

Санкт-Петербургский политехнический университет Петра Великого  
Институт компьютерных наук и технологий  
Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе  
«Разработка клиентского приложения FTP»  
по дисциплине «Сети ЭВМ и телекоммуникации»

Работу выполнил:  
студент гр. 43501/3  
Хуторной Я. В.

Руководитель:  
Вылегжанина К. Д.  
«\_\_» \_\_\_\_\_ 2016 г

Санкт-Петербург  
2016

## **1. Техническое задание**

Разработать приложение для операционных систем семейства Windows или Linux, обеспечивающее функции клиента протокола FTP, работающего с сервером в пассивном режиме.

## **2. Основные возможности**

Приложение должно реализовывать следующие функции:

- Подключение к указанному серверу
- Получение списка файлов в каталоге
- Навигация по системе каталогов
- Копирование файла на сервер
- Копирование файла с сервера
- Создание и удаление каталогов
- Удаление файлов
- Обеспечение работы со ссылками на файлы и каталоги
- Поддержка передачи в бинарном и текстовом режиме
- Протоколирование соединения сервера с клиентом

## **3. Поддерживаемые команды**

Разработанное приложение должно реализовывать следующие команды протокола FTP:

- USER – передача серверу идентификационной информации пользователя
- PASS – передача серверу пароля пользователя
- CWD – смена текущего каталога сервера
- MKD – создание каталога
- RMD – удаление каталога
- DELE – удаление файла на сервере
- PASV – переключение сервера в пассивный режим
- LIST – получение списка файлов в текущем каталоге сервера в расширенном формате
- NLST – получение списка файлов в текущем каталоге сервера в сокращённом формате

- RETR – получение файла с сервера
- STOR – посылка файла на сервер
- QUIT – завершение сеанса

#### **4. Настройка приложения**

Разработанное приложение должно обеспечивать настройку следующих параметров:

- IP-адрес или доменное имя файлового сервера
- Имя пользователя
- Пароль пользователя
- Режим передачи: текстовый / бинарный

#### **5. Методика тестирования**

Для тестирования приложения следует использовать файловые серверы, имеющиеся в лаборатории, а также открытые файловые серверы, имеющиеся в сети Internet (ftp://ftp.funet.fi, ftp://ftp.relcom.ru и т.п.).

Для разработанных приложений проверяется возможности подсоединения к серверу, копирования файла на сервер, копирования файла с сервера, удаления файла на сервере, переименовании файла на сервере, создания каталога, удаления каталога.

#### **6. Теоретическое обоснование**

FTP - стандартный протокол, предназначенный для передачи файлов по ТСР-сетям. Использует 21-й порт. FTP часто используется для загрузки сетевых страниц и других документов с частного устройства разработки на открытые сервера хостинга.

Особенность протокола FTP в том, что он использует множественное (как минимум — двойное) подключение. При этом один канал является управляющим, через который поступают команды серверу и возвращаются его ответы (обычно через ТСР-порт 21), а через остальные происходит собственно передача данных, по одному каналу на каждую передачу.

FTP может работать в активном или пассивном режиме, от выбора которого зависит способ установки соединения. В активном режиме клиент создаёт управляющее ТСР-соединение с сервером и отправляет серверу свой

IP-адрес и произвольный номер клиентского порта, после чего ждёт, пока сервер запустит TCP-соединение с этим адресом и номером порта. В случае, если клиент находится за брандмауэром и не может принять входящее TCP-соединение, может быть использован пассивный режим. В этом режиме клиент использует поток управления, чтобы послать серверу команду PASV, и затем получает от сервера его IP-адрес и номер порта, которые затем используются клиентом для открытия потока данных с произвольного клиентского порта к полученному адресу и порту.

FTP-аутентификация использует схему имя пользователя/пароль для предоставления доступа. Имя пользователя посылается серверу командой USER, а пароль — командой PASS. Если предоставленная клиентом информация принята сервером, то сервер отправит клиенту приглашение и начинается сессия. Пользователи могут, если сервер поддерживает эту особенность, войти в систему без предоставления учётных данных, но сервер может предоставить только ограниченный доступ для таких сессий. Хост, обеспечивающий FTP-сервис, может предоставить анонимный доступ к FTP. Пользователи обычно входят в систему как «anonymous» (может быть регистрозависимым на некоторых FTP-серверах) в качестве имени пользователя. Хотя обычно пользователей просят прислать адрес их электронной почты вместо пароля, никакой проверки фактически не производится. Многие FTP-хосты, предоставляющие обновления программного обеспечения, поддерживают анонимный доступ.

Ниже представлено краткое описание кодов ответа, которые могут быть возвращены FTP-сервером. Эти коды были стандартизированы IETF в RFC 959. Код ответа — трёхзначное число.

Первая цифра отвечает за один из трёх исходов: успех, отказ или указание на ошибку либо неполный ответ.

- 2xx — Успешный ответ
- 4xx/5xx — Команда не может быть выполнена
- 1xx/3xx — Ошибка или неполный ответ

Вторая цифра определяет тип ошибки:

- x0z — Синтаксическая.
- x1z — Информация. Соответствует информационному сообщению.
- x2z — Соединения. Сообщение относится к управляющему соединению либо к соединению данных.
- x3z — Соответствует сообщениям об аутентификации пользователя и его правах.

- x4z — Не определено.
- x5z — Файловая система. Соответствует сообщению о состоянии файловой системы.

Третья цифра окончательно специфицирует ошибку.

## 7. Структура приложения

Приложение написано на языке C++. Программа реализует все возможности, перечисленные в техническом задании. Для каждой команды реализованы одноименные функции:

```
#include <iostream>
#include <sys/types.h>
#include <windows.h>

using namespace std;

#pragma comment ( lib, "ws2_32.lib" )

int flag = 0;

+int init_sock(char *ip){ ... }
+int readServ(int s){ ... }
+int login(int client_sockfd){ ... }
+int init_data(int client_sockfd, char *ip){ ... }
+int list(int s, int s2){ ... }
+int nlst(int s, int s2){ ... }
+int retr(int s, int s2, char *file){ ... }
+int cwd(int s, char *dir){ ... }
+int quit(int s, int s2){ ... }
+int start (char *a){ ... }
+int mkd (int s, char *dir){ ... }
+int rmd (int s, char *dir){ ... }
+int dele (int s, char *name){ ... }
+int stor(int s, int s2, char *file){ ... }

+void main () { ... }
```

Рис. 1. Структура приложения

Функция `main` в основном состоит из оператора множественного выбора `switch`. В зависимости от выбора пользователем предлагаемых действий, осуществляется вызов той или иной функции. Код приложения представлен в приложении.

## 8. Тестирование приложения

Для тестирования клиентского приложения на компьютере был настроен сервер в программе FileZillaServer. Был добавлен пользователь "anonymouse" с паролем "password". Для него были установлены все права (чтение, запись, удаление).

### 8.1. Запуск приложения

После запуска приложения пользователю предлагается ввести `ip`-адрес сервера. После этого отображается меню с доступными пользователю действиями.

