```
/*H*****************************************************************************************************
*
* File:       SFTPClient.c
*
* Author1:    Mani Bhargavi Ketha
* Author2:    Misha Yalavarthy
*
* Description: The program uses the Simple File Transfer Protocol to send ASCII and binary files
*                  from the client to the server.
*                  The client accepts five arguments which include the client executable code
*                  name,<input file name>,<output file name>,<IP address of server><port number>
*                  The client must be able to send the output file name to the server as well as the
*                  contents of the input file in chunk sizes of 10 bytes.
*                  The client must provide a way to inform the server of the completion of file transfer
*                  through the socket and must close the socket.
*
* Date:      October 15,2015
*
*
*
*H*/


#include<stdio.h>
#include<stdlib.h>
#include<sys/socket.h>   // Header defines the socket address stucture
#include<sys/types.h>
#include<netdb.h>      // Header defines the hostent structure
#include<string.h>

int main(int argc, char **argv)
{
        int sd,c,s,i;      //sd is the socket descriptor returned by the socket function

        char fname[50],fnameout[50],buff[10],*temp,*temp1;
/*fname & fnamout are the two buffers used to store outputfilename and input file contents*/

        struct sockaddr_in caddr;

        FILE *fp;

        socklen_t clen=sizeof(caddr);

        if(argc != 5)
/*must check whether four arguments are entered with first argument being ./STPClient*/
        {
        printf("\nUSAGE: $client<input file name><output file name><ipaddress><PORT>\n");
                exit(0);
        }
```

```
        sd=socket(AF_INET,SOCK_STREAM,0);
/*socket function, AF_INET is for ipv4 type and SOCK_STREAM is used for TCP protocol*/

        if(sd!=-1)
//socket descriptor returns two values 0(socket created) and -1(socket not created)
        {
                printf("Socket is created\n");
        }

        else
        {
                printf("Socket is not created\n");
        }
//socket has been created

//Initializing the socket address structure
        caddr.sin_family=AF_INET;
        caddr.sin_addr.s_addr=inet_addr(argv[3]);
        caddr.sin_port=htons(atoi(argv[4]));
/*Converting argument 4 from command line into string format and then into network format*/

        c=connect(sd,(struct sockaddr *) &caddr,sizeof(caddr));      //establishing connection

        if(c==0)          //checking whether connection has been established
                printf("Connected to server\n");
        else
        {
                printf("Not connected\n");
                return -1;
        }

        temp = argv[1];
        for(i=0;temp[i]!='\0';i++)
        {
          fname[i] = temp[i];//fname buffer now holds the input file(argument1) name
      }
      fname[i] = '\0';

//using buffer fnameout to send argument 2 (name of output file)

        temp1 = argv[2];

        for(i=0;temp1[i]!='\0';i++)
        {
          fnameout[i] = temp1[i];
//fnameout buffer now holds the output file name(argument2)
      }
```

```
        fnameout[i] = '\0';
           printf("\nOutput file : %s",fnameout);

           write(sd,fnameout,sizeof(fnameout),0);
//writing the contents of buffer(input file name) fnameout onto socket
/*
           if(send(sd,fname,sizeof(fname),0)!=0)
                    printf("file %s is transferring to server\n",fname);
*/


//Output file(argument 2) has been sent to server
//Now input file contents will be sent to the server through fread and write functions

           printf("\n\nInput file %s is transferring to server\n",fname);

           if((fp = fopen(fname,"rb"))==NULL)              //opening input file using argument 1
           {
          printf("Sorry can't open %s",fname);
          return -1;
        }

           while(!feof(fp))
//checking end of file condition while reading and writing contents of input file to socket
           {

           /*Using fread function the contents of the input file are read in chunk size of 10 bytes and
loaded into buffer*/
                    if(fread(buff,sizeof(char),10,fp))
                    {
               //printf("\nsending %s ",buff);
            write(sd,buff,sizeof(buff),0);   //The contents of buffer are written onto socket
              }
           }
           if(write(sd,"success",sizeof("success"),0)!=0)
           /*we are sending a success message to the server to indicate file transfer has been
completed*/
                    printf("\nfile transfer completed from client side\n");

           fclose(fp);        /*closing the input file*/

           close(sd);         /*closing the socket connection*/
           return 0;

}
```