, организации и общественные объединения обязаны представлять документированную информацию органам и организациям, ответственным за формирование и использование государственных информационных ресурсов.

Перечни представляемой в обязательном порядке документированной информации и перечни органов и организаций, ответственных за сбор и обработку федеральных информационных ресурсов, утверждает правительство РФ. Порядок и условия обязательного представления информации доводятся до сведения граждан и организаций.

По категориям доступа информация делится на открытую информацию и информацию с ограниченным доступом. Информация с ограниченным доступом делится, в свою очередь, на *информацию, отнесенную к государственной тайне,* и *конфиденциальную.*

Отнесение информации к государственной тайне осуществляется в соответствии Закон «О Государственной тайне». Настоящий Закон регулирует отношения, возникающие в связи с отнесением сведений к государственной тайне, их рассекречиванием и защитой в интересах обеспечения безопасности Российской Федерации. Принят 21 июля 1993 года. С изменениями и дополнениями от 01.12.2007 N 294-ФЗ, N 318-ФЗ.

Государство имеет право выкупа документированной информации у физических и юридических лиц в случае отнесения этой информации к государственной тайне. Собственник информационных ресурсов, содержащих сведения, отнесенные к государственной тайне, вправе распоряжаться этой собственностью только с разрешения соответствующих органов государственной власти.

Запрещено относить к информации с ограниченным доступом:

законодательные и другие нормативные акты, устанавливающие правовой статус органов государственной власти, органов местного самоуправления, организаций, общественных объединений, а также права, свободы и обязанности граждан, порядок их реализации;

документы, содержащие информацию о чрезвычайных ситуациях, экологическую, метеорологическую, демографическую, санитарно-эпидемиологическую и другую информацию, необходимую для обеспечения функционирования населенных пунктов, производственных объектов, безопасности граждан и населения в целом;

документы, содержащие информацию о деятельности органов государственной власти и местного самоуправления, об использовании бюджетных средств и других госу-

дарственных и местных ресурсов, о состоянии экономики и потребностях населения, за исключением сведений, относящихся к государственной тайне;

документы, накапливаемые в открытых фондах библиотек и архивов, информационных системах органов государственной власти, органов местного самоуправления, общественных объединений, организаций, представляющие общественный интерес или необходимые для реализации прав, свобод и обязанностей граждан.

Персональные данные (информация о гражданах) относятся к конфиденциальной информации. Подлежит обязательному лицензированию деятельность негосударственных организаций и частных лиц, связанная с обработкой и предоставлением пользователям персональных данных. Порядок лицензирования определяется законодательством РФ.

Лекция 9 Библиотечная сеть РФ.

В этой сети хранится главным образом опубликованная и тиражированная информация, представленная в виде различных отечественных и зарубежных изданий. Библиотеки обеспечивают доступность этой информации для массового пользователя.

Библиотечная сеть РФ насчитывает свыше 150 тыс. библиотек, куда включаются:
публичные библиотеки всех уровней (федерального, регионального, муниципального уровней);
система научно-технических библиотек и справочно-информационных фондов, входящую в российскую государственную систему научно-технической информации (ГСНТИ);
информационно-библиотечную систему РАН;
библиотечную систему высших учебных заведений;
сеть муниципальных библиотек;
сеть сельскохозяйственных библиотек.

При построении сетей сочетаются отраслевой и региональный принципы. На федеральном уровне крупнейшими публичными библиотеками являются:
российская государственная библиотека – 38 млн. единиц хранения;
российская национальная библиотека – 30 млн. единиц хранения;
всероссийская государственная библиотека иностранной литературы – данных нет.

На региональном уровне имеется 217 центральных библиотек во всех субъектах федерации и 49700 публичных городских и муниципальных библиотек. Наряду с обслуживанием индивидуальных пользователей библиотеки также ведут обслуживание организаций.

Сеть научно-технических библиотек включается в себя 7000 библиотек с фондом 10 млн. единиц хранения.

Библиотеки РАН объединены в 4 системы:
система библиотек Сибирского региона – 67 библиотек с 13,5 млн. единиц хранения;
система библиотек Санкт-Петербургского региона – 41 библиотека, 16,5 млн. единиц хранения;
отраслевая система библиотек по естественным наукам – 13 млн. единиц хранения;
отраслевая система библиотек институтов РАН гуманитарного и общественного направления – 13 млн. единиц хранения.

Библиотечная сеть ВУЗов – 500 библиотек с фондом 300 млн. единиц хранения.

Важным законом, определяющим порядок формирования фондов государственной библиотечной системы и органов научно-технической информации, является Федеральный закон от 29 декабря 1994 г. № 77-ФЗ «Об обязательном экземпляре документов» и Постановление Правительства РФ от 24 июля 1995 г. № 739 «Об обязательных экземплярах изданий».

Обязательные экземпляры – экземпляры различных видов тиражированных документов, подлежащие передаче производителями в соответствующие учреждения и организации в порядке и в количестве, установленных Федеральным законом.

Обязательные бесплатные экземпляры — экземпляры различных видов документов, подлежащие безвозмездной передаче их производителями в соответствующие учреждения и организации в порядке и в количестве, установленных Федеральным законом.

Лекция 10. Авторское право. Патентная информация.

История законодательства авторского права в России:

3 августа 1992 года, в период смены законодательства в Российской Федерации, в действие введены Основы гражданского законодательства Союза ССР и республик. Новая регламентация авторских прав была предусмотрена Разделом IV Основ («Авторское право»), где появилось понятие охраны смежных прав.

9 июля 1993 года Президент Российской Федерации подписал Закон «Об авторском праве и смежных правах» (вступил в силу 03 августа 1993 года).

19 июля 1995 года в Закон «Об авторском праве и смежных правах» внесены изменения и дополнения.

1995 год Присоединение Российской Федерации к Всемирной Конвенции об авторском праве в редакции 1971 года (Парижская редакция) и Бернской Конвенции об охране литературных и художественных произведений 1886 года (в редакции Парижского акта 1979 года)

2004 год Внесены поправки в Закон Российской Федерации «Об авторском праве и смежных правах», в частности:

восстановлен режим так называемой ретроохраны, установленной Бернской Конвенцией,

скорректированы сроки предоставления охраны произведениям,

введены нормы, призванные обеспечить эффективное правовое регулирование использования объектов авторского права в цифровых сетях.

1 января 2008 года вступила в силу 4 часть Гражданского кодекса РФ.

Патентное право - подотрасль гражданского права, регулирующая правоотношения, связанные с созданием и использованием (изготовление, применение, продажа, иное введение в гражданский оборот) объектов интеллектуальной собственности, охраняемых патентом. Упомянутые объекты интеллектуальной собственности называются также промышленной собственностью.

Российское законодательство не содержит в явном виде определения патента, но на практике под патентом понимается документ, выдаваемый от имени государства лицу, подавшему заявку в установленном законом порядке, в подтверждение его прав на изобретение, полезную модель или промышленный образец. «Патент удостоверяет приоритет, авторство изобретения, полезной модели или промышленного образца и исключительное право на изобретение, полезную модель или промышленный образец». Под правом авторства понимается право признаваться автором изобретения. Под исключительным правом понимается то, что использование соответствующего объекта возможно либо самим правообладателем, либо с его прямого разрешения.

Объектами патентного права являются:

Изобретение. В качестве изобретения охраняется техническое решение в любой области, относящееся к продукту (в частности, устройству, веществу, штамму микроорганизма, культуре клеток растений или животных) или способу (процессу осуществления действий над материальным объектом с помощью материальных средств). Изобретению предоставляется правовая охрана, если оно является новым, имеет изобретательский уровень и промышленно применимо.

Полезная модель. В качестве полезной модели охраняется техническое решение, относящееся к устройству. Условиями патентоспособности полезной модели будут являться новизна и промышленная применимость. Законодатель не требует наличия изобретательского уровня для полезных моделей. Как видно из определения, в качестве полезной

модели может признаваться техническое решение, относящееся только к устройству, в отличие от изобретений, которыми, помимо устройства, могут быть вещество, штамм микроорганизма, культура клеток растений или животных, процесс осуществления действий над материальным объектом с помощью материальных средств.

Промышленный образец. В качестве промышленного образца охраняется художественно-конструкторское решение изделия промышленного или кустарно-ремесленного производства, определяющее его внешний вид. Промышленный образец сильно отличается от изобретения или полезной модели, он даже похож на один из объектов авторского права, поскольку имеет в совокупности с художественным решением также конструкторское. Промышленному образцу предоставляется охрана, если по своим признакам он является новым и оригинальным. Примером может служить стеклянная бутылка спрайта, имеющая оригинальный внешний вид изделия.

Лекция 11 Статистическая информация. Научно-техническая информация.

Вхождение России в мировое экономическое пространство, интенсивное развитие негосударственного сектора, становление рыночной инфраструктуры и другие коренные изменения в экономике обуславливают необходимость применения новых методов организации статснабжения, также разработки новых показателей и перехода и перехода на методологическую базу, используемую в мировой практике. Одной из важнейших задач статистики является охват статистического учета быстро увеличивающегося числа хозяйствующих субъектов.

Единая система государственной статистики включает 2250 организаций: на федеральном уровне 7 организаций, на региональном уровне 87 комитетов и управлений статистики, на районном уровне 2156 региональных отделов статистики. Основными задачами Госкомстата РФ являются:

разработка научно обоснованной статистической методологии, соответствующей потребностям общества на современном этапе, а также международным стандартам;

предоставление официальной статистической информации Президенту РФ, Правительству РФ, Федеральному собранию Госдумы РФ, федеральным органам исполнительной власти, общественности и международным организациям;

предоставление всем пользователям равного доступа к открытой статистической информации путем распространения официальных докладов, публикаций статистических сборников и других материалов.

На федеральном уровне осуществляется интеграция информационных ресурсов всей системы органов государственной статистики. Анализ экономических и социальных процессов проводится на базе показателей, содержащихся в государственной статистической отчетности, и на основе выборочных статистических обследований.

Важнейшим элементом отечественной информационной инфраструктуры является Всероссийский научно-технический информационный центр (ВНТИЦ). В хранилищах ВНТИЦ формируется национальный фонд отчетов о научно-исследовательских и опытно-конструкторских работах (НИР и ОКР), кандидатских и докторских диссертаций, алгоритмов и программ, обеспечивается их сохранность и общественно полезное использование в научно-производственной и культурной сфере страны.

Важнейшей составной частью национальных информационных ресурсов являются так называемые непубликуемые источники- научно-технические отчеты, диссертации и переводы Они (в первую очередь переводы) являются важным каналом научных коммуникаций, носителями информации, необходимой для решения практических задач. Непубликуемые источники научно-технической информации (НТИ) существуют всего в нескольких экземплярах, поэтому доступ к ним обеспечивается созданием на федеральном уровне централизованных фондов Выявлены следующие мировые тенденции использования и обращения неопубликованной научно-технической информации (ННТИ) на инфор-

мационном рынке, которые в большинстве своем имеют самое прямое отношение к деятельности ВНТИЦ:

ННТИ как «электронная» информация или как фактически публикуемая НТИ, которая может быть распространена любым тиражом в минимальные сроки,

ННТИ как коммерческая информация При заинтересованности коллективов в получении денежного вознаграждения в отчеты о НИР могут быть включены сведения об имеющихся «ноу-хау» или иные дополнительные сведения внедренческого харакгера. Федеральный информационный центр в этом случае может играть роль агента-посредника и получать свой процент от прибыли за реализацию конкретного информационного продукта,

ННТИ как форма защиты прав человека и интеллектуальной собственности. Зачастую единственной возможностью включить результаты исследований в информационно-коммуникационный процесс для определенной категории научно-технических работников является возможность поместить отчет о НИОКР, заявку или техническое предложение в тот или иной депозитарий ННТИ Например, опыт Японии показал, что покупка отвергнутых заявок является весьма прибыльным делом, «провоцирующим» научно-техническое творчество по аналогии или иным способом. При этом по отвергнутой патентной заявке, описанию изобретения или другому документу, помещенному в фонд идей, составляется регистрационная карта, подтверждающая дату приоритета и первичное содержание;

ННТИ как архивная информация;

ННТИ как культурно-историческое наследие.

Федеральным законом РФ от 29 декабря 1994 г № 77-ФЗ «Об обязательном экземпляре документов» на ВНТИЦ возлагается комплектование обязательных бесплатных экземпляров неопубликованных до-кументов, их регистрация и учет, выпуск информационных изданий о них, обеспечение их сохранности и использования ВНТИЦ готовит и распространяет следующие периодические информационные издания:

бюллетени регистрации НИР и ОКР по 28 тематическим сериям;

сборники рефератов НИР и ОКР по 28 тематическим сериям;

библиографический указатель информационных лист ков о научно-технических достижениях (перечень информационных и рекламных листков),

аналитические обзоры по актуальным научно-техническим проблемам.

В России имеется огромный не использованный прошлой системой инновационный потенциал (запас идей), который в значительной мере сосредоточен в фондах ВНТИЦ, однако в ходе реформ государство не смогло создать сектор обмена их на инвестиционные ресурсы. Поиск инвесторов научно-технической деятельности в настоящее время представляет собой очень сложный вопрос

К перспективным направлениям научных исследований ВНТИЦ следует отнести задачи освоения громадного наследия советской науки 1960-1980-х гг, когда были накоплены колоссальные научно-технические заделы, не востребованные в силу тяжелого финансового положения российской науки в 1990-х гг. ВНТИЦ является естественным хранителем этого ценнейшего ресурса и может служить ин-формационной базой при реализации программы обращения научного сообщества к освоению исследовательских заделов советского времени.

4.1.4. Вычислительные сети. Классификация сетей. Требования, предъявляемые к современным сетям. Функциональные устройства сети. Коммуникационные устройства. Топологии подключения устройств. Структуризация сети. 8

4.1.5. Стандарты сетей. Понятие «открытая система». Модель взаимодействия открытых систем (OSI). Организации по стандартизации. Спецификации сети Ethernet. Метод доступа, взаимодействие устройств в сети Ethernet. 6

Лекция 12 Понятие и классификация вычислительных сетей. Основные характери-

стики и требования современной вычислительной сети.

Вычислительной сетью называют совокупность компьютеров, соединенную линиями связи и коммуникационными устройствами. Все оборудование сети функционирует под управлением системного и прикладного программного обеспечения.

Информационно-вычислительные системы принято делить на три основных типа по территориальному признаку: LAN – *локальная сеть,* MAN – городская или региональная сеть, WAN – глобальная сеть.

Еще одним популярным способом классификации сетей являет их классификация по масштабу: локальные *сети рабочих групп,* локальные *сети отделов, сети кампусов, корпоративные сети*.