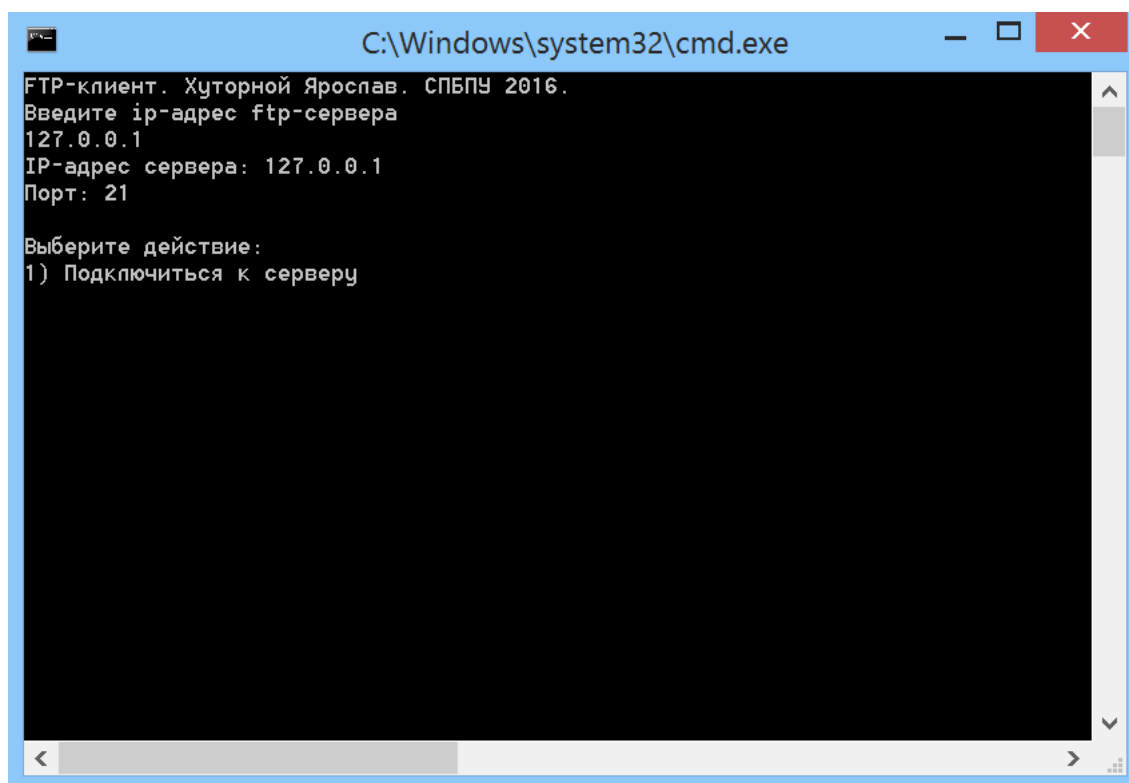
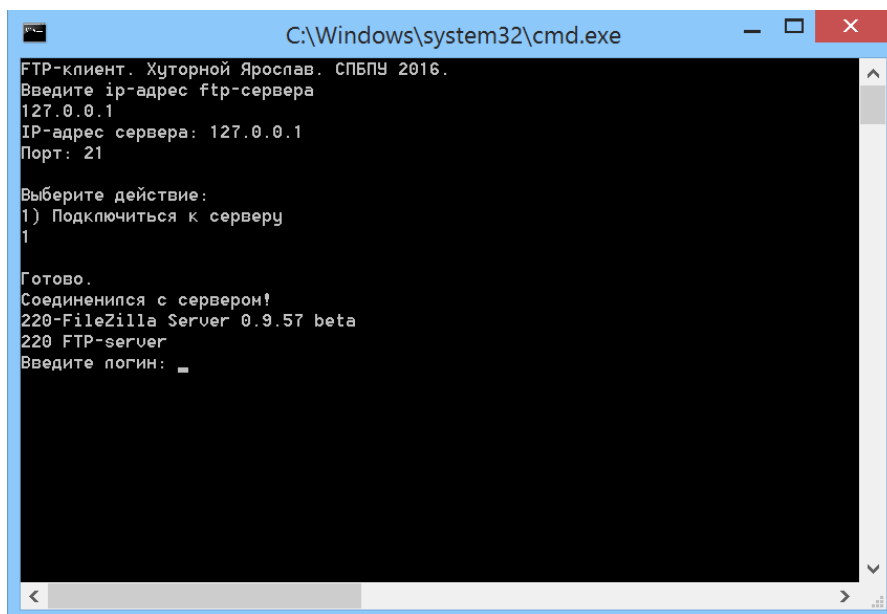


Рис. 2. Окно программы

## 8.2. Подключение к серверу

После ввода адреса сервера будет предложено подключиться к серверу. Для этого нужно выбрать действие "1) Подключиться к серверу". После подключения сервер отправляет клиенту сообщения с кодом 220 и приветственным текстом (рис. 3).



```
C:\Windows\system32\cmd.exe
FTP-клиент. Хуторной Ярослав. СПбПУ 2016.
Введите ip-адрес ftp-сервера
127.0.0.1
IP-адрес сервера: 127.0.0.1
Порт: 21

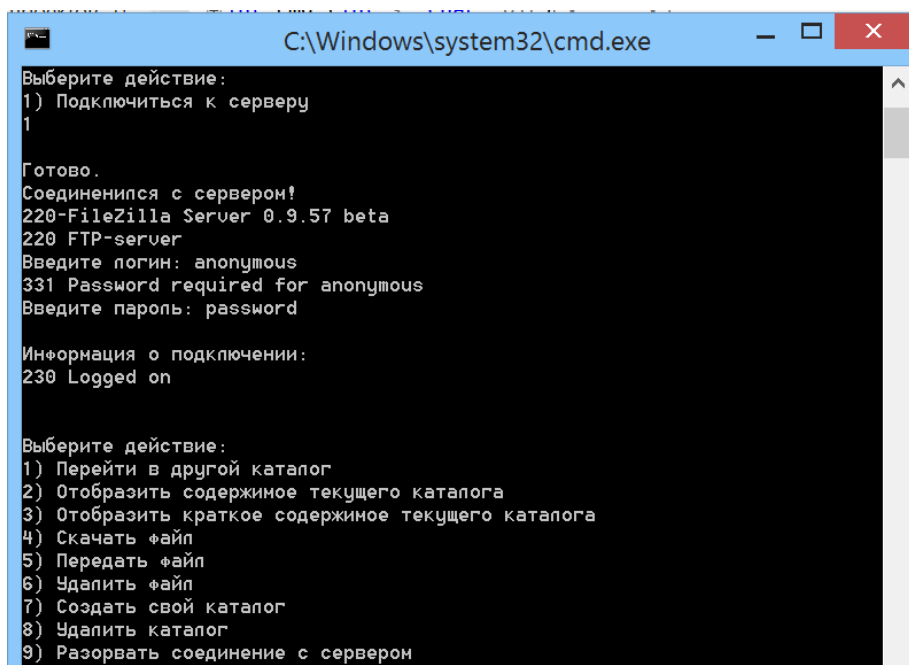
Выберите действие:
1) Подключиться к серверу
1

Готово.
Соединились с сервером!
220-FileZilla Server 0.9.57 beta
220 FTP-server
Введите логин: _
```

Рис. 3. Соединение с сервером

## 8.3. Процедура аутентификации клиента

Имя пользователя посылается серверу командой USER, а пароль — командой PASS.



```
C:\Windows\system32\cmd.exe
Выберите действие:
1) Подключиться к серверу
1

Готово.
Соединились с сервером!
220-FileZilla Server 0.9.57 beta
220 FTP-server
Введите логин: anonymous
331 Password required for anonymous
Введите пароль: password

Информация о подключении:
230 Logged on

Выберите действие:
1) Перейти в другой каталог
2) Отобразить содержимое текущего каталога
3) Отобразить краткое содержимое текущего каталога
4) Скачать файл
5) Передать файл
6) Удалить файл
7) Создать свой каталог
8) Удалить каталог
9) Разорвать соединение с сервером
```

