ФЕДЕРАЛЬНОЕ АГЕНТСТВО ВОЗДУШНОГО ТРАНСПОРТА
(РОСАВИАЦИЯ)

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ГРАЖДАНСКОЙ АВИАЦИИ» (МГТУ ГА)

Кафедра вычислительных машин, комплексов, систем и сетей

Н.И. Романчева

ИНФОРМАТИКА

ТЕХНОЛОГИИ АРХИВАЦИИ, ПОИСКА ИНФОРМАЦИИ И ПРОВЕРКИ РАБОТОСПОСОБНОСТИ СЕРВЕРНЫХ СЦЕНАРИЕВ

Учебно-методическое пособие
по выполнению лабораторных работ № 4, 5

для студентов
специальности 10.05.02
очной формы обучения

Москва
ИД Академии Жуковского
2021

УДК 681.3.05+004.7
ББК 6Ф6.5
Р69

Рецензент:
Терентьев А.И. – канд. техн. наук, доцент кафедры ВМКСС

Романчева Н.И.

Р69 Информатика. Технологии архивации, поиска информации и проверки работоспособности серверных сценариев [Текст] : учебно-методическое пособие по выполнению лабораторных работ № 4, 5 / Н.И. Романчева. – М.: ИД Академии Жуковского, 2021. – 32 с.

Данное учебно-методическое пособие издается в соответствии с рабочей программой учебной дисциплины «Информатика. Технологии архивации, поиска информации и проверки работоспособности серверных сценариев» по учебному плану для студентов специальности 10.05.02 очной формы обучения.

Рассмотрено и одобрено на заседаниях кафедры 25.02.2021 г. и методического совета 02.03.2021 г.

УДК 681.3.05+004.7
ББК 6Ф6.5

В авторской редакции

Подписано в печать 14.05.2021 г.
Формат 60х84/16 Печ. л. 2 Усл. печ. л. 1,86
Заказ № 732/0330-УМП03 Тираж 40 экз.

Московский государственный технический университет ГА
125993, Москва, Кронштадтский бульвар, д. 20

Издательский дом Академии имени Н. Е. Жуковского
125167, Москва, 8-го Марта 4-я ул., д. 6А
Тел.: (495) 973-45-68
E-mail: zakaz@itsbook.ru

С О Д Е Р Ж А Н И Е

1. ОСНОВНЫЕ ТРЕБОВАНИЯ И ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

Настоящее учебно-методическое пособие предназначено для студентов, по специальности 10.05.02 Информационная безопасность телекоммуникационных систем (специалитет), выполняющих лабораторные работы по дисциплине Информатика в соответствии с ФГОС3++. В данное учебно-методическое пособие включены материалы для выполнения лабораторных работ № 4-5.

Продолжительность каждой лабораторной работы - 4 часа.

Целью проведения лабораторных работ является закрепление основных теоретических положений, изложенных в лекциях по дисциплине Информатика.

В процессе выполнения лабораторных работ студенты должны получить практические навыки архивации и работы со сжатыми данными при хранении и передаче данных, поиска информации в глобальных сетях на примере сети Интернет, освоение приемов и методов работы с поисковыми системами и работы с PHP-машиной, проектирование элементов пользовательского интерфейса.

Лабораторная работа состоит из следующих этапов:

1) домашняя подготовка;

2) выполнение работы на компьютере в соответствии с заданием;

3) сдача выполненной работы преподавателю на персональном компьютере;

4) распечатка результатов работы на принтере;

5) оформление отчета;

6) защита лабораторной работы.

В процессе домашней подготовки студент:

- изучает лекционный материал,
- материал по теме лабораторной работы данного учебно-методического пособия и дополнительной литературы;
- знакомится с заданием на выполнение лабораторной работы;
- готовит проект отчета по выполнению лабораторной работы.

ВЫПОЛНЕНИЕ лабораторной работы проводится во время занятий в лаборатории кафедры ВМКСС МГТУГА в присутствии преподавателя в соответствии с расписанием, утвержденным проректором по УМР МГТУ ГА. В процессе выполнения лабораторной работы студент последовательно выполняет задание. По завершению работы - демонстрирует преподавателю результаты.

СДАЧА РАБОТЫ преподавателю на персональном компьютере заключается в демонстрации выполненной работы и выполнении непосредственно при преподавателе индивидуального дополнительного задания.

ОТЧЕТ по каждой лабораторной работе должен содержать: название работы; цель лабораторной работы; задание на выполнение лабораторной работы; краткие комментарии по выполнению лабораторной работы; распечатки файлов результатов или диаграмм, подписанные преподавателем.

ЗАЩИТА лабораторной работы преподавателю проводится по контрольным вопросам и при наличии оформленного отчета (распечатки результатов выполнения лабораторной работы должны быть аккуратно приклеены). После защиты лабораторной работы преподавателем делается соответствующая запись на отчете студента.

РЕЙТИНГОВЫЙ КОНТРОЛЬ по лабораторным работам производится при их сдаче во время лабораторных занятий. Перед выполнением лабораторной работы производится экспресс-опрос для определения готовности студентов к выполнению работы (знания теоретического материала, целей работы и т.д.).

1) Допуск к ЛР

Допуск к выполнению ЛР происходит в виде устного опроса (при условии наличия у студента печатной версии титульного листа отчета по лабораторной работе).

Студент, не прошедший устный опрос к выполнению лабораторной работы не допускается.

2) Выполнение и защита ЛР

Минимальная оценка выставляется за выполненную и защищенную лабораторную работу, а дополнительными баллами оценивается качество выполненной лабораторной работы и полнота знаний, показанная студентами при ее защите. Лабораторная работа считается сделанной, если она выполнена полностью в соответствии с заданием.

Сроки сдачи лабораторных работ

Номер лабораторной работы	Лабораторная работа	Срок сдачи*
1	Лабораторная работа №4	Лабораторная работа №4
2	Лабораторная работа №5	Лабораторная работа №5

**Срок сдачи лабораторной работы – в соответствии с расписанием проведения лабораторных работ.*

3) Отчет по ЛР

Отчет по лабораторной работе представляется в печатном виде в формате, предусмотренном шаблоном отчета по лабораторной работе.

4) Защита ЛР

Отчет не может быть принят и подлежит доработке в случае:

- небрежного выполнения,
- низкого качества представленного материала (неполное решение, не указаны единицы измерения),

- отсутствия необходимых разделов отчета,
- отсутствия необходимого графического материала,
- отсутствия выводов по работе;
- некорректной обработки результатов выполнения лабораторной работы, и т.п.

2. ЛАБОРАТОРНАЯ РАБОТА №4
ТЕХНОЛОГИИ АРХИВАЦИИ

2.1. Цель работы

Целью данной работы является получение практических навыков архивации и работы со сжатыми данными при хранении и передаче данных.

2.2. Перечень компетенций, формируемых в ходе выполнения лабораторной работы

Общепрофессиональная компетенция ОПК-2:
Способен применять информационно-коммуникационные технологии, программные средства системного и прикладного назначений, в том числе отечественного производства, для решений задач профессиональной деятельности.

2.3. Задание на выполнение работы

1) Создать или скопировать в директории BI202n, где n – последняя цифра года поступления в ВУЗ, на рабочем диске 5-7 файлов (текстовых, исполняемых, командных, программных), скопировать с других дисков.
2) Создать архивы для этих файлов с помощью архиваторов WinZIP, WinRAR,Peazip и др.
3) Сравнить объемы получившихся файлов, результаты занести в таблицу:

название архиватора	тип файла	размер файла	размер файла после сжатия	Степень сжатия(%)
WinZIP	.txt			
	.exe			
	.doc			
	.cpp			
	.bat			
WinRAR	.txt			
	.exe			
	.doc			
	.cpp			
	.bat			
ReaZip	.txt			
	.exe			
	.doc			
	.cpp			
	.bat			

4) Построить графики по полученным данным, используя табличный процессор MS Exel и сделать выводы об эффективности использования того или иного архиватора для конкретного типа файла.
5) Создать командный файл, который с помощью архиватора позволяет расположить файлы в архиве в заданном порядке, просмотреть архив, извлечь файлы из архива в заранее созданный каталог.
6) Создать с помощью архиваторов многотомные архивы с паролем, заданным в параметрах, на диске, заданном в параметрах, поместив в них все файлы из каталога LAB рабочего диска, исключив файлы с расширением EXE.
7) Просмотреть списки созданных архивов.
8) С помощью архиватора WinRar выполнить следующие команды:

а) добавить в архив заданный файл;

б) поместить в архив все файлы из текущего каталога, за исключением файлов с заданным расширением;

в) создать защищенный архив;

г) создать архивный файл, позволяющий сохранить структуру каталогов;

д) добавить комментарии к архивам;

е) извлечь заданный файл из архива.

