Stop.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet recv_packet,ack_packet;
 int server_sock,client_sock,n,j=0;
 struct sockaddr_in serveraddr,clientaddr;
 socklen_t addr_size;
int main(int argc, char **argv)
{
 if(argc != 2)
  {
    printf("Error");
    exit(0);  }
 int port =atoi(argv[1]);
 server_sock=socket(AF_INET,SOCK_DGRAM,0);
 if(server_sock ==-1)
 {
  printf("Server not created\n");
  exit(0); }
 printf("Server is created successfully\n");
 memset(&serveraddr,'\0',sizeof(serveraddr));
 serveraddr.sin_family=AF_INET;
 serveraddr.sin_port=htons(port);
 serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
 bind(server_sock,(struct
sockaddr*)&serveraddr,sizeof(serveraddr));
 addr_size=sizeof(clientaddr);
 printf("Bind to Port number %d\n",port);
  while (1) {
     if
(recvfrom(server_sock,&recv_packet,sizeof(recv_packet),0
,(struct sockaddr*)&clientaddr,&addr_size) < 0) {
        perror("recvfrom failed");
        exit(0);    }
     printf("Frame received %s\n",recv_packet.data);
     printf("Received packet with sequence number:
%d\n", recv_packet.seq_num);
     sleep(2);
     ack_packet.seq_num = recv_packet.seq_num;
     if(j!=3)
     {
if(sendto(server_sock,&ack_packet,sizeof(ack_packet),0,(s
truct sockaddr*)&clientaddr,addr_size) < 0) {
         perror("sendto failed");
         exit(EXIT_FAILURE);
       }
      printf("Sent acknowledgment for sequence number:
%d\n", ack_packet.seq_num);
      }
     j++;
     if(recv_packet.flag == 1)
     {
       close(server_sock);
       exit(0);
     }
  }
  return 0;
}
```

wait.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet send_packet, recv_ack;
int main(int argc, char **argv) {
if(argc != 2)
  {
    printf("Error");
    exit(0);}
 int port = atoi(argv[1]);
 int clientsock,i=0;
 struct sockaddr_in addr;
 fd_set readfds;
 socklen_t addr_size;
 char word[] ="Hello my name is sree";
 clientsock=socket(AF_INET,SOCK_DGRAM,0);
 printf("Client is created succesfully\n");
 memset(&addr,'\0',sizeof(addr));
 addr.sin_family=AF_INET;
 addr.sin_port=htons(port);
 addr.sin_addr.s_addr=inet_addr("127.0.0.1");
 int base=0,window_size=3,j=0,flag=0;
 int length=strlen(word);
 addr_size=sizeof(addr);
```

```
  while (1) {
      for( i=base;i<window_size;i++)
      {
        send_packet.data[i]=word[i];
        if(window_size > strlen(word))
        {
          flag=1;
          send_packet.flag=1;
        } }
      send_packet.seq_num=j++;

sendto(clientsock,&send_packet,sizeof(send_packet),0,(str
uct sockaddr*)&addr,addr_size);
      printf("Frame sent\n");
            FD_ZERO(&readfds);
            FD_SET(clientsock, &readfds);
            struct timeval timeout;
            timeout.tv_sec = 3;
            timeout.tv_usec = 0;
      int a = select(clientsock+1, &readfds, NULL, NULL,
&timeout);
            if (a == -1) {
                        perror("select");
                        exit(EXIT_FAILURE);
                        }
                        else if (a == 0) {
                        printf("Timeout occurred. No data
received from server.\n");
                         window_size=3;
                         j=0;
                         base=0;
                         bzero(send_packet.data,1024);
                         length=strlen(word);  }
            else{

recvfrom(clientsock,&recv_ack,sizeof(recv_ack),0,(struct
sockaddr*)&addr,&addr_size);
                printf("Received acknowledgment for
sequence number: %d\n", recv_ack.seq_num);
                base=window_size;
                if(length <=3){
                  window_size=length;
                 }
                 else   {
                    window_size=window_size+3;
                              length=length-3;
                 }                    }

                        if(flag==1){
                          close(clientsock);
                          exit(0);
                         }              }
return 0;}
```

Output

server

```
Server is created successfully
Bind to Port number 5001
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Sent acknowledgment for sequence number: 3
Frame received Hello my name i
Received packet with sequence number: 4
Sent acknowledgment for sequence number: 4
Frame received Hello my name is s
Received packet with sequence number: 5
Sent acknowledgment for sequence number: 5
Frame received Hello my name is sree
Received packet with sequence number: 6
Sent acknowledgment for sequence number: 6
```

client

```
Client is created succesfully
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Timeout occurred. No data received from server.
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
```

```
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Received acknowledgment for sequence number: 3
Frame sent
Received acknowledgment for sequence number: 4
Frame sent
Received acknowledgment for sequence number: 5
Frame sent
Received acknowledgment for sequence number: 6
```

leaky

```
#include <stdio.h>
#include <unistd.h>
int main() {
   int n, incoming, outgoing, store = 0, bucketsize;
   printf("Enter the bucket size:- ");
   scanf("%d", &bucketsize);
   printf("Enter the outgoing rate:- ");
   scanf("%d", &outgoing);
   printf("Enter the number of inputs:- ");
   scanf("%d", &n);
   while (n > 0) {
      printf("\nEnter the incoming size:- ");
      scanf("%d", &incoming);
      printf("\nIncoming size is %d\n", incoming);
      if (incoming <= (bucketsize - store)) {
         store += incoming;
         printf("Bucket buffer size is %d out of %d\n",
store, bucketsize);
      } else {
         printf("packet loss = %d\n", incoming -
(bucketsize - store));
         store = bucketsize;
         printf("Buffer is full\n");
      }
      store -= outgoing;
      printf("After outgoing %d packets are left out of %d
in the buffer\n", store, bucketsize);
      n--;
   }
   return 0;
}
```

Output

```
Enter the bucket size:- 10
Enter the outgoing rate:- 4
Enter the number of inputs:- 3
Enter the incoming size:- 5
Incoming size is 5
Bucket buffer size is 5 out of 10
After outgoing 1 packets are left out of 10 in the buffer
Enter the incoming size:- 6
Incoming size is 6
Bucket buffer size is 7 out of 10
After outgoing 3 packets are left out of 10 in the buffer
Enter the incoming size:- 4
Incoming size is 4
Bucket buffer size is 7 out of 10
After outgoing 3 packets are left out of 10 in the buffer
```

Select.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet recv_packet,ack_packet;
 int server_sock,client_sock,n,j=0;
 struct sockaddr_in serveraddr,clientaddr;
 socklen_t addr_size;
int main(int argc, char **argv)
{
 if(argc != 2)
  {
    printf("Error");
    exit(0);
  }
 int port =atoi(argv[1]);
 server_sock=socket(AF_INET,SOCK_DGRAM,0);
 if(server_sock ==-1)
 {
  printf("Server not created\n");
  exit(0);
 }
 printf("Server is created successfully\n");
 memset(&serveraddr,'\0',sizeof(serveraddr));
```