Рис. 4. Процедура аутентификации

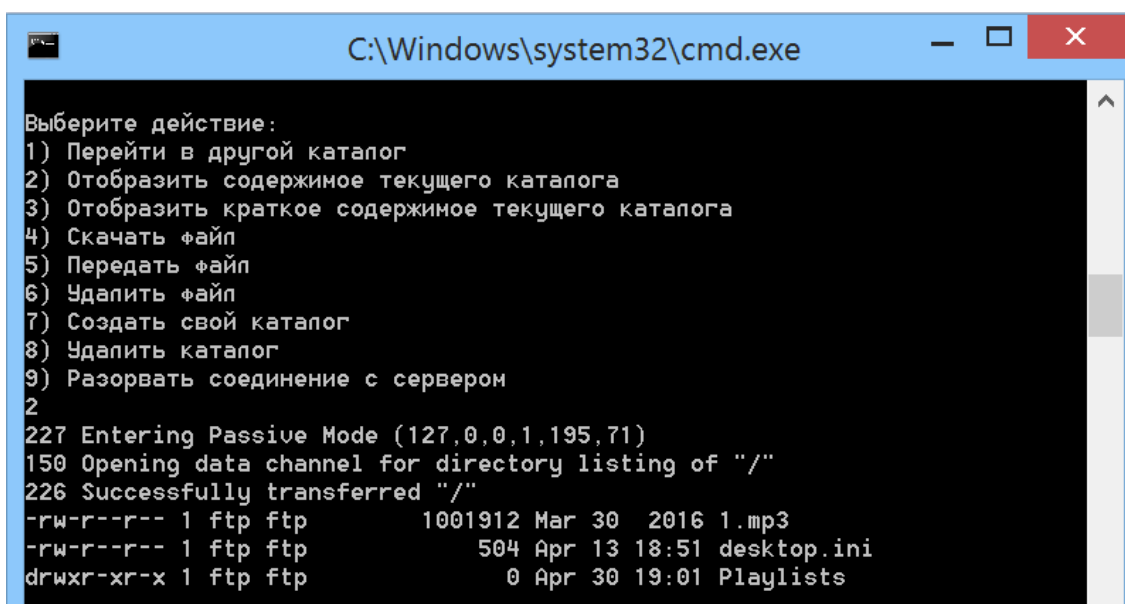
Процедуру подключения и аутентификации можно отследить в приложении сервера FileZillaServer:

```
(000040)17.05.2016 19:44:38 - (not logged in) (127.0.0.1)> Connected on port 21, sending welcome message...
(000040)17.05.2016 19:44:38 - (not logged in) (127.0.0.1)> 220-FileZilla Server 0.9.57 beta
(000040)17.05.2016 19:44:38 - (not logged in) (127.0.0.1)> 220 FTP-server
(000040)17.05.2016 19:44:41 - (not logged in) (127.0.0.1)> USER anonymous
(000040)17.05.2016 19:44:41 - (not logged in) (127.0.0.1)> 331 Password required for anonymous
(000040)17.05.2016 19:44:43 - (not logged in) (127.0.0.1)> PASS *****
(000040)17.05.2016 19:44:43 - anonymous (127.0.0.1)> 230 Logged on
```

Рис. 5. Процедура подключения и аутентификации на стороне сервера

#### 8.4. Отображение содержимого текущего каталога

Для отображения содержимого текущего каталога следует выбрать действие номер 2. При этом на сервер отсылается команда LIST и PASV для установления канала данных.



```
C:\Windows\system32\cmd.exe

Выберите действие:
1) Перейти в другой каталог
2) Отобразить содержимое текущего каталога
3) Отобразить краткое содержимое текущего каталога
4) Скачать файл
5) Передать файл
6) Удалить файл
7) Создать свой каталог
8) Удалить каталог
9) Разорвать соединение с сервером
2
227 Entering Passive Mode (127,0,0,1,195,71)
150 Opening data channel for directory listing of "/"
226 Successfully transferred "/"
-rw-r--r-- 1 ftp ftp      1001912 Mar 30  2016 1.mp3
-rw-r--r-- 1 ftp ftp         504 Apr 13 18:51 desktop.ini
drwxr-xr-x 1 ftp ftp           0 Apr 30 19:01 Playlists
```

Рис. 6. Результат выполнения команды LIST

Для отображения содержимого текущего каталога в кратком формате следует выбрать действие номер 3. При этом на сервер также отсылается команда PASV для установления канала данных и команда NLST.



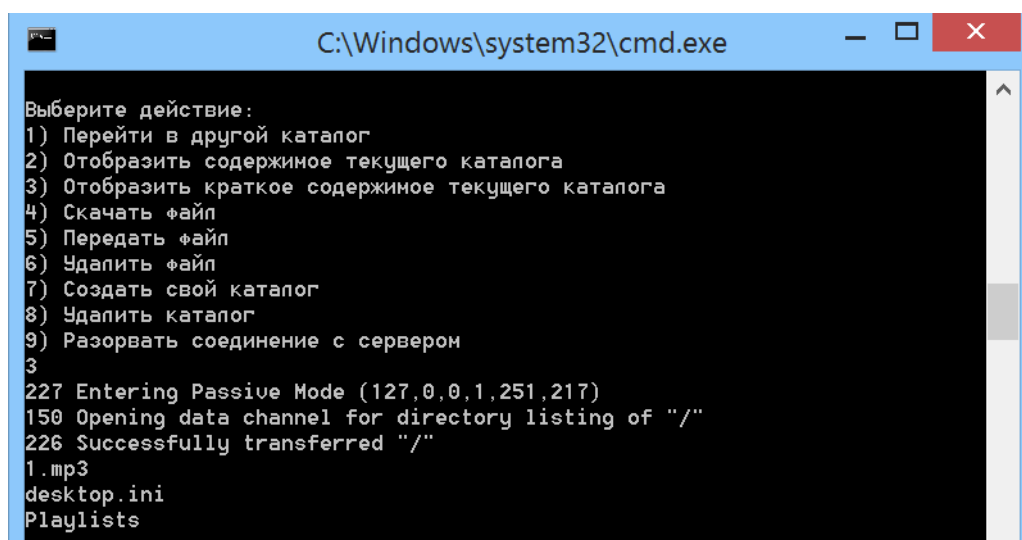


Рис. 7. Результат выполнения команды NLST

Процедуру отображения содержимого текущего каталога можно отследить в приложении сервера FileZillaServer:

```

[000041]17.05.2016 19:54:18 - anonymous (127.0.0.1)> PASV
[000041]17.05.2016 19:54:18 - anonymous (127.0.0.1)> 227 Entering Passive Mode (127,0,0,1,195,71)
[000041]17.05.2016 19:54:18 - anonymous (127.0.0.1)> LIST
[000041]17.05.2016 19:54:18 - anonymous (127.0.0.1)> 150 Opening data channel for directory listing of "/"
[000041]17.05.2016 19:54:18 - anonymous (127.0.0.1)> 226 Successfully transferred "/"
[000041]17.05.2016 19:55:12 - anonymous (127.0.0.1)> PASV
[000041]17.05.2016 19:55:12 - anonymous (127.0.0.1)> 227 Entering Passive Mode (127,0,0,1,251,217)
[000041]17.05.2016 19:55:12 - anonymous (127.0.0.1)> NLST
[000041]17.05.2016 19:55:12 - anonymous (127.0.0.1)> 150 Opening data channel for directory listing of "/"
[000041]17.05.2016 19:55:12 - anonymous (127.0.0.1)> 226 Successfully transferred "/"

```

Рис. 8. Процедуру отображения содержимого текущего каталога

## 8.5. Создание каталога

Для создания нового каталога нужно выбрать действие номер 7. После этого будет предложено ввести название каталога (рис. 8).