ж) создать многотомный архив, указав размер тома –например, 2 Мб;

з) выполнить поиск заданной строки в архивах по различным поисковым признакам.

9) Создать самораспаковывающиеся RAR- и ZIP-архивы, не поддерживающие распределенные архивы (включить переключатель «Без распределения» в группе Spanning Support - Поддержка распределенного архива).
10) Создать самораспаковывающиеся распределенные архивы RAR- и ZIP-архивы.
11) Используя диспетчер архивов WinZip, выполнить интеграцию служебных и прикладных программ с операционной системой Windows 200x.
12) Исследовать свойства форматов сжатия графических данных (файлы .bmp, .gif, .jpg). Результаты занесите в таблицу:

Формат файла	Размер файла (Кбайт)	Степень сжатия (%)
24 разрядный .bmp		
.gif		
.jpg		

13) Построить графики по полученным данным, используя табличный процессор MS Excel и сделать выводы об эффективности использования того или иного архиватора для конкретного типа файла.

2.4. Основные теоретические сведения

Характерной особенностью большинства типов данных, с которыми традиционно работают пользователи, является определенная избыточность. Степень избыточности зависит от типа данных.

При обработке информации избыточность также играет важную роль. Так, например, при преобразовании или селекции информации избыточность используют для повышения ее качества (репрезентативности, актуальности, адекватности и т. п.). Однако, когда речь заходит не об обработке, а о хранении готовых документов или их передаче, то избыточность можно уменьшить, что дает эффект сжатия данных.

Если методы сжатия информации применяют к готовым документам, то нередко термин *сжатие данных* подменяют термином *архивация данных,* а программные средства, выполняющие эти операции, называют *архиваторами.*

2.4.1. Объекты сжатия

В зависимости от того, в каком объекте размещены данные, подвергаемые сжатию, различают:

- уплотнение (архивацию) файлов;
- уплотнение (архивацию) папок;
- уплотнение дисков.

Уплотнение файлов применяют для уменьшения их размеров при подготовке к передаче по каналам электронных сетей или к транспортировке на внешнем носителе малой емкости, например на гибком диске.

Уплотнение папок используют как средство архивации данных перед длительным хранением, в частности, при резервном копировании.

Уплотнение дисков служит целям повышения эффективности использования их рабочего пространства и, как правило, применяется к дискам, имеющим недостаточную емкость.

Несмотря на изобилие алгоритмов сжатия данных, теоретически есть только три способа уменьшения их избыточности. Это либо изменение содержания данных, либо изменение их структуры, либо и то и другое вместе.

Если при сжатии данных происходит изменение их содержания, метод сжатия необратим и при восстановлении данных из сжатого файла не происходит полного восстановления исходной последовательности. Такие методы называют также *методами сжатия с регулируемой потерей информации.* Они применимы только для тех типов данных, для которых формальная утрата части содержания не приводит к значительному снижению потребительских свойств. В первую очередь, это относится к мультимедийным данным: видеорядам, музыкальным записям, звукозаписям и рисункам.

Методы сжатия с потерей информации обычно обеспечивают гораздо более высокую степень сжатия, чем обратимые методы, но их нельзя применять к текстовым документам, базам данных и, тем более, к программному коду. Характерными форматами сжатия с потерей информации являются: JPG для графических данных; .MPG - для видеоданных; .MP3- для звуковых данных.

Если при сжатии данных происходит только изменение их структуры, то метод сжатия обратим. Из результирующего кода можно восстановить исходный массив путем применения обратного метода. Обратимые методы применяют для сжатия любых типов данных. Характерными форматами сжатия без потери

информации являются: .GIF, .TIP, .PCX и многие другие для графических данных; .AVI для видеоданных; .ZIP, .ARJ, .RAR, .LZH, .LH, .CAB и многие другие для любых типов данных.

2.4.2. Архиваторы WinRAR и WinZip

Современные программные средства для создания и обслуживания архивов отличаются большим объемом функциональных возможностей, многие из которых выходят за рамки простого сжатия данных и эффективно дополняют стандартные средства операционной системы. В этом смысле современные средства архивации данных называют *диспетчерами архивов*.

К базовым функциям, которые выполняют современные диспетчеры архивов, относятся:

- извлечение файлов из архивов,
- создание новых архивов,
- добавление файлов в имеющийся архив,
- создание самораспаковывающихся архивов,
- создание распределенных архивов на носителях малой емкости,
- тестирование целостности структуры архивов,
- полное или частичное восстановление поврежденных архивов,
- защита архивов от просмотра и несанкционированной модификации.

К дополнительным функциям диспетчеров архивов относятся сервисные функции, делающие работу более удобной. Они часто реализуются внешним подключением дополнительных служебных программ и обеспечивают:

- просмотр файлов различных форматов без извлечения их из архива;
- поиск файлов и данных внутри архивов;
- установку программ из архивов без предварительной распаковки;
- проверку отсутствия компьютерных вирусов в архиве до его распаковки;
- криптографическую защиту архивной информации;
- декодирование сообщений электронной почты;
- «прозрачное» уплотнение исполнимых файлов .EXE и .DLL;
- создание самораспаковывающихся многотомных архивов;
- выбор или настройку коэффициента сжатия информации.

Структура окон WinRAR и WinZip типична для приложений Windows и ряд примеров окон WinZip приведены на рис.2.1-2.3.