```
 serveraddr.sin_family=AF_INET;
 serveraddr.sin_port=htons(port);
 serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
 bind(server_sock,(struct
sockaddr*)&serveraddr,sizeof(serveraddr));
 addr_size=sizeof(clientaddr);
 printf("Bind to Port number %d\n",port);
  while (1) {
     if
(recvfrom(server_sock,&recv_packet,sizeof(recv_packet
),0,(struct sockaddr*)&clientaddr,&addr_size) < 0) {
        perror("recvfrom failed");
        exit(0);
     }
     printf("Frame received %s\n",recv_packet.data);
     printf("Received packet with sequence number:
%d\n", recv_packet.seq_num);
     sleep(2);
     ack_packet.seq_num = recv_packet.seq_num;
     if(j!=3)
     {
 if
(sendto(server_sock,&ack_packet,sizeof(ack_packet),0,(
struct sockaddr*)&clientaddr,addr_size) < 0) {
        perror("sendto failed");
        exit(0);
     }
     printf("Sent acknowledgment for sequence
number: %d\n", ack_packet.seq_num);
     }
    j++;
    if(recv_packet.flag == 1)
    {
      close(server_sock);
      exit(0);
    }
  }
  return 0;
}
```

repeat.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
  int seq_num;
  char data[1024];
  int flag,i;
} packet;
packet send_packet, recv_ack;
int main(int argc, char **argv) {
if(argc != 2)
  {
    printf("Error");
    exit(0);}
 int port = atoi(argv[1]);
 int clientsock,i=0;
 struct sockaddr_in addr;
 fd_set readfds;
 socklen_t addr_size;
 char word[] ="Hello my name is sree";
 clientsock=socket(AF_INET,SOCK_DGRAM,0);
 printf("Client is created succesfully\n");
 memset(&addr,'\0',sizeof(addr));
 addr.sin_family=AF_INET;
 addr.sin_port=htons(port);
 addr.sin_addr.s_addr=inet_addr("127.0.0.1");
 int base=0,window_size=3,j=0,flag=0;
 int length=strlen(word);
 addr_size=sizeof(addr);
  while (1) {
      for(i=base;i<window_size;i++)
       {
         send_packet.data[i]=word[i];
         if(window_size > strlen(word))
          {
           flag=1;
           send_packet.flag=1;
          } }
      send_packet.seq_num=j++;

sendto(clientsock,&send_packet,sizeof(send_packet),0,(
struct sockaddr*)&addr,addr_size);
      printf("Frame sent\n");
            FD_ZERO(&readfds);
            FD_SET(clientsock, &readfds);
            struct timeval timeout;
            timeout.tv_sec = 3;
            timeout.tv_usec = 0;
      int a = select(clientsock+1, &readfds, NULL, NULL,
&timeout);
           if (a == -1) {
                    perror("select");
                    exit(0);     }
                    else if (a == 0) {
                     printf("Timeout occurred. No data
received from server.\n");
```

```
window_size=3*(send_packet.seq_num+1);
                        j=send_packet.seq_num;
                        base=window_size-3;
                        length=length+3;
                 }
            else{

recvfrom(clientsock,&recv_ack,sizeof(recv_ack),0,(struc
t sockaddr*)&addr,&addr_size);
                printf("Received acknowledgment for
sequence number: %d\n", recv_ack.seq_num);
                base=window_size;
                if(length <=3)
                {
                  window_size=length;
                }
                else{
                   window_size=window_size+3;
                          length=length-3;   }
}
                       if(flag==1)
                       {
                         close(clientsock);
                         exit(0);
                       }   }
   return 0;}
```

Output

```
Server is created successfully
Bind to Port number 6001
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Frame received Hello my nam
Received packet with sequence number: 3
Sent acknowledgment for sequence number: 3
Frame received Hello my name i
Received packet with sequence number: 4
Sent acknowledgment for sequence number: 4
Frame received Hello my name is s
Received packet with sequence number: 5
Sent acknowledgment for sequence number: 5
Frame received Hello my name is sree
Received packet with sequence number: 6
Sent acknowledgment for sequence number: 6
Frame received Hello my name is sree
Received packet with sequence number: 7
Sent acknowledgment for sequence number: 7
```

repeat.c

```
Client is created succesfully
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Timeout occurred. No data received from server.
Frame sent
Received acknowledgment for sequence number: 3
Frame sent
Received acknowledgment for sequence number: 4
Frame sent
Received acknowledgment for sequence number: 5
Frame sent
Received acknowledgment for sequence number: 6
Frame sent
Received acknowledgment for sequence number: 7
```

fs_ser.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/wait.h>
#include <errno.h>
#define PORT 8080
#define MAX_PENDING_CONNECTIONS 10
#define MAX_BUFFER_SIZE 1024
void handle_client_request(int client_socket) {
 char buffer[MAX_BUFFER_SIZE];
 ssize_t bytes_received;
 ssize_t bytes_sent;
pid_t pid = getpid();
 bytes_received = recv(client_socket, buffer,
sizeof(buffer), 0);
 if (bytes_received < 0) {
```

```
 perror("Error receiving data from client");
 exit(EXIT_FAILURE);
 }
 buffer[bytes_received] = '¥0';
 FILE* file = fopen(buffer, "rb");
 if (file != NULL) {
 while ((bytes_sent = fread(buffer, 1, sizeof(buffer), file))
> 0) {
 if (send(client_socket, buffer, bytes_sent, 0) !=
bytes_sent) {
 perror("Error sending file to client");
 exit(EXIT_FAILURE);
 }
 }
 fclose(file);
 } else {
const char* message = "File not found.";
 if (send(client_socket, message, strlen(message), 0) < 0)
{
 perror("Error sending message to client");
 exit(EXIT_FAILURE);
 }
 }
 snprintf(buffer, sizeof(buffer), "%d", pid);
 if (send(client_socket, buffer, strlen(buffer), 0) < 0) {
 perror("Error sending PID to client");
 exit(EXIT_FAILURE);
 }
 close(client_socket);
}
int main() {
 int server_socket, client_socket;
 struct sockaddr_in server_addr, client_addr;
 socklen_t client_addr_len = sizeof(client_addr);
 pid_t pid;
if ((server_socket = socket(AF_INET, SOCK_STREAM, 0))
< 0) {
 perror("Error creating socket");
 exit(EXIT_FAILURE);
 }
 memset(&server_addr, 0, sizeof(server_addr));
 server_addr.sin_family = AF_INET;
 server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
 server_addr.sin_port = htons(PORT);
 if (bind(server_socket, (struct sockaddr*)&server_addr,
sizeof(server_addr)) < 0) {
 perror("Error binding socket");
 exit(EXIT_FAILURE);
 }
 if (listen(server_socket,
MAX_PENDING_CONNECTIONS) < 0) {
 perror("Error listening on socket");
 exit(EXIT_FAILURE);
 }
 printf("Server listening on port %d...¥n", PORT);
 while (1) {
 if ((client_socket = accept(server_socket, (struct
sockaddr*)&client_addr, &client_addr_len)) <
0) {
 perror("Error accepting connection");
 exit(EXIT_FAILURE);
 }
 pid = fork();
 if (pid < 0) {
 perror("Error forking child process");
 exit(EXIT_FAILURE);
 } else if (pid == 0) {
 close(server_socket);
 handle_client_request(client_socket);
 exit(EXIT_SUCCESS);
 } else {
 close(client_socket);
 waitpid(-1, NULL, WNOHANG);
 }
 }
 close(server_socket);
 return 0;
}
```

fs_cli.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#define SERVER_IP "127.0.0.1"
#define PORT 8080
#define MAX_BUFFER_SIZE 1024
int main() {
 int client_socket;
 struct sockaddr_in server_addr;
 char filename[MAX_BUFFER_IZE];
 ssize_t bytes_received;
 if ((client_socket = socket(AF_INET, SOCK_STREAM, 0))
< 0) {
 perror("Error creating socket");
 exit(EXIT_FAILURE);
```

```
 }
 memset(&server_addr, 0, sizeof(server_addr));
 server_addr.sin_family = AF_INET;
 server_addr.sin_port = htons(PORT);
 server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
 if (connect(client_socket, (struct
sockaddr*)&server_addr, sizeof(server_addr)) < 0) {
 perror("Error connecting to server");
 exit(EXIT_FAILURE);
 }
 printf("Enter filename: ");
 fgets(filename, sizeof(filename), stdin);
 filename[strcspn(filename, "¥n")] = '¥0';
 if (send(client_socket, filename, strlen(filename), 0) <
0) {
perror("Error sending filename to server");
 exit(EXIT_FAILURE);
 }
 char buffer[MAX_BUFFER_SIZE];
 while ((bytes_received = recv(client_socket, buffer,
sizeof(buffer), 0)) > 0) {
 fwrite(buffer, 1, bytes_received, stdout);
 }
 if (bytes_received < 0) {
 perror("Error receiving data from server");
 exit(EXIT_FAILURE);
 }
 close(client_socket);
 return 0;
}
```

Output

fs_ser.c

```
Server listening to port 5505
```

fs_cli.c

```
Enter filename:you.txt
filenot found
PID:54186

Enter filename:sree.txt
HI IAM SREELAKSHMI M NAIR
PID:34035

//sree
//Nishal
```

Output

```
Enter the bucket size:- 10
Enter the outgoing rate:- 4
Enter the number of inputs:- 3
Enter the incoming size:- 5
Incoming size is 5
Bucket buffer size is 5 out of 10
After outgoing 1 packets are left out of 10 in the buffer
Enter the incoming size:- 6
Incoming size is 6
Bucket buffer size is 7 out of 10
After outgoing 3 packets are left out of 10 in the buffer
Enter the incoming size:- 4
Incoming size is 4
Bucket buffer size is 7 out of 10
After outgoing 3 packets are left out of 10 in the buffer
```