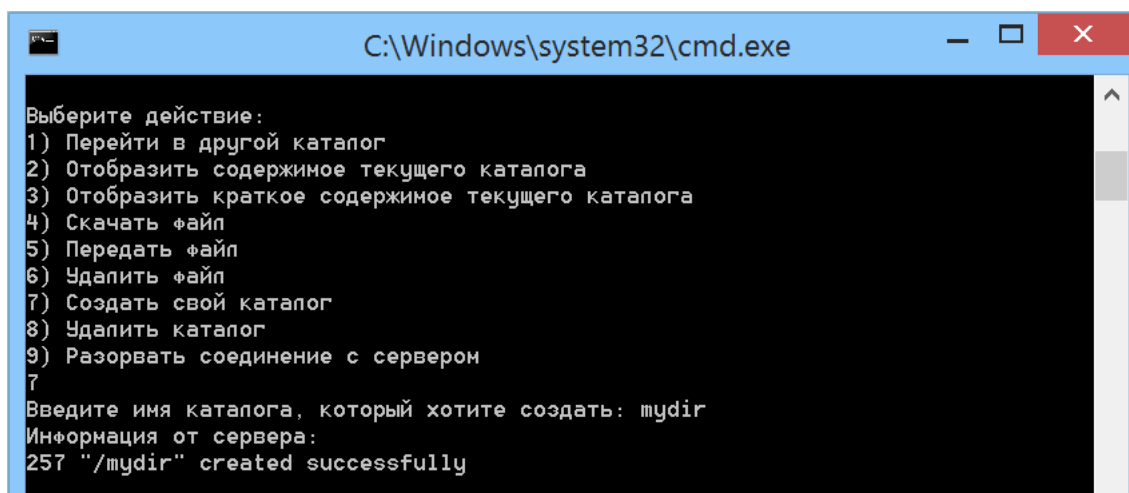
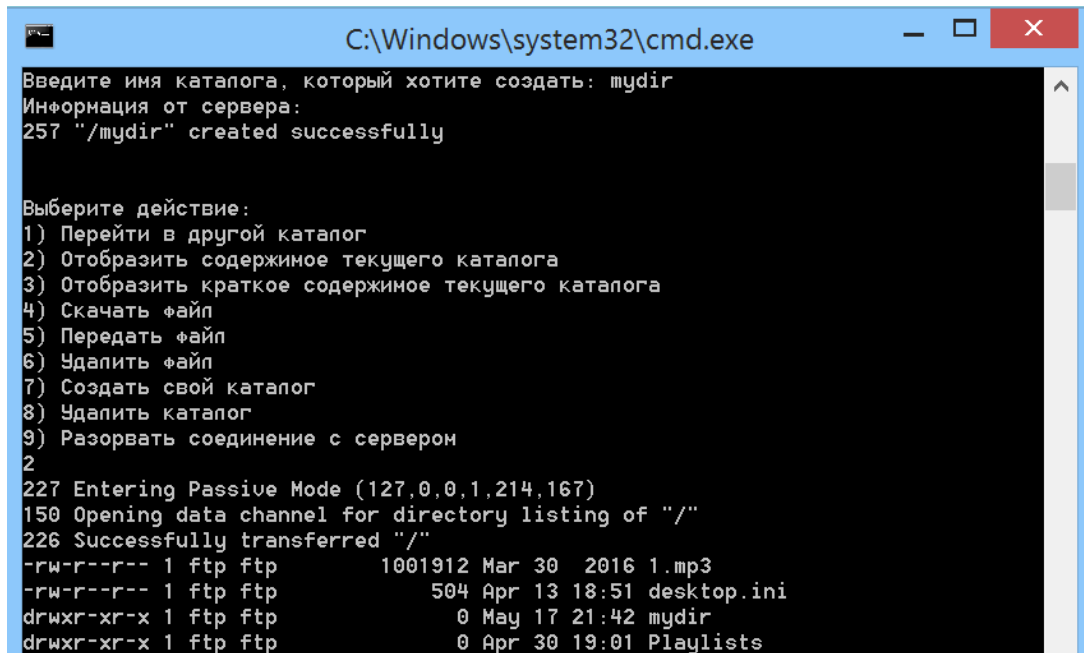


Рис. 8. Процедура создания каталога

Можно убедиться в том, что каталог создан, просмотрев содержимое текущего каталога (рис. 9).



```
C:\Windows\system32\cmd.exe
Введите имя каталога, который хотите создать: mydir
Информация от сервера:
257 "/mydir" created successfully

Выберите действие:
1) Перейти в другой каталог
2) Отобразить содержимое текущего каталога
3) Отобразить краткое содержимое текущего каталога
4) Скачать файл
5) Передать файл
6) Удалить файл
7) Создать свой каталог
8) Удалить каталог
9) Разорвать соединение с сервером
2
227 Entering Passive Mode (127,0,0,1,214,167)
150 Opening data channel for directory listing of "/"
226 Successfully transferred "/"
-rw-r--r-- 1 ftp ftp      1001912 Mar 30  2016 1.mp3
-rw-r--r-- 1 ftp ftp         504 Apr 13  18:51 desktop.ini
drwxr-xr-x 1 ftp ftp           0 May 17  21:42 mydir
drwxr-xr-x 1 ftp ftp           0 Apr 30  19:01 Playlists
```

Рис. 9. Процедура создания каталога

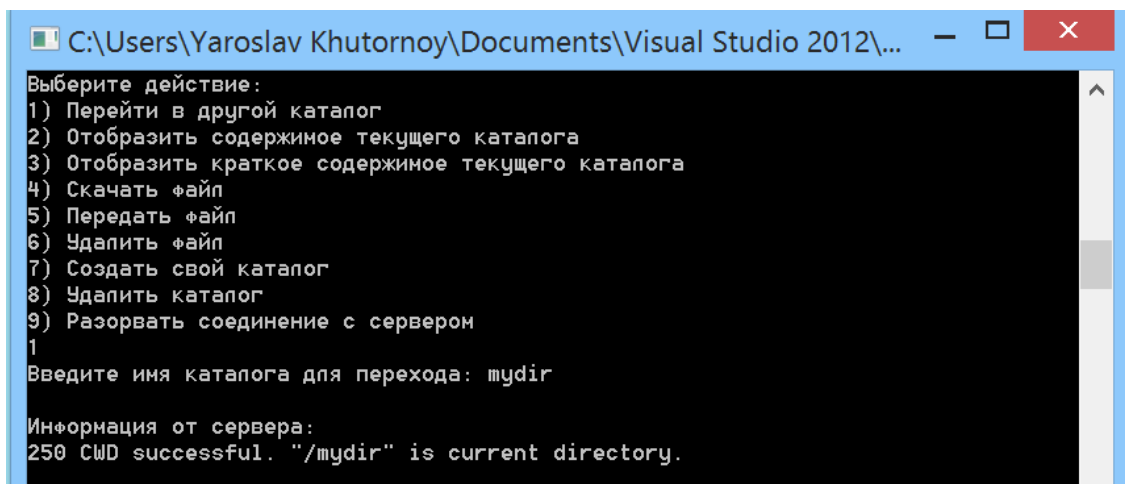


```
(000043)17.05.2016 21:42:16 - anonymous (127.0.0.1)> MKD mydir
(000043)17.05.2016 21:42:16 - anonymous (127.0.0.1)> 257 "/mydir" created successfully
(000043)17.05.2016 21:42:53 - anonymous (127.0.0.1)> PASV
(000043)17.05.2016 21:42:53 - anonymous (127.0.0.1)> 227 Entering Passive Mode (127,0,0,1,214,167)
(000043)17.05.2016 21:42:53 - anonymous (127.0.0.1)> LIST
(000043)17.05.2016 21:42:53 - anonymous (127.0.0.1)> 150 Opening data channel for directory listing of "/"
(000043)17.05.2016 21:42:53 - anonymous (127.0.0.1)> 226 Successfully transferred "/"
```

Рис. 10. Процедура создания каталога на сервере

## 8.6. Навигация по каталогам

Для того, чтобы перейти в другой каталог, следует выбрать действие номер 1. После этого будет предложено ввести название пути.



```
C:\Users\Yaroslav Khutornoy\Documents\Visual Studio 2012\...
Выберите действие:
1) Перейти в другой каталог
2) Отобразить содержимое текущего каталога
3) Отобразить краткое содержимое текущего каталога
4) Скачать файл
5) Передать файл
6) Удалить файл
7) Создать свой каталог
8) Удалить каталог
9) Разорвать соединение с сервером
1
Введите имя каталога для перехода: mydir
Информация от сервера:
250 CWD successful. "/mydir" is current directory.
```

Рис. 10. Процедура смены текущего каталога

```
(000045)17.05.2016 21:54:22 - anonymous (127.0.0.1)> CWD mydir
(000045)17.05.2016 21:54:22 - anonymous (127.0.0.1)> 250 CWD successful. "/mydir" is current directory.
```

Рис. 11. Процедура смены текущего каталога на сервере

### 8.7. Удаление каталога

Для того, чтобы удалить каталог, следует выбрать действие номер 6. Будет предложено ввести имя удаляемого каталога (рис. 11).

