Факультет компьютерных наук
Кафедра вычислительных систем

КОНТРОЛЬНЫЕ ЗАДАНИЯ ПО ДИСЦИПЛИНЕ
«ПРОГРАММИРОВАНИЕ СЕТЕВЫХ ПРИЛОЖЕНИЙ»

Задание 1.

Написать клиент для сервера, осуществляющего поиск указанной пользователем строки в некотором текстовом файле. Сервер принимает строки (группу символов, оканчивающихся символом конца строки), которые имеют формат «команда аргумент». Команды должны начинаться с символа «\». Пустые строки игнорируются. Сервер использует порт 1130/TCP.

Сервер поддерживает следующие команды:

• \search string — запрос на поиск подстроки string в файле данных. Возвращаются все строки файла данных, в которых встречается подстрока string. Вывод завершается строкой \end.

• \quit — окончание работы данного клиента. Возвращается сообщение «Closing connection» и соединение закрывается.

Коды ответа сервера:

• «400 Unknown command» — указанная команда не поддерживается сервером;

• «401 Empty argument» — пустой аргумент команды \search;

• «402 Illegal command» — полученная от клиента строка не начинается с команды.

Клиент считывает команды, вводимые пользователем, через стандартный ввод и пересылает их на сервер. Приняв ответ сервера, клиент подсчитывает количество найденных сервером строк и отображает на экране строки и результат подсчета. Строка, содержащая команду \end, при этом игнорируется. Сообщения об ошибках сервера выводятся на экран.

Задание 2.

Написать клиент для сервера, возвращающего случайную последовательность символов заданной длины из множества «abcdefghij».

Датаграмма клиента должна начинаться со строки запроса, которая имеет формат «команда аргумент». Команды начинаются с символа «\».

Ответы сервера начинаются со строки состояния в формате «код_ответа описание аргументы». Данные отделяются от строки состояния пустой строкой. Размер отправляемых сервером датаграмм не превышает 1000 байт. Сервер использует порт 1140/UDP.

Сервер поддерживает следующие команды:

• \get number (number — положительное целое) — запрос на генерацию случайной последовательности символов из вышеуказанного множества длины number. Возвращается сгенерированная последовательность, оканчивающаяся символом конца строки. Если размер ответа превышает 1000 байт, он разбивается

на несколько датаграмм. Строка состояния в каждой датаграмме ответа имеет вид: «200 OK begin=number1 more=number2», где number1 — номер первого символа в данной датаграмме, number2 равно 0, если данная датаграмма является последней датаграммой ответа, и 1 в противном случае.

Коды ответа сервера:

• «400 Unknown command» — команда, указанная в строке запроса, не поддерживается сервером;

• «401 Empty argument» — пустой аргумент команды \get;

• «402 Argument is not int» — аргумент команды \get не является целым числом;

• «500 Illegal request format» — неверный формат запроса клиента.

Клиент считывает команды, вводимые пользователем, через стандартный ввод и пересылает их на сервер. Приняв ответ сервера, клиент подсчитывает, сколько раз встречается каждый символ в полученной от сервера случайной последовательности, и отображает результаты на экране. Сообщения об ошибках сервера выводятся на экран.

Задание 3.

Написать программу с графическим интерфейсом для автоматического получения новых сообщений с почтового сервера по протоколу POP3.

Программа периодически устанавливает соединение с почтовым сервером и проверяет наличие новых сообщений в почтовом ящике некоторого пользователя. Если новые сообщения есть, программа получает их с сервера и сохраняет в локальном файле. При этом на экран выводится список полученных сообщений с указанием адреса отправителя и темы каждого сообщения. Если новых сообщений нет, на экран выводится сообщение об этом.

В программе должна быть предусмотрена возможность указания IP-адреса почтового сервера, имени и пароля пользователя. Программа должна корректно обрабатывать ошибки протокола POP3 и выводить сообщения о них.

Задание 4.

Написать программу с графическим интерфейсом для отправки электронных сообщений через почтовый сервер по протоколу SMTP.

Программа предоставляет пользователю интерфейс для указания IP-адреса почтового сервера, адреса получателя сообщения, обратного адреса, темы сообщения, а также ввода текста сообщения. Программа должна корректно обрабатывать ошибки протокола SMTP и выводить сообщения о них.

Задание 5.

Написать программу с графическим интерфейсом, которая запрашивает с Web-сервера некоторый гипертекстовый документ и проверяет корректность всех локальных ссылок, содержащихся в данном документе.

Программа предоставляет пользователю интерфейс для ввода полного URL запрашиваемого документа. По введенному URL определяются доменное имя или IP-адрес сервера, с которым нужно установить соединение, и относительный URL документа, который затем используется при формировании HTTP-запроса. В случае указания некорректного URL выводится сообщение об ошибке.

Программа извлекает из полученного документа все ссылки на документы, находящиеся на том же сервере, проверяет, существуют ли они, и выводит на экран их список. Для существующих документов указывается их размер, для не существующих — пометка, что документ не найден.

Программа должна обрабатывать ошибки протокола HTTP и выводить сообщения о них.

Задание 6.