Stop.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet recv_packet,ack_packet;
 int server_sock,client_sock,n,j=0;
 struct sockaddr_in serveraddr,clientaddr;
 socklen_t addr_size;
int main(int argc, char **argv)
{
 if(argc != 2)
  {
    printf("Error");
    exit(0);  }
 int port =atoi(argv[1]);
 server_sock=socket(AF_INET,SOCK_DGRAM,0);
 if(server_sock ==-1)
 {
   printf("Server not created\n");
   exit(0); }
 printf("Server is created successfully\n");
 memset(&serveraddr,'\0',sizeof(serveraddr));
 serveraddr.sin_family=AF_INET;
 serveraddr.sin_port=htons(port);
 serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
```

```
  bind(server_sock,(struct
sockaddr*)&serveraddr,sizeof(serveraddr));
  addr_size=sizeof(clientaddr);
  printf("Bind to Port number %d\n",port);
    while (1) {
       if
(recvfrom(server_sock,&recv_packet,sizeof(recv_packet),0
,(struct sockaddr*)&clientaddr,&addr_size) < 0) {
          perror("recvfrom failed");
          exit(0);    }
       printf("Frame received %s\n",recv_packet.data);
       printf("Received packet with sequence number:
%d\n", recv_packet.seq_num);
       sleep(2);
       ack_packet.seq_num = recv_packet.seq_num;
       if(j!=3)
       {
       if
(sendto(server_sock,&ack_packet,sizeof(ack_packet),0,(str
uct sockaddr*)&clientaddr,addr_size) < 0) {
          perror("sendto failed");
          exit(EXIT_FAILURE);
       }
       printf("Sent acknowledgment for sequence number:
%d\n", ack_packet.seq_num);
       }
      j++;
      if(recv_packet.flag == 1)
       {
        close(server_sock);
        exit(0);
       }
    }
    return 0;
}
```

Wait.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
    int seq_num;
    char data[1024];
    int flag;
} packet;
packet send_packet, recv_ack;
int main(int argc, char **argv) {
if(argc != 2)
   {
    printf("Error");
    exit(0);}
  int port = atoi(argv[1]);
  int clientsock,i=0;
  struct sockaddr_in addr;
  fd_set readfds;
  socklen_t addr_size;
  char word[] ="Hello my name is Nishal";
  clientsock=socket(AF_INET,SOCK_DGRAM,0);
  printf("Client is created succesfully\n");
  memset(&addr,'\0',sizeof(addr));
  addr.sin_family=AF_INET;
  addr.sin_port=htons(port);
  addr.sin_addr.s_addr=inet_addr("127.0.0.1");
  int base=0,window_size=3,j=0,flag=0;
  int length=strlen(word);
  addr_size=sizeof(addr);
   while (1) {
       for( i=base;i<window_size;i++)
        {
          send_packet.data[i]=word[i];
          if(window_size > strlen(word))
          {
            flag=1;
            send_packet.flag=1;
          } }
       send_packet.seq_num=j++;

sendto(clientsock,&send_packet,sizeof(send_packet),0,(str
uct sockaddr*)&addr,addr_size);
       printf("Frame sent\n");
              FD_ZERO(&readfds);
              FD_SET(clientsock, &readfds);
              struct timeval timeout;
              timeout.tv_sec = 3;
              timeout.tv_usec = 0;
       int a = select(clientsock+1, &readfds, NULL, NULL,
&timeout);
              if (a == -1) {
                        perror("select");
                        exit(EXIT_FAILURE);
                        }
                        else if (a == 0) {
                        printf("Timeout occurred. No data
received from server.\n");
                         window_size=3;
                         j=0;
                         base=0;
                         bzero(send_packet.data,1024);
                         length=strlen(word);  }
              else{
```

```
recvfrom(clientsock,&recv_ack,sizeof(recv_ack),0,(struct
sockaddr*)&addr,&addr_size);
                printf("Received acknowledgment for
sequence number: %d\n", recv_ack.seq_num);
                base=window_size;
                if(length <=3)
                {
                  window_size=length;
                }
                else
                {
                   window_size=window_size+3;
                              length=length-3;
                }                    }

                          if(flag==1)
                          {
                            close(clientsock);
                            exit(0);
                          }
   }
   return 0;
}
```

output

stop.c

```
Server is created successfully
Bind to Port number 5002
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Sent acknowledgment for sequence number: 3
Frame received Hello my name i
Received packet with sequence number: 4
Sent acknowledgment for sequence number: 4
Frame received Hello my name is N
Received packet with sequence number: 5
Sent acknowledgment for sequence number: 5
Frame received Hello my name is Nish
Received packet with sequence number: 6
Sent acknowledgment for sequence number: 6
Frame received Hello my name is Nishal
Received packet with sequence number: 7
Sent acknowledgment for sequence number: 7
```

wait.c

```
Client is created succesfully
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Timeout occurred. No data received from server.
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Received acknowledgment for sequence number: 3
Frame sent
Received acknowledgment for sequence number: 4
Frame sent
Received acknowledgment for sequence number: 5
Frame sent
Received acknowledgment for sequence number: 6
Frame sent
Received acknowledgment for sequence number: 7
```

leaky

```
#include <stdio.h>
#include <unistd.h>
int main() {
   int n, incoming, outgoing, store = 0, bucketsize;
   printf("Enter the bucket size:- ");
   scanf("%d", &bucketsize);
   printf("Enter the outgoing rate:- ");
   scanf("%d", &outgoing);
   printf("Enter the number of inputs:- ");
   scanf("%d", &n);
   while (n > 0) {
      printf("\nEnter the incoming size:- ");
      scanf("%d", &incoming);
      printf("\nIncoming size is %d\n", incoming);
      if (incoming <= (bucketsize - store)) {
         store += incoming;
         printf("Bucket buffer size is %d out of %d\n",
store, bucketsize);
      } else {
         printf("packet loss = %d\n", incoming -
(bucketsize - store));
         store = bucketsize;
         printf("Buffer is full\n");
      }
      store -= outgoing;
      printf("After outgoing %d packets are left out of %d
in the buffer\n", store, bucketsize);
      n--;
   }
   return 0;
}
```

Select.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet recv_packet,ack_packet;
 int server_sock,client_sock,n,j=0;
 struct sockaddr_in serveraddr,clientaddr;
 socklen_t addr_size;
int main(int argc, char **argv)
{
 if(argc != 2)
  {
    printf("Error");
    exit(0);
  }
 int port =atoi(argv[1]);
 server_sock=socket(AF_INET,SOCK_DGRAM,0);
 if(server_sock ==-1)
 {
   printf("Server not created\n");
   exit(0);
 }
 printf("Server is created successfully\n");
 memset(&serveraddr,'\0',sizeof(serveraddr));
 serveraddr.sin_family=AF_INET;
 serveraddr.sin_port=htons(port);
 serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
 bind(server_sock,(struct
sockaddr*)&serveraddr,sizeof(serveraddr));
 addr_size=sizeof(clientaddr);
 printf("Bind to Port number %d\n",port);
  while (1) {
     if
(recvfrom(server_sock,&recv_packet,sizeof(recv_packet
),0,(struct sockaddr*)&clientaddr,&addr_size) < 0) {
        perror("recvfrom failed");
        exit(0);
     }
     printf("Frame received %s\n",recv_packet.data);
     printf("Received packet with sequence number:
%d\n", recv_packet.seq_num);
     sleep(2);
     ack_packet.seq_num = recv_packet.seq_num;
     if(j!=3)
     {
 if
(sendto(server_sock,&ack_packet,sizeof(ack_packet),0,(
struct sockaddr*)&clientaddr,addr_size) < 0) {
        perror("sendto failed");
        exit(0);
     }
     printf("Sent acknowledgment for sequence
number: %d\n", ack_packet.seq_num);
     }
    j++;
    if(recv_packet.flag == 1)
     {
      close(server_sock);
```

