```c
#include <sys/socket.h>
#include <sys/un.h>
#include <unistd.h>
#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>


#define UNIX_PATH_MAX 108
int main() {
   int fd = -1;
   if((fd = socket(AF_UNIX, SOCK_DGRAM, 0 )) == -1) {
       perror("Error craeting socket");
   }

   struct sockaddr_un addr;
   addr.sun_family = AF_UNIX;
   addr.sun_path[0] = '\0';
   if(bind(fd,(struct sockaddr *)&addr, sizeof(struct
sockaddr)) == -1) {
       perror("Error binding");
   }

   char buf[20];
   if(read(fd, buf, 20) == -1)
       perror("Error receiving message");
   printf("Message from client \"%s\"\n",buf);

   shutdown(fd, SHUT_RDWR);
   close(fd);
   return 0;
}
```

```c
#include <sys/socket.h>
#include <sys/un.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>
```

```c
#include <errno.h>

#define UNIX_PATH_MAX 108

int main() {
   int fd = -1;
   if((fd = socket(AF_UNIX, SOCK_DGRAM, 0 )) == -1) {
       perror("Error craeting socket");
   }

   struct sockaddr_un addr;
   addr.sun_family = AF_UNIX;
   strcpy(addr.sun_path,"\0");

   if(connect(fd, (struct sockaddr *)&addr,
sizeof(struct sockaddr)) == -1)
       perror("Error connecting to server");

   char buff[20];
   int to_send = sprintf(buff, "HELLO From: %zu",
getpid());

   if(write(fd, buff, to_send+1) == -1) {
       perror("Error sending msg to server");
   }

   shutdown(fd, SHUT_RDWR);


   return 0;
}
```

Domena internetowa
Serwer

```c
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <unistd.h>
#include <errno.h>
```

```c
#include <fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/un.h>
#include <netdb.h>

#define PORT 7777

int main() {
    int fd = -1;
    if ((fd = socket(AF_INET, SOCK_DGRAM, 0)) == -1) {
        printf("Error creating socket\n");
    }

    struct sockaddr_in addr;
    addr.sin_family = AF_INET;
    addr.sin_port = htons(PORT);
    addr.sin_addr.s_addr = inet_addr("0.0.0.0");
    addr.sin_zero[0] = '\0';
    if (bind(fd, (struct sockaddr*)&addr, sizeof(struct sockaddr)) ==
        -1) {
        printf("Error binding\n");
    }

    char buf[64];
    if (read(fd, buf, 64) == -1) {
        printf("Error receiving message\n");
    }
    printf("Message from client: \"%s\"\n", buf);
    shutdown(fd, SHUT_RDWR);
    close(fd);

    return 0;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <unistd.h>
#include <errno.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
```

```
#include <sys/types.h>
#include <sys/un.h>

#define PORT 7777

int main() {
   int fd = -1;
   if ((fd = socket(AF_INET, SOCK_DGRAM, 0)) == -1) {
       printf("Error creating socket\n");
   }

   struct sockaddr_in addr;
   addr.sin_family = AF_INET;
   addr.sin_port = htons(PORT);
   addr.sin_addr.s_addr = inet_addr("0.0.0.0");
   addr.sin_zero[0] = '\0';

   if (connect(fd, (struct sockaddr*)&addr,
sizeof(struct sockaddr)) ==
       -1) {
       printf("Error connecting\n");
   }

   char buf[64];
   sprintf(buf, "My pid is %d", getpid());
   if (write(fd, buf, 64) == -1) {
       printf("Error sending message\n");
   }
   close(fd);

   return 0;
}
```

Tryb polaczeniowy, domena internetowa

Serwer:

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <unistd.h>
#include <errno.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/un.h>
#include <netdb.h>

#define PORT 7777

int main() {
	int fd = -1;
```

```
     if ((fd = socket(AF_INET, SOCK_STREAM, 0)) == -1)
{
         printf("Error creating socket\n");
     }

     struct sockaddr_in addr;
     addr.sin_family = AF_INET;
     addr.sin_port = htons(PORT);
     addr.sin_addr.s_addr = inet_addr("0.0.0.0");
     addr.sin_zero[0] = '\0';
     if (bind(fd, (struct sockaddr*)&addr,
sizeof(struct sockaddr)) ==
-1) {
         printf("Error binding\n");
     }

     if (listen(fd, 3) == -1) {
         printf("Error listening\n");
     }

     struct sockaddr_in cl_addr;
     int cl_fd = -1;
     socklen_t addr_size = sizeof(cl_addr);

     if ((cl_fd = accept(fd, (struct
sockaddr*)&cl_addr, &addr_size)) ==
-1) {
         printf("Error accepting\n");
     }