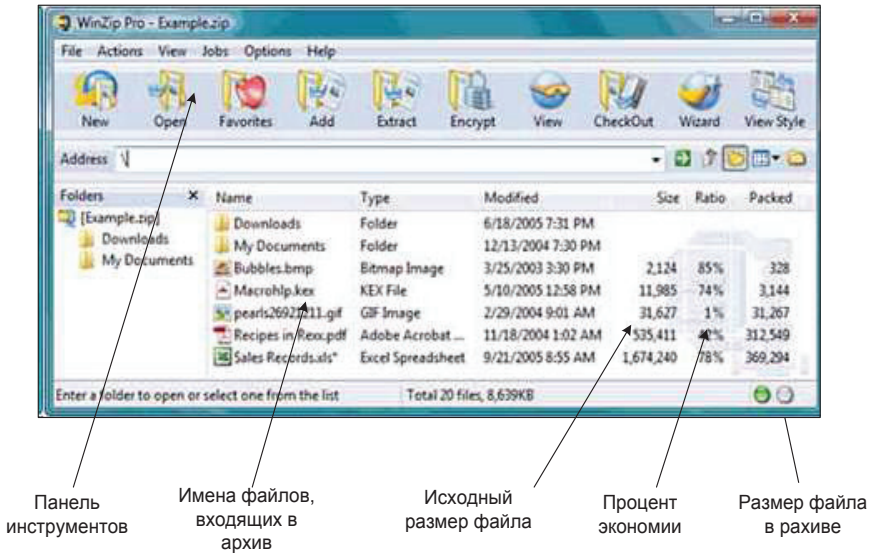

Рисунок 2.1 – Просмотр содержимого архива с помощью WinZip

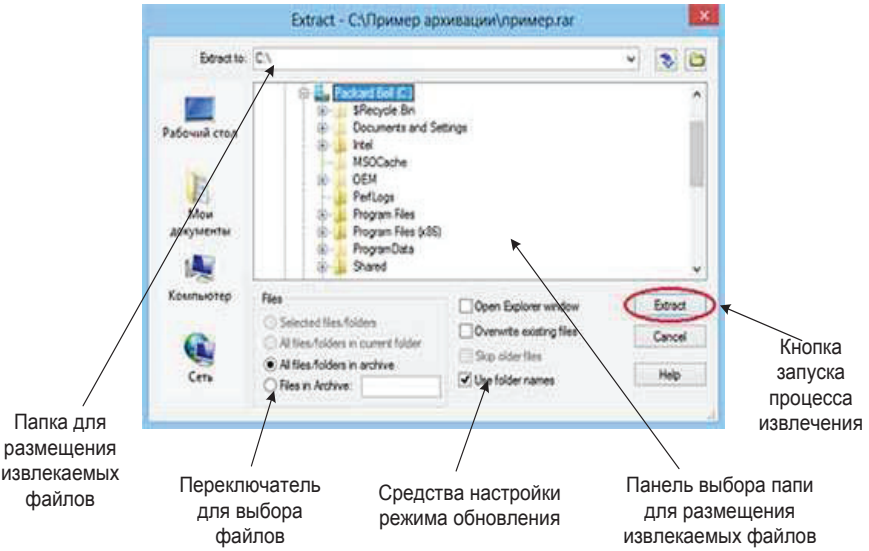

Рисунок 2.2 – Управление извлечением из архива

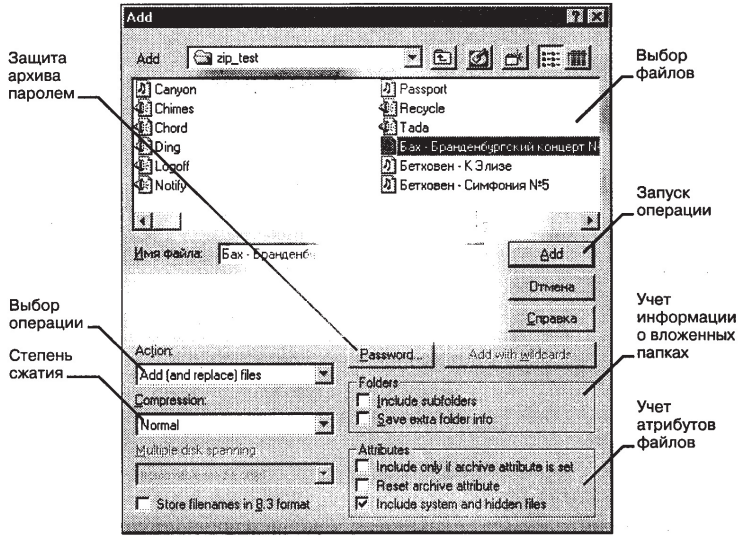

Рисунок 2.3 – Управление добавлением файлов в архив

Структура окна программы PeaZip приведена на рис.2.4.

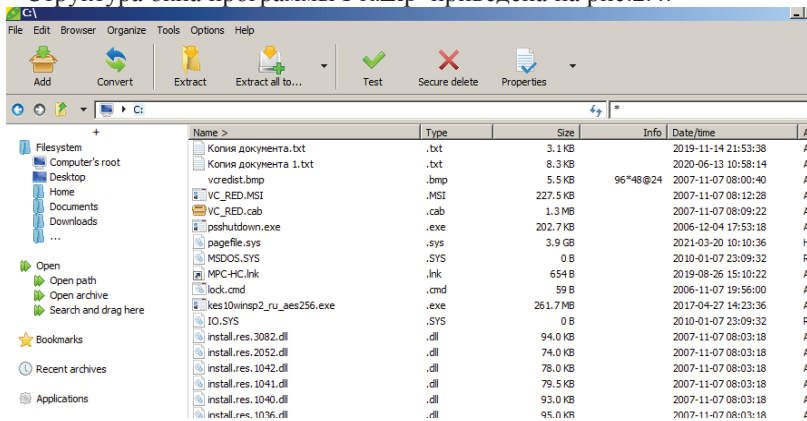

Рисунок 2.4. -Структура окна архиватора ReaZip

2.4.3. Самораспаковывающиеся архивы

В тех случаях, когда архивация производится для передачи документа потребителю, следует предусмотреть наличие у него программного средства, необходимого для извлечения исходных данных из уплотненного архива. Если таких средств у потребителя нет - создают самораспаковывающиеся архивы.

Самораспаковывающийся архив готовится на базе обычного архива путем присоединения к нему небольшого программного модуля. Сам архив получает расширение .EXE, характерное для исполняемых файлов. Потребитель сможет выполнить его запуск как программы, после чего распаковка архива произойдет на его компьютере автоматически.

2.4.4. Распределенные архивы

В тех случаях, когда предполагается передача большого архива на носителях малой емкости, например на гибких дисках, возможно распределение одного архива в виде малых фрагментов на нескольких носителях. Некоторые диспетчеры (например, WinZip) выполняют разбиение сразу на внешние носители, а некоторые (например, WinRAR) позволяют выполнить предварительное разбиение архива на фрагменты заданного размера на жестком диске. Впоследствии их можно перенести на внешние носители путем копирования.

При создании распределенных архивов диспетчер WinZip обладает особенностью: каждый том несет файлы с одинаковыми именами. В результате этого нет возможности установить номера томов, хранящихся на каждом из гибких дисков, по названию файла. Поэтому каждый диск следует маркировать пометками на наклейке, а при создании распределенного архива следует быть внимательнее, чтобы не перепутать последовательность немаркированных томов.

В случае необходимости узнать номер тома можно не по названию файла, а по метке на диске, хотя эта операция не слишком удобна. Для этого следует открыть окно «Мой компьютер», выбрать значок дисковода, щелкнуть на нем правой кнопкой мыши и выбрать в контекстном меню пункт «Свойства». В диалоговом окне «Свойства: Диск N:)» на вкладке «Общие» можно узнать номер тома распределенного архива в поле «Метка тома».

2.4.5. Создание самораспаковывающегося ZIP-архива

1) Запустите программу WinZip.
2) Выполните команду File/Open Archive (Файл /Открыть архив). Откройте ранее созданный архив .zip.
3) Выполните команду Actions /Make .Exe File (Действия/Создать исполнимый файл) - откроется диалоговое окно WinZip Self-Extractor (Генератор самораспаковывающегося архива).
4) В поле Create Self-Extracting Zip files from (Создать самораспаковывающийся архив из ...) необходимо записать адрес исходного ZIP-файла. Можно воспользоваться кнопкой Browse (Обзор) для поиска нужного файла.
4) В группе Self Extractor Type (Тип самораспаковывающегося архива) включите переключатель, соответствующий операционной системе компьютера, для которого готовится архив.
5) В группе Spanning Support (Поддержка распределенного архива) включите переключатель No spanning (Без распределения) и нажмите кнопку ОК.

2.4.6. Создание самораспаковывающегося распределенного архива

1) Запустите программу WinZip.

2) Выполните команду File/Open Archive (Файл /Открыть архив). Откройте ранее созданный архив .zip.
3) Выполните команду Actions /Make .Exe File (Действия /Создать исполнимый файл) - откроется диалоговое окно WinZip Self-Extractor (Генератор самораспаковывающегося архива).
4) В группе элементов управления Spanning Support (Поддержка распределенного архива) включите переключатель Safe Spanning Method (Защищенный метод распределения) или Old Spanning Method (Обычный метод распределения).

Защищенный метод создает на первом гибком диске два файла: исполнимый файл, выполняющий автоматическую распаковку, и первый том распределенного архива. На последующих дисках создается продолжение распределенного архива. Такой подход повышает уровень безопасности, поскольку даже в том случае, когда исполнимый файл поврежден, например компьютерным вирусом, информация остается в архивном файле. Этот метод применяют для передачи архивных материалов на гибких дисках.

Обычный метод не создает отдельного исполнимого файла, и весь архив хранится в одном исполнимом файле, распределенном по нескольким носителям. Данный метод используют для самораспаковывающихся архивов, передаваемых по каналам компьютерных сетей.
5) Откройте диалоговое окно WinZip Self-Extractor (Генератор самораспаковывающегося архива) и установите флажок Erase any existing files on the new disk before continuing (Предварительно стереть все существующие файлы на гибких дисках).
6) Далее нажмите кнопку OK - начнется процесс создания первого тома распределенного архива. По окончании процесса по указанию программы извлеките записанный гибкий диск и вставьте новый.
7) Создав последний том, программа предложит извлечь последний диск и вставить первый для внесения правок в заголовок архива.