```
        exit(0);
       }
   }
   return 0;
}
```

repeat.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag,i;
} packet;
packet send_packet, recv_ack;
int main(int argc, char **argv) {
if(argc != 2)
  {
    printf("Error");
    exit(0);}
 int port = atoi(argv[1]);
 int clientsock,i=0;
 struct sockaddr_in addr;
 fd_set readfds;
 socklen_t addr_size;
 char word[] ="Hello my name is Nishal";
 clientsock=socket(AF_INET,SOCK_DGRAM,0);
 printf("Client is created succesfully\n");
 memset(&addr,'\0',sizeof(addr));
 addr.sin_family=AF_INET;
 addr.sin_port=htons(port);
 addr.sin_addr.s_addr=inet_addr("127.0.0.1");
 int base=0,window_size=3,j=0,flag=0;
 int length=strlen(word);
 addr_size=sizeof(addr);
  while (1) {
      for(i=base;i<window_size;i++)
      {
        send_packet.data[i]=word[i];
        if(window_size > strlen(word)){
          flag=1;
          send_packet.flag=1;
        } }
      send_packet.seq_num=j++;
sendto(clientsock,&send_packet,sizeof(send_packet),0,(
struct sockaddr*)&addr,addr_size);
      printf("Frame sent\n");
            FD_ZERO(&readfds);
            FD_SET(clientsock, &readfds);
            struct timeval timeout;
            timeout.tv_sec = 3;
            timeout.tv_usec = 0;
      int a = select(clientsock+1, &readfds, NULL, NULL,
&timeout);
            if (a == -1) {
                  perror("select");
                  exit(0);     }
                  else if (a == 0) {
                   printf("Timeout occurred. No data
received from server.\n");

window_size=3*(send_packet.seq_num+1);
                   j=send_packet.seq_num;
                   base=window_size-3;
                   length=length+3;}
            else{

recvfrom(clientsock,&recv_ack,sizeof(recv_ack),0,(struc
t sockaddr*)&addr,&addr_size);
                printf("Received acknowledgment for
sequence number: %d\n", recv_ack.seq_num);
                base=window_size;
                if(length <=3)
                {
                  window_size=length;
                }
                else{
                   window_size=window_size+3;
                      length=length-3;   }
}
                  if(flag==1)
                   {
                     close(clientsock);
                     exit(0);
                   }   }
   return 0;
}
```

Output

Select.c

```
Server is created successfully
Bind to Port number 6002
Frame received Hel
```

Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Frame received Hello my nam
Received packet with sequence number: 3
Sent acknowledgment for sequence number: 3
Frame received Hello my name i
Received packet with sequence number: 4
Sent acknowledgment for sequence number: 4
Frame received Hello my name is N
Received packet with sequence number: 5
Sent acknowledgment for sequence number: 5
Frame received Hello my name is Nish
Received packet with sequence number: 6
Sent acknowledgment for sequence number: 6
Frame received Hello my name is Nishal
Received packet with sequence number: 7
Sent acknowledgment for sequence number: 7

Repeat.c

Client is created succesfully
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Timeout occurred. No data received from server.
Frame sent
Received acknowledgment for sequence number: 3
Frame sent
Received acknowledgment for sequence number: 4
Frame sent
Received acknowledgment for sequence number: 5
Frame sent
Received acknowledgment for sequence number: 6
Frame sent
Received acknowledgment for sequence number: 7

fs_ser.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/wait.h>
#include <errno.h>
#define PORT 8080
#define MAX_PENDING_CONNECTIONS 10
#define MAX_BUFFER_SIZE 1024
void handle_client_request(int client_socket) {
 char buffer[MAX_BUFFER_SIZE];
 ssize_t bytes_received;
 ssize_t bytes_sent;
pid_t pid = getpid();
 bytes_received = recv(client_socket, buffer,
sizeof(buffer), 0);
 if (bytes_received < 0) {
 perror("Error receiving data from client");
 exit(EXIT_FAILURE);
 }
 buffer[bytes_received] = '¥0';
 FILE* file = fopen(buffer, "rb");
 if (file != NULL) {
 while ((bytes_sent = fread(buffer, 1, sizeof(buffer), file))
> 0) {
 if (send(client_socket, buffer, bytes_sent, 0) !=
bytes_sent) {
 perror("Error sending file to client");
 exit(EXIT_FAILURE);
 } }
 fclose(file);
 } else {
const char* message = "File not found.";
 if (send(client_socket, message, strlen(message), 0) < 0)
{
 perror("Error sending message to client");
 exit(EXIT_FAILURE);
 } }
 snprintf(buffer, sizeof(buffer), "%d", pid);
 if (send(client_socket, buffer, strlen(buffer), 0) < 0) {
 perror("Error sending PID to client");
 exit(EXIT_FAILURE);
 }
 close(client_socket);}
int main() {
 int server_socket, client_socket;
 struct sockaddr_in server_addr, client_addr;
 socklen_t client_addr_len = sizeof(client_addr);
 pid_t pid;
```

```
if ((server_socket = socket(AF_INET, SOCK_STREAM, 0))
< 0) {
 perror("Error creating socket");
 exit(EXIT_FAILURE);
 }
 memset(&server_addr, 0, sizeof(server_addr));
 server_addr.sin_family = AF_INET;
 server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
 server_addr.sin_port = htons(PORT);
 if (bind(server_socket, (struct sockaddr*)&server_addr,
sizeof(server_addr)) < 0) {
 perror("Error binding socket");
 exit(EXIT_FAILURE);
 }
 if (listen(server_socket,
MAX_PENDING_CONNECTIONS) < 0) {
 perror("Error listening on socket");
 exit(EXIT_FAILURE);
 }
 printf("Server listening on port %d...¥n", PORT);
 while (1) {
 if ((client_socket = accept(server_socket, (struct
sockaddr*)&client_addr, &client_addr_len)) <
0) {
 perror("Error accepting connection");
 exit(EXIT_FAILURE); }
 pid = fork();
 if (pid < 0) {
 perror("Error forking child process");
 exit(EXIT_FAILURE);
 } else if (pid == 0) {
 close(server_socket);
 handle_client_request(client_socket);
 exit(EXIT_SUCCESS);
 } else {
 close(client_socket);
 waitpid(-1, NULL, WNOHANG);
 } }
 close(server_socket);
 return 0;
}
```

fs_cli.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#define SERVER_IP "127.0.0.1"
#define PORT 8080
#define MAX_BUFFER_SIZE 1024
int main() {
 int client_socket;
 struct sockaddr_in server_addr;
 char filename[MAX_BUFFER_IZE];
 ssize_t bytes_received;
 if ((client_socket = socket(AF_INET, SOCK_STREAM, 0))
< 0) {
 perror("Error creating socket");
 exit(EXIT_FAILURE);
 }
 memset(&server_addr, 0, sizeof(server_addr));
 server_addr.sin_family = AF_INET;
 server_addr.sin_port = htons(PORT);
 server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
 if (connect(client_socket, (struct
sockaddr*)&server_addr, sizeof(server_addr)) < 0) {
 perror("Error connecting to server");
 exit(EXIT_FAILURE); }
 printf("Enter filename: ");
 fgets(filename, sizeof(filename), stdin);
 filename[strcspn(filename, "¥n")] = '¥0';
 if (send(client_socket, filename, strlen(filename), 0) <
0) {
perror("Error sending filename to server");
 exit(EXIT_FAILURE); }
 char buffer[MAX_BUFFER_SIZE];
 while ((bytes_received = recv(client_socket, buffer,
sizeof(buffer), 0)) > 0) {
 fwrite(buffer, 1, bytes_received, stdout); }
 if (bytes_received < 0) {
 perror("Error receiving data from server");
 exit(EXIT_FAILURE);
 }
 close(client_socket);
 return 0;
}
```

Output

fs_ser.c

```
Server listening to port 5016
```

fs_cli.c

```
Enter filename:nx.txt
filenot found
PID:54186
Enter filename:nw.txt
ABCDEFG
PID:34069
```