В зависимости от способа организации обработки данных и взаимодействия пользователей выделяют два типа сетей: иерархические сети; сети клиент-сервер;

В иерархических системах задачи хранение данных; организация доступа пользователей или программ к данным; обработка данных; передача данных или результатов обработки данных пользователям или программам; выполняются аппаратными и программными средствами одного основного компьютера. В архитектуре клиент/сервер часть этих задач решает сервер, а часть компьютер или программа, выступающая в качестве клиента. Сервер – это устройство или компьютер, выполняющий обработку поступившего от клиента запроса.

По организации взаимодействия принято выделять два типа систем, использующих метод клиент-сервер: равноправная сеть; сеть с выделенным сервером.

Равноправная сеть – сеть, в которой нет единого центра управления взаимодействием рабочих станций, нет единого устройства хранения данных. Операционная система такой сети распределена по всем рабочим станциям, поэтому каждая рабочая станция одновременно может выполнять функции как сервера, т.е. обслуживать запросы от других рабочих станций, так и клиента - направляет свои запросы другим рабочим станциям.

В сети с выделенным сервером один из компьютеров (сервер сети) выполняет функции хранения данных общего пользования, организации взаимодействия между рабочими станциями, выполнения сервисных услуг. На таком компьютере, как правило, выполняется сетевая операционная система, и все разделяемые устройства подключаются непосредственно к нему.

Достоинства сети с выделенным сервером: выше скорость обработки данных; надежная система зашиты информации и обеспечения секретности; простота в управлении по сравнению с равноправными сетями. Недостатки: быстродействие и надежность сети зависят от компьютера, используемого в качестве сервера сети; сеть с выделенным сервером менее гибкая по сравнению с равноправной.

Вычислительная сеть создается для обеспечения потенциального доступа к любому ресурсу сети для любого пользователя сети. Качество доступа к ресурсу может быть описано многими показателями, выбор которых зависит от задач, стоящих перед сетью.

Производительность вычислительной сети может быть оценена с разных позиций. С точки зрения пользователя, важным числовым показателем производительности сети является время реакции системы, т.е. время между моментом возникновения запроса и моментом получения ответа. Пропускная способность определяется количеством информации, переданной через сеть или ее сегмент в единицу времени и измеряемая в битах в секунду. Пропускная способность может быть мгновенной, максимальной и средней.

Задержка передачи – задержка между моментом поступления пакета на вход какого-либо устройства и моментом появления его на выходе.

Надежность работы вычислительной сети определяется надежностью работы всех ее компонентов. Для технических устройств используются такие показатели надежности, как среднее время обработки на отказ, вероятность отказа, интенсивность отказов. Для оценки надежности сложных систем применяют *готовность* или *коэффициент го-*

товности.

Для оценки надежности передачи пакетов используются показатели вероятности потери пакета при его передаче, либо вероятности доставки пакета. В современных сетях важна и другая сторона надежности – безопасность – способность сети обеспечить защиту информации от несанкционированного доступа. Еще одной характеристикой надежности является *отказоустойчивость.*

Любая вычислительная сеть является развивающимся объектом, и не только в плане модернизации ее элементов, но и в плане ее физического расширения, добавления новых элементов сети (пользователей, компьютеров, служб). Существование таких возможностей, трудоемкость их осуществления входят в понятие *расширяемости.* Другой похожей характеристикой является масштабируемость сети, которая определяет возможность расширения сети без существенного снижения ее производительности.

Прозрачность сети предполагает скрытие (невидимость) особенностей сети от конечного пользователя. Другой важной стороной прозрачности сети является возможность распараллеливания работы между разными элементами сети.

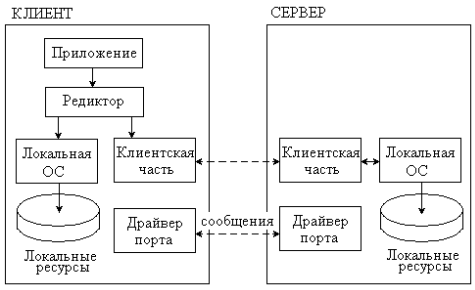
Интегрируемость (или *совместимость)* означает возможность подключения к вычислительной сети разнообразного и разнотипного оборудования, программного обеспечения от разных производителей.

Под коммутацией данных понимают их передачу, при которой канал данных может использоваться попеременно. Различают коммутация сообщений, каналов, пакетов.

Лекция 13 Функциональные устройства сети. Коммуникационные устройства.

Изучение сети в целом предполагает знание принципов работы ее отдельных компонентов: компьютеров, коммуникационного оборудования, операционных систем, сетевых приложений.

Для обмена данными между компьютером и периферийным устройством в компьютере предусмотрен внешний интерфейс, то есть набор проводов, соединяющих компьютер и периферийное устройство, а также набор правил обмена информацией по этим проводам (интерфейс или протокол). Примерами интерфейсов, используемых в компьютерах, являются параллельный интерфейс Centronics, предназначенный, как правило, для подключения принтеров, и последовательный интерфейс RS-232C, через который подключаются мышь, модем и много других устройств. Со стороны ПУ интерфейс чаще всего реализуется аппаратным устройством управления, хотя встречаются и программно-управляемые периферийные устройства. Схема взаимодействие программных компонентов при связи двух компьютеров приведена на рис.



Узел – любое устройство, подключенное к сети. Рабочая станция – это персональный компьютер, подключенные к сети, на котором пользователь сети выполняет свою работу. Удаленная рабочая станция – рабочая станция, подключенная к локальной сети через медленную линию связи, отличную от используемой в локальной сети.

Сервер сети – компьютер, подключенный к сети и выполняющий для пользователей сети определенные услуги. Часть сети, в которую не входит устройство расширения, принято называть сегментом сети.

Повторитель (хаб, концентратор) – устройство, позволяющее расширить сеть подключением дополнительных сегментов кабеля.

Мост – устройство, позволяющее объединить несколько сегментов, так что передача данных между станциями внутри одного сегмента не будет влиять на передачу данных в других сегментах. Коммутатор он является своего рода коммуникационным мультипроцессором, так как каждый его порт оснащен специализированным процессором, который

обрабатывает кадры по алгоритму моста независимо от процессоров других портов.

Мультиплексор – устройство, позволяющее мультиплексировать данные, приходящие одновременно от различных станций или сегментов и передавать их через передающую среду.

Маршрутизатор – устройство, соединяющее сети разного типа, но использующие одну сетевую операционную систему или протокол обмена данными.

Шлюз – устройство, позволяющее организовать обмен данными между сетевыми объектами, использующими различные протоколы обмена данными.

Линия связи в общем случае состоит из физической среды, по которой передаются электрические информационные сигналы, аппаратуры передачи данных и промежуточной аппаратуры. Синонимом термина линия связи является термин канал связи.

В зависимости от среды передачи данных линии связи разделяют на проводные, кабельные (медные, волоконно-оптические), радиоканалы наземной и спутниковой связи.

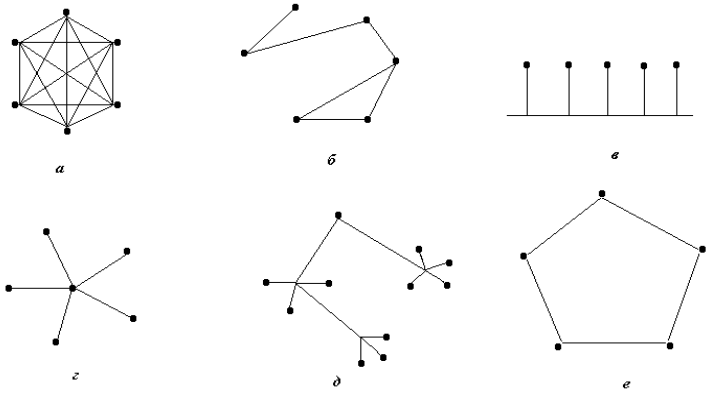
В компьютерных сетях применяются типы кабеля: на основе скрученных пар медных проводов, коаксиальные с медной жилой и волоконно-оптические. Основные характеристики кабельных линий связи: амплитудно-частотная характеристика, полоса пропускания, затухание, помехоустойчивость, перекрестные наводки на ближнем конце линии, пропускная способность, достоверность передачи данных, удельная стоимость.

Лекция 14 Топология и структуризация сети

Под топологией вычислительной сети понимается конфигурация графа, вершинам которого соответствуют компьютеры сети (иногда и другое оборудование, например концентраторы), а ребрам – физические связи между ними. Необходимо различать физическую и логическую топологии сети.

Конфигурация физических связей определяется электрическими соединениями компьютеров между собой. Логические связи представляют собой маршруты передачи данных между узлами сети и образуются путем соответствующей настройки коммуникационного оборудования.

При полносвязной топологии (рис. а) каждый компьютер сети связан со всеми остальными. Ячеистая топология получается из полносвязной путем удаления некоторых возможных связей (рис. б). Шина (рис. в) – передаваемая информация может передаваться в обе стороны. В топологии звезда (рис. г) каждый компьютер подключается отдельным кабелем к общему устройству, называемому концентратором. Иногда имеет смысл строить сеть с использованием нескольких концентраторов,



иерархически соединенных друг с другом связями типа звезда (рис. д). В сетях с кольцевой конфигурацией (рис. е) данные передаются по кольцу, как правило, в одном направлении. В то время как небольшие сети, как правило, имеют типовую типологию, для крупных сетей характерно наличие произвольных связей между узлами. Их называют сетями смешанной топологии.

Сотовая топология метод разделения географической области на зоны, в каждой из которых обеспечивается обмен информацией между станциями сети, находящимися внутри зоны.

Физическая структуризация сети с помощью концентраторов полезна не только для увеличения расстояния между узлами сети, но и для повышения ее надежности.

Сеть с типовой топологией (шина, кольцо, звезда), в которой все физические сегменты рассматриваются в качестве одной разделяемой среды, оказывается неадекватной структуре информационных потоков в большой сети. Решение проблемы состоит в отказе от идеи использования единой однородной разделяемой среды. Логическая структуризация сети – это процесс разбиения сети на сегменты с локализованным трафиком. Для логической структуризации сети используются такие коммуникационные устройства, как мосты, коммутаторы, маршрутизаторы и шлюзы.

Лекция 15 Стандартизация компьютерных сетей.

Работы по стандартизации вычислительных сетей ведутся большим количеством организаций. В зависимости от их статуса различаются и виды стандартов: стандарты отдельных фирм, стандарты специальных комитетов и объединений, национальные стандарты, международные стандарты.

К организациям наиболее активно и успешно занимающихся разработкой стандартов в области вычислительных сетей относятся:

Международная организация по стандартизации (International Organization for Standardization (ISO);

Международный союз электросвязи (International Telecommunications Union, ITU) – организация, являющаяся специализированным органом ООН. Наиболее важную роль играет постоянно действующий в ней Международный комитет по телефонии и телеграфии (Consultative Committee on International Telegraphy and Telephony, CCITT). В 1993 году в результате реорганизации ITU он сменил название на сектор телекоммуникационной стандартизации ITU (ITU Telecommunications Standardization Sector – ITU-T).

Институт инженеров по электротехнике и радиоэлектронике – Institute of Electrical and Electronics Engineers, IEEE – национальная организация США, определяющая сетевые стандарты.

Ассоциация электронной промышленности (Electronic Industries Association, EIA) – промышленно-торговая группа производителей электронного и сетевого оборудования, национальная коммерческая ассоциация США.

Американский институт национальных стандартов (American National Standards Institut, ANSI) – представляет США в ISO.

В широком смысле открытой системой может называться любая система (компьютер, вычислительная сеть, аппаратные или программные продукты), которая построена в соответствии с открытыми спецификациями. Термин «спецификация» означает формализованное описание аппаратных или программных компонент, способ их функционирования, взаимодействия с другими компонентами, условий эксплуатации, ограничений и особых характеристик. Не всякая спецификация является стандартом. Открытые спецификации – опубликованные, общедоступные, соответствующие стандарту и принятые после всестороннего обсуждения.

Для обеспечения обмена данными между компьютерными сетями ISO совместно с CCITT разработала многоуровневый комплект протоколов, известный как эталонная мо-

дель взаимосвязи открытых систем (Open System Interconnection – OSI) (1977 г).

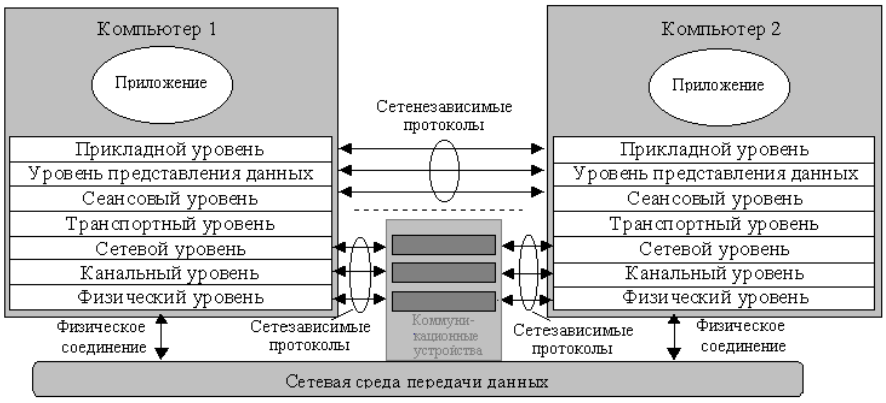
Одна из основных ее идей – обеспечение относительно легкого и простого обмена информацией при использовании изготовленных разными фирмами аппаратных и программных средств, соответствующих стандартам OSI.

Модель OSI касается только одного аспекта открытости – открытости взаимодействия устройств, связанных в вычислительную сеть.

Ярким примером открытой системы является Internet.

Лекция 16 Модель взаимосвязи открытых систем. Стеки коммуникационных протоколов

Все задачи, которые необходимо решить для организации взаимодействия между объектами информационной системы, разделены на семь отдельных процедур или уровней, представленных на рисунке. Каждый уровень выполняет определенную логическую функцию и обеспечивает определенный набор услуг для расположенного над ним уровня. Отдельные уровни модели OSI удобно рассматривать как группы программ, предназначенных для выполнения конкретных функций. При обращении некоторого сетевого приложения к прикладному уровню на основании запроса программное обеспечение формирует сообщение стандартного формата, состоящее из заголовка и поля данных. После формирования сообщения прикладной уровень направляет его вниз. Протокол каждого уровня на основании информации, полученной из заголовка вышележащего уровня, выполняет требуемые действия и добавляет к сообщению собственную служебную информацию. Когда сообщение по сети поступает на станцию-адресат, оно принимается физическим уровнем и перемещается вверх. Последовательно каждый уровень анализирует и обрабатывает заголовок своего уровня, выполняя соответствующие данному уровню функции, а затем его удаляет.



Три нижних уровня – физический, канальный, сетевой - являются сетезависимыми, то есть протоколы этих уровней тесно связаны с технической реализацией сети и используемым коммуникационным оборудованием.