2.4.7. Интеграция служебных и прикладных программ с ОС

Под интеграцией программного обеспечения понимают возможность совместной работы нескольких различных программ в рамках единой системы управления. Так, например, известным системным средством интеграции является концепция внедрения и связывания объектов и основанный на ней буфер обмена Windows. Другим приемом является изменение свойств программы «Проводник». Для эпизодических работ по архивации и извлечению файлов и папок удобнее использовать систему, хорошо интегрированную в Windows, например, WinZip. Для регулярных работ по созданию резервных копий папок и дисков удобнее использовать автономные средства, поскольку для них проще
организуется взаимодействие с прочими программами (в частности, со средствами автоматизации). В этих случаях можно рекомендовать, например, программу WinRAR.

1) Запустите программу «Проводник» (Пуск / Программы / Проводник).
2) Скопируйте в созданную папку несколько произвольных файлов.
3) Выделите один из файлов и откройте контекстное меню. Обратите внимание на то, что в нем имеются два пункта для создания архива (создание архива с произвольным именем и с именем, соответствующим текущему файлу). Появление этих пунктов связано с наличием в компьютерной системе диспетчера архивов и интеграции WinZip с Проводником Windows.
4) Выполните команду Add to Zip (Добавить в архив). Далее произойдет автоматический запуск диспетчера архивов WinZip и откроется диалоговое окно Add (Добавление в архив).
5) В поле Add to archive (Добавить в архив) ввести название файла создаваемого архива, адрес текущей папки заносится автоматически. Проверив настройку прочих элементов управления, запустите процесс архивации щелчком на командной кнопке Add (Добавить).
6) Перейдите в окно программы Проводник и убедитесь в том, что в папке появился архивный файл test.zip. Щелкните на значке архивного файла правой кнопкой мыши и изучите новые команды контекстного меню, позволяющие выполнить операции с архивным файлом.
7) Выполните команду Create Self-Extractor (Создать самораспаковывающийся архив). В открывшемся диалоговом окне щелкните на командной кнопке «Да» и в последующих диалоговых окнах откажитесь от проверки созданного архива.
8) Закройте открытые окна программы WinZip, и в программе Проводник убедитесь в том, что в рабочей папке появился исполняемый файл (.EXE).
9) В программе Проводник выполните перетаскивание значка любого файла (или группы файлов) на значок созданного ZIP-архива. При отпускании кнопки мыши в конце перетаскивания происходит автоматическое добавление новых файлов в архив. Если содержимое правой панели Проводника открыто в режиме Таблица, после каждого перетаскивания можно наблюдать увеличение размера файла архива.

2.4.8. Исследование свойств форматов сжатия графических данных

1) Откройте графический редактор Paint (Пуск/Программы/Стандартные/ Paint).
2) Загрузите в него заранее подготовленный многоцветный рисунок.
3) Определите размер рисунка в пикселях (Рисунок/Атрибуты).
4) Оцените теоретический размер рисунка в 24-разрядной палитре (3 байта на точку) по формуле:

$$S=M-N-3,$$

где S — размер файла с рисунком (байт);
M - ширина рисунка (точек);
N - высота рисунка (точек).

5) Сохраните рисунок в папку C:\Temp\Pictures, выбрав имя файла test и назначив тип файла: 24-разрядный рисунок (.BMP).

6) Повторно сохраните рисунок, выбрав то же имя test, но назначив тип файла .GIF. При сохранении произойдет потеря определенной части графической информации.
7) Восстановите рисунок, загрузив его из ранее сохраненного файла Test.bmp.
8) Вновь сохраните его под тем же именем, но выбрав в качестве типа файла формата *.JPEG*.
9) Запустите программу Проводник.
10) Откройте папку C:\Temp\Pictures в режиме Таблица.
11) Определите размеры файлов Test.bmp, Test.gif и Test.jpg.
12) Определите коэффициент сжатия файлов (R), взяв отношения размеров файлов к теоретической величине, полученной расчетным путем.

2.5. Вопросы к защите лабораторной работы

1) Что понимается под термином архивация?
2) Для чего необходимо создавать архив?
3) Какие программы предназначены для работы с архивами?
4) Поясните наиболее известные алгоритмы архивации.
5) Как можно упаковать информацию при хранении на диске?
6) Приведите последовательность упаковки данных в архив и распаковки данных из архиватора WinRar.
7) Приведите команды упаковки данных в архив и распаковки данных из архива для архиватора ReaZip.
8) Как создать многотомный архив?
9) Укажите расширение имен файлов на сменных носителях продолжения архива.
10) Как получить полную справку по всем возможным режимам работы программы-архиватора?
11) Как создать самораспаковывающий архив?
12) Что понимается под интеграцией служебных и прикладных программ с ОС?

2.6. Список рекомендуемой литературы

1) Экслер А.Б. Архиваторы (Программы для хранения и обработки информации в сжатом виде).- М.: МП Алекс- 1992- 150 с.
2) Информатика: Базовый курс / С.В. Симонович и др. – СПб.: Питер, 2019 – 640 с.

3. ЛАБОРАТОРНАЯ РАБОТА №5
ТЕХНОЛОГИИ ПОИСКА ИНФОРМАЦИИ И ПРОВЕРКИ РАБОТОСПОСОБНОСТИ СЕРВЕРНЫХ СЦЕНАРИЕВ

3.1. Цель работы

Целью данной работы является получение практических навыков и умений поиска информации в глобальных сетях на примере сети Интернет, работы с PHP-машиной и проектирования элементов пользовательского интерфейса.

3.2. Перечень компетенций, формируемых в ходе выполнения лабораторной работы

Общепрофессиональная компетенция ОПК-2:
Способен применять информационно-коммуникационные технологии, программные средства системного и прикладного назначений, в том числе отечественного производства, для решений задач профессиональной деятельности.

3.3. Задание на выполнение работы

1) Осуществите простой и расширенный поиск информации по термам и категориям, используя различные поисковые системы. При поиске обязательными для использования являются поисковые системы Google, Yandex. Кроме этого, рекомендуется использовать хотя бы две из следующих систем: Yahoo, AltaVista, MSN, Rambler, Aport:

а) сформируйте запрос, в соответствии с заданием преподавателя, при этом используйте различные комбинации управляющих символов (+, -, круглые скобки), например:

- *информационная безопасность,*
- *информационная безопасность - кибербезопасность,*
- *информационная безопасность – кибербезопасность- безопасность информации,*
- *информационная безопасность + экспертные системы – кибербезопасность – безопасность информации,*
- *(кибербезопасность) +кибербезопасность*

б) сравните изменения результатов поиска и занесите их в отчет.

в) выполните поиск информации, используя тематические каталоги;

г) выполните поиск графической информации (например, иллюстрации к лабораторной работе);

г) выполните поиск научной информации, используя алгоритм, приведенный в п.3.3.1

2) Составьте таблицу - маршрутную карту, содержащую следующие разделы: URL-адрес ресурса (поисковой системы или каталога), название

ресурса, количество найденных источников по запросу, аннотация первых 5-ти найденных источников, фактическая релевантность.

3) Запустить PHP-сценарий для вывода информации о версии PHP-машины.

4) Разработать программу и сценарий для решения уравнения (в соответствии с вариантом, выданным преподавателем). Предусмотреть формы ввода данных (однострочные поля) и кнопки управления (например: «Решить», «Очистить»).

5) Разработать программу для ввода и получения данных пользователем. Документ ввода данных должен содержать два поля для однострочного ввода: «Имя пользователя», «e-mail», поле многострочного ввода, например «Комментарий», переключатели единственного и множественного выбора. Если пользователь не сделал выбор, считать выбор, установленный по умолчанию.

6) Написать программу генерации равномерного пароля.