Stop.c

```
        //elvis
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet recv_packet,ack_packet;
 int server_sock,client_sock,n,j=0;
 struct sockaddr_in serveraddr,clientaddr;
 socklen_t addr_size;
int main(int argc, char **argv)
{
 if(argc != 2)
  {
    printf("Error");
   exit(0);  }
 int port =atoi(argv[1]);
 server_sock=socket(AF_INET,SOCK_DGRAM,0);
 if(server_sock ==-1)
 {
  printf("Server not created\n");
  exit(0); }
 printf("Server is created successfully\n");
 memset(&serveraddr,'\0',sizeof(serveraddr));
 serveraddr.sin_family=AF_INET;
 serveraddr.sin_port=htons(port);
 serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
 bind(server_sock,(struct
sockaddr*)&serveraddr,sizeof(serveraddr));
 addr_size=sizeof(clientaddr);
 printf("Bind to Port number %d\n",port);
  while (1) {
     if
(recvfrom(server_sock,&recv_packet,sizeof(recv_packet),0
,(struct sockaddr*)&clientaddr,&addr_size) < 0) {
        perror("recvfrom failed");
        exit(0);    }
     printf("Frame received %s\n",recv_packet.data);
     printf("Received packet with sequence number:
%d\n", recv_packet.seq_num);
     sleep(2);
     ack_packet.seq_num = recv_packet.seq_num;
     if(j!=3)
     {
      if
(sendto(server_sock,&ack_packet,sizeof(ack_packet),0,(str
uct sockaddr*)&clientaddr,addr_size) < 0) {
         perror("sendto failed");
         exit(EXIT_FAILURE);
       }
       printf("Sent acknowledgment for sequence number:
%d\n", ack_packet.seq_num);
      }
     j++;
     if(recv_packet.flag == 1)
     {
       close(server_sock);
       exit(0);
      }
   }
   return 0;
}
```

Wait.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet send_packet, recv_ack;
int main(int argc, char **argv) {
if(argc != 2)
  {
    printf("Error");
   exit(0);}
 int port = atoi(argv[1]);
 int clientsock,i=0;
 struct sockaddr_in addr;
 fd_set readfds;
 socklen_t addr_size;
 char word[] ="Hello my name is Elvis";
 clientsock=socket(AF_INET,SOCK_DGRAM,0);
 printf("Client is created succesfully\n");
 memset(&addr,'\0',sizeof(addr));
 addr.sin_family=AF_INET;
 addr.sin_port=htons(port);
 addr.sin_addr.s_addr=inet_addr("127.0.0.1");
 int base=0,window_size=3,j=0,flag=0;
 int length=strlen(word);
 addr_size=sizeof(addr);
```

```
  while (1) {
      for( i=base;i<window_size;i++)
      {
        send_packet.data[i]=word[i];
        if(window_size > strlen(word))
        {
          flag=1;
          send_packet.flag=1;
        } }
      send_packet.seq_num=j++;

sendto(clientsock,&send_packet,sizeof(send_packet),0,(str
uct sockaddr*)&addr,addr_size);
      printf("Frame sent\n");
              FD_ZERO(&readfds);
              FD_SET(clientsock, &readfds);
              struct timeval timeout;
              timeout.tv_sec = 3;
              timeout.tv_usec = 0;
      int a = select(clientsock+1, &readfds, NULL, NULL,
&timeout);
              if (a == -1) {
                        perror("select");
                        exit(EXIT_FAILURE);
                        }
                        else if (a == 0) {
                        printf("Timeout occurred. No data
received from server.\n");
                          window_size=3;
                          j=0;
                          base=0;
                          bzero(send_packet.data,1024);
                          length=strlen(word);  }
              else{

recvfrom(clientsock,&recv_ack,sizeof(recv_ack),0,(struct
sockaddr*)&addr,&addr_size);
                  printf("Received acknowledgment for
sequence number: %d\n", recv_ack.seq_num);
                  base=window_size;
                  if(length <=3)
                  {
                    window_size=length;
                  }
                  else
                  {
                     window_size=window_size+3;
                              length=length-3;
                  }                    }

                        if(flag==1)
                        {
                          close(clientsock);
                          exit(0);
                        }
    }
    return 0;
}
```

output

stop.c

```
Server is created successfully
Bind to Port number 5005
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Sent acknowledgment for sequence number: 3
Frame received Hello my name i
Received packet with sequence number: 4
Sent acknowledgment for sequence number: 4
Frame received Hello my name is E
Received packet with sequence number: 5
Sent acknowledgment for sequence number: 5
Frame received Hello my name is Elvi
Received packet with sequence number: 6
Sent acknowledgment for sequence number: 6
Frame received Hello my name is Elvis
Received packet with sequence number: 7
Sent acknowledgment for sequence number: 7
```

wait.c

```
Client is created succesfully
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
```

```
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Timeout occurred. No data received from server.
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Received acknowledgment for sequence number: 3
Frame sent
Received acknowledgment for sequence number: 4
Frame sent
Received acknowledgment for sequence number: 5
Frame sent
Received acknowledgment for sequence number: 6
Frame sent
Received acknowledgment for sequence number: 7
```

leaky

```
#include <stdio.h>
#include <unistd.h>
int main() {
   int n, incoming, outgoing, store = 0, bucketsize;
   printf("Enter the bucket size:- ");
   scanf("%d", &bucketsize);
   printf("Enter the outgoing rate:- ");
   scanf("%d", &outgoing);
   printf("Enter the number of inputs:- ");
   scanf("%d", &n);
   while (n > 0) {
      printf("\nEnter the incoming size:- ");
      scanf("%d", &incoming);
      printf("\nIncoming size is %d\n", incoming);
      if (incoming <= (bucketsize - store)) {
         store += incoming;
         printf("Bucket buffer size is %d out of %d\n",
store, bucketsize);
      } else {
         printf("packet loss = %d\n", incoming -
(bucketsize - store));
         store = bucketsize;
         printf("Buffer is full\n");
      }
      store -= outgoing;
      printf("After outgoing %d packets are left out of %d
in the buffer\n", store, bucketsize);
      n--;
   }
   return 0;
}
```

Select.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet recv_packet,ack_packet;
 int server_sock,client_sock,n,j=0;
 struct sockaddr_in serveraddr,clientaddr;
 socklen_t addr_size;
int main(int argc, char **argv)
{
 if(argc != 2)
  {
    printf("Error");
    exit(0);
  }
 int port =atoi(argv[1]);
 server_sock=socket(AF_INET,SOCK_DGRAM,0);
 if(server_sock ==-1)
 {
  printf("Server not created\n");
  exit(0);
 }
 printf("Server is created successfully\n");
 memset(&serveraddr,'\0',sizeof(serveraddr));
 serveraddr.sin_family=AF_INET;
 serveraddr.sin_port=htons(port);
 serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
 bind(server_sock,(struct
sockaddr*)&serveraddr,sizeof(serveraddr));
 addr_size=sizeof(clientaddr);
 printf("Bind to Port number %d\n",port);
  while (1) {
     if
(recvfrom(server_sock,&recv_packet,sizeof(recv_packet
),0,(struct sockaddr*)&clientaddr,&addr_size) < 0) {
         perror("recvfrom failed");
         exit(0);
```

```
      }
      printf("Frame received %s\n",recv_packet.data);
      printf("Received packet with sequence number:
%d\n", recv_packet.seq_num);
      sleep(2);
      ack_packet.seq_num = recv_packet.seq_num;
      if(j!=3)
      {
 if
(sendto(server_sock,&ack_packet,sizeof(ack_packet),0,(
struct sockaddr*)&clientaddr,addr_size) < 0) {
          perror("sendto failed");
          exit(0);
      }
      printf("Sent acknowledgment for sequence
number: %d\n", ack_packet.seq_num);
      }
     j++;
     if(recv_packet.flag == 1)
     {
       close(server_sock);
       exit(0);
     }
   }
   return 0;
}
```

repeat.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag,i;
} packet;
packet send_packet, recv_ack;
int main(int argc, char **argv) {
if(argc != 2)
  {
    printf("Error");
    exit(0);}
 int port = atoi(argv[1]);
 int clientsock,i=0;
 struct sockaddr_in addr;
 fd_set readfds;
 socklen_t addr_size;
 char word[] ="Hello my name is Elvis";
 clientsock=socket(AF_INET,SOCK_DGRAM,0);
 printf("Client is created succesfully\n");
 memset(&addr,'\0',sizeof(addr));
 addr.sin_family=AF_INET;
 addr.sin_port=htons(port);
 addr.sin_addr.s_addr=inet_addr("127.0.0.1");
 int base=0,window_size=3,j=0,flag=0;
 int length=strlen(word);
 addr_size=sizeof(addr);
  while (1) {
      for(i=base;i<window_size;i++)
       {
         send_packet.data[i]=word[i];
         if(window_size > strlen(word)){
           flag=1;
           send_packet.flag=1;
         } }
      send_packet.seq_num=j++;

sendto(clientsock,&send_packet,sizeof(send_packet),0,(
struct sockaddr*)&addr,addr_size);
      printf("Frame sent\n");
            FD_ZERO(&readfds);
            FD_SET(clientsock, &readfds);
            struct timeval timeout;
            timeout.tv_sec = 3;
            timeout.tv_usec = 0;
      int a = select(clientsock+1, &readfds, NULL, NULL,
&timeout);
            if (a == -1) {
                  perror("select");
                  exit(0);     }
                  else if (a == 0) {
                   printf("Timeout occurred. No data
received from server.\n");

window_size=3*(send_packet.seq_num+1);
                   j=send_packet.seq_num;
                   base=window_size-3;
                   length=length+3;}
            else{

recvfrom(clientsock,&recv_ack,sizeof(recv_ack),0,(struc
t sockaddr*)&addr,&addr_size);
                printf("Received acknowledgment for
sequence number: %d\n", recv_ack.seq_num);
                base=window_size;
                if(length <=3)
                 {
```