Три верхних уровня – прикладной, представительский и сеансовый – ориентированы на приложения и мало зависят от технических особенностей построения сети. На протоколы этих уровней не влияют изменения в топологии сети, замена оборудования или переход на другую сетевую технологию.

Транспортный уровень является промежуточным, он скрывает все детали функционирования нижних уровней от верхних, что позволяет разрабатывать приложения, независящие от технических средств непосредственной транспортировки сообщений.

В стандартах ISO для обозначения единиц данных, с которыми имеют протоколы разных уровней, используется общее название протокольный блок данных (*Protocol Data Unit, PDU*). Для обозначения блоков данных определенных уровней используются специальные названия: кадр (frame), пакет (packet), дейтаграмма (datagram), сегмент (segment).

Важнейшим направлением стандартизации в области вычислительных сетей явля-

ется стандартизация коммуникационных протоколов. Наиболее популярными являются стеки: TCP/IP, IPX/SPX, NetBIOS/SMB, DECnet, SNA, OSI.

Все стеки, кроме SNA, на физическом и канальном уровнях используют одни и те же хорошо стандартизированные протоколы Ethernet, Token Ring, FDDI и позволяют использовать во всех сетях одну и ту же аппаратуру. На верхних уровнях все стеки работают по своим собственным протоколам, которые часто не соответствуют рекомендуемому моделью OSI разбиению на уровни. В таблице показано соответствие популярных стеков протоколов модели OSI.

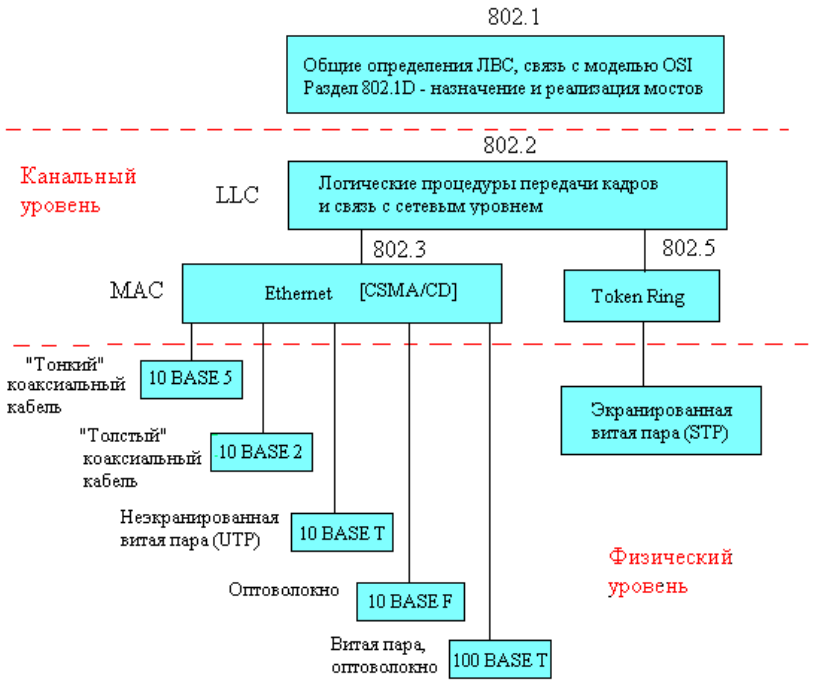
<table>
<tr><th>модель OSI</th><th>IBM/
Microsoft</th><th>TCP/IP</th><th>Стек OSI</th></tr>
<tr><td>Прикладной</td><td rowspan="2">SMB</td><td rowspan="2">telnet
FTP
SMPT
SNPT
WWW</td><td>X.400
X.500
FTAM</td></tr>
<tr><td>Представительский</td><td>Представительный протокол модели OSI</td></tr>
<tr><td>Сеансовый</td><td rowspan="2">NetBIOS</td><td rowspan="2">TCP</td><td>Сеансовый протокол модели OSI</td></tr>
<tr><td>Транспортный</td><td>Транспортный протокол модели OSI</td></tr>
<tr><td>Сетевой</td><td></td><td>IP
RIP
OSPF</td><td>ES-ES
IS-IS</td></tr>
<tr><td>Канальный</td><td colspan="3">Ethernet, Token Ring, FDDI, Fast Ethernet, SLIP, X.25, ATM, PPP, 100VG-AnyLAN</td></tr>
<tr><td>Физический</td><td colspan="3">Коаксиальный кабель, экранированная и неэкранированная витая пара, волоконно-оптический кабель, радиоволны</td></tr>
</table>

Стек TCP/IP – один из самых распространенных стеков транспортных протоколов вычислительных сетей. Основными протоколами стека являются, давшие ему название, протоколы IP и TCP. Они относятся соответственно к сетевому и транспортному уровням. IP обеспечивает продвижение пакета по составной сети, TCP гарантирует надежность его доставки. TCP/IP вобрал в себя популярные протоколы прикладного уровня – протокол пересылки файлов FTP, протокол эмуляции терминала telnet, почтовый протокол SMPT. Особенностью технологии TCP/IP является гибкая система адресации, позволяющая достаточно просто включать в интерсеть сети других технологий. Однако мощные функциональные возможности стека TCP/IP требуют для своей реализации высоких вычислительных затрат и предъявляют высокие требования к администрированию IP-сетей.

Лекция 17 Стандарты и технологии локальных сетей.

В 1980 году в институте IEEE был организован комитет 802 по стандартизации локальных сетей. Результатом его работы стало семейство стандартов IEEE 802.x, содержащих рекомендации по проектированию нижних уровней локальных сетей.

Стандарты IEEE 802 имеют достаточно четкую структуру, приведенную на рисунке.



Специфика локальных сетей нашла свое отражение в разделении канального уровня на два подуровня:

- логической передачи данных (Logical Link Control – LLC),
- управления доступом к среде (Media Access Control, MAC).

Уровень MAC появился из-за существования в ЛВС разделяемой среды передачи данных. Этот уровень обеспечивает корректное совместное использование общей среды, предоставляя ее в соответствии с определенным алгоритмом в распоряжении той или иной станции сети. После того как доступ к среде получен, ею может пользоваться более высокий уровень – LLC, организующий передачу логических единиц данных с различным уровнем качества транспортных услуг. В современных ЛВС получили распространение несколько протоколов уровня MAC, реализующих различные алгоритмы доступа к передающей среде. Эти протоколы полностью определяют специфику отдельных технологий.

Стандарты 802.3, 802.4, 802.5 и 802.12 описывают технологии локальных сетей, полученные в результате улучшений фирменных технологий, легших в их основу. К други известным стандартам относятся:

802.6 – Metropolitan Area Network , MAN – сети мегаполисов;

802.8 – Fiber Optic Technical Advisory Group – техническая консультационная группа по волоконно-оптическому кабелю;

802.9 – Integrated Voice and data Networks – интегрированная среда передачи голоса и данных;

802.10 – Network Security – сетевая безопасность;

802.11 – Wireless Networks – беспроводные сети.

Ethernet – самый распространенный в настоящее время стандарт локальных сетей.

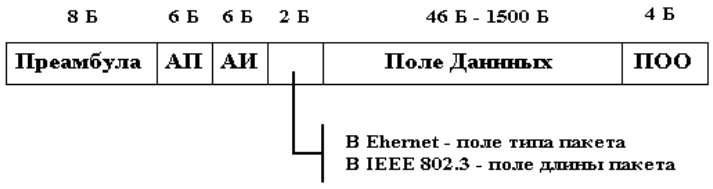
Ethernet версии 1 появилась в 1980 г. В 1982 году была опубликована спецификация на Ethernet версии 2.0. Обе версии используются до сих пор, причем между ними существуют различия и по интерфейсу, и по уровню сигналов. На базе Ethernet версии 2 институтом IEEE был разработан стандарт IEEE 802.3. В 1995 году был принят стандарт Fast Ethernet, который во многом не является самостоятельным стандартом. Его описание 802.3u является дополнительным разделом к основному стандарту. Аналогично, принятый 1998 году стандарт Gigabit Ethernet описан в разделе 802.z основного документа. В таблице приведены данные «классических» технологий сети Ethernet.

Ethernet	Thick Wire Ethernet	Thin Wire Ethernet	UTP Ethernet	Fiber Optic Ethernet	Broadband Ethernet
IEEE 802.3	10BASE-5	10BASE-2	10BASE-T	10BASE-F	10BROAD36
Скорость передачи данных, Мбит/с	10	10	10	10	10
Метод передачи сигналов	Одно-Полосной	Одно-полосной	Одно-полосной	Одно-полосной	Широко-полосной
Длина сегмента кабеля,м	500	185	100	2000	1800

Максим. расстояние между узлами сети (при использовании повторителей), м	2500	925	500	2500	3600
Максимальное число станций на сегменте	100	30	1024	1024	100
Максим. число повторителей Между любыми станциями сети	4	4	4	4	4
Тип кабеля	коаксиальный, «толстый» RG-8 или RG-11	коакси-альный, «тонкий» RG-58	Неэкранированная витая пара категории 3, 4, 5	Многомодовый волоконно-оптический кабель	75 Омный коаксиальный, «толстый»

Во всех конфигурациях используется один метод доступа станций к среде – множественный доступ с контролем несущей и обнаружением коллизий – Carrier Sense Multiple Access/Collision Detection (CSMA/CD).

Во время работы станция постоянно проверяет среду передачи. Передающая среда может быть свободна, если ни одна другая станция не передает данные, или занята. Любая станция, обнаружив, что среда свободна, может начать передачу своих данных. Поэтому возможна ситуация, когда одновременно несколько станций начнут передавать данные – коллизия. При этом происходит смешение и искажение сигналов. Четкое распознавание коллизий всеми станциями сети является необходимым условием корректной работы сети Ethernet. Если какая-либо станция не распознает коллизию и решит, что переданный ею кадр верен, то кадр будет утерян или значительно искажен. Для надежного распознавания коллизий должно выполняться соотношение: $T_{min} \geq PDV$, где T_{min} – время передачи кадра минимальной длины, а PDV – время, за которое сигнал коллизии успевает распространиться до самого дальнего узла сети – время двойного оборота.



Минимальная длина пакета в IEEE 802.3 и Ethernet - 64 байта, соответственно длина поля данных - 46 байтов. IEEE 802.3 позволяет использовать символы-заполнители (PAD) в поле данных, если требуется передать сообщение короче, чем 46 байтов. В Ethernet пакет с длиной поля данных менее 46 байтов просто не будет обрабатываться. На рисунке показана структура пакета в сети Ethernet.

Назначение полей:

Преамбула служит для синхронизации работы приемника и передатчика.

АП - Адрес Приемника адрес станции, которой направляется пакет.

АИ - Адрес Источника - адрес передающей станции.

Поле Типа Пакета определяет тип Ethernet пакета.

Поле Длины Пакета определяет в IEEE 802.3 количество байт данных в поле данных. Количество символов-заполнителей (PAD) не входит в это число.

Поле Данных содержит данные и символы-заполнители в IEEE 802.3, данные - в Ethernet.

ПОО Поле Обнаружения Ошибок служит для определения достоверности полученной информации.

IEEE 802.3 определяет, что после поля адреса источника следует двухбайтное поле длины пакета, а в Ethernet - поле типа пакета

Лекция 18 Высокоскоростные варианты сети Ethernet

К ранним высокоскоростным вариантам относятся: коммутированная Ethernet, ду-

плексная Ethernet, 100BaseVG AnyLAN. Первые две использовались лишь в качестве временного решения, в 100BaseVG AnyLAN используется другой метод доступа – обработка запросов по приоритету, что усложняет совместимость с существующими сетями Ethernet.

Технология 100BaseX или Fast Ethernet реализуется с помощью дуплексной передачи сигналов по традиционной для Ethernet схеме доступа с контролем несущей и обнаружением коллизий в комбинации с уровнем зависимости от физической среды. Она принята в 1995 году в качестве дополнения 802.3u к существующему стандарту 802.3.

100BASE-T - общий термин для обозначения стандартов, использующих в качестве среды передачи данных витую пару. Длина сегмента до 100 метров. Включает в себя стандарты 100BASE-TX, 100BASE-T4 и 100BASE-T2.

100BASE-TX, IEEE 802.3u – развитие стандарта 10BASE-T для использования в сетях топологии "звезда". Задействована витая пара категории 5, фактически используются только две неэкранированные пары проводников, поддерживается дуплексная передача данных, расстояние до 100 м.

Более сложная структура физического уровня технологии Fast Ethernet вызвана тем, что в ней используется три варианта кабельных систем: волоконно-оптический многомодовый кабель (2 волокна), 2-х парная витая пара категории 5; 4-х парная витая пара категории 3. Fast Ethernet всегда имеет иерархическую древовидную структуру, построенную на концентраторах, подобно технологиям 10Base-T и 10Base-F. Диаметр сети сокращен до 200 метров, что объясняется уменьшением времени передачи кадра минимальной длины.

При использовании коммутаторов протокол Fast Ethernet может работать в полнодуплексном режиме, в котором нет ограничения на длину сети, а существуют лишь ограничения на длину физических сегментов. Формат кадра не отличается от формата 10-мегабитного Ethernet.

Гигабит Ethernet – поддерживает скорость обмена до 1 Гбит/с. Сеть Gigabit Ethernet представляет собой ЛВС, в которой применяется многомодовый оптоволоконный кабель, но IEEE изучает методы ее применения на сетях с UTP 5 категории. Топология звездообразная. Оптоволоконный кабель используется для магистрали, коаксиальный кабель для отводов к концентраторам.

1000BASE-T, IEEE 802.3ab – стандарт, использующий витую пару категорий 5e. В передаче данных участвуют все 4 пары. Скорость передачи данных – 250 Мбит/с по одной паре. Используется метод кодирования PAM5, частота основной гармоники 62,5 МГц.

Новый стандарт 10 Гигабит Ethernet включает в себя семь стандартов физической среды для LAN, MAN и WAN. В настоящее время он описывается поправкой IEEE 802.3ae и должен войти в следующую ревизию стандарта IEEE 802.3.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ

Методические указания по изучению дисциплины

Для успешного освоения данной дисциплины студентам предлагаются:

- содержание разделов и тем дисциплины, включающей лекционные и лабораторные занятия;
- контрольные вопросы для самостоятельной работы;
- список рекомендуемой основной и дополнительной литературы;
- вопросы к экзамену.

В течении семестра студент не только должен изучать материал лекций, но и готовить вопросы самостоятельной работы на основе рекомендуемой литературы.

Основными формами самостоятельной (внеаудиторной) работы студентов являются:

подготовка отдельных вопросов по темам программы;

участие в научных и научно-практических студенческих конференциях;

подготовке к лабораторной работе по определенной теме;

подготовка и написание отчетов к лабораторным работам;

подготовка к промежуточному и итоговому контролю.