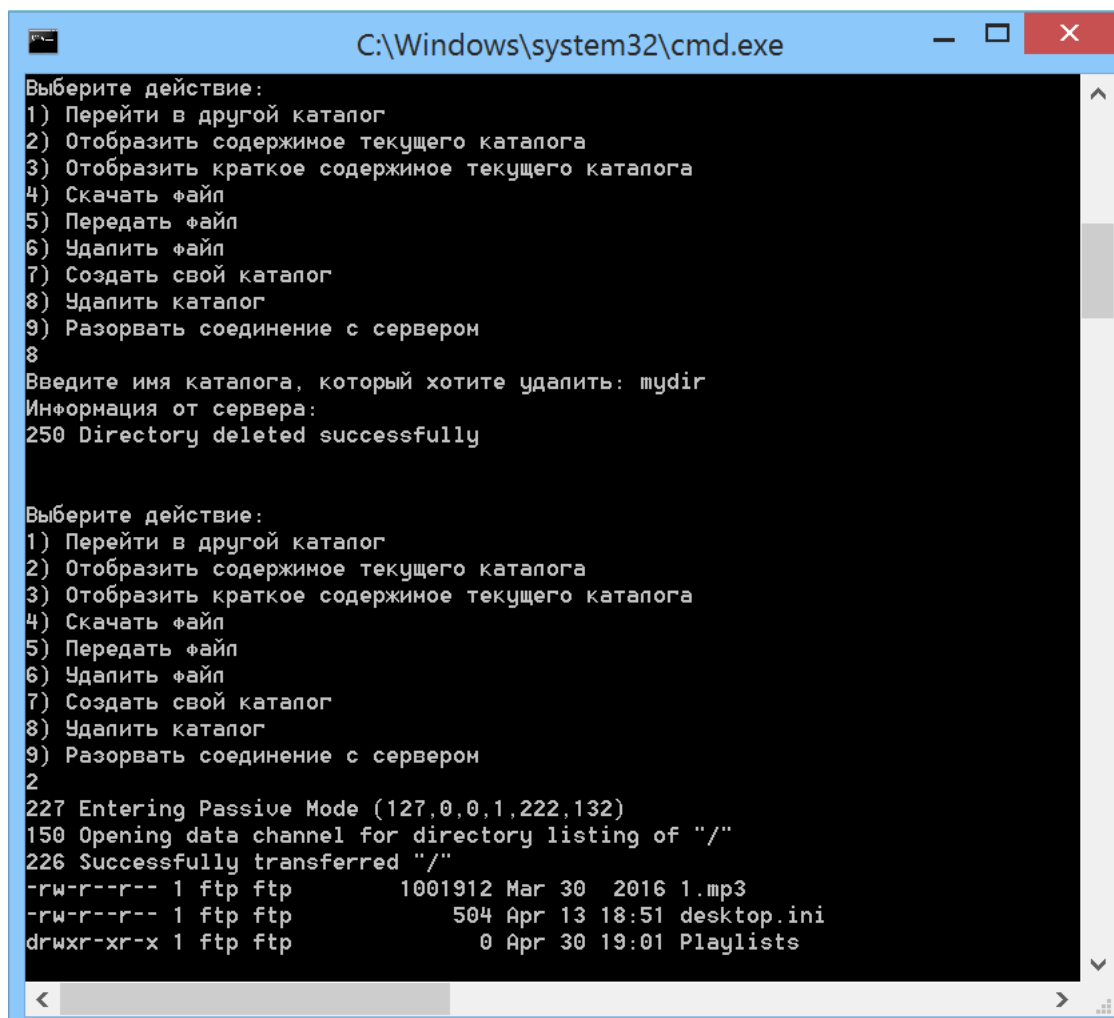


Рис. 11. Процедура удаления каталога

```
(000046)17.05.2016 22:32:21 - anonymous (127.0.0.1)> RMD mydir
(000046)17.05.2016 22:32:21 - anonymous (127.0.0.1)> 250 Directory deleted successfully
(000046)17.05.2016 22:32:26 - anonymous (127.0.0.1)> PASV
(000046)17.05.2016 22:32:26 - anonymous (127.0.0.1)> 227 Entering Passive Mode (127,0,0,1,222,132)
(000046)17.05.2016 22:32:26 - anonymous (127.0.0.1)> LIST
(000046)17.05.2016 22:32:26 - anonymous (127.0.0.1)> 150 Opening data channel for directory listing of "/"
(000046)17.05.2016 22:32:26 - anonymous (127.0.0.1)> 226 Successfully transferred "/"
```

Рис. 12. Процедура удаления каталога на сервере

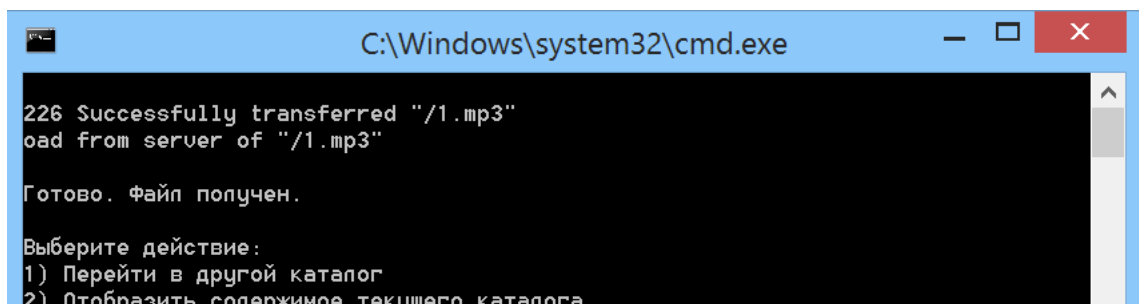
## 8.8. Скачивание файла

Для того, чтобы загрузить файл с сервера, следует выбрать действие номер 4. После этого будет предложено ввести название требуемого файла.

```
-rw-r--r-- 1 ftp ftp      1001912 Mar 30  2016 1.mp3
-rw-r--r-- 1 ftp ftp        504 Apr 13 18:51 desktop.ini
drwxr-xr-x 1 ftp ftp          0 Apr 30 19:01 Playlists

Выберите действие:
1) Перейти в другой каталог
2) Отобразить содержимое текущего каталога
3) Отобразить краткое содержимое текущего каталога
4) Скачать файл
5) Передать файл
6) Удалить файл
7) Создать свой каталог
8) Удалить каталог
9) Разорвать соединение с сервером
4
Введите имя файла, который нужно загрузить: 1.mp3
```

Рис. 13. Процедура удаления файла



```
C:\Windows\system32\cmd.exe

226 Successfully transferred "/1.mp3"
oad from server of "/1.mp3"

Готово. Файл получен.

Выберите действие:
1) Перейти в другой каталог
2) Отобразить содержимое текущего каталога
```

Рис. 14. Процедура удаления файла

```
(000047)17.05.2016 22:36:11 - anonymous (127.0.0.1)> LIST
(000047)17.05.2016 22:36:11 - anonymous (127.0.0.1)> 150 Opening data channel for directory listing of "/"
(000047)17.05.2016 22:36:11 - anonymous (127.0.0.1)> 226 Successfully transferred "/"
(000047)17.05.2016 22:36:17 - anonymous (127.0.0.1)> PASV
(000047)17.05.2016 22:36:17 - anonymous (127.0.0.1)> 227 Entering Passive Mode (127,0,0,1,207,210)
(000047)17.05.2016 22:36:17 - anonymous (127.0.0.1)> RETR 1.mp3
(000047)17.05.2016 22:36:17 - anonymous (127.0.0.1)> 150 Opening data channel for file download from server of "/1.mp3"
(000047)17.05.2016 22:36:17 - anonymous (127.0.0.1)> 226 Successfully transferred "/1.mp3"
```

Рис. 15. Процедура удаления файла на сервере

## 8.9. Загрузка файла на сервер

Для того, чтобы загрузить файл на сервер, следует выбрать действие номер 5. При этом сначала инициализируется канал для данных командой PASV, после чего отправляется команда STOR. После этого будет предложено ввести название файла (рис. 15).