```
                window_size=length;
               }
               else{
                  window_size=window_size+3;
                        length=length-3;   }
}
                      if(flag==1)
                       {
                        close(clientsock);
                        exit(0);
                       }   }
   return 0;
}
```

Output

Select.c

```
Server is created successfully
Bind to Port number 6004
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Frame received Hello my nam
Received packet with sequence number: 3
Sent acknowledgment for sequence number: 3
Frame received Hello my name i
Received packet with sequence number: 4
Sent acknowledgment for sequence number: 4
Frame received Hello my name is E
Received packet with sequence number: 5
Sent acknowledgment for sequence number: 5
Frame received Hello my name is Elvi
Received packet with sequence number: 6
Sent acknowledgment for sequence number: 6
Frame received Hello my name is Elvis
Received packet with sequence number: 7
Sent acknowledgment for sequence number: 7
```

Repeat.c

```
Client is created succesfully
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Timeout occurred. No data received from server.
Frame sent
Received acknowledgment for sequence number: 3
Frame sent
Received acknowledgment for sequence number: 4
Frame sent
Received acknowledgment for sequence number: 5
Frame sent
Received acknowledgment for sequence number: 6
Frame sent
Received acknowledgment for sequence number: 7
```

fs_ser.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/wait.h>
#include <errno.h>
#define PORT 8080
#define MAX_PENDING_CONNECTIONS 10
#define MAX_BUFFER_SIZE 1024
void handle_client_request(int client_socket) {
 char buffer[MAX_BUFFER_SIZE];
 ssize_t bytes_received;
 ssize_t bytes_sent;
pid_t pid = getpid();
 bytes_received = recv(client_socket, buffer,
sizeof(buffer), 0);
 if (bytes_received < 0) {
 perror("Error receiving data from client");
 exit(EXIT_FAILURE);
 }
 buffer[bytes_received] = '¥0';
 FILE* file = fopen(buffer, "rb");
 if (file != NULL) {
 while ((bytes_sent = fread(buffer, 1, sizeof(buffer), file))
> 0) {
 if (send(client_socket, buffer, bytes_sent, 0) !=
bytes_sent) {
 perror("Error sending file to client");
 exit(EXIT_FAILURE);
 } }
 fclose(file);
 } else {
```

```
const char* message = "File not found.";
 if (send(client_socket, message, strlen(message), 0) < 0)
{
 perror("Error sending message to client");
 exit(EXIT_FAILURE);
 } }
 snprintf(buffer, sizeof(buffer), "%d", pid);
 if (send(client_socket, buffer, strlen(buffer), 0) < 0) {
 perror("Error sending PID to client");
 exit(EXIT_FAILURE);
 }
 close(client_socket);}
int main() {
 int server_socket, client_socket;
 struct sockaddr_in server_addr, client_addr;
 socklen_t client_addr_len = sizeof(client_addr);
 pid_t pid;
if ((server_socket = socket(AF_INET, SOCK_STREAM, 0))
< 0) {
 perror("Error creating socket");
 exit(EXIT_FAILURE);
 }
 memset(&server_addr, 0, sizeof(server_addr));
 server_addr.sin_family = AF_INET;
 server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
 server_addr.sin_port = htons(PORT);
 if (bind(server_socket, (struct sockaddr*)&server_addr,
sizeof(server_addr)) < 0) {
 perror("Error binding socket");
 exit(EXIT_FAILURE);
 }
 if (listen(server_socket,
MAX_PENDING_CONNECTIONS) < 0) {
 perror("Error listening on socket");
 exit(EXIT_FAILURE);
 }
 printf("Server listening on port %d...¥n", PORT);
 while (1) {
 if ((client_socket = accept(server_socket, (struct
sockaddr*)&client_addr, &client_addr_len)) <
0) {
 perror("Error accepting connection");
 exit(EXIT_FAILURE);
 }
 pid = fork();
 if (pid < 0) {
 perror("Error forking child process");
 exit(EXIT_FAILURE);
 } else if (pid == 0) {
 close(server_socket);
 handle_client_request(client_socket);
 exit(EXIT_SUCCESS);
 } else {
 close(client_socket);
 waitpid(-1, NULL, WNOHANG);
 } }
 close(server_socket);
 return 0;
}
```

fs_cli.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#define SERVER_IP "127.0.0.1"
#define PORT 8080
#define MAX_BUFFER_SIZE 1024
int main() {
 int client_socket;
 struct sockaddr_in server_addr;
 char filename[MAX_BUFFER_IZE];
 ssize_t bytes_received;
 if ((client_socket = socket(AF_INET, SOCK_STREAM, 0))
< 0) {
 perror("Error creating socket");
 exit(EXIT_FAILURE);
 }
 memset(&server_addr, 0, sizeof(server_addr));
 server_addr.sin_family = AF_INET;
 server_addr.sin_port = htons(PORT);
 server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
 if (connect(client_socket, (struct
sockaddr*)&server_addr, sizeof(server_addr)) < 0) {
 perror("Error connecting to server");
 exit(EXIT_FAILURE); }
 printf("Enter filename: ");
 fgets(filename, sizeof(filename), stdin);
 filename[strcspn(filename, "¥n")] = '¥0';
 if (send(client_socket, filename, strlen(filename), 0) <
0) {
perror("Error sending filename to server");
 exit(EXIT_FAILURE); }
 char buffer[MAX_BUFFER_SIZE];
 while ((bytes_received = recv(client_socket, buffer,
sizeof(buffer), 0)) > 0) {
 fwrite(buffer, 1, bytes_received, stdout); }
```