Самостоятельная работа начинается до прихода студента на лекцию. Весьма эффективно использование «системы опережающего чтения», т.е. предварительного прочитывания лекционного материала, содержащегося в учебниках, учебных пособиях, в результате чего закладывается база для более глубокого восприятия лекции.

В процессе организации самостоятельной работы большое значение имеют консультации преподавателя, в ходе которых решаются многие проблемы изучаемого курса, уясняются наиболее сложные вопросы.

Контроль самостоятельной работы осуществляется тестированием, которое может проводиться в письменной форме, либо с использованием компьютерной системы тестирования. Примерные тестовые задания приведены в рабочей программе. Тест включает в себя вопросы с открытыми и закрытыми вариантами ответов. Результаты тестирования включены в балльно-рейтинговую систему оценки знаний (см. рабочую программу).

Экзамен является завершающим этапом учебного процесса, на котором проводится подведение итогов всей самостоятельной работы студентов.

Подготовку к экзамену требуется начинать с просмотра перечня всех вопросов с целью оценки требуемого объема учебного материала, логики и структуры построения курса. С учетом накопленных за семестр знаний студент должен запланировать распределение времени на подготовку. Желательно зарезервировать время для повторения материала. Работа над каждым из вопросов рекомендуется прочитать конспект лекции, дополнительно прочитать рекомендованный учебник, если материал трудно усваивается. Завершается работа восстановление в памяти прочитанного.

Экзаменационный билет включает в себя два теоретических вопроса из перечня.

При оценке на экзамене учитывается: полнота ответа на поставленный вопрос, точность формулировок, логичность ответа, умение делать выводы, выявлять закономерности, соблюдение норм литературной речи и использования специализированной терминологии. Высшего балла заслуживает ответ, удовлетворяющий всем этим требованиям.

Методические указания к лабораторным работам

Лабораторные работы проводятся по подгруппам в компьютерном классе. Каждый студент получает индивидуальное задание в соответствии с вариантом.

Выполняя задание, студент пользуется материалом, изученным в тексте лабораторной работы.

Перед созданием любой программы требуется точно продумать алгоритм. Записать его блок-схемой или словесно. Надо четко определить, что в нее требуется ввести и что получить в результате, в какой последовательности выполнять действия. В случае необходимости выделить циклические структуры и подпрограммы. В циклах четко определить параметры, задать их начальные значения, определить условия повторения и завершения цикла. В функциях определить количество передаваемых и возвращаемых значений.

При кодировании программы нужно определить тип используемых данных в зависимости от возможного диапазона принимаемых значений. При вводе величины не забывать осведомить об этом пользователя, а иногда сообщить и о типе, диапазоне или порядке ввода значений. Такое сообщение должно быть информативно и коротко. Вывод данных лучше сопровождать текстом и форматированием. Формат вывода можно уточнить при помощи модификаторов.

В именах переменных необходимо отражать их назначение, что повышает читаемость и понимание программы.

При записи сложных выражений нужно обращать внимание на приоритет операций. Текст программы лучше сопровождать краткими и информативными комментариями, что облегчает как понимание программы, так и ее отладку.

Объявление локальных переменных предпочтительнее по сравнению с глобальными.

Для отладки программы нужно запустить ее на выполнение несколько раз, задавая различные значения вводимых величин. Перед запуском необходимо иметь заранее подготовленные тестовые примеры, содержащие исходные данные и ожидаемые результаты. Их количество зависит от алгоритма. Проверьте реакцию программы на заведомо неверные исходные данные.

Для быстрого поиска ошибки в алгоритме рекомендуется выводить промежуточные данные.

При сдаче лабораторной работы студент должен продемонстрировать преподавателю созданную программу, правильно работающую, отлаженную.

Преподаватель, принимая лабораторную работу, тестирует программу студента и задает ему вопросы по конструкциям, используемым в программе и теоретическим основам программирования.

ЗАДАНИЯ К ЛАБОРАТОРНЫМ РАБОТАМ

Лабораторная работа 1 **Основы JavaScript**

Для выполнения программ JavaScript в качестве интерпретатора (исполнительной системы) можно использовать веб-браузер Internet Explorer 6.0 (5.0). В качестве редактора программ текстовый редактор, например Блокнот. Создавать программы на JavaScript можно и с помощью специальных программ, предназначенных для разработки веб-сайтов, например Microsoft FrontPage.

Можно легко создать и собственный редактор программ, например кодом:

```
<HTML>
<H3>Редактор кодов</H3>
код:<br>
<TEXTAREA id="mycode" ROWS=10 COLS=60></TEXTAREA>
<p>Результат:<br>
<TEXTAREA id="myrezult" ROWS=3 COLS=60></TEXTAREA>
<P>
<BUTTON onclick="document.all.myrezult.value=eval(mycode.value)">
Выполнить</BUTTON>
<BUTTON onclick="document.all.mycode.value=' ';
```

```
document.all.myrezalt.value=' ' ">
Очистить </BUTTON>
<P>
<!Комментарий>
Введите выражения в верхнее поле.
Выражения разделяются точкой с запятой.
Можно также каждое выражение писать в отдельной строке
</HTML>
```

Ввод-вывод данных

Для обеспечения ввода-вывода JavaScript предоставляет несколько методов: alert( ), confirm ( ), prompt( ).

Первый из перечисленных выводит на экран диалоговое окно с заданным сообщением и кнопкой ОК.

Синтаксис данного метода:

alert (сообщение)

Сообщение может представлять собой данные любого типа: последовательность символов, заключенную в кавычки, число, переменную или выражение.

Текст необходимо заключить в кавычки. Например,

alert (“Всем привет! ”)

Для формирования строк используют служебные символы:

\n – новая строка,
\t – табуляция,
\f – новая страница,
\b – забой,
\r – возврат каретки.

Окно, создаваемое alert( ) является модальным (останавливающим все последующие действия программы и пользователя). Его можно убрать, щелкнув по кнопке ОК.

Метод confirm выводит на экран диалоговое окно с сообщением и двумя кнопками – ОК и Отмена. Этот метод возвращает логическую величину, значение которой зависит от того, по какой из кнопок щелкнет пользователь. Возвращаемое значение можно обработать в программе, создавая тем самым интерактивный эффект. Синтаксис применения данного метода аналогичен синтаксису метода alert.

Окно, создаваемое confirm также является модальным.

Метод prompt осуществляет вывод диалогового окна с сообщением и кнопками ОК и Отмена, а также с текстовым полем, в которое пользователь может ввести данные. В отличие от alert( ) и confirm( ) данный метод принимает два параметра: сообщение и значение, которое должно появиться в текстовом поле ввода данных по умолчанию. Если пользователь щелкнет по кнопке ОК, метод вернет содержимое поле ввода данных, если – по кнопке Отмена, то возвращается значение ложь. Возвращаемое значение можно также обработать в программе. Синтаксис метода:

prompt (сообщение, значение_поля_ввода данных)

Оба параметра не являются обязательными. Если они не указаны, на экране появится окно без сообщения, а в поле ввода данных подставлено значение по умолчанию – undefined (не определено). Чтобы значение по умолчанию не появилось, в качестве второго параметра указывается пустая строка (“ ”).

Типы данных

Типы данных языка JavaScript приведены таблице 1. При создании программ за типом данных следит сам программист. Интерпретатор не выдаст ошибки при неверном их использовании. Он просто попытается привести данные к типу, требуемому в данной операции.

Тип данных	*Описание значений*
Строковый или символьный тип (string)	Последовательность символов, заключенная в кавычки, двойные или одинарные
Числовой (number)	Положительное или отрицательное число. Целая и дробная части разделяются точкой
Логический	Два значения: true или false
Null	Отсутствие какого-либо значения
Объект (object)	Программный объект с собственными свойствами. В частности, массив также является объектом.
Функция (function)	Программный код, выполнение которого может возвращать некоторое значение.

Табл.1. Типы данных JavaScript

Для преобразования строк в числа предусмотрены встроенные функции parseInt( ) и parseFloat( ). Синтаксис:

parseInt(строка, основание)

parseFloat(строка, основание)

Если основание не указано, то предполагается 10 – десятеричная система счисления. В качестве основания можно также использовать 8, 10, 16.

При преобразовании строки в целое число округление не происходит – дробная часть просто отбрасывается.

Для определения того, является ли значение выражения числом, служит встроенная функция isNaN(значение). Функция возвращает логический тип. Данная функция считывает числа, данные числового типа и строки, содержащие только числа.

Переменные

Имя переменой представляет собой конечную последовательность символов, содержащую буквы, цифры, символ подчеркивания. Имя переменной не должно начинаться с цифры или содержать пробелы. Для имен переменных нельзя использовать ключевые слова языка.

В отличие от многих других языков программирования переменной не нужно задавать тип при объявлении. Тип переменной определяется типом ее значения. Переменная может принимать значения разных типов и неоднократно его изменять.

Создавать переменную в программе можно несколькими способами. Можно ей просто присвоить значение с помощью оператора присваивания в формате: имя_переменной = значение

Например,

Month= “Январь”

Можно использовать ключевое слово var перед именем переменной. В этом случае переменная не будет иметь первоначальное значение, но в дальнейшем его можно передать с помощью оператора присваивания. Например,

var Month

Month = “Январь”

При использовании var допускается и инициализация переменной, например:

var Month = “Январь”

Для каждой переменной инициализация при объявлении возможна только один раз.

Можно сразу объявить несколько переменных, используя var и разделяя их запятой, при этом возможно инициализировать их все или некоторые:

var Month= “Январь”, day, pi=3.14, x

Комментарии

В JavaScript допустимы два вида операторов комментария:
- одна строка символов, расположенная справа от //
- произвольное количество строк, заключенных между /* и */.

Операции выполняются в соответствии с приоритетами. Приоритет операций аналогичен языку С++. Для изменения порядка выполнения операций используются круглые скобки.

Операторы ветвления

Условный оператор if используется для разветвления процесса вычислений на два направления. Синтаксис оператора if:

```
if ( условие ) оператор1  else оператор 2
```

Пример

```
if ( fvalue>=0.0 )  fvalue = fvalue  else fvalue = -fvalue
```

Условные операторы могут быть вложенными.

```
if ( a<b ) { if ( a<c ) m = a  else m = c }  else { if ( b<c ) m = b  else m = c }
```

Необходимо помнить, что в этом случае else относится к ближайшему if. Операторные скобки после первого if необязательны.

Если требуется проверить несколько условий, их объединяют знакам логических операций.

Оператор switch (переключатель) предназначен для разветвления процесса вычислений на несколько направлений. Синтаксис оператора:

```
switch ( выражение) {
case  константное выражение 1  : операторы1   break
case  константное выражение 2  : операторы2   break
…
case  константное выражение n  : операторыN   break
default : операторы }
```

Параметр выражения оператора switch может принимать строковые, числовые и логические значения. В следующем примере переменная x содержит название языка, который выбрал пользователь. А выражение window.open ( ) открывает новое окно браузера и загружает в него указанный в скобках HTML-документ.

```
 switch (x ) {
        case " английский" : window.open ( "engl.htm") ; break
        case  " французский"  : window.open ( "french.htm") ;     break
        case  " русский" :  window.open ( "russ.htm") ;           break
        default  :  alert(  "Нет документа на таком языке" )
}
```

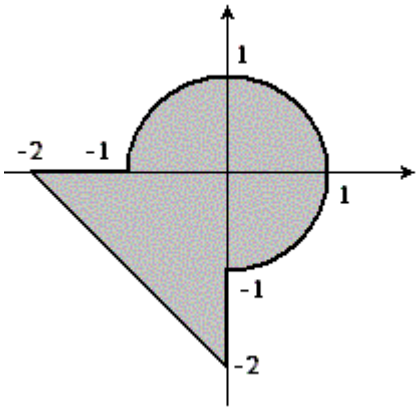
Задание

1. Создать собственный редактор программ, на основе HTML-кода, модифицировав код, приведенный выше. Файл открыть в браузере как веб-страницу.
2. Протестировать редактор, изучив в нем функции вывода и основные операции JavaScript.
3. Составить программу на основе разветвляющего алгоритма для задачи:

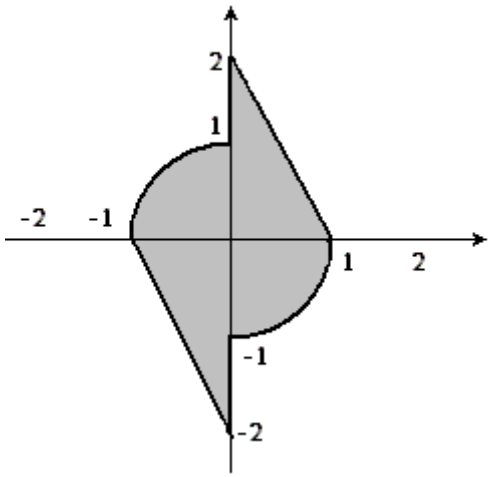
Дана «мишень» в виде закрашенной области, изображенной на рисунке. Создать алгоритм для определения, попадает ли заданная точка с координатами (x, y) в мишень.

4. Для удобства пользователя рисунок мишени отобразить в редакторе.

Вариант 1



Вариант 2



Лабораторная работа №2 **Операторы цикла. Организация пользовательских функций**

JavaScript содержит следующие встроенные функции (некоторые уже были рассмотрены):

parseInt(строка, основание) – преобразует указанную строку в целое число по указанному основанию(8, 10, 16); по умолчанию - десятеричная система.

parseFloat(строка, основание) – преобразует строку в число с плавающей точкой.

isNaN(строка, основание) – возвращает true, если указанное в параметре значение не является числом.

eval(строка) – вычисляет выражение в указанной строке, выражение не должно содержать тегов HTML.

Например,

```
var a = 10;                          // значение a равно 10
var b = “if (a<=25) { a *= 2 }”      // значение b равно строке символов
eval (b)                             // значение a равно 20
```

Другой пример применения данной функции в тексте программы редактора. Там сценарии записаны в качестве значений атрибутов onclik, определяющих событие щелчок кнопкой мыши на HTML-кнопках, заданных тегами <button>.

escape(строка) – возвращает строку в виде %XX, где XX – ASCII-код указанного символа. Такую строку называют escape-последовательностью.

unescape(строка) – обратное преобразование.

type0f (объект) – возвращает тип указанного объекта в виде символьной строки, например “boolean”, “function”.

В программах на JavaScript пользователям также разрешено создавать собственные функции. Объявление функция состоит из заголовка и тела:

```
function имя_функции ( параметры )          //заголовок функции
{ тело функции }
```

Тело функции ограничивается фигурными скобками. Параметры функции, стоящие в круглых скобках перечисляются через запятую.

Возвращение значений из функции происходит с помощью оператора return, за которым помещается само возвращаемое значение.

Для вызова функции можно воспользоваться выражениями вида:
имя_функции (параметры) или
имя_переменной = имя_функции(параметры)

Параметры в вызове функции должны быть представлены конкретными значениями.

Например, если описание функции дано в виде:

```
function cube (x)
{ return x*x*x }
```

Данная функция вычисляет куб от параметра функции. Ее вызов может быть записан в виде:

```
y=cube(25)
```

В JavaScript функции могут вызываться как после их определения, так и до него. Можно не поддерживать соответствие количества параметров в определении функции и в ее вызове. Если в определение функции параметров больше, чем в вызове, то недостающим параметрам автоматически присваивается значение null. Лишние параметры в вызове функции игнорируются.

Внутри функции можно создавать переменные с помощью оператора присваивания или с помощью ключевого слова var.

Если в теле функции переменная в составе оператора присваивания встречается впервые в программе, или она была определена до этого – она действует как глобальная.

Если в теле функции используется переменная, объявленная только во внешней программе, она также является глобальной.

Если для определения переменной в теле функции используется ключевое слово var, она будет локальной вне зависимости от того определена она во внешней программе или нет.

Операторы цикла

Как и другие языки программирования JavaScript имеет три вида оператора цикла: цикл с предусловием (while), цикл с постусловием (do while), цикл с параметром (for).

Синтаксис цикла с предусловием:

```
while ( условие )
{
операторы
}
```

Выражение, стоящее в круглых скобках, определяет условие повторения тела цикла, представленного простым или составным оператором. Если оператор простой операторные скобки { } могут не ставиться.

Выполнение оператора цикла начинается с вычисления выражения. Если оно истинно, выполняется тело цикла. Если при первой проверке выражение ложно (false), цикл не выполнится ни разу.

Значение выражения вычисляется перед каждой итерацией цикла.

Пример. Программа вычисляет возведение 10 в степень 5.

```
var x =10
y= 2
while ( y <=  5 )
        {
        x*=10;  y++
        }
```

Цикл while обычно используется в тех случаях, когда число повторений заранее не известно. В этом случае используется и оператор цикла с постусловием. Его отличительной чертой является выполнение тела цикла хотя бы один раз. И только после первого его выполнения проверяется, надо ли его выполнять еще раз. Таким образом, даже если условие заведомо ложно цикл выполняется один раз. Если условие истинно, тело цикло выполнится еще раз. Цикл завершается, когда выражение станет равным false или в теле цикла будет выполнен оператор передачи управления.

Цикл for называют также циклом с заданным числом повторений. Он имеет следующий формат:

```
for (инициализация; выражение (условие); модификации )
{
операторы
}
```

Инициализация используется для объявления и присвоения начальных значений величинам, используемым в цикле. Инициализация выполняется один раз перед выполнением тела цикла.

Выражение определяет условие выполнения цикла: если его результат равен истине, то цикл выполняется. Цикл с параметром реализуется как цикл предусловием.

Модификации выполняются после каждой итерации цикла и служат обычно для изменения параметров цикла.

Тело цикла представляет собой простой или составной оператор.

```
for (i = 1, s=1; i<11; i++)
{
s*=i;                    // вычисления факториала 10
}
```

Для принудительного выхода из тела любого цикла используются операторы break и continue. break позволяет переход в точку программы, находящуюся непосредственно за оператором, внутри которого находится, т.е. управление передается первой строке, следующей за телом цикла.

Например,

```
 time=1
 sum=0
while (time<10)
 { sum+=time
                if ( sum>20 )  break;
    time++
                    }
```

Инструкция continue заставляет программу пропустить все оставшиеся строки цикла, но сам цикл при этом не завершается. Для решения некоторых задач удобно комбинировать инструкции break и continue.

Задание

1. Составьте программу на основе циклического алгоритма для вычисления суммы ряда с заданной точностью ε. Определите и выведите на экран значение суммы и число элементов ряда, вошедших в сумму.

Номер варианта	***Задание***	***Точность***

1	$\frac{\pi}{3}+\sum_{n=1}^{\infty}(-1)^n\frac{(\pi/3)^{2n+1}}{(2n+1)!}$	$0.5*\ 10^{-4}$
2	$\sum_{n=1}^{\infty}(-1)^n\frac{(\pi/6)^{2n}}{(2n)!}$	$0.5*\ 10^{-4}$

Лабораторная работа №3 **Встроенные объекты JavaScript. Строки и массивы**

Объекты представляют собой программные единицы, обладающие заданными свойствами. Как любой объект, объект JavaScript, обладает свойствами (данными) и методами (функциями) для их обработки. Программный код встроенных объектов JavaScript недоступен.

Управление web-страницами с помощью сценариев, созданных на JavaScript, заключается в использовании и изменении свойств объектов HTML-документа и самого браузера.

Встроенные объекты имеют фиксированные названия. Наиболее важными объектами в разработке web-сайтов являются String (символьные строки), Array (массивы), Math (математические формулы и константы), Date (работа с датами).

Объекты, названия которых совпадают с их фиксированными названиями, называются статическими. Можно создавать и экземпляры (копии) статических объектов, имеющие собственные имена и наследующие все свойства и методы статических объектов.

Встроенные объекты имеют прототипы (prototype), позволяющие добавлять новые свойства и методы к уже существующим в экземплярах объектов.

Объект String

С помощью объекта String можно создавать строку или строковый объект. Синтаксис:

имя_переменной = new String (“значение”)

Создать строковый объект можно и с помощью обычного оператора присваивания:

имя_переменной = “значение”

или

var имя_переменной = “значение”

Например,

mystring1 = new String (“строка”)

mystring2 = “строка”

Свойство String

length – длина (количество символов), включая пробелы.

Доступ к свойствам и методам объекта осуществляется через операцию точка. Например,

str = “весна”

str.length // значение равно 5

“лето”.length // значение равно 4

Методы String

Как и любые функции, методы могут иметь параметры. Параметры перечисляются через запятую в круглых скобках, стоящих после имени метода.

chartAt (индекс) – возвращает символ, занимающий в строке указанную позицию. Индекс является числом. Необходимо помнить, что нумерация элементов строки начинается с нуля.

Примеры

y = “Осень”.charAt(3) // значение “н”

str = “Зима”
str.charAt (str . length – 2) // значение “м”

charCodeAt(индекс) – преобразует символ в указанной позиции в его код. Поддерживает систему кодов Unicode, NN4 – ISO-Latin1.

fromCharCode(номер [1, номер2[, номер3, …, номерn]]) – возвращает строку символов, числовые коды символов которой указаны в строке параметров.

concat(строка) – конкатенация (слияние) строк.

Синтаксис: строка1.concat(строка2)

Возвращает строку, полученную дописыванием символов строки2 к строке1.

Пример

x = “Петр”
y = x .concat (“ Семенович”) // результат “Петр Семенович”

index0f(строка_поиска [, индекс]) - поиск строки, указанной параметром. Метод возвращает индекс первого вхождения строки. Поиск в пустой строке возвращает -1. Второй параметр, не является обязательным, о чем говоря квадратные скобки. Индекс указывает позицию, с которой начинается поиск.

lastIndex0f (строка_поиска [, индекс]) – поиск первого вхождения строки, указанной параметром. Причем поиск начинается с конца исходной строки, но возвращаемый индекс отсчитывается сначала.

localeCompare (строка) – сравнение строк в кодировке Unicode, то есть с учетом используемого браузером языка общения с пользователем. Синтаксис:

строка1.localeCompare(строка2)

Если строки одинаковы, метод возвращает 0. Если строка1 меньше, чем строка2, метод возвращается отрицательное число, в противном случае - положительное.

slice(индекс1[, индекс2]) – возвращает подстроку исходной строки, начальный и конечный индексы которой указываются параметрами, за исключением последнего символа. Второй параметр не обязателен. Если он не указан – подразумевается до конца строки.

split(разделитель [, ограничитель]) – возвращает массив элементов, полученных из исходной строки. Первый параметр – строка символов, используемая в качестве разделителя строки на элементы. Второй параметр – число, указывающее количество элементов возвращаемого массива из строки, полученной при разделении. Второй параметр необязателен. Если разделитель – пустая строка, возвращается массив символов строки.

Например,

x = “Привет всем!”
x.split(“ ”) // значение – массив из элементов “Привет”, “всем!”
x. split(“е”) // значение – массив из элементов “Прив”, “т вс”, “м!”

substr(индекс[, длина]) – возвращает подстроку исходной строки, начальный индекс и длина, которой указываются параметрами. Если второй параметр не указан, возвращается подстрока с начальной позиции до конца.

substring(индекс1, индекс2) – возвращает подстроку исходной строки, с позиции1 до позиции2.

toLocaleLowerCase( ), toLowerCase( ) – переводят строку в нижний регистр.

toLocaleUpperCase( ), toUpperCase ( ) – переводят строку в верхний регистр.

Тексты web-страниц, как правило, создаются и форматируются с помощью тегов HTML. Это же можно сделать средствами JavaScript.

Например, для вывода строки полужирным шрифтом используют метод bold( ). Данный метод не выводит строку в окно браузера, а лишь форматирует ее. Для вывода строки в HTML-документе используется метод write( ) для объекта document:

```
<HTML>
<SCRIPT>
st = “Доброе утро!”.bold( )
document. write(st)
</SCRIPT>
</HTML>
```

Методы форматирования строк носят названия, соответствующие тегам HTML:

```
anchor(“anchor_имя”)
blinc( )
bold( )
fixed( )
fontcolor(значение цвета)
fontsize(число от 1 до 7)
italics( )
link(расположение или URL)
big( )
small( )
strike( )
sub( )
sup( )
```

Объект Array

Массив представляет собой упорядоченный набор данных. Нумерация элементов массива начинается с нуля. К элементам массива можно обращаться по их порядковому номеру, заключив его в квадратные скобки, расположенные после имени массива. Элементы массива в JavaScript могут быть разного типа.

Существует несколько способов создания массива:

1. имя_массива = new Array ([длина_массива])

Если длина массива не указана, создается пустой массив, не содержащий ни одного элемента. Иначе создается массив указанной длины, все элементы которого имеют значение null. Создав пустой массив, можно присвоить значения его элементам операцией присваивания.

2. инициализация массива при объявлении:

имя_массива = new Array (значение1[, значение2[, … значениен]])

3. инициализация каждого отдельного элемента массива, подобно свойствам объекта

```
имя_массива = new Array( )
имя_массива. имя_элемента1 = значение1
[имя_массива. имя_элемента2 = значение 2
[… имя_массива. имя_элемента n = значение n] ]
Например,
child = new Array (4)
child[0]= “Женя”
child[1]= 22
child[2]= “ июнь”
child[3]= 1996
child = new Array ( “Женя”,  22,  “ июнь”, 1996)
y=child.length// y=4

child = new Array ( )
```

```
child.name = “Женя”
child.day= 22
child.month= “ июнь”
child.year= 1996
```

Свойства объекта Array

Свойство length, возвращает количество элементов объекта Array.

Свойство prototype позволяет добавлять новые свойства и методы для всех созданных массивов.

Например,

```
function SumNegative (massiv)
{var s = 0
  for (i=0; i<=massiv.length-1; i++)
          if (massiv[i]<0 s +=massiv[i]
         return s
}
mass = new Array(1, -9, 7, -23, -3)
Array.protptype.SumN = SumNegative        //добавляем метод к объекту
Massiv.SumN(mass)            // применяем метод SumN к массиву mass
```

Многомерные массивы

Для создания многомерного массива требуется указать все его размерности, заключив каждую из них в квадратные скобки.

Например,

```
matrix = new Array ( )
matrix[0] = new Array ( 2, 3, 8)
matrix[1] = new Array ( 1, -3, 7)
matrix[2] = new Array ( 0, 2, -6)      // массив размерности 3 на 3
```

Методы объекта Array

concat( ) – объединяет два массива в третий и возвращает полученный массив.

Синтаксис: имя_массива3 = имя_массива1.concat(имя_массива2)

join( ) – создает строку из элементов массива с указанным разделителем между ними, возвращает строку символов.

Синтаксис: имя_массива2 = имя_массива1.join(строка)

pop( ) – удаляет последний элемент массива и возвращает его значение.

Синтаксис: имя_массива1.pop( )

push( ) – добавляет к массиву последний элемент, значение которого указанно в качестве параметра и возвращает новую длину массива.

shift( ) – удаляет первый элемент массива и возвращает его значение.

unshift( ) – добавляет к массиву первый элемент, значение которого указанное в качестве аргумента.

reserve( ) – переписывает массив в обратном порядке, возвращает массив.

slise(индекс1[, индекс2]) – создает массив из элементов исходного массива с индексами указанного диапазона. Возвращает массив. Если второй индекс не указан, то новый массив создается из элементов с индекса1 до конца исходного массива.

sort( ) – упорядочивает элементы массива. Если параметр не указан, сортировка производится на основе ASCII-кодов символов значений, что удобно для строк, но не походит для чисел. Параметром может служить имя функции сравнивающей два элемента массива.

Пример,

```
massiv = new Array (7, 1, 34, 5, 63)
```

```
function cmp (x, y)
{ return x - y  }
massiv.sort(cmp)                //массив будет сортироваться по возрастанию
```

Эта функция дает критерий сортировки.

splice(индекс, количество [, элем1[, элем2[, … элемn]]]) – удаляет (заменяет) из массива элементы. Возвращает массив из удаленных элементов. Первый параметр является индексом первого удаляемого элемента, второй - количеством удаляемых элементов. Если указаны необязательные параметры, то происходит замена элементов указанного диапазона на указанные значения параметров. Но это справедливо, если второй параметр не равен нулю.

Пример

```
b = new Array( “один”, 2, 3,  4, “пять”)
c = b.splice(1, 3, “два”,  “три”, “четыре”)
// массив b из элементов «один», «два», «три», «четыре», «пять»
// массив c из элементов 2. 3, 4
```

toLocaleString( ), toString( ) – преобразуют содержимое массива в символьную строку.

Задание

1. Создайте пользовательскую функцию для работы со строками с использованием методов объекта String.

Вариант 1.

Функция вставки строки в исходную строку. Функция должна иметь три параметра: исходную строку, вставляемую строку и позицию вставки.

Вариант 2.

Функция замены в исходной строке все вхождения заданной подстроки на подстроку замены. Функция должна иметь три параметра: исходную строку, заменяемую подстроку и подстроку, которой следует заменить все вхождения заменяемой подстроки.

2. Изучите функции форматирования строк. Продемонстрируйте работу функции из задания 1, отформатировав вновь полученную строку тремя различными способами в HTML-документе.

3. Создайте функцию для работы с объектом Array.

Вариант 1.

Замена минимального элемента значением, заданным как параметр функции.

Вариант 2.

Сортировка по возрастанию элементов массива, расположенных между максимальным и минимальным его элементами.

4. Создайте 2 массива. Добавьте созданную функцию из задания 3 в качестве нового свойства ко всем созданным массивам.