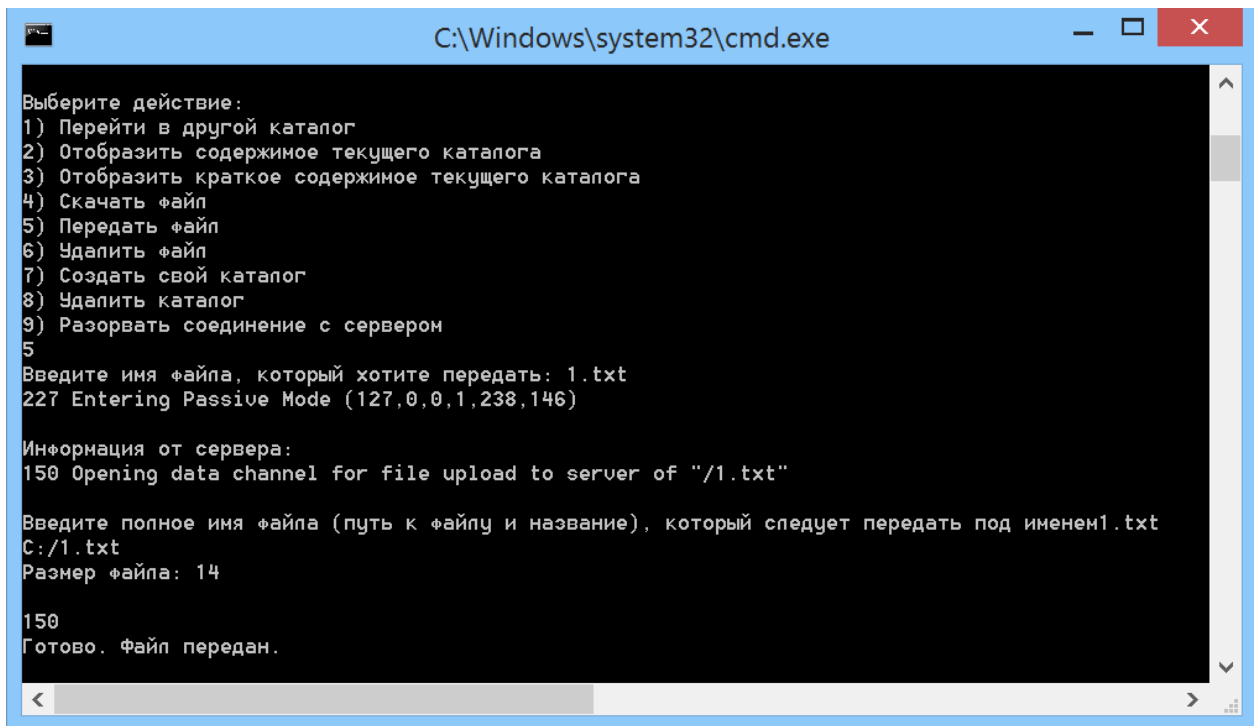


Рис. 15. Процедура загрузки файла на сервер

Мы можем убедиться в том, что файл передан, просмотрев содержимое текущего каталога:

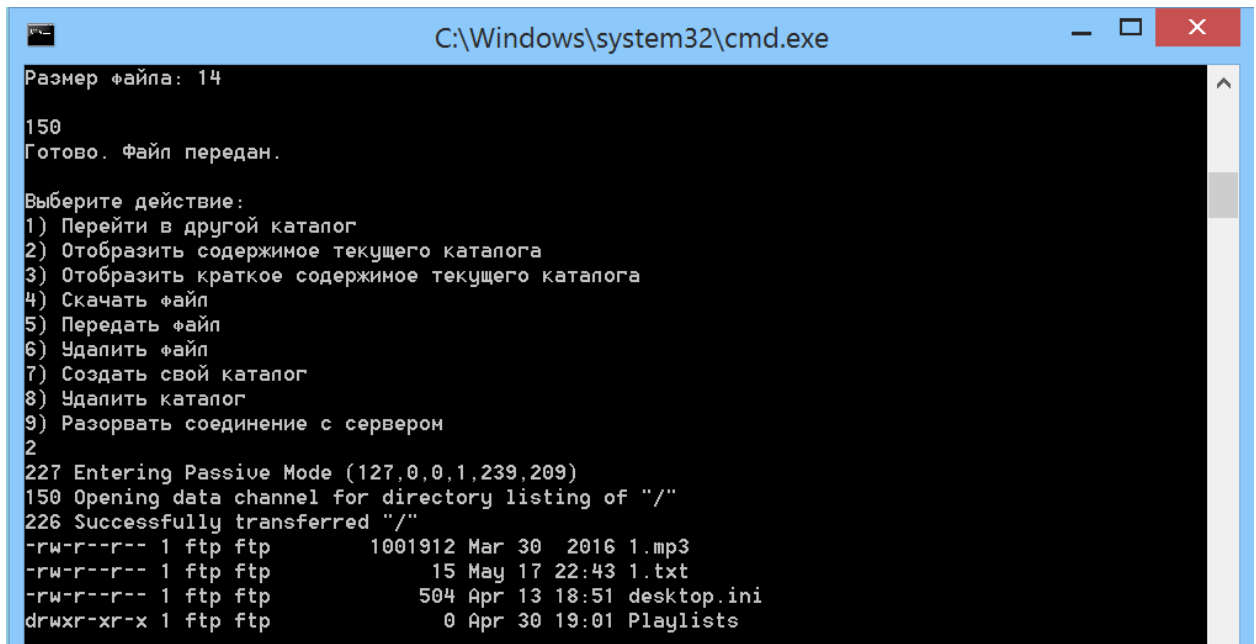


Рис. 16. Процедура загрузки файла на сервер

```
[000048]17.05.2016 22:43:25 - anonymous (127.0.0.1)> PASV
[000048]17.05.2016 22:43:25 - anonymous (127.0.0.1)> 227 Entering Passive Mode (127,0,0,1,238,146)
[000048]17.05.2016 22:43:25 - anonymous (127.0.0.1)> STOR 1.txt
[000048]17.05.2016 22:43:25 - anonymous (127.0.0.1)> 150 Opening data channel for file upload to server of "/1.txt"
```

Рис. 17. Процедура загрузки файла на сервер на сервере

## 8.10. Удаление файла

Для того, чтобы удалить файл на сервере, следует выбрать действие номер 8. После этого будет предложено ввести имя требуемого файла.