3.4. Основные теоретические сведения и приемы работы

3.4.1. Поиск информации

Информационно-поисковые системы (ИПС) являются наиболее мощным механизмом поиска сетевых информационных ресурсов Интернет. Назначение информационно-поисковых систем Интернет: *ретро-поиск* - свободный поиск информации в информационных массивах по совокупности признаков (обычно ключевых слов), *избирательное распространение информации* - поиск информации по заранее подготовленным запросам с определенной периодичностью.

Основные причины существования ИПС в Интернет: большой объем информации; ее недостаточная или чересчур подробная структуризация; широкий тематический профиль информационных массивов. Идея применить гипертекстовую модель к информационным ресурсам, распределенным в сети, и сделать это максимально простым способом была предложена Т. Бернерс-Ли. Он заложил три краеугольных камня системы из четырех существующих ныне, разработав:

- язык гипертекстовой разметки документов HTML (*HyperText Markup Language*);
- универсальный способ адресации ресурсов в сети URL (*Universal Resource Locator*);
- протокол обмена гипертекстовой информацией HTTP (*HyperText Transfer Protocol*).

Позже команда NCSA добавила к этим трем компонентам четвертый: универсальный интерфейс шлюзов CGI (*Common Gateway Interface*).

Существует два вида информационных баз данных о Web-страницах: поисковые машины и каталоги:

- поисковые машины (spiders, crawlers) постоянно исследуют Сеть с целью пополнения собственной базы данных документов. Обычно это не требует никаких усилий со стороны человека. Для поисковых систем довольно важна конструкция каждого документа. Большое значение имеют title, meta-tag и содержимое страницы. Примером может быть поисковая система Altavista;

- каталоги: в отличие от поисковых машин, информация в каталоги заносится по инициативе человека. Добавляемая страница должна быть жестко привязана к принятым в каталоге категориям. Конструкция страниц значения не имеет. Примером каталога может служить Yahoo.

В сети Интернет используются следующие основные способы поиска информации:

1) *Навигация:* гипертекстовые ссылки (World Wide Web), строки-селекторы (Gopher);

2) *Информационный поиск:* Archie, Veronica, Wais и др.

В ходе выполнения данной лабораторной работы изучаются существующие способы поиска информации в WWW. Поиск строится на основе преобразования предложений некоторого информационно-поискового языка (ИПС) в запросы к информационной системе. Выделяют следующие типы информационно поисковых языков: *традиционные; взвешивание терминов; ИПЯ типа "Like this"*. Традиционное определение процедуры поиска документов в ИПС было введено для решения проблемы автоматического индексирования документов. Например, запрос к системе "Найти все документы, в которых встречается термин "информационная" и "безопасность", либо термин "ИБ", но не встречается термин "безопасность"" выглядит следующим образом:

((информационная and безопасность) or ИБ) not безопасность

Такая форма запроса имеет недостатки*:* плохая масштабируемость выдачи результатов, оператор OR приводит к слишком большому расширению списка релевантных документов, а оператор AND резко сужает отклик. Поэтому существуют модификации традиционного языка - взвешенный запрос, когда каждому термину приписывают некоторый вес. Суть его в том, что для индексирования используют те термины, которые имеют высокую частоту встречаемости внутри документа и низкую во всем информационном массиве. Сама характеристика вычисляется как отношение частоты встречаемости термина в документе к частоте встречаемости термина в массиве. Используя эту меру, системы индексирования приписывают документу первые 20-40 символов, которые и составляют его поисковый образ.

Для оптимальной и быстрой работы с поисковыми системами существуют определенные правила написания запросов. Подробный перечень для конкретного поискового сервера можно, как правило, найти на самом сервере по ссылкам *Помощь, Подсказка, Правила составления запроса* и т.п.

В запросах можно использовать управляющие символы, например, знаки препинания. Они позволяют более точно описать, что необходимо найти. С

помощью символов + и – показывают значимость того или иного слова. Символ «+» означает, что слово обязательно должно встречаться на странице. Символ «-» означает, что данное слово не должно встречаться вообще. Эти символы ставятся вплотную к слову (без пробела).

Для поиска информации, например, в поисковой системе Яндекс www.yandex.ru, необходимо:

1) В поле ввода запроса введите *ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ* и нажмите кнопку Enter. Подождите, пока загрузятся результаты поиска в виде ссылок на подходящие документы и краткие выдержки из них (рис.3.1).

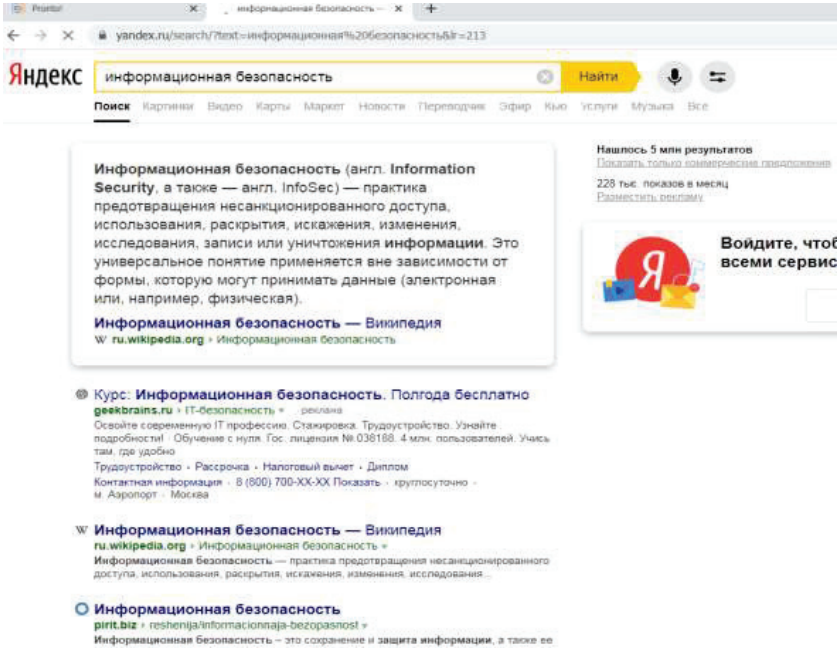

Рисунок 3.1- Окно браузера с результатами поиска по запросу

2) Прочитайте названия ссылок и краткие выдержки из текста.

3) Щелкните по одной из ссылок – соответствующая страница откроется в новом окне.

4) Оцените, насколько загруженная страница соответствует введенному запросу (определение фактической релевантности).

Для выполнения расширенного поиска необходимо выбрать режим РАСШИРЕННЫЙ ПОИСК, форма представлена на рис.3.2.

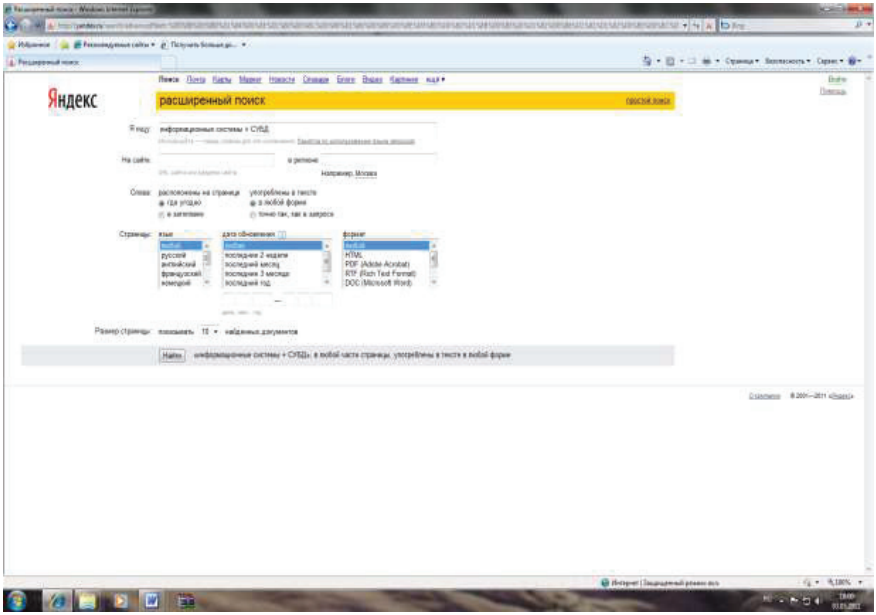

Рисунок 3.2- Форма для расширенного запроса

Для поиска графической информации необходимо:

1) Набрать адрес поисковой системы, например, http://www.yandex.ru и инициализировать процесс загрузки ресурса.