```
 if (bytes_received < 0) {
 perror("Error receiving data from server");
 exit(EXIT_FAILURE);
 }
 close(client_socket);
 return 0;
}
```

Output

fs_ser.c

Server listening to port 5016

fs_cli.c

Enter filename:ab.txt
filenot found
PID:54176
Enter filename:abc.txt
elvis
PID:34079

Output

Enter the bucket size:- 10
Enter the outgoing rate:- 4
Enter the number of inputs:- 3
Enter the incoming size:- 5
Incoming size is 5
Bucket buffer size is 5 out of 10
After outgoing 1 packets are left out of 10 in the buffer
Enter the incoming size:- 6
Incoming size is 6
Bucket buffer size is 7 out of 10
After outgoing 3 packets are left out of 10 in the buffer
Enter the incoming size:- 4
Incoming size is 4
Bucket buffer size is 7 out of 10
After outgoing 3 packets are left out of 10 in the buffer

//sayanth

Output

Enter the bucket size:- 10
Enter the outgoing rate:- 4
Enter the number of inputs:- 3
Enter the incoming size:- 5
Incoming size is 5
Bucket buffer size is 5 out of 10
After outgoing 1 packets are left out of 10 in the buffer
Enter the incoming size:- 6
Incoming size is 6
Bucket buffer size is 7 out of 10
After outgoing 3 packets are left out of 10 in the buffer
Enter the incoming size:- 4
Incoming size is 4
Bucket buffer size is 7 out of 10
After outgoing 3 packets are left out of 10 in the buffer

Output

fs_ser.c

Server listening to port 5018

fs_cli.c

Enter filename:ss.txt
filenot found
PID:54176
Enter filename:sayanth.txt
rtghjsj
PID:34039

fs_ser.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/wait.h>
#include <errno.h>
#define PORT 8080
#define MAX_PENDING_CONNECTIONS 10
#define MAX_BUFFER_SIZE 1024
void handle_client_request(int client_socket) {
 char buffer[MAX_BUFFER_SIZE];
 ssize_t bytes_received;
 ssize_t bytes_sent;
pid_t pid = getpid();
 bytes_received = recv(client_socket, buffer,
sizeof(buffer), 0);
 if (bytes_received < 0) {
 perror("Error receiving data from client");
 exit(EXIT_FAILURE);
 }
 buffer[bytes_received] = '¥0';
 FILE* file = fopen(buffer, "rb");
 if (file != NULL) {
 while ((bytes_sent = fread(buffer, 1, sizeof(buffer), file))
> 0) {
 if (send(client_socket, buffer, bytes_sent, 0) !=
bytes_sent) {
 perror("Error sending file to client");
 exit(EXIT_FAILURE);
 } }
```

```
 fclose(file);
 } else {
const char* message = "File not found.";
 if (send(client_socket, message, strlen(message), 0) < 0)
{
 perror("Error sending message to client");
 exit(EXIT_FAILURE);
 } }
 snprintf(buffer, sizeof(buffer), "%d", pid);
 if (send(client_socket, buffer, strlen(buffer), 0) < 0) {
 perror("Error sending PID to client");
 exit(EXIT_FAILURE);
 }
 close(client_socket);}
int main() {
 int server_socket, client_socket;
 struct sockaddr_in server_addr, client_addr;
 socklen_t client_addr_len = sizeof(client_addr);
 pid_t pid;
if ((server_socket = socket(AF_INET, SOCK_STREAM, 0))
< 0) {
 perror("Error creating socket");
 exit(EXIT_FAILURE);
 }
 memset(&server_addr, 0, sizeof(server_addr));
 server_addr.sin_family = AF_INET;
 server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
 server_addr.sin_port = htons(PORT);
 if (bind(server_socket, (struct sockaddr*)&server_addr,
sizeof(server_addr)) < 0) {
 perror("Error binding socket");
 exit(EXIT_FAILURE);
 }
 if (listen(server_socket,
MAX_PENDING_CONNECTIONS) < 0) {
 perror("Error listening on socket");
 exit(EXIT_FAILURE);
 }
 printf("Server listening on port %d...¥n", PORT);
 while (1) {
 if ((client_socket = accept(server_socket, (struct
sockaddr*)&client_addr, &client_addr_len)) <
0) {
 perror("Error accepting connection");
 exit(EXIT_FAILURE);
 }
 pid = fork();
 if (pid < 0) {
 perror("Error forking child process");
 exit(EXIT_FAILURE);
 } else if (pid == 0) {
 close(server_socket);
 handle_client_request(client_socket);
 exit(EXIT_SUCCESS);
 } else {
 close(client_socket);
 waitpid(-1, NULL, WNOHANG);
 } }
 close(server_socket);
 return 0;
}
```

fs_cli.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#define SERVER_IP "127.0.0.1"
#define PORT 8080
#define MAX_BUFFER_SIZE 1024
int main() {
 int client_socket;
 struct sockaddr_in server_addr;
 char filename[MAX_BUFFER_IZE];
 ssize_t bytes_received;
 if ((client_socket = socket(AF_INET, SOCK_STREAM, 0))
< 0) {
 perror("Error creating socket");
 exit(EXIT_FAILURE);
 }
 memset(&server_addr, 0, sizeof(server_addr));
 server_addr.sin_family = AF_INET;
 server_addr.sin_port = htons(PORT);
 server_addr.sin_addr.s_addr = htonl(INADDR_ANY);
 if (connect(client_socket, (struct
sockaddr*)&server_addr, sizeof(server_addr)) < 0) {
 perror("Error connecting to server");
 exit(EXIT_FAILURE); }
 printf("Enter filename: ");
 fgets(filename, sizeof(filename), stdin);
 filename[strcspn(filename, "¥n")] = '¥0';
 if (send(client_socket, filename, strlen(filename), 0) <
0) {
perror("Error sending filename to server");
 exit(EXIT_FAILURE); }
 char buffer[MAX_BUFFER_SIZE];
```

```
 while ((bytes_received = recv(client_socket, buffer,
sizeof(buffer), 0)) > 0) {
 fwrite(buffer, 1, bytes_received, stdout); }
 if (bytes_received < 0) {
 perror("Error receiving data from server");
 exit(EXIT_FAILURE);
 }
 close(client_socket);
 return 0;
}
```

leaky

```
#include <stdio.h>
#include <unistd.h>
int main() {
   int n, incoming, outgoing, store = 0, bucketsize;
   printf("Enter the bucket size:- ");
   scanf("%d", &bucketsize);
   printf("Enter the outgoing rate:- ");
   scanf("%d", &outgoing);
   printf("Enter the number of inputs:- ");
   scanf("%d", &n);
   while (n > 0) {
      printf("\nEnter the incoming size:- ");
      scanf("%d", &incoming);
      printf("\nIncoming size is %d\n", incoming);
      if (incoming <= (bucketsize - store)) {
         store += incoming;
         printf("Bucket buffer size is %d out of %d\n",
store, bucketsize);
      } else {
         printf("packet loss = %d\n", incoming -
(bucketsize - store));
         store = bucketsize;
         printf("Buffer is full\n");
      }
      store -= outgoing;
      printf("After outgoing %d packets are left out of %d
in the buffer\n", store, bucketsize);
      n--;
   }
   return 0;
}
```

Stop.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
    int seq_num;
    char data[1024];
    int flag;
} packet;
packet recv_packet,ack_packet;
  int server_sock,client_sock,n,j=0;
  struct sockaddr_in serveraddr,clientaddr;
  socklen_t addr_size;
int main(int argc, char **argv)
{
  if(argc != 2)
   {
     printf("Error");
     exit(0);  }
  int port =atoi(argv[1]);
  server_sock=socket(AF_INET,SOCK_DGRAM,0);
  if(server_sock ==-1)
  {
    printf("Server not created\n");
    exit(0); }
  printf("Server is created successfully\n");
  memset(&serveraddr,'\0',sizeof(serveraddr));
  serveraddr.sin_family=AF_INET;
  serveraddr.sin_port=htons(port);
  serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
  bind(server_sock,(struct
sockaddr*)&serveraddr,sizeof(serveraddr));
  addr_size=sizeof(clientaddr);
  printf("Bind to Port number %d\n",port);
    while (1) {
        if
(recvfrom(server_sock,&recv_packet,sizeof(recv_packet),0
,(struct sockaddr*)&clientaddr,&addr_size) < 0) {
           perror("recvfrom failed");
           exit(0);    }
        printf("Frame received %s\n",recv_packet.data);
        printf("Received packet with sequence number:
%d\n", recv_packet.seq_num);
        sleep(2);
        ack_packet.seq_num = recv_packet.seq_num;
        if(j!=3)
        {
        if
(sendto(server_sock,&ack_packet,sizeof(ack_packet),0,(str
uct sockaddr*)&clientaddr,addr_size) < 0) {
           perror("sendto failed");
           exit(EXIT_FAILURE);
        }
        printf("Sent acknowledgment for sequence number:
%d\n", ack_packet.seq_num);
        }
       j++;
```

```
    if(recv_packet.flag == 1)
     {
      close(server_sock);
      exit(0);
     }
   }
   return 0;
}
```