     char buf[64];
     if (read(cl_fd, buf, 64) == -1) {
         printf("Error receiving message\n");
     }
     printf("Message from client: \"%s\"\n", buf);
     shutdown(fd, SHUT_RDWR);
     close(fd);

     return 0;
}
```

Klient:

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <unistd.h>
#include <errno.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/types.h>
```

```
#include <sys/un.h>

#define PORT 7777

int main() {
     int fd = -1;
     if ((fd = socket(AF_INET, SOCK_STREAM, 0)) == -1)
{
          printf("Error creating socket\n");
     }

     struct sockaddr_in addr;
     addr.sin_family = AF_INET;
     addr.sin_port = htons(PORT);
     addr.sin_addr.s_addr = inet_addr("0.0.0.0");
     addr.sin_zero[0] = '\0';

     if (connect(fd, (struct sockaddr*)&addr,
sizeof(struct sockaddr)) ==
-1) {
          printf("Error connecting\n");
     }

     char buf[64];
     sprintf(buf, "My pid is %d", getpid());
     if (write(fd, buf, 64) == -1) {
         printf("Error sending message\n");
     }
     close(fd);

     return 0;
}
```

Tryb datagramowy, domena unixowa wielu klientow

Wielu klientow

serwer:

```
#include <sys/socket.h>
#include <sys/un.h>
#include <unistd.h>
#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>


#define UNIX_PATH_MAX 108
int main() {
        int fd = -1;
        if((fd = socket(AF_UNIX, SOCK_DGRAM, 0 )) == -1) {
```

```
        perror("Error craeting socket");
        }

        struct sockaddr_un addr;
        addr.sun_family = AF_UNIX;
        addr.sun_path[0] = '\0';
        if(bind(fd,(struct sockaddr *)&addr, sizeof(struct sockaddr)) == -1) {
        perror("Error binding");
        }

        char buf[20];
        while(1){
        if(read(fd, buf, 20) == -1)
        perror("Error receiving message");
        printf("Message from client \"%s\"\n",buf);
        if(buf[0] == 'e') break;
        }

        shutdown(fd, SHUT_RDWR);
        close(fd);
return 0;
}
```

klient:

```
#include <sys/socket.h>
#include <sys/un.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <errno.h>

#define UNIX_PATH_MAX 108

int main() {
        int fd = -1;
        if((fd = socket(AF_UNIX, SOCK_DGRAM, 0 )) == -1) {
        perror("Error craeting socket");
        }

        struct sockaddr_un addr;
        addr.sun_family = AF_UNIX;
        strcpy(addr.sun_path,"\0");

        if(connect(fd, (struct sockaddr *)&addr, sizeof(struct sockaddr))
== -1)
        perror("Error connecting to server");
        char buff[20];
        int to_send = sprintf(buff, "HELLO From: %zu", getpid());

        if(write(fd, buff, to_send+1) == -1) {
        perror("Error sending msg to server");
        }

        shutdown(fd, SHUT_RDWR);


        return 0;
}
```

Wersja streamowa, domena internetowa, wielu klientow, obsluga polaczen przez proces potomny

Serwer

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <unistd.h>
#include <errno.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/un.h>
#include <netdb.h>

#define PORT 7777

int main() {
    int fd = -1;
    if ((fd = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
        printf("Error creating socket\n");
    }

    struct sockaddr_in addr;
    addr.sin_family = AF_INET;
    addr.sin_port = htons(PORT);
    addr.sin_addr.s_addr = inet_addr("0.0.0.0");
    addr.sin_zero[0] = '\0';
    if (bind(fd, (struct sockaddr*)&addr, sizeof(struct sockaddr)) ==
-1) {
        printf("Error binding\n");
    }

    if (listen(fd, 3) == -1) {
        printf("Error listening\n");
    }

    struct sockaddr_in cl_addr;
    int cl_fd = -1;
    socklen_t addr_size = sizeof(cl_addr);

    while(1) {
     if ((cl_fd = accept(fd, (struct sockaddr*)&cl_addr, &addr_size))
== -1) {
          printf("Error accepting\n");
     }

     int pid = fork();
     if(pid == 0) {
        while(1) {
          char buf[64];
          int bytes_read = read(cl_fd, buf, 64);
          if (bytes_read == -1) {
               printf("Error receiving message\n");
               return 1;
          }

          if (bytes_read == 0) {
               printf("Client disconnected\n");
               return 0;
          }
          printf("Message from client: \"%s\"\n", buf);
        }
     }
    }

    shutdown(fd, SHUT_RDWR);
    close(fd);

    return 0;
}
```