Лабораторная работа № 4 **Работа с окнами**

Главное окно браузера создается автоматически при запуске браузера. С помощью сценария можно создать любое количество окон, а также разбить окно на несколько прямоугольных областей, называемых фреймами. Окну браузера соответствует объект window, а HTML-документу, загруженному в окно, соответствует объект document. Эти объекты могут содержать в себе другие объекты. В частности, объект document входит в состав window.

Доступ к свойствам и методам данного объекта происходит, как и в других объектах, через точку. Поскольку объект document является подобъектом объекта window,

ссылка на HTML-документ, загруженный в текущее окно: window.document. Объект document имеет метод write (запись строки в текущий HTML-документ).

Для его применения используют window.document.write(строка). Объект окна window - корневой объект, имеющий свои подобъекты. Например, location хранит информацию об URL-адресе загруженного документа, screen – данные о возможностях экрана монитора пользователя.

В объектной модели документа объекты сгруппированы в *коллекции*. Коллекция – промежуточный объект, содержащий объекты собственно документа. Коллекция является упорядоченным массивом объектов, отсортированных в порядке упоминания соответствующих им элементов в HTML-документе. Индексация объектов в коллекции начинается с нуля. Синтаксис обращения к элементам коллекции аналогичен синтаксису обращению к элементам массива. Коллекция имеет длину – свойство length.

Коллекция всех графических изображений документа называется images, коллекция всех форм – forms, ссылок – links. Коллекция всех объектов документа называется all.

Один и тот же объект может входить в частную коллекцию (например, images), но он обязательно входит в коллекцию all. При этом его индексы могут быть разными в разных коллекциях.

При использовании документа, загруженного в текущее окно, объект window можно не упоминать, а сразу начинать с объекта document.

Например,

document. images(0)

Вместо индекса можно использовать значение атрибута ID в теге, который определяет соответствующий элемент HTML-документа.

Однако универсальный способ обращения к объектам документа – обращение посредством коллекции all.

С помощью сценария можно создавать любое количество окон. Для этого применяется метод open( ):

window. open(параметры)

Данному методу передаются следующие необязательные параметры:

адрес документа, который нужно загрузить в создаваемое окно;

имя окна (как имя переменной);

строка описания свойств окна (features).

В строке свойств записываются пары свойство = значение, которые отделяются друг от друга запятыми.

Свойства, передаваемые в строке features

Свойство	Значения	Описание
channel mode	yes, no, 1, 0	Показывает элементы управления channel
directories	yes, no, 1, 0	Включают кнопки каталога
ful1screen	yes, no, 1, 0	Полностью разворачивает окно
height	число	Высота окна в пикселях
left	число	Положение по горизонтали относительно левого края экран в пикселях
location	yes, no, 1, 0	Текстовое поле Address
menubar	yes, no, 1, 0	Стандартное меню браузера
resizeable	yes, no, 1, 0	Возможность пользователя изменять размер окна
scrollbars	yes, no, 1, 0	Горизонтальные и вертикальные полосы прокрутки
		Стандартная строка состояния
status	yes, no, 1, 0	Включает панели инструментов браузера
toolbar	yes, no, 1, 0	Положение по вертикали относительно верхнего края
top	число	экрана в пикселях

width	число	Ширина окна в пикселях

Примеры

window. open (“mypage.htm”, “NewWin”, “height=150, width=300”)
window. open (“mypage.htm”)
strfeatures = “top=100, left=15, height=250, width=300, location=no”
window. open (“www.amsu.ru”, strfeatures)

Вместо строки strfeatures можно использовать значение true, тогда указанный документ загружается в существующее окно, вытесняя предыдущий документ.

Метод window.open() возвращает ссылку на объект окна, сохранив которую, можно использовать позднее, например, при закрытии окна.

Для закрытия используют метод close(). Однако, выражение window.close() закрывает главное окно. Для закрытия других окон используют ссылки.

Пример

var str = window. open (“mypage.htm”, “моя страница”)
str.close()

Объект document является центральным в иерархической объектной модели. Он предоставляет всю информацию о HTML-документе с помощью коллекций и свойств и множество методов для работы с документами.

Коллекция document

all	все теги и элементы основной части документа
anchor	якоря (закладки) документа
applets	все объекты документа, включая встроенные элементы управления, графические элементы, апплеты, внедренные объекты
embeds	все внедренные объекты документа
forms	все формы на странице
frames	фреймы, определенные в теге <FRAMESET>
images	графические элементы
links	ссылки и блоки <AREA>
plugins	другое название внедренных документов
scripts	все разделы <SCRIPT> на странице
styleSheets	контейнерные свойства стиля, определенные в документе

Методы document

clear	очищает выделенный участок
close	закрывает текущее окно браузера
createElement	создает экземпляр элемента для выделенного тега
elementFromPoint	возвращает элемент с заданными координатами
execCommand	выполняет команду над выделенной областью
open	открывает документ
queryCommandEnabled	сообщает, доступна ли данная команда
queryCommandIndeterm	сообщает, если данная команда имеет неопределенный статус
queryCommandState	возвращает текущее состояние команды
queryCommandSupported	сообщает, поддерживается ли данная команда
queryCommandText	возвращает строку, с которой работает команда
queryCommandValue	возвращает значение команды, определенное для документа или объекта TextRange
write (writeln)	записывает текст и код HTML в документ, находящийся в указанном окне

Свойства document

Свойство	Атрибут	Назначение
activeElement		Активизирует активный элемент
alinkColor	ALINK	Цвет ссылок на странице
bgColor	BGCOLOR	Определяет цвет фона элемента
body		Ссылка только для чтения на неявный основной объект документа, определенный в теге <BODY>
cookie		Строка cokie-записи. Значение этого свойства приводит к записи на диск.
domain		Устанавливает или возвращает домен документа для его защиты или идентификации.
fgColor	TEXT	Устанавливает цвет текста переднего плана
lastModified		Дата последней модификации страницы, доступна как строка
linkColor	LINK	Цвет еще непосещенных гиперссылок на странице
location		Полный URL документа
parentWindow		Возвращает родительское окно для документа
readyState		Определяет текущее состояние загружаемого объекта
referer		URL страницы, которая вызвала текущую
selection		Ссылка только для чтения на дочерний для document объект selection
title	TITLE	Определяет справочную информацию элемента, используемую при загрузке и всплывающей подсказке
url	URL	URL-адрес документа клиента или в теге <META>
vlinkColor	VLINK	Цвет посещенных ссылок на странице

Объект window кроме дочерних объектов имеет свои методы, свойства, события.

Свойства window:

parent	возвращает родительское окно для текущего
self	возвращает ссылку на текущее окно
top	возвращает ссылку на главное окно
name	название окна
opener	окно, создаваемое текущим
closed	сообщает, если окно закрыто
status	текст, показываемый в строке состояния браузера
defaultStatus	текст по умолчанию строки состояния браузера
returnValue	позволяет определить возвращаемое значение для события или диалогового окна
client	ссылка, возвращаемая объект навигатора браузеру
document	ссылка только для чтения на объект окна document
event	ссылка только для чтения на глобальный объект event
history	ссылка только для чтения на объект окна history
location	ссылка только для чтения на объект окна location
navigator	ссылка только для чтения на объект окна navigator
screen	ссылка только для чтения на объект окна screen

Например,

window.status = «работает сценарий»

Свойство parent позволяет обратиться к объекту, расположенному в иерархии на одну ступень выше. Для перемещения на две ступени выше используют parent.parent. Для обращения к самому главному окну – окну браузера, используют свойство top.

Свойство status используют для вывода сообщений во время работы сценария. Например, window. status = “сценарий работает”

Методы window

open( )	открывает новое окно браузера
close( )	закрывает текущее окно браузера
showHelp( )	показывает окно подсказки как диалоговое
showModalDialog( )	показывает новое модальное(диалоговое) окно
alert( )	окно предупреждения с сообщением и кнопкой ОК
prompt( )	окно приглашения с сообщение, текстовым полем и кнопками ОК и Cancel (Отмена)
confirm( )	окно подтверждения с сообщением и кнопками ОК и Cancel
navigate( )	загружает другую страницу с указанным адресом
blur( )	убирает фокус с текущей страницы
focus( )	устанавливает страницу в фокус
scroll( )	разворачивает окно на заданную ширину и высоту
setInterval( )	указывает процедуре выполняться автоматически через заданное число миллисекунд
setTimeout( )	запускает программу через заданное количество миллисекунд после загрузки страницы
clearInterval( )	обнуляет таймер, заданный методом setInterval( )
clearTimeout( )	обнуляет таймер, заданный методом setTimeout( )
execScript( )	выполняет код сценария, по умолчанию Jscript

Рассмотренные выше методы позволяют работать с независимыми (немодальными) окнами. Для создания модального окна используется метод showModalDialog(). В качестве параметра данный метод принимает адрес документа (файла), имя окна, и строку свойств.

При работе с модальными окнами пользователь не может обратиться к другим окнам, в том числе и к главному. Окна, создаваемые методами alert(), promt(), confirm() являются модальными.

Одним из главных назначений сценариев в HTML-документе является обработка событий, таких как щелчок кнопки мыши по элементу документа, помещение указателя мыши на элемент, нажатие клавиши и др. Для одного и того же элемента можно определить несколько событий на которые он будет реагировать.

Сообщение о событии формируется в виде объекта, т.е. контейнера для хранения информации. Объект события в одном из свойств содержит ссылку на элемент, с которым связано данное событие (на кнопку, изображение и т.п.)

Обычно обработчики событий оформляются в виде функций, определения которых помещаются в контейнерный тег <SCRIPT>.

События window

onblur	выход окна из фокуса
onfocus	окно становится активным
onhelp	нажатие пользователем клавиши F1
onresize	изменение пользователем размеров окна
onscroll	прокрутка окна пользователем
onerror	ошибка при передаче
onbeforeunload	для сохранения данных перед выгрузкой страницы
onload	страница полностью загружена

onunload	непосредственно перед выгрузкой страницы

В случае открытия нескольких окон браузера, пользователь может переключаться между ними, переводя фокус с одного окна на другое. Эти действия инициируются программными событиями onblur и onfocus. Эти же действия можно вызвать, используя методы blur и focus.

Событие onerror происходит при ошибки загрузки страницы или ее элемента. Его можно использовать в программе при попытке вновь загрузить страницу. Например,

```
<SCRIPT>
function window.onerror( ) {
alert (“ Ошибка! Повтори попытку!”)
}
</SCRIPT>
```

События document

onafterupdate	окончание передачи данных
onbeforeupdate	перед выгрузкой страницы
onclick	при щелчке левой кнопкой мыши
ondblclick	при двойном щелчке левой кнопкой мыши
ondragstart	при возникновении перетаскивания
onerror	ошибка при передаче
onhelp	нажатие клавиши F1
onkeydown	нажатие клавиши
onkeypress	возникает при нажатии клавиши и продолжается при удержании клавиши в нажатом состоянии
onkeyup	пользователь отпускает клавишу
onload	при полной загрузке документа
onmousedown	при нажатии кнопки мыши
omousemove	при перемещении указателя мыши
onmouseout	когда указатель мыши выходит за границы элемента
onmouseover	когда указатель мыши входит на документ
onmouseup	пользователь отпускает кнопку мыши
onreadystatechange	возникает при изменении свойства readystate
onselectstart	когда пользователем впервые запускается выделенная часть документа

Динамическое изменение элементов документа

Элементы HTML-документа задаются тегами, большинство из которых имеют параметры (атрибуты). В объектной модели документа тегам соответствуют объекты, а атрибутам – свойства этих объектов. Названия свойств объектов, как правило, совпадают с названиями атрибутов, но записываются в нижнем регистре.

Наиболее удобный способ динамического изменения HTML-документа основан на использовании свойств innerText, outerText, innerHTML и outerHTML. С их помощью можно получить доступ к содержимому элемента. Изменяя значения перечисленных свойств можно частично или полностью изменить сам элемент. Например, можно изменить только надпись на кнопке, а можно превратить кнопку в изображение или Flash-анимацию.

Значением свойства innerText является все текстовое содержимое между открывающим и закрывающим тегами элемента. Внутренние теги игнорируются. Данные открывающего и закрывающего тегов соответствующего элемента также не входят.

В отличие от предыдущего свойство outerText включает в себя данные открывающего и закрывающего тегов. Таким образом, outerTextесть весь текст, содержащийся в контейнере, включая его внешние теги. Например, задан HTML-код:

<DIV ID = “my” >
<A HREF = ‘raznoe.htm’>
<IMG SRC = ‘picture.jpg> Ссылка на раздел <B> Разное <\B>
</A>
<\DIV>

Здесь свойства innerText и outerText для элемента, заданного контейнерным тегом <DIV>, совпадают:

document.all.my.innerText //значение равно – «Ссылка на раздел Разное»

При присвоении свойствам innerText и outerText новых значений нужно помнить, что если значения содержат теги, то они не интерпретируются, а воспринимаются как обычный текст.

Свойство innerHTML содержит внутренний HTML-код контейнера элемента. Присвоение этому свойству нового значения, содержащего HTML-код, приводит к интерпретации кода. Свойство outerText дополнительно включает внешние открывающие и закрывающие теги элемента.

Для приведенного HTML-кода значение document. all. my. innerHTML равно “<A HREF = ‘raznoe.htm’> <IMG SRC = ‘picture.jpg’> Ссылка на раздел <B> Разное <\B> </A>”/

Значение document. all. my. outerHTML – “<DIV ID = “my” > <A HREF = ‘raznoe.htm’> <IMG SRC = ‘picture.jpg> Ссылка на раздел <B> Разное <\B>
</A> <\DIV>”.

Если в сценарии выполнить выражение

document.all.my.innerHTML = “<BUTTON>Щелкни здесь</BUTTON>”

ссылка, изображение и текст будут заменены кнопкой с надписью «Щелкни здесь». При этом контейнерный тег “<DIV ID = “my” > сохранится. Если аналогичным образом использовать outerHTML, кнопка также появится, но уже без контейнера “<DIV ID = “my” >.

Свойства innerHTML и outerHTML могут применяться к элементам, заданным неконтейнерными тегами. Тогда innerHTML и outerHTML совпадают.

Для ускорения загрузки графики можно использовать следующие возможности JavaScript. Можно организовать предварительную загрузку изображений в кэш-память браузера, не отображая их на экране. Это особенно эффективно при начальной загрузке страницы. Пока изображения загружаются в память, оставаясь невидимыми, пользователь может рассматривать текстовую информацию.

Для предварительной загрузки изображения требуется создать его объект в памяти браузера. Это можно сделать следующим выражением:

myimg = new Image (ширина, высота)

Параметры должны соответствовать значениям атрибутов WIDTH и HEIGHT тега <IMG>, который используется для отображения предварительно загруженного изображения.

Для созданного в памяти объекта изображения можно создать имя или URL-адрес графического файла:

myimg.scr = “URL-адрес изображения”

что предписывает браузеру загрузить изображения без его отображения.

После загрузки в кэш-память всех изображений и загрузки всего документа можно сделать их видимыми. Для этого свойству src элемента <IMG> нужно присвоить значение этого же свойства объекта изображения в кэш-памяти. Например,

document. images[0]. scr = myimg. scr

Здесь слева указано свойство scr первого в документе элемента, соответствующего тега <IMG>, справа – свойство scr объекта изображения в кэш-памяти.

С помощью JavaScript можно через заданный интервал времени запускать код или функцию. При этом создается эффект одновременного (параллельного) выполнения вычислительных процессов.

Для организации повторения через заданный интервал выполнения некоторого выражения служит метод setInterval( ) объекта window:

setInterval(выражение, период, [, язык])

Первым параметром является строка, например вызов функции. Период указывается в миллисекундах. Третий параметр – необязательный, в котором указывается язык с помощью которого написано заданное выражение. По умолчанию – JavaScript.

Метод setInterval( ) возвращает некоторое целое число – идентификатор временного интервала, который может быть использован в дальнейшем, например для прекращения выполнения процесса методом clearInterval( ).Например,

```
var pr = setInterval( “myfunc(), 100” )
if (confirm ( “Прервать процесс?”) )
clearInterval(pr)
```

Если требуется выполнить действие с некоторой временной задержкой, используется метод setTimeout( ), имеющий синтаксис аналогичный setInterval( ). Для отмены задержки процесса, запущенного setTimeout( ), используют clearTimeout( ).

Задание

Создать HTML-документ, расположив в нем список названий графических объектов, одно исходное отображение, две кнопки.

Щелчок на элементе списка должен приводить к изменению цвета элемента списка и отображению соответствующего графического элемента, и соответствующего ему тестового сопровождения.

При этом изображение кнопки должно быть также изменено.

Щелчок по первой кнопке через 5 секунд должен инициализировать функцию открытия документа в окне, заданного размера, определенного размера текстового поля и название. Окно должно содержать горизонтальные и вертикальные полосы прокрутки, размер окна не должен изменяться по желанию пользователя. Выведенный в окне текст должен быть синим на сером фоне, иметь выделенный заголовок, ссылки на другие объекты. По выбору продемонстрируйте по пять событий и свойств объектов window и document.

Это действие может быть отменено с помощью второй кнопки.

Лабораторная работа №5 **Работа с фреймами**

Фрейм – прямоугольная область окна браузера, в которую можно загрузить HTML-документ. Разбиение окна браузера на отдельные окна производится с помощью тега <FRAMESET>, внутрь которого вставляются теги <FRAME> с атрибутами, указывающими имя фрейма и адрес HTML-документа.

Пример

```
<HTML>
<FRAMESET ROWS= “30%, 70%”>
<FRAMESET SRC= “документ1.htm” NAME = “frame1” >
<FRAMESET SRC= “документ2.htm” NAME = “frame2” >
</FRAMESET>
</HTML>
```

Здесь применяется вертикальное расположение фреймов. Для горизонтального размещения фреймов вместо атрибута ROWS в теге <FRAMESET> использовать COLS.

Используя вложение тега <FRAMESET>, можно разбить уже имеющийся фрейм на два других.

При разбиении окна на фреймы и, в свою очередь, фрейма на другие фреймы возникают отношения родитель-потомок. Каждому из фреймов соответствует свой объект document. Обеспечение доступа к в иерархии объектов представлено на рисунке.

Так при обращении из одного фрейма-потомка к другому, необходимо помнить, что прямой связи между фреймами-потомками не существует. Поэтому сначала нужно обратиться к родительскому окну, а затем к его второму потомку:

parent. frame2. document. write(“ Привет от первого фрейма.“)

Можно изменить элемент одного фрейма из другого. Например, при щелчке на тексте в правом фрейме в левом изменится один из текстовых документов. Тогда документ в левом фрейме с именем LEFT:

```
<HTML>
Делай раз<BR>
Делай два
<H1 ID = "XXX"> Делай три </H1>
</HTML>
```

Документ в правом фрейме:

```
<HTML>
<SCRIPT>
function change( ) {
parent.LEFT.document.all.XXX. innerText = "Делай пять!!!!"
}
</SCRIPT>
<H1 onclick = "change( )"> Щелкни здесь</H1>
</HTML>
```

В теле функции change( ) происходит обращение к левому фрейму с именем LEFT (задается в установочном HTML-файле) через parent. Изменение элемента происходит за счет присвоения значения свойству innerText. Кроме данного свойства можно использовать outerText, innerHTML или outerHTML.

Важно, что изменения в одном фрейме по событию в другом происходят без перезагрузки HTML-документа.

Фреймы удобно использовать при создании навигационных панелей. В одном фрейме располагаются ссылки, а второй предназначен для отображения документов, вызываемых при активизации соответствующих ссылок.

Пример

```
// установочный файл frame.htm
<HTML>
<FRAMESET COLS = “25%, 75%”>
<FRAME SRC = “menu.htm” NAME = “menu” >
<FRAME SRC = “start.htm” NAME = “main” >
</FRAMESET>
</HTML>
```

Здесь start.htm – документ, который первоначально показан во фрейме main.

```
//menu.htm – навигационная панель
<HTML>
<SCRIPT>
function load (url) {
parent. main. location. href = url;
}
</SCRIPT>
<BODY>
<A HREF = “javascript:load(‘первый.htm’)”>Первый </A>
<A HREF = “второй.htm” TARGET = “main”> Второй </A>
```

<A HREF = “третий.htm” TARGET = “top”> Третий </A>
</BODY>
</HTML>

В примере окно браузера разделено на два фрейма. Первый из них играет роль навигационной панели, а второй – окна для отображения документов. Продемонстрированы два способа загрузки новой страницы во фрейм main. В первом случае используется функция load( ), параметр которой указывает, какой файл следует загрузить. При этом место, в которое он загружается, определяется самой функцией load( ). Во второй ссылке используется атрибут TARGET. В третьем ссылке демонстрируется, как можно избавиться от фреймов.

Для удаления фрейма с помощью load( ) достаточно записать:

parent. location. href = url

Атрибут TARGET в теге ссылки <A HREF> обычно применяется в случаях, когда требуется загрузить одну страницу в один фрейм. Язык сценариев используют при необходимости выполнения нескольких действий.

Для ссылок из родительского окна к объектам его дочерних фреймов можно использовать коллекцию frames. Обращение к определенному фрейму из этой коллекции возможно по индексу или по имени фрейма:

window. frames [индекс]

window. имя_фрейма

При обращении к объекту документа, загруженного во фрейм, следует сначала упомянуть объект document:

window. frames(0). document. all. Myinput. Value

window. LEFT. document. all. Myinput. Value

Ссылка из дочернего фрейма на родительский - осуществляется с использованием parent.

При использовании top следует учитывать, что создаваемый сайт может быть загружен в другой. Тогда объект top окажется объектом другого сайта. Поэтому лучше использовать parent для ссылок на вышестоящее окно или фрейм.

Ссылки top или self используют для предотвращения отображения сайта внутри фреймов другого сайта. Сценарий, выполняющий это, следует разместить в начале документа, например:

<SCRIPT>
if (top != self)
    top. Location = location
</SCRIPT>

т.е., ссылка на свойство top на верхнее окно, должна совпадать со ссылкой self на текущее окно.

Для вставки одного HTML-документа в тело другого средствами браузера служит контейнерный тег <IFRAME>:

<IFRAME SRC = “адрес документа” > </IFRAME>

Данный элемент представляет собой прямоугольную область с прокруткой или без. Такое окно называют плавающим фреймом. Данный документ можно позиционировать с помощью параметров таблицы стилей (тег <STYLE> или атрибут STYLE).

Плавающий фрейм аналогичен обычному фрейму. При создании он помещается в коллекцию frames. Среди его свойств широко используется align – выравнивание плавающего фрейма относительно окружающего содержимого документа. Его возможные значения:

absbotton – выравнивает нижнюю границу фрейма по подстрочной линии символов окружающего текста,

absmiddle – выравнивает середину границу фрейма по центральной линии между top и absbotton окружающего текста,

baseline – выравнивает нижнюю границу фрейма по базовой линии окружающего текста,

botton – совпадает с baseline (только IE)

left – выравнивает фрейм по левому краю элемента-контейнера,

middle – выравнивает воображаемую центральную линию окружающего текста по воображаемой центральной линии фрейма,

right – выравнивает фрейм по правому краю элемента-контейнера,

texttop – выравнивает верхнюю границу фрейма по надстрочной линии символов окружающего текста,

top – выравнивает верхнюю границу фрейма по верхней границе окружающего текста.

Задание

Вариант 1. Окно браузера поделить на два фрейма. В левом расположить небольшое изображение. Организовать возможность вывода полномасштабного изображения в левом окне по щелчку мыши на миниатюре изображения в правом фрейме. В левом фрейме дополнительно организовать навигационную модель. Создать новый HTML-документ и вставить его в ранее созданный, как плавающий вертикальный фрейм, выравнивая нижнюю границу фрейма по базовой линии окружающего текста.

Вариант 2. Окно браузера поделить на три фрейма. В левом расположить небольшое изображение. Организовать возможность вывода полномасштабного изображения в среднем окне по щелчку мыши на миниатюре изображения в правом фрейме. В левом фрейме дополнительно организовать навигационную модель. Создать новый HTML-документ и вставить его в ранее созданный, как плавающий вертикальный фрейм, выравнивая воображаемую центральную линию окружающего текста по воображаемой центральной линии фрейма.

Лабораторная работа №6 **Простые визуальные эффекты**

Смена изображений

Для смены одного изображения на другое достаточно с помощью сценария заменить значение атрибута SCR тега <IMG>. Например:

```
<HTML>
<IMG ID = “myimg” SRC= ‘pict1.gif ’
onclick = “document.all.myimg.src = ‘pict2.gif’ ”>
</HTML>
```

Здесь смена изображения из файла pict1.gif на изображение из файла pict2 происходит при первом щелчке на нем. Последующие щелчки не приведут к видимым изменениям, поскольку второе изображение будет заменяться им же. Чтобы при повторном щелчке происходила замена изображения на предыдущее необходимо создать переменную-триггер (флаг), принимающий одно из двух возможных значений, по которому можно определить, какое из двух значений надо отобразить.

```
<HTML>
<IMG ID = “myimg” SRC= ‘pict1.gif ’
onclick = “ imgchange( )”>
<SCRIPT>
var flag=false
function  imgchange( ) {
if (flag) document. all. myimg. src = “pict1.gif ”
        else document. all. myimg. src = “pict2.gif ”
flag=!flag
}
```

```
</SCRIPT>
</HTML>
```

Цветовые эффекты

Задача изменения цвета кнопки при наведении на нее указателя мыши и возвращения в первоначальное состояние при удалении указателя с кнопки может быть решена следующим образом:

```
<HTML>
<STYLE>
mystile {font-weight.bold; background-color: a0a0ao} // серый цвет кнопок
</STYLE>

<FORM onmouseover = “colorchange ( ‘yellow’)” onmouseout
= “colorchange (‘a0a0ao’)”>
<INPUT TYPE = “BUTTON” VALUE = “Кнопка” CLASS = “mystile”
onclick = “alert( ‘Вы нажали кнопку’)” >
</FORM>
<SCRIPT>
function colorchange (color){
if (event. scrElement.type = = “button”)
        event. scrElement. style. backgroundColor = color;
}
</SCRIPT>
</HTML>
```

Функция colorchange( ) проверяет, является ли инициатор события объектом типа button. Если это так, то цвет кнопки меняется. Без этой проверки менялся бы не только цвет кнопок, но и текста.

Аналогичным способом можно изменять цвет фрагментов текста. Но в этом случае текст должен быть заключен в контейнер, например в теги <P>, <B>, <I>, <DIV>.

Можно создать прямоугольную рамку, окаймляющую текст, которая периодически изменяет цвет. Рамка создается тегами одноячеечной таблицы с заданием нужных атрибутов и параметров стиля:

```
<TABLE ID= “tab” BORDER=1 WIDH=200 style= “border:10 solid : red”>
<TR><TD> Доброе утро! </TR></TD>
</TABLE>
```

Функция изменения цвета:

```
<SCRIPT>
function flash( ) {
if ( !document.all) return null;
if (tab.style.borderColor = = ‘red’)   tab.style.borderColor = ‘yellow’
else tab.style.borderColor = ‘red’;
}
setInterval (“flash”, 500);              //мигание рамки с интервалом 500 мс
</SCRIPT>
```

Объемные заголовки

Объемные заголовки часто используются на веб-страницах. Идея создания объемного заголовка состоит в наложении нескольких надписей с одинаковым содержанием с некоторым сдвигом по координатам. Наилучший эффект достигается путем подбора цветов надписей (игрой света и тени) с учетом цвета фона. Для этого используют библиотеку стилей. Функция, создающая заголовок с заданными параметрами:

```
function d3 (text, x, y, tcolor, fsize, fweight, family, zind) {
```

```
/*      text – текс заголовка
        x – горизонтальная координата (left)
        y – вертикальная координата (top)
        tcolor – цвет переднего плана
        fsize – размер шрифта (пт)
        fweight – вес (толщина шрифта)
        family – название семейства шрифтов
        zind z-Index   */
if (!text) return null // если текст не указан ничего не выполняется
//значение параметров по умолчанию
if (!x) x=0
if (!y) y=0
if (!tcolor) tcolor='00aaff '
if (!fsize)  fsize=36
if (!fweight)  fweight =800
if (!family) ffamily='arial'
// внутренние настройки
var sd=5, hd=2
var xzind= " "
if (zind) xzind= "; - Index:"+zind
var xstyle ='font-family:' + family + ';font-size:' + fsize + ';font-weight:' + fweight + ';'
var xstr = '<DIV STYLE = "position: absolute; top:'  + (y +sd ) + '; left :' +
( x + sd ) +  xzind + ' ">'
xstr+='<P styl e  = " ' + xstyle + 'color: darked">' + text + '</P></DIV>'
xstr+= '<DIV STYLE = " position: absolute; top:' + y  + '; left :' +
x  +  xzind + ' ">'
xstr+='<P styl e  = " ' + xstyle + 'color: silver">' + text + '</P></DIV>'
xstr+= '<DIV STYLE = " position: absolute; top:' + ( y + hd )  + '; left :' +
(x +hd)  +  xzind + ' ">'
xstr+='<P styl e  = " ' + xstyle + 'color:' + tcolor +   '">' + text + '</P></DIV>'
document.write(xstr)            //запись в документ
}
```

Параметр z-Index позволяет установить слой, в котором находится заголовок, и тем самым указать, будет ли заголовок располагаться над или под другим видимым элементом документа. Элементы с более высоким значением z-Index находятся над элементами, у которых z-Index меньше. Перекрытие элементов с одинаковыми значениями z-Index определяется порядком их следования в HTML-документе.

Вызов приведенной выше функции может выглядеть так:

```
d3 ("это не графика, это просто стиль текста", 50, 50, 'blue', 72, 800, 'times')
```

Задание

Выполнить следующие действия на веб-странице:

1. Создать программу для работы с галереей миниатюр. При щелчке кнопкой мыши по миниатюре изображение должно увеличиваться, а затем при щелчке на увеличенном изображении оно должно уменьшаться. Доработайте приведенную в тексте функцию функцию imgchange( ). Для решения этой задачи потребуется массив флагов и функция обработчик, определяющая на каком именно изображении произошел щелчок:

```
var p1=new Array ("pict1.gif" , … )  //массив имен исходных файлов
var p2=new Array ("pict2.gif" , … )  //массив имен замещающих файлов
//формирование тегов, описывающих изображения
var xstr =  " "
for (i=0; i<p1.length; i++)
```

```
xstr+= '<IMG ID = "i' + i +' " SRC = " '+ p1[i]+' " onclick = "imgchange( )" >'
}
document.write(xstr)            // запись в документ
```

2.Выполнить замену фрагмента текста с черного на красный при наведении на него указателя мыши.

3. Выделите фрагмент текста мигающей трехцветной рамкой.

4.Создать эффект динамического изменения цвета ссылок. Различать цвета мерцания использованных и неиспользованных ссылок. (Множество цветов задать массивом. Использовать свойства linkColor и linkColor объекта document). Для изменения цвета случайным образом можно использовать метод random( ) (счетчик случайных чисел) встроенного объекта Math. Если требуется получить случайное число x, лежащее в интервале от A до B, то x=A+(B – A)*Math. random( )

5. Создать три объемных заголовка, поэкспериментировав со значениями внутренних параметров sd, hd, а также с заданием параметров по умолчанию.

Лабораторная работа №7 **Применение фильтров**

С помощью фильтров каскадных таблиц стилей можно получить разнообразные эффекты: постепенное появление или исчезновение рисунка, плавное преобразование одного изображения в другое, задание степени прозрачности и др.

Фильтр следует понимать как некий инструмент преобразования изображения, взятого из графического файла и вставленного в HTML – документ с помощью тега <IMG>. Следует иметь в виду, что фильтры работают только в IE4+.

Фильтры можно применять не только к графическим объектам, но и к текстам, текстовым областям, кнопкам.

Прозрачность

С помощью фильтра alpha можно установить прозрачность графического объекта. Сквозь прозрачные графические объекты видны нижележащие изображения. Прозрачность имеет несколько вариантов градиентной формы. Например, она может увеличиваться от центра к краям изображения

Фильтр alpha задается с помощью каскадной таблицы стилей и имеет ряд параметров. В примере для графического изображения стиль определяется с помощью атрибута STYLE:

```
<IMG ID = "myimg" SCR = " pict. gif "
STYLE = "position: absolute; top:10; left: 50;
filter: alpha (opacity = 70, style = 3)">
```

Здесь целочисленный параметр opacity определяет степень непрозрачности. Значение 0 соответствует полной прозрачности изображения, а 100 – полной непрозрачности. Параметр style задает градиентную форму распределения прозрачности по изображению как целое число от 0 до 3. По умолчанию значение параметра равно 0, и градиент не применяется. Фильтр имеет и другие параметры, определяющие прямоугольную область изображения, к которому применяется фильтр. По умолчанию фильтр применяется ко всему изображению.

Фильтр можно определить в каскадной таблице силей внутри контейнерного тега <STYLE>:

```
<HTML>
<STYLE>
#myimg{position: absolute; top:10; left: 50; filter: alpha (opacity = 70, style = 3)}
</STYLE>
< IMG ID = "myimg" SCR = " pict. gif "
</HTML>
```

Доступ к свойствам фильтра в сценарии:
document. all. id_изображения. filters [“имя_фильтра”]. параметр = значение

Для рассмотренного примера это выражение имеет вид:
document. all. myimg. filters [“alpha”]. opacity = 30
Для остальных параметров alpha аналогично.

Для IE5.5+ можно использовать другой синтаксис, в котором в каскадной таблице стилей задается ссылка на специальный компонент и имя фильтра:
#myimg { filter: progid: DXImageTransform . Microsoft . alpha
(opacity = 70, style = 3)}

Тогда доступ к свойствам фильтра:
document.all.myimg.filters [“DXImageTransform. Microsoft alpha”]. opacity = 30

Трансформация

Фильтр alpha статический. Существуют и динамические фильтры: apply( ) – фиксирует изображение, play( ) – трансформирует, revealtrans( ) –преобразовывает изображение, stop( ) останавливает процесс преобразования при необходимости.

Фильтр revealtrans( ) имеет параметры: duration – длительность преобразования в секундах (число с плавающей точкой) и transition – тип преобразования (целое от 0 до 23).

Для эффекта появления изображения можно воспользоваться фрагментом, который происходит после загрузки документа, т.е. по событию onload:

```
<HTML>
<BODY onload = “transform()” >
<IMG ID = “myimg” SRC = “ pict. gif ” STYLE = “position: absolute; top:10;
left: 50; visibility = “hidden” filter: revealtrans (duration = 3, transition =12)” >
//transition =12 соответствует плавной трансформации
</BODY>
<SCRIPT>
function transform ( ) {                                    //появление изображения
document.all.myimg.style. visibility = “hidden”     // изображение невидимо
myimg. filters ( “revealtrans”). apply( )
myimg.style. visibility = “visible”
myimg. Filters ( “revealtrans”). play( )                 // выполняем преобразования
}
</SCRIPT>
</HTML>
```

Для замены одного изображения на другое необходимо установить начальное и конечное изображение путем присвоения нужных значений свойству src объекта, соответствующего изображению, например фрагментом:
document.all.myimg.src = “pict2. gif ”

Рассмотренный синтаксис воспринимается браузерами IE4+. Для IE5.5+ в каскадной таблице стилей задается ссылка на специальный компонент и имя фильтра. Так для трансформации изображения по щелчку мыши на графическом объекте в другое, и обратно, можно воспользоваться программой:

```
<HTML>
<STYLE>
#myimg{position: absolute; top:10; left: 50; filter: progid: DXImageTransform . Microsoft revealtrans (duration = 3, transition = 12)}
</STYLE>
< IMG ID = “myimg” onclick = “transform( )” SCR = “ ear. gif ”>
<SCRIPT>
 function transform( ) {
//фиксация исходного изображения
```

```
myimg. filters (“DXImageTransform . Microsoft revealtrans”). apply ( )
//определение конечного изображения
if (document. all. mying. src. index0f (“ear”)!= -1)
document. all. mying. src = “s.gif”
        else document. all. mying. src = “ear.gif”
//выполняем преобразование
myimg. filters (“DXImageTransform . Microsoft revealtrans”). play ( )
}
</SCRIPT>
</HTML>
```

В браузере IE5.5+ возможно прменение фильтра basicimage, с помощью которого изображение можно повернуть на угол, кратный 90 градусам, задать прозрачность, зеркально отразить и др.

Задание

В HTML-документе из предыдущей лабораторной работы применить к рисункам методы трансформации и изменения прозрачности изображения и фона.

Лабораторная работа №8 **Динамическое изменение элементов документа**

Элементы HTML-документа задаются тегами, большинство из которых имеют параметры (атрибуты). В объектной модели документа тегам соответствуют объекты, а атрибутам – свойства этих объектов. Названия свойств объектов, как правило, совпадают с названиями атрибутов, но записываются в нижнем регистре.

Наиболее удобный способ динамического изменения HTML-документа основан на использовании свойств innerText, outerText, innerHTML и outerHTML. С их помощью можно получить доступ к содержимому элемента. Изменяя значения перечисленных свойств можно частично или полностью изменить сам элемент. Например, можно изменить только надпись на кнопке, а можно превратить кнопку в изображение или Flash-анимацию.

Значением свойства innerText является все текстовое содержимое между открывающим и закрывающим тегами элемента. Внутренние теги игнорируются. Данные открывающего и закрывающего тегов соответствующего элемента также не входят.

В отличие от предыдущего свойство outerText включает в себя данные открывающего и закрывающего тегов. Таким образом, outerTextесть весь текст, содержащийся в контейнере, включая его внешние теги. Например, задан HTML-код:

```
<DIV ID = “my” >
<A HREF = ‘raznoe.htm’>
<IMG SRC = ‘picture.jpg> Ссылка на раздел <B> Разное <\B>
</A>
<\DIV>
```

Здесь свойства innerText и outerText для элемента, заданного контейнерным тегом <DIV>, совпадают:

```
document.all.my.innerText    //значение равно – «Ссылка на раздел Разное»
```

При присвоении свойствам innerText и outerText новых значений нужно помнить, что если значения содержат теги, то они не интерпретируются, а воспринимаются как обычный текст.

Свойство innerHTML содержит внутренний HTML-код контейнера элемента. Присвоение этому свойству нового значения, содержащего HTML-код, приводит к интерпретации кода. Свойство outerText дополнительно включает внешние открывающие и закрывающие теги элемента.

Для приведенного HTML-кода значение document. all. my. innerHTML равно “<A HREF = ‘raznoe.htm’> <IMG SRC = ‘picture.jpg> Ссылка на раздел <B> Разное <\B> </A>”/

Значение document. all. my. outerHTML – “<DIV ID = “my” > <A HREF = ‘raznoe.htm’> <IMG SRC = ‘picture.jpg> Ссылка на раздел <B> Разное <\B>
</A> <\DIV>”.

Если в сценарии выполнить выражение

document.all.my.innerHTML = “<BUTTON>Щелкни здесь</BUTTON>”

ссылка, изображение и текст будут заменены кнопкой с надписью Щелкни здесь. При этом контейнерный тег “<DIV ID = “my” > сохранится. Если аналогичным образом использовать outerHTML, кнопка также появится, но уже без контейнера “<DIV ID = “my” >.

Свойства innerHTML и outerHTML могут применяться к элементам, заданным неконтейнерными тегами. Тогда innerHTML и outerHTML совпадают.

Для ускорения загрузки графики можно использовать следующие возможности JavaScript. Можно организовать предварительную загрузку изображений в кэш-память браузера, не отображая их на экране. Это особенно эффективно при начальной загрузке страницы. Пока изображения загружаются в память, оставаясь невидимыми, пользователь может рассматривать текстовую информацию.

Для предварительной загрузки изображения требуется создать его объект в памяти браузера. Это можно сделать следующим выражением:

myimg = new Image (ширина, высота)

Параметры должны соответствовать значениям атрибутов WIDTH и HEIGHT тега <IMG>, который используется для отображения предварительно загруженного изображения.

Для созданного в памяти объекта изображения можно создать имя или URL-адрес графического файла:

myimg.scr = “URL-адрес изображения”

что предписывает браузеру загрузить изображения без его отображения.

После загрузки в кэш-память всех изображений и загрузки всего документа можно сделать их видимыми. Для этого свойству src элемента <IMG> нужно присвоить значение этого же свойства объекта изображения в кэш-памяти. Например,

document. images[0]. scr = myimg. scr

Здесь слева указано свойство scr первого в документе элемента, соответствующего тега <IMG>, справа – свойство scr объекта изображения в кэш-памяти.

С помощью JavaScript можно через заданный интервал времени запускать код или функцию. При этом создается эффект одновременного (параллельного) выполнения вычислительных процессов.

Для организации повторения через заданный интервал выполнения некоторого выражения служит метод setInterval( ) объекта window:

setInterval(выражение, период, [, язык])

Первым параметром является строка, например вызов функции. Период указывается в миллисекундах. Третий параметр – необязательный, в котором указывается язык с помощью которого написано заданное выражение. По умолчанию – JavaScript.

Метод setInterval( ) возвращает некоторое целое число – идентификатор временного интервала, который может быть использован в дальнейшем, например для прекращения выполнения процесса методом clearInterval( ).Например,

```
var pr = setInterval( “myfunc(), 100” )
if (confirm ( “Прервать процесс?”) )
clearInterval(pr)
```

Если требуется выполнить действие с некоторой временной задержкой, используется метод setTimeout( ), имеющий синтаксис аналогичный setInterval( ). Для отмены задержки процесса, запущенного setTimeout( ), используют clearTimeout( ).

Задание

1. Создать HTML-документ, в котором отображается список названий графических объектов и одно исходное отображение. Щелчок на элементе списка должен приводить к отображению соответствующего элемента.

2. Создать в HTML-документе две кнопки. Щелчок по кнопке ПУСК открывает через 5 секунд новое окно и загружает в него некоторый документ. Это действие может быть отменено с помощью кнопки ОТМЕНА.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО САМОСТОЯТЕЛЬНОЙ РАБОТЕ СТУДЕНТОВ

Самостоятельная работа – планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (при частичном непосредственном участии преподавателя, оставляющем ведущую роль за работой студентов).

Самостоятельная работа является важным видом учебной и научной деятельности студента и играет значительную роль в рейтинговой технологии обучения. Государственным стандартом предусматривается, как правило, 50% часов из общей трудоемкости дисциплины на самостоятельную работу студентов. Количество часов самостоятельной работы и ее распределение между дидактическими единицами приведено в рабочей программе.

Задачами самостоятельной работы являются:

систематизация и закрепление полученных теоретических знаний и практических умений студентов;

углубление и расширение теоретических знаний;

формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;

развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;

формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;

развитие исследовательских умений;

использование собранного и полученного в ходе самостоятельных занятий для эффективной подготовки к итоговым зачетам и экзаменам.

В процессе самостоятельной работы студент приобретает навыки самоорганизации, самоконтроля, самоуправления, становится активным самостоятельным субъектом учебной деятельности.

Выполняя самостоятельную работу под контролем преподавателя, студент должен:

– освоить минимум содержания, выносимый на самостоятельную работу студентов и предложенный преподавателем в соответствии с Государственными образовательными стандартами высшего профессионального образования по данной дисциплине.

– планировать самостоятельную работу в соответствии с графиком самостоятельной работы, предложенным преподавателем.

– самостоятельную работу студент должен осуществлять в организационных формах, предусмотренных учебным планом и рабочей программой преподавателя.

– выполнять самостоятельную работу и отчитываться по ее результатам в соответствии с графиком представления результатов, видами и сроками отчетности по самостоятельной работе студентов.

Основной формой самостоятельной работы студента является изучение конспекта лекций, их дополнение, рекомендованной литературы, подготовка отчетов к лабораторным работам.

КОНТРОЛЬ ЗНАНИЙ

Текущий контроль знаний предполагает защиту лабораторных работ и выполнение тестовых заданий. В тестовые задания входят вопросы по лекционному материалу, примеры которых приведенные в рабочей программе.

СОДЕРЖАНИЕ