Написать клиент-серверное приложение для обмена сообщениями по локальной сети с помощью широковещательных UDP-датаграмм.

В начале сеанса работы клиент каким-то образом регистрируется на сервере. Затем он отсылает введенные пользователем сообщения на сервер с помощью UDP-датаграмм, а также принимает широковещательные датаграммы, рассылаемые сервером, и отображает содержащиеся в них сообщения. Перед завершением работы клиент отправляет серверу сообщение об окончании сеанса.

Сервер поддерживает список активных клиентов — при регистрации клиента добавляет его идентификатор в этот список; при получении сообщения о завершении работы клиента удаляет его идентификатор из списка. Сервер принимает датаграммы от зарегистрированных клиентов, прибавляет к содержащимся в них сообщениям идентификатор клиента и отсылает на широковещательный IP-адрес.

Сообщения от незарегистрированных клиентов отбрасываются, их отправителям посылается сообщение об ошибке.

Задание 7.

Написать клиент-серверное приложение, позволяющее загрузить на компьютер клиента указанный пользователем файл.

Сервер должен обрабатывать следующие запросы пользователя: вывод списка доступных файлов, загрузку выбранного пользователем файла, завершение работы. Клиент обеспечивает для пользователя графический интерфейс. Для связи клиента с сервером используется протокол TCP.

Задание 8.

Написать CGI-приложение, позволяющее пользователю прочитать с сервера предназначенные для него сообщения. Все сообщения содержатся в одном текстовом файле. CGI- приложение просматривает этот файл, выбирает из него только те сообщения, которые нужны данному клиенту, и формирует из них Web-

страницу. Клиенты различаются по IP-адресам компьютеров, с которых поступают запросы. Если для некоторого компьютера сообщений нет, создается Web-страница, содержащая сообщение об ошибке.

Задание 9.

Написать CGI-приложение, отображающее информацию о пользователях.

Данные о пользователях находятся на сервере в нескольких файлах: в одном файле — фамилии, имена и отчества, во втором — домашние адреса, в третьем — место работы и должность. Приложение принимает регистрационное имя пользователя, извлекает из файлов информацию об этом пользователе и формирует из нее Web-страницу. Если данные о пользователе с указанным именем отсутствуют, создается Web-страница, содержащая сообщение об ошибке.

Задание 10.

Написать CGI-приложение, позволяющее через сервер отправить сообщение по электронной почте.

Приложение должно вызываться со страницы, на которой имеется интерфейс для создания электронного сообщения: текстовые поля для адреса получателя и темы сообщения и текстовая область для тела сообщения. CGI-приложение принимает эти данные от клиента, а затем отправляет сообщение через почтовый сервер, используя протокол SMTP. IP-адрес почтового сервера указывается в коде CGI-приложения. По окончании SMTP-сеанса, приложение формирует Web-страницу с информацией о результате данной попытки отправить сообщение.

Вопросы к экзамену

1. Абстрактные сокеты и их параметры.
2. Адресные структуры для стека семейства TCP/IP и функции для работы с ними.
3. Общая схема работы TCP-сокетов. Обмен данными через TCP-сокет.
4. Реализация TCP-клиента.
5. Реализация TCP-сервера.
6. UDP-сокеты и особенности их использования.
7. Присоединенные UDP-сокеты.
8. Блокируемый ввод-вывод через TCP-сокеты.
9. Неблокируемый ввод-вывод через TCP-сокеты.
10. Мультиплексирование ввода-вывода в сетевом программировании.
11. Ввод-вывод через TCP-сокеты, управляемый сигналом.
12. Асинхронный ввод-вывод через TCP-сокеты.
13. Последовательный сервер.
14. Параллельный сервер с использованием процессов.
15. Параллельный сервер с использованием потоков.
16. Неструктурированные сокеты.
17. Особенности ввода через неструктурированные сокеты.
18. Интерфейс с канальным уровнем.
19. Программирование TCP-клиента на Java.
20. Программирование TCP-сервера на Java.
21. Работа с UDP-сокетами в Java.
22. Объектная модель сетевого программирования в C++ Builder.
23. Архитектура технологии Remote Procedure Call (RPC).
24. Реализация клиента и сервера в технологии RPC.
25. Учет состояний сеанса в технологии RPC.
26. Архитектура технологии CORBA.
27. Службы CORBA.
28. Интерфейс CGI.
29. Механизм ключиков (cookies) в CGI-программировании.
30. Технологии обработки данных на стороне сервера (SSI, ASP, JSP, PHP).

СПИСОК ЛИТЕРАТУРЫ

1. *Печерицын А.А.* Программирование сетевых приложений. - Омск: Изд-во ОмГУ, 2008.
2. *Фленов М.Е.* Программирование на C++ глазами хакера. – СПб.: БХВ – Петербург, 2005.
3. *Стивенс У.Р.* Unix: разработка сетевых приложений. – СПб.: Питер, 2004.
4. *Олифер В.Г., Олифер Н.А.* Сетевые операционные системы. – СПб.: Питер, 2003.
5. *Моли Б.* Unix/Linux: теория и практика программирования. - М.: КУДИЦ-ОБРАЗ, 2004.