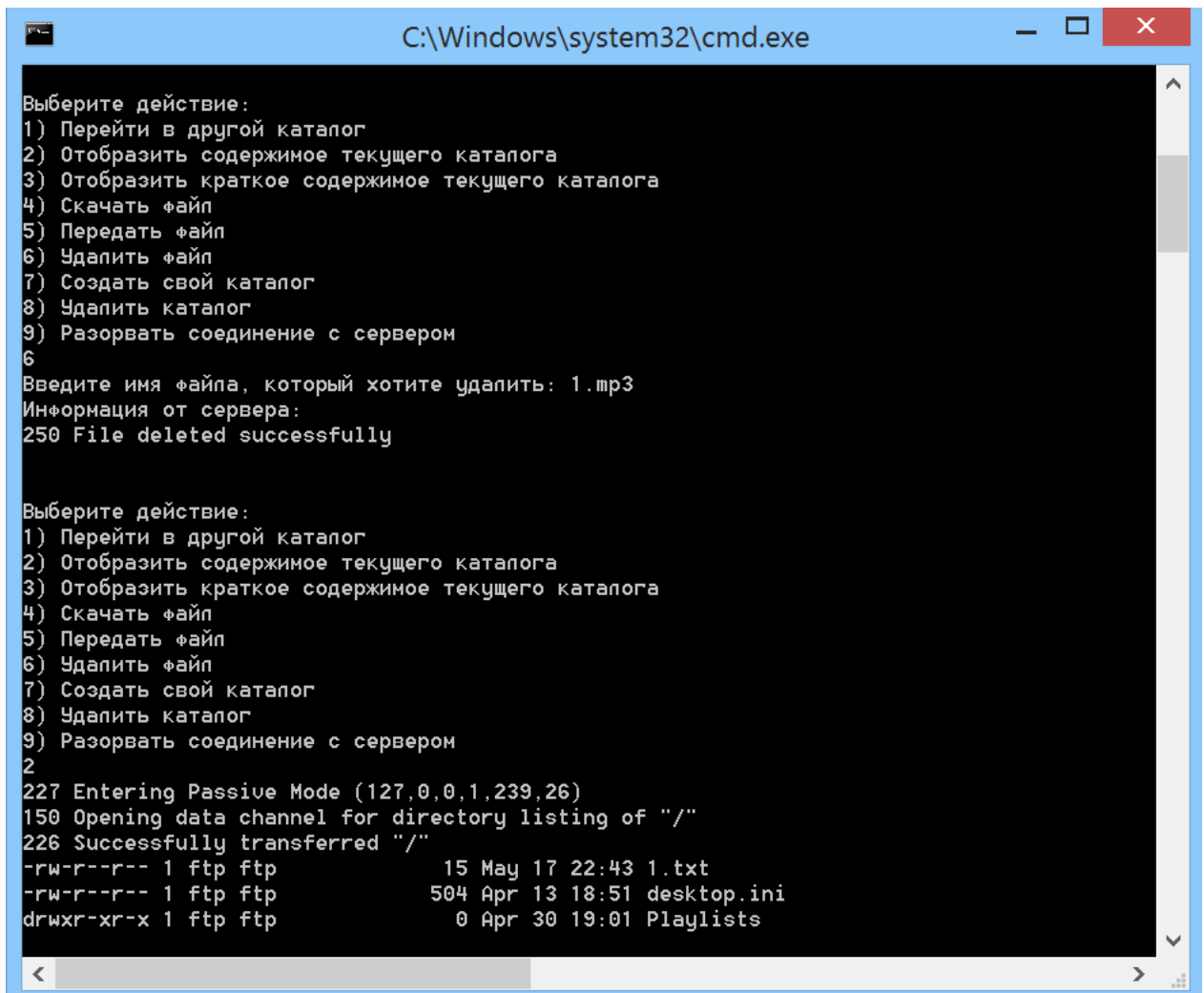


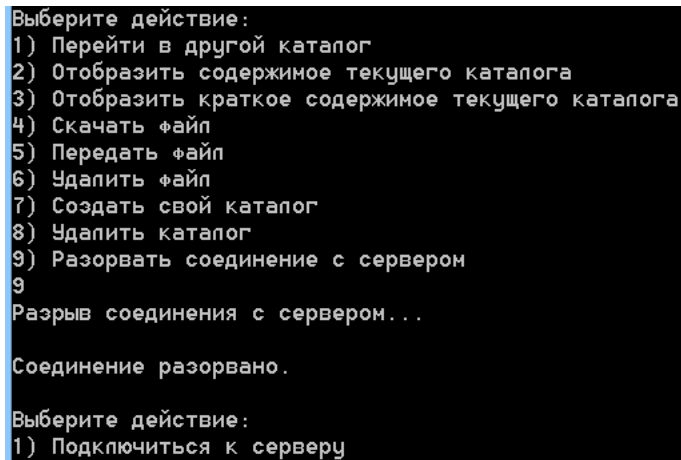
Рис. 18. Процедура удаления файла

```
(000049)17.05.2016 22:48:32 - anonymous (127.0.0.1)> DELE 1.mp3
(000049)17.05.2016 22:48:32 - anonymous (127.0.0.1)> 250 File deleted successfully
(000049)17.05.2016 22:48:40 - anonymous (127.0.0.1)> PASV
(000049)17.05.2016 22:48:40 - anonymous (127.0.0.1)> 227 Entering Passive Mode (127.0.0.1,239,26)
(000049)17.05.2016 22:48:40 - anonymous (127.0.0.1)> LIST
(000049)17.05.2016 22:48:40 - anonymous (127.0.0.1)> 150 Opening data channel for directory listing of "/"
(000049)17.05.2016 22:48:40 - anonymous (127.0.0.1)> 226 Successfully transferred "/"
```

Рис. 19. Процедура удаления файла на сервере

### 8.11. Разрыв соединения с сервером

Для того, чтобы разорвать соединение с сервером, следует выбрать действие номер 9.



```
Выберите действие:
1) Перейти в другой каталог
2) Отобразить содержимое текущего каталога
3) Отобразить краткое содержимое текущего каталога
4) Скачать файл
5) Передать файл
6) Удалить файл
7) Создать свой каталог
8) Удалить каталог
9) Разорвать соединение с сервером
9
Разрыв соединения с сервером...

Соединение разорвано.

Выберите действие:
1) Подключиться к серверу
```

Рис. 20. Процедура разрыва соединения

### Вывод

Был изучен протокол FTP и написано приложение FTP-клиент в пассивном режиме на языке C++. Тестирование выполнено успешно, ошибок не обнаружено.

## Приложение. Листинг программы.

```
#include <iostream>
#include <sys/types.h>
#include <windows.h>

using namespace std;

#pragma comment ( lib, "ws2_32.lib" )

int flag = 0;

int init_sock(char *ip) {
    char buff[1024];
    int len;
    int result;
    int client_sockfd;

    sockaddr_in address;

    if (WSAStartup(0x202, (WSADATA *)&buff[0]))
    {
        printf("WSAStart error %d\n", WSAGetLastError());
        return -1;
    }

    client_sockfd = socket(AF_INET , SOCK_STREAM , 0);
    if (client_sockfd == -1)
    {
        printf("Невозможно создать сокет!");
    }

    address.sin_family = AF_INET;
    address.sin_addr.s_addr = inet_addr(ip);
    address.sin_port = htons(21);

    len = sizeof(address);

    result = connect(client_sockfd, (sockaddr *)&address, len);
    if (result < 0)
    {
        perror("Невозможно подключиться к серверу");
        return 1;
    }

    printf("\nГотово.\n");
    printf("Соединен с сервером!\n");

    return client_sockfd;
}

int readServ(int s) {
    int rc;

    fd_set fdr;
    FD_ZERO(&fdr);
    FD_SET(s, &fdr);
    timeval timeout;
    timeout.tv_sec = 1;
    timeout.tv_usec = 0;
```



```

do {
    char buff[512] = { ' ' };
    recv(s,&buff[0],512,0);
    cout << buff;
    rc = select(s+1,&fdr,NULL,NULL,&timeout);
} while(rc);

return 2;
}

int login(int client_sockfd) {

    cout << "Введите логин: ";
    char name[64];
    cin >> name;

    char str[512];
    sprintf(str,"USER %s\r\n",name);
    send(client_sockfd,str,strlen(str),0);
    readServ(client_sockfd);

    cout << "Введите пароль: "; char pass[64]; cin >> pass;
    sprintf(str,"PASS %s\r\n",pass);
    send(client_sockfd,str,strlen(str),0);

    char size[512] = { ' ' };
    recv(client_sockfd,size,512,0);
    cout << "\nИнформация о подключении: \n" << size << endl;

    char *tmp_char;
    int a;
    tmp_char = strtok(size, " ");
    sscanf(tmp_char, "%d",&a);
    if (a/100 == 5)
    {
        flag = 0;
        return 0;
    }

    flag = 1;
    return 0;
}

int init_data(int client_sockfd, char *ip) {

    int client_data_sockfd;
    int a,b;
    int c,d,e,f;
    int len;
    int result;
    int port;

    char *tmp_char;
    char buff[128] = { ' ' };

    sockaddr_in address;

    send(client_sockfd,"PASV\r\n",strlen("PASV\r\n"),0);
    recv(client_sockfd,buff,128,0);

    cout << buff;

    tmp_char = strtok(buff,"(");
    tmp_char = strtok(NULL,"(");

```

```

tmp_char = strtok(tmp_char, "");
sscanf(tmp_char, "%d,%d,%d,%d,%d,%d",&c,&d,&e,&f,&a,&b);

port = a*256 + b;

client_data_sockfd = socket(AF_INET, SOCK_STREAM,0);

address.sin_family = AF_INET;
address.sin_addr.s_addr = inet_addr(ip);
address.sin_port = htons(port);

len = sizeof(address);

result = connect(client_data_sockfd, (sockaddr *)&address, len);
if (result == -1)
{
perror("Ошибка создания канала данных!");
return -1;
}

return client_data_sockfd;
}

int list(int s, int s2){

char buff[7] = "LIST\r\n";

send(s,buff,strlen(buff),0);
readServ(s);

int readed;

while(1)
{
char buff2[2048] = {' '};
readed = recv(s2,&buff2[0],sizeof(buff2),0);

if ((readed < sizeof(buff2)))
{
cout << buff2;
break;
}
cout << buff2;
}

return 0;
};

int nlst(int s, int s2){

char buff[7] = "NLST\r\n";
send(s,buff,strlen(buff),0);
readServ(s);

int readed;

while(1)
{
char buff2[2048] = {' '};
readed = recv(s2,&buff2[0],sizeof(buff2),0);

if ((readed < sizeof(buff2)))
{
cout << buff2;