Wait.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet send_packet, recv_ack;
int main(int argc, char **argv) {
if(argc != 2)
  {
    printf("Error");
    exit(0);}
 int port = atoi(argv[1]);
 int clientsock,i=0;
 struct sockaddr_in addr;
 fd_set readfds;
 socklen_t addr_size;
 char word[] ="Hello my name is Sayanth";
 clientsock=socket(AF_INET,SOCK_DGRAM,0);
 printf("Client is created succesfully\n");
 memset(&addr,'\0',sizeof(addr));
 addr.sin_family=AF_INET;
 addr.sin_port=htons(port);
 addr.sin_addr.s_addr=inet_addr("127.0.0.1");
 int base=0,window_size=3,j=0,flag=0;
 int length=strlen(word);
 addr_size=sizeof(addr);
  while (1) {
      for( i=base;i<window_size;i++)
      {
        send_packet.data[i]=word[i];
        if(window_size > strlen(word))
        {
          flag=1;
          send_packet.flag=1;
        } }
      send_packet.seq_num=j++;
sendto(clientsock,&send_packet,sizeof(send_packet),0,(str
uct sockaddr*)&addr,addr_size);
      printf("Frame sent\n");
            FD_ZERO(&readfds);
            FD_SET(clientsock, &readfds);
            struct timeval timeout;
            timeout.tv_sec = 3;
            timeout.tv_usec = 0;
      int a = select(clientsock+1, &readfds, NULL, NULL,
&timeout);
           if (a == -1) {
                        perror("select");
                        exit(EXIT_FAILURE);
                        }
                        else if (a == 0) {
                        printf("Timeout occurred. No data
received from server.\n");
                         window_size=3;
                         j=0;
                         base=0;
                         bzero(send_packet.data,1024);
                         length=strlen(word);  }
           else{

recvfrom(clientsock,&recv_ack,sizeof(recv_ack),0,(struct
sockaddr*)&addr,&addr_size);
               printf("Received acknowledgment for
sequence number: %d\n", recv_ack.seq_num);
               base=window_size;
               if(length <=3)
               {
                 window_size=length;
               }
               else
               {
                  window_size=window_size+3;
                        length=length-3;
               }                  }

                     if(flag==1)
                     {
                       close(clientsock);
                       exit(0);
                     }
   }
   return 0;
}
```

output

stop.c

```
Server is created successfully
Bind to Port number 5010
```

```
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Sent acknowledgment for sequence number: 3
Frame received Hello my name i
Received packet with sequence number: 4
Sent acknowledgment for sequence number: 4
Frame received Hello my name is S
Received packet with sequence number: 5
Sent acknowledgment for sequence number: 5
Frame received Hello my name is Saya
Received packet with sequence number: 6
Sent acknowledgment for sequence number: 6
Frame received Hello my name is Sayanth
Received packet with sequence number: 7
Sent acknowledgment for sequence number: 7
```

wait.c

```
Client is created succesfully
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Timeout occurred. No data received from server.
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Received acknowledgment for sequence number: 3
Frame sent
Received acknowledgment for sequence number: 4
Frame sent
Received acknowledgment for sequence number: 5
Frame sent
Received acknowledgment for sequence number: 6
Frame sent
Received acknowledgment for sequence number: 7
```

Select.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag;
} packet;
packet recv_packet,ack_packet;
 int server_sock,client_sock,n,j=0;
 struct sockaddr_in serveraddr,clientaddr;
 socklen_t addr_size;
int main(int argc, char **argv)
{
 if(argc != 2)
  {
    printf("Error");
    exit(0);
  }
 int port =atoi(argv[1]);
 server_sock=socket(AF_INET,SOCK_DGRAM,0);
 if(server_sock ==-1)
 {
   printf("Server not created\n");
   exit(0);
 }
 printf("Server is created successfully\n");
 memset(&serveraddr,'\0',sizeof(serveraddr));
 serveraddr.sin_family=AF_INET;
 serveraddr.sin_port=htons(port);
 serveraddr.sin_addr.s_addr=inet_addr("127.0.0.1");
 bind(server_sock,(struct
sockaddr*)&serveraddr,sizeof(serveraddr));
 addr_size=sizeof(clientaddr);
```

```
  printf("Bind to Port number %d\n",port);
   while (1) {
      if
(recvfrom(server_sock,&recv_packet,sizeof(recv_packet
),0,(struct sockaddr*)&clientaddr,&addr_size) < 0) {
         perror("recvfrom failed");
         exit(0);
      }
      printf("Frame received %s\n",recv_packet.data);
      printf("Received packet with sequence number:
%d\n", recv_packet.seq_num);
      sleep(2);
      ack_packet.seq_num = recv_packet.seq_num;
      if(j!=3)
      {
 if
(sendto(server_sock,&ack_packet,sizeof(ack_packet),0,(
struct sockaddr*)&clientaddr,addr_size) < 0) {
         perror("sendto failed");
         exit(0);
      }
      printf("Sent acknowledgment for sequence
number: %d\n", ack_packet.seq_num);
      }
     j++;
     if(recv_packet.flag == 1)
      {
       close(server_sock);
       exit(0);
      }
   }
   return 0;
}
```

repeat.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <signal.h>
#include <sys/time.h>
typedef struct {
   int seq_num;
   char data[1024];
   int flag,i;
} packet;
packet send_packet, recv_ack;
int main(int argc, char **argv) {
if(argc != 2)
  {
   printf("Error");
   exit(0);}
 int port = atoi(argv[1]);
 int clientsock,i=0;
 struct sockaddr_in addr;
 fd_set readfds;
 socklen_t addr_size;
 char word[] ="Hello my name is Sayanth";
 clientsock=socket(AF_INET,SOCK_DGRAM,0);
 printf("Client is created succesfully\n");
 memset(&addr,'\0',sizeof(addr));
 addr.sin_family=AF_INET;
 addr.sin_port=htons(port);
 addr.sin_addr.s_addr=inet_addr("127.0.0.1");
 int base=0,window_size=3,j=0,flag=0;
 int length=strlen(word);
 addr_size=sizeof(addr);
  while (1) {
      for(i=base;i<window_size;i++)
       {
        send_packet.data[i]=word[i];
        if(window_size > strlen(word)){
          flag=1;
          send_packet.flag=1;
        } }
      send_packet.seq_num=j++;

sendto(clientsock,&send_packet,sizeof(send_packet),0,(
struct sockaddr*)&addr,addr_size);
      printf("Frame sent\n");
            FD_ZERO(&readfds);
            FD_SET(clientsock, &readfds);
            struct timeval timeout;
            timeout.tv_sec = 3;
            timeout.tv_usec = 0;
      int a = select(clientsock+1, &readfds, NULL, NULL,
&timeout);
            if (a == -1) {
                  perror("select");
                  exit(0);     }
                  else if (a == 0) {
                   printf("Timeout occurred. No data
received from server.\n");

window_size=3*(send_packet.seq_num+1);
                   j=send_packet.seq_num;
                   base=window_size-3;
                   length=length+3;}
            else{
```

```
recvfrom(clientsock,&recv_ack,sizeof(recv_ack),0,(struc
t sockaddr*)&addr,&addr_size);
                printf("Received acknowledgment for
sequence number: %d\n", recv_ack.seq_num);
                base=window_size;
                if(length <=3)
                {
                  window_size=length;
                }
                else{
                   window_size=window_size+3;
                            length=length-3;   }
}
                        if(flag==1)
                        {
                          close(clientsock);
                          exit(0);
                        }   }
   return 0;
}
```

Output

Select.c

```
Server is created successfully
Bind to Port number 6003
Frame received Hel
Received packet with sequence number: 0
Sent acknowledgment for sequence number: 0
Frame received Hello
Received packet with sequence number: 1
Sent acknowledgment for sequence number: 1
Frame received Hello my
Received packet with sequence number: 2
Sent acknowledgment for sequence number: 2
Frame received Hello my nam
Received packet with sequence number: 3
Frame received Hello my nam
Received packet with sequence number: 3
Sent acknowledgment for sequence number: 3
Frame received Hello my name i
Received packet with sequence number: 4
Sent acknowledgment for sequence number: 4
Frame received Hello my name is S
Received packet with sequence number: 5
Sent acknowledgment for sequence number: 5
Frame received Hello my name is Saya
Received packet with sequence number: 6
Sent acknowledgment for sequence number: 6
Frame received Hello my name is Sayanth
Received packet with sequence number: 7
Sent acknowledgment for sequence number: 7
Frame received Hello my name is Sayanth
Received packet with sequence number: 8
Sent acknowledgment for sequence number: 8
```

Repeat.c

```
Client is created succesfully
Frame sent
Received acknowledgment for sequence number: 0
Frame sent
Received acknowledgment for sequence number: 1
Frame sent
Received acknowledgment for sequence number: 2
Frame sent
Timeout occurred. No data received from server.
Frame sent
Received acknowledgment for sequence number: 3
Frame sent
Received acknowledgment for sequence number: 4
Frame sent
Received acknowledgment for sequence number: 5
Frame sent
Received acknowledgment for sequence number: 6
Frame sent
Received acknowledgment for sequence number: 7
Frame sent
Received acknowledgment for sequence number: 8
```