2) В интерфейсе начальной страницы поисковой системы Яndex найти форму для поиска и строку ввода запроса. Щелчком левой клавишей мыши по строке установить в ней курсор и ввести запрос, например, *Информационная безопасность*. Далее щелчком левой клавиши мыши выбрать в строке меню пункт *КАРТИНКИ*.

3) Инициализировать процесс поиска в поисковой системе, нажав на кнопку *НАЙТИ*.

4) Просмотреть результаты поиска и найти среди них наиболее подходящие (релевантные) запросу.

Ниже приводится алгоритм поиска научной информации:

а) Сформулируете тему запроса в форме предметной рубрики и/или набора ключевых слов.

б) Определите цели и задачи поиска информации:

- Учебная работа;
- Исследовательская работа.

в) Определите ограничения поиска:

- по типу и виду литературы;
- по месту нахождения информации/публикации;
- по языку информации/публикации;

-по ретроспекции (возрасту информации/публикации).

г) Обратитесь к библиотечно-библиографическим ресурсам МГТУ ГА:

- электронная библиотека;
- электронный каталог библиотеки;
- электронное хранилище полнотекстовых документов;

д) Обратитесь ресурсами ГСНТИ:

- Традиционные текущие информационные издания;
- Глобальные сетевые ресурсы;
- Научная электронная библиотека eLIBRARY.ru.

Существует ряд правил индексирования документов поисковыми системами. В Интернет имеются страницы, которые явно ухудшают качество поиска. Они специально созданы с целью обмануть поисковую систему. Для этого, например, на странице размещают невидимый или бессмысленный текст. Или создают дорвеи - промежуточные страницы, которые перенаправляют посетителей на сторонние сайты. Некоторые сайты умеют замещать страницу, с которой перешел пользователь на какую-нибудь другую. То есть когда пользователь переходит на такой сайт по ссылке из результатов поиска, а потом хочет снова вернуться к ним и посмотреть другие результаты, он видит какой-то другой ресурс.

Такие ресурсы не представляют интереса для пользователей и вводят их в заблуждение - и, соответственно, ухудшают качество поиска. Поисковые системы, например, Яндекс, автоматически исключает их из поиска или понижает в ранжировании.

2) Есть сайты, которые содержат popunder-баннеры (они перемещаются по экрану вслед за прокруткой страницы и закрывают ее содержание, а при попытке закрытия такого баннера открывается новое окно) и clickunder-рекламу (она неожиданно для пользователя открывает рекламную страницу при любом клике по сайту, в том числе - по ссылкам). Оба этих вида рекламы мешают навигации по сайту и нормальному восприятию информации. Поэтому сайты с такой рекламой располагаются в поисковой выдаче ниже, чем сайты, на которых пользователь может найти ответ на свой вопрос без лишних проблем.

3.4.2. Основы синтаксиса PHP

PHP (Hypertext Preprocessor) –является языком серверных сценариев для быстрого создания динамических web-страниц, его поддерживают практически все web-серверы, вне зависимости от их платформ. Для написания PHP-сценариев можно использовать любой редактор, при этом следует дать файлу расширение .php.

Основное преимущество PHP- простота, гибкость, скорость выполнения.

Синтаксис PHP заимствован из таких языков как C, Java, Perl. Кроме того, этот язык существует в связке с Apache и SQL.

Для PHP существует четыре способа отделения его от общего кода HTML:

<? echo (“SGML инструкции | n”) ; ?>

<?php echo (“XML документ | n”) ; ?>
<script language= “php”>
echo (“специально для FrontPage”);
</script>
<% echo (“ASP-стиль”) ; %>
<% = $variable; # комментарий “echo (“ASP-стиль”) ; %>

Следует сказать, что инструкции в PHP отделяются друг от друга точкой с запятой (;), хотя перед закрывающимся тегом (?>) ставить точку с запятой (;) не обязательно.

В PHP, как и во многих языках программирования, существует средство для хранения данных – переменная. Для выбора имени переменной в PHP существует ряд синтаксических правил:

- любое имя переменной должно начинаться со знака доллара ($);
- после него может идти либо буква, либо знак подчеркивания (_), но не цифра;
- далее могут следовать буквы, цифры и символы подчеркивания в любой последовательности. Знак пробела недопустим!
- имена переменных чувствительны к регистру ($ lata и $Lata не будут эквиваленты).

Кроме того, в соответствии с Coding Standart, при выборе имени переменной используются символы в нижнем регистре, слова разделяются знаками подчеркивания.

PHP поддерживает четыре скалярных (Integer, Double, Boolean,String) и два смешанных (Array, Object) типа данных. Помимо скалярных и смешанных, PHP поддерживает два специальных (resource, NULL) типа данных, а также несколько псевдотипов (mixed, number, callback).

В отличие от других языков программирования, в PHP не обязательно задавать тип данных, так как он автоматически определяется, когда присваивается значение.

Кроме того, в одной программе переменная может быть, например, числом и строкой. Для отслеживания текущего типа данных применяется функция *gettype ().* В круглых скобках указывается имя переменной, например, *echo gettype ($var).*

Если требуется проверка переменной на соответствие определенному типу данных, то применяются функции *is_integer(), is_double(), is_string(), is_(array), is_object( )* и *is_bool().* В случае соответствия типа переменной выводится 1, иначе - 0.

В PHP имеется возможность использовать переменные, имена которых содержат переменные, например:

```
<?php
$name =’stroka’;          // $name содержит строку ’stroka’
$stroka  = 7;             // $id содержит число 7
echo $$name          // выводит  7
```

?>

Такие переменные, как $$name, называют динамическими. Они применяются, как правило, если требуется в ходе выполнения программы создавать много переменных.

В PHP заранее определен ряд констант, которые называются предопределенными. Их основное назначение – хранить информацию о системе. Например, PHP_VERSION, PHP_OS содержат соответственно версию PHP и название операционной системы, на которую установлен сервер.

Существует также несколько констант, которые могут менять свое значение в зависимости от их использования. Например, константы __LINE__ и __FILE__ содержат в себе соответственно номер строки и имя файла сценария. Эти константы можно писать как прописными буквами, так и заглавными.

Операторы классифицируются по количеству операндов, на которые они действуют. Обычно используются бинарные операторы, такие как сложение, вычитание и др. Они задействуют два операнда. Но в PHP есть и унарные операторы (используют один операнд), и тернарные (три операнда).

Для отладки сценариев применяется оператор подавления ошибки (@). Если поставить его перед выражением, то любые возникающие в нем ошибки или предупреждения, не будут выводиться в окне браузера:

```
<?php
  @$a = 5/0;   // подавляет ошибку деления на 0
?>
```

При запуске php-файла автоматически инициализируются переменные с такими же именами, как у элементов формы. Для доступа к данным из HTML–формы используются суперглобальные массивы $_GET и $_POST. Использование того или иного массива зависит от метода передачи данных. В ниже приведенном примере выбирается метод передачи данных GET

```
<?php
  echo  $_GET ['text']   ;   // выводит значение поля text
?>
```

Если выбирается метод передачи данных GET, то создается элемент массива $_GET, ключ которого имеет название text. При использовании метода POST данные заносятся в массив $_POST.

Несмотря на то, что базы данных являются более удобными хранилищами данных, файлы до сих пор не утратили своей актуальности. При малых количествах информации (счетчик посещений, гостевая книга) лучше использовать именно файлы.

Для работы с файлом его нужно открыть. Для этого используется функция *fopen ()*. В качестве входных параметров выступает строка с именем файла и специальный признак, по которому определяется режим открытия файла. Функция возвращает дескриптор файла, который имеет значение типа source, в случае неудачной попытки открытия файла она возвратит FALSE. Пример PHP-кода:

```
<?php
  $file = fopen (“info.txt”, “r”);   // открытие файла info.txt для чтения
?>
```

Режимы открытия файла приведены ниже:

r - файл открывается только для чтения. При попытке открыть несуществующий файл выводится сообщение об ошибке. Если файл открылся, то указатель текущей позиции устанавливается в начало;

r+ - файл открывается одновременно для чтения и записи. Указатель текущей позиции устанавливается в начало файла. При этом запись в файл будет происходить поверх уже существующих данных;

w - файл открывается для записи. При этом все данные, которые были в нем, уничтожаются. Если файл с таким именем не существует, то он создается;

w+ - действия аналогичные режиму w, но файл открывается для чтения и для записи;

a - файл открывается для записи. При этом его указатель устанавливается в конец файла. Если файла не существует, то выводится сообщение об ошибке;

a+ - файл открывается для чтения и записи. При этом его указатель устанавливается в конец файла. Если файла не существует, то он создается.

Если работа с файлом завершена, то его следует закрыть. Эта операция осуществляется с помощью функции *fclose (),* которая в качестве входного параметра принимает дескриптор файла. Функция возвращает булевское значение, которое равно TRUE при удачном закрытии файла и FALSE в противном случае. Пример PHP-кода:

```
<?php
  $file  =  fopen (“info.txt”, “r”);   // открытие файла только для чтения
//закрытие файла
if   (fclose ($file))
{
  echo “закрытие файла произошло удачно”;
}
else
{
  echo “ошибка закрытия файла”;
}
?>
```

Чтение из файлов в PHP можно осуществлять различными способами. Например, с помощью функции *fread().* Она принимает дескриптор файла – число, которое определяет длину строки из файла и возвращает эту строку. В приведенном ниже примере из файла info.txt читаются первые пять символов, которые записываются в виде строки в переменную $txt:

```
<?php
```

```
  $file  =  fopen (“info.txt”, “r”);   // открытие файла только для чтения
// чтение 5 символов файла
$txt = fread ($file, 5  );
//закрытие файла
fclose ($file);
//вывод строки
echo  $txt;
?>
```

Если при чтении достигнут конец файла, функция прекращает свое выполнение и возвращает строку, содержащую ровно столько символов, сколько прочиталось (несмотря на то, что длина строки может быть меньше запрашиваемой).

В PHP имеется похожая на выполнение функция *fgets(),* которая читает символы из файлов в количестве равном заданному минус единица и возвращает их. Чтение прекращается не только при достижении конца файла, но и если встречается символ перевода строки \n. Эту функцию удобно применять в том случае, когда нужно просмотреть файл по строкам:

```
<?php
  $file  =  fopen (“info.txt”, “r”);   // открытие файла только для чтения
// чтение 4-х символов файла
$txt = fgets ($file, 5 );
fclose ($file);
//вывод строки
echo  $txt;
?>
```

3.4.3. Примеры сценариев

Пример 1. Для того чтобы узнать версии PHP- машины, установленной на компьютере или на сервере, услугами которого Вы пользуетесь, необходимо запустить сценарий, приведенный ниже:

```
<?php
 php(info);
?>
```

Пример2. Простой сценарий на PHP

```
<html><body>
<h1>Пример простого сценария на PHP</h1>
<?php
// Вычисляем текущую дату в формате "день.месяц год"
1 $dat=date("d.m y");
// Вычисляем текущее время
2 $tm=date("h:i:s");
# Выводим их
echo "Текущая дата: $dat года<br>\n";
```

```
echo "Текущее время: $tm<br>\n";
# Выводим цифры
echo " Квадраты первых 5 натуральных чисел:<br>\n";
3  for($i=1; $i<=5; $i++)
4  echo "<li>$i в квадрате = ".($i*$i);
?>
</body></html>
```

В первой строке переменной с именем $dat присваивается значение, которое вернула функция date(). У нее задается один параметр, который определяет формат результата, в нашем случае это будет строка вида "дата, месяц, год".

В третьей строке находится определение цикла for (счетчик $i, которому присваивается начальное значение 1, инкрементируется на единицу на каждом шаге, пока не достигнет пяти). Затем следует оператор, выполняющий вывод квадрата i-го числа.

Пример3. Решение квадратного уравнения.

В приведенном примере сценария пронумерованы строки, которые относятся к PHP-коду. Непронумерованные строки представляют собой простой HTML-код.

```
<html><head><title>Квадратное уравнение</title></head><body>
1<?php
2       if (isset($_POST['act'])&& $_POST['act'] == 'act') {
3
if(((!is_numeric($_POST['a']))||(!is_numeric($_POST['b'])))||(!is_numeric($_POST['c'])
))){
4  exit("Ошибка ввода!<br><br><a href=". $_SERVER['PHP_SELF'] .">Ещё
раз</a>");}
5   echo"a = ".$_POST['a'];$a = $_POST['a'];
6   echo"  b = ".$_POST['b'];$b = $_POST['b'];
7   echo"  c = ".$_POST['c'];$c = $_POST['c'];
8   $d =$b*$b-4*$a*$c; echo"  d = ", $d;
9 if($d<0)echo "<br>Нет вещественных корней!";
10 if($d>0)echo "<br>x1 = ",((-$b+sqrt($d))/(2*$a))," x2 = ",((-$b-sqrt($d))/(2*$a));
11 if($d==0)echo"<br>x1 = x2 = ",((-$b)/(2*$a));
12 echo"<br><br><a href=". $_SERVER['PHP_SELF'] .">Следующее</a>";
exit;}
13 else{
14 ?>
15 <form action="<?php echo $_SERVER['PHP_SELF'];?>" method="POST">
Введите коэффициенты квадратного уравнения a, b, c<br><br>
  a = <input type="text" name="a">
  b = <input type="text" name="b">
  c = <input type="text" name="c">
<input type="hidden" name="act"  value="act"><br><br>  //скрытая переменная
```

<input type="submit" name="submit" value="Решить">
<input type="reset" name="reset" value="Очистить">
</form></body></html>
16 <?php
17 }
18 ?>

1-я и 16-я строки содержат символы начала PHP-кода, а 14-я и 18-я — символы конца PHP-кода соответственно. В 15-й строке - внедрение PHP-кода между открывающим и закрывающим символами HTML-кода. Атрибут action указывает путь к ресурсу, на который возложена задача обработки введенных пользователем данных. В данном случае, сам этот файл и является тем ресурсом, который обрабатывает данные, введенные пользователем (можно было бы записать action="kvadrat.php". Во 2-й строке написано: если (if) некоторое условие выполняется, то выполняется код с 3-й по 12-ю строку, а иначе (else) переходим к 14-й строке и выполняем дальше по порядку.

Допустим, что условие в 3-ей строке не выполнено, тогда интерпретатор в 14-й строке закроет PHP-код, в 15-й строке выдаст название файла и «отдохнет» до 16-й строки. В 16-й строке он опять включится, выдаст закрывающую операторную скобку и отключится опять, закрыв PHP-код в 18-й строке. Строки, находящиеся между 14-й и 16-ой будут переданы web-сервером клиентскому браузеру, который выведет на экран форму ввода данных для пользователя.

В строке *// скрытая переменная* в форму вводится невидимая переменная, которая будет играть важную роль в PHP-коде. Тип этой переменной *hidden* можно перевести на русский язык как «спрятанная» или «скрытая», то есть переменная спрятана от пользователя. До щелчка на кнопке Submit переменные никуда не передаются. После щелчка на этой кнопке ресурс, указанный в атрибуте формы *action,* начинает их анализировать. Для передачи данных применен метод POST. При использовании этого метода передаваемые переменные заполняют массив *$_POST['имя переменной]* значениями соответствующей переменной *$_POST[имя переменной]=значение переменной*.

Во 2-й строке проверяется сложное условие, состоящее из двух частей. Первая часть условия - установлена ли *(isset)* переменная *act*. Второе условие - равно ли *(==)* значение этой переменной, в данном случае *act*. Эти условия соединены знаком &&, который означает, что они должны выполняться оба одновременно. Это условие, если нет неисправностей на сервере, всегда выполняется при щелчке на кнопке *Submit.* А если сложное условие выполнено, PHP-интерпретатор выполняет строки кода с 2-ой по 12-ю. Строки 5-7 одинаковы, они выводят на экран пользователя введенные им ранее значения. То, что заключено в двойные кавычки, функция echo отображает на экране без изменений, как правило, отображая и пробелы (иногда не все). Вместо символов элементов массива отображаются их значения. Для соединения подписи со значением служит точка (можно было использовать и запятую). Оператор вывода на экран echo завершается точкой с запятой. Следующий в каждой строке -

оператор присвоения. Значения элементов массива присваиваются трем переменным — $a, $в и $с соответственно. В 8-й строке, используя уже введенные переменные, определяется дискриминант $d и выводится его значение на экран с помощью оператора echo. Для разнообразия используем в качестве связки между надписью и значением не точку, как в предыдущих строчках, а запятую.

Из математики известно, что в зависимости от значения дискриминанта $d существует три варианта решения квадратного уравнения. В зависимости от значения $d выполняется одна из строк кода - 9я, 10-я или 11-я. Все три строки начинаются с проверки условия, и если условие не выполняется, то строка пропускается PHP-интерпретатором.

В строках 3, 4 используются два логических оператора «!» - НЕ и «||» - ИЛИ, а также встроенная PHP-функция *is_numeric(значение).* Эта функция является истинной (true), если значение введено и является числом. В противном случае функция возвращает ложь (false). Условие в строке 3 можно объяснить так: если хотя бы одно из введенных пользователем значений отсутствует или не является числом, следует выполнять код строки 4, иначе строка 4 пропускается. 4-я строка очень похожа на 12-ю, однако вместо функции echo используется функция exit (информация). А это означает, что после вывода информации из круглых скобок функции exit() программа завершается. При этом пользователь получит сообщение об ошибке и возможность вернуться к началу.

В предыдущем сценарии использовались однострочные поля ввода. Однако в сети встречаются и другие способы ввода информации пользователем.

Пример 4. Сценарий с использованием множественного и единственного выбора представлен ниже.

```
<html><head><title>Data Form</title></head><body>
<?php
if (isset($_POST['act'])&& $_POST['act'] == 'act')
{
        echo"ИМЯ ".$_POST['name'];
        echo"<br>Email ".$_POST['email'];
        echo"<br><u>Выбрано элементов - ".(count($_POST) - 6)."</u>";
   for($j=0;$j<5;$j++){
   if(isset($_POST['chk'.$j]))echo"<br>Выбрано - ".$_POST['chk'.$j];
   }
         echo"<br>Единственный выбор -  ".$_POST['rad'];
         echo"<br>Комментарий - ".$_POST['coment'];
        echo"<br><br><a href=". $_SERVER['PHP_SELF'] .">Еще раз</a>";
        exit;}
 else{
?>
<form action="<?php echo $_SERVER['PHP_SELF'];?>" method="POST">
```

```
Имя: <input type="text" name="name"><br>
Email:<input type="text" name="email"><br><br>
Множественный выбор:<br>
<?php
for($i=1; $i<4; $i++){
echo "<INPUT TYPE='checkbox' NAME='chk$i' VALUE='Выбор $i'>Выбор $i<br>";
}
?>
<br>Единственный выбор:<br>
<INPUT TYPE=radio NAME="rad" VALUE="Выбор 1" CHECKED>Выбор 1<br>
<INPUT TYPE=radio NAME="rad" VALUE=" Выбор 2">Выбор 2<br>
<INPUT TYPE=radio NAME="rad" VALUE="Выбор 3">Выбор 3<br><br>
Комментарий:<br>
<textarea name="coment" rows=5 cols=20 >Здесь комментарий </textarea><br><br>
<input type="hidden" name="act" value="act">
<input type="submit" name="submit" value="Отправить">
<input type="reset" name="reset" value="Очистить">
</form></body></html>
<?php
}
?>
```

На рис.3.3 приведен пример выполнения PHP-сценария с использованием полей ввода данных (однострочные поля Email и Имя ()) и кнопок управления (Очистить, Отправить).

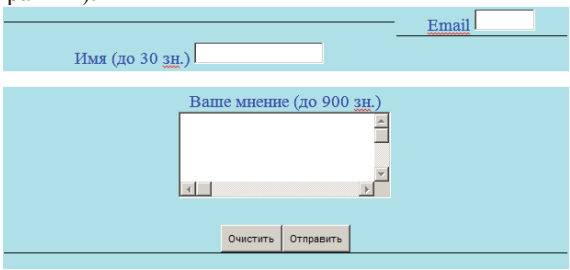

Рисунок 3.3- Пример выполнения PHP-сценария

3.4.4. Генератор паролей на PHP

Пароль генерируется случайным образом при помощи функции *uniqid*. Эта функция возвращает уникальный идентификатор, основываясь на значениях текущего времени в микросекундах.

При таком варианте использования функции возвращается 128-битный хеш-код.

Пример программы генерации пароля содержащего лишь буквы английского языка в нижнем регистре и цифры, по алгоритму MD5, приведен ниже.

```
<?php
        $id = md5(uniqid (rand), true));
        print  $id. “<br>”;
?>
```

Для генерации более стойкого к подбору паролей можно воспользоваться скриптом приведённым ниже.

```
<form method="post">
<input type="text" name="number" value="10">
<input type="submit" value="Генерировать">
<form><br><br>
<?php
 // Параметр $number - сообщает число   // символов в пароле
 echo generate_password(intval($_POST['number']));

 function generate_password($number)
 {
  $arr = array('a','b','c','d','e','f',
            'g','h','i','j','k','l',
            'm','n','o','p','r','s',
            't','u','v','x','y','z',
            'A','B','C','D','E','F',
            'G','H','I','J','K','L',
            'M','N','O','P','R','S',
            'T','U','V','X','Y','Z',
            '1','2','3','4','5','6',
            '7','8','9','0','.',',',
            '(',')','[',']','!','?',
            '&','^','%','@','*','$',
            '<','>','/','|','+','-',
            '{','}','`','~');
  // Генерируем пароль
  $pass = "";
  for($i = 0; $i < $number; $i++)
  {
   // Вычисляем случайный индекс массива
   $index = rand(0, count($arr) - 1);
   $pass .= $arr[$index];
  }
```

```
  return $pass;
 }
?>
```

Пример работы скрипта приведен на рис.3.4.

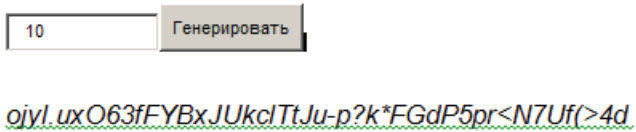

Рисунок 3.4 – Пример результата работы программы генерации равномерного пароля

3.5. Вопросы к защите лабораторной работы

1) Поясните основные способы и средствами получения информации в Интернет.
2) Назовите основные способы адресации информационных ресурсов в Интернет.
4) Перечислите основные способы поиска информации в Интернет.
5) Перечислите основные отличия поисковых систем от каталогов.
6) Назовите основные типы информационно-поисковых языков.
7) Что понимается под релевантностью?
8) Назовите компоненты служб WWW.
9) Что такое PHP, его возможности?
10) Поясните способы отделения PHP-кода от общего кода HTML.
11) Сформулируйте правила для выбора имени переменной в PHP.
12) Поясните способы ввода данных пользователем.
13) Поясните алгоритм проверки открытия файла PHP-сценарием
14) Как выполняется проверка вводимой пользователем информации?
15) Поясните процесс формирования элементов пользовательского интерфейса
16) Поясните назначение переменной *hidden.*
17) Приведите несколько способов указания пути к ресурсу, на который возложена задача обработки введенных пользователем данных.

3.6. Список рекомендуемой литературы

1) Романчева Н.И. Современные Интернет-технологии: Учебное пособие.-М.:РИО МГТУГА, 2007.
1) Котеров Д. PHP 7 /Д. В. Котеров, И. В. Симдянов. — СПб.: БХВ-Петербург, 2016. — 1088 с.
3) http://microsoft.com/php