```

```

        break;
    }
    cout << buff2;
}

    return 0;
};

int retr(int s, int s2, char *file) {

    int read_old;
    int read_new = 0;
    int i;
    char str[512];
    char *tmp_char;

    sprintf(str, "RETR %s\r\n", file);
    send(s, str, strlen(str), 0);

    char size[512] = { ' ' };
    recv(s, size, 512, 0);
    cout << "\nИнформация о скачиваемом файле: \n" << size << endl;

    int a;
    tmp_char = strtok(size, " ");
    sscanf(tmp_char, "%d", &a);
    if (a/100 == 5)
    {
        closesocket(s2);
        return 0;
    }

    FILE *f;
    f = fopen(file, "wb");

    if (f == NULL)
    {
        fputs("Ошибка создания файла", stderr);
        exit(1);
    }

    i = 0;

    do {
        char buff[2048];
        int readed = recv(s2, buff, sizeof(buff), 0);
        fwrite(buff, 1, readed, f);
        read_old = read_new;
        read_new += readed;

        if(i==1024)
        {
            i = 0;
            system("cls");
            cout << "\nСкачано данных: ";
            cout << read_new << " Байт\n";
        }

        i++;
    }
}

```

```

        while (read_old < read_new);

fclose(f);
system("cls");
recv(s,size,512,0);
cout << "\n" << size << endl;
cout << "Готово. Файл получен.\n";
return 0;
}

int cwd(int s, char *dir)
{
    char str[512];

    sprintf(str, "CWD %s\r\n", dir);
    send(s, str, strlen(str), 0);

    char size[512] = {' '};
    recv(s, size, 512, 0);

    cout << "\nИнформация от сервера: \n" << size << endl;
    return 0;
};

int quit(int s, int s2)
{
    char str[7] = {"QUIT\r\n"};
    char size[512] = {' '};
    cout << "Разрыв соединения с сервером...\n";
    send(s, str, strlen(str), 0);
    recv(s, size, 512, 0);
    cout << size << endl;
    cout << "Соединение разорвано.\n";
    flag = 0;
    closesocket(s);
    closesocket(s2);
    return 0;
};

int start (char *a)
{
    char buff[17] = {' '};
    int ip1, ip2, ip3, ip4;

    while(1)
    {
        cout << "FTP-клиент. Хуторной Ярослав. СПбПУ 2016." << endl;
        cout << "Введите ip-адрес ftp-сервера" << endl;
        cin >> buff;

        sscanf(buff, "%d.%d.%d.%d", &ip1, &ip2, &ip3, &ip4);

        if (((ip1 < 256) && (ip1 >= 0)) && ((ip2 < 256) && (ip2 >= 0)) && ((ip3 < 256) && (ip3 >= 0)) &&
            ((ip4 < 256) && (ip4 >= 0)))
        {
            sprintf(a, "%d.%d.%d.%d", ip1, ip2, ip3, ip4);

            cout << "IP-адрес сервера: " << a << endl;
            cout << "Порт: 21" << endl;
            break;
        }

        cout << "Неверный IP. Повторите.\n";
        Sleep(1000);
    }
}

```

```

        system("cls");
    }

    return 0;
};

int mkd (int s, char *dir)
{
    char size[512] = {' '};
    char str[512];

    sprintf(str, "MKD %s\r\n", dir);
    send(s, str, strlen(str), 0);

    recv(s, size, 512, 0);
    cout << "Информация от сервера:\n";

    cout << size << endl;

    return 0;
}

int rmd (int s, char *dir)
{
    char size[512] = {' '};
    char str[512];

    sprintf(str, "RMD %s\r\n", dir);
    send(s, str, strlen(str), 0);

    recv(s, size, 512, 0);
    cout << "Информация от сервера:\n";

    cout << size << endl;

    return 0;
}

int dele (int s, char *name)
{
    char size[512] = {' '};
    char str[512];

    sprintf(str, "DELE %s\r\n", name);
    send(s, str, strlen(str), 0);

    recv(s, size, 512, 0);
    cout << "Информация от сервера:\n";

    cout << size << endl;

    return 0;
}

int stor(int s, int s2, char *file)
{
    char str[512];
    char *tmp_char;
    char vvod[128] = {' '};

    sprintf(str, "STOR %s\r\n", file);
    send(s, str, strlen(str), 0);

```

```

    char size[512] = { ' ' };
    recv(s, size, 512, 0);
    cout << "\nИнформация от сервера: \n" << size << endl;

    int a;
    tmp_char = strtok(size, " ");
    sscanf(tmp_char, "%d", &a);
    if (a/100 == 5)
    {
        closesocket(s2);
        return 0;
    }

    cout<<"Введите полное имя файла (путь к файлу и название), который следует
передать под именем"<<file<<endl;
    cin>>vvod;

    FILE *f;
    f = fopen(vvod, "rb");

    if (f == NULL)
    {
        fputs("Ошибка файла", stderr);
        exit(1);
    }

    fseek(f, 0, SEEK_END);
    long lSize = ftell(f);
    rewind (f);

    cout << "Размер файла: " << lSize << "\n";

    char buff[2];
    int writed = 0;
    int i = 0;
    int summ = 0;

    while(!feof(f))
    {
        fgets(buff, sizeof(buff), f);
        writed = send(s2, buff, strlen(buff), 0);
        summ = summ + writed;

        if(i==1024)
        {
            i = 0;
            system("cls");
            cout << "\nПередано данных: ";
            cout << summ << " Байт\n";
        }
        i++;
    }

    fclose(f);

    cout << "\n" << size << endl;
    cout << "Готово. Файл передан.\n";

    return 0;
}

void main ()
{

```

```

setlocale(LC_CTYPE, "RUS");

int sok_info;
int sok_data;

char ip[18];
char vibor;
char vvod[128];

//cout <<"Адреса серверов для теста: \n";
//cout <<"193.166.3.2"<<endl;
//cout <<"89.108.120.88"<<endl<<endl;

start(ip);

while(1)
{
    cout << "\nВыберите действие: \n";

    switch(flag)
    {
    case 1:
    {
        cout << "1) Перейти в другой каталог\n";
        cout << "2) Отобразить содержимое текущего каталога\n";
        cout << "3) Отобразить краткое содержимое текущего каталога\n";
        cout << "4) Скачать файл\n";
        cout << "5) Передать файл\n";
        cout << "6) Удалить файл\n";
        cout << "7) Создать свой каталог\n";
        cout << "8) Удалить каталог\n";
        cout << "9) Разорвать соединение с сервером\n";
        break;
    }
    case 0:
    {
        cout << "1) Подключиться к серверу\n";
        break;
    }
    }

    cin >> vibor;

    switch(vibor)
    {
    case '1' :
    {
        if (flag == 0)
        {
            sok_info = init_sock(ip);
            readServ(sok_info);
            login(sok_info);
            break;
        }
        if (flag == 1)
        {
            cout << "Введите имя каталога для перехода: ";
            cin >> vvod;
            cwd(sok_info, vvod);
            break;
        }
    }
    }
}

```

```

case '4' :
{
if (flag == 1)
{
cout << "Введите имя файла, который нужно загрузить: ";
cin >> vvod;
sok_data = init_data(sok_info,ip);
retr(sok_info, sok_data, vvod);
break;
}
}
case '9' :
{
if (flag == 1)
{
quit(sok_info, sok_data);
break;
}
}
case '2' :
{
if (flag == 1)
{
sok_data = init_data(sok_info,ip);
list(sok_info,sok_data);
break;
}
}
case '3' :
{
if (flag == 1)
{
sok_data = init_data(sok_info,ip);
nlst(sok_info,sok_data);
break;
}
}
case '7' :
{
if (flag == 1)
{
cout << "Введите имя каталога, который хотите создать: ";
cin >> vvod;
mkd (sok_info, vvod);
break;
}
}
case '8' :
{
if (flag == 1)
{
cout << "Введите имя каталога, который хотите удалить: ";
cin >> vvod;
rmd (sok_info, vvod);
break;
}
}
case '6' :
{
if (flag == 1)
{
cout << "Введите имя файла, который хотите удалить: ";
cin >> vvod;
dele(sok_info, vvod);
}
}

```



```

        break;
    }
}
case '5' :
{
    if (flag == 1)
    {
        cout << "Введите имя файла, который хотите передать: ";
        cin >> vvod;
        sok_data = init_data(sok_info,ip);
        stor(sok_info, sok_data, vvod);
        break;
    }
}

default:
{
    cout << "В списке нет такого действия/n";
}

